Week 1: Setting up your Data Science Environment

* The Hitchhiker’s Guide to Python: best practice handbook to the installation, configuration, and usage of Python - <https://docs.python-guide.org/>
* Anaconda – One of the most popular Python data science distributions: <https://wiki.ubuntuusers.de/Anaconda/>
* Python
* Spyder IDE
* VS Code IDE for Python
* Jupyter Notebook – open source web application supporting live code, equations, visualizations etc.
* Package management tools for Python
* Conda: <https://conda.io/docs/>
* pip: <https://pypi.org/project/pip/>
* Virtual Environments
* Pip Virtual Environments: <https://docs.python-guide.org/dev/virtualenvs/>
* Conda Environments: <https://conda.io/docs/user-guide/tasks/manage-environments.html>
* Working with Jupyter Notebooks: <https://www.cheatography.com/weidadeyue/cheat-sheets/jupyter-notebook/>
* Google Colaboratory: Now public former internal tool for machine learning workflow built upon top of Jupyter Notebook. Additional free computing power on Google servers. <https://colab.research.google.com/notebooks/welcome.ipynb>
* Structuring Data Science Projects: [https://gitpitch.com/paroscha/data\_science\_projects#/](https://gitpitch.com/paroscha/data_science_projects)
* Cookiecuttier: command-line utility that creates project from cookiecutter templates - [https://github.com/audreyr/cookiecutter#data-science](https://github.com/audreyr/cookiecutter)
* Cookiecutter: Better Project Templates - <https://cookiecutter.readthedocs.io/en/latest/>
* Project Documentation
* Markdown vs. CommonMark

<http://ericholscher.com/blog/2016/mar/15/dont-use-markdown-for-technical-docs/>

<https://commonmark.org/>

* reStructuredText vs. Markdown for documentation <http://www.zverovich.net/2016/06/16/rst-vs-markdown.html>

<http://docutils.sourceforge.net/docs/user/rst/quickref.html>

* Libraries for documentation
* Sphinx (reStructuredText, used by scikit-learn): <http://www.sphinx-doc.org/en/master/>
* MKdocs (Markdown, used by keras): <https://www.mkdocs.org/>
* Version Control for Data Science – GitHub / GitLab / Bitbucket
* Git workflow and collaboration – Interactive guide for git branching: <https://learngitbranching.js.org/>

Week 1: Deep Learning with Python

- Tutor: Dr. Tristan Behrens <http://ai-guru.de/>

Until around 2015, Artificial Neural Networks were regarded as just one specific subset of the field of Machine Learning. Since you need serious amounts of multi-core computing power to build and train deep neural networks, the rise of deep learning did not really begin until the processing power of modern GUPs became accessible for Machine Learning. Hence, within the last few years, a categorization of Machine Learning into Deep Learning and Classical Machine Learning (anything but Deep Learning, e.g. Linear Models, Tree-Based-Models, SVM etc.) is widely applied.

* Different kinds of neural networks
* ANN (Artificial Neural Network)
* CNN (Convolutional Neural Network)
* RNN (Recurrent Neural Network)
* LSTM (Long short-term memory)
* GRU (Gated recurrent unit)
* Deep Learning packages in Python
* TensorFlow
* Keras <https://keras.io/>
* Numpy
* Sklearn
* Network architecture
* Nonlinearities (activation functions)
* Loss-Functions
* Metrics
* Optimization Algorithms
* Line-Search Algorithms
* Validation & Hyperparameter tuning
* Applications
* NLP (Natural Language Processing) with CNN
* Image Recognition with CNN
* Time Series Analysis with LSTM
* TSA Performance Comparison: ARIMA vs. LSTM <https://rpubs.com/zkajdan/316135>

Week 1 - SQL for Data Scientists

* Tutor: Dania Meira <http://www.linkedin.com/in/daniameira>
* PostgreSQL: open source object-relational database management system <https://www.postgresql.org/>
* DataGrip: cross-platform IDE for working with SQL Databases <https://www.jetbrains.com/datagrip/>
* MySQL
* PostgreSQL
* Amazon Redshift
* DB2
* Derby
* Exasol
* Sybase
* SQL Server
* Microsoft Azure
* Oracle
* H2
* HSQLDB
* Sqlite
* MariaDB
* ModeAnalytics – free data analysis platform for SQL, Python & R: <https://modeanalytics.com/>

