Course reader Basic Machine Learning for Bioinformatics 2025

Figure 1. Pretentious Machine Learning images on the covers of course readers, who can do without them? Taken from: https://www.synaptica.com/machine-and-deep-learning/

Table of Contents

Introduction

This course focusses on the very basics of Machine Learning (ML). This means that we'll implement linear regression (yes, that's ML), logistic regression (which, confusingly, is a classification method), a simple neural network, K-means clustering and, and Principal Component Analysis (PCA) ourselves using only basic functions and packages like Numpy and Matplotlib for plotting. Along the way we look at overfitting and underfitting, look at how we measure model performance, and other things to keep in mind.

Machine Learning algorithms are implemented using Linear Algebra computations. Linear Algebra is a whole domain of mathematics with many interesting applications. If you are a studying BIBC, you have already come across its use in 2D and/or multi-D systems of ODEs. A proper course about Linear Algebra can take weeks, and we don't have that time. Therefore, for our purposes, Linear Algebra is just a nice way of manipulating numbers (such as multiplying the weights for each feature in a multivariate linear regression with the feature values for all training examples) in a form that's efficient for computation. The main ideas from Linear Algebra that are relevant for us will be introduced in a small crash course during the lectures, supplemented with suggestions for videos and other resources for further study.

We focus on the basics because it is important that you get a feel for what's really going on. There is an aura of magic around ML, and the excitement around AI has reached a fever pitch. True, advances like AlphaFold 2, ChatGPT, MusicLM and some of Google's and Facebook's convolutional neural nets are true marvels of engineering and a sight to behold. Yet, at it's core, we're just shuffling numbers around. That's why we work on implementing things ourselves: to get a feel for what is really happening.

It is likely that most of what you'll be doing during your studies and later on is applied ML: taking a problem from your domain of study, using an ML algorithm on it, and thereby solving said problem. Finding clusters (subtypes) in single cell cancer sequencing data, training a classifier to predict these different types of cancers based on biopsy sequencing data, that sort of thing. If you're going to train a neural network, you won't implement it yourself, you'll use Keras and TensorFlow, or PyTorch. If you're making a Support Vector Machine, Random Forest, Linear Regressor, using PCA or a clustering of any kind, you're going to use Scikit-learn (or R equivalents). There are free, open-source packages for almost every popular ML algorithm, and you'd be foolish not to use them. For this reason, in the second week, we give a (very short) bird's eye view of Scikit-learn and Keras, so you'll have a basic familiarity and know enough to dive deeper yourself after the course.

The rest of the second week consists of a two-day ML project, where you're going to work in groups to build the best classifier for a biological dataset using Scikit-learn, which culminates in an annotated Jupyter notebook that you hand in. Finally, on the 15th of March from 13:30-16:30, there's a pen and paper exam in Ruppert Blauw that will probe your knowledge about what you've done.

Course structure

General

The first 3.5 days of week 1 cover supervised learning methods. The last 1.5 days of week 1 cover unsupervised learning methods. Week 2 switches from implementing the methods ourselves to using modern established ML libraries (Scikit-learn, Keras) that you'll use for your applied ML projects. There's a ~two-day ML project where you work in teams to get the best classification performance on a certain biological dataset and hand in your work afterwards. The pen and paper exam is on the 13th of March from 13:30-16:30.

Daily structure

We start at 9:00. Lectures and short practicals applying the lecture concepts are interleaved in the morning session. This means 45 minutes of lecture, followed by 60 minutes of direct application in programming exercises, another lecture from 10:45 to 11:30. Followed by practical from 11:30 to 12:30. We have lunch from ~12:30-13:15. The afternoon lecture is from 13:15-14:00 with the rest of the afternoon for the afternoon practical. Note: 60 minutes might not be enough for the 'short' practicals. Just continue in sequence. This schedule is approximate: I will move things around as needed. You are free to miss parts of the course: the lecture slides and practical exercise answers are available for self-study to catch up.

