Core Concepts

1 Abstract

2 Table of Contents

2.1 Preface ......................................................... 5
2.2 Introduction .................................................. 6
  2.2.1 What is covered in this book? ......................... 7
  2.2.2 Who should read this book? ......................... 9
  2.2.3 Why another book on Machine Learning? .......... 10
  2.2.4 Is Machine Learning complex? ..................... 10
  2.2.5 Organization of this book ......................... 10
  2.2.6 Errata, updates and support ..................... 11
2.3 Why Free and Open Machine Learning .................. 11
  2.3.1 Open Source (FOSS) .................................. 12
  2.3.2 Open data ............................................. 14
  2.3.3 Open Science and open algorithms ............... 15
  2.3.4 Open architectures .................................. 16
  2.3.5 Green ML ............................................ 17
2.4 What is machine learning .................................. 18
  2.4.1 ML, AI and NLP: What is what ...................... 18
  2.4.2 The paradigm shift: Creating smart software .... 20
    2.4.2.1 What is a machine learning model ............ 21
    2.4.2.2 Statistics is not machine learning .......... 21
  2.4.3 Overview machine learning methods ............... 22
    2.4.3.1 Supervised Learning .......................... 23
    2.4.3.2 Unsupervised Learning ...................... 23
    2.4.3.3 Reinforcement learning (RL) ............... 24
    2.4.3.4 Deep learning (DL) .......................... 24
    2.4.3.5 AutoML ........................................ 25
  2.4.4 Other common terms used in the ML world ......... 25
    2.4.4.1 Data science .................................. 25
    2.4.4.2 Generative model ............................ 25
    2.4.4.3 Neural networks (NNs) ..................... 26
    2.4.4.4 Vision .......................................... 26
    2.4.4.5 Speech .......................................... 26
    2.4.4.6 Language ...................................... 26
    2.4.4.7 Knowledge ...................................... 27
2.5 Machine Learning for Business Problems
   2.5.1 When to use machine learning?
   2.5.2 Common business use cases
      2.5.2.1 Healthcare
      2.5.2.2 Language translation
      2.5.2.3 Chat bots
      2.5.2.4 eCommerce Recommendation systems
      2.5.2.5 Quality inspection and improvement
      2.5.2.6 Vision
      2.5.2.7 Financial services
      2.5.2.8 Marketing
      2.5.2.9 HR services
      2.5.2.10 Predicting services
      2.5.2.11 Software
      2.5.2.12 Security
      2.5.2.13 Privacy
      2.5.2.14 Risk and compliance
   2.5.3 Business Examples
   2.5.4 Business Challenges
   2.5.5 Business capabilities
   2.5.6 Business ethics

2.6 ML Reference Architecture
   2.6.1 The machine learning process
   2.6.2 Architecture Building Blocks for ML
      2.6.2.1 Principles for Machine learning
      2.6.2.2 Constraints
   2.6.3 ML Reference Architecture
      2.6.3.1 Business Processes
      2.6.3.2 Business Services
      2.6.3.3 Business Functions
      2.6.3.4 People, Skills and Culture
      2.6.3.5 Business organization
      2.6.3.6 Partners
      2.6.3.7 Risk management
      2.6.3.8 Development tools
      2.6.3.9 Machine learning Frameworks
      2.6.3.10 Programming Tools
      2.6.3.11 Data
      2.6.3.12 Data Tools
      2.6.3.13 Hosting
      2.6.3.14 Containers
      2.6.3.15 GPU - CPU or TPU
      2.6.3.16 Storage

2.7 Security, Privacy and Safety
   2.7.1 Introduction
   2.7.2 Security
      2.7.2.1 Top Machine Learning Security Risks
   2.7.3 Privacy
   2.7.4 Safety

2.8 Natural Language Processing
   2.8.1 Introduction
   2.8.2 Basic NLP functions
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.8.3</td>
<td>NLP Business challenges</td>
<td>60</td>
</tr>
<tr>
<td>2.8.4</td>
<td>NLP Business Examples</td>
<td>61</td>
</tr>
<tr>
<td>2.9</td>
<td>ML implementation challenges</td>
<td>61</td>
</tr>
<tr>
<td>2.9.1</td>
<td>Performance</td>
<td>62</td>
</tr>
<tr>
<td>2.9.2</td>
<td>Testing machine learning models</td>
<td>63</td>
</tr>
<tr>
<td>2.9.3</td>
<td>Interoperability</td>
<td>64</td>
</tr>
<tr>
<td>2.9.4</td>
<td>Debugging</td>
<td>65</td>
</tr>
<tr>
<td>2.9.5</td>
<td>Continuous improvements</td>
<td>65</td>
</tr>
<tr>
<td>2.9.6</td>
<td>Maturity of ML technology</td>
<td>66</td>
</tr>
<tr>
<td>2.9.7</td>
<td>Data and bias</td>
<td>66</td>
</tr>
<tr>
<td>2.9.8</td>
<td>Quality of Machine Learning frameworks</td>
<td>67</td>
</tr>
<tr>
<td>2.10</td>
<td>Building Blocks for FOSS ML</td>
<td>68</td>
</tr>
<tr>
<td>2.11</td>
<td>Open Machine Learning Frameworks</td>
<td>68</td>
</tr>
<tr>
<td>2.12</td>
<td>ML Frameworks</td>
<td>68</td>
</tr>
<tr>
<td>2.12.1</td>
<td>Acme</td>
<td>69</td>
</tr>
<tr>
<td>2.12.2</td>
<td>AdaNet</td>
<td>69</td>
</tr>
<tr>
<td>2.12.3</td>
<td>Analytics Zoo</td>
<td>70</td>
</tr>
<tr>
<td>2.12.4</td>
<td>Apache MXNet</td>
<td>71</td>
</tr>
<tr>
<td>2.12.5</td>
<td>Apache Spark MLlib</td>
<td>71</td>
</tr>
<tr>
<td>2.12.6</td>
<td>auto_ml</td>
<td>72</td>
</tr>
<tr>
<td>2.12.7</td>
<td>BigDL</td>
<td>73</td>
</tr>
<tr>
<td>2.12.8</td>
<td>Blocks</td>
<td>73</td>
</tr>
<tr>
<td>2.12.9</td>
<td>Caffe</td>
<td>74</td>
</tr>
<tr>
<td>2.12.10</td>
<td>ConvNetJS</td>
<td>74</td>
</tr>
<tr>
<td>2.12.11</td>
<td>Datumbox</td>
<td>75</td>
</tr>
<tr>
<td>2.12.12</td>
<td>DeepDetect</td>
<td>76</td>
</tr>
<tr>
<td>2.12.13</td>
<td>Deeplearning4j</td>
<td>76</td>
</tr>
<tr>
<td>2.12.14</td>
<td>Detectron2</td>
<td>77</td>
</tr>
<tr>
<td>2.12.15</td>
<td>Dopamine</td>
<td>77</td>
</tr>
<tr>
<td>2.12.16</td>
<td>Fastai</td>
<td>78</td>
</tr>
<tr>
<td>2.12.17</td>
<td>Featuretools</td>
<td>78</td>
</tr>
<tr>
<td>2.12.18</td>
<td>FlyingSquid</td>
<td>79</td>
</tr>
<tr>
<td>2.12.19</td>
<td>Karate Club</td>
<td>79</td>
</tr>
<tr>
<td>2.12.20</td>
<td>Keras</td>
<td>80</td>
</tr>
<tr>
<td>2.12.21</td>
<td>learn2learn</td>
<td>80</td>
</tr>
<tr>
<td>2.12.22</td>
<td>Lore</td>
<td>81</td>
</tr>
<tr>
<td>2.12.23</td>
<td>Microsoft Cognitive Toolkit (CNTK)</td>
<td>82</td>
</tr>
<tr>
<td>2.12.24</td>
<td>ml5.js</td>
<td>82</td>
</tr>
<tr>
<td>2.12.25</td>
<td>Mljar</td>
<td>83</td>
</tr>
<tr>
<td>2.12.26</td>
<td>MLsquare</td>
<td>83</td>
</tr>
<tr>
<td>2.12.27</td>
<td>NeuralStructuredLearning</td>
<td>84</td>
</tr>
<tr>
<td>2.12.28</td>
<td>NNI (Neural Network Intelligence)</td>
<td>84</td>
</tr>
<tr>
<td>2.12.29</td>
<td>NuPIC</td>
<td>85</td>
</tr>
<tr>
<td>2.12.30</td>
<td>Plato</td>
<td>85</td>
</tr>
<tr>
<td>2.12.31</td>
<td>Polyaxon</td>
<td>86</td>
</tr>
<tr>
<td>2.12.32</td>
<td>PyCaret</td>
<td>86</td>
</tr>
<tr>
<td>2.12.33</td>
<td>Pylearn2</td>
<td>87</td>
</tr>
<tr>
<td>2.12.34</td>
<td>Pyro</td>
<td>87</td>
</tr>
<tr>
<td>2.12.35</td>
<td>Pythia</td>
<td>88</td>
</tr>
<tr>
<td>2.12.36</td>
<td>PyTorch</td>
<td>88</td>
</tr>
<tr>
<td>2.12.37</td>
<td>ReAgent</td>
<td>89</td>
</tr>
<tr>
<td>2.12.38</td>
<td>RLCard</td>
<td>89</td>
</tr>
<tr>
<td>2.12.39</td>
<td>Scikit-learn</td>
<td>90</td>
</tr>
<tr>
<td>2.12.40</td>
<td>SINGA</td>
<td>90</td>
</tr>
<tr>
<td>Section</td>
<td>Tool</td>
<td>Page</td>
</tr>
<tr>
<td>---------</td>
<td>-------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>2.12.41</td>
<td>Streamlit</td>
<td>91</td>
</tr>
<tr>
<td>2.12.42</td>
<td>Tensorflow</td>
<td>91</td>
</tr>
<tr>
<td>2.12.43</td>
<td>TF Encrypted</td>
<td>92</td>
</tr>
<tr>
<td>2.12.44</td>
<td>Theano</td>
<td>92</td>
</tr>
<tr>
<td>2.12.45</td>
<td>Thinc</td>
<td>93</td>
</tr>
<tr>
<td>2.12.46</td>
<td>Turi</td>
<td>93</td>
</tr>
<tr>
<td>2.12.47</td>
<td>TuriCreate</td>
<td>94</td>
</tr>
<tr>
<td>2.12.48</td>
<td>Vowpal Wabbit</td>
<td>94</td>
</tr>
<tr>
<td>2.12.49</td>
<td>XAI</td>
<td>95</td>
</tr>
<tr>
<td>2.13</td>
<td>Computer vision</td>
<td>95</td>
</tr>
<tr>
<td>2.13.1</td>
<td>libfacedetection</td>
<td>95</td>
</tr>
<tr>
<td>2.13.2</td>
<td>YOLOv3</td>
<td>96</td>
</tr>
<tr>
<td>2.13.3</td>
<td>Raster Vision</td>
<td>96</td>
</tr>
<tr>
<td>2.13.4</td>
<td>DeOldify</td>
<td>97</td>
</tr>
<tr>
<td>2.13.5</td>
<td>SOD</td>
<td>97</td>
</tr>
<tr>
<td>2.13.6</td>
<td>makesense.ai</td>
<td>97</td>
</tr>
<tr>
<td>2.13.7</td>
<td>DeepPrivacy</td>
<td>98</td>
</tr>
<tr>
<td>2.13.8</td>
<td>Face_recognition</td>
<td>98</td>
</tr>
<tr>
<td>2.13.9</td>
<td>DeepFaceLab</td>
<td>99</td>
</tr>
<tr>
<td>2.13.10</td>
<td>FaceSwap</td>
<td>99</td>
</tr>
<tr>
<td>2.13.11</td>
<td>JeelizFaceFilter</td>
<td>100</td>
</tr>
<tr>
<td>2.13.12</td>
<td>OpenCV: Open Source Computer Vision Library</td>
<td>100</td>
</tr>
<tr>
<td>2.13.13</td>
<td>Luminoth</td>
<td>101</td>
</tr>
<tr>
<td>2.14</td>
<td>ML Tools</td>
<td>101</td>
</tr>
<tr>
<td>2.14.1</td>
<td>AI Explainability 360</td>
<td>101</td>
</tr>
<tr>
<td>2.14.2</td>
<td>Apollo</td>
<td>102</td>
</tr>
<tr>
<td>2.14.3</td>
<td>Data Science Version Control (DVC)</td>
<td>102</td>
</tr>
<tr>
<td>2.14.4</td>
<td>Espresso</td>
<td>103</td>
</tr>
<tr>
<td>2.14.5</td>
<td>EuclidesDB</td>
<td>103</td>
</tr>
<tr>
<td>2.14.6</td>
<td>Fabrik</td>
<td>104</td>
</tr>
<tr>
<td>2.14.7</td>
<td>Face_recognition</td>
<td>104</td>
</tr>
<tr>
<td>2.14.8</td>
<td>Kedro</td>
<td>105</td>
</tr>
<tr>
<td>2.14.9</td>
<td>Ludwig</td>
<td>106</td>
</tr>
<tr>
<td>2.14.10</td>
<td>makesense.ai</td>
<td>106</td>
</tr>
<tr>
<td>2.14.11</td>
<td>MLflow</td>
<td>107</td>
</tr>
<tr>
<td>2.14.12</td>
<td>MLPerf</td>
<td>107</td>
</tr>
<tr>
<td>2.14.13</td>
<td>ModelDB</td>
<td>108</td>
</tr>
<tr>
<td>2.14.14</td>
<td>Netron</td>
<td>108</td>
</tr>
<tr>
<td>2.14.15</td>
<td>NLP Architect</td>
<td>108</td>
</tr>
<tr>
<td>2.14.16</td>
<td>ONNX</td>
<td>109</td>
</tr>
<tr>
<td>2.14.17</td>
<td>OpenML</td>
<td>110</td>
</tr>
<tr>
<td>2.14.18</td>
<td>Orange</td>
<td>110</td>
</tr>
<tr>
<td>2.14.19</td>
<td>PySyft</td>
<td>111</td>
</tr>
<tr>
<td>2.14.20</td>
<td>RAPIDS</td>
<td>111</td>
</tr>
<tr>
<td>2.14.21</td>
<td>SHAP</td>
<td>112</td>
</tr>
<tr>
<td>2.14.22</td>
<td>Skater</td>
<td>112</td>
</tr>
<tr>
<td>2.14.23</td>
<td>Snorkel</td>
<td>112</td>
</tr>
<tr>
<td>2.14.24</td>
<td>Streamlit</td>
<td>113</td>
</tr>
<tr>
<td>2.14.25</td>
<td>TensorWatch</td>
<td>113</td>
</tr>
<tr>
<td>2.14.26</td>
<td>VisualDL</td>
<td>114</td>
</tr>
<tr>
<td>2.14.27</td>
<td>What-If Tool</td>
<td>114</td>
</tr>
<tr>
<td>2.15</td>
<td>ML hosting</td>
<td>114</td>
</tr>
<tr>
<td>2.15.1</td>
<td>BentoML</td>
<td>114</td>
</tr>
<tr>
<td>2.15.2</td>
<td>Streamlit</td>
<td>115</td>
</tr>
</tbody>
</table>
CHAPTER 1

Abstract

This publication is created to promote and advocate the use of FOSS machine learning for real practical business use cases. Machine learning is a fascinating technology. Free and Open machine learning should be the norm for business innovation. So simple to use for complex problems. Freedom to control machine learning technology is not self-evident. Free and Open Machine Learning puts you in full control.

This publication empowers everyone to make a head start using the powerful machine learning technology in a Free, Open and Simple way.

Note: This is a living document. A stable version of this publication (version 1.0) is available as hard copy. You can order it at Amazon, click here\(^1\) to order. So support this project and buy a hard copy!

Machine learning is an exciting and powerful technology. The continuous use and growth of machine learning technology opens new opportunities. It also enables solving complex problems in a simple way. Problems that are impossible to solve by using traditional software technologies. This great machine learning technology should available for everyone. This means that everyone should be able to learn, play and create great applications using machine learning technology. But also reuse existing machine learning solutions, inspect solutions and improve solutions of others. Without borders or strings attached.

The key focus of this publication is on Free and Open Machine Learning technologies. This to remove barriers for learning, playing, using and reusing machine learning technologies for real practical use cases for everyone.

Of course you can use or switch to cloud company solutions to deploy your machine learning driven application in production. But besides vendor lock-in, crucial aspects like safety, privacy and security for machine learning applications are only possible when using fully transparent Free and Open machine learning building blocks and solutions.

This document describes an open machine learning architecture. Including key aspects that are involved for real business use. This means e.g. that we focus on FOSS machine learning software and open datasets.

Since the majority of humans are not a graduated mathematician, we skip deep mathematical background concepts of machine learning algorithms in this publication. Good books with lots of mathematical background information on

how machine learning works are available for more than 70 years. There are plenty excellent free and open publications
available if you want to learn everything about the inner working of the mathematical algorithms that power the current
exciting machine learning applications. In the learning resources section in this publication you can find a list of good
references. All references in this publication are publications available under a creative commons license (cc-by).

This publication has a core focus on outlining how Free and Open machine learning can be used for real business use
cases. This is done by describing:

- Key machine learning concepts. The focus is on concepts that are needed in order to use solid FOSS machine
  learning frameworks and datasets when creating a machine learning powered application.

- An open reference architecture for creating and maintaining a machine learning solution architecture and IT
  landscape.

- Presenting useful and most used FOSS machine learning building blocks. Most Open Solution Building Blocks
  for machine learning are FOSS based. The most used solutions have a healthy ecosystem of (open) tools and
  service companies that enables you to create your machine learning application as fast as possible.

- Key quality aspects for engineering and maintaining your machine learning application.

- Important safety, privacy and security aspects to prevent disasters.

- Ethical issues (like bias) and guidelines for handling these issues in a transparent way.

No pieces of program code or mathematical formulas is presented in this publication. The emphasis is on machine
learning concept and applying machine learning technology for real business use cases. No programming knowledge
is needed to enjoy and learn machine learning.

This publication is created to give you a head start with using Free and Open machine learning technology to solve
your business problems. Without any strings attached, so the focus is on Free and Open transparent machine learning
technologies and solutions only!

**Warning:** This document is a living document! Collaboration is fun, so *Help Us* by contributing! Some more
background information of the project can be found in the readme on github.com². And do not forget to join the
ROI movement³!

---

² [https://github.com/nocomplexity/FreeAndOpenMachineLearning](https://github.com/nocomplexity/FreeAndOpenMachineLearning)
³ [https://www.bm-support.org/projects/](https://www.bm-support.org/projects/)
2.1 Preface

We humans are since the beginning of the development of modern computers obsessed with creating computers that have super powers. Even before the birth of computers, research has been done on artificial intelligence (AI). The question what artificial intelligence really is, is hard and fuel for philosophical discussions.

Nowadays we see more and more products created that claim to have super powers that come close to AI. A look under the surface shows however that the real progress on AI is made by a tangible technique, called machine learning. So our the focus in this publication is on machine learning. And not on philosophical views on what will be possible in the future when machine learning evolves towards AI.

Machine learning today is capable of solving challenging problems that impact everyone around the world. Problems that were impossible to solve in the past. Or problems that were too expensive or too complex to solve using traditional computer technologies. Nowadays solving a certain type of complex problems is possible using new machine learning technology.

Very complex problems and meaningful problems are currently solved using applications based on machine learning algorithms. Many firms involved are willing to tell and show you how easy it is! But you must be aware: machine learning is a buzzword in the industry! So the machine learning field is full of companies that use fads, all kind of vendor lock-in options and marketing buzz to take your money without delivering long running solutions. That is why this publication advocates for Free and Open machine learning.

This publication is aimed to give you practical information so you can start with applying free and open machine learning tools and frameworks. With minimum cost and no strings attached. This publication enables you also with the knowledge of what is possible with machine learning technology and what is still wishful thinking.

Everything described in this publication is with no strings attached. So the focus is on openness for machine learning tools, algorithms and knowledge. The core focus is outlining core concepts of machine learning and showing an open machine learning architecture that make machine learning possible for real business use cases. So this publication is also focused on outlining open source machine learning solutions (FOSS) that make it possible to start your machine learning journey.

This publication is to enable business IT consultants, IT architects, and software developers to get a practical grounding in open machine learning and its business applications. So no programming exercises and no complex mathematical
formulas in this publication. Showing programming code is avoided on purpose. In the reference section of this publication you can find good open references for hands-on machine learning tutorials. As an add-on to this publication some hands-on machine learning tutorials are published as addendum with the online version of this publication.

Understanding core concepts of machine learning and using open machine learning technology is possible without coding. This publication empowers you to start transforming your organization into an innovative and open company for the future using new open machine learning technologies. If your company is committed to openness and you endorse key open principles to create value, you are an open company. See https://www.bm-support.org/open-company-principles/ for showing your commitment to openness.

Machine learning is and should not be the exclusive domain of commercial companies, data scientists, mathematics, computer scientists or hackers. Every business and everyone involved with automation should be able to take advantage of the machine learning techniques and applications available. This is possible within the field of machine learning as you learn in this publication.

Nowadays knowledge is more and more openly shared, thanks to open access, open publication licenses and open source software. So everyone can and should benefit from the possibilities that open machine learning frameworks and tools provide.

To create this publication a lot of papers, books and reports on machine learning are examined. And doing some ‘hands-on’ to experiences and feel the power of machine learning algorithms turned out to be crucial for understanding and creating this publication as well. This publication is focussed on making a the complex machine learning technology simple to use.

In my journey on learning how to apply machine learning for real business use cases many books turned out to be either too theoretical, or too much focused on programming machine learning algorithms. As an IT architect I missed the overall machine learning architecture picture from a typical IT architecture point of view. So business, information, application, infrastructure, security and privacy perspective. This publication fills up that gap.

Applying machine learning should be easy and simple. When barriers for using machine learning technology are lowered many more great applications can be developed for the benefit for everyone. This publication simplifies the use of the complex field of machine learning frameworks, software and applications for real business use cases. Creating meaningful machine learning applications in a already complex context is another discipline than creating and understanding the complex machine learning algorithms behind the machine learning frameworks. So this publication is for everyone who is short on time but is dedicated to make use of machine learning capabilities.

This publication is not an end, but is constructed as a continuous effort to provide usable open and non commercial information for applying machine learning technology. You can join this project too. See the HELP section in this publication.

This publication was only possible with the help of you! If you have a suggestion or correction, please send an email to info [at] bm-support [dot] org. I add you to the contributor list, unless you ask to be omitted.

## 2.2 Introduction

Machine learning (ML) is a rapidly advancing technology, made possible by the Internet, that already has significant impacts on our everyday lives. With the use of Machine learning you can solve challenging problems that impact everyone around the world. Machine Learning (ML) and Artificial Intelligence (AI) are rapidly emerging technologies that have the potential to change our world with speed that humankind has never experienced before.

Machine Learning and Artificial Intelligence are not the same, although the current technologies developed for ML do help research and developments on AI. ML can be characterized with a stricter definition from an engineering perspective. Trying to define AI raises more philosophical discussions on what intelligence is. This publication is focused on Free and Open machine learning. But beware that the terms machine learning and artificial intelligence are intertwined and many so called AI applications are in fact driven by machine learning technology.
You should be aware of the commercial buzz and fads surrounding AI and ML: Machine Learning, deep learning and a lot of tools developed are not ‘a universal solvent’ for solving all current problems. There is magic machine learning tool or method yet that can solve all your complex challenges. Machine learning is just a tool to solve a **certain type** of problems. Maybe in future the use of machine learning can be applied to a broader landscape of problems than currently possible. But do not try to solve all your problems with one (new)technology or toolset.

Artificial Intelligence and Machine Learning are now again in the forefront of global discourse, garnering increased attention from practitioners, industry leaders, policymakers, and the general public.

But despite the hype and money invested in machine learning technology the recent 5 years, one big question remains: Can machine learning technology already help us to solve hard and complex business problems like climate change, health welfare for all humans and other urgent problems?

This publication gives you a reality check. You learn what is easily possible using new machine learning technologies and tools, what the current potential is and what still remains wishful thinking for the future. We like transparency, so we focus solely on free and open machine learning technologies.

Innovation needs openness. This is also more than valid for machine learning technologies. Without real openness new developments and innovations in machine learning are impossible. As a practitioner in your business domain and with your unique expertise you can start making a difference. This publication gives you a starting point for trying to apply free and open machine learning technology on your unique use cases.

**2.2.1 What is covered in this book?**

Nowadays many people are talking about the transformative power of machine learning and how it will revolutionize the economy, but what does that mean for your business and how do you start? How to get solid independent advice to learn and how to apply machine learning? Can you improve or disrupt your business using FOSS machine learning tools that are widely available? This book gives you an introduction to get started with applying FOSS machine learning.

Machine learning concepts are mostly taught by academics and for academics. That’s why most learning material is
Free and Open Machine Learning, Release 1.0.1

dry and maths heavy. The theory behind machine learning is great, but requires also a very deep understanding within statistics and math. There is a large gap between theory and practice. Practice counts, because in a practical business context you want to determine if you can solve your problems with machine learning tools. Or at minimum do a short and cost efficient run to determine if a project has potential and more investments make sense.

To apply machine learning in real business use cases other skills besides some feelings for statistics and math are required. You need e.g. be able to have some knowledge about all typical IT things that are still needed before you can make use of the new paradigm that machine learning brings.

This publication is created for applying free and open machine learning in practice for real world use cases. This is where the rubber meets the road. So the core focus is on the ‘How’ questions. So key concepts are outlined and a conceptual and logical reference for free and open machine learning architecture is given. This to empower you to make use of FOSS machine learning technology in a simple and efficient way.

The field of machine learning is making rapid progress. Do you know what kind of applications for direct business use are already possible today? Are you aware of the currently low entry barriers that exist, to take direct advantage of machine learning? Is your knowledge of free and open source solutions available in the machine learning eco system up to date? How do you classify safety, security and privacy risk when using machine learning? These and other relevant questions for using machine learning in a business context are the foundation of this book.

Within the FOSS machine learning domain new toolsets, applications and companies are being created on a daily basis. So it is difficult to get a hold on what ML applications are viable, and which are a hype, fads or simply a hoax. Especially when the terms ML and AI are intertwined. This publication guides you through tangible working open source machine learning software.

The mentioned FOSS machine learning software building blocks in this publication are used at large. For real business use cases, and maybe with large similarities for your use case. And because a lot of ML software and tools needed is open (FOSS) software, solutions and tools available can be studied and improved.

Given that machine learning tools and techniques are already an increasingly part of our everyday lives, it is crucial for professionals in the IT industry to gain more knowledge on machine learning. You should start asking critical questions and maybe try to do some simple experiments. What will you do with machine learning tools and applications the coming 3 years? Are you really aware of the safety and privacy concerns evolving that are part of this technology? Do you really understand and control the working?

This publication is all about taking advantage of the new FOSS machine learning technologies for your business. The major machine learning concepts are explained, but the main emphasis of this book is to give insights in the various possibilities that are available within the open source machine learning ecosystem. This so you can start applying machine learning in your business today, without hidden dependencies or unknown strings attached towards a vendor or cloud hosting provider.

This publication gives an overview of all important FOSS machine learning frameworks and FOSS machine learning support tools that you can use for prototyping or for real business use cases and production systems.

This publication does not explain and dive into the statistics and deep mathematical algorithms behind machine learning. Also the algebra functions that form the foundation under machine learning algorithms and software libraries are only explained if needed for practical use and experiments. If you are interested in learning the mathematical foundations on which machine learning is developed you can find good free and open material in the reference section of this book.

This publication aims to cover the high level machine learning concepts and gives you information to get started to work with free and open machine learning for your business use case.

So this publication is concentrated on machine learning aspects where software, business and technology touch each other.
When we write Open Source Software or OSS in this report we explicitly mean FOSS as defined by the Free Software Foundation - FSF.org.

2.2.2 Who should read this book?

This book is created for everyone who wants to learn and get started with machine learning without being already forced into a specific solution. Creating Machine learning applications is possible with the use of FOSS building blocks only and on premise. So you do not need to use directly expensive Cloud infrastructure or commercial software packages. So if you like IT architecture, simple concepts and want to be empowered to play with machine learning and create your own solution, then this publication is for you.

This book is primary written with software developers, system administrators, security architects, privacy controllers, IT managers, directors, business owners, system engineers, quality managers, IT architects and other curious people interested in open technologies in mind.

This book crucial outlines machine learning concepts, but will not go into mathematical or technical details. But after reading this book you will have a more complete and realistic overview of the possibilities applying machine learning (ML) for your use cases.
2.2.3 Why another book on Machine Learning?

There are many books, courses and tutorials that learn you what machine learning is. However most of these books and courses are focused on hands-on learning and requires you to program. Also many books are focused on explaining concepts without a clear focus on how tools can be used on real business use cases. Also a publication that is truly open and is focused on the broad landscape that is needed for Free and Open Machine learning was simply not available.

Despite the enormous buzz and attention for machine learning it is proven to be hard to apply machine learning for real profitable use cases. Applying machine learning starts with understanding the core concepts, business architecture needs, constraints and insights in the technology components that are present. Also some notion of the typical pitfalls and challenges for applying machine learning for business use is needed.

2.2.4 Is Machine Learning complex?

You might get the impression when visiting presentations from commercial vendors that machine learning is simple. The hard work is already done and all you have to do is get your credit card and make use of the incredible machine learning cloud offering. This machine learning as a service (MaaS) takes your company to the next level and the advise of the sales consultant is clear: Using their MaaS service is so simple that entering your credit card number is probably the hardest part. Maybe it takes a minute, maybe more. But in the end you discover that solving problems using machine learning is not that simple after all. The great offerings of many large and small vendors selling MaaS from a fantastic cloud offering do not solve your business problem in a simple way. As with all new technologies and especially IT technology: There are over promises on advantages and getting the return on your investments is not simple. You are confronted with complex terminology, a machine learning back-box from your vendor that is of course great at billing, data collection and data cleaning problems you had never heard of, and security, privacy and even safety issues. And if you think it can not get worse also legal and ethical issues will slow your project down.

By using an open approach (tools, methods, datasets) for machine learning a lot of risks can be mitigated. E.g. it is easier to control spending in the important ramp up phase of your project. If you need more performance you can always move hosting to a cloud platform in a later stage. But you need to start with a flexible and scalable architecture that is no limitation for future goals.

There have been tremendous advances made in making machine learning more accessible over the past few years. This publication outlines some great OSS applications ready to be used, even if you really hate difficult mathematical formulas. Multiple developments are in progress that now really make it possible to drop your data and let a complex machine learning algorithm do the hard work.

But don’t be fooled. Even solving only ‘some type of problems’ using machine learning tools is a relatively ‘hard’ problem. So only equipped with the right knowledge, tools and resources it is possible to get great results. Solving soft business problems with machine learning requires far more than a good computer scientist alone. Using machine learning for soft problems requires a variety of disciples and a lot of creativity, experimentation and tenacity.

2.2.5 Organization of this book

The topics explored in this publication include:

- Why Free and Open Machine Learning. This section outlines why we all should promote and advocate for openness and freedom regarding this promising technology.

- What is Machine Learning. This is the section to read if you are short on time and want a simple outline of complex machine learning concepts.

- Machine Learning for business problems. New technologies come with new opportunities for innovation. This section outlines common business use cases that are possible today using machine learning technology.
• Machine learning Reference architecture. Starting with machine learning can be overwhelming. This section gives an overview of the business and technology aspects that you face when applying machine learning for real business use cases. But this section also helps you with developing your machine learning solution architecture.

• Security, Privacy and Safety. The things you do not see are often the most important aspects. Security, Privacy and safety are very complex to deal with for normal IT solutions. But for machine learning these non functional aspects must be taken into your design upfront from a system perspective. This section outlines the key aspects for security, privacy and safety you should be aware of when creating machine learning applications.