Week 2 - Classical Machine Learning Fundamentals

* Tutor: Gerrit A. Gruben <https://de.linkedin.com/in/ggruben>
* The five fundamental schools of Machine Learning: <https://medium.com/42ai/the-5-tribes-of-the-ml-world-670ebce96b4>
* Fundamental Tasks:
* Supervised Learning:
* Classification
* Regression
* Recommendations
* Rankings
* Structured Prediction
* Unsupervised Learning:
* Clustering
* ‘Feature Learning’
* Latent Factors
* Dimensionality Reduction (de-noising / compression)
* Fundamental Meta-Problems:
* How to parallelize / distribute computations? By data points / attributes?
* How take several learners and improve them? (Boosting, Ensembling)
* How good can our models generalize? (Statistical Learning Theory, Empirical Risk Minimization)
* What hardware let us do learning the most efficient? (GPU, FPGAs)
* Fundamental Models:
* (Generalized) Linear Models
* Bayesian Inference & Learning
* Non-Parametric Models (KNN etc.)
* Common Machine Learning errors:
* p-hacking: A Tutorial on Hunting Statistical Significance by Chasing N: <https://www.frontiersin.org/articles/10.3389/fpsyg.2016.01444/full>
* Underfitting / Overfitting
* Forget to regularize your model
* Mixing up validation & evaluation
* On Over-fitting in Model Selection and Subsequent Selection Bias in Performance Evaluation: <http://www.jmlr.org/papers/volume11/cawley10a/cawley10a.pdf>
* Performance-Estimation Properties of Cross-Validation-Based Protocols with Simultaneous Hyper-Parameter Optimization: <https://www.researchgate.net/publication/284030151_Performance-Estimation_Properties_of_Cross-Validation-Based_Protocols_with_Simultaneous_Hyper-Parameter_Optimization>
* Different settings of learning
* Batch Learning
* Online Learning with feedback loops
* Transfer Learning

Addendum: ML Fundamentals & Learning Theory:

* A Few Useful Things to Know about Machine Learning: <https://homes.cs.washington.edu/~pedrod/papers/cacm12.pdf>
* <https://work.caltech.edu/telecourse>
* On nested cross validation: <https://www.quora.com/I-train-my-system-based-on-the-10-fold-cross-validation-framework-Now-it-gives-me-10-different-models-Which-model-to-select-as-a-representative>
* Accurately Measuring Model Prediction Error <http://scott.fortmann-roe.com/docs/MeasuringError.html>
* Understanding the Bias-Variance Tradeoff <http://scott.fortmann-roe.com/docs/BiasVariance.html>
* [From D's list] Visuals for Model Tuning and Bias-Variance Tradeoff: <http://www.r2d3.us/visual-intro-to-machine-learning-part-2/>

Week 2 - Tree-Based Methods

* Tutor: Dr. Jose Quesada <https://www.datascienceretreat.com/dr-jose-quesada/?locale=en>
* Why AdaBoost works: <https://www.cs.princeton.edu/courses/archive/spring07/cos424/papers/boosting-survey.pdf>

Zero-Shot / Low-Shot Learning

* Tutor: Nour Karessli

Presentation skills

* Tutor: Kevin Wong
* Videos (bis Ende Dezember): <https://www.mediafire.com/folder/z8idqf8fpoc7v/Videos>

Deep Learning with Pytorch / Skorch

* Tutor: Daniel Nouri <https://github.com/dnouri>

Spark

* Tutor: Daniel Voight Godoy: <https://github.com/dvgodoy>
* Install Docker: <https://docs.docker.com/docker-for-mac/install/>
* Comprehensive Introduction to Apache Spark, RDDs & Dataframes (using PySpark): <https://www.analyticsvidhya.com/blog/2016/09/comprehensive-introduction-to-apache-spark-rdds-dataframes-using-pyspark/>
* Understanding Spark Caching: <https://spark.rstudio.com/guides/caching/>
* Datacamp: Beginner’s guide to Apache Spark: <https://www.datacamp.com/community/tutorials/apache-spark-python>
* Spark Closures: [https://spark.apache.org/docs/latest/rdd-programming-guide.html#understanding-closures-](https://spark.apache.org/docs/latest/rdd-programming-guide.html)