Topics Week 1

- Day 1: Linear regression, gradient descent, introduction to linear algebra
- Day 2: Logistic regression, regularisation, ROC curve, introduction to neural networks (NNs)
- Day 3: Neural network backpropagation algorithm, convolutional neural networks explained.
- Day 4: Continue with backpropagation and neural networks, K-means clustering + hierarchical clustering.
- Day 5: Problems with high-dimensional data, Principal Component Analysis (PCA)

Topics Week 2

- Day 1: Working with scikit-learn, introduction to Keras, project introduction and start
- Day 2: Working on the project
- Day 3: Working on the project. Hand-in of project at 17:00.
- Day 4: Free for studying. Q&A session at 15:00.
- Day 5: Pen-and-paper exam from 13:30-16:30.

Before the course starts

Software requirements

Make sure you have a working installation of Python 3 (preferably the newest version which works with all the libraries, which is probably python 3.9 at the time of writing), and have installed numpy, matplotlib, seaborn, scikit-learn, ipywidgets, Jupyter notebook, PyTorch, Keras, and optionally Biopython and pandas-plink in your working environment. Rather than manually setting this up, please download Anaconda. It comes with many of the most-used packages pre-installed, a graphical user interface to manage packages, and many other tools. You might want to make an environment for the course specifically (**recommended)**. Here are the steps:

1. Install Anaconda (download here). ***Note: it is highly likely that you already have Anaconda!***

2. Update Anaconda. In the terminal (Anaconda prompt on Windows), type `conda update -n base conda`

3. In the (Anaconda) terminal, type: `conda create -n BMLB2025 -c conda-forge python=3.9 numpy=1.26 matplotlib seaborn scikit-learn jupyter ipywidgets`

4. In the (Anaconda) terminal, type `conda activate BMLB2025`

5. In the (Anaconda) terminal type `pip install --upgrade keras`

6. In the (Anaconda) terminal type `pip3 install torch torchvision torchaudio`

7. **Optional** Install Biopython and pandas plink. In the (Anaconda) terminal type: `conda install -c conda-forge biopython pandas-plink`

Environments allow you to keep projects separate: rather than installing packages globally, you install them only for a specific environment, in this case the environment you make for this course. See here if you want to do that. **Biopython and pandas-plink are only necessary for optional parts of the practicals for now, so not absolutely required**.

Downloading practical materials and slides

All practical material, slides, and this reader are on GitHub here. Clone the git repository locally such that you can work with the files. If you do not know how to do this, do the following:

1. If on Windows, install Git for Windows; if on Mac, install Git for Mac; if on Linux (Ubuntu distro), run sudo apt update, followed by sudo apt-get install git. Follow all instructions for your respective case.

2. Go to the folder where you want the files to be downloaded. In Windows, you can open Git Bash there by right clicking (Figure 2). This opens a console. In all cases, you can type git clone, followed by https://github.com/DieStok/BMLB2025.git in the console in the directory where you want the files. This will automatically download everything there.

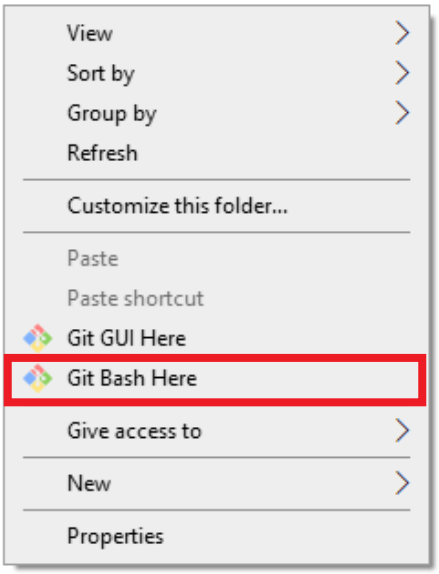

Figure 2. Opening Git Bash in a folder by right clicking in Windows.

3. Alternatively, if this doesn't work, you can download a .zip of all files manually, by clicking on the area indicated by the red rectangle in Figure 3.