• Natural language processing (NLP). Hard to solve speech and text processing problems are now far more easily solved using machine learning algorithms. This section outlines still on of the most used applications for machine learning: NLP.

• Machine learning implementation challenges: Knowing what machine learning can do and how it works is no guarantee that creating an machine learning application succeeds. The failure rate of normal IT projects are already very high for decades. Machine learning projects are complex and risky. This sections gives guidance on avoiding pitfalls when applying machine learning for real business use.

• FOSS System Building Blocks for machine learning. This publication presents an opinionated list of FOSS software building blocks that can be used when creating machine learning applications. Starting with FOSS machine learning building blocks means you start with no strings attached. Switching to cloud hosting solutions later is always possible, but machine learning needs experimentation and playing. With open data and open tools.

• Learning Resources. Some very good learning resources for machine learning and NLP are open. So licensed using a creative commons license. After reading this publication a next step can be to dive in depth into a specific machine learning aspect, framework or technology. This section provides references to open learning resources, including references to hands-on tutorials.

2.2.6 Errata, updates and support

We made serious efforts to create a first readable version of this book. However if you notice typos, spelling and grammar errors please notify us so we can improve this publication. You can create a pull request on github or simply send an email to us.

Since the world of machine learning is rapidly evolving this book be continuously updated. That’s why there is an open on-line version of this book available that always incorporates the latest updates.

Note: If like to contribute to promote the Free and Open Machine Learning principles and to make this book better: Please CONTRIBUTE! See the HELP section.

2.3 Why Free and Open Machine Learning

Free and Open machine learning is comparable with open source software (FOSS - Free and Open Source Software). But openness for machine learning requires more than open source software alone. So we advocate for using Free and Open machine learning.

The term open source software (OSS) means FOSS in this publication. Freedom is important for free and open machine learning. ‘Open source software’ is sometimes also called “Free software”, “libre software”, “Free/open source software (FOSS or F/OSS)”, and “Free/Libre/Open Source Software (FLOSS)”. But the term “Free software” has been sometimes misinterpreted as meaning “no cost”, which is not the intended meaning. It is all about Freedom, so a better term would have been to call it Freedom Software. So ‘Free’ open source software (FOSS) refers to freedom, not price. This also applies for Free and Open Machine Learning. Free refers to freedom.
The Freedom part makes a key difference in making sure machine learning technology and all related aspects, secure freedom in a sustainable way.

FOSS machine learning is crucial for everyone. In our view machine learning technology must be inclusive for all. This means that besides using FOSS machine learning frameworks like Tensorflow all aspects must be open and transparent. In this way machine learning becomes a real open and inclusive technology that can be used for the advantage of everyone. And everyone should be able to experiment, play and create a new machine learning application. Without major obstacles in terms of cost for technology usage or hardware required.

Free and Open machine learning means that everyone must be able to develop, test, play and deploy machine learning based solutions. Large investments should not be needed for using and applying machine learning. So not only companies or people who can afford the enormous investments needed in specialized GPU hardware benefit of machine learning technology, but everyone can benefit. In this way everyone is able to create meaningful applications to create a better world. Without making enormous investments upfront.

FOSS machine learning involves more than FOSS software. The following aspects are needed for real Free and Open Machine Learning:

- FOSS Machine learning software (Free and Open Source software)
- Open Data
- Open Algorithms (Transparent machine learning algorithms)
- Open Architectures
- Open Science

These aspects are the core pillars of Free and Open Machine Learning.

### 2.3.1 Open Source (FOSS)

Free and open-source software (FOSS) is software that can be classified as both free software and open-source software. FOSS is an inclusive term that covers both free software (FLOSS) and open-source software (OSS).

Open Source is an approach for the design, development, and distribution of new products & knowledge offering practical accessibility to its source. Real open source solutions have a license that is approved by the Free Software Foundation (FSF) (https://www.fsf.org/) or the Open Source Initiative (OSI) foundation (https://opensource.org/). Open source is all about collaboration and Freedom. Collaboration is key for developing, applying and using machine learning functionality.

Software is free software if users have four essential freedoms:

- The freedom to run the program as you wish, for any purpose.
• The freedom to study how the program works, and change it so it does your computing as you wish. Access to the source code is a precondition for this.

• The freedom to redistribute copies so you can help others.

• The freedom to distribute copies of your modified versions to others. By doing this you can give the whole community a chance to benefit from your changes. Access to the source code is a precondition for this.

Open Source Software (FOSS) is the standard for machine learning algorithms. However using open source software is still a new and innovative concept for many companies. If you really want to benefit from new machine learning software you must go for a solid FOSS machine learning ecosystem. This makes you flexible, independent and you can still use thousands of consultancy firms and (Cloud)hosting companies that can help you, or are willing to provide hosting facilities.

A transition towards FOSS software can already be very hard and can be disruptive for many companies. It takes the right mindset, attitude and culture within a company. Applying machine learning for real business cases is also complex and challenging. So taking advantage of machine learning requires the right innovative mindset. Using machine learning without using the benefits that come with the FOSS ecosystems of choice, is like learning to swim without hitting the water. So hit the water as soon as possible, after a while you see and use the benefits.

Machine learning applications are expensive to develop and to adopt. This accounts for the development process itself but also good skilled professional IT engineers and scientists are expensive. But it accounts also for the needed infrastructure and other software resources needed to develop meaningful applications for your business. This means that currently big firms like Google, IBM, Microsoft, Facebook and Amazon are at the front of the queue and smaller counterparts get left behind. But most of the scientific knowledge of machine learning technology and a lot of software is open and freely available. The core concepts of the technique behind machine learning is crucial to known before starting business projects. Machine learning for real use cases requires adjustments and continuous tweaking, which is hard when you are using inflexible black-box solutions.

FOSS developments in the machine learning field are absolutely no hobby projects. Almost all major FOSS machine learning developments are backed by small or large companies (e.g. Google, Microsoft, Facebook, Uber) active in the deep learning ecosystem. Also many great FOSS machine learning frameworks are backed by research groups of universities or research communities organized by universities. Small machine learning FOSS projects are often developed by PhD researchers and are supported by a strong scientific foundation.

A focus on open source (FOSS) software for applying machine learning for real is crucial. FOSS machine learning applications and frameworks have the following benefits:

• Create solutions software faster, better and with less friction. You can adjust what you want without limitations.

• Lower cost for creating your first pilot project. Mind: Your first attempts will fail. And the faster your pilot projects fail, the better. This since applying the new machine learning capabilities requires a learning curve. Technical, but also for the organization and business side point of view.

• Flexibility and changeability.

• No vendor lock ins. Of course the machine learning cloud offerings of the major tech companies are great (Azure ML, IBM Watson, Amazon, Google etc). But playing around without any strings attached and limitations set for you gives you a head start.

• Software is less dependent on a single company or software developer. Healthy FOSS projects have a large ecosystem of companies and independent contributors that maintain the code and preserve the quality.

• Software is often more compatible with a wide range of other open systems. Most FOSS projects build upon open platforms. Also good ML frameworks want to be used and improved. So open and easy integration with other systems and tools is often built-in.

• Open code is better science. The field of machine learning is still improving. Many researchers work on algorithms and improvements. Open code enables open science. Community input and feedback increases the quality. Also openness means that when papers of researchers are published everyone can inspect, use and improve the code that was developed. This openness enforces quality.
FOSS machine learning and machine learning in general is very popular. See e.g. the diagram below which shows a view of the increase in google searches for the recent decade. You should have very strong arguments, also from a business perspective. This is because investments for real world application have always have business risks. Choosing a commercial black box solution often increases business risks and mitigation of risks is harder. E.g. security and privacy risk mitigation is hard with blackbox solutions.

All IT companies advertise with machine learning powered software products nowadays. This also means that existing software that has been sold for decades is now re-branded with the new machine learning buzz words. Also terms like cognitive, artificial intelligence (AI) powered and data driven are used to sell you old solutions using this new trend. You can easily be fooled since massive marketing efforts (time, money, material) are invested to sell old buggy solutions as new innovative machine learning powered solutions. In reality black box solutions from small or large vendors that seems good to be true for your use case, are almost always based on fads. This is why you should be very suspicious when using cloud based machine offerings that offers you instant new business and customers. Make sure to do a fast and cheap hands on innovation project first. Evaluate if and how your business use case can really benefit from machine learning. If a new machine learning solution looks too good to be true, be aware.

To use machine learning for real business applications you should use and reuse good FOSS tools, frameworks and knowledge available. But you should also take the quality aspects, technical and non-technical, that comes with a machine learning framework choice into account.

When using machine learning FOSS solutions you can and should inspect the working and evaluate all risks involved. By using a FOSS solution you can ask every IT company or consultant with the right skills to audit the application. Because in the end: When security, safety or privacy of your customers is at risk, you are accountable.

### 2.3.2 Open data

Free and Open machine learning does not only need FOSS software, but also open data sets. Data is one of the most important aspects for making machine learning work. Without data and open transparent insights in the various quality aspects of the data, machine learning is not open.
Without data machine learning is not possible. FOSS Machine learning systems need open data to function. To function properly the following is needed for FOSS machine learning:

- **Open data.** Open data is data that can be freely used, re-used and redistributed by anyone.
- **Lots of data.** Training machine learning models requires large amounts of data.
- **Data variety.** For good training sets variety in data used is crucial. Else the bias problem turns up directly.
- **Data veracity.** This means the truthfulness of data.
- **Trust in the outcome of applications powered by machine learning technology is only possible when the input data is fully available.**

Open and reusable quality datasets are crucial for creating machine learning driven applications. If you use a trained machine learning algorithms, it is crucial that you have full insight in the origin of all training data. How it was collected, filtered and used.

Creating a data set to test and develop machine learning algorithms is hard and time consuming. Many current machine learning algorithms are developed and verified by using open data sets. In [https://en.wikipedia.org/wiki/List_of_datasets_for_machine-learning_research](https://en.wikipedia.org/wiki/List_of_datasets_for_machine-learning_research) a short overview can be found of various data sets used for scientific machine learning research.

Free and open machine learning means that everyone should be able to access and use data that is used to train machine learning applications. So Google, Facebook and many other companies who donate a lot of machine learning knowledge and frameworks in the open source domain rarely release datasets that are used for their fantastic commercial machine learning offerings. Not knowing details about datasets, especially for live saving systems that are powered using machine learning technology, means verification of claims is impossible. There can also be large privacy risks involved, since training machine learning algorithms requires large datasets. Seldom do people give permission for using their valuable data for developing applications that are not beneficial for them. E.g. why should a government use your data in order to develop an application that is not in your interest.

Data collection and data preparation is a major bottleneck in open machine learning. As machine learning becomes more widely used, it is important to acquire large amounts of open data. Especially for state-of-the-art neural networks.

Free and open machine learning means that everyone should be able to access and use data that is used to train machine learning applications. So Google, Facebook and many other companies who donate a lot of machine learning knowledge and frameworks in the open source domain rarely release datasets that are used for their fantastic commercial machine learning offerings. Not knowing details about datasets, especially for live saving systems that are powered using machine learning technology, means verification of claims is impossible. There can also be large privacy risks involved, since training machine learning algorithms requires large datasets. Seldom do people give permission for using their valuable data for developing applications that are not beneficial for them. E.g. why should a government use your data in order to develop an application that is not in your interest.

In the ideal FOSS machine learning world all non-personal information is open and free for everyone to use, build on and share. So every organisation, small or big, can create new machine learning applications.

Preparing data to be used for training machine learning models is still very time consuming and cost intensive. So most business machine learning applications created make use of already trained models. E.g. for speech or image recognition. But for your unique use cases: training your own machine learning model is crucial.

Machine learning involves data, so you and your business should act based on leading data ethics principles. Some obvious data ethics principles are:

- **Foresighted responsibility.** So think ahead or imagining or anticipate what might happen in the future.
- **Use open data.**
- **Be transparent.**
- **Respect data privacy regulations and laws (e.g. EU GDPR)**

### 2.3.3 Open Science and open algorithms

Machine learning is a challenging science. Many researchers on universities worldwide are working to develop new knowledge for solving a range of complex problems.

Universities are funded by taxpayers. So in an ideal world everyone should benefit from knowledge developed. Also almost all knowledge developed is based on work developed earlier by others. This is how science works. We build upon knowing of others to develop new knowledge and insights.
Open science represents an approach to the scientific process based on cooperative work and new ways of diffusing knowledge by using digital technologies and new collaborative tools. This idea captures a systemic change to the way science and research have been carried out for the latest fifty years: shifting from the standard practices of publishing research results in scientific publications towards sharing and using all available knowledge at an earlier stage in the research process.

Developing machine learning knowledge using open science means that publications, data, results, and software is accessible without borders for everyone to learn and build upon. Key pillars of open science important for open machine learning are:

- Open Data:
- Open source software
- Open access

This so everyone can validate claims, inspect algorithms used and can created and read machine learning experiments done without large upfront costs. Transparency is needed for trust. This also accounts for machine learning applications, algorithms and frameworks used.

For real open machine learning applications providing real transparency in terms of explaining how results are created is a complex problem. This is a direct result of how some type of machine learning algorithms work. The current generation of machine learning systems offer tremendous benefits, but their effectiveness is limited by the machine’s inability to explain its decisions and actions to users. The so called ‘explainable’ machine learning tools will be essential for users to understand and trust machine learning applications.

Only when the basic principles for open science are followed, trust in machine learning algorithms and software frameworks is possible.

The key of machine learning is smart algorithms. Algorithms that operate as “black boxes” should never be trusted. Fighting against e.g. your government is very difficult is no insight in the used algorithms. Open algorithms developed in an open scientific environment are key for trust.

FOSS machine learning with the use of open algorithms is needed to prevent a “black box society”. That is a society” in which key moments of our lives are mediated by unknown, unseen, and arbitrary algorithms. Open algorithms and algorithmic accountability is a way to stop this pattern. An open algorithm makes it possible for anyone to analyse. There is a freely available description and a FOSS reference implementation.

### 2.3.4 Open architectures

Architecture is a minefield. Architecture is not by definition high level and sometimes relevant details are of the utmost importance. It is not strange that the added value of architecture and architects within large companies and projects is under heavy pressure due to architecture failures at large and the emergence of agile approaches to solve business IT problems.

Architecture (business, information, application and technical) of digital systems have an enormous impact on the products we use daily. For developing and creating large complex systems you still need an architecture. Developing a solid solution architecture and creating solutions by working using an agile method should reinforces each other.

Open architectures should be concentrated around the following pillars:

- Solutions should be created using FOSS system building blocks.
- The created architecture blueprint is available for everyone. so use a friendly (creative commons) license.
- The architecture is developed from an open process in which everyone participates to improve the architecture. E.g. also customers, business stakeholders other stakeholders that will be impacted by the architecture design in future. Borders that hinder participation should be removed.
- The architecture is based around good usable standards that anyone can and may implement, use and improve. Unfortunate not all open standards are really open and usable.
2.3.5 Green ML

Applying new technology brings new responsibilities. Computations power needed for deep learning research have been doubling every few months. Machine learning computations can have a very large carbon footprint. This is a results of the way most algorithms are designed.

Most machine learning algorithms give only good results when large amounts of data are used and an enormous number of calculations are performed. Computers do use a lot of energy when calculations at large are performed.

Ironically, deep learning was inspired by the human brain, which is remarkably energy efficient. Moreover, the financial cost of the computations can make it difficult for academics, students, and researchers, in particular those from emerging economies, to engage in deep learning research.

Green machine learning means is machine learning optimized to minimize resource utilization and environmental impact. This can be done by data center resource optimization, balancing training data requirements versus accuracy, choosing less resource intensive models or in some cases transfer learning versus new models.

Besides the cost factor, green machine learning is an important factor for Free and Open machine learning since the benefits machine learning can bring should not harm the environment of all living cells that have no direct relationship with your machine learning application.

The Freedom to use the powerful machine learning technology should not limit the freedom to live in good health of others. So green ML is a difficult but important aspects for machine learning developments. So chose algorithms that perform well without weeks of calculation on datasets. Or make sure expensive and time consuming calculations can be reused by others in an easy way.
2.4 What is machine learning

To understand the basic principles of machine learning you do not need to have a PhD in computer science or have done a complex mathematical or technological study with a Master of Science (MSc) degree. Machine learning should be open and beneficial for everyone. So it is important that everyone can learn and understand the basics and the underlying principles of machine learning.

This section outlines common used terms that are used within the machine learning field. If you are short on time and want to know what the machine learning buzz is all about: This is the section you should read!

Before introducing terms and definitions: Be aware that no unified de-facto definition of machine learning exists. So be aware that when people are writing and talking about ‘machine learning’ they can be talking about totally different things and subjects. The machine learning (ML) label is often misused and intertwined with artificial intelligence (AI).

Investments in machine learning by large commercial companies are still growing. But a lot of documentation that is freely available on machine learning, especially some documents created by commercial vendors, are sometimes biased. In the reference section of this book you find a collection of open access resources to do a more in depth study on various machine learning subjects. Be aware that also open access publications are not free from commercial interest. So also open access publications on machine learning are not always objective and free from bias.

**Tip:** Be aware of facts and fads when reading machine learning papers and books. Always be critical.

This section outlines essential concepts surrounding machine learning more in depth.

2.4.1 ML, AI and NLP: What is what

Machine Learning (ML) and Artificial Intelligence (AI) are terms that are crucial to know when creating machine learning driven solutions. But also the term NLP (Natural language processing) is a term that is crucial for understanding current machine learning application that are created for speech or text. E.g. for bots with which you can converse instead of humans.

So let’s start with a high level separation of common used terms and their meaning:

- **AI** (Artificial intelligence) is concerned with solving tasks that are easy for humans but hard for computers.
- **ML** (Machine learning) is the science of getting computers to act without being explicitly programmed. Machine learning (ML) is basically a learning through doing. Often machine learning is regarded as a subset of AI.
- **NLP** (Natural language processing) is the part of machine learning that has to do with language (usually written). NLP concepts are outlined more in depth in another chapter of this book.
A clear distinction between AI and ML is hard to make. Discussions on making a clear distinguishing are often a waste of time and are heavily biased. For this publication we use the term machine learning (ML), since machine learning can be brought down to tangible hard mathematical algebra, software implementations and tangible applications. Philosophical discussions on questions ‘what is intelligence?’ are mostly related to AI discussions.

At its core, machine learning is simply a way of achieving AI. Machine learning can be seen as currently the only viable approach to building AI systems that can operate in complicated real-world environments.

A few other definitions of artificial intelligence:

- A branch of computer science dealing with the simulation of intelligent behaviour in computers.
- The capability of a machine to imitate intelligent human behaviour.
- A computer system able to perform tasks that normally require human intelligence, such as visual perception, speech recognition, decision-making, and translation between languages.

There are a lot of ways to simulate human intelligence, and some methods are more intelligent than others. AI raises questions on the philosophical spectrum, like ‘What is intelligence?’, ‘How do we measure intelligence?’ AI also gives a lot of fuel for ethical discussions like:

- Should AI driven machine learning be a legal entity?
- How do we prevent AI machines to kill human life, since AI machines will be ‘smarter’ than human intelligence ever will be.

These ethical questions should not be neglected. In the section ‘ML in Business problems’ a deep dive in the ethical issues for applying machine learning for business use cases is given.

Machine Learning is the most used current application of AI based around the idea that we should really just be able to give machines access to data and let them learn for themselves.

2.4. What is machine learning
2.4.2 The paradigm shift: Creating smart software

To really understand machine learning a new view on how software can be created and how it works is needed. Most of our current computer programs are coded by using requirements, logic and design principles for creating good software. E.g. When you add an item to your shopping cart, you trigger an application component to store an entry in a shopping cart database table. So humans create an algorithm to solve a problem. Algorithms are a sequence of computer instructions used to solve a problem.

Many real world problems aren’t easy to solve. A good solution requires knowledge of the context and a lot of domain knowledge built from experience. The domain knowledge needed is often difficult to identify exactly.

Determining the exact context of a car in traffic and in order to make a decision within milliseconds to go left or right is very hard programming challenge. It takes you decades and you will never do it right. This is why a paradigm shift in creating software for the next phase of automation is needed.

Programming computers the traditional way made it possible to put a man on the moon. To break new barriers in automation in our daily lives and science requires new ways of thinking about creating intelligent software. Machine learning is a new way to ‘program’ computers. When a programming challenge is too large to solve with traditional programming methods (requirements collection, decision rules collection, etc) a program for a computer should be ‘generated’. Generated based on some known desired output types. But knowing all desired output types in front for a problem solution is often impossible. So your new machine learning ‘program’ will get it wrong sometimes. Large amounts of input data increases the quality of the generated prediction model. In the old traditional paradigm called ‘the program’.

![Diagram: Difference between general programming and (supervised) machine learning.](image)

In essence machine learning makes computers learn the same way people learn: Through experience. And just as with humans, algorithms exist that makes it possible to make use of learned experience of other computers to make your machine learning application faster and better.

The essence of machine learning is that a model is constructed based on so called training data. In machine learning, learning algorithms, so not computer programmers, create the rules.

The term machine learning model refers to the model artefact that is created by the training process. With this machine learning model it is now possible to create meaningful output based on new input. At least when the trained model is
functioning as intended. In the figure below another view of the essence of the working of machine learning.

![Diagram of machine learning process]

2.4.2.1 What is a machine learning model

A machine learning model consists of numbers. Most of the time a very large amount of numbers. With the danger of getting into math: A machine learning model is a collection of numbers that are presented in a large multi dimensional matrix.

A model in the machine learning world is not different than any other mathematical model that presents some knowledge or (trained)information. It is just a large amount of numbers. So you need the algorithm to use it.

A model of data (plain numbers) can be used for any number of things. E.g.:

- To simply tell you about the behaviour of your data. For example, the mean is a model. If you imaging picking numbers at random from 1-10, a mean does summarize some useful information about your data. The same with the median and the variance. These are extremely lossy models, but they are models of your data.

- To classify data. Say you’ve trained a classifier that classifies whether a photo contains a cat or not. That classifier concisely summarizes your data as “cat photo” or “non-cat photo.”

- An efficient way to represent data for some other task. For example, you might generate paraphrases of a documents and model this as vector data. You can then use this model to classify the unique author of the text. So if you present a new document to this model using a simple machine learning algorithms the model gives you a number that indicates if this new document is from the same author or not.

2.4.2.2 Statistics is not machine learning

Statistics is not machine learning. So let repeat this one more time: Statistics is not machine learning. But the truth is that statistics and machine learning are intertwined and can not be seen separated. So for a good understanding and basic knowledge of machine learning, basic statistics knowledge is important.

The question ‘What’s the difference between Machine Learning and Statistics?’ is a questions that occurs often and leads to heavy discussion among scientists. To get it straight: A very clear separation between machine learning and statistics is hard to make. Machine Learning is however more a hybrid field than statistics. Some answers on this question are:

- Machine learning is essentially a form of applied statistics.

- Machine learning is glorified statistics.

2.4. What is machine learning
• Machine learning is statistics scaled up to big data.

• Machine learning improves a model by learning using data, where a statistical model is not automatically improved feeding it more data.

• Statistics emphasizes inference, whereas machine learning emphasized prediction.

Of course all answers are a bit true. With Machine Learning insights improve based when using more data. Using pure statistical models, learning and improving is not automatically guaranteed when more data is added. Statistical and machine learning methods and the reasoning about data do have a large overlap, but the purpose of using statistics is often very different than when machine learning is used.

Machine Learning can be defined as:

• Machine learning is a field of computer science that uses statistical techniques to give computer systems the ability to “learn” with data, without being explicitly programmed. (source Wikipedia) So for example progressively improve learning performance for a specific task based on data input.

The underlying algorithms used for machine learning are essentially based around statistics methods. Machine learning is similar to the concepts around data mining. An algorithm attempts to find patterns in data to classify, predict, or uncover meaningful trends. Machine learning is often only useful if enough data is available. And if the data has been prepared correctly. So despite the promises of machine learning, when you want to apply machine learning you always have a data challenge. Getting good and large amounts of data that is usable for input of a machine learning algorithm is often not a simple problem to solve. Not only getting enough quality data, but also managing (storing, processing etc) the retrieved data is hard. Most of the time the storage and performance aspect are the easiest problems to solve regarding data. Getting good quality data is often very hard.

For machine learning, four things are needed:

1. Data. More is better.
2. A model of how to transform the data.
3. A loss function to measure how good the model is performing.
4. An algorithm to tweak the model parameters such that the loss function is minimized

Machine learning algorithms discover patterns in data, and construct mathematical models using these discoveries.

2.4.3 Overview machine learning methods

Whenever you are confronted with machine learning it is good to known that different methods, and thus approaches, exist.

At the highest level, machine learning can be categorized into the following core types:

• Supervised learning.
• Unsupervised learning.
• Reinforcement Learning.
2.4.3.1 Supervised Learning

Supervised learning addresses the task of predicting targets given input data.

Most practical business machine learning solutions use supervised learning. Supervised learning encompasses approaches to satisfy the need to classify things into categories, known as classification. It also includes approaches to address the need to provide variable real-value solutions such as weight or height known as regression.

With supervised learning the learning algorithm is given labelled data and the desired output. For example, pictures of cats labelled “cat” help the algorithm to identify the rules to classify pictures of cats.

2.4.3.2 Unsupervised Learning

The goal of this type of learning is to model data and uncover trends that are not obvious in its original state. The input data given to the learning algorithm is unlabelled, and the algorithm is asked to identify patterns in the input data.

This type of learning is used to learn about data. Unsupervised learning methods are suited for unlabelled data. It is used to find patterns where the patterns are still unknown. Unsupervised learning seems attractive since it does not require a lot of hard work of data cleaning before starting. However there are also serious challenges when applying unsupervised learning.

To name a few:

- Without a possibility to tell the machine learning algorithm what you want (like in classification), it is difficult to judge the quality of the results.
- You have to select a lot of good examples from each class while you are training the classifier. If you consider classification of big data that can be a real challenge.
- Training needs a lot of computation time, so do the classification.
- Unsupervised learning is more subjective than supervised learning, as there is no clear goal set for the analysis, such as prediction of a response.
- The order of the data can have an impact on the final results.
- Rescaling your datasets can completely change results.
In machine learning there is no single algorithm that works best for every problem. This is especially relevant for supervised learning (i.e. predictive modelling). So machine learning is a bit like cooking. You have to try some things before it fits your taste.

### 2.4.3.3 Reinforcement learning (RL)

Reinforcement Learning is close to human learning. Reinforcement learning differs from standard supervised learning in that correct input/output pairs are never presented, nor sub-optimal actions explicitly corrected. Instead the focus is on performance. Reinforcement learning can be seen as learning best actions based on reward or punishment.

Reinforcement learning (RL) is learning by interacting with an environment. An RL agent learns from the consequences of its actions, rather than from being explicitly taught and it selects its actions on basis of its past experiences (exploitation) and also by new choices (exploration), which is essentially trial and error learning.

In reinforcement learning (RL) there is no answer key, but your reinforcement learning agent still has to decide how to act to perform its task. In the absence of existing training data, the agent learns from experience. It collects the training examples ("this action was good, that action was bad") through trial-and-error as it attempts its task, with the goal of maximizing long-term reward.

RL methods are employed to address the following typical problems:

- The Prediction Problem and
- the Control Problem.

### 2.4.3.4 Deep learning (DL)

Deep Learning (DL) is an approach to machine learning which drives the current hype wave of self driving cars and more.

Deep Learning (DL) is a type of machine learning that enables computer systems to improve with experience and data. Deep learning is a subfield of machine learning.

Deep learning uses layers to progressively extract features from the raw input. For example, in image processing, lower layers may identify edges, while higher layers may identify the concepts relevant to a human such as digits or letters or faces.

Deep learning models can achieve excellent accuracy, sometimes exceeding human-level performance. Most deep learning methods use neural network architectures, which is why deep learning models are often referred to as deep neural networks.

The figure below positions Deep Learning (DL) in the spectrum of AI and ML.
### 2.4.3.5 AutoML

Of course every technology evolves continuously. So when you have mastered a bit of the machine learning concepts you will be faced with more and more machine learning innovations. The big next promising thing for machine learning is automated machine learning in short autoML.

AutoML can be defined as: the automated process of algorithm selection, hyperparameter tuning, iterative modelling, and model assessment. AutoML accelerates the model building process, the time consuming ‘human’ part within ML. So with the current machine learning we have:

\[
\text{Solution} = \text{ML expertise} + \text{data} + \text{computation}
\]

With AutoML the challenge is to turn this into:

\[
\text{Solution} = \text{data} + 100\times \text{computation}
\]

### 2.4.4 Other common terms used in the ML world

Within the world of machine learning you read and hear about concepts and terms as networks, deep learning, reinforcement learning and more. Many of these terms are derived from years of scientific progress and discussions.

#### 2.4.4.1 Data science

Data science can be defined as:

- The practice of, and methods for, reporting and decision making based on data.

So Data science is a umbrella term for several disciplines (technical and non technical) that deal with data. Even storing data in a retrievable way is a real science with many pitfalls.

#### 2.4.4.2 Generative model

A Generative model can be defined as:
• A model for generating all values for a phenomenon, both those that can be observed in the world and “target” variables that can only be computed from those observed

2.4.4.3 Neural networks (NNs)

Neural networks (NNs) can be defined as:

• The algorithms in machine learning are implemented by using the structure of neural networks. These neural networks model the data using artificial neurons. Neural networks thus mimic the functioning of the brain.

The ‘thinking’ or processing that a brain carries out is the result of these neural networks in action. A brain’s neural networks continuously change and update themselves in many ways, including modifications to the amount of weighting applied between neurons. This happens as a direct result of learning and experience.

NN are can be regarded as statistical models directly inspired by, and partially modelled on biological neural networks. They are capable of modelling and processing non-linear relationships between inputs and outputs in parallel. The related algorithms are part of the broader field of machine learning, and can be used in many applications.

Features (also called attributes): Properties of an data object to train a machine learning system. Think of features as number of colours in your street, the number of leafs on a tree, or the size of a garden. A smart selection of features is crucial to train a machine learning system.

2.4.4.4 Vision

A lot of machine learning applications work on vision. But vision for computers is different from vision for humans. Humans can not see without thinking. And when we see something our mind is continuously playing with us.

Vision for computers can be defined as:

• The ability of computers to “see” by recognizing what is in a picture or video.

2.4.4.5 Speech

One of the great things we can do with computers to create applications that transfer words to speech or when we need a lot of data transfer speech to data. Great progress has been made on automatically analysing conversations without human intervention needed.

Speech:

• the ability of computers to listen by understanding the words that people say and to transcribe them into text.

2.4.4.6 Language

Understanding each other is hard. But this is typical a field where machine learning applications, mainly NLP driven have made great progress using (new)machine learning techniques and technologies.

A definition of language as used within the machine learning field:

• The ability of computers to comprehend the meaning of the words, taking into account the many nuances and complexities of language (such as slang and idiomatic expressions).
2.4.4.7 Knowledge

Defining knowledge is hard, but crucial for many machine learning applications. An attempt to define knowledge in the context of machine learning:

Knowledge:
- The ability of a computer to reason by understanding the relationship between people, things, places, events and context.

2.4.4.8 Overfitting

Overfitting means the model fits the parameters too closely with regard to the particular observations in the training dataset, but does not generalize well to new data. Most of the time the model is too complex for the given training data.

2.4.4.9 Program synthesis

Program synthesis can automatically produce software code. Its applications range from web automation, hardware security, operating system extensions, programming for non-programmers, authoring of SQL queries, configuration management, automatic code translation, and superoptimization.

2.5 Machine Learning for Business Problems

Reading and talking about futuristic potential options for machine learning is nice and should be done. But applying machine learning today for your business is where you can make a real difference. This section is focussed on applying machine learning for real business use cases.

Example use cases that are possible with current available FOSS machine learning building blocks are outlined. And some real world business use cases where machine learning is applied are shown. This to give you inspiration and information on possible options for your business.