<https://data-flair.training/blogs/scala-closures/>

* <https://www.datacamp.com/community/tutorials/apache-spark-python>
* Understanding Caching in Spark: <https://spark.rstudio.com/guides/caching/>
* Vectorized UDF for PySpark: <https://databricks.com/blog/2017/10/30/introducing-vectorized-udfs-for-pyspark.html>

Advanced Python

* Tutor: Marek Gagolewski
* <https://docs.python.org/3/reference/datamodel.html>
* [https://docs.python.org/3/reference/expressions.html#operator-precedence](https://docs.python.org/3/reference/expressions.html)
* <https://pandas.pydata.org/pandas-docs/stable/groupby.html>
* <https://pandas.pydata.org/pandas-docs/stable/api.html>
* really helpful tool to visualize how list objects work behind the scenes: [http://pythontutor.com/visualize.html#mode=edit](http://pythontutor.com/visualize.html)

Evolutionary Strategies

* Tutor: Shreyas Gite
* Evolution Strategies as a Scalable Alternative to Reinforcement Learning <https://blog.openai.com/evolution-strategies/>
* Genetic algorithms as a competitive alternative for training deep neural <https://eng.uber.com/deep-neuroevolution/>
* A Visual Guide to Evolution Strategies <http://blog.otoro.net/2017/10/29/visual-evolution-strategies/>
* Gym: toolkit for developing and comparing reinforcement learning algorithms: <https://gym.openai.com/>
* Can agents learn inside of their own dreams: <https://worldmodels.github.io/>

Pipelines

* Tutor: Rachel Berryman
* Scikit-learn: choosing the right estimator <http://scikit-learn.org/stable/tutorial/machine_learning_map/index.html>

Reinforcement Learning

* Tutor: Adam Green

Natural Language Processing

* Tutor: Tal

Not classified:

* Python Tutorial on Scipy-Lectures: <http://www.scipy-lectures.org/intro/language/python_language.html>
* Python Tutorial on tutorialspoint: <https://www.tutorialspoint.com/python/python_loops.htm>
* String Formatting in Python: <https://pyformat.info/>
* Numpy Tutorial on tutorialspoint: <https://www.tutorialspoint.com/numpy/>
* 100 numpy exercises: The goal is both to offer a quick reference for new and old users and to provide also a set of exercises for those who teach: <http://www.labri.fr/perso/nrougier/teaching/numpy.100/>
* A step by step guide on preparing and submitting a pull request: <https://github.com/PointCloudLibrary/pcl/wiki/A-step-by-step-guide-on-preparing-and-submitting-a-pull-request>
* Version control with Git: <https://swcarpentry.github.io/git-novice/>
* Intuitive Explanation for Neural Networks: <https://www.quora.com/What-is-an-intuitive-explanation-for-neural-networks>
* Librosa python package for music and audio analysis: <https://librosa.github.io/librosa/>
* Dota OpenAI Five Benchmark: Results <https://blog.openai.com/openai-five-benchmark-results/>
* R-like reverse indexing in Python: <https://docs.scipy.org/doc/numpy/reference/generated/numpy.delete.html>
* Python Operator Overloading: <https://www.programiz.com/python-programming/operator-overloading>
* REDUCING WORLD HUNGER. DEEP LEARNING IN HUMANITARIAN WORK <http://ai-guru.de/reducing-world-hunger-deep-learning-in-humanitarian-work/>
* 3D – Learning:
* Voxnet – 3D/Volumetric Convolutional Neural Network with Theano + Lasagne <https://github.com/dimatura/voxnet>
* 3D Machine Learning: <https://github.com/timzhang642/3D-Machine-Learning>
* The keras blog: How CNNs see the world: <https://blog.keras.io/how-convolutional-neural-networks-see-the-world.html>
* PyEnv is the new Conda: <https://bastibe.de/2017-11-20-pyenv.html>
* Google developers Machine Learning Glossary: <https://developers.google.com/machine-learning/glossary/>
* Data Cleaning Tutorial with the Titanic dataset: <https://www.kaggle.com/pmarcelino/data-analysis-and-feature-extraction-with-python>
* YouTube-Lecture Reinforcement Learning as recommended by Christian: <https://www.youtube.com/watch?v=w33Lplx49_A>
* [Heroku Postgres](https://elements.heroku.com/addons/heroku-postgresql) is a managed SQL database service provided directly by Heroku. You can access a Heroku Postgres database from any language with a PostgreSQL driver, including all languages [officially supported by Heroku](https://www.heroku.com/languages). <https://www.heroku.com/>
* Data Science workflow at Netflix: <https://medium.com/netflix-techblog/tagged/data-science>
* Netflix & Jupyter Notebooks: <https://medium.com/netflix-techblog/notebook-innovation-591ee3221233>
* Scikit-Learn Tutorial: <https://colab.research.google.com/github/jakevdp/PythonDataScienceHandbook/blob/master/notebooks/05.02-Introducing-Scikit-Learn.ipynb>
* Gerrit Gruben: Levelling up your Jupyter Notebook skills <https://github.com/uberwach/leveling-up-jupyter> + YouTube video: <https://www.youtube.com/watch?v=b8g-8T0amuk>
* Visualising high-dimensional datasets using PCA and t-SNE in Python: <https://medium.com/@luckylwk/visualising-high-dimensional-datasets-using-pca-and-t-sne-in-python-8ef87e7915b>
* Gerrit-recommended package for automated hyperparameter search w/ sklearn integration: <https://github.com/hyperopt/hyperopt-sklearn>
* Credible Interval vs. Confidence Interval: <https://en.wikipedia.org/wiki/Credible_interval>
* Article on Leading Countries in AI-Space: <https://qz.com/1264673/ai-is-the-new-space-race-heres-what-the-biggest-countries-are-doing/>