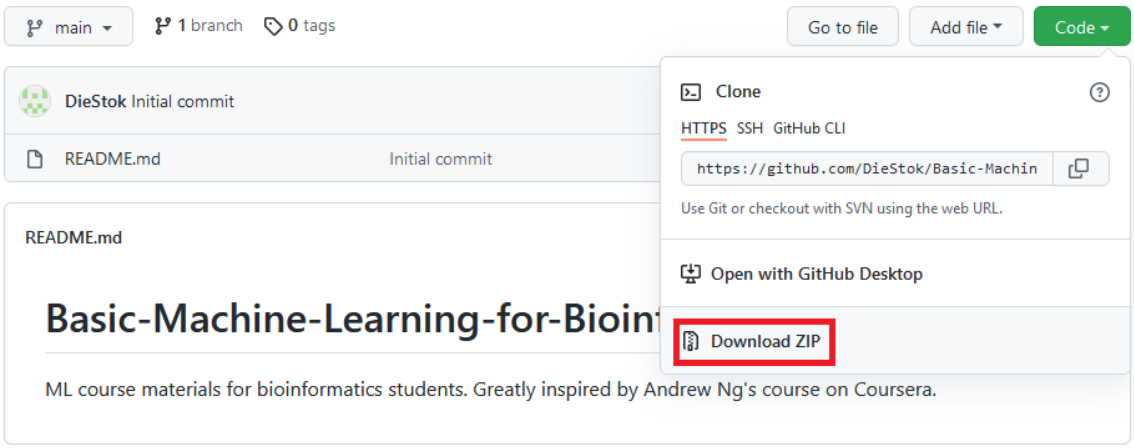

Figure 3. Manually downloading the files from the GitHub page at https://github.com/DieStok/BMLB2025

Materials will be updated as the course progresses, so use git pull every day to update.

Doing the practicals

Normal method

The practicals all take place in Jupyter Notebooks, which are similar to R notebooks or R markdown files you might know. Read more on Jupyter Notebooks here. All you need to do to access the practicals is:

1. Open an Anaconda Prompt (search for Anaconda Prompt in your start menu)

2. Activate the environment for the course, if using. If you followed the installation instructions, this means: `conda activate BMLB2025`

3. Navigate to where you cloned the GitHub repository. For me, that's "J:/Teaching/BMLB2025". To change drive in Windows, simply type the drive letter with a colon (*J:* in my example). Then use *cd* Teaching/BMLB2025.

4. Finally, type *jupyter notebook* in the anaconda prompt. A web browser should open up, where you can navigate to the practical you need. Take care not to close the prompt in which you are running jupyter notebook.

If you didn't manage to install Anaconda

First come to me and let us troubleshoot the issues together. Perhaps there are quick fixes. The following is a last resort. If we cannot get things working on your local machine, do the following:

1. Make sure you have a Google account
2. Go to https://colab.research.google.com/
3. Open a new notebook (File>new notebook)
4. In that new notebook, write in your code cell:

```
from google.colab import drive
drive.mount('/content/drive')
```

and then run it (ctrl+enter with the code cell selected). This will ask you to authenticate that this notebook can access your drive. Allow this.

5. Insert a new code cell below (shortcut keys: esc+b). In it, write:

```
import os
os.chdir("/content/drive/MyDrive/")
```

run this code cell too.

6. Insert a new code cell below. There, clone the course Github. You do that with the following command: `!git clone https://github.com/DieStok/BMLB2025.git`

7. You can't open the notebooks you download there directly from Colab, unfortunately. You need to go to your google drive in the browser, locate the appropriate notebook there, and then right click and open with Colab. See Figure 4.