Be aware that besides technology more is needed for applying machine learning in a business with success. This section gives you some more in depth input on organisational factors that should be taken into account when applying machine learning for real business use.

2.5.1 When to use machine learning?

Before starting and applying machine learning for solving business problems you must be aware that machine learning is not a tool for every problem. Or to put it even more clear: In most cases applying machine learning is overkill, too expensive, does not work, and other traditional software solutions make far more sense. So the short answer for most use case is: do not use machine learning. Keep it simple! Using machine learning is a complex and risky journey and it makes your business more complex.

But the temptation to use machine learning to solve complex problems is too promising to ignore. So you should try it. Preferably by using a fast innovation project with minimal cost and no strings attached. If only try it to see if is has real some real opportunities for your use case. But be aware from the start that machine learning doesn’t give perfect answers or a perfect solution. Risk will always exist, so you should get a feeling on the likelihood of a risk occurring.

In some use cases machine learning can save you a lot of time and can make things possible that are out of reach using normal traditional software approaches.

With the use of machine learning it is possible to learn from patterns and conditions to get new solid outcomes or predictions based on new data. Machine learning is able to learn from changes in patterns (data) at a pace that the
human mind can not. This makes that machine learning as a technology is useful for a set of use cases were learning from data is possible or needed. So this is one reason why machine learning only makes sense for a limited class of use cases.

Machine learning should not be used for use cases that can easily be solved in another way. For example do not use machine learning driven solutions if your use case matches one of the following criteria:

- If it’s possible to structure a set of rules or “if-then scenarios” to handle your problem entirely, then there is usually no need to use machine learning at all.
- Your problem can be solved using traditional statistical tools(algorithms) and software.

Machine learning is an appropriate tool to use for problems whose only commonality is that they involve statistical inference. This means that problems where machine learning makes real sense have e.g. the following characteristics:

- Classification challenges. E.g., is this a picture of a cat or a gorilla? Looks this human happy? Is this person writing emotional replies on twitter?
- Clustering challenges. E.g., group all cat pictures by ones that are most similar.
- Reinforcement learning challenges. E.g., learn to predict how people behave when they book a holiday with a large discount. Are you willing to buy something you do not need without discount?

A good question to ask is: Can this problem be solved by looking at statistical outcomes? If the answer is yes, use traditional statistical software and avoid machine learning directly. Avoid complexity at all cost before trying to find if using machine learning is a viable option.

In general: All areas where there is a lot of data and too much data for manual inspection are candidates for applying machine learning.

So summarized for most business problems using machine learning should be avoided. Like blockchain or other industry IT buzzwords: Avoid the trap of using a solution and finding a problem to use it on! A particularly bad use case for machine learning is when the problem can be described using clear and precise mathematical equations. Only when a problem can not be described using clear and existing mathematical equations and an outcome can be predicted using large numbers of input data, then the use of machine learning should be considered.

When you want to apply machine learning for your business use cases you need to develop a solid architecture before starting. A standard solution for your business use case does not exist. Your company and your context is unique. So for real and significant business advantage you should also develop your own machine learning enabled application or ML powered information system. Machine learning is just a component in the complete system architecture needed. But a good and simple overall architecture when applying machine learning is needed. Especially since all developed solutions deployed in production need maintenance. In the section ‘ML Reference Architecture’ a view of the complete system architecture is given.

The usage of a Cloud (SaaS or ML-SaaS) machine learning solution will not always give you the competitive advantage you are searching for. This because standard solutions only work on standard use cases. Most use cases are unique. So if your business is special, your data is unique and your use case is unique than your own developed machine learning driven application should give you a head start and competitive advantage. In the section ‘Machine learning reference architecture’ an in depth outline is given on the various system building blocks that are needed for applying machine learning in a successful way. Make use of the machine learning reference architecture outlined in this publication to create your own ML enabled solution faster.

In order to solve business problems using machine learning technology you need to have an organisation structure that powers innovation and experimenting. Experimenting with machine learning should be simple and can be done in a short time. But this requires a business culture with an innovation approach where learning and playing with new technology is possible without predefined rules.
2.5.2 Common business use cases

2.5.2.1 Healthcare

Healthcare is due to the large amounts of data available a perfect domain for solving challenges using machine learning. E.g. a challenging question for machine learning for healthcare is: Given a patient’s electronic medical record data, can we prevent a person getting sick?

Machine learning is more and more used for automatic diagnostics. This can be data provided by X-ray scans or data retrieved from blood and tissue samples. Machine learning has already proven to be valuable in detecting and predicting diseases for real people. But beware already sensors and camera data in public spaces are used to gather data, also for healthcare related use cases without your approval.

Predictive tasks for healthcare is maybe the way to keep people healthier and lower healthcare cost. The transformation from making people better towards preventing people getting sick is long and hard, since this means a real shift for the healthcare industry.

But given a large set of training data of de-identified medical records it is already possible to predict interesting aspects of the future for a patient that is not in the training set.

Machine learning applications for healthcare are also to create better medicines by making use of all the data already available.

2.5.2.2 Language translation

Machine learning is already used for automatic real-time message translation. E.g. Rocket Chat (The OSS Slack alternative, https://rocket.chat/) is using machine learning for real time translation.

Since language translation needs context and lots of data, typically these use cases are often NLP driven. Language translation as speech recognition is a typical NLP application. Natural language processing (NLP) is area of machine learning that operates on (human)text and speech. See the section on NLP in this book for more use cases and insight in the specific NLP technologies.

Other areas for language translation are speech recognition. Some great real time machine learning driven application already exists.

When building speech recognition machine learning applications you discover that data needed for speech recognition is not quite open. To create voice systems you need an extremely large amount of voice data. Most of the data used by large companies isn’t available to the majority of people. E.g. Amazon, Microsoft and Google offer great APIs but you interact with a black-box model. Also speech recognition needs openness and freedom. Mozilla launched Common Voice project in 2017. A project to make voice recognition data and APIs open and accessible to everyone. Contributing to this great project is simple: Go to https://voice.mozilla.org/ and speak some sentences and validate some. All you need is a browser and a few minutes to contribute so everyone can make use of this technology in the future.

2.5.2.3 Chat bots

Currently all major tech companies like Amazon(Alexis), Google, Apple (Siri) have built a smart chatbot for the consumer market. Creating a chatbot (e.g. IRQ bot) was not new and difficult, however building a real ‘intelligent’ chat bot that has learning capabilities is another challenge.

Machine learning powered chatbots with real human like voices help computers communicate with humans. But algorithms still have a hard time trying to figure out what you are saying, because context and tone of voice is hard to get right. Even for us humans, communication with other humans is most of the time hard. So building a smart chatbot that understands basic emotions in your voice is difficult. Machine learning isn’t advanced enough yet to carry on a dialogue without help, so a lot of the current chatbot software needs to be hand-coded.
2.5.2.4 eCommerce Recommendation systems

A well known application for machine learning for eCommerce systems is a machine learning enabled recommendation system. Whether you buy a book, trip, music or visit a movie: On all major online ecommerce sites you get a recommendation for a product that seems to fit your interest perfectly. Of course the purpose is to drive up the sale, but these algorithms used are good examples of still evolving machine learning algorithms for recommendation systems.

Examples of these systems are:

- Selling tickets to concerts based on your profile.
- NetFlix or cinema systems to make sure you stay hooked on watching more series and films you like.
- Finding similar products in an eCommerce environment with a great chance you buy it. E.g. similar hotels, movies, books, etc.

2.5.2.5 Quality inspection and improvement

When computer vision technologies are combined with machine learning capabilities new opportunities arise. Examples of real world applications are:

- Inspecting tomatoes (and other fruit / vegetables) for quality and diseases.
- Inspecting quality of automatic created constructions (e.g. Constructions made by robots)

2.5.2.6 Vision

Since vision is captured in data machine learning is a great tool for building applications using vision (images, movies) technology. E.g.:

- Face detection. Writing software to detect faces and do recognition is very very hard to do using traditional programming methods.

- Image classification. In the old days we were happy when software was able to distinguish a cat and dog. In 2018 far more advanced applications are possible. E.g. giving details on all kind of aspects of photos. E.g. when you organize a conference you can use software to check the amount of suits or hoodies visiting your conference. Which is of course great for marketing.

- Image similarity. Given an image, the goal of an image similarity model is to find “similar” images. Just like in image classification, deep learning methods have been shown to give incredible results on this challenging problem. However, unlike in image similarity, there isn’t a need to generate labelled images for model creation. This model is completely unsupervised.

- Object Detection. Object detection is the task of simultaneously classifying (what) and localizing (where) object instances in an image.

2.5.2.7 Financial services

- Real-time trade: Like bidding sites for online advertising and stock exchange markets are more and more driven by software algorithms. Knowing this you must be a fool if you take part on a stock exchange market without the power these automated machine learning driven systems have. You will never earn anything…

- Banking and credit services: More and more banks and large financial companies are using their data to get more profit out of existing customers. Based on a smart combination of banking data and financial transaction data, your bank knows better than you can imagine how to make even more profit from their customers.
2.5.2.8 Marketing

- Marketing and acquisition: By analysing mass amounts of data you can better target your existing and potential users for your service. Machine learning makes a large difference here, as proven by Google and Facebook (both ad-service companies in essence). Analysing works using machine learning works well for consumer markets where user data and user behaviour data is widespread and for sale. And since tracking users on the internet is the number one data leak almost all data is available somewhere. Also business to business marketing is perfect to automate using machine learning. This is because also here the only input needed is often data.

Of course if you do care about privacy and embrace the values of Free and Open machine learning the marketing use cases for machine learning are almost impossible to create due to privacy issues involved.

2.5.2.9 HR services

- HR management and HR services: Finding the right new employing, talent management, performance management all tangible HR work is powered by ML software more and more. Even the scary face and voice recognition tools are used to check if your new employee matches your ideal profile. Until HR is fully automated ML powered software helps HR professionals to improve decision-making and create more efficient ways to interact with employees.

When using machine learning for HR services be aware of bias issues in using datasets. Bias when hire new personnel is for humans already difficult to handle. But you do not want a machine learning application that only selects people based on old paradigms in society.

2.5.2.10 Predicting services

- Predicting services: Almost all predicting services in all business domains can benefit from the combination of large data sets and using machine learning algorithms. E.g. you can empower predicting services by using weather data (historical and new), financial and demographic data and local production data to find out in more detail how your next sales campaign goes. But prediction is also possible on failures on production lines, where historical data is combined with sensor data.

2.5.2.11 Software

A holy grail for software developers is of course creating a machine learning algorithms that creates software for use cases that require expensive and complex human programming work.

In recent years some real progress for using machine learning for creating software is made. Use cases seen are e.g.:

- Software code improvement: Manual programming is hard and error prone. By training machine learning on a large code base to learn the model what ‘bugs’ are, it is possible to use machine learning to prevent programming bugs in new developed software code. In this way code can not be committed since the automated checks provided spotted an error. Detecting a bug before software is tested and deployed is far cheaper than correcting errors in code when a program is already released. A game development company has used this application of machine learning for real with success already. (reference http://www.wired.co.uk/article/ubisoft-commit-assist-ai)

- Creating new software programs: Based on a problem it is proven by different companies that software can be generated instead of manual crafted (programmed). Feeding a algorithm massive inputs of examples programs it is possible to generate a new program based for your specific problem. Of course this application of machine learning is still in its early phase. It is also questionable if this application of machine learning makes real sense since the new paradigm of machine learning is no longer program a solution but create a program outcome based on input data.
2.5.2.12 Security

- Email spam filters. Although simple rules can and should be applied, the enormous creativity of spammers and the amount send good fighting spam is a solid use case for a supervised machine learning problem.
- Network filtering. Due to the learning capability of machine learning network security devices are improved using machine learning techniques.
- Fraud detection. Fraud detection is possible using enormous data and searching for strange patterns.

Besides fraud detection machine learning can also applied for IT security detections since intrusion detection systems and virus scanners are more and more shipped with self learning algorithms. Also Complex financial fraud schemes can be easily detected using predictive machine learning models.

2.5.2.13 Privacy

Privacy can be protected using machine learning. E.g. images can be made invisible by using a machine learning enabled application. A scientific proof is demonstrated and the code is named 'DeepPrivacy'. In the section with a collection of Computer Vision Building Blocks for more information on this SBB.) The technique used is based on Generative Adversarial Network (GAN) for face anonymization. It’s far from perfect, but usable for most low quality images.

**Warning:** Besides protecting privacy machine learning is still too often a privacy nightmare.

2.5.2.14 Risk and compliance

- Evaluating risks can be done using large amounts of data. Natural language processing techniques can be used to validate highly automatic if your company meets regulations. Since audit and inspecting work is mostly based on standardized rules performed by knowledge workers this kind of work can be automated using machine learning techniques.
- Detecting danger and safety risks. E.g. for autonomous vehicles (robots). More and more machine learning software is developed to make transport safer for us humans.

2.5.3 Business Examples

Applications for real business use of machine learning to solve real tangible problems are growing at a rapid pace. To outline some use cases that have been realized using machine learning technology, this paragraph summarize some real world cases to get some inspiration. So in this section some worth mentioning exiting real business examples for companies that really make use of new ML solutions possible.

- Medical researchers are using machine learning to assess a person’s cardiovascular risk of a heart attack and stroke.
- Air Traffic Controllers are using TensorFlow to predict flight routes through crowded airspace for safe and efficient landings.
- Engineers are using TensorFlow to analyse auditory data in the rainforest to detect logging trucks and other illegal activities.
- Scientists in Africa are using TensorFlow to detect diseases in Cassava plants to improve yield for farmers.
- Finding free parking space. [http://www.peazy.in](http://www.peazy.in) has developed an app using machine learning to assist with finding a free parking space in crowded cities.
• All kinds of card games. With the use of the FOSS RLCard toolkit (see open ML Software section) the use of Reinforcement Learning (RL) in card games is possible.

• AI Driven Logos. An AI solution which selects the best possible logos for your brand based on a large number of designs it has seen over time. Check https://www.designwithai.com/

• Cardiac Ultrasound Software. The software, called Caption Guidance, is an accessory to compatible diagnostic ultrasound systems and uses artificial intelligence to help the user capture images of a patient’s heart that are of acceptable diagnostic quality. Approved for use by the U.S. Food and Drug Administration (FDA).

2.5.4 Business Challenges

Applying machine learning for real business use cases is complex and difficult.

Common business challenges when applying machine learning in business products or services are e.g.:

• Determining when applying machine learning is a good choice for solving a business problem.

• Getting the right data and preparation of the data to be used for training a machine learning model.

• Dealing with privacy, security and safety aspects.

• Engineering solid and maintainable machine learning applications. Designing, creating and debugging machine learning applications is specialized IT work.

• Dealing with terrible math and statistics foundations. Of course most software building blocks keep this away from you, but you must make choices that require some more in depth knowledge of the foundations behind the chosen algorithms used.

• Have access to skilled IT engineers. Not only machine learning engineers are needed, but also good engineers that are skilled in setting up IT environments. This accounts for cloud and also for on premise environments. Choices that are possible for machine learning cloud environments are often not trivial, unless you have an unlimited credit card.

The number one challenge is: How to integrate machine learning into your current business operations and products in order to really benefit from this technology?

Normal IT projects have a bad reputation. Projects are often delayed and do not deliver what was needed. Machine learning are still not different. In fact machine learning projects are still complex and risky IT projects. So an agile approach is recommended to reduce risks.

Integration of machine learning software pipelines, especially when it also involves digital integration between companies and systems of different companies is known to be hard, complex and make you poor if handled wrong. If you have a bad track record when it comes to executing traditional IT projects, machine learning projects have the same challenges with a couple of new real high risks elements.

Machine learning is not a logical and intuitive way to solve problems. For many engineers and software programmers solving problems using a machine learning approach is against the learned and trained intuition. So training and building an intuition for what tool should be leveraged to solve a problem is needed for engineers involved. At a minimum engineers involved should be aware of available machine learning algorithms and machine learning building blocks (SBBs) and the trade-offs and constraints of each one. This publication contains an overview of the typical algorithms and an overview of diverse machine learning FOSS building blocks available. This increases the insights and improves the awareness of available options.

Machine learning needs trial and error before it works well. But debugging a machine learning application is a real complex challenge. An endless number of factors must be taken into account. Not only technical but even more from a business perspective. When are risks in outcomes acceptable? You need insights in the context where the results are used in order to evaluate if machine learning results are usable enough. When you want to improve the output you can face problems e.g. the following problems:

• Is there a bug in the used software framework?
• Is the data quality below an acceptable level?
• Is the chosen algorithm the right choice?
• Are other IT issues influencing the outcome, e.g. performance?
• With machine learning finding bugs and working on optimizations is almost ‘exponentially’ harder due to the complex nature of the various aspects involved. So to figure out what is wrong when things don’t work as expected can take far more time than available.
• Are the risks for business use acceptable? For live saving systems you should make other choices than for a marketing system.

2.5.5 Business capabilities

To take advantage of machine learning your organisation needs to have or develop the needed capabilities. Before starting a proof of concept or project with machine learning you need to dive into the subject and options. Warning: Don’t fall for a vendor hype. So beware of demo’s and courses of vendors who sell you perfect SaaS ML solutions. If a promise for new business innovation based on a new machine learning application seems too good to be true: It often is.

Only you know your business systems, your requirements, your financial objectives, your customers and thus the right trade-offs to make. Good and simple tools can make the process for using machine learning easier. But tools are no magic bullet. You still need to have to integrate machine learning outcomes with your business products or services. To use the power of machine learning collaboration is needed. So the focus should also be on solving business problems and not only IT challenges.

The following capabilities are often needed to successfully apply machine learning for your business use case:

• Capability to experiment and learn. So a real learning culture.
• Managers, architects, developers and engineers with an open mindset. So open for learning and experimenting.
• Descent knowledge of key quality aspects involved. E.g. privacy, safety and security. A must take these privacy, safety and security serious from the start. Do it by design. It can initially take some extra time. But once key safeguards are in place experimenting with data and machine learning outcomes are possible with lower risks. So make sure you involve some privacy and security experts from the start.
• Solid business innovation strategy, innovation management system (process and people) available.

If your goal is to use machine learning to reduce cost by automating human workflows make sure everyone shares this goal upfront.

2.5.6 Business ethics

When machine learning algorithms make decisions that affect human lives, what standards of transparency, openness and accountability should apply to those decisions? If the decisions are “wrong”, who is legally and ethically responsible?

There are always good and bad uses for any technology. This accounts also for machine learning technology. Working with machine learning can, will and must raise severe ethical questions. Machine learning can be used in many bad ways. Saying that you ‘Don’t be evil’, like the mission statement of Google (https://en.wikipedia.org/wiki/Don%27t_be_evil) was for decades, does not save you. Any business that uses machine learning should develop a process in order to handle ethical issues before they arrive. And ethical questions will arise.

A growing number of experts believe that a third revolution occurs during the 21st century, through the invention of machines with intelligence which surpasses our own intelligence. The rapid progress in machine learning technology
turns out to be input for all kind of disaster scenarios. When the barriers to apply machine learning are lowered one of the fears is that knowledge work and various mental tasks currently performed by humans become obsolete.

When machine learning develops and the border with artificial intelligence will be approached many more philosophical and ethical discussions will take place. One of the core question is: What is human intelligence? But the more important question is: Who is responsible for mistakes? The self learning algorithm? To put it in the context of machine learning: What is the real value of human intelligence when machine learning algorithms can take over many common mental tasks and control tasks of humans? Who is responsible for accidents with autonomous vehicles?

Many experts believe that there is a significant chance we develop machines more intelligent than ourselves within a few decades. This could lead to large, rapid improvements in human welfare, or mass unemployment and poverty on a large scale. History learns that there are good reasons to think that this could lead to disastrous outcomes for our current societies. If machine learning research advances without enough research work going on security, safety on privacy, catastrophic accidents are likely to occur. Or if we look back at history: Incidents will occur since regulations are always developed afterwards with new technology.

With FOSS machine learning capabilities you should be able to take some control over the rapid pace machine learning driven software is hitting our lives. So instead of trying to stop developments and use, it is better to steer developments into a positive, safe, human centric direction. So apply machine learning using a decent machine learning architecture were also some critical ethical business questions are addressed.

Advances within machine learning could lead to extremely positive developments, presenting solutions to now-intractable global problems. But applying machine learning without good architectures where ethical questions are also addressed, using machine learning at large can pose severe risks. Humanity’s superior intelligence is the sole reason that we are the dominant species on our planet. If technology with advanced machine learning algorithms surpass humans in intelligence, then just as the fate of gorillas currently depends on the actions of humans, the fate of humanity may come to depend more on the actions of machines than our own.

To address ethical questions for your machine learning solution architecture you can use the high level framework with ethical requirements below. All requirements are of equal importance, support each other, and should be implemented and evaluated throughout the system’s lifecycle.

Some basic common ethical questions for every machine learning architecture are:

- Bias in data sets. How do you weigh this? Are you fully aware of the impact?
- Impact on your company.
- Impact on your employees.
- Impact on your customers (short and long term).
- Impact on society.
- Impact on available jobs and future manpower needed.
Free and Open Machine Learning, Release 1.0.1

- Who is responsible and who is liable when the application developed using machine learning goes seriously wrong?
- Do you and your customers find it acceptable all kinds of data are combined to make more profit?
- How transparent should you inform your customers on how privacy aspects are taken into account when using the machine learning software? Legal baselines, like the EU GDPR do not answer these ethical questions for you!
- How transparent are you towards stakeholders regarding various direct and indirect risks factors involved when applying machine learning applications?
- Who is responsible and liable when risks in your machine learning application do occur?

A lot of ethical questions come back to crucial privacy and other risks questions like safety and security. We live in a digital world where our digital traces are everywhere. Most of the time we are fully unaware. In most western countries mass digital surveillance cameras generates great data to be used for machine learning algorithms. This can be noble by detecting diseases based on cameras, but all nasty use cases thinkable are of course also under development. Continuous track and trace of civilians including face recognition is not that uncommon any more!

The question regarding who is accountable for negative effects when you use machine learning technology is simple to answer. You are! Accountability is about holding individuals and organisations responsible for how any machine learning enabled application is used. But this is not trivial: The outcome of a machine learning application system is the product of the software itself, or any single decision-maker. This is because the success or failure of a ML enabled system may be the product of one or several components. In most cases, a system failure is the result of multiple factors, and responsibility is not easily apportioned. So: If you do not understand the technology, the impact for your business and on society you should not use it.

Regulations for applying machine learning are not yet developed. Although some serious thinking is already be done in the field regarding:
- Safety and
- Liability

Many governmental bodies promote adopting a risk-adapted regulatory approach when it comes to ethical issues regarding algorithmic systems (machine learning). History learns that risks based approaches that depend on human discipline, especially in areas where safety issues are clear, are fuel for disasters waiting to happen. It makes more sense to adopt an approach that bans the human factor and risks can be calculated using long proven scientific statistical methods.

Government rules and laws are formed during the transition the coming decade. Machine learning techniques are perfect to use for autonomous weapons. So drones will in near future decide based on hopefully predefined rules when to launch a missile and when not. But as with all technologies: Failures are going to happen! And we all hope it will not hit us.

Using machine learning comes with responsibilities. These responsibilities apply for all institutions that fund, develop, and deploy ML based systems. So adopt ethical and open standards.

2.6 ML Reference Architecture

When you are going to apply machine learning for your business for real you should develop a solid architecture. A good architecture covers all crucial concerns like business concerns, data concerns, security and privacy concerns. And of course a good architecture should address technical concerns in order to minimize the risk of instant project failure.

Unfortunately it is still not a common practice for many companies to share architectures as open access documents. So most architectures you will find are more solution architectures published by commercial vendors.
Architecture is a minefield. And creating a good architecture for new innovative machine learning systems and applications is an unpaved road. Architecture is not by definition high level and sometimes relevant details are of the utmost importance. But getting details of the inner working on the implementation level of machine learning algorithms can be very hard. So a reference architecture on machine learning should help you in several ways.

Unfortunately there is no de-facto single machine learning reference architecture. Architecture organizations and standardization organizations are never the front runners with new technology. So there are not yet many mature machine learning reference architectures that you can use. You can find vendor specific architecture blueprints, but these architecture mostly lack specific architecture areas as business processes needed and data architecture needed. Also the specific vendor architecture blueprints tend to steer you into a vendor specific solution. What is of course not always the most flexible and best fit for your business use case in the long run.

In this section we will describe an open reference architecture for machine learning. Of course this reference architecture is an open architecture, so open for improvements and discussions. So all input is welcome to make it better! See section Help.

The scope and aim of this open reference architecture for machine learning is to enable you to create better and faster solution architectures and designs for your new machine learning driven systems and applications.

You should also be aware of the important difference between:

- Architecture building Blocks and
- Solution building blocks
This reference architecture for machine learning describes architecture building blocks. So you could use this reference architecture and ask vendors for input on for delivering the needed solution building blocks. However in another section of this book we have collected numerous great FOSS solution building blocks so you can create an open architecture and implement it with FOSS solution building blocks only.

Before describing the various machine learning architecture building blocks we briefly describe the machine learning process. This because in order to setup a solid reference architecture high level process steps are crucial to describe the most needed architecture needs.

Applying machine learning for any practical use case requires beside a good knowledge of machine learning principles and technology also a strong and deep knowledge of business and IT architecture and design aspects.

2.6.1 The machine learning process

Setting up an architecture for machine learning systems and applications requires a good insight in the various processes that play a crucial role. The basic process of machine learning is feed training data to a learning algorithm. The learning algorithm then generates a new set of rules, based on inferences from the data. So to develop a good architecture you should have a solid insight in:
• The business process in which your machine learning system or application is used.
• The way humans interact or act (or not) with the machine learning system.
• The development and maintenance process needed for the machine learning system.
• Crucial quality aspects, e.g. security, privacy and safety aspects.

In its core a machine learning process exist of a number of typical steps. These steps are:
• Determine the problem you want to solve using machine learning technology
• Search and collect training data for your machine learning development process.
• Select a machine learning model
• Prepare the collected data to train the machine learning model
• Test your machine learning system using test data
• Validate and improve the machine learning model. Most of the time you need is to search for more training data within this iterative loop.

You need to improve your machine learning model after the first test. Improving can be done using more training data or by making model adjustments.

### 2.6.2 Architecture Building Blocks for ML

This reference architecture for machine learning gives guidance for developing solution architectures where machine learning systems play a major role. Discussions on what a good architecture is, can be a senseless use of time. But input on this reference architecture is always welcome. This to make it more generally useful for different domains and different industries. Note however that the architecture as described in this section is technology agnostics. So it is aimed at getting the architecture building blocks needed to develop a solution architecture for machine learning complete.
Every architecture should be based on a strategy. For a machine learning system this means an clear answer on the question: What problem must be solved using machine learning technology? Besides a strategy principles and requirements are needed.

The way to develop a machine learning architecture is outlined in the figure below.

In essence developing an architecture for machine learning is equal as for every other system. But some aspects require special attention. These aspects are outlined in this reference architecture.

### 2.6.2.1 Principles for Machine learning

Every good architecture is based on principles, requirements and constraints. This machine learning reference architecture is designed to simplify the process of creating machine learning solutions.

Principles are statements of direction that govern selections and implementations. That is, principles provide a foundation for decision making. A good principle hurts. Always good and common sense principles are nice for vision documents and policy makers. But when it comes to creating tangible solutions you must have principles that steer your development.
Principles are common used within business architecture and design and successful IT projects. A simple definition of a what a principle is:

• A principle is a qualitative statement of intent that should be met by the architecture.

Every solution architecture that for business use of a machine learning application should hold a minimum set of core business principles.

Machine learning architecture principles are used to translate selected alternatives into basic ideas, standards, and guidelines for simplifying and organizing the construction, operation, and evolution of systems. In essence every good project is driven by principles. But since quality and cost aspects for machine learning driven application can have a large impact, a good machine learning solution is created based on principles.

Key principles that are used for this Free and Open Machine learning reference architecture are:

1. The most important machine learning aspects must be addressed.
2. The quality aspects: Security, privacy and safety require specific attention.
3. The reference architecture should address all architecture building blocks from development till hosting and maintenance.
4. Translation from architecture building blocks towards FOSS machine learning solution building blocks should be easily possible.
5. The machine learning reference architecture is technology agnostic. The focus is on the outlining the conceptual architecture building blocks that make a machine learning architecture.

For your use case you must make a more explicit variant of one of the above general principles.

By writing down business principles is will be easier to steer discussions regarding quality aspects of the solution you are developing. Creating principles also makes is easier for third parties to inspect designs and solutions and perform risks analysis on the design process and the product developed.

**Example Business principles for Machine Learning applications**

In this section some general principles for machine learning applications. For your specific machine learning application use the principles that apply and make them SMART. So include implications and consequences per principle.

**Collaborate**

Statement: Collaborate Rationale: Successful creation of ML applications require the collaboration of people with different expertises. You need e.g. business experts, infrastructure engineers, data engineers and innovation experts. Implications: Organisational and culture must allow open collaboration.

**Unfair bias**

Statement: Avoid creating or reinforcing unfair bias Rationale: Machine learning algorithms and datasets can reflect, reinforce, or reduce unfair biases. Recognize fair from unfair biases is not simple, and differs across cultures and societies. However always make sure to avoid unjust impacts on sensitive characteristics such as race, ethnicity, gender, nationality, income, sexual orientation, ability, and political or religious belief. Implications: Be transparent about your data and training datasets. Make models reproducible and auditable.
Built and test for safety

Statement: Built and test for safety. Rationale: Use safety and security practices to avoid unintended results that create risks of harm. Design your machine learning driven systems to be appropriately cautious. Implications: Perform risk assessments and safety tests.

Privacy by design

Statement: Incorporate privacy by design principles. Rationale: Privacy by principles is more than being compliant with legal constraints as e.g. EU GDPR. It means that privacy safeguards, transparency and control over the use of data should be taken into account from the start. This is a hard and complex challenge.

2.6.2.2 Constraints

Important constraints for a machine learning reference architecture are the aspects:

- Business aspects (e.g. capabilities, processes, legal aspects, risk management)
- Information aspects (data gathering and processing, data processes needed)
- Machine learning applications and frameworks needed (e.g. type of algorithm, easy of use)
- Hosting (e.g. compute, storage, network requirements but also container solutions)
- Security, privacy and safety aspects
- Maintenance (e.g. logging, version control, deployment, scheduling)
- Scalability, flexibility and performance

2.6.3 ML Reference Architecture

A full stack approach is needed to apply machine learning. A full stack approach means that in order to apply machine learning successfully you must be able to master or at least have a good overview of the complete technical stack. This means for machine learning vertical and horizontal. With vertical we mean from hardware towards machine learning enabled applications. With horizontal we mean that the complete tool chain for all process steps must be taken into account.

The machine learning reference model represents architecture building blocks that can be present in a machine learning solution. Information architecture (IT) and especially machine learning is a complex area so the goal of the metamodel below is to represent a simplified but usable overview of aspects regarding machine learning. Using this model gives you a head start when developing your specific machine learning solution.
Conceptual overview of machine learning reference architecture

Since this simplified machine learning reference architecture is far from complete it is recommended to consider e.g. the following questions when you start creating your solution architecture where machine learning is part of:

- Do you just want to experiment and play with some machine learning models?
- Do you want to try different machine learning frameworks and libraries to discover what works best for your use case? Machine learning systems never work directly. You need to iterate, rework and start all over again. Its innovation!
- Is performance crucial for your application?
- Are human lives direct or indirect dependent of your machine learning system?

In the following sections more in depth description of the various machine learning architecture building blocks are given.

### 2.6.3.1 Business Processes

To apply machine learning with success it is crucial that the core business processes of your organization that are affected with this new technology are determined. In most cases secondary business processes benefit more from machine learning than primary processes. Think of marketing, sales and quality aspects that make your primary business processes better.
2.6.3.2 Business Services

Business services are services that your company provides to customers, both internally and externally. When applying machine learning for business use you should create a map to outline what services are impacted, changed or disappear when using machine learning technology. Are customers directly impacted or will your customer experience indirect benefits?