<https://www.futuresplatform.com/blog/5-countries-leading-way-ai-artificial-intelligence-machine-learning>

* Outlier detection: <https://machinelearningmastery.com/how-to-identify-outliers-in-your-data/>
* Icons for everything: <https://thenounproject.com/>
* Job offer: Machine Intelligence Engineer at Tensor Technologies: <https://angel.co/tensor-technologies/jobs/373415-machine-intelligence-engineer>
* Pandas-like Data frames in Spark: <https://towardsdatascience.com/announcing-optimus-v2-agile-data-science-workflows-made-easy-c127a12d9e13>
* Command-line tutorial: <https://www.learnenough.com/command-line-tutorial>
* Learn OpenCV – Fine-tuning using pre-trained models: <https://www.learnopencv.com/keras-tutorial-fine-tuning-using-pre-trained-models/>
* Marc Andreessen: career advices: <https://pmarchive.com/guide_to_career_planning_part0.html>
* Palladium – predictive analytics services as web service: <https://palladium.readthedocs.io/en/latest/>
* A practical guide to PCA in Python & R: <https://www.analyticsvidhya.com/blog/2016/03/practical-guide-principal-component-analysis-python/>
* How-to Machine Learning Engineer: <https://hackernoon.com/i-want-to-learn-artificial-intelligence-and-machine-learning-where-can-i-start-7a392a3086ec>
* Slawek’s hybrid Exponential Smoothing-Recurrent Neural Networks (ES-RNN) method <https://eng.uber.com/m4-forecasting-competition/>
* A time-series library for Apache Spark: <https://github.com/twosigma/flint>
* A library for time series analysis on Apache Spark: <https://github.com/sryza/spark-timeseries>
* Difference between RNN /LSTM /GRU: <https://medium.com/@saurabh.rathor092/simple-rnn-vs-gru-vs-lstm-difference-lies-in-more-flexible-control-5f33e07b1e57>
* [From Gareth] Great Visuals about Fourier transform: <https://www.youtube.com/watch?v=spUNpyF58BY>
* [From Christian] Why Random Forest is my favorite ML-model <https://towardsdatascience.com/why-random-forest-is-my-favorite-machine-learning-model-b97651fa3706>
* To Remove all barriers in the way of science: sci-hub.tw
* Unit-Tests for Data (on top of Spark): <https://github.com/awslabs/deequ>
* Linguistic Entity Recognition System: <https://spacy.io/usage/linguistic-features#entity-types>
* Pythex -test your python regular expressions online: <https://pythex.org/>
* Practical Text Classification with Python and Keras: <https://realpython.com/python-keras-text-classification/>
* Keras Applications – pretrained deep learning models: <https://keras.io/applications/#vgg19>
* Chihuahua or muffin? My search for the best computer vision API: <https://medium.freecodecamp.org/chihuahua-or-muffin-my-search-for-the-best-computer-vision-api-cbda4d6b425d>
* An overview of gradient descent optimization algorithms: [ruder.io/opti HYPERLINK "ruder.io/optimizing-gradient-descent/" HYPERLINK "ruder.io/optimizing-gradient-descent/" HYPERLINK "ruder.io/optimizing-gradient-descent/"m HYPERLINK "ruder.io/optimizing-gradient-descent/" HYPERLINK "ruder.io/optimizing-gradient-descent/" HYPERLINK "ruder.io/optimizing-gradient-descent/"izing-gradient-descent/](ruder.io/optimizing-gradient-descent/)
* How to Develop an Encoder-Decoder Model for Sequence-to-Sequence Prediction in Keras: <https://machinelearningmastery.com/develop-encoder-decoder-model-sequence-sequence-prediction-keras/>
* Applications of Autoencoders in Natural Language Processing: <https://www.doc.ic.ac.uk/~js4416/163/website/nlp/>
* Applied Deep Learning – Autoencoders: <https://towardsdatascience.com/applied-deep-learning-part-3-autoencoders-1c083af4d798>