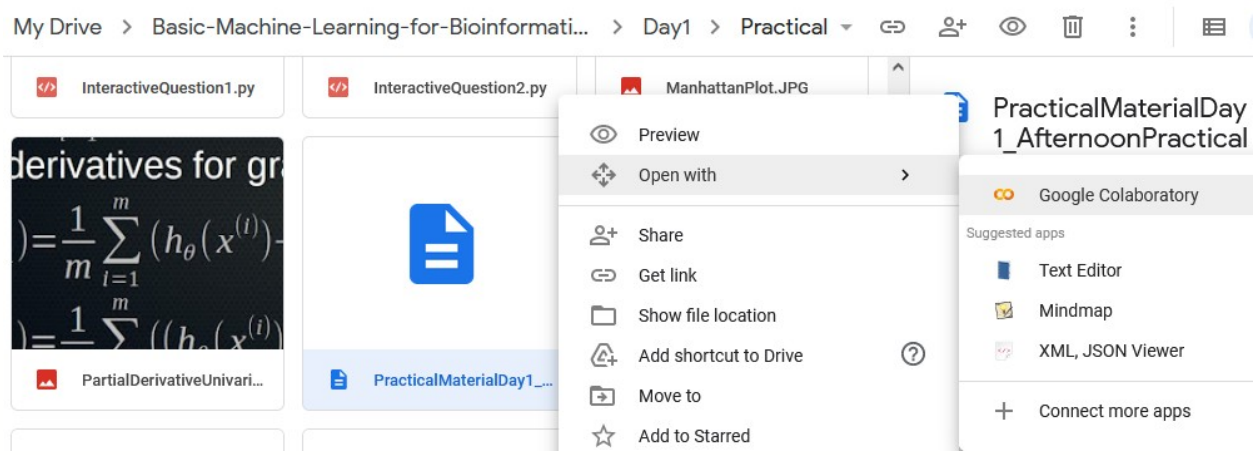

Figure 4. Opening a Jupyter Notebook from Google Drive. Navigate to your Drive in the browser, right-click, and select open with Colab.

8. **Before you can run the notebook you need to do four things**:
   -If it says somewhere in the notebook *%matplotlib notebook*, change this to *%matplotlib inline*. Otherwise plotting won't show anything.
   -Again mount your google drive:

```
from google.colab import drive
drive.mount('/content/drive')
```

-insert the same os code as above, but now navigating to the correct practical directory, say for practical one:

```
import os
os.chdir("/content/drive/MyDrive/BMLB2025/Day1/Practical")
```

–Insert calls to install any dependencies that are not yet there. These use the package manager pip. So for when we are using pandas-plink:

```
!pip install pandas-plink
```

You can also use:

`!pip list` to see what's installed. You put these in a code cell and execute. You will know soon enough when you are missing a dependency, because you will see errors.

Day 1 materials

Description

Today we focus on the most basic supervised ML algorithm: linear regression. We talk about the equations for linear regression, its cost function, using gradient descent to minimise the cost function, going from univariate linear regression (one feature, x, one predicted quantity based on it, y) to multivariate linear regression (predicting, say, disease severity based on levels of 10 biomarkers in the blood). We also discuss the bias-variance trade-off and using cross-validation for measuring ML performance. Finally, you are introduced to linear algebra for programming ML algorithms efficiently.

Recommended videos for review:

Video 1; Video 2; Video 3; Video 4 (~55 minutes)

Extra resources

Videos

- Linear Regression short StatQuest: https://www.youtube.com/watch?v=PaFPbb66DxQ
- More in-depth StatQuest Linear Regression: https://www.youtube.com/watch?v=nk2CQITm_eo
- Cross-validation: https://www.youtube.com/watch?v=fSytzGwwBVw
- Hyperparameters: https://www.youtube.com/watch?v=VTE2KlfoO3Q
- Bias and variance (Video 2 above): https://www.youtube.com/watch?v=EuBBz3bI-aA
- Linear algebra: https://www.youtube.com/watch?v=fNk_zzaMoSs&list=PLZHQObOWTQDPD3MizzM2xVFitgF8hE_ab (3Blue1Brown playlist)