2.6.3.3 Business Functions

A business function delivers business capabilities that are aligned to your organization, but not necessarily directly governed by your organization. For machine learning it is crucial that the information that a business function needs is known. Also the quality aspects of this information should be taken into account. To apply machine learning it is crucial to know how information is exactly processed and used in the various business functions.

2.6.3.4 People, Skills and Culture

Machine learning needs a culture where experimentation is allowed. When you start with machine learning you and your organization need to build up knowledge and experience. Failure is going to happen and must be allowed. Fail hard and fail fast. Take risks. However your organization culture should be open to such a risk based approach. IT projects in general fail often, so doing an innovative IT project using machine learning is a risk that must be able to cope with.

To make a shift to a new innovative experimental culture make sure you have different types of people directly and indirectly involved in the machine learning project. Also make use of good temporary independent consultants. So consultants that have also a mind set of taking risks and have an innovative mindset. Using consultants for machine learning of companies who sell machine learning solutions as cloud offering do have the risk that needed flexibility in an early stage is lost. Also to be free on various choices make sure you are not forced into a closed machine learning SaaS solution too soon. Since skilled people on machine learning with the exact knowledge and experience are not available you should use creative developers. Developers (not programmers) who are keen on experimenting using various open source software packages to solve new problems.

2.6.3.5 Business organization

Machine learning experiments need an organization that stimulate creativity. In general hierarchical organizations are not the perfect placed where experiments and new innovative business concepts can grow.

Applying machine learning in an organization requires an organization that is data and IT driven. A perfect blueprint for a 100% good organization structure does not exist, but flexibility, learning are definitely needed. Depending on the impact of the machine learning project you are running you should make sure that the complete organization is informed and involved whenever needed.

2.6.3.6 Partners

Since your business is properly not Amazon, Microsoft or Google you need partners. Partners should work with you together to solve your business problems. If you select partners pure doing a functional aspect, like hosting, data cleaning ,programming or support and maintenance you miss the needed commitment and trust. Choosing the right partners for your machine learning project is even harder than for ordinary IT projects, due to the high knowledge factor involved. Some rule of thumbs when selecting partners: Big partners are not always better. With SMB partners who are committed to solve your business challenge with you governance structures are often easier and more flexible. Be aware of vendor lock-ins. Make sure you can change from partners whenever you want. So avoid vendor specific and black-box approaches for machine learning projects. Machine learning is based on learning, and learning requires openness.
Trust and commitment are important factors when selecting partners. Commitment is needed since machine learning projects are in essence innovation projects that need a correct mindset. Use the input of your created solution architecture to determine what kind of partners are needed when. E.g. when your project is finished you need stability and continuity in partnerships more than when you are in an innovative phase.

### 2.6.3.7 Risk management

Running machine learning projects involves risk. Within your architecture it is crucial to address business and projects risks early. Especially when security, privacy and safety aspects are involved mature risks management is recommended. To make sure your machine learning project is not dead at launch, risk management requires a flexible and creative approach for machine learning projects. Of course when your project is more mature openness and management on all risks involved are crucial. To avoid disaster machine learning projects it is recommended to create your:

- solution architecture using:
  - Safety by design principles.
  - Security by design principles and
  - Privacy by design principles

In the beginning this slows down your project, but doing security/privacy or safety later as ‘add-on’ requirements is never a real possibility and takes exponential more time and resources.

### 2.6.3.8 Development tools

In order to apply machine learning you need good tools to do e.g.:

- Create experiments for machine learning fast.
- Create a solid solution architecture
- Create a data architecture
- Automate repetitive work (integration, deployment, monitoring etc)

Fully integrated tools that cover all aspects of your development process (business design and software and system design) are hard to find. Even in the OSS world. Many good architecture tools, like Arch for creating architecture designs are still usable and should be used. A good overview for general open architecture tools can be found here [https://nocomplexity.com/architecture-playbook/](https://nocomplexity.com/architecture-playbook/).

Within the machine learning domain the de-facto development tool to use is ‘The Jupyter Notebook’. The Jupyter notebook is an web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text. A Jupyter notebook is perfect for various development steps needed for machine learning suchs as data cleaning and transformation, numerical simulation, statistical modelling, data visualization and testing/tuning machine learning models. More information on the Jupyter notebook can be found here [https://jupyter.org/](https://jupyter.org/).

But do not fall in love with a tool too soon. You should be confronted with the problem first, before you can evaluate what tool makes your work more easy for you.

### 2.6.3.9 Machine learning Frameworks

Machine Learning frameworks offer software building blocks for designing, training and validating your machine learning model. Most of the time you are only confronted with your chosen machine learning framework when using a high level programming interface. All major FOSS machine learning frameworks offer APIs for all major
programming languages. Almost all ‘black magic’ needed for creating machine learning application is hidden in a various software libraries that make a machine learning framework.

In another section of this book a full overview of all major machine learning frameworks are presented. But for creating your architecture within your specific context choosing a machine learning framework that suits your specific use case is a severe difficult task. Of course you can skip this task and go for e.g. Tensorflow in the hope that your specific requirements are offered by simple high level APIs.

Some factors that must be considered when choosing a machine learning framework are:

- Stability. How mature, stable is the framework?
- Performance. If performance really matters a lot for your application (training or production) doing some benchmark testing and analysis is always recommended.
- Features. Besides the learning methods that are supported what other features are included? Often more features, or support for more learning methods is not better. Sometimes simple is enough since you don’t change your machine learning method and model continuously.
- Flexibility. How easy is it to switch to another machine learning framework, learning method or API?
- Transparency. Machine learning development is a very difficult tasks that involve a lot of knowledge of engineers and programmers. Not many companies have the capabilities to create a machine learning framework. But in case you use a machine learning framework: How do you know the quality? Is it transparent how it works, who has created it, how it is maintained and what your business dependencies are!
- License. Of course we do not consider propriety machine learning frameworks. But do keep in mind that the license for a machine learning framework matters. And make sure that no hooks or dual-licensing tricks are played with what you think is an open machine learning Framework.
- Speeding up time consuming and recurrent development tasks.

Debugging a machine learning application is no fun and very difficult. Most of the time you spend time with model changes and retraining. But knowing why your model is not working as well as expected is a crucial task that should be supported by your machine learning framework.

There are too many open source machine learning frameworks available which enables you to create machine learning applications. Almost all major OSS frameworks offer engineers the option to build, implement and maintain machine learning systems. But real comparison is a very complex task. And the only way to do some comparison is when machine learning frameworks are open source. And since security, safety and privacy should matter for every use case there is no viable alternative than using a mature OSS machine learning framework.

### 2.6.3.10 Programming Tools

You can use every programming language for developing your machine learning application. But some languages are better suited for creating machine learning applications than others. The top languages for applying machine learning are:

- Python.
- Java and
- R

The choice of the programming language you choice depends on the machine learning framework, the development tools you want to use and the hosting capabilities you have. For fast iterative experimentation a language as Python is well suited. And besides speeds for running your application in production also speed for development should be taken into concern.

There is no such thing as a ‘best language for machine learning’.
There are however bad choices that you can make. E.g. use a new development language that is not mature, has no rich toolset and no community of other people using it for machine learning yet.

Within your solution architecture you should justify the choice you make based upon dependencies as outlined in this reference architecture. But you should also take into account the constraints that account for your project, organisation and other architecture factors that drive your choice. If have e.g. a large amount of Java applications running and all your processes and developers are Java minded, you should take this fact into account when developing and deploying your machine learning application.

### 2.6.3.11 Data

Data is the heart of the machine earning and many of most exciting models don’t work without large data sets. Data is the oil for machine learning. Data is transformed into meaningful and usable information. Information that can be used for humans or information that can be used for autonomous systems to act upon.

In normal architectures you make a clear separation when outlining your data architecture. Common view points for data domains are: business data, application data and technical data For any machine learning architecture and application data is of utmost importance. Not all data that you use to train your machine learning model needs can be originating from you own business processes. So sooner or later you need to use data from other sources. E.g. photo collections, traffic data, weather data, financial data etc. Some good usable data sources are available as open data sources. For a open machine learning solution architecture it is recommended to strive to use open data. This since open data is most of the time already cleaned for privacy aspects. Of course you should take the quality of data in consideration when using external data sources. But when you use data retrieved from your own business processes the quality and validity should be taken into account too.

Free and Open Machine learning needs to be feed with open data sources. Using open data sources has also the advantage that you can far more easily share data, reuse data, exchange machine learning models created and have a far easier task when on and off boarding new team members. Also cost of handling open data sources, since security and privacy regulations are lower are an aspect to take into consideration when choosing what data sources to use.

For machine learning you need ‘big data’. Big data is any kind of data source that has one the following properties:

- Big data is data where the volume, velocity or variety of data is (too) great. So big is really a lot of data!
- The ability to move that data at a high Velocity of speed.
- An ever-expanding Variety of data sources.
- Refers to technologies and initiatives that involve data that is too diverse, fast-changing or massive for conventional technologies, skills and infra- structure to address efficiently.

Every Machine Learning problem starts with data. For any project most of the time large quantities of training data are required. Big data incorporates all kinds of data, e.g. structured, unstructured, metadata and semi-structured data from email, social media, text streams, images, and machine sensors (IoT devices).

Machine learning requires the right set of data that can be applied to a learning process. An organization does not have to have big data in order to use machine learning techniques; however, big data can help improve the accuracy of machine learning models. With big data, it is now possible to virtualize data so it can be stored in the most efficient and cost-effective manner whether on-premises or in the cloud.

Within your machine learning project you need to perform data mining. The goal of data mining is to explain and understand the data. Data mining is not intended to make predictions or back up hypotheses.

One of the challenges with machine learning is to automate knowledge to make predictions based on information (data). For computer algorithms everything processed is just data. Only you know the value of data. What data is value information is part of the data preparation process. Note that data makes only sense within a specific context.

The more data you have, the easier it is to apply machine learning for your specific use case. With more data, you can train more powerful models.
Some examples of the kinds of data machine learning practitioners often engage with:

- **Images:** Pictures taken by smartphones or harvested from the web, satellite images, photographs of medical conditions, ultrasounds, and radiologic images like CT scans and MRIs, etc.
- **Text:** Emails, high school essays, tweets, news articles, doctor’s notes, books, and corpora of translated sentences, etc.
- **Audio:** Voice commands sent to smart devices like Amazon Echo, or iPhone or Android phones, audio books, phone calls, music recordings, etc.
- **Video:** Television programs and movies, YouTube videos, cell phone footage, home surveillance, multi-camera tracking, etc.
- **Structured data:** Webpages, electronic medical records, car rental records, electricity bills, etc
- **Product reviews** (on Amazon, Yelp, and various App Stores)
- **User-generated content** (Tweets, Facebook posts, StackOverflow questions)
- **Troubleshooting data** from your ticketing system (customer requests, support tickets, chat logs)

When developing your solution architecture be aware that data is most of the time:

- Incorrect and
- Useless.

So meta data and quality matters. Data only becomes valuable when certain minimal quality properties are met. For instance if you plan to use raw data for automating creating translating text you will discover that spelling and good use of grammar do matter. So the quality of the data input is an import factor of the quality of the output. E.g. automated Google translation services still struggle with many quality aspects, since a lot of data captures (e.g. captured text documents or emails) are full of style, grammar and spell faults.

Data science is a social process. Data is generated by people within a social context. Data scientists are social people who do a lot of communication with all kind of business stakeholders. Data scientist should not work in isolation because the key thing is to find out what story is told within the data set and what import story is told over the data set.

### 2.6.3.12 Data Tools

Without data machine learning stops. For machine learning you deal with large complex data sets (maybe even big data) and the only way to making machine learning applicable is data cleaning and preparation. So you need good tools to handle data.

The number of tools you need depends of the quality of your data sets, your experience, development environment and other choice you must make in your solution architecture. But a view use cases where good solid data tools certainly help are:

- Data visualization and viewer tools; Good data exploration tools give visual information about the data sets without a lot of custom programming.
- Data filtering, data transformation and data labelling;
- Data anonymizer tools;
- Data encryption / decryption tools
- Data search tools (analytics tools)

Without good data tools you are lost when doing machine learning for real. The good news is: There are a lot of OSS data tools you can use. Depending if you have raw csv, json or syslog data you need other tools to prepare the dataset. The challenge is to choose tools that integrate good in your landscape and save you time when preparing your data for starting developing your machine learning models. Since most of the time when developing machine learning
applications you are fighting with data, it is recommended to try multiple tools. Most of the time you experience that a mix of tools is the best option, since a single data tool never covers all your needs. So leave some freedom within your architecture for your team members who deal with data related work (cleaning, preparation etc).

The field of ‘data analytics’ and ‘business intelligence’ is a mature field for decades within IT. So you will discover that many FOSS tools that are excellent for data analytics. But keep in mind that the purpose of fighting with data for machine learning is in essence only for data cleaning and feature extraction. So be aware of ‘old’ tools that are rebranded as new data science tools for machine learning. There is no magic data tool preparation of data for machine learning. Sometimes old-skool unix tool like awk or sed just do the job simple and effective.

Besides tools that assist you with preparing the data pipeline, there are also good (open) tools for finding open datasets that you can use for your machine learning application. See the reference section for some tips.

To prepare your data working with the data within your browser seems a nice idea. You can visual connect data sources and e.g. create visuals by clicking on data. Or inspecting data in a visual way. There is however one major drawback: Despite the great progress made on very good and nice looking JavaScript frameworks for visualization, handling data within a browser DOM still takes your browser over the limit. You can still expect hang-ups, indefinitely waits and very slow interaction. At least when not implemented well. But implementation of on screen data visualisation (Drag-and-Drop browser based) is requires an architecture and design approach that focus on performance and usability from day 1. Unfortunately many visual web based data visualization tools use an generic JS framework that is designed from another angle. So be aware that if you try to display all your data, it eats all your resources(CPU, memory) and you get a lot of frustration. So most of the time using a Jupyter Notebook is a safe choice when preparing your data sets.

2.6.3.13 Hosting

Hosting infrastructure is the platform that is capable of running your machine learning application(s). Hosting is a separate block in this reference architecture to make you aware that you must make a number of choices. These choices concerning hosting your machine learning application can make or break your machine learning adventure.

It is a must to make a clear distinguishing in:

1. Hosting infrastructure needed for development and training and
2. Hosting infrastructure needed for production

Depending on your application it is e.g. possible that you need a very large and costly hosting infrastructure for development, but you can do deployment of your trained machine learning model on e.g. a Raspberry PI or Arduino board.

Standard hosting capabilities for machine learning are not very different as for ‘normal’ IT services. Expect scalability and flexibility capabilities require solid choices from the start. The machine learning hosting infrastructure exist e.g. out of:

- Physical housing and power supply.
- Operating system (including backup services).
- Network services.
- Availability services and Disaster recovery capabilities.
- Operating services e.g. deployment, administration, scheduling and monitoring.

For machine learning the cost of the hosting infrastructure can be significant due to performance requirements needed for handling large datasets and training your machine learning model.

A machine learning hosting platform can make use of various commercial cloud platforms that are offered(Google, AWS, Azure, etc). But since this reference architecture is about Free and Open you should consider what services you to use from external Cloud Hosting Providers (CSPs) and when. The crucial factor is most of the time cost and the
number of resources needed. To apply machine learning it is possible to create your own machine learning hosting platform. But in reality this is not always the fastest way if you have not the required knowledge on site.

All major Cloud hosting platforms do offer various capabilities for machine learning hosting requirements. But since definitions and terms differ per provider it is hard to make a good comparison. Especially when commercial products are served instead of OSS solutions. So it is always good to take notice of:

- Flexibility (how easy can you switch from your current vendor to another?).
- Operating system and APIs offered. And
- Hidden cost

For experimenting with machine learning there is not always a direct need for using external cloud hosting infrastructure. It all depends on your own data center capabilities. In a preliminary phase even a very strong gaming desktop with a good GPU can do.

When you want to use machine learning you need a solid machine learning infrastructure. Hosting Infrastructure done well requires a lot of effort and is very complex. E.g. providing security and operating systems updates without impacting business applications is a proven minefield.

For specific use cases you can not use a commodity hosting infrastructure of a random cloud provider. First step should be to develop your own machine learning solution architecture. Based on this architecture you can check what capabilities are needed and what the best way is to start.

The constant factor for machine learning is just as with other IT systems: Change. A machine learning hosting infrastructure should be stable. Also a machine learning hosting infrastructure should be designed as simple as possible. This since the following characteristics apply:

- A Machine learning hosting environment must be secured since determining the quality of the outcome is already challenging enough.
- Machine learning infrastructure hosting that works now for your use cases is no guarantee for the future. Your use case evolves in future and hosting infrastructure evolves also. At minimum security patches are needed. But a complete hosting infrastructure is not replaced or drastically changed on a frequent basis. The core remains for a long period.
- Incorporating new technology and too frequent changes within your hosting infrastructure can introduce security vulnerabilities and unpredictable outcomes.
- Changes on your machine learning hosting infrastructure do apply on your complete ML pipeline.
- Machine learning hosting infrastructure components should be hardened. This means protecting is needed for accidentally changes or security breaches.
- Separation of concerns is just as for any IT architecture a good practice.

So to minimize the risks make sure you have a good view on all your risks. Your solution architecture should give you this overview, including a view of all objects and components that will be changed (or updated) sooner or later. Hosting a machine learning application is partly comparable with hosting large distributed systems. And history learns that this can still be a problem field if not managed well. So make sure what dependencies you accept regarding hosting choices and what dependencies you want to avoid.

2.6.3.14 Containers

Understanding container technology is crucial for using machine learning. Using containers within your hosting infrastructure can increase flexibility or if not done well decrease flexibility due to the extra virtualization knowledge needed.
The advantage and disadvantages of the use of Docker or even better Kubernetes or LXD or FreeBSD jails should be known. However is should be clear: Good solid knowledge of how to use and manage a container solution so it benefits you is hard to get.

Using containers for developing and deploying machine learning applications can make life easier. You can also be more flexible towards your cloud service provider or storage provider. Large clusters for machine learning applications deployed on a container technology can give a great performance advantage or flexibility. All major cloud hosting providers also allow you to deploy your own containers. In this way you can start small and simple and scale-up when needed.

Summarized: Container solutions for machine learning can be beneficial for:

- Development. No need to install all tools and frameworks.
- Hosting. Availability and scalability can be solved using the container infrastructure capabilities.
- Integration and testing. Using containers can simplify and ease a pipeline needed to produce quality machine learning application from development to production. However since the machine learning development cycle differs a bit from a traditional CICD (Continuous Integration - Continuous Deployment) pipeline, you should outline this development pipeline to production within your solution architecture in detail.

2.6.3.15 GPU - CPU or TPU

Machine learning requires a lot of calculations. Not so long ago very large (scientific) computer cluster were needed for running machine learning applications. However due to the continuous growth of power of ‘normal’ consumer CPUs or GPUs this is no longer needed.

GPUs are critical for many machine learning applications. This because machine learning applications have very intensive computational requirements. GPUs are general better equipped for some massive number calculation operations that the more generic CPUs.

A way this process is optimized is by using GPUs instead of CPUs. However the use of GPUs that are supported by the major FOSS ML frameworks, like Pytorch is limited. Only Nvidia GPUs are supported by CUDA.

CUDA (Compute Unified Device Architecture) is a parallel computing platform and application programming interface (API) model created by Nvidia. It allows software to use a CUDA-enabled graphics processing of NVIDIA. So it is a proprietary standard.

An alternative for CUDA is OpenCL. OpenCL (Open Computing Language) is a framework for writing programs that execute across heterogeneous platforms. OpenCL (https://opencv.org/opencl/) has a growing support in terms of hardware and also ML frameworks that are optimized for this standard.

You might have read and heard about TPUs. A tensor processing unit (TPU) is an AI accelerator application-specific integrated circuit (ASIC). First developed by Google specifically for neural network machine learning. But currently more companies are developing TPUs to support machine learning applications.

Within your solution architecture you should be clear on the compute requirements needed. Some questions to be answered are:

- Do you need massive compute requirements for training your model?
- Do you need massive compute requirements for running of your trained model?

In general training requires far more compute resources than is needed for production use of your machine learning application. However this can differ based on the used machine learning algorithm and the specific application you are developing.

Many machine learning applications are not real time applications, so compute performance requirements for real time applications (e.g. real time facial recognition) can be very different for applications where quality and not speed is more important. E.g. weather applications based on real time data sets.
2.6.3.16 Storage

Machine learning needs a lot of data. At least when you are training your own model. E.g. medical, scientific or geological data, as well as imaging data sets frequently combine petabyte scale storage volumes.

Storing data on commercial cloud storage becomes expensive. If not for storage than the network cost involved when data must be connected to different application blocks are high.

If you are using very large data sets you will dive into the world of NoSQL storage and cluster solutions. E.g. Hadoop is an open source software platform managed by the Apache Software Foundation that has proven to be very helpful in storing and managing vast amounts of data cheaply and efficiently.

The bad news is that the number of open (FOSS) options that are really good for unstructured (NoSQL) storage is limited.

Some examples:

- Riak® KV is a distributed NoSQL key-value database with advanced local and multi-cluster replication that guarantees reads and writes even in the event of hardware failures or network partitions. Riak is written in erlang so by nature very stable. Use for big data in ml data pipelines (https://riak.com/index.html).

2.7 Security, Privacy and Safety

2.7.1 Introduction

This section outlines security, privacy and safety concerns that keep you awake when applying machine learning for real business use.

The complexity of ML technologies has fuelled fears that machine learning applications causes harm in unforeseen circumstances, or that they are manipulated to act in harmful ways. Think of a self driving car with its own ethics or algorithms that make prediction based on your personal data that really scare you. E.g. Predicting what diseases hit you based on data from your grocery store.

As with any technology: Technology is never neutral. You have to think before starting what values you implicitly use to design your new technology. All technology can and will be misused. But it is up to the designers to think of the risks when technology will be misused. On purpose or by accident.

Machine learning systems should be operated reliably, safely and consistently. Not only under normal circumstances but also in unexpected conditions or when they are under attack for misuse.

Machine learning software differs from traditional software because:

- The outcome is not easily predictable.
- The used trained models are a black box, with very few options for transparency.
- Logical reasoning (or cause and effect) is not present. Predictions are made based on statistical number crunching complex algorithms which are non linear.
- Both Non IT people and trained IT people have a hard time figuring out machine learning systems, due to the new paradigms in use.

What makes security and safety more than normal aspects for machine learning driven applications is that by design neural networks are not designed to to make the inner workings easy to understand for humans and quality and risk managers.

Without a solid background in mathematics and software engineering evaluating the correct working of most machine learning application is impossible for security researchers and safety auditors.
However more and more people dependent on the correct outcome of decisions made by machine learning software. So we should ask some critical questions:

- Is the system making any mistakes?
- How do you know what alternatives were considered?
- What is the risk of trusting the outcome blind?

Understanding how output produced by machine learning software is created will make more people comfortable with self-driving cars and other safety critical systems that are machine learning enabled. In the end systems that can kill you must be secure and safe to use. So how do we get the process and trust chain to a level that we are not longer depended of:

- Software bugs
- Machine learning experts
- Auditors
- A proprietary certification process that end with a stamp (if paid enough)

From other sectors, like finance or oil industry we know that there is no simple solution. However regarding the risks involved only FOSS machine learning applications have the right elements needed to start working on processes that give enough trust to use machine learning system for society at large.

To reduce risks for machine learning systems needed is:

- Transparency: ML systems should be understandable. However they will never be. Computer science is a complex field. Only a fraction of the people are able to grasp the complete working of software and hardware in modern computer systems. So we need to find ways to manage and reduce risks in order to trust systems enabled by ML software. Transparency can be realized by using FOSS software (for everything). But beware that real trust requires that anyone with the needed expertise should be able to rebuild the software and retrain also the created machine learning model using the same training input. In open science for machine learning this is now becoming the new de-facto standard for scientific research.

- Reproducible. All data and created models must be available so other research can verify the working independently.

Trusts means no security by obscurity. So open research, open science, open software and open business innovation principles should be used when machine learning applications are developed and deployed.

### 2.7.2 Security

Using machine learning technology gives some serious new threats. More and more new ways for exploiting the technology are published. IT security is proven to be hard and complex to control and manage. But machine learning technology makes the problem of IT security even worse. This is due to the fact that the special created machine learning exploits are very hard to determine.

Machine learning challenges many current security measurements. This because machine learning software:

- Lowers the cost of applying current known attacks on all devices which depend on software. So almost all modern technology devices.
- Machine learning software enables the easy creation of new threats and vulnerabilities on existing systems. E.g. you can take the CVE security vulnerability database (https://www.cvedetails.com/) and train a machine learning model how to create attack on the published omissions.
- When machine learning software will be in hospitals, traffic control systems, chemical fabrics and IoT devices machine learning gives easier options to create a complete new attack surface as with traditional software.
Security aspects for machine learning accounts for the application where machine learning is used, but also for the developed algorithms self. So machine learning security aspects are divided into the following categories:

- Machine learning attacks aimed to fool the developed machine learning systems. Since machine learning is often a ‘black-box’ these attacks are very hard to determine.
- System attacks special for machine learning systems. Machine learning offers new opportunities to break existing traditional software systems.
- Machine learning usage threats. The outcome of many machine learning systems is far from correct. If you base decisions or trust on machine learning applications you can make serious mistakes. This accounts e.g. for self driving vehicles, health care systems and surveillance systems. Machine learning systems are known for producing racially biased results often caused by using biased data sets. Think about problematic forms of “profiling” based on surveillance cameras with face detection.
- Machine learning hosting and infrastructure security aspects. This category is not special for machine learning but is relevant for all IT systems. Protecting ‘normal’ software solutions was already a known challenge. But inspecting and protecting machine learning systems require besides already deep knowledge of cyber security also knowledge of nature of machine learning systems. And remember: Machine learning systems are not traditional software systems. A machine learning systems is a complete other paradigm that requires new knowledge of building a thread model to take measurements to reduce security risks. When manipulated training data is used when training your machine learning model it make results horrible and can be dangerous.

So key threads for machine learning system can be seen as:

- Attacks which compromise confidentiality
- Attacks which compromise integrity by manipulation of input.
- ‘Traditional’ attacks that have impact on availability.

Attack vectors for machine learning systems can be categorized in:

- Input manipulation
- Data manipulation
- Model manipulation
- Input extraction
- Data extraction
- Model extraction
- Environmental attacks (so the IT system used for hosting the machine learning algorithms and data)

Taxonomy and terminology of machine learning is not yet fully standardized. The US NIST publication 8269 (The National Institute of Standards and Technology) a taxonomy and terminology of Adversarial Machine Learning is proposed. See https://csrc.nist.gov/publications/detail/nistir/8269/draft. Adversarial Machine Learning (AML) introduces additional security challenges in training and testing (inference) phases of system operations. AML is concerned with the design of ML algorithms that can resist security challenges, the study of the capabilities of attackers, and the understanding of attack consequences.

### 2.7.2.1 Top Machine Learning Security Risks

- Adversarial attacks: The basic idea is to fool a machine learning system by providing malicious input that cause the system to make a false prediction or categorization.
- Data poisoning: Machine learning systems learn directly from data. Intentionally manipulated data can compromise the machine learning application. If you want to make yourself e.g. invisible for face recognition you can create or buy special clothes.
• Data confidentiality: An unique challenge in machine learning is protecting confidential data.

• Data trustworthiness: Data integrity is essential. Are the data suitable and of high enough quality to support machine learning? Are e.g. sensors to capture data reliable? How is data integrity preserved? Understanding machine learning data sources, both during training and during execution, is of critical importance.

• Overfitting Attacks: Overfitting means the model fits the parameters too closely with regard to the particular observations in the training dataset, but does not generalize well to new data. Most of the time the model is too complex for the given training data. Overfit models are particularly easy to attack.

• Output integrity. If an attacker can interpose between a machine learning system and produced output, a direct attack on output is possible. The inscrutability of machine learning models (so not really understanding how they work) may make an output integrity attack easy and hard to spot.

Some examples of machine learning exploits:

• Google’s Cloud Computing service can be tricked into seeing things that are not there. In one test it perceived a rifle as a helicopter.

• Fake videos made with help from machine learning software are spreading online, and the law can’t do much about it. E.g. videos with speeches given by political leaders created by machine learning software are created and spread online. E.g. a video where some president declares a war to another country is of course very dangerous. Even more dangerous is the fact that the fake machine learning created videos are very hard to diagnose as machine learning creations. This since besides machine learning a lot of common Hollywood special effects are also used to make it hard to distinguish real videos from fake video’s. Creating online fake porn video sites were you can use a photo of a celebrity or someone you do not like, is nowadays only just three mouse clicks away. And the reality is that you can do very little against these kinds of damaging threads. Even from a legal point of view.

Users and especially developers of machine learning applications must be more paranoid from a security point of view. But unfortunately security cost a lot of effort and money and a lot of special expertise is needed to minimize the risks.

2.7.3 Privacy

Machine learning raises serious privacy concerns since machine learning is using massive amounts of data that contain often personal information.

It is a common believe that personal information is needed for experimenting with machine learning before you can create good and meaningful applications. E.g. for health applications, travel applications, eCommerce and of course marketing applications. Machine learning models are often loaded with massive amounts of personal data for training and to make in the end good meaningful predictions.

The belief that personal data is needed for machine learning creates a tension between developers and privacy aware consumers. Developers want the ability to create innovative new products and services and need to experiment, while consumers and GDPR regulators are concerned for the privacy risks involved.

The applicability of machine learning models is hindered in settings where the risk of data leakage raises serious privacy concerns. Examples of such applications include scenarios where clients hold sensitive private information, e.g., medical records, financial data, or location.

It is commonly believed that individuals must provide a copy of their personal information in order for AI to train or predict over it. This belief creates a tension between developers and consumers. Developers want the ability to create innovative products and services, while consumers want to avoid sending developers a copy of their data.

Machine learning models can be trained in environments that are not secure on data it never has access to. Secure machine learning that works on anonymized data sets is still an obscure and unpaved path. But some companies and organizations are already working on creating deep learning technology that works on encrypted data. Using encryption on data to train machine learning models raises the complexity in various ways. It is already hard or impossible to understand the inner working of the ‘black-box’ machine learning models. Using advanced data encryption will
require even more knowledge and competences for all engineers involved when developing machine learning applications.

In the EU the use of personal data is protected by law in all countries by a single law. The EU General Data Protection Regulation (GDPR). This GDPR does not prohibit the use of machine learning. But when you use personal data you will have a severe challenge to explain to DPOs (Data Protection Officers) and consumers what you actually do with the data and how you comply with the GDPR.

Machine learning systems must be data responsible. They should use only what they need and delete it when it is no longer needed (“data minimization”). They should encrypt data in transit and at rest, and restrict access to authorized persons (“access control”). Machine learning systems should only collect, use, share and store data in accordance with privacy and personal data laws and best practices. Since FOSS machine learning needs full transparency and reproducibility using private data should be avoided if possible.

When you apply machine learning for your business application you should consider the following questions:

- In what way will your customers be happy with their data usage for their and your benefit?
- Do you really have a clear and good overview of all GDPR implications when using personal data in your machine learning model? What happens if you invite other companies to use your model?
- What are the ethical concerns when using massive amounts of data of your customers to develop new products? Is the way you use the data to train your model congruent with your business vision and moral?
- What are the privacy risks involved for your machine learning development chain and application?

Since security and privacy is complex to apply, frameworks are being developed to make this challenge easier. E.g. Tensorflow Encrypted aims to make privacy-preserving deep learning simple and approachable, without requiring expertise in cryptography, distributed systems, or high-performance computing. And PySyft is a Python library for secure, private Deep Learning. More on both frameworks can be found in the section on open ML software.