* Using TensorFlow Autoencoders with Music: <https://blog.goodaudience.com/using-tensorflow-autoencoders-with-music-f871a76122ba>
* A Practical guide to Autoencoders: <https://sadanand-singh.github.io/posts/autoencoders/>
* Convolutional Autoencoders: Clustering Images with Neural Networks: <https://sefiks.com/2018/03/23/convolutional-autoencoder-clustering-images-with-neural-networks/>
* Variational Autoencoders: Melody 2-bar “Loop” Interpolation: <https://www.youtube.com/watch?v=G5JT16flZwM>
* Working with the python super function: <https://www.pythonforbeginners.com/super/working-python-super-function>
* Batch Normalization: <https://keras.io/layers/normalization/>
* Gentle Introduction to the Adam Optimization Algorithm for Deep Learning: <https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/>
* Network-Based Job Search: <https://www.thinkful.com/projects/network-based-job-search-591/>
* AI-Art at Christie’s – created by GAN: <https://www.nytimes.com/2018/10/25/arts/design/ai-art-sold-christies.html>
* What are Siamese neural networks, what applications are they good for, and why? <https://www.quora.com/What-are-Siamese-neural-networks-what-applications-are-they-good-for-and-why>
* Frequently Asked Deep Learning Questions During Interview Round By Professionals: <https://www.houseofbots.com/news-detail/3874-1-frequently-asked-deep-learning-questions-during-interview-round-by-professionals>
* Object Oriented Python basics via Flashcards: <https://www.flashcardsfordevelopers.com/decks/5bd0192d320599b44536b216/cards>
* PyPy Status Blog: Reverse debugging for Python: <https://morepypy.blogspot.com/2016/07/reverse-debugging-for-python.html>
* DataFramed – DataCamp’s official podcast: <https://www.datacamp.com/community/podcast>
* Andrew Ng: Machine Learning Learning – how do you organize and AI project? [www.mlyearning.org/](http://www.mlyearning.org/)
* Scott Fortmann-Row – Accurately Measuring Model Prediction Error: <http://scott.fortmann-roe.com/docs/MeasuringError.html>
* Scott Fortmann-Row: Understanding the Bias-Variance Tradeoff: <http://scott.fortmann-roe.com/docs/BiasVariance.html>
* Why Random Forest is my favorite Machine Learning Model: <https://towardsdatascience.com/why-random-forest-is-my-favorite-machine-learning-model-b97651fa3706>
* Animierte Einführung in die Fouriertransformation: <https://www.youtube.com/watch?v=spUNpyF58BY>
* Understanding LSTM Networks: colah.github.io/posts/2015-08-Understanding-LSTMs/
* Node.js in one hour: <https://www.youtube.com/watch?v=TlB_eWDSMt4>
* Command Line Challenge: <https://cmdchallenge.com/>
* The Art of Spaced Repetition: <https://ncase.me/remember/>
* A Google ML engineer’s guide to entering the field: <https://80000hours.org/articles/ml-engineering-career-transition-guide/>
* Daniel Hnyk: Creating your own estimator in scikit-learn: danielhnyk.cz/creating-your-own-estimator-scikit-learn/
* Daniel Godoy’s package to make PySpark easier: <https://towardsdatascience.com/handyspark-bringing-pandas-like-capabilities-to-spark-dataframes-5f1bcea9039e> --> GitHub: <https://github.com/dvgodoy/handyspark>
* [from Rachel Berryman] Chartify – a new visualization package made by Spotify: <https://github.com/spotify/chartify/>
* [Daniel Godoy’s promised list of ML materials] <https://gist.github.com/dvgodoy/fa570f3727eaba250ffc7abd538dd515>
* [Tristan’s blog] How to not get robbed – use deep reinforcement learning: ai-guru.de/how-to-not-get-robbed-use-deep-reinforcement-learning/