Reading

- Cross-validation: https://machinelearningmastery.com/k-fold-cross-validation/
- Hyperparameters: https://machinelearningmastery.com/difference-between-a-parameter-and-a-hyperparameter/
- In-depth treatment of bias-variance trade-off: http://scott.fortmann-roe.com/docs/BiasVariance.html
- Linear algebra: https://cbmm.mit.edu/sites/default/files/documents/algebra.pdf (slide deck with a lot of depth) ; https://minireference.com/static/tutorials/linear_algebra_in_4_pages.pdf
- Math symbols primer: https://amitness.com/2019/08/math-for-programmers/
- L2-norm: https://kawahara.ca/what-does-the-l2-or-euclidean-norm-mean/

- Jupyter (Notebook) tutorial: https://www.tutorialspoint.com/jupyter/jupyter_notebook_introduction.htm

Other

- Bias-variance trade-off and overfitting discussed with interactive examples: https://machinelearningcompass.com/model_optimization/bias_and_variance/
- Jupyter Guide to Linear Algebra: https://bvanderlei.github.io/jupyter-guide-to-linear-algebra/Matrix_Algebra.html
- Linear algebra course Khan Academy: https://www.khanacademy.org/math/linear-algebra

Day 2 materials

Description

Today we focus on a simple ML algorithm for classification: predicting discrete classes from data (as opposed to continuous variables, which we covered yesterday). This is logistic regression, which you've probably heard of before. It will turn out that the implementation is extremely similar to that of linear regression, just with a different cost function. After this, we will look into regularisation to prevent overfitting, and get started on the basics of neural networks.

Recommended videos for review:

Video 1; Video 2; Video 3; Video 4 (~60 minutes)

Extra resources

Videos

- Regularisation: https://www.youtube.com/watch?v=KvtGD37Rm5I ; https://www.youtube.com/watch?v=Q81RR3yKn30
- Neural networks: https://www.youtube.com/watch?v=HGwBXDKFk9I&list=PLblh5JKOoLUIxGDQs4LFFD--41Vzf-ME1 (StatQuest playlist)
- (AUC) ROC: https://www.youtube.com/watch?v=4jRBRDbJemM
- Nested cross-validation: https://www.youtube.com/watch?v=LpOsxBeggM0; https://www.youtube.com/watch?v=DuDtXtKNpZs

Reading

- Logistic regression: https://towardsdatascience.com/logistic-regression-detailed-overview-46c4da4303bc
- Regularisation: https://explained.ai/regularization/index.html (highly recommended for a *deep* understanding)
- Neural networks: http://neuralnetworksanddeeplearning.com/
- (AUC) ROC and (AUC) PRC: https://glassboxmedicine.com/2019/02/23/measuring-performance-auc-auroc/ ; https://glassboxmedicine.com/2019/03/02/measuring-performance-auprc/ ; https://machinelearningmastery.com/roc-curves-and-precision-recall-curves-for-classification-in-python/

Other

Day 3 materials

Description

Today we focus on neural networks in a bit more depth. Specifically, we take a look at backpropagation and how it allows us to optimise the weights and biases of a neural network. You'll be implementing backpropagation yourself and training a simple network. However, simple (or dense) neural networks are not what have made the huge strides in image recognition (and some other areas) in the last ~10 years possible. For that, we need to look to convolutional neural networks. We'll discuss what convolution is, and the core ideas that make these networks tick. You'll see how you can train a simple convolutional neural net quickly and easily in Keras.

Recommended videos for review:

Video 1; Video 2; Video 3 (~50 minutes)

Extra resources

Videos

- Whole Neural Network playlist StatQuest: https://www.youtube.com/watch?v=CqOfi41LfDw&list=PLblh5JKOoLUIxGDQs4LFFD--41Vzf-ME1
- Backpropagation: https://www.youtube.com/watch?v=tIeHLnjs5U8 ; https://www.youtube.com/watch?v=8d6jf7s6_Qs&t=0s ; https://www.youtube.com/watch?v=GlcnxUlrtek&list=PLiaHhY2iBX9hdHaRr6b7XevZtgZRa1PoU&index=4
- Numerical gradient checking: https://www.youtube.com/watch?v=pHMzNW8Agq4&list=PLiaHhY2iBX9hdHaRr6b7XevZtgZRa1PoU&index=5
- Batch normalisation: https://youtu.be/DtEq44FTPM4?t=127
- Dropout: https://www.youtube.com/watch?v=ARq74QuavAo ; https://www.youtube.com/watch?v=vAVOY8frLlQ (quite in-depth)