### 2.7.4 Safety

Machine learning is a powerful tool for businesses. But it can also lead to unintended and dangerous consequences for users of systems powered by machine learning software. The cause of safety issues is linked to the people and data that train and deploy the machine learning software and systems. Everyone involved in creating machine learning based systems should be aware of possible safety risks come when using machine learning technology.

Safety is a multifaceted area of research, with many sub-questions in areas such as reward learning, robustness, and interpretability.

A machine learning driven system can currently only be as good as the data it is given to work with. However you almost can never traceback to the data that was used to train and develop the system. This makes that the safety aspect should be kept in mind when dealing with security aspects for systems that deal direct or indirect with humans.

To avoid dangerous bias or incorrect actions from systems, you should develop machine learning system in the open and make the everything reproducible from the start.

However safety risks will always be there: It is impossible to cover all perspectives and variables for a machine learning system in development before it is released. And the nature of machine learning systems means that the outcome of machine learning is never perfect. Risks will always be present. So not all use cases possible for machine learning are acceptable from an ethical point of view.

The following activities will reduce safety risks and increase reliability of machine learning systems:

- Systematic evaluation: So evaluate the data and models used to train and operate machine learning based products and services.
- Create processes for solid documenting and auditing operations.
• Involve domain experts. Involvement of domain experts in the design process and operation of machine learning systems. Also involve real people in advance who are in the end targeted by outcomes of ml systems especially when decisions about people are made using machine learning applications.

• Evaluation of when and how a machine learning system should seek human input during critical situations, and how a system controlled by a human in a manner that is meaningful and intelligible.

• A robust feedback mechanism so that users can report issues they experience.

2.8 Natural Language Processing

2.8.1 Introduction

A real tangible and still the most applied use case for machine learning is natural language processing (NLP). Many businesses use cases for machine learning are based on information. This can be input or processed information. In fact most business use cases that can benefit of machine learning have nothing to do with images or video. Most business do have information of customers in digital form available and want to use this data to develop more value added services for their customers.

Everything that has to do with text processing and involves machine learning can be categorized as Natural Language Processing (NLP).

NLP is concerned with programming computers to process natural language. NLP is at the intersection of computer science, machine learning and linguistics. The most innovative form of NLP are applications that use using the latest machine learning technologies to derive meaning from human languages. NLP is something that’s been part of our lives for decades. In fact, consumers from across the globe interact with NLP on a daily basis, without even realizing
it. But with more FOSS machine learning building blocks available even with the latest machine learning algorithm, also NLP innovation is growing rapidly.

NLP technology is important for scientific, economic, social, and cultural reasons. Computers that can communicate with humans as humans do are a holy grail. Including understanding context and emotions. Due to infinite number of cultures solving this problem has proven to be hard. Communication is not only verbal. Nonverbal is the significant part of communication. Nonverbal communication is the nonlinguistic transmission of information through visual, auditory, tactile, and kinesthetic (physical) channels.

NLP is experiencing rapid growth as its theories and methods are deployed in a variety of new machine learning technologies.

More and more NLP techniques are used products that have serious impact on our daily lives. A misinterpreted pronounced word like ‘stop’ can have several meanings when used within an autonomous driving car. Maybe you just meant to warn your children instead of stopping the vehicle on a dangerous intersection.

Creating good NLP based applications using machine learning is hard. A simple test that gives an indication of the quality is to use a the sentence “Buffalo buffalo Buffalo buffalo buffalo buffalo Buffalo buffalo”. This sentence is correct. But many NLP algorithms and applications cannot handle this very well. Similar sentences exist in other languages. We humans are by nature good with complicated linguistic constructs, but many NLP algorithms still fail with this simple example. Some background information on this sentence can be found on Wikipedia (https://en.wikipedia.org/wiki/Buffalo_buffalo_Buffalo_buffalo_buffalo_buffalo_buffalo_Buffalo_buffalo)

2.8.2 Basic NLP functions

NLP is able to do all kinds of basic text functions. A short overview:

- Text Classification (e.g. document categorization).
- Finding words with the same meaning for search.
- Understanding how much time it take to read a text.
- Understanding how difficult it is to read a text.
- Identifying the language of a text.
- Generating a summary of a text.
- Finding similar documents.
- Identifying entities (e.g., cities, people, locations) in a text.
- Translating a text.
- Text Generation.

Machine learning capabilities have proven to be very useful for improving typical NLP based use cases. This is due to the fact that text in digital format is widely available and machine learning algorithms typically perform good on large amounts of data.

NLP papers and NLP software comes with a typical terminology:

- Tokenizer: Splitting the text into words or phrases.
- Stemming and lemmatization. Normalizing words so that different forms map to the canonical word with the same meaning. For example, “running” and “ran” map to “run.”
- Entity extraction: Identifying subjects in the text.
- Part of speech detection: Identifying text as a verb, noun, participle, verb phrase, and so on.
- Sentence boundary detection: Detecting complete sentences within paragraphs of text.
To create NLP enabled applications you need to set up a ‘pipeline’ for the various software building blocks. For each step in a NLP development pipeline another FOSS building block can be needed. The figure below shows a typical NLP pipeline.

In the figure below a typical NLP architecture for providing input on common user questions. A lot of Bot systems (Build–operate–transfer) or FAQ answering systems are created with no machine learning algorithms at all. Often simple keyword extraction is done and a simple match in a database is performed. More state-of-the-art NLP systems make intensive use of machine learning algorithms. The general rule is: If a application should be user friendly and value added than learning algorithms are needed, since it is no longer possible to program output based on given input.

2.8.3 NLP Business challenges

When using NLP technology to extract information and insight from text, the starting point is typically the raw documents stored on websites, unstructured documents and structured documents. Also the fact that documents are stored
in a variety of formats, like PDF, MSword, TIFFs make that the time needed before text can be send towards an algorithm long and often manual intensive. Even the most advanced web scraping techniques (software to store raw text of websites) is manual intensive. Unstructured text must be structured first before using these text for a NLP driven application is possible.

Privacy is a large concern when dealing with documents. To comply with the GDPR (in EU) using text with personal information of e.g. customers for other purposes is often not allowed without explicit permission of the owners of the personal data.

2.8.4 NLP Business Examples

NLP enabled application make a good fit for businesses where analysing or processing text is done at large. Below some example NLP business use cases.

- Social media monitoring: Almost all social media monitoring tools are basically built using NLP technology. These tools help to monitor social media channels for mentions of your brand, and alert you when consumers are talking about your brand. Real time monitoring of social media channels is important for companies for a lot of large companies, e.g. to ensure that any potential crises is noticed immediately.

- Sentiment analysis: Sentiment analysis is a smaller subset of social media monitoring. It refers to monitoring the social media landscape and listening in on conversations and identifying opinions and determining whether the author of the post holds a positive, negative, or neutral opinion towards a brand. Using NLP sentiment analysis tools is it easy to filter emotionally-charged words that are used to describe a brand or a customer’s experience with a brand. It is also possible to automate research on how consumers speak about a product or service.

- Text analysis: Text analysis can be broken into several sub-categories e.g. grammatical, syntactic and semantic analyses. By analysing text and extracting different types of key elements (such as topics, people, dates, locations, companies), it is easier to organize data and identify useful patterns and insights.

- Survey analytics: If you are analysing large amounts of surveys you can use old-skool reporting and statistical techniques. But a NLP based analyse makes it possible to faster and easier find patterns, categories and anomalies that are hard to find manual in survey responses with more than 20.000 responses.

- Spam filters: Emails that contain text and words such as “free”, “promotion”, “buy now”, ‘coin’, ‘offer’ , etc, have a high chance of being spam. With the use of NLP technology you can create self learning spam filters. Sometimes too good.

- Autocomplete: With an autocomplete function it is possible to predicts what the next characters or words will be that you will enter. This makes the use of text based UI simpler.

- Hiring Tools: Large companies still do receive many resumes, analysing can be time-consuming and the task of sorting overwhelming. Natural language processing software is able to speed up the process of sorting resumes. But be aware NLP software is not flawless. So you will miss good candidates if you only scan resumes.

- Conversational Search: Traditional search often brings not the results you want. Wit conversational search the context of your search is taken into account. So only for the current context relevant search items will be presented. Since searching quality information is time consuming , streamlining search in real-time conversation NLP software will improve productivity for knowledge workers.

2.9 ML implementation challenges

Machine learning is a technology that currently comes with some special challenges. This section outlines the most important challenges you hit when you start using machine learning for real.
2.9.1 Performance

Machine learning is amazingly slow. So before you use cloud services for your machine learning performance challenges, it is wise to do some experiments on a simple laptop. After you have a feeling of your model and the main performance bottlenecks than using clusters of advantaged GPUs offered by all Cloud providers is more effective. Also from a cost perspective.

Special machine learning hardware is at its infancy. Even faster systems and wider deployment lead to many more breakthroughs across a wide range of domains.

Many ML OSS software solutions are created in the Python programming language. Python is considered ‘slow’ and hard to parallelize. However many solutions exist to solve this problem. The most important OSS machine learning software solutions are by design capable to run on complete clusters of GPU (Graphical Processor Units). Scalability over GPU has proven to be more efficient for machine learning calculations than using more CPUs. This is because GPUs are by design more suited for the complex calculations needed to perform than CPUs. GPUs are better for speeding up calculations that are needed for distributed machine learning algorithms. A GPU core is designed for
handling multiple tasks simultaneously. A core for a GPU is simpler than a CPU core, but a GPU has many more cores than a CPU.

New CPUs are being developed especially for machine learning algorithms. E.g. Google is developing Tensor Processing Units (TPU’s) to speed up the machine learning calculations. Of course this TPU’s are optimized for Google’s tensorflow software. But since this software is OSS everyone can take advantages if needed, since Google offers TPU’s in their Cloud offerings. Of course Microsoft and other big Cloud providers are also developing their specialized machine learning processing units.

Performance for training machine learning solutions is not always simple to solve. This is due to the fact that in essence training machine learning models means doing mass matrix calculations. Despite the fact that Python is a good choice for machine learning, processing large calculations can be slow. So optimization can be needed to speed up pre-processing during the data preparation phase. The good news is that since Python is becoming the de-facto standard for machine learning almost all problems are known and often already solved.

### 2.9.2 Testing machine learning models

Automated testing is a large part of software development. Unfortunately performing testing on software and infrastructure is still a mandatory requirement for solid ML projects. Once a project reaches a certain level of complexity, the only way that it can be maintained is if it has a set of tests that identify the main functionality and allow you to verify that functionality is intact. Without automatic tests, it’s nearly impossible to identify where errors are occurring, and to fix those errors without causing further problems.

Testing should be done on:

- Data (training data)
- Infrastructure
- Software QA aspects. In fact all ISO QA factors should be evaluated.
- Security, Privacy and safety aspects.

Overview ISO Quality Standard(25010)

The ISO documents are not open. But since quality matters and ISO 25010 is used heavily for managing quality aspects within business IT systems you should keep these factors in mind when developing test to minimize business risks.

Data testing for ML pipelines is different and can be complex. Data preparation and testing for machine learning is not comparable with data testing for traditional IT projects. This is because it requires a statistical test performed on the data set. If input data changes this can have a significant effect on the outcome. Good statistical practice is an essential component of good scientific practice but also for real world ML applications. Especially when safety aspects play a
role. Also mind that ML can be in essence is still seen as applied statistics. Validation of outcomes using statistical methods is proven science.

Good statistical practice is an essential component of good scientific practice but also for real world ML applications. Especially when safety aspects play a role. Also mind that ML can be in essence is still seen as applied statistics.

In statistical hypothesis testing, the p-value or probability value is the probability of obtaining test results at least as extreme as the results actually observed during the test. The p-value was never intended to be a substitute for scientific reasoning. Well-reasoned statistical arguments contain much more than the value of a single number and whether that number exceeds an arbitrary threshold. So for evaluating machine learning results the principles of American Statistical Association (ASA) can be useful. These principles are:

- P-values can indicate how incompatible the data are with a specified statistical model.
- P-values do not measure the probability that the studied hypothesis is true, or the probability that the data were produced by random chance alone.
- Scientific conclusions and business or policy decisions should not be based only on whether a p-value passes a specific threshold.
- Proper inference requires full reporting and transparency.
- A p-value, or statistical significance, does not measure the size of an effect or the importance of a result.
- By itself, a p-value does not provide a good measure of evidence regarding a model or hypothesis.

Machine learning projects tend to be pretty under-tested, which is unfortunate because they have all of the same complexity and maintainability issues as software projects.

- Tests codify your expectations at the point when you actually remember what you're expecting.
- They allow you to offload verification to a machine.
- You can make changes to your code with confidence.

2.9.3 Interoperability

Open standards do help. And with open standards you should look for real open standards. There are standards that not only specify how things should work, but also have an open source implementation that is using the standards for real. Keep away from standards that exist on paper only or standards that only have a reference implementation. Good standards are used and born from a practical need.

In this way everyone implementing the standards is forced to make sure the outcome and use of APIs is the same. Most of the time open standards lack an open implementation, so vendors can implement the specification and still lock you in.

With interoperability for machine learning a trained model can be reused using different frameworks for an application. A trained model is the result of applying a machine learning algorithm to a set of training data. A new model using an already trained model can make predictions based on new input data. For example, a model that’s been trained on a region’s historical house prices may be able to predict a house’s price when given the number of bedrooms and bathrooms.

Currently de facto standards on machine learning are just emerging. But due to all tools offered for applying machine learning it makes sense that models can be reused between machine learning frameworks. Open Neural Network Exchange (ONNX) is the first step toward an open ecosystem that empowers AI developers to choose the right tools as their project evolves. ONNX provides an open source format for AI models. It defines an extensible computation graph model, as well as definitions of built-in operators and standard data types. Caffe2, PyTorch, Microsoft Cognitive Toolkit, Apache MXNet and other tools are developing ONNX support. Enabling interoperability between different frameworks and streamlining the path from research to production will increase the speed of innovation for ML applications. See: http://onnx.ai/ for more information.
A standard that is already for many years (first version in 1998) available is the PMML standard. This Predictive Model Markup Language (PMML) is an XML-based predictive model interchange format. However, many disadvantages exist that seem to prevent PMML from becoming a real interoperability standard for ML. (See http://dmg.org/pmml/v4-3/GeneralStructure.html)

Besides standards on interoperability for use of machine learning frameworks, you need some standardization on datasets first. The good news is that raw datasets are often presented in a standard format like CSV, JSON, or XML. In this way, some reuse of data is already possible. But knowing the data pipeline needed for machine learning more is needed. E.g., currently, there is no standard way to identify how a dataset was created, and what characteristics, motivations, and potential skews it represents. Some answers that a good standardized metadata description on data should provide are e.g.:

- Why was the dataset created?
- What (other) tasks could the dataset be used for?
- Has the dataset been used for any tasks already?
- Who funded the creation of the dataset?
- Are relationships between instances made explicit in the data?
- What preprocessing/cleaning was done?
- Was the “raw” data saved in addition to the preprocessed/cleaned data?
- Under what license can the data be (re)used?
- Are there privacy or security concerns related to the content of the data?

2.9.4 Debugging

Machine learning is a fundamentally hard debugging problem. Debugging for machine learning is needed when:

- your algorithm doesn’t work or
- your algorithm doesn’t work well enough.

What is unique about machine learning is that it is ‘exponentially’ harder to figure out what is wrong when things don’t work as expected. Compounding this debugging difficulty, there is often a delay in debugging cycles between implementing a fix or upgrade and seeing the result. Very rarely does an algorithm work the first time and so this ends up being where the majority of time is spent in building algorithms.

2.9.5 Continuous improvements

Machine learning models degrade in accuracy in production. This since the new input data is different from the used training data. Input data changes over time. This problem of the changes in the data and relationships within data sets is called concept drift.

Machine learning models are not a typical category of software. In fact, a machine learning model should not be regarded as software at all. This means that maintenance should be organized and handled in a different way. There is never a final version of a machine learning model. So when using machine learning, you need engineers that continuously update and improve the model.

So setting up end user feedback, accuracy measurements, monitoring data trends are important factors for organizations when using machine learning. But the traditional IT maintenance task as monitoring servers, network and infrastructure, security threats and application health is also still needed.
2.9.6 Maturity of ML technology

Machine Learning is moving from the realm of universities and hard core data science into a technology that can be integrated for mainstream application for every business. However machine learning technology is not yet idiot proof. Many algorithms are not used for real world applications on a large scale. Also many machine learning building blocks are still in heavy development. Of course in near future machine learning applications will never be idiot proof, since this is the nature of current machine learning technologies. But acceptable margins for normal errors and disasters are not yet solid predictable at the start of a project.

But thanks to the development of many quality OSS machine learning building blocks and platforms doing a Proof of Concept becomes within reach for every business.

FOSS Machine learning still needs a lot of boring work that is invisible but crucial for overall quality. The boring work is avoided at most universities and most companies choose the easy path towards commercial offerings. But for high value FOSS machine learning applications everyone who shares the principles for FOSS ML can and should contribute to the foundation work needed for machine learning.

2.9.7 Data and bias

Machine learning is only as good as the data used for training. So too often machine learning applications are biased based. This is is a consequence of the used input.

In general almost all development time is spent on data related tasks. E.g. prepare data to be used as training data and manual classification.

Data is selecting is expensive and complex since often privacy aspects are involved.

“Garbage-in, garbage-out” is too often true for machine learning applications. The “black box” algorithms of machine learning prevents understanding why a certain output is seen. Often input data was not appropriate, but determining the root cause of the problem data is a challenge.
Bias is a problem that relates to output seen and has a root cause in the used input data set. Biased data sets are not representative, have skewed distribution, or contain other quality issues. Biased training data input results in biased output that makes machine learning application useless.

Dealing with unwanted bias in data is a challenging pitfall to avoid when using recommendations of algorithms. Bias challenges are playing out in health care, in hiring, credit scoring, insurance, and criminal justice.

When evaluating outcomes of machine learning applications there are many ways you can be fooled. Common data quality aspects to be aware of are:

- Cherry picking: Only results that fit the claim are included.
- Survivorship bias: Drawing conclusions from an incomplete set of data, because that data has survived the selection criteria.
- False causality: Falsely assuming when two events appear related that one must have caused the other.
- Sampling bias: drawing conclusions from a set of data that isn’t representative of the population you are trying to understand.
- Hawthorne effect: The act of monitoring someone affects their behaviour, leading to spurious findings. Also known as the observer effect.
- MCNamara fallacy: Relying solely on metrics in complex situations and losing sight of the bigger picture.

Machine learning can be easily susceptible to attacks and notoriously difficult to control. Some people are collecting public information regarding machine learning disasters and unethical applications in practice. A few examples:

- AI-based Gaydar - Artificial intelligence can accurately guess whether people are gay or straight based on photos of their faces, according to new research that suggests machines can have significantly better "gaydar" than humans.
- Infer Genetic Disease From Your Face - DeepGestalt can accurately identify some rare genetic disorders using a photograph of a patient’s face. This could lead to payers and employers potentially analyzing facial images and discriminating against individuals who have pre-existing conditions or developing medical complications. [Nature Paper]
- Racist Chat Bots - Microsoft chatbot called Tay spent a day learning from Twitter and began spouting anti-semitic messages.
- Social Media Propaganda - The Military is studying and using data-driven social media propaganda to manipulate news feeds in order to change the perceptions of military actions.
- Infer Criminality From Your Face - A program that judges if you’re a criminal from your facial features.

For the complete list and more examples, see: https://github.com/daviddao/awful-ai

Data quality and problems to get your data quality right before starting should be your greatest concern when starting with machine learning with a goal to develop a real production application.

### 2.9.8 Quality of Machine Learning frameworks

Only a few people understand the complex mathematical algorithms behind machine learning. History learns that implementing an algorithms into software correctly has proven to be very complex and difficult. When you use FOSS machine learning software you have one large advantage over commercial ‘black-box’ software: You can inspect the software or ask an IT consultancy company to provide a quality audit.

The recent years there is a continuous growth of open machine learning tools and frameworks. Determining which toolkits are good enough for your business case is not trivial.

A simple checklist to start with this challenge:

- A clear description of the used mathematical model and algorithm used must be available.
• All source code, including all dependencies, including external libraries must be available for download and specified.
• A test suite so you can analyse the machine learning framework (time, sample size) of the algorithm should be available.
• A healthy open community should be active around the framework and eco-system. A healthy FOSS community has a written way of working, so it is transparent how governance of the software is arranged.
• Openness: It should be transparent why people and companies contribute to the FOSS machines learning software.

2.10 Building Blocks for FOSS ML

This section presents the most widespread, mature and promising open source ML software available. The purpose of this section is just to make you curious to maybe try something that suits you.

ML software comes in many different forms. A lot can be written on the differences on all packages below, the quality or the usability. Truth is however there is never one best solution. Depending your practical use case you should make a motivated choice for what package to use.

As with many other evolving technologies in heavy development: Standards are still lacking, so you must ensure that you can switch to another application with minimal pain involved. By using a real open source solution you already have taken the best step! Using OSS makes you are far more independent than using ML cloud solutions. This because these work as ‘black-box’ solutions and by using OSS you can always build your own migration interfaces if needed. Lock-in for ML is primarily in the data and your data cleansing process. So always make sure that you keep full control of all your data and steps involved in the data preparation steps you follow. The true value for all ML solutions are of course always the initial data sources used.

2.11 Open Machine Learning Frameworks

There are a number of stable and production ready ML frameworks. But choosing which framework to use depends on the use case. If you want to experiment with the latest research insights implemented you will make another choice than if you need to implement your solution in production into a critical environment. For business use: So doing innovation experiments and creating machine learning application most of the time you want a framework that is stable and widely used.

If you have an edge use case experimenting with different frameworks can be a valid choice.

PyTorch is dominating the research, but is now extending this success to industry applications. TensorFlow is already used for many production business cases. But as it is with all software: Transitions from major versions (from TensorFlow 1.0 to 2.0) is difficult. Interoperability standards to easily switch from ML framework are not mature for production use yet.

2.12 ML Frameworks

Choosing a machine learning (ML) framework or library to solve your use case is easier said than done. Selecting a ML Framework involves making an assessment to decide what is right for your use case. Several factors are important for this assessment for your use case. E.g.:

• Easy of use;
• Support in the market. Some major FOSS ML Frameworks are supported by many consultancy firms. But maybe community support using mailing lists or internet forums is sufficient to start.

• Short goal versus long term strategy. Doing fast innovation tracks means the cost for starting from scratch again should be low. But if you directly focus on a possible production deployment, whether on premise or using cloud hosting this can significantly delay startup time. Often it is recommended to experiment fast and in a later phase take new requirements like maintenance and production deployment into account.

• Research of business use case. Some ML frameworks are focussed on innovation and research. If your company is not trying to develop a better ML algorithms this may not be the best ML framework for experimenting for business use cases.

• Closed (Commercial) dependencies. Some FOSS frameworks have a dependency with a commercial data collection. E.g. many translation frameworks need an API key of Google or AWS to function. All costs aspects of these dependencies should be taken into account before starting. There is nothing wrong with using commercial software, but transparency on used data sets and models can be crucial for acceptance of your machine learning application.

A special-purpose framework may be better at one aspect than a general-purpose. But the cost of context switching is high:

  • different languages or APIs
  • different data formats
  • different tuning tricks

Your first model for experimenting should be about getting the infrastructure and development tools right. Simple models are usually interpretable. Interpretable models are easier to debug. Complex model erode boundaries beware of the CACE principle (CACE principle: Changing Anything Changes Everything)

### 2.12.1 Acme

Acme is a library of reinforcement learning (RL) agents and agent building blocks. Acme strives to expose simple, efficient, and readable agents, that serve both as reference implementations of popular algorithms and as strong baselines, while still providing enough flexibility to do novel research. The design of Acme also attempts to provide multiple points of entry to the RL problem at differing levels of complexity.

Overall Acme strives to expose simple, efficient, and readable agent baselines while still providing enough flexibility to create novel implementations.

<table>
<thead>
<tr>
<th>SBB License</th>
<th>Apache License 2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Python</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="https://github.com/deepmind/acme">https://github.com/deepmind/acme</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/deepmind/acme">https://github.com/deepmind/acme</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>ML Framework</td>
</tr>
</tbody>
</table>

### 2.12.2 AdaNet

AdaNet is a lightweight TensorFlow-based framework for automatically learning high-quality models with minimal expert intervention. AdaNet builds on recent AutoML efforts to be fast and flexible while providing learning guaran-
Importantly, AdaNet provides a general framework for not only learning a neural network architecture, but also for learning to ensemble to obtain even better models.

This project is based on the *AdaNet algorithm*, presented in “AdaNet: Adaptive Structural Learning of Artificial Neural Networks" at ICML 2017, for learning the structure of a neural network as an ensemble of subnetworks.

AdaNet has the following goals:

- **Ease of use**: Provide familiar APIs (e.g. Keras, Estimator) for training, evaluating, and serving models.
- **Speed**: Scale with available compute and quickly produce high quality models.
- **Flexibility**: Allow researchers and practitioners to extend AdaNet to novel subnetwork architectures, search spaces, and tasks.
- **Learning guarantees**: Optimize an objective that offers theoretical learning guarantees.


---

### 2.12.3 Analytics Zoo

Analytics Zoo provides a unified analytics + AI platform that seamlessly unites *Spark, TensorFlow, Keras and BigDL* programs into an integrated pipeline; the entire pipeline can then transparently scale out to a large Hadoop/Spark cluster for distributed training or inference.

- **Data wrangling and analysis using PySpark**
- **Deep learning model development using TensorFlow or Keras**
- **Distributed training/inference on Spark and BigDL**
- **All within a single unified pipeline and in a user-transparent fashion!**

---

<table>
<thead>
<tr>
<th>SBB License</th>
<th>Apache License 2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Python</td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/tensorflow/adanet">https://github.com/tensorflow/adanet</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>ML, ML Framework</td>
</tr>
</tbody>
</table>

---


5 [https://icml.cc/Conferences/2017](https://icml.cc/Conferences/2017)
2.12.4 Apache MXNet

Lightweight, Portable, Flexible Distributed/Mobile Deep Learning with Dynamic, Mutation-aware Dataflow Dep Scheduler; for Python, R, Julia, Scala, Go, Javascript and more.

All major GPU and CPU vendors support this project, but also the real giants like Amazon, Microsoft, Wolfram and a number of very respected universities. So watch this project or play with it to see if it fits your use case.

Apache MXNet (incubating) is a deep learning framework designed for both efficiency and flexibility. It allows you to mix symbolic and imperative programming\(^6\) to maximize efficiency and productivity. At its core, MXNet contains a dynamic dependency scheduler that automatically parallelizes both symbolic and imperative operations on the fly. A graph optimization layer on top of that makes symbolic execution fast and memory efficient. MXNet is portable and lightweight, scaling effectively to multiple GPUs and multiple machines.

MXNet is also more than a deep learning project. It is also a collection of blue prints and guidelines\(^7\) for building deep learning systems, and interesting insights of DL systems for hackers.

Gluon is the high-level interface for MXNet. It is more intuitive and easier to use than the lower level interface. Gluon supports dynamic (define-by-run) graphs with JIT-compilation to achieve both flexibility and efficiency. The perfect starters documentation with a great crash course on deep learning can be found here: https://d2l.ai/index.html An earlier version of this documentation is still available on: http://gluon.mxnet.io/\(^\star\)\(^\star\)

Part of the project is also the the Gluon API specification (see https://github.com/gluon-api/gluon-api)

The Gluon API specification (Python based) is an effort to improve speed, flexibility, and accessibility of deep learning technology for all developers, regardless of their deep learning framework of choice. The Gluon API offers a flexible interface that simplifies the process of prototyping, building, and training deep learning models without sacrificing training speed.

<table>
<thead>
<tr>
<th>SBB License</th>
<th>Apache License 2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>CPP</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="https://mxnet.apache.org/">https://mxnet.apache.org/</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/apache/incubator-mxnet">https://github.com/apache/incubator-mxnet</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>ML, ML Framework</td>
</tr>
</tbody>
</table>

2.12.5 Apache Spark MLlib

Apache Spark MLlib. MLlib is Apache Spark’s scalable machine learning library. MLlib is a Spark subproject providing machine learning primitives. MLlib is a standard component of Spark providing machine learning primitives on top of Spark platform.

Apache Spark is a FOSS platform for large-scale data processing. The Spark engine is written in Scala and is well suited for applications that reuse a working set of data across multiple parallel operations. It’s designed to work as a standalone cluster or as part of Hadoop YARN cluster. It can access data from sources such as HDFS, Cassandra or Amazon S3.

MLlib can be seen as a core Spark’s APIs and interoperates with NumPy in Python and R libraries. And Spark is very fast! MLlib ships with Spark as a standard component.

MLlib library contains many algorithms and utilities, e.g.:

---
\(^6\) https://mxnet.incubator.apache.org/architecture/index.html#deep-learning-system-design-concepts
\(^7\) https://mxnet.incubator.apache.org/architecture/index.html#deep-learning-system-design-concepts
- Classification: logistic regression, naive Bayes.
- Regression: generalized linear regression, survival regression.
- Decision trees, random forests, and gradient-boosted trees.
- Recommendation: alternating least squares (ALS).
- Clustering: K-means, Gaussian mixtures (GMMs).
- Topic modeling: latent Dirichlet allocation (LDA).
- Frequent item sets, association rules, and sequential pattern mining.

Using Spark MLlib gives the following advantages:
- Excellent scalability options
- Performance
- User-friendly APIs
- Integration with Spark and its other components

But using Spark means that also the Spark platform must be used.

<table>
<thead>
<tr>
<th>SBB License</th>
<th>Apache License 2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Java</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="https://spark.apache.org/mllib/">https://spark.apache.org/mllib/</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/apache/spark">https://github.com/apache/spark</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>ML, ML Framework</td>
</tr>
</tbody>
</table>

### 2.12.6 auto_ml

Automated machine learning for analytics & production.

Automates the whole machine learning process, making it super easy to use for both analytics, and getting real-time predictions in production.

Unfortunate unmaintained currently, but still worth playing with.

<table>
<thead>
<tr>
<th>SBB License</th>
<th>MIT License</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Python</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="http://auto-ml.readthedocs.io">http://auto-ml.readthedocs.io</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/ClimbsRocks/auto_ml">https://github.com/ClimbsRocks/auto_ml</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>ML, ML Framework</td>
</tr>
</tbody>
</table>
2.12.7 BigDL

BigDL is a distributed deep learning library for Apache Spark; with BigDL, users can write their deep learning applications as standard Spark programs, which can directly run on top of existing Spark or Hadoop clusters.

- **Rich deep learning support.** Modeled after Torch\(^8\), BigDL provides comprehensive support for deep learning, including numeric computing (via Tensor\(^9\)) and high level neural networks\(^{10}\); in addition, users can load pre-trained Caffe\(^{11}\) or Torch\(^{12}\) or Keras\(^{13}\) models into Spark programs using BigDL.

- **Extremely high performance.** To achieve high performance, BigDL uses Intel MKL\(^{14}\) and multi-threaded programming in each Spark task. Consequently, it is orders of magnitude faster than out-of-box open source Caffe\(^{15}\), Torch\(^{16}\) or TensorFlow\(^{17}\) on a single-node Xeon (i.e., comparable with mainstream GPU).

- **Efficiently scale-out.** BigDL can efficiently scale out to perform data analytics at “Big Data scale”, by leveraging Apache Spark\(^{18}\) (a lightning fast distributed data processing framework), as well as efficient implementations of synchronous SGD and all-reduce communications on Spark.

<table>
<thead>
<tr>
<th>SBB License</th>
<th>Apache License 2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Java</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="https://bigdl-project.github.io/master/">https://bigdl-project.github.io/master/</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/intel-analytics/BigDL">https://github.com/intel-analytics/BigDL</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>ML, ML Framework</td>
</tr>
</tbody>
</table>

2.12.8 Blocks

Blocks is a framework that is supposed to make it easier to build complicated neural network models on top of Theano\(^{19}\).