Christian Wegmann – Building PowerApps

* Designing a relational data model: <https://sqldbm.com/en/Home/>
* Icons: <https://thenounproject.com/>
* Creating a corporate identity: <https://studio.tailorbrands.com/>
* Generating appealing color palettes: <https://coolors.co/app>
* PowerApps: <https://powerapps.microsoft.com/>

Drone-Steering / Gesture Recognition:

* Introductory YouTube videos to PoseNet:
* <https://www.youtube.com/watch?v=PCBTZh41Ris>
* <https://www.youtube.com/watch?v=cEBgi6QYDhQ>
* PoseNet-GitHub: <https://github.com/tensorflow/tfjs-models/tree/master/posenet>
* Is there a way to export the serialized PoseNet model, in order to import it into python tf: <https://github.com/tensorflow/tfjs/issues/572>
* PoseNet: Real-time Human Pose Estimation in the Browser with TensorFlow.js: <https://medium.com/tensorflow/real-time-human-pose-estimation-in-the-browser-with-tensorflow-js-7dd0bc881cd5>
* PoseNet webcam demo: <https://storage.googleapis.com/tfjs-models/demos/posenet/camera.html>
* Converter From PoseNet(TensorFlowJS) to CoreML-Model: <https://github.com/infocom-tpo/PoseNet-CoreML/tree/master/converter>
* amongst others tutorials how one can train a model from webcam data, how we can import existing Keras models in tensorflow.js: <https://js.tensorflow.org/tutorials/>
* Paper: Hand Gesture Controlled Drones: An Open Source Library: <https://arxiv.org/pdf/1803.10344.pdf>
* PoseNet CoreML: <https://github.com/infocom-tpo/PoseNet-CoreML>
* Keras Temporal Conv Network: <https://github.com/philipperemy/keras-tcn>
* Learning Acrobatics by Watching YouTube: <https://bair.berkeley.edu/blog/2018/10/09/sfv/>
* NTU RGB+D: A Large Scale Dataset for 3D Human Activity Analysis <https://www.cv-foundation.org/openaccess/content_cvpr_2016/papers/Shahroudy_NTU_RGBD_A_CVPR_2016_paper.pdf>

Sequence Classification:

* How to Reshape Input Data for Long Short-Term Memory Networks in Keras <https://machinelearningmastery.com/reshape-input-data-long-short-term-memory-networks-keras/>
* How to prepare Univariate Time Series Data for Long Short-Term Memory Networks <https://machinelearningmastery.com/prepare-univariate-time-series-data-long-short-term-memory-networks/>
* How to convert a time series to a supervised learning problem in Python <https://machinelearningmastery.com/convert-time-series-supervised-learning-problem-python/>
* Sequence classification with LSTM Recurrent Neural Networks in Python with Keras <https://machinelearningmastery.com/sequence-classification-lstm-recurrent-neural-networks-python-keras/>
* How to Develop a Bidirectional LSTM For Sequence Classification in Python with Keras <https://machinelearningmastery.com/develop-bidirectional-lstm-sequence-classification-python-keras/>
* Time Series Classification from Scratch with Deep Neural Networks: A Strong Baseline <https://arxiv.org/pdf/1611.06455.pdf>
* Gesture Recognition with a Convolutional Long Short-Term Memory Recurrent Neural Network <https://www2.informatik.uni-hamburg.de/wtm/ps/Tsironi_ESANN_2016.pdf>
* Supervised Sequence Labelling with Recurrent Neural Networks <https://www.cs.toronto.edu/~graves/preprint.pdf>
* Training LSTMs in Keras with time series of different length: <https://stackoverflow.com/questions/50748986/training-lstms-in-keras-with-time-series-of-different-length>