Reading

- Backpropagation: http://neuralnetworksanddeeplearning.com/chap2.html ; https://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example/
- Neural networks in general: http://neuralnetworksanddeeplearning.com
- Great 4-part series of Medium articles on neural networks and the (linear algebra) math behind optimising them: 1, 2, 3, 4.
- SENet architecture (complicated!): https://github.com/hujie-frank/SENet
- Understanding convolutions intuitively: https://towardsdatascience.com/intuitively-understanding-convolutions-for-deep-learning-1f6f42faee1
- Batch normalisation: https://tungmphung.com/batch-normalization-and-why-it-works/

- Dropout: https://towardsdatascience.com/simplified-math-behind-dropout-in-deep-learning-6d50f3f47275
- Learning logical functions with neurons illustrated (until halfway through subheading 4): Solving XOR with a single Perceptron | by Lucas Araújo | Medium

Other

- Interactive TensorFlow Neural Network: http://playground.tensorflow.org/
- **Highly recommended** backpropagation slide deck: http://cs231n.stanford.edu/slides/2019/cs231n_2019_lecture04.pdf (especially slide 42-55).

Day 4 materials

Description

Today we first continue implementing our own neural network. Then, we change gears completely and switch from supervised to unsupervised learning. In unsupervised learning, there are no labels, no known training data on which to train our algorithms. Instead, we want to find some structure in the data, without knowing what we are looking for: clustering. This means we must contend with the fact that there is no 'correct' clustering, and that depending on your exact criterion for clustering, there may be many possible clusters to make. We'll look at prototype clustering (K-means as an example) and hierarchical clustering.

Recommended videos for review:

Video 1; Video 2; Video 3; Video 4 (~45 minutes)

Extra resources

Videos

- Clustering algorithms overview: https://www.youtube.com/watch?v=Se28XHI2_xE
- K-means maths: https://www.youtube.com/watch?v=Z2VXrwMYHFk ; https://www.youtube.com/watch?v=0MQEt10e4NM
- Lecture series hierarchical clustering: https://www.youtube.com/watch?v=GVz6Y8r5AkY&list=PLBv09BD7ez_7qIbBhyQDr-LAKWUeycZtx

Reading

- Clustering algorithms overview: https://towardsdatascience.com/the-5-clustering-algorithms-data-scientists-need-to-know-a36d136ef68
- K-means clustering: https://towardsdatascience.com/k-means-clustering-algorithm-applications-evaluation-methods-and-drawbacks-aa03e644b48a
- Hierarchical clustering: https://www.tutorialspoint.com/machine_learning_with_python/machine_learning_with_python_clustering_algorithms_hierarchical.htm

Other

Day 5 materials

Description

Today we continue on with unsupervised learning, now focussing on dimensionality reduction and its flagship/most well-known method: Principal Component Analysis. We first take a look at the curse of dimensionality: how more dimensions cause problems by exponentially increasing the need for training data, inducing overfitting, and making it impossible to use distance metrics. Luckily, we have intelligent ways to reduce the dimensionality of data. Using StatQuest's excellent basis, we'll dive a bit deeper into performing PCA ourselves, and the mathematics behind it. Finally, with knowledge of PCA and multivariate linear regression in hand, we know enough to perform our very own GWAS, in principle. If we get that far, we will perform one in the afternoon practical (Experience teaches me that it's a big if, but no matter. It's cool that you now know enough to do one in principle!).