Blocks is a framework that helps you build neural network models on top of Theano. Currently it supports and provides:

- Constructing parametrized Theano operations, called “bricks”
- Pattern matching to select variables and bricks in large models
- Algorithms to optimize your model
- Saving and resuming of training
- Monitoring and analyzing values during training progress (on the training set as well as on test sets)
- Application of graph transformations, such as dropout

---

\(^{8}\) http://torch.ch/
\(^{9}\) https://github.com/intel-analytics/BigDL/tree/master/spark/dl/src/main/scala/com/intel/analytics/bigdl/tensor
\(^{10}\) https://github.com/intel-analytics/BigDL/tree/master/spark/dl/src/main/scala/com/intel/analytics/bigdl/nn
\(^{11}\) http://caffe.berkeleyvision.org/
\(^{12}\) http://torch.ch/
\(^{13}\) https://faroit.github.io/keras-docs/1.2.2/
\(^{14}\) https://software.intel.com/en-us/intel-mkl
\(^{15}\) http://caffe.berkeleyvision.org/
\(^{16}\) http://torch.ch/
\(^{17}\) https://www.tensorflow.org/
\(^{18}\) http://spark.apache.org/
\(^{19}\) http://www deeplearning.net/software/theano/
### 2.12.9 Caffe

Caffe is a deep learning framework made with expression, speed, and modularity in mind. It is developed by Berkeley AI Research (BAIR\textsuperscript{20})/The Berkeley Vision and Learning Center (BVLC) and community contributors.

Caffe is an Open framework, models, and worked examples for deep learning:
- 4.5 years old
- 7,000+ citations, 250+ contributors, 24,000+ stars
- 15,000+ forks, >1 pull request / day average at peak

Focus has been vision, but also handles, reinforcement learning, speech and text.

Why Caffe?
- **Expressive architecture** encourages application and innovation. Models and optimization are defined by configuration without hard-coding. Switch between CPU and GPU by setting a single flag to train on a GPU machine then deploy to commodity clusters or mobile devices.
- **Extensible code** fosters active development. In Caffe’s first year, it has been forked by over 1,000 developers and had many significant changes contributed back. Thanks to these contributors the framework tracks the state-of-the-art in both code and models.
- **Speed** makes Caffe perfect for research experiments and industry deployment. Caffe can process over 60M images per day with a single NVIDIA K40 GPU\textsuperscript{*}. That’s 1 ms/image for inference and 4 ms/image for learning and more recent library versions and hardware are faster still. We believe that Caffe is among the fastest convnet implementations available.

<table>
<thead>
<tr>
<th>SBB License</th>
<th>MIT License</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Python</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="http://blocks.readthedocs.io/en/latest/">http://blocks.readthedocs.io/en/latest/</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/mila-udem/blocks">https://github.com/mila-udem/blocks</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>ML, ML Framework</td>
</tr>
</tbody>
</table>

### 2.12.10 ConvNetJS

ConvNetJS is a Javascript library for training Deep Learning models (Neural Networks) entirely in your browser. Open a tab and you’re training. No software requirements, no compilers, no installations, no GPUs, no sweat.

\textsuperscript{20} http://bair.berkeley.edu
ConvNetJS is a Javascript implementation of Neural networks, together with nice browser-based demos. It currently supports:

- Common **Neural Network modules** (fully connected layers, non-linearities)
- Classification (SVM/Softmax) and Regression (L2) **cost functions**
- Ability to specify and train **Convolutional Networks** that process images
- An experimental **Reinforcement Learning** module, based on Deep Q Learning

For much more information, see the main page at convnetjs.com

Note: Not actively maintained, but still useful to prevent reinventing the wheel.

<table>
<thead>
<tr>
<th>SBB License</th>
<th>MIT License</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Javascript</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="https://cs.stanford.edu/people/karpathy/convnetjs/">https://cs.stanford.edu/people/karpathy/convnetjs/</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/karpathy/convnetjs">https://github.com/karpathy/convnetjs</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>Javascript, ML, ML Framework</td>
</tr>
</tbody>
</table>

### 2.12.11 Datumbox

The Datumbox Machine Learning Framework is an open-source framework written in Java which allows the rapid development Machine Learning and Statistical applications. The main focus of the framework is to include a large number of machine learning algorithms & statistical methods and to be able to handle large sized datasets.

Datumbox comes with a large number of pre-trained models which allow you to perform Sentiment Analysis (Document & Twitter), Subjectivity Analysis, Topic Classification, Spam Detection, Adult Content Detection, Language Detection, Commercial Detection, Educational Detection and Gender Detection.

Datumbox is not supported by a large team of commercial developers or large group of FOSS developers. Basically one developer maintains it as a side project. So review this FOSS project before you make large investments building applications on top of it.

<table>
<thead>
<tr>
<th>SBB License</th>
<th>Apache License 2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Java</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="http://www.datumbox.com/">http://www.datumbox.com/</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/datumbox/datumbox-framework">https://github.com/datumbox/datumbox-framework</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>ML, ML Framework</td>
</tr>
</tbody>
</table>

21 [http://convnetjs.com](http://convnetjs.com)
2.12.12 DeepDetect

DeepDetect implements support for supervised and unsupervised deep learning of images, text and other data, with focus on simplicity and ease of use, test and connection into existing applications. It supports classification, object detection, segmentation, regression, autoencoders and more.

It has Python and other client libraries.

Deep Detect has also a REST API for Deep Learning with:

- JSON communication format
- Pre-trained models
- Neural architecture templates
- Python, Java, C# clients
- Output templating

<table>
<thead>
<tr>
<th>SBB License</th>
<th>MIT License</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>C++</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="https://deepdetect.com">https://deepdetect.com</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/beniz/deepdetect">https://github.com/beniz/deepdetect</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>ML, ML Framework</td>
</tr>
</tbody>
</table>

2.12.13 Deeplearning4j

Deep Learning for Java, Scala & Clojure on Hadoop & Spark With GPUs.

Eclipse Deeplearning4J is an distributed neural net library written in Java and Scala.

Eclipse Deeplearning4j a commercial-grade, open-source, distributed deep-learning library written for Java and Scala. DL4J is designed to be used in business environments on distributed GPUs and CPUs.

Deeplearning4J integrates with Hadoop and Spark and runs on several backends that enable use of CPUs and GPUs. The aim of this project is to create a plug-and-play solution that is more convention than configuration, and which allows for fast prototyping. This project is created by Skymind who delivers support and offers also the option for machine learning models to be hosted with Skymind’s model server on a cloud environment

<table>
<thead>
<tr>
<th>SBB License</th>
<th>Apache License 2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Java</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="https://deeplearning4j.org">https://deeplearning4j.org</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/deeplearning4j/deeplearning4j">https://github.com/deeplearning4j/deeplearning4j</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>ML, ML Framework</td>
</tr>
</tbody>
</table>
2.12.14 Detectron2

Detectron is Facebook AI Research’s software system that implements state-of-the-art object detection algorithms, including Mask R-CNN\(^{22}\). Detectron2 is a ground-up rewrite of Detectron that started with maskrcnn-benchmark. The platform is now implemented in PyTorch\(^{23}\). With a new, more modular design, Detectron2 is flexible and extensible, and able to provide fast training on single or multiple GPU servers. Detectron2 includes high-quality implementations of state-of-the-art object detection algorithms.

New in Detectron 2:

- It is powered by the PyTorch\(^{24}\) deep learning framework.
- Includes more features such as panoptic segmentation, densepose, Cascade R-CNN, rotated bounding boxes, etc.
- Can be used as a library to support different projects\(^{25}\) on top of it. We’ll open source more research projects in this way.
- It trains much faster\(^{26}\).

The goal of Detectron is to provide a high-quality, high-performance codebase for object detection research. It is designed to be flexible in order to support rapid implementation and evaluation of novel research.

A number of Facebook teams use this platform to train custom models for a variety of applications including augmented reality and community integrity. Once trained, these models can be deployed in the cloud and on mobile devices, powered by the highly efficient Caffe2 runtime.

Documentation on: https://detectron2.readthedocs.io/index.html

<table>
<thead>
<tr>
<th>SBB License</th>
<th>Apache License 2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Python</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="https://github.com/facebookresearch/Detectron2">https://github.com/facebookresearch/Detectron2</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/facebookresearch/detectron2">https://github.com/facebookresearch/detectron2</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>ML, ML Framework, Python</td>
</tr>
</tbody>
</table>

2.12.15 Dopamine

Dopamine is a research framework for fast prototyping of reinforcement learning algorithms. It aims to fill the need for a small, easily grokked codebase in which users can freely experiment with wild ideas (speculative research).

Our design principles are:

- **Easy experimentation:** Make it easy for new users to run benchmark experiments.
- **Flexible development:** Make it easy for new users to try out research ideas.
- **Compact and reliable:** Provide implementations for a few, battle-tested algorithms.
- **Reproducible:** Facilitate reproducibility in results.

\(^{22}\) https://arxiv.org/abs/1703.06870  
\(^{23}\) https://pytorch.org/  
\(^{24}\) https://pytorch.org  
\(^{25}\) https://github.com/facebookresearch/detectron2/blob/master/projects  
\(^{26}\) https://detectron2.readthedocs.io/notes/benchmarks.html
2.12.16 FASTAI

The fastai library simplifies training fast and accurate neural nets using modern best practices. Fast.ai’s mission is to make the power of state-of-the-art deep learning available to anyone. fastai sits on top of PyTorch\(^{27}\), which provides the foundation.

fastai is a deep learning library which provides high-level components that can quickly and easily provide state-of-the-art results in standard deep learning domains, and provides researchers with low-level components that can be mixed and matched to build new approaches. It aims to do both things without substantial compromises in ease of use, flexibility, or performance.

Docs can be found on: [http://docs.fast.ai/](http://docs.fast.ai/)

---

2.12.17 FEATURETOOLS

*One of the holy grails of machine learning is to automate more and more of the feature engineering process.* — Pedro

Featuretools\(^{28}\) is a python library for automated feature engineering. Featuretools automatically creates features from temporal and relational datasets. Featuretools works alongside tools you already use to build machine learning pipelines. You can load in pandas dataframes and automatically create meaningful features in a fraction of the time it would take to do manually.

Featuretools is a python library for automated feature engineering. Featuretools can automatically create a single table of features for any “target entity”.

Featuretools is a framework to perform automated feature engineering. It excels at transforming transactional and relational datasets into feature matrices for machine learning.

\(^{27}\) [https://pytorch.org/](https://pytorch.org/)
\(^{28}\) [https://www.featuretools.com](https://www.featuretools.com)
### 2.12.18 FlyingSquid

FlyingSquid is a ML framework for automatically building models from multiple noisy label sources. Users write functions that generate noisy labels for data, and FlyingSquid uses the agreements and disagreements between them to learn a *label model* of how accurate the *labeling functions* are. The label model can be used directly for downstream applications, or it can be used to train a powerful end model.

<table>
<thead>
<tr>
<th>SBB License</th>
<th>BSD License 2.0 (3-clause, New or Revised) License</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Python</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="https://www.featuretools.com/">https://www.featuretools.com/</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/Featuretools/featuretools">https://github.com/Featuretools/featuretools</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>ML, ML Framework, Python</td>
</tr>
</tbody>
</table>

### 2.12.19 Karate Club

Karate Club is an unsupervised machine learning extension library for NetworkX\(^29\).

**Karate Club** consists of state-of-the-art methods to do unsupervised learning on graph structured data. To put it simply it is a Swiss Army knife for small-scale graph mining research. First, it provides network embedding techniques at the node and graph level. Second, it includes a variety of overlapping and non-overlapping community detection methods. Implemented methods cover a wide range of network science (NetSci, Complenet), data mining (ICDM\(^30\), CIKM\(^31\), KDD\(^32\)), artificial intelligence (AAAI\(^33\), IJCAI\(^34\)) and machine learning (NeurIPS\(^35\), ICML\(^36\), ICLR\(^37\)) conferences, workshops, and pieces from prominent journals.

The documentation can be found at: https://karateclub.readthedocs.io/en/latest/

The Karate Club API draws heavily from the ideas of scikit-learn and the output generated is suitable as input for scikit-learn’s machine learning procedures.

The paper can be found at: https://arxiv.org/pdf/2003.04819.pdf

---

\(^29\) https://networkx.github.io/
\(^30\) http://icdm2019.bigke.org/
\(^31\) http://www.cikm2019.net/
\(^32\) https://www.kdd.org/kdd2020/
\(^33\) http://www.aaai.org/Conferences/conferences.php
\(^34\) https://www.ijcai.org/
\(^35\) https://nips.cc/
\(^36\) https://icml.cc/
\(^37\) https://iclr.cc/
2.12.20 Keras

Keras is a high-level neural networks API, written in Python and capable of running on top of TensorFlow, CNTK, or Theano. It was developed with a focus on enabling fast experimentation. Being able to go from idea to result with the least possible delay is key to doing good research.

Use Keras if you need a deep learning library that:

- Allows for easy and fast prototyping (through user friendliness, modularity, and extensibility).
- Supports both convolutional networks and recurrent networks, as well as combinations of the two.
- Runs seamlessly on CPU and GPU.

2.12.21 learn2learn

learn2learn is a PyTorch library for meta-learning implementations.

The goal of meta-learning is to enable agents to learn how to learn. That is, we would like our agents to become better learners as they solve more and more tasks.

Features:

learn2learn provides high- and low-level utilities for meta-learning. The high-level utilities allow arbitrary users to take advantage of existing meta-learning algorithms. The low-level utilities enable researchers to develop new and better meta-learning algorithms.

Some features of learn2learn include:

- Modular API: implement your own training loops with our low-level utilities.
- Provides various meta-learning algorithms (e.g. MAML, FOMAML, MetaSGD, ProtoNets, DiCE)
- Task generator with unified API, compatible with torchvision, torchtext, torchaudio, and cherry.
- Provides standardized meta-learning tasks for vision (Omniglot, mini-ImageNet), reinforcement learning (Particles, Mujoco), and even text (news classification).
- 100% compatible with PyTorch — use your own modules, datasets, or libraries!
### 2.12.22 Lore

Lore is a python framework to make machine learning approachable for Engineers and maintainable for Data Scientists.

**Features**

- Models support hyper parameter search over estimators with a data pipeline. They will efficiently utilize multiple GPUs (if available) with a couple different strategies, and can be saved and distributed for horizontal scalability.

- Estimators from multiple packages are supported: Keras\(^{38}\) (TensorFlow/Theano/CNTK), XGBoost\(^{39}\) and SciKit Learn\(^{40}\). They can all be subclassed with build, fit or predict overridden to completely customize your algorithm and architecture, while still benefiting from everything else.

- Pipelines avoid information leaks between train and test sets, and one pipeline allows experimentation with many different estimators. A disk based pipeline is available if you exceed your machines available RAM.

- Transformers standardize advanced feature engineering. For example, convert an American first name to its statistical age or gender using US Census data. Extract the geographic area code from a free form phone number string. Common date, time and string operations are supported efficiently through pandas.

- Encoders offer robust input to your estimators, and avoid common problems with missing and long tail values. They are well tested to save you from garbage in/garbage out.

- IO connections are configured and pooled in a standard way across the app for popular (no)sql databases, with transaction management and read write optimizations for bulk data, rather than typical ORM single row operations. Connections share a configurable query cache, in addition to encrypted S3 buckets for distributing models and datasets.

- Dependency Management for each individual app in development, that can be 100% replicated to production. No manual activation, or magic env vars, or hidden files that break python for everything else. No knowledge required of venv, pyenv, pyvenv, virtualenv, virtualenvwrapper, pipenv, conda. Ain’t nobody got time for that.

- Tests for your models can be run in your Continuous Integration environment, allowing Continuous Deployment for code and training updates, without increased work for your infrastructure team.

- Workflow Support whether you prefer the command line, a python console, jupyter notebook, or IDE. Every environment gets readable logging and timing statements configured for both production and development.

---

\(^{38}\) [https://keras.io/](https://keras.io/)

\(^{39}\) [https://xgboost.readthedocs.io/](https://xgboost.readthedocs.io/)

2.12.23 Microsoft Cognitive Toolkit (CNTK)

The Microsoft Cognitive Toolkit (https://cntk.ai) is a unified deep learning toolkit that describes neural networks as a series of computational steps via a directed graph. In this directed graph, leaf nodes represent input values or network parameters, while other nodes represent matrix operations upon their inputs. CNTK allows users to easily realize and combine popular model types such as feed-forward DNNs, convolutional nets (CNNs), and recurrent networks (RNNs/LSTMs). It implements stochastic gradient descent (SGD, error backpropagation) learning with automatic differentiation and parallelization across multiple GPUs and servers. CNTK has been available under an open-source license since April 2015.

Docs on: https://docs.microsoft.com/en-us/cognitive-toolkit/

2.12.24 ml5.js

ml5.js aims to make machine learning approachable for a broad audience of artists, creative coders, and students. The library provides access to machine learning algorithms and models in the browser, building on top of TensorFlow.js with no other external dependencies.

The library is supported by code examples, tutorials, and sample data sets with an emphasis on ethical computing. Bias in data, stereotypical harms, and responsible crowdsourcing are part of the documentation around data collection and usage.

ml5.js is heavily inspired by Processing and p5.js.

---

41 https://js.tensorflow.org/
42 https://processing.org/
43 https://p5js.org/
2.12.25 MLjar

MLjar is a platform for rapid prototyping, developing and deploying machine learning models. MLjar makes algorithm search and tuning painless. It checks many different algorithms for you. For each algorithm hyper-parameters are separately tuned. All computations run in parallel in MLjar cloud, so you get your results very quickly. At the end the ensemble of models is created, so your predictive model will be super accurate.

There are two types of interface available in MLjar:

- You can run Machine Learning models in your browser, you don’t need to code anything. Just upload dataset, click which attributes to use, which algorithms to use and go! This makes Machine Learning super easy for everyone and make it possible to get really useful models.
- There is a python wrapper over MLjar API so you don’t need to open any browser or click on any button, just write fancy python code! We like it and hope you will like it too! To start using MLjar python package please go to our github.

2.12.26 MLsquare

[ML]² – ML Square is a python library that utilises deep learning techniques to:

- Enable interoperability between existing standard machine learning frameworks.
- Provide explainability as a first-class function.
- Make ML self learnable.

The following are the design goals:

- Bring Your Own Spec First.
- Bring Your Own Experience First.
- Consistent.
- Compositional.
- Modular.

44 https://github.com/mljar/mljar-api-python

2.12. ML Frameworks
2.12.27 NeuralStructuredLearning

Neural Structured Learning (NSL) is a new learning paradigm to train neural networks by leveraging structured signals in addition to feature inputs. Structure can be explicit as represented by a graph or implicit as induced by adversarial perturbation.

Structured signals are commonly used to represent relations or similarity among samples that may be labeled or unlabeled. Leveraging these signals during neural network training harnesses both labeled and unlabeled data, which can improve model accuracy, particularly when the amount of labeled data is relatively small. Additionally, models trained with samples that are generated by adversarial perturbation have been shown to be robust against malicious attacks, which are designed to mislead a model’s prediction or classification.

NSL generalizes to Neural Graph Learning as well as to Adversarial Learning. The NSL framework in TensorFlow provides the following easy-to-use APIs and tools for developers to train models with structured signals:

- **Keras APIs** to enable training with graphs (explicit structure) and adversarial pertubations (implicit structure).
- **TF ops and functions** to enable training with structure when using lower-level TensorFlow APIs
- **Tools** to build graphs and construct graph inputs for training

NSL is part of the TensorFlow framework. More info on: [https://www.tensorflow.org/neural_structured_learning/](https://www.tensorflow.org/neural_structured_learning/)

---

### 2.12.28 NNI (Neural Network Intelligence)

NNI (Neural Network Intelligence) is a toolkit to help users run automated machine learning (AutoML) experiments. The tool dispatches and runs trial jobs generated by tuning algorithms to search the best neural architecture and/or hyper-parameters in different environments like local machine, remote servers and cloud. (Microsoft ML project)

Who should consider using NNI:
- Those who want to try different AutoML algorithms in their training code (model) at their local machine.
- Those who want to run AutoML trial jobs in different environments to speed up search (e.g. remote servers and cloud).
- Researchers and data scientists who want to implement their own AutoML algorithms and compare it with other algorithms.
- ML Platform owners who want to support AutoML in their platform.

<table>
<thead>
<tr>
<th>SBB License</th>
<th>MIT License</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Python</td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/Microsoft/nni">https://github.com/Microsoft/nni</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>ML, ML Framework</td>
</tr>
</tbody>
</table>

### 2.12.29 NuPIC

The Numenta Platform for Intelligent Computing (NuPIC) is a machine intelligence platform that implements the HTM learning algorithms\(^{45}\). HTM is a detailed computational theory of the neocortex. At the core of HTM are time-based continuous learning algorithms that store and recall spatial and temporal patterns. NuPIC is suited to a variety of problems, particularly anomaly detection and prediction of streaming data sources.

Note: This project is in Maintenance Mode.

<table>
<thead>
<tr>
<th>SBB License</th>
<th>GNU Affero General Public License Version 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Python</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="https://numenta.org/">https://numenta.org/</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/numenta/nupic">https://github.com/numenta/nupic</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>ML Framework, Python</td>
</tr>
</tbody>
</table>

### 2.12.30 Plato

The Plato Research Dialogue System is a flexible framework that can be used to create, train, and evaluate conversational AI agents in various environments. It supports interactions through speech, text, or dialogue acts and each conversational agent can interact with data, human users, or other conversational agents (in a multi-agent setting). Every component of every agent can be trained independently online or offline and Plato provides an easy way of wrapping around virtually any existing model, as long as Plato’s interface is adhered to.

OSS by Uber.

\(^{45}\) https://numenta.com/resources/papers-videos-and-more/
2.12.31 Polyaxon

A platform for reproducible and scalable machine learning and deep learning on kubernetes

Polyaxon is a platform for building, training, and monitoring large scale deep learning applications.

Polyaxon deploys into any data center, cloud provider, or can be hosted and managed by Polyaxon, and it supports all the major deep learning frameworks such as Tensorflow, MXNet, Caffe, Torch, etc.

Polyaxon makes it faster, easier, and more efficient to develop deep learning applications by managing workloads with smart container and node management. And it turns GPU servers into shared, self-service resources for your team or organization.

2.12.32 PyCaret

PyCaret is an open source low-code machine learning library in Python that aims to reduce the hypothesis to insights cycle time in a ML experiment. It enables data scientists to perform end-to-end experiments quickly and efficiently. In comparison with the other open source machine learning libraries, PyCaret is an alternate low-code library that can be used to perform complex machine learning tasks with only few lines of code. PyCaret is essentially a Python wrapper around several machine learning libraries and frameworks such as scikit-learn, XGBoost, Microsoft LightGBM, spaCy and many more.

The design and simplicity of PyCaret is inspired by the emerging role of citizen data scientists, a term first used by Gartner. Citizen Data Scientists are power users who can perform both simple and moderately sophisticated analytical tasks that would previously have required more expertise. Seasoned data scientists are often difficult to find and expensive to hire but citizen data scientists can be an effective way to mitigate this gap and address data related challenges in business setting.

PyCaret claims to be simple, easy to use and deployment ready. All the steps performed in a ML experiment can be reproduced using a pipeline that is automatically developed and orchestrated in PyCaret as you progress through the experiment. A pipeline can be saved in a binary file format that is transferable across environments.
### 2.12.33 Pylearn2

Pylearn2 is a library designed to make machine learning research easy.

This project does not have any current developer.

### 2.12.34 Pyro

Deep universal probabilistic programming with Python and PyTorch. Pyro is in an alpha release. It is developed and used by Uber AI Labs\(^\text{46}\).

Pyro is a universal probabilistic programming language (PPL) written in Python and supported by PyTorch\(^\text{47}\) on the backend. Pyro enables flexible and expressive deep probabilistic modeling, unifying the best of modern deep learning and Bayesian modeling. It was designed with these key principles:

- Universal: Pyro can represent any computable probability distribution.
- Scalable: Pyro scales to large data sets with little overhead.
- Minimal: Pyro is implemented with a small core of powerful, composable abstractions.
- Flexible: Pyro aims for automation when you want it, control when you need it.

Documentation on: [http://docs.pyro.ai/](http://docs.pyro.ai/)

### License and Technology

<table>
<thead>
<tr>
<th>SBB License</th>
<th>MIT License</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Python</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="https://www.pycaret.org">https://www.pycaret.org</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/pycaret/pycaret">https://github.com/pycaret/pycaret</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>ML Framework</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SBB License</th>
<th>BSD License 2.0 (3-clause, New or Revised) License</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Python</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="http://deeplearning.net/software/pylearn2/">http://deeplearning.net/software/pylearn2/</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/lisa-lab/pylearn2">https://github.com/lisa-lab/pylearn2</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>ML, ML Framework</td>
</tr>
</tbody>
</table>

\(^{46}\) [http://uber.ai](http://uber.ai)  
\(^{47}\) [http://pytorch.org](http://pytorch.org)
2.12.35 Pythia

Pythia is a modular framework for supercharging vision and language research built on top of PyTorch created by Facebook.

You can use Pythia to bootstrap for your next vision and language multimodal research project. Pythia can also act as starter codebase for challenges around vision and language datasets (TextVQA challenge, VQA challenge).

It features:

- **Model Zoo**: Reference implementations for state-of-the-art vision and language model including LoRRA\(^{48}\) (SoTA on VQA and TextVQA), Pythia\(^{49}\) model (VQA 2018 challenge winner) and BAN\(^{50}\).
- **Multi-Tasking**: Support for multi-tasking which allows training on multiple dataset together.
- **Datasets**: Includes support for various datasets built-in including VQA, VizWiz, TextVQA and VisualDialog.
- **Modules**: Provides implementations for many commonly used layers in vision and language domain
- **Distributed**: Support for distributed training based on DataParallel as well as DistributedDataParallel.
- **Unopinionated**: Unopinionated about the dataset and model implementations built on top of it.
- **Customization**: Custom losses, metrics, scheduling, optimizers, tensorboard; suits all your custom needs.

<table>
<thead>
<tr>
<th>SBB License</th>
<th>BSD License 2.0 (3-clause, New or Revised) License</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Python</td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/facebookresearch/pythia">https://github.com/facebookresearch/pythia</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>ML, ML Framework, Python</td>
</tr>
</tbody>
</table>

2.12.36 PyTorch

PyTorch is a Python-first machine learning framework that is utilized heavily towards deep learning. It supports CUDA technology (From NVIDIA) to fully use the the power of the dedicated GPUs in training, analyzing and validating neural networks models.

Deep learning frameworks have often focused on either usability or speed, but not both. PyTorch is a machine learning library that shows that these two goals are in fact compatible: it provides an imperative and Pythonic programming style that supports code as a model, makes debugging easy and is consistent with other popular scientific computing libraries, while remaining efficient and supporting hardware accelerators such as GPUs.

PyTorch is very widely used, and is under active development and support. PyTorch is:

- a deep learning framework that puts Python first.
- a research-focused framework.
- Python package that provides two high-level features:

\(^{48}\) https://arxiv.org/abs/1904.08920
\(^{49}\) https://arxiv.org/abs/1807.09956
\(^{50}\) https://github.com/facebookresearch/pythia/blob/master
PyTorch uses tensor computation (like NumPy) with strong GPU acceleration. It can use deep neural networks built on a tape-based autograd system.

PyTorch is a Python package that provides two high-level features:

- Tensor computation (like NumPy) with strong GPU acceleration
- Deep neural networks built on a tape-based autograd system

You can reuse your favorite Python packages such as NumPy, SciPy and Cython to extend PyTorch when needed. PyTorch has become a popular tool in the deep learning research community by combining a focus on usability with careful performance considerations.

A very good overview of the design principles and architecture of PyTorch can be found in this paper https://arxiv.org/pdf/1912.01703.pdf.

### 2.12.37 ReAgent

ReAgent is an open source end-to-end platform for applied reinforcement learning (RL) developed and used at Facebook. ReAgent is built in Python and uses PyTorch for modeling and training and TorchScript for model serving. The platform contains workflows to train popular deep RL algorithms and includes data preprocessing, feature transformation, distributed training, counterfactual policy evaluation, and optimized serving. For more detailed information about ReAgent see the white paper here.

The platform was once named “Horizon” but we have adopted the name “ReAgent” recently to emphasize its broader scope in decision making and reasoning.

### 2.12.38 RLCard

RLCard is a toolkit for Reinforcement Learning (RL) in card games. It supports multiple card environments with easy-to-use interfaces. The goal of RLCard is to bridge reinforcement learning and imperfect information games, and

---

51 [https://research.fb.com/publications/horizon-facebooks-open-source-applied-reinforcement-learning-platform/](https://research.fb.com/publications/horizon-facebooks-open-source-applied-reinforcement-learning-platform/)
push forward the research of reinforcement learning in domains with multiple agents, large state and action space, and sparse reward. RLCard is developed by DATA Lab\textsuperscript{52} at Texas A&M University.


<table>
<thead>
<tr>
<th>SBB License</th>
<th>MIT License</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Python</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="http://rlcard.org/">http://rlcard.org/</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/datamlab/rlcard">https://github.com/datamlab/rlcard</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>ML Framework, Python</td>
</tr>
</tbody>
</table>

### 2.12.39 Scikit-learn

scikit-learn is a Python module for machine learning. It is a Python module for machine learning built on top of SciPy and is distributed under the 3-Clause BSD license.

Key features:
- Simple and efficient tools for predictive data analysis
- Accessible to everybody, and reusable in various contexts
- Built on NumPy, SciPy, and matplotlib
- Open source, commercially usable – BSD license

<table>
<thead>
<tr>
<th>SBB License</th>
<th>BSD License 2.0 (3-clause, New or Revised) License</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Python</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="http://scikit-learn.org">http://scikit-learn.org</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/scikit-learn/scikit-learn">https://github.com/scikit-learn/scikit-learn</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>ML, ML Framework</td>
</tr>
</tbody>
</table>

### 2.12.40 SINGA

Distributed deep learning system.

SINGA was initiated by the DB System Group at National University of Singapore in 2014, in collaboration with the database group of Zhejiang University.

SINGA’s software stack includes three major components, namely, core, IO and model:

1. The core component provides memory management and tensor operations.
2. IO has classes for reading (and writing) data from (to) disk and network.

\textsuperscript{52} http://faculty.cs.tamu.edu/xiahu/
3. The model component provides data structures and algorithms for machine learning models, e.g., layers for neural network models, optimizers/initializer/metric/loss for general machine learning models.

<table>
<thead>
<tr>
<th>SBB License</th>
<th>Apache License 2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Java</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="http://singa.apache.org/">http://singa.apache.org/</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/apache/singa">https://github.com/apache/singa</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>ML Framework</td>
</tr>
</tbody>
</table>

### 2.12.41 Streamlit

The fastest way to build custom ML tools. Streamlit lets you create apps for your machine learning projects with deceptively simple Python scripts. It supports hot-reloading, so your app updates live as you edit and save your file. No need to mess with HTTP requests, HTML, JavaScript, etc. All you need is your favorite editor and a browser.

Documentation on: [https://streamlit.io/docs/](https://streamlit.io/docs/)

<table>
<thead>
<tr>
<th>SBB License</th>
<th>Apache License 2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Javascript, Python</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="https://streamlit.io/">https://streamlit.io/</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/streamlit/streamlit">https://github.com/streamlit/streamlit</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>ML, ML Framework, ML Hosting, ML Tool, Python</td>
</tr>
</tbody>
</table>

### 2.12.42 Tensorflow

TensorFlow is an Open Source Software Library for Machine Intelligence. TensorFlow is by far the most used and popular ML open source project. And since the first initial release was only just in November 2015 it is expected that the impact of this OSS package will expand even more.