Vizdoom:

* Tutorial: <http://vizdoom.cs.put.edu.pl/tutorial>

DinoRun:

* A DinoRun-Tutorial: <https://blog.paperspace.com/dino-run/>
* <https://github.com/Paperspace/DinoRunTutorial>

Books:

* Neural Networks and Deep Learning [Charu C. Aggarwal]
* Deep Learning with Python [Francois Chollet]
* Machine Learning with Python [Jason Brownlee]
* Reinforcement Learning: An Introduction [Sutton/Barto]
* Bishop - Pattern Recognition And Machine Learning - Springer 2006
* Think Python - How to Think Like a Computer Scientist - 2nd Edition
* [From José] Feature Engineering for Machine Learning; Alice Zheng, Amanda Casari: <https://www.goodreads.com/book/show/31393737-feature-engineering-for-machine-learning?from_search=true>

Datasets:

* Boston Housing <https://www.kaggle.com/c/boston-housing>
* Home Credit Default Risk: <https://www.kaggle.com/c/home-credit-default-risk>
* Dogs vs. Cats <https://www.kaggle.com/c/dogs-vs-cats/data>
* Find Datasets with Google: <https://toolbox.google.com/datasetsearch>
* Source for Health Care Datasets: <https://hacking-health.org/berlin/>
* bigml Machine Learning Datasets – Healthcare: <https://bigml.com/gallery/datasets/healthcare?reload>

Additional Reading (papers):

* Neural Feature Learning From Relational Database: <https://arxiv.org/pdf/1801.05372.pdf>
* Google Research paper on Machine Learning Applications for Data Center Optimization: <https://static.googleusercontent.com/media/research.google.com/de//pubs/archive/42542.pdf>
* Performance-Estimation Properties of Cross-Validation-Based Protocols with Simultaneous Hyper-Parameter Optimization: <https://www.researchgate.net/publication/284030151_Performance-Estimation_Properties_of_Cross-Validation-Based_Protocols_with_Simultaneous_Hyper-Parameter_Optimization>
* Audio & Speech Processing - CREPE: A CONVOLUTIONAL REPRESENTATION FOR PITCH ESTIMATION <http://www.justinsalamon.com/uploads/4/3/9/4/4394963/kim_crepe_icassp_2018.pdf>
* Learning from Synthetic Humans: <https://arxiv.org/pdf/1701.01370.pdf>
* The Unreasonable Effectiveness of Data: <https://static.googleusercontent.com/media/research.google.com/de//pubs/archive/35179.pdf>
* A Tutorial on Hunting Statistical Significance by Chasing N: <https://www.frontiersin.org/articles/10.3389/fpsyg.2016.01444/full>
* On Over-fitting in Model Selection and Subsequent Selection Bias in Performance Evaluation: <http://www.jmlr.org/papers/volume11/cawley10a/cawley10a.pdf>
* <https://machinelearningmastery.com/grid-search-hyperparameters-deep-learning-models-python-keras/>

A list of skills & requirements for (Senior) Data Engineers at Idealo:

* A deep understanding of distributed computing frameworks such as Spark (particularly SparkML, SparkSQL, tune/optimize and debug Spark jobs), Hadoop and/or Flink
* Experience with big data at AWS, in particular using EMR and S3
* Experience with Docker and container orchestration like Kubernetes, Swarm or similar
* Experience with pipeline management tools like Airflow, Luigi or NiFi
* Experience with programming languages such as Python, Go and/or Scala
* Good knowledge of SQL/RDBMS
* Experience with the command line, shell scripting and version control (Git)
* Excellent communication skills in English, both oral and written; German is nice to have
* Preferably experience with automatic configuration management like Terraform and Puppet
* Preferably experience with modern agile software development practices like microservices, test-driven development, pair programming, CI/CD etc