Recommended videos for review:

Video 1 ; Video 2 (28 minutes)

Extra resources

Videos

- Counter-intuitive phenomena in high dimensions: https://www.youtube.com/watch?v=dr2sIoD7eeU
- Worked example of PCA calculations: https://www.youtube.com/watch?v=S51bTyIwxFs
- PCA lecture series: https://www.youtube.com/watch?v=IbE0tbjy6JQ&list=PLBv09BD7ez_5_yapAg86Od6JeeypkS4YM
- PCA in GWAS: https://youtu.be/qMkuYrlh7jw?t=370
- t-SNE and UMAP: https://www.youtube.com/watch?v=NEaUSP4YerM ; https://www.youtube.com/watch?v=6BPl81wGGP8
- Eigenvalue and eigenvector calculations: https://www.youtube.com/watch?v=TQvxWaQnrqI

Reading

- Curse of dimensionality: https://www.visiondummy.com/2014/04/curse-dimensionality-affect-classification/
- PCA in GWAS: https://pubmed.ncbi.nlm.nih.gov/16862161/ ; https://stats.stackexchange.com/a/8780
- PCA stackoverflow gems: https://stats.stackexchange.com/questions/217995/what-is-an-intuitive-explanation-for-how-pca-turns-from-a-geometric-problem-wit?noredirect=1&lq=1 (mathy) ; https://stats.stackexchange.com/questions/117695/why-is-the-eigenvector-in-pca-taken-to-be-unit-norm (unit length vector) ;

https://stats.stackexchange.com/questions/2691/making-sense-of-principal-component-analysis-eigenvectors-eigenvalues/140579#140579 (simple explanation from the level you'd tell your grandmother to more of an expert)

- In-depth PCA tutorial with examples: http://www.cs.otago.ac.nz/cosc453/student_tutorials/principal_components.pdf (unfortunately some images are missing but other than that it is a great read!)

Other

- Interactive exploration of the UMAP technique: https://pair-code.github.io/understanding-umap/

Day 6 materials

Description

Today we switch gears again. You've now been duly introduced to the basics of ML: supervised and unsupervised learning, (nested) cross-validation, bias and variance, and implementation of various classifiers yourself. There are great open-source libraries that vastly speed up Machine Learning workflows and do all the heavy lifting for you. We didn't start with those because it is important to know what's really going on, but when applying ML in future projects you'd be foolish to do everything by yourself. Therefore, today we switch resolutely to Scikit-learn and Keras, and also discuss some other best practices. In the afternoon, you will be introduced to the project and start work on that.

Recommended videos for review:

Video 1 ; Video 2 (you can probably watch on 1.5 speed); Video 3 (~5 hours 10 minutes: You don't need to watch all of this! The latter two videos are full courses on scikit-learn and Keras which you can browse as you see fit.)

Extra resources

Videos

- Scikit-learn pipelines tutorial (with code): https://www.youtube.com/watch?v=XvnkUg1yVmk ; https://www.youtube.com/watch?v=9vz0n1cyMbc
- Freecodecamp 2+-hour Keras introduction: https://www.youtube.com/watch?v=qFJeN9V1ZsI
- What a tensor is: https://www.youtube.com/watch?v=f5liqUk0ZTw
- Optimizers (gradient descent, stochastic gradient descent, mini-batch gradient descent, AdaGrad, Adam): https://www.youtube.com/watch?v=mdKjMPmcWjY
- Pandas: https://www.youtube.com/watch?v=vmEHCJofslg

Reading

- Hyperparameter optimisation strategies (GridSearchCV and/or RandomSearchCV): https://machinelearningmastery.com/hyperparameter-optimization-with-random-search-and-grid-search/ ; https://scikit-learn.org/stable/modules/grid_search.html
- One-hot encoding https://machinelearningmastery.com/why-one-hot-encode-data-in-machine-learning/
- Nested cross-validation: https://weina.me/nested-cross-validation/ ; https://scikit-learn.org/stable/auto_examples/model_selection/plot_nested_cross_validation_iris.html#sphx-glr-auto-examples-model-selection-plot-nested-cross-validation-iris-py ; https://stats.stackexchange.com/questions/65128/nested-cross-validation-for-model-selection