TensorFlow™ is an open source software library for numerical computation using data flow graphs. Nodes in the graph represent mathematical operations, while the graph edges represent the multidimensional data arrays (tensors) communicated between them. The flexible architecture allows you to deploy computation to one or more CPUs or GPUs in a desktop, server, or mobile device with a single API. TensorFlow was originally developed by researchers and engineers working on the Google Brain Team within Google’s Machine Intelligence research organization for the purposes of conducting machine learning and deep neural networks research, but the system is general enough to be applicable in a wide variety of other domains as well.

TensorFlow comes with a tool called TensorBoard which you can use to get some insight into what is happening. TensorBoard is a suite of web applications for inspecting and understanding your TensorFlow runs and graphs.

There is also a version of TensorFlow that runs in a browser. This is TensorFlow.js ([https://js.tensorflow.org/](https://js.tensorflow.org/)). TensorFlow.js is a WebGL accelerated, browser based JavaScript library for training and deploying ML models.

53 [https://www.tensorflow.org/versions/r0.11/how_tos/graph_viz/index.html](https://www.tensorflow.org/versions/r0.11/how_tos/graph_viz/index.html)
Since privacy is a contentious fight TensorFlow has now (2020) also a library called ‘TensorFlow Privacy’. This is a python library that includes implementations of TensorFlow optimizers for training machine learning models with differential privacy. The library comes with tutorials and analysis tools for computing the privacy guarantees provided. See: https://github.com/tensorflow/privacy

<table>
<thead>
<tr>
<th>SBB License</th>
<th>Apache License 2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>C</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="https://www.tensorflow.org/">https://www.tensorflow.org/</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/tensorflow/tensorflow">https://github.com/tensorflow/tensorflow</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>ML, ML Framework</td>
</tr>
</tbody>
</table>

### 2.12.43 TF Encrypted

TF Encrypted is a framework for encrypted machine learning in TensorFlow. It looks and feels like TensorFlow, taking advantage of the ease-of-use of the Keras API while enabling training and prediction over encrypted data via secure multi-party computation and homomorphic encryption. TF Encrypted aims to make privacy-preserving machine learning readily available, without requiring expertise in cryptography, distributed systems, or high performance computing.

<table>
<thead>
<tr>
<th>SBB License</th>
<th>Apache License 2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Python</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="https://tf-encrypted.io/">https://tf-encrypted.io/</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/tf-encrypted/tf-encrypted">https://github.com/tf-encrypted/tf-encrypted</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>ML, ML Framework, Privacy</td>
</tr>
</tbody>
</table>

### 2.12.44 Theano

Theano is a Python library that allows you to define, optimize, and evaluate mathematical expressions involving multi-dimensional arrays efficiently. It can use GPUs and perform efficient symbolic differentiation.

Note: After almost ten years of development the company behind Theano has stopped development and support (Q4-2017). But this library has been an innovation driver for many other OSS ML packages!

Since a lot of ML libraries and packages use Theano you should check (as always) the health of your ML stack.
2.12.45 Thinc

Thinc is the machine learning library powering spaCy. It features a battle-tested linear model designed for large sparse learning problems, and a flexible neural network model under development for spaCy v2.0.

Thinc is a lightweight deep learning library that offers an elegant, type-checked, functional-programming API for composing models, with support for layers defined in other frameworks such as PyTorch, TensorFlow and MXNet. You can use Thinc as an interface layer, a standalone toolkit or a flexible way to develop new models.

Thinc is a practical toolkit for implementing models that follow the “Embed, encode, attend, predict” architecture. It’s designed to be easy to install, efficient for CPU usage and optimised for NLP and deep learning with text – in particular, hierarchically structured input and variable-length sequences.

2.12.46 Turi

Turi Create simplifies the development of custom machine learning models. Turi is OSS machine learning from Apple. Turi Create simplifies the development of custom machine learning models. You don’t have to be a machine learning expert to add recommendations, object detection, image classification, image similarity or activity classification to your app.
2.12.47 TuriCreate

This SBB is from Apple. Apple, is with Siri already for a long time active in machine learning. But even Apple is releasing building blocks under OSS licenses now.

Turi Create simplifies the development of custom machine learning models. You don’t have to be a machine learning expert to add recommendations, object detection, image classification, image similarity or activity classification to your app.

- **Easy-to-use:** Focus on tasks instead of algorithms
- **Visual:** Built-in, streaming visualizations to explore your data
- **Flexible:** Supports text, images, audio, video and sensor data
- **Fast and Scalable:** Work with large datasets on a single machine
- **Ready To Deploy:** Export models to Core ML for use in iOS, macOS, watchOS, and tvOS apps

<table>
<thead>
<tr>
<th>SBB License</th>
<th>BSD License 2.0 (3-clause, New or Revised) License</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Python</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="https://turi.com/index.html">https://turi.com/index.html</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/apple/turicreate">https://github.com/apple/turicreate</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>ML, ML Framework, Python</td>
</tr>
</tbody>
</table>

2.12.48 Vowpal Wabbit

Vowpal Wabbit is a machine learning system which pushes the frontier of machine learning with techniques such as online, hashing, allreduce, reductions, learning2search, active, and interactive learning. There is a specific focus on reinforcement learning with several contextual bandit algorithms implemented and the online nature lending to the problem well. Vowpal Wabbit is a destination for implementing and maturing state of the art algorithms with performance in mind.

- **Input Format.** The input format for the learning algorithm is substantially more flexible than might be expected. Examples can have features consisting of free form text, which is interpreted in a bag-of-words way. There can even be multiple sets of free form text in different namespaces.

- **Speed.** The learning algorithm is fast — similar to the few other online algorithm implementations out there. There are several optimization algorithms available with the baseline being sparse gradient descent (GD) on a loss function.

- **Scalability.** This is not the same as fast. Instead, the important characteristic here is that the memory footprint of the program is bounded independent of data. This means the training set is not loaded into main memory before learning starts. In addition, the size of the set of features is bounded independent of the amount of training data using the hashing trick.

- **Feature Interaction.** Subsets of features can be internally paired so that the algorithm is linear in the cross-product of the subsets. This is useful for ranking problems. The alternative of explicitly expanding the features before feeding them into the learning algorithm can be both computation and space intensive, depending on how it’s handled.

Microsoft Research is a major contributor to Vowpal Wabbit.
2.12.49 XAI

XAI is a Machine Learning library that is designed with AI explainability in its core. XAI contains various tools that enable for analysis and evaluation of data and models. The XAI library is maintained by The Institute for Ethical AI & ML\(^54\), and it was developed based on the 8 principles for Responsible Machine Learning\(^55\). You can find the documentation at [https://ethicalml.github.io/xai/index.html](https://ethicalml.github.io/xai/index.html).

<table>
<thead>
<tr>
<th>SBB License</th>
<th>MIT License</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>CPP</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="https://vowpalwabbit.org/">https://vowpalwabbit.org/</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/VowpalWabbit/vowpal_wabbit">https://github.com/VowpalWabbit/vowpal_wabbit</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>ML, ML Framework</td>
</tr>
</tbody>
</table>

2.13 Computer vision

Computer vision is a field that deals with how computers can be made to gain high-level understanding of digital images and videos. Machine learning is a good match for image classification.

2.13.1 libfacedetection

This is an open source library for CNN-based face detection in images. The CNN model has been converted to static variables in C source files. The source code does not depend on any other libraries. What you need is just a C++ compiler. You can compile the source code under Windows, Linux, ARM and any platform with a C++ compiler. SIMD instructions are used to speed up the detection. You can enable AVX2 if you use Intel CPU or NEON for ARM.

\(^{54}\) [http://ethical.institute/](http://ethical.institute/)
\(^{55}\) [http://ethical.institute/principles.html](http://ethical.institute/principles.html)
2.13.2 YOLOv3

A minimal PyTorch implementation of YOLOv3, with support for training, inference and evaluation.

You only look once (YOLO) is a state-of-the-art, real-time object detection system. In depth paper on YOLOv3 is on: https://pjreddie.com/media/files/papers/YOLOv3.pdf

2.13.3 Raster Vision

Raster Vision is an open source Python framework for building computer vision models on satellite, aerial, and other large imagery sets (including oblique drone imagery).

It allows users (who don’t need to be experts in deep learning!) to quickly and repeatably configure experiments that execute a machine learning workflow including: analyzing training data, creating training chips, training models, creating predictions, evaluating models, and bundling the model files and configuration for easy deployment.

Some features:

- There is built-in support for chip classification, object detection, and semantic segmentation with backends using PyTorch and Tensorflow.
- Experiments can be executed on CPUs and GPUs with built-in support for running in the cloud using AWS Batch. The framework is extensible to new data sources, tasks (eg. object detection), backends (eg. TF Object Detection API), and cloud providers.

Documentation on: https://docs.rastervision.io/
2.13.4 DeOldify

A Deep Learning based project for colorizing and restoring old images (and video!)

<table>
<thead>
<tr>
<th>SBB License</th>
<th>MIT License</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Python</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="https://github.com/jantic/DeOldify">https://github.com/jantic/DeOldify</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/jantic/DeOldify">https://github.com/jantic/DeOldify</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>Computer vision, ML</td>
</tr>
</tbody>
</table>

2.13.5 SOD

SOD is an embedded, modern cross-platform computer vision and machine learning software library that expose a set of APIs for deep-learning, advanced media analysis & processing including real-time, multi-class object detection and model training on embedded systems with limited computational resource and IoT devices.

SOD was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in open source as well commercial products.

Designed for computational efficiency and with a strong focus on real-time applications. SOD includes a comprehensive set of both classic and state-of-the-art deep-neural networks with their pre-trained models.

<table>
<thead>
<tr>
<th>SBB License</th>
<th>GNU General Public License (GPL) 3.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>C</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="https://sod.pixlab.io/">https://sod.pixlab.io/</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/symisc/sod">https://github.com/symisc/sod</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>Computer vision, ML</td>
</tr>
</tbody>
</table>

2.13.6 makesense.ai

makesense.ai is a free to use online tool for labelling photos. Thanks to the use of a browser it does not require any complicated installation – just visit the website and you are ready to go. It also doesn’t matter which operating system you’re running on – we do our best to be truly cross-platform. It is perfect for small computer vision deeplearning projects, making the process of preparing a dataset much easier and faster.

56 https://pixlab.io/downloads

2.13. Computer vision
2.13.7 DeepPrivacy

DeepPrivacy is a fully automatic anonymization technique for images.

The DeepPrivacy GAN never sees any privacy sensitive information, ensuring a fully anonymized image. It utilizes bounding box annotation to identify the privacy-sensitive area, and sparse pose information to guide the network in difficult scenarios.

DeepPrivacy detects faces with state-of-the-art detection methods. Mask R-CNN\(^\text{57}\) is used to generate a sparse pose information of the face, and DSFD\(^\text{58}\) is used to detect faces in the image.


2.13.8 Face_recognition

The world’s simplest facial recognition api for Python and the command line.

Recognize and manipulate faces from Python or from the command line with the world’s simplest face recognition library.

Built using dlib\(^\text{60}\)’s state-of-the-art face recognition built with deep learning. The model has an accuracy of 99.38% on the Labeled Faces in the Wild\(^\text{61}\) benchmark.

This also provides a simple face_recognition command line tool that lets you do face recognition on a folder of images from the command line!

Full API documentation can be found here: https://face-recognition.readthedocs.io/en/latest/

Git quick-scan report:

- Date of git statistics quick-scan report: 2019/12/19
- Number of files in the git repository: 96

---

57 https://arxiv.org/abs/1703.06870
58 https://arxiv.org/abs/1810.10220
60 http://dlib.net/
61 http://vis-www.cs.umass.edu/lfw/
• Total Lines of Code (of all files): 70415 total
• Most recent commit in this repository: Tue Dec 3 16:53:45 2019 +0530
• Number of authors: 33

First commit info:
• Author: Adam Geitgey
• Date: Fri Mar 3 16:29:23 2017 -0800

<table>
<thead>
<tr>
<th>SBB License</th>
<th>MIT License</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Python</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="https://github.com/ageitgey/face_recognition">https://github.com/ageitgey/face_recognition</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/ageitgey/face_recognition">https://github.com/ageitgey/face_recognition</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>Computer vision, face detection, ML, ML Tool, Python</td>
</tr>
</tbody>
</table>

### 2.13.9 DeepFaceLab

DeepFaceLab is a tool that utilizes machine learning to replace faces in videos. More than 95% of deepfake videos are created with DeepFaceLab.

<table>
<thead>
<tr>
<th>SBB License</th>
<th>GNU General Public License (GPL) 3.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Python</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="https://github.com/iperov/DeepFaceLab">https://github.com/iperov/DeepFaceLab</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/iperov/DeepFaceLab">https://github.com/iperov/DeepFaceLab</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>Computer vision, Deepfakes, ML, Python</td>
</tr>
</tbody>
</table>

### 2.13.10 FaceSwap

FaceSwap is a tool that utilizes deep learning to recognize and swap faces in pictures and videos.

When faceswapping was first developed and published, the technology was groundbreaking, it was a huge step in AI development. It was also completely ignored outside of academia because the code was confusing and fragmentary. It required a thorough understanding of complicated AI techniques and took a lot of effort to figure it out. Until one individual brought it together into a single, cohesive collection. Before “deepfakes” these techniques were like black magic, only practiced by those who could understand all of the inner workings as described in esoteric and endlessly complicated books and papers.

Powered by Tensorflow, Keras and Python; Faceswap will run on Windows, macOS and Linux. And GPL licensed!
2.13.11 JeelizFaceFilter

Javascript/WebGL lightweight face tracking library designed for augmented reality webcam filters. Features: multiple faces detection, rotation, mouth opening. Various integration examples are provided (Three.js, Babylon.js, FaceSwap, Canvas2D, CSS3D...).

Enables developers to solve computer-vision problems directly from the browser.

Features:
- face detection,
- face tracking,
- face rotation detection,
- mouth opening detection,
- multiple faces detection and tracking,
- very robust for all lighting conditions,
- video acquisition with HD video ability,
- interfaced with 3D engines like THREE.JS, BABYLON.JS, A-FRAME,
- interfaced with more accessible APIs like CANVAS, CSS3D.

2.13.12 OpenCV: Open Source Computer Vision Library

OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. Being a BSD-licensed product, OpenCV makes it easy for businesses to utilize and modify the code.

The library has more than 2500 optimized algorithms, which includes a comprehensive set of both classic and state-of-the-art computer vision and machine learning algorithms. These algorithms can be used to detect and recognize faces, identify objects, classify human actions in videos, track camera movements, track moving objects, extract 3D models of objects, produce 3D point clouds from stereo cameras, stitch images together to produce a high resolution...
image of an entire scene, find similar images from an image database, remove red eyes from images taken using flash, follow eye movements, recognize scenery and establish markers to overlay it with augmented reality, etc.

<table>
<thead>
<tr>
<th>SBB License</th>
<th>BSD License 2.0 (3-clause, New or Revised) License</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>C</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="https://opencv.org/">https://opencv.org/</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/opencv/opencv">https://github.com/opencv/opencv</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>Computer vision, ML</td>
</tr>
</tbody>
</table>

2.13.13 Luminoth

Luminoth is an open source toolkit for computer vision. Currently, we support object detection and image classification, but we are aiming for much more. It is built in Python, using TensorFlow and Sonnet.

Note: No longer maintained.

<table>
<thead>
<tr>
<th>SBB License</th>
<th>BSD License 2.0 (3-clause, New or Revised) License</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Python</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="https://luminoth.ai">https://luminoth.ai</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/tryolabs/luminoth">https://github.com/tryolabs/luminoth</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>Computer vision, ML</td>
</tr>
</tbody>
</table>

2.14 ML Tools

Besides FOSS machine learning frameworks there are special tools that save you time when creating ML applications. This section is a opinionated collection of FOSS ML tools that can make creating applications easier.

2.14.1 AI Explainability 360

The AI Explainability 360 toolkit is an open-source library that supports interpretability and explainability of datasets and machine learning models. The AI Explainability 360 Python package includes a comprehensive set of algorithms that cover different dimensions of explanations along with proxy explainability metrics.

It is OSS from IBM (so apache2.0) so mind the history of openness IBM has regarding OSS product development. The documentation can be found here: https://aix360.readthedocs.io/en/latest/
2.14.2 Apollo

Apollo is a high performance, flexible architecture which accelerates the development, testing, and deployment of Autonomous Vehicles.

Apollo 2.0 supports vehicles autonomously driving on simple urban roads. Vehicles are able to cruise on roads safely, avoid collisions with obstacles, stop at traffic lights, and change lanes if needed to reach their destination.

Apollo 5.5 enhances the complex urban road autonomous driving capabilities of previous Apollo releases, by introducing curb-to-curb driving support. With this new addition, Apollo is now a leap closer to fully autonomous urban road driving. The car has complete 360-degree visibility, along with upgraded perception deep learning model and a brand new prediction model to handle the changing conditions of complex road and junction scenarios, making the car more secure and aware.

2.14.3 Data Science Version Control (DVC)

Data Science Version Control or DVC is an open-source tool for data science and machine learning projects. With a simple and flexible Git-like architecture and interface it helps data scientists:

1. manage machine learning models – versioning, including data sets and transformations (scripts) that were used to generate models;
2. make projects reproducible;
3. make projects shareable;
4. manage experiments with branching and metrics tracking;

It aims to replace tools like Excel and Docs that are being commonly used as a knowledge repo and a ledger for the team, ad-hoc scripts to track and move deploy different model versions, ad-hoc data file suffixes and prefixes.
2.14.4 Espresso

Espresso is an open-source, modular, extensible end-to-end neural automatic speech recognition (ASR) toolkit based on the deep learning library PyTorch\(^{62}\) and the popular neural machine translation toolkit `fairseq`\(^{<https://github.com/pytorch/fairseq>}`. Espresso supports distributed training across GPUs and computing nodes, and features various decoding approaches commonly employed in ASR, including look-ahead word-based language model fusion, for which a fast, parallelized decoder is implemented.


2.14.5 EuclidesDB

EuclidesDB is a multi-model machine learning feature database that is tight coupled with PyTorch and provides a backend for including and querying data on the model feature space. Some features of EuclidesDB are listed below:

- Written in C++ for performance;
- Uses protobuf for data serialization;
- Uses gRPC for communication;
- LevelDB integration for database serialization;
- Many indexing methods implemented (Annoy\(^{63}\), Faiss\(^{64}\), etc);
- Tight PyTorch integration through libtorch;
- Easy integration for new custom fine-tuned models;
- Easy client language binding generation;
- Free and open-source with permissive license;

---

\(^{62}\) https://github.com/pytorch/pytorch

\(^{63}\) https://github.com/spotify/annoy

\(^{64}\) https://github.com/facebookresearch/faiss
2.14.6 Fabrik

Fabrik is an online collaborative platform to build, visualize and train deep learning models via a simple drag-and-drop interface. It allows researchers to collaboratively develop and debug models using a web GUI that supports importing, editing and exporting networks written in widely popular frameworks like Caffe, Keras, and TensorFlow.

2.14.7 Face_recognition

The world’s simplest facial recognition api for Python and the command line.

Recognize and manipulate faces from Python or from the command line with the world’s simplest face recognition library.

Built using dlib\(^{65}\)’s state-of-the-art face recognition built with deep learning. The model has an accuracy of 99.38% on the Labeled Faces in the Wild\(^{66}\) benchmark.

This also provides a simple face_recognition command line tool that lets you do face recognition on a folder of images from the command line!

Full API documentation can be found here: https://face-recognition.readthedocs.io/en/latest/

Git quick-scan report:

- Date of git statics quick-scan report: 2019/12/19
- Number of files in the git repository: 96
- Total Lines of Code (of all files): 70415 total
- Most recent commit in this repository: Tue Dec 3 16:53:45 2019 +0530
- Number of authors: 33

First commit info:

- Author: Adam Geitgey
- Date: Fri Mar 3 16:29:23 2017 -0800

\(^{65}\) http://dlib.net/
\(^{66}\) http://vis-www.cs.umass.edu/lfw/
### 2.14.8 Kedro

Kedro is a workflow development tool that helps you build data pipelines that are robust, scalable, deployable, reproducible and versioned. We provide a standard approach so that you can:

- spend more time building your data pipeline,
- worry less about how to write production-ready code,
- standardise the way that your team collaborates across your project,
- work more efficiently.

**Features:**

- A standard and easy-to-use project template, allowing your collaborators to spend less time understanding how you’ve set up your analytics project
- Data abstraction, managing how you load and save data so that you don’t have to worry about the reproducibility of your code in different environments
- Configuration management, helping you keep credentials out of your code base
- Pipeline visualisation with Kedro-Viz: [Link](https://github.com/quantumblacklabs/kedro-viz) making it easy to see how your data pipeline is constructed
- Seamless packaging, allowing you to ship your projects to production, e.g. using Docker [Link](https://github.com/quantumblacklabs/kedro-docker) or Kedro-Airflow [Link](https://github.com/quantumblacklabs/kedro-airflow)
- Versioning for your datasets and machine learning models whenever your pipeline runs

**Documentation on:** https://kedro.readthedocs.io/

**The REACT visualization for Kedro is on:** [Link](https://github.com/quantumblacklabs/kedro-viz)

---

67 [Link](https://github.com/quantumblacklabs/kedro-viz)
2.14.9 Ludwig

Ludwig is a toolbox built on top of TensorFlow that allows to train and test deep learning models without the need to write code. Ludwig provides two main functionalities: training models and using them to predict. It is based on datatype abstraction, so that the same data preprocessing and postprocessing will be performed on different datasets that share data types and the same encoding and decoding models developed for one task can be reused for different tasks.

All you need to provide is a CSV file containing your data, a list of columns to use as inputs, and a list of columns to use as outputs, Ludwig will do the rest. Simple commands can be used to train models both locally and in a distributed way, and to use them to predict on new data.

A programmatic API is also available in order to use Ludwig from your python code. A suite of visualization tools allows you to analyze models’ training and test performance and to compare them.

Ludwig is built with extensibility principles in mind and is based on data type abstractions, making it easy to add support for new data types as well as new model architectures.

It can be used by practitioners to quickly train and test deep learning models as well as by researchers to obtain strong baselines to compare against and have an experimentation setting that ensures comparability by performing standard data preprocessing and visualization.

2.14.10 makesense.ai

makesense.ai is a free to use online tool for labelling photos. Thanks to the use of a browser it does not require any complicated installation – just visit the website and you are ready to go. It also doesn’t matter which operating system you’re running on – we do our best to be truly cross-platform. It is perfect for small computer vision deeplearning projects, making the process of preparing a dataset much easier and faster.
2.14.11 MLflow

MLflow offers a way to simplify ML development by making it easy to track, reproduce, manage, and deploy models. MLflow (currently in alpha) is an open source platform designed to manage the entire machine learning lifecycle and work with any machine learning library. It offers:

- Record and query experiments: code, data, config, results
- Packaging format for reproducible runs on any platform
- General format for sending models to diverse deploy tools

2.14.12 MLPerf

A broad ML benchmark suite for measuring performance of ML software frameworks, ML hardware accelerators, and ML cloud platforms.

The MLPerf effort aims to build a common set of benchmarks that enables the machine learning (ML) field to measure system performance for both training and inference from mobile devices to cloud services. We believe that a widely accepted benchmark suite will benefit the entire community, including researchers, developers, builders of machine learning frameworks, cloud service providers, hardware manufacturers, application providers, and end users.
2.14.13 ModelDB

A system to manage machine learning models.

ModelDB is an end-to-end system to manage machine learning models. It ingests models and associated metadata as models are being trained, stores model data in a structured format, and surfaces it through a web-fronted for rich querying. ModelDB can be used with any ML environment via the ModelDB Light API. ModelDB native clients can be used for advanced support in spark.ml and scikit-learn.

The ModelDB frontend provides rich summaries and graphs showing model data. The frontend provides functionality to slice and dice this data along various attributes (e.g. operations like filter by hyperparameter, group by datasets) and to build custom charts showing model performance.

<table>
<thead>
<tr>
<th>SBB License</th>
<th>MIT License</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Python, Javascript</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="https://mitdbg.github.io/modeldb/">https://mitdbg.github.io/modeldb/</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/mitdbg/modeldb">https://github.com/mitdbg/modeldb</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>Administration, ML, ML Tool</td>
</tr>
</tbody>
</table>

2.14.14 Netron

Netron is a viewer for neural network, deep learning and machine learning models.

Netron supports ONNX\(^{68}\) (.onnx, .pb), Keras (.h5, .keras), CoreML (.mlmodel) and TensorFlow Lite (.tflite). Netron has experimental support for Caffe (.caffemodel), Caffe2 (predict_net.pb), MXNet (-symbol.json), TensorFlow.js (model.json, .pb) and TensorFlow (.pb, .meta).

<table>
<thead>
<tr>
<th>SBB License</th>
<th>GNU General Public License (GPL) 2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Python, Javascript</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="https://www.lutzroeder.com/ai/">https://www.lutzroeder.com/ai/</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/lutzroeder/Netron">https://github.com/lutzroeder/Netron</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>Data viewer, ML, ML Tool</td>
</tr>
</tbody>
</table>

2.14.15 NLP Architect

NLP Architect is an open-source Python library for exploring the state-of-the-art deep learning topologies and techniques for natural language processing and natural language understanding. It is intended to be a platform for future research and collaboration.

Features:

- Core NLP models used in many NLP tasks and useful in many NLP applications

\(^{68}\) http://onnx.ai
• Novel NLU models showcasing novel topologies and techniques
• Optimized NLP/NLU models showcasing different optimization algorithms on neural NLP/NLU models
• Model-oriented design:
  – Train and run models from command-line.
  – API for using models for inference in python.
  – Procedures to define custom processes for training, inference or anything related to processing.
  – CLI sub-system for running procedures
• Based on optimized Deep Learning frameworks:
  – TensorFlow\textsuperscript{69}
  – PyTorch\textsuperscript{70}
  – Dynet\textsuperscript{71}
• Essential utilities for working with NLP models – Text/String pre-processing, IO, data-manipulation, metrics, embeddings.

<table>
<thead>
<tr>
<th>SBB License</th>
<th>Apache License 2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Python</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="http://nlp_architect.nervanasys.com/">http://nlp_architect.nervanasys.com/</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/NervanaSystems/nlp-architect">https://github.com/NervanaSystems/nlp-architect</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>ML, ML Tool, NLP, Python</td>
</tr>
</tbody>
</table>

2.14.16 ONNX

ONNX provides an open source format for AI models. It defines an extensible computation graph model, as well as definitions of built-in operators and standard data types. Initially we focus on the capabilities needed for inferencing (evaluation).

Open Neural Network Exchange (ONNX) is an open standard format for representing machine learning models. ONNX is supported by a community of partners who have implemented it in many frameworks and tools.

Caffe2, PyTorch, Microsoft Cognitive Toolkit, Apache MXNet and other tools are developing ONNX support. Enabling interoperability between different frameworks and streamlining the path from research to production will increase the speed of innovation in the AI community. We are an early stage and we invite the community to submit feedback and help us further evolve ONNX.

Companies behind ONNX are AWS, Facebook and Microsoft Corporation and more.

\textsuperscript{69} https://www.tensorflow.org/
\textsuperscript{70} https://pytorch.org/
\textsuperscript{71} https://dynet.readthedocs.io/en/latest/
2.14.17 OpenML

OpenML is an on-line machine learning platform for sharing and organizing data, machine learning algorithms and experiments. It claims to be designed to create a frictionless, networked ecosystem, so that you can readily integrate into your existing processes/code/environments. It also allows people from all over the world to collaborate and build directly on each other’s latest ideas, data and results, irrespective of the tools and infrastructure they happen to use. So nice ideas to build an open science movement. The people behind OpenML are mostly (data)scientist. So using this product for real world business use cases will take some extra effort.

Although OpenML is exposed as an foundation based on openness, a quick inspection learned that the OpenML platform is not as open as you want. Also the OSS software is not created to be run on premise. So be aware when doing large (time) investments into this OpenML platform.

2.14.18 Orange

Orange is a comprehensive, component-based software suite for machine learning and data mining, developed at Bioinformatics Laboratory.

Orange is available by default on Anaconda Navigator dashboard. Orange\(^2\) is a component-based data mining software. It includes a range of data visualization, exploration, preprocessing and modeling techniques. It can be used through a nice and intuitive user interface or, for more advanced users, as a module for the Python programming language.

One of the nice features is the option for visual programming. Can you do visual interactive data exploration for rapid qualitative analysis with clean visualizations. The graphic user interface allows you to focus on exploratory data analysis instead of coding, while clever defaults make fast prototyping of a data analysis workflow extremely easy.

\(^2\) http://orange.biolab.si/
2.14.19 PySyft

A library for encrypted, privacy preserving deep learning. PySyft is a Python library for secure, private Deep Learning. PySyft decouples private data from model training, using Multi-Party Computation (MPC)\(^73\) within PyTorch. View the paper on Arxiv\(^74\).

2.14.20 RAPIDS

The RAPIDS suite of software libraries gives you the freedom to execute end-to-end data science and analytics pipelines entirely on GPUs. It relies on NVIDIA® CUDA®\(^75\) primitives for low-level compute optimization, but exposes that GPU parallelism and high-bandwidth memory speed through user-friendly Python interfaces.

RAPIDS also focuses on common data preparation tasks for analytics and data science. This includes a familiar DataFrame API that integrates with a variety of machine learning algorithms for end-to-end pipeline accelerations without paying typical serialization costs--. RAPIDS also includes support for multi-node, multi-GPU deployments, enabling vastly accelerated processing and training on much larger dataset sizes.

---

\(^{73}\) https://en.wikipedia.org/wiki/Secure_multi-party_computation

\(^{74}\) https://arxiv.org/abs/1811.04017

\(^{75}\) https://developer.nvidia.com/cuda-toolkit
2.14.21 SHAP

SHAP (SHapley Additive exPlanations) is a unified approach to explain the output of any machine learning model. SHAP connects game theory with local explanations, uniting several previous methods [1-7] and representing the only possible consistent and locally accurate additive feature attribution method based on expectations (see our papers\textsuperscript{76} for details and citations).

There are also sample notebooks that demonstrate different use cases for SHAP in the github repro.

<table>
<thead>
<tr>
<th>SBB License</th>
<th>MIT License</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Python</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="https://github.com/slundberg/shap">https://github.com/slundberg/shap</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/slundberg/shap">https://github.com/slundberg/shap</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>ML, ML Tool</td>
</tr>
</tbody>
</table>

2.14.22 Skater

Skater is a python package for model agnostic interpretation of predictive models. With Skater, you can unpack the internal mechanics of arbitrary models; as long as you can obtain inputs, and use a function to obtain outputs, you can use Skater to learn about the models internal decision policies.

The project was started as a research idea to find ways to enable better interpretability(preferably human interpretability) to predictive “black boxes” both for researchers and practioners.

Documentation at: https://datascienceinc.github.io/Skater/overview.html

<table>
<thead>
<tr>
<th>SBB License</th>
<th>MIT License</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Python</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="https://www.datascience.com/resources/tools/skater">https://www.datascience.com/resources/tools/skater</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/datascienceinc/Skater">https://github.com/datascienceinc/Skater</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>ML, ML Tool</td>
</tr>
</tbody>
</table>

2.14.23 Snorkel

Snorkel is a system for rapidly creating, modeling, and managing training data, currently focused on accelerating the development of structured or “dark” data extraction applications for domains in which large labeled training sets are not available or easy to obtain.

\textsuperscript{76} https://github.com/slundberg/shap#citations
2.14.24 Streamlit

The fastest way to build custom ML tools. Streamlit lets you create apps for your machine learning projects with deceptively simple Python scripts. It supports hot-reloading, so your app updates live as you edit and save your file. No need to mess with HTTP requests, HTML, JavaScript, etc. All you need is your favorite editor and a browser.

Documentation on: https://streamlit.io/docs/

2.14.25 TensorWatch

TensorWatch is a debugging and visualization tool designed for data science, deep learning and reinforcement learning from Microsoft Research. It works in Jupyter Notebook to show real-time visualizations of your machine learning training and perform several other key analysis tasks for your models and data.

TensorWatch is designed to be flexible and extensible so you can also build your own custom visualizations, UIs, and dashboards. Besides traditional “what-you-see-is-what-you-log” approach, it also has a unique capability to execute arbitrary queries against your live ML training process, return a stream as a result of the query and view this stream using your choice of a visualizer (we call this Lazy Logging Mode).

TensorWatch is under heavy development with a goal of providing a platform for debugging machine learning in one easy to use, extensible, and hackable package.
2.14.26 VisualDL

VisualDL is an open-source cross-framework web dashboard that richly visualizes the performance and data flowing through your neural network training. VisualDL is a deep learning visualization tool that can help design deep learning jobs. It includes features such as scalar, parameter distribution, model structure and image visualization.

<table>
<thead>
<tr>
<th>SBB License</th>
<th>Apache License 2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>C++</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="http://visualdl.paddlepaddle.org/">http://visualdl.paddlepaddle.org/</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/PaddlePaddle/VisualDL">https://github.com/PaddlePaddle/VisualDL</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>ML, ML Tool</td>
</tr>
</tbody>
</table>

2.14.27 What-If Tool

The What-If Tool\(^7\) (WIT) provides an easy-to-use interface for expanding understanding of a black-box ML model. With the plugin, you can perform inference on a large set of examples and immediately visualize the results in a variety of ways. Additionally, examples can be edited manually or programatically and re-run through the model in order to see the results of the changes. It contains tooling for investigating model performance and fairness over subsets of a dataset.

The purpose of the tool is that give people a simple, intuitive, and powerful way to play with a trained ML model on a set of data through a visual interface with absolutely no code required.

<table>
<thead>
<tr>
<th>SBB License</th>
<th>Apache License 2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Python</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="https://pair-code.github.io/what-if-tool/">https://pair-code.github.io/what-if-tool/</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/tensorflow/tensorboard/tree/master/tensorboard/plugins/interactive_inference">https://github.com/tensorflow/tensorboard/tree/master/tensorboard/plugins/interactive_inference</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>ML, ML Tool</td>
</tr>
</tbody>
</table>

2.15 ML hosting

Machine learning needs an infrastructure stack and various components to run. For training and for production. Some open frameworks makes creating ML solutions easier and faster.

2.15.1 BentoML

BentoML makes it easy to serve and deploy machine learning models in the cloud.

\(^7\) https://pair-code.github.io/what-if-tool

---

114 Chapter 2. Table of Contents
It is an open source framework for machine learning teams to build cloud-native prediction API services that are ready for production. BentoML supports most popular ML training frameworks and common deployment platforms including major cloud providers and docker/kubernetes.


<table>
<thead>
<tr>
<th>SBB License</th>
<th>Apache License 2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Python</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="http://BentoML.ai">http://BentoML.ai</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/bentoml/BentoML">https://github.com/bentoml/BentoML</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>ML, ML Hosting, Python</td>
</tr>
</tbody>
</table>

### 2.15.2 Streamlit

The fastest way to build custom ML tools. Streamlit lets you create apps for your machine learning projects with deceptively simple Python scripts. It supports hot-reloading, so your app updates live as you edit and save your file. No need to mess with HTTP requests, HTML, JavaScript, etc. All you need is your favorite editor and a browser.

Documentation on: https://streamlit.io/docs/

<table>
<thead>
<tr>
<th>SBB License</th>
<th>Apache License 2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Javascript, Python</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="https://streamlit.io/">https://streamlit.io/</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/streamlit/streamlit">https://github.com/streamlit/streamlit</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>ML, ML Framework, ML Hosting, ML Tool, Python</td>
</tr>
</tbody>
</table>

### 2.15.3 RAPIDS

The RAPIDS suite of software libraries gives you the freedom to execute end-to-end data science and analytics pipelines entirely on GPUs. It relies on NVIDIA® CUDA® primitives for low-level compute optimization, but exposes that GPU parallelism and high-bandwidth memory speed through user-friendly Python interfaces.

RAPIDS also focuses on common data preparation tasks for analytics and data science. This includes a familiar DataFrame API that integrates with a variety of machine learning algorithms for end-to-end pipeline accelerations without paying typical serialization costs—. RAPIDS also includes support for multi-node, multi-GPU deployments, enabling vastly accelerated processing and training on much larger dataset sizes.

2.15.4 Acumos AI

Acumos AI is a platform and open source framework that makes it easy to build, share, and deploy AI apps. Acumos standardizes the infrastructure stack and components required to run an out-of-the-box general AI environment.

Acumos is a platform which enhances the development, training and deployment of AI models. Its purpose is to scale up the introduction of AI-based software across a wide range of industrial and commercial problems in order to reach a critical mass of applications. In this way, Acumos will drive toward a data-centric process for producing software based upon machine learning as the central paradigm. The platform seeks to empower data scientists to publish more adaptive AI models and shield them from the task of custom development of fully integrated solutions. Ideally, software developers will use Acumos to change the process of software development from a code-writing and editing exercise into a classroom-like code training process in which models will be trained and graded on their ability to successfully analyze datasets that they are fed. Then, the best model can be selected for the job and integrated into a complete application.

Acumos is part of the LF Deep Learning Foundation, an umbrella organization within The Linux Foundation that supports and sustains open source innovation in artificial intelligence, machine learning, and deep learning while striving to make these critical new technologies available to developers and data scientists everywhere.

2.15.5 Ray

Ray is a flexible, high-performance distributed execution framework for AI applications. Ray is currently under heavy development. But Ray has already a good start, with good documentation (http://ray.readthedocs.io/en/latest/index.html) and a tutorial. Also Ray is backed by scientific researchers and published papers.

Ray comes with libraries that accelerate deep learning and reinforcement learning development:

- Ray Tune\(^80\): Hyperparameter Optimization Framework
- Ray RLlib\(^81\): A Scalable Reinforcement Learning Library

---

\(^80\) http://ray.readthedocs.io/en/latest/tune.html
\(^81\) http://ray.readthedocs.io/en/latest/rllib.html
2.15.6 Turi

Turi Create simplifies the development of custom machine learning models. Turi is OSS machine learning from Apple. Turi Create simplifies the development of custom machine learning models. You don’t have to be a machine learning expert to add recommendations, object detection, image classification, image similarity or activity classification to your app.

<table>
<thead>
<tr>
<th>SBB License</th>
<th>Apache License 2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Python</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="https://ray-project.github.io/">https://ray-project.github.io/</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/ray-project/ray">https://github.com/ray-project/ray</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>ML, ML Hosting</td>
</tr>
</tbody>
</table>

2.16 NLP Frameworks

Natural language processing (NLP) is a field located at the intersection of data science and machine learning (ML). It is focussed on teaching machines how to understand human languages and extract meaning from text. Using good open FOSS NLP software saves you time and has major benefits above using closed solutions.

NLP tools make it simple to handle NLP-related tasks such as document classification, topic modeling, part-of-speech (POS) tagging, word vectors, and sentiment analysis.

2.16.1 AllenNLP

An open-source NLP research library, built on PyTorch. AllenNLP is a NLP research library, built on PyTorch, for developing state-of-the-art deep learning models on a wide variety of linguistic tasks. AllenNLP makes it easy to design and evaluate new deep learning models for nearly any NLP problem, along with the infrastructure to easily run them in the cloud or on your laptop.

AllenNLP was designed with the following principles:

- **Hyper-modular and lightweight.** Use the parts which you like seamlessly with PyTorch.
- **Extensively tested and easy to extend.** Test coverage is above 90% and the example models provide a template for contributions.
- **Take padding and masking seriously,** making it easy to implement correct models without the pain.
- **Experiment friendly.** Run reproducible experiments from a json specification with comprehensive logging.
## 2.16.2 Apache OpenNLP

The Apache OpenNLP library is a machine learning based toolkit for the processing of natural language text. It supports the most common NLP tasks, such as tokenization, sentence segmentation, part-of-speech tagging, named entity extraction, chunking, parsing, and coreference resolution. These tasks are usually required to build more advanced text processing services. OpenNLP also included maximum entropy and perceptron based machine learning.

The goal of the OpenNLP project will be to create a mature toolkit for the abovementioned tasks. An additional goal is to provide a large number of pre-built models for a variety of languages, as well as the annotated text resources that those models are derived from.

<table>
<thead>
<tr>
<th>SBB License</th>
<th>Apache License 2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Python</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="http://opennlp.apache.org/">http://opennlp.apache.org/</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="http://opennlp.apache.org/source-code.html">http://opennlp.apache.org/source-code.html</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>NLP</td>
</tr>
</tbody>
</table>

## 2.16.3 Apache Tika

The Apache Tika™ toolkit detects and extracts metadata and text from over a thousand different file types (such as PPT, XLS, and PDF). All of these file types can be parsed through a single interface, making Tika useful for search engine indexing, content analysis, translation, and much more.

Several wrappers are available to use Tika in another programming language, such as Julia\(^{82}\) or Python\(^{83}\).

---

\(^{82}\) https://github.com/aviks/Taro.jl
\(^{83}\) https://github.com/chrismattmann/tika-python
2.16.4 BERT

BERT, or Bidirectional Encoder Representations from Transformers, is a new method of pre-training language representations which obtains state-of-the-art results on a wide array of Natural Language Processing (NLP) tasks.

Our academic paper which describes BERT in detail and provides full results on a number of tasks can be found here: https://arxiv.org/abs/1810.04805.

OSS NLP training models from Google Research.

2.16.5 Bling Fire

A lightning fast Finite State machine and REgular expression manipulation library. Bling Fire Tokenizer is a tokenizer designed for fast-speed and quality tokenization of Natural Language text. It mostly follows the tokenization logic of NLTK, except hyphenated words are split and a few errors are fixed.

2.16.6 ERNIE

An Implementation of ERNIE For Language Understanding (including Pre-training models and Fine-tuning tools)

ERNIE 2.0\textsuperscript{84} is a continual pre-training framework for language understanding in which pre-training tasks can be incrementally built and learned through multi-task learning. In this framework, different customized tasks can be

\textsuperscript{84} https://arxiv.org/abs/1907.12412v1

2.16. NLP Frameworks

---

<table>
<thead>
<tr>
<th>SBB License</th>
<th>Apache License 2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Java</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="https://tika.apache.org/">https://tika.apache.org/</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://tika.apache.org/">https://tika.apache.org/</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>NLP</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SBB License</th>
<th>MIT License</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>CPP</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="https://github.com/Microsoft/BlingFire">https://github.com/Microsoft/BlingFire</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/Microsoft/BlingFire">https://github.com/Microsoft/BlingFire</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>NLP</td>
</tr>
</tbody>
</table>
incrementally introduced at any time. For example, the tasks including named entity prediction, discourse relation recognition, sentence order prediction are leveraged in order to enable the models to learn language representations.

<table>
<thead>
<tr>
<th>SBB License</th>
<th>Apache License 2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Python</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="https://github.com/PaddlePaddle/ERNIE">https://github.com/PaddlePaddle/ERNIE</a>:</td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/PaddlePaddle/ERNIE">https://github.com/PaddlePaddle/ERNIE</a>:</td>
</tr>
<tr>
<td>Tag(s)</td>
<td>NLP, Python</td>
</tr>
</tbody>
</table>

### 2.16.7 fastText

fastText\(^5^\) is a library for efficient learning of word representations and sentence classification. Models can later be reduced in size to even fit on mobile devices.

Created by Facebook OpenSource, now available for us all. Also used for the new search on StackOverflow, see [https://stackoverflow.blog/2019/08/14/crokage-a-new-way-to-search-stack-overflow/](https://stackoverflow.blog/2019/08/14/crokage-a-new-way-to-search-stack-overflow/)

<table>
<thead>
<tr>
<th>SBB License</th>
<th>MIT License</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>CPP, Python</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="https://fasttext.cc/">https://fasttext.cc/</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/facebookresearch/fastText">https://github.com/facebookresearch/fastText</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>NLP</td>
</tr>
</tbody>
</table>

### 2.16.8 Flair

A very simple framework for **state-of-the-art NLP**. Developed by Zalando Research\(^6^\).

Flair is:

- **A powerful NLP library.** Flair allows you to apply our state-of-the-art natural language processing (NLP) models to your text, such as named entity recognition (NER), part-of-speech tagging (PoS), sense disambiguation and classification.

- **Multilingual.** Thanks to the Flair community, we support a rapidly growing number of languages. We also now include ‘one model, many languages’ taggers, i.e. single models that predict PoS or NER tags for input text in various languages.

- **A text embedding library.** Flair has simple interfaces that allow you to use and combine different word and document embeddings, including our proposed Flair embeddings\(^7^\), BERT embeddings and ELMo embeddings.

\(^5^\) [https://fasttext.cc/](https://fasttext.cc/)

\(^6^\) [https://research.zalando.com/](https://research.zalando.com/)

• **A Pytorch NLP framework.** Our framework builds directly on Pytorch\(^8\), making it easy to train your own models and experiment with new approaches using Flair embeddings and classes.

<table>
<thead>
<tr>
<th>SBB License</th>
<th>MIT License</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Python</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="https://github.com/zalandoresearch/flair">https://github.com/zalandoresearch/flair</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/zalandoresearch/flair">https://github.com/zalandoresearch/flair</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>ML, NLP, Python</td>
</tr>
</tbody>
</table>

2.16.9 Gensim

Gensim is a Python library for *topic modelling*, *document indexing* and *similarity retrieval* with large corpora. Target audience is the *natural language processing* (NLP) and *information retrieval* (IR) community.

<table>
<thead>
<tr>
<th>SBB License</th>
<th>MIT License</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Python</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="https://github.com/RaRe-Technologies/gensim">https://github.com/RaRe-Technologies/gensim</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/RaRe-Technologies/gensim">https://github.com/RaRe-Technologies/gensim</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>ML, NLP, Python</td>
</tr>
</tbody>
</table>

2.16.10 Icecaps

Microsoft Icecaps is an open-source toolkit for building neural conversational systems. Icecaps provides an array of tools from recent conversation modeling and general NLP literature within a flexible paradigm that enables complex multi-task learning setups.

Background information can be found here [https://www.aclweb.org/anthology/P19-3021](https://www.aclweb.org/anthology/P19-3021)

<table>
<thead>
<tr>
<th>SBB License</th>
<th>MIT License</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Python</td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/microsoft/icecaps">https://github.com/microsoft/icecaps</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>NLP, Python</td>
</tr>
</tbody>
</table>

\(^8\) [https://pytorch.org/](https://pytorch.org/)
2.16.11 jiant

jiant is a software toolkit for natural language processing research, designed to facilitate work on multitask learning and transfer learning for sentence understanding tasks.

New software for the The General Language Understanding Evaluation (GLUE) benchmark. This software can be used for evaluating, and analyzing natural language understanding systems.

See also: https://super.gluebenchmark.com/

<table>
<thead>
<tr>
<th>SBB License</th>
<th>MIT License</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Python</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="https://jiant.info/">https://jiant.info/</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/nyu-mll/jiant">https://github.com/nyu-mll/jiant</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>NLP, Python, Research</td>
</tr>
</tbody>
</table>

2.16.12 Klassify

Redis based text classification service with real-time web interface.

What is Text Classification: Text classification, document classification or document categorization is a problem in library science, information science and computer science. The task is to assign a document to one or more classes or categories.

<table>
<thead>
<tr>
<th>SBB License</th>
<th>MIT License</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Python</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="https://github.com/fatiherikli/klassify">https://github.com/fatiherikli/klassify</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/fatiherikli/klassify">https://github.com/fatiherikli/klassify</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>ML, NLP, Text classification</td>
</tr>
</tbody>
</table>

2.16.13 Neuralcoref

State-of-the-art coreference resolution based on neural nets and spaCy.

NeuralCoref is a pipeline extension for spaCy 2.0 that annotates and resolves coreference clusters using a neural network. NeuralCoref is production-ready, integrated in spaCy’s NLP pipeline and easily extensible to new training datasets.
2.16.14 NLP Architect

NLP Architect is an open-source Python library for exploring the state-of-the-art deep learning topologies and techniques for natural language processing and natural language understanding. It is intended to be a platform for future research and collaboration.

Features:

- Core NLP models used in many NLP tasks and useful in many NLP applications
- Novel NLU models showcasing novel topologies and techniques
- Optimized NLP/NLU models showcasing different optimization algorithms on neural NLP/NLU models
- Model-oriented design:
  - Train and run models from command-line.
  - API for using models for inference in python.
  - Procedures to define custom processes for training, inference or anything related to processing.
  - CLI sub-system for running procedures
- Based on optimized Deep Learning frameworks:
  - TensorFlow[^89]
  - PyTorch[^90]
  - Dynet[^91]
- Essential utilities for working with NLP models – Text/String pre-processing, IO, data-manipulation, metrics, embeddings.

[^89]: https://www.tensorflow.org/
[^90]: https://pytorch.org/
[^91]: https://dynet.readthedocs.io/en/latest/

<table>
<thead>
<tr>
<th>SBB License</th>
<th>MIT License</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Python</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="https://huggingface.co/coref/">https://huggingface.co/coref/</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/huggingface/neuralcoref">https://github.com/huggingface/neuralcoref</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>ML, NLP, Python</td>
</tr>
</tbody>
</table>

[^89]: https://www.tensorflow.org/
[^90]: https://pytorch.org/
[^91]: https://dynet.readthedocs.io/en/latest/
### 2.16.15 NLTK (Natural Language Toolkit)

NLTK is a leading platform for building Python programs to work with human language data. It provides easy-to-use interfaces to over 50 corpora and lexical resources\(^2\) such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, wrappers for industrial-strength NLP libraries.

Check also the (free) online Book (OReily published)

<table>
<thead>
<tr>
<th>SBB License</th>
<th>Apache License 2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Python</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="http://www.nltk.org">http://www.nltk.org</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/nltk/nltk">https://github.com/nltk/nltk</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>NLP</td>
</tr>
</tbody>
</table>

### 2.16.16 Pattern

Pattern is a web mining module for Python. It has tools for:

- Data Mining: web services (Google, Twitter, Wikipedia), web crawler, HTML DOM parser
- Natural Language Processing: part-of-speech taggers, n-gram search, sentiment analysis, WordNet
- Machine Learning: vector space model, clustering, classification (KNN, SVM, Perceptron)
- Network Analysis: graph centrality and visualization.

<table>
<thead>
<tr>
<th>SBB License</th>
<th>BSD License 2.0 (3-clause, New or Revised) License</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Python</td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/clips/pattern">https://github.com/clips/pattern</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>ML, NLP, Web scraping</td>
</tr>
</tbody>
</table>

### 2.16.17 Rant

Rant is an all-purpose procedural text engine that is most simply described as the opposite of Regex. It has been refined to include a dizzying array of features for handling everything from the most basic of string generation tasks to advanced dialogue generation, code templating, automatic formatting, and more.

The goal of the project is to enable developers of all kinds to automate repetitive writing tasks with a high degree of creative freedom.

Features:

\(^2\) http://nltk.org/nltk_data/
• Recursive, weighted branching with several selection modes
• Queryable dictionaries
• Automatic capitalization, rhyming, English indefinite articles, and multi-lingual number verbalization
• Print to multiple separate outputs
• Probability modifiers for pattern elements
• Loops, conditional statements, and subroutines
• Fully-functional object model
• Import/Export resources easily with the .rantpkg format
• Compatible with Unity 2017

<table>
<thead>
<tr>
<th>SBB License</th>
<th>MIT License</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>.NET</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="https://berkin.me/rant/">https://berkin.me/rant/</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/TheBerkin/rant">https://github.com/TheBerkin/rant</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>.NET, ML, NLP, text generation</td>
</tr>
</tbody>
</table>

2.16.18 SpaCy

Industrial-strength Natural Language Processing (NLP) with Python and Cython

Features:
• Non-destructive **tokenization**
• **Named entity** recognition
• Support for 26+ **languages**
• 13 **statistical models** for 8 languages
• Pre-trained **word vectors**
• Easy **deep learning** integration
• Part-of-speech tagging
• Labelled dependency parsing
• Syntax-driven sentence segmentation
• Built in **visualizers** for syntax and NER
• Convenient string-to-hash mapping
• Export to numpy data arrays
• Efficient binary serialization
• Easy **model packaging** and deployment
• State-of-the-art speed

2.16. NLP Frameworks
• Robust, rigorously evaluated accuracy

<table>
<thead>
<tr>
<th>SBB License</th>
<th>MIT License</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Python</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="https://spacy.io/">https://spacy.io/</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/explosion/spaCy">https://github.com/explosion/spaCy</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>NLP</td>
</tr>
</tbody>
</table>

### 2.16.19 Stanford CoreNLP

Stanford CoreNLP provides a set of human language technology tools. It can give the base forms of words, their parts of speech, whether they are names of companies, people, etc., normalize dates, times, and numeric quantities, mark up the structure of sentences in terms of phrases and syntactic dependencies, indicate which noun phrases refer to the same entities, indicate sentiment, extract particular or open-class relations between entity mentions, get the quotes people said, etc.

Choose Stanford CoreNLP if you need:

- An integrated NLP toolkit with a broad range of grammatical analysis tools
- A fast, robust annotator for arbitrary texts, widely used in production
- A modern, regularly updated package, with the overall highest quality text analytics
- Support for a number of major (human) languages
- Available APIs for most major modern programming languages
- Ability to run as a simple web service

<table>
<thead>
<tr>
<th>SBB License</th>
<th>GNU General Public License (GPL) 3.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Java</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="https://stanfordnlp.github.io/CoreNLP/">https://stanfordnlp.github.io/CoreNLP/</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/stanfordnlp/CoreNLP">https://github.com/stanfordnlp/CoreNLP</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>NLP</td>
</tr>
</tbody>
</table>

### 2.16.20 Sumeval

Well tested & Multi-language evaluation framework for text summarization. Multi-language.
2.16.21 Texar-PyTorch

Texar-PyTorch is a toolkit aiming to support a broad set of machine learning, especially natural language processing and text generation tasks. Texar provides a library of easy-to-use ML modules and functionalities for composing whatever models and algorithms. The tool is designed for both researchers and practitioners for fast prototyping and experimentation.

Texar-PyTorch integrates many of the best features of TensorFlow into PyTorch, delivering highly usable and customizable modules superior to PyTorch native ones.

2.16.22 TextBlob: Simplified Text Processing

TextBlob is a Python (2 and 3) library for processing textual data. It provides a simple API for diving into common natural language processing (NLP) tasks such as part-of-speech tagging, noun phrase extraction, sentiment analysis, classification, translation, and more.

2.16.23 Features

- Noun phrase extraction
- Part-of-speech tagging
- Sentiment analysis
- Classification (Naive Bayes, Decision Tree)
- Language translation and detection powered by Google Translate
- Tokenization (splitting text into words and sentences)
- Word and phrase frequencies
- Parsing
- n-grams
- Word inflection (pluralization and singularization) and lemmatization
- Spelling correction
- Add new models or languages through extensions
- WordNet integration

<table>
<thead>
<tr>
<th>SBB License</th>
<th>MIT License</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Python</td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/sloria/textblob">https://github.com/sloria/textblob</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>NLP, Python</td>
</tr>
</tbody>
</table>

### 2.16.24 Thinc

Thinc is the machine learning library powering spaCy. It features a battle-tested linear model designed for large sparse learning problems, and a flexible neural network model under development for spaCy v2.0.

Thinc is a lightweight deep learning library that offers an elegant, type-checked, functional-programming API for composing models, with support for layers defined in other frameworks such as PyTorch, TensorFlow and MXNet. You can use Thinc as an interface layer, a standalone toolkit or a flexible way to develop new models.

Thinc is a practical toolkit for implementing models that follow the “Embed, encode, attend, predict” architecture. It’s designed to be easy to install, efficient for CPU usage and optimised for NLP and deep learning with text – in particular, hierarchically structured input and variable-length sequences.

<table>
<thead>
<tr>
<th>SBB License</th>
<th>MIT License</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Technology</td>
<td>Python</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="https://thinc.ai/">https://thinc.ai/</a></td>
</tr>
<tr>
<td>Source Location</td>
<td><a href="https://github.com/explosion/thinc">https://github.com/explosion/thinc</a></td>
</tr>
<tr>
<td>Tag(s)</td>
<td>ML, ML Framework, NLP, Python</td>
</tr>
</tbody>
</table>

### 2.16.25 Torchtext

Data loaders and abstractions for text and NLP. Build on PyTorch.
### 2.16.26 Transformers

Transformers (formerly known as pytorch-transformers and pytorch-pretrained-bert) provides state-of-the-art general-purpose architectures (BERT, GPT-2, RoBERTa, XLM, DistilBert, XLNet…) for Natural Language Understanding (NLU) and Natural Language Generation (NLG) with over 32+ pretrained models in 100+ languages and deep interoperability between TensorFlow 2.0 and PyTorch.

Features:
- As easy to use as pytorch-transformers
- As powerful and concise as Keras
- High performance on NLU and NLG tasks
- Low barrier to entry for educators and practitioners

State-of-the-art NLP for everyone:
- Deep learning researchers
- Hands-on practitioners
- AI/ML/NLP teachers and educators

Lower compute costs, smaller carbon footprint
- Researchers can share trained models instead of always retraining
- Practitioners can reduce compute time and production costs
- 8 architectures with over 30 pretrained models, some in more than 100 languages

---

## 2.17 ML Learning resources

Learning machine learning does not have to be very expensive or time consuming. Great learning material for machine learning is licensed under a Creative Commons license. For starters but also people who are already more familiar with the key concepts.
This section presents an opinionated list of great machine learning learning resources. A lot of garbage is produced on the internet and even paid courses are often not that good. But most material released under an open license is of excellent quality. This list consist of very readable references and some great hands-on courses.

Only resources that are real open, so resources published using a Creative Commons license (cc-by mostly) or other types of real open licensed material is included.

Most learning resources include hands-on tutorials. So be ready to use a notebook, but most tutorials offer notebooks ready to use directly.

- A Course in Machine Learning, http://ciml.info/
- Cryptography and Machine Learning, Mixing both for privacy-preserving machine learning, https://mortendahl.github.io/
- Dive into Deep Learning, An interactive deep learning book with code, math, and discussions, https://d2l.ai/
2.17. ML Learning resources


- Machine Learning Crash Course with TensorFlow APIs, https://developers.google.com/machine-learning/crash-course/ This is a great course published by Google’s. It is advertised as a ‘A self-study guide for aspiring machine learning practitioners’


• Seeing Theory, A visual introduction to probability and statistics. Interactive learning book that visualizes the fundamental statistical concepts, https://seeing-theory.brown.edu/


• The Elements of AI, learn the basics of AI, https://www.elementsofai.com/

• TensorFlow, Keras and deep learning, without a PhD, https://codelabs.developers.google.com/codelabs/cloud-tensorflow-mnist/#0
2.18 NLP Learning resources

There is a large overlap between machine learning and current NLP technology. But it makes sense to outline specific NLP resources separately. This to make searching for good open NLP resources easier.

So this section is an opinionated list of great NLP learning resources. Of course also only resources that are open, so only resources published using a Creative Commons license (cc-by mostly) or other real open licenses are included. So all references are open access resources.


- Advanced NLP with spaCy, https://course.spacy.io/

- NLP concepts with spaCy (notebook) https://gist.github.com/nocomplexity/b7c4c0aa5a0b53f4f5ff1c4784084be6

2.19 Help

This publication on FOSS machine learning is created to be shared as much as possible. This to make sure machine learning remains a full open technology that can be used by everyone. And remember FOSS machine learning is all about Freedom. So commercial development of Free and Open Machine Learning technology is very important for sustainable development of a healthy FOSS machine learning ecosystem. And of course FOSS machine learning means using open principles.

You can contribute to FOSS machine learning in a simple way. This publication is created to be shared as much as possible. So please:

Tip: Share this report!

This publication is created as a starting and living document to support FOSS machine learning. So this publication is likely to be incomplete or outdated if not maintained. I will update it frequently but I also need your help. So I encourage all information professionals to help to improve this Free and Open Machine Guide.

FOSS machine learning is crucial for open innovation the coming years. Using and applying Machine learning technology should be in reach for all companies worldwide with no strings attached.

The core focus in this first publication was to explained why FOSS machine learning is needed and showed that you or your company can make the choice to use open machine learning. Even for real business use cases.

Using FOSS building blocks for machine learning does not mean that you cannot benefit from the great advantages that some (Cloud)Companies offer. FOSS machine learning is an enabler for business innovation. So guard your freedom, but also take responsibility for the freedom of others that benefit from your machine learning application(s).
In a next version of this publication your input is needed to improve the practical use of FOSS machine learning in practice. So if you:

- Like to share your real life experience with FOSS machine learning.
- Want to share your experience on using FOSS machine learning tools or ML building blocks.
- Like to share resources that should be mentioned in this FOSS machine learning publication.
- Discuss the content so it gets better. We do discussion on-line and off-line (meetups).

**Tip:** Feel free to contact me or create a pull request on github.

### 2.19.1 Contributors

The following people have contributed to the Free and Open Machine project and this publication:

[name] [OPTIONAL email] [Optional Organization name ]

If you like your name stated here: This book is open source. Issues and pull requests are welcome. All contributors will be added to this list.

**So Get involved in the discussion to make it better!**

If you wish to make comments regarding this document, please raise them as GitHub issues. Or send comments by email if you are unable to raise issues on GitHub. All input is welcome!

### 2.20 About

This publication to fight for real Free and Open Machine learning is initially started and created by Maikel Mardjan. Maikel is a hands-on practical business IT architect and loves to make simple designs for complex IT systems. Maikel has more than 25 years of relevant experience on various IT roles in famous (international) companies. Maikel holds both a Master (Msc) Business Studies of University of Groningen ([https://www.rug.nl/](https://www.rug.nl/)) and a Master degree (Msc) Electrical Engineering, of Delft University of Technology ([https://www.tudelft.nl/en/](https://www.tudelft.nl/en/)). Maikel is TOGAF 9 Certified and CISSP (Certified Information Systems Security Professional) certified. Maikel is also an OWASP member ([https://owasp.org/](https://owasp.org/)) and supporter.

Check [https://nocomplexity.com](https://nocomplexity.com) for more information about Maikel.

Machine learning is a complex technology. So we need simple and Free and Open solutions to create applications so that will solve complex problems we humans face. To trust machine learning applications there is simply no other option than using fully transparent technologies. So Free and Open in the spirit of the Free Software Foundation ([https://fsf.org](https://fsf.org)).

If you or your company is committed to openness make sure to support the BM-Support.org Foundation. Supporting this foundation is free! Check [https://www bm-support.org join/](https://www.bm-support.org/join/)

This publication would never have reached version 1.0 without your help. So I gratefully thank all people who devote time and knowledge to give input to this publication. Will we continue this FOSS machine learning journey so machine learning technology will stay Free and Open so everyone can benefit.

### 2.21 License

Copyright (c) 2018-2020 BM-Support.org and Maikel Mardjan.
This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License. Third-party product names may be the trademarks of their respective owners.

See http://creativecommons.org/licenses/by-sa/4.0/ for the full license text or here below:

You are free to: - Share — copy and redistribute the material in any medium or format - Adapt — remix, transform, and build upon the material for any purpose, even commercially.

The licensor cannot revoke these freedoms as long as you follow the license terms.

Under the following terms:

• Attribution — You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.

• ShareAlike — If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original.

No additional restrictions — You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits.

Notices:

You do not have to comply with the license for elements of the material in the public domain or where your use is permitted by an applicable exception or limitation.

No warranties are given. The license may not give you all of the permissions necessary for your intended use. For example, other rights such as publicity, privacy, or moral rights may limit how you use the material.