- Global average pooling: https://adventuresinmachinelearning.com/global-average-pooling-convolutional-neural-networks/ ; https://paperswithcode.com/method/global-average-pooling
- Keras training loop from scratch: https://keras.io/guides/writing_a_training_loop_from_scratch/
- RMSprop (explanation and Python implementation): https://machinelearningmastery.com/gradient-descent-with-rmsprop-from-scratch/
- Adam (standard optimisation method deep learning): https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/
- Mini-batch gradient descent: https://towardsdatascience.com/batch-mini-batch-stochastic-gradient-descent-7a62ecba642a
- Tuning models with KerasTuner: https://keras.io/guides/keras_tuner/getting_started/
- Gradient descent total overview: https://ruder.io/optimizing-gradient-descent/
- Detailed hyperparameter optimisation guide: https://nanonets.com/blog/hyperparameter-optimization/

Other

- Adam gradient optimizer slide deck: https://moodle2.cs.huji.ac.il/nu15/pluginfile.php/316969/mod_resource/content/1/adam_pres.pdf
- Different optimizers visualised (very cool, though does not translate 1-on-1 to the real high-dimensional case): https://github.com/Jaewan-Yun/optimizer-visualization

Day 7 + 8 materials

Description
These days will be spent on the group project. **Details on the group project will be given at the beginning of week 2.** Grading of the project will be mostly on the basis of how well you implemented the basics: did you correctly do the guided exercises, did you correctly cross-validate, did you correctly normalise and correct missing values in pipelines, etc. A small component will be your creativity and use of advanced methods (if any). To work together on Jupyter notebooks on git without hassle, check out nbdev. Be sure to look at least as professional and pleased with yourself when working on the project as our relative below (Figure 5).

Figure 5: Taken from: http://media.gettyimages.com/photos/male-chimpanzee-in-business-clothes-picture-id184941527

Day 9 materials

Description

Today is for studying for the exam. We have a question hour + short project discussion at 15:00. Questions should be sent in by e-mail before 12:00. **If no questions are sent in, I will communicate this to you and we will only discuss the project you did on the previous two days.**

Day 10 materials

Today's the pen-and-paper exam from 13:30-16:30. After that, you are free! You could spend your time wisely, or you could read Saturday Morning Breakfast Cereal. I recommend the latter.

Extra Resources

Basic

- Simple overview of steps of Machine Learning: https://www.youtube.com/watch?v=nKW8Ndu7Mjw + https://towardsdatascience.com/the-7-steps-of-machine-learning-2877d7e5548e
- Numpy video tutorial: https://www.youtube.com/watch?v=GB9ByFAIAH4
- Pandas video tutorial: https://www.youtube.com/watch?v=vmEHCJofslg
- Explanation of Gini Impurity for Random Forests: https://victorzhou.com/blog/gini-impurity/

Libraries

- eli5 (from explain like I'm five) for making ML models interpretable: https://eli5.readthedocs.io/en/latest/overview.html#
- imbalanced-learn for imbalanced datasets: https://imbalanced-learn.org/stable/

Words of gratitude

I am indebted to Dr. Jeroen de Ridder for expert support in preparing the lectures and practical material. The teachings in the course follow Andrew Ng's Coursera course in much of the material, which I highly recommend for further learning and consolidation of the material presented here. The PCA section is based mostly on the lecture series by Victor Lavrenko. I highly recommend both this lecture series and the other content on his channel for further viewing. StatQuest is invaluable for intuitive and simple explanations of the ideas behind various ML methods, stripped of their mathy mystery. I'd like to thank the countless volunteers who make and maintain open-source libraries like MatPlotlib, Keras, and Scikit-learn and prevent the world from falling apart (relevant XKCD). Finally, I'd like to thank Prof. Dr. Berend Snel and Dr. ir. Bas van Breukelen for giving me the opportunity to make this course (and/or cobble it together based on many different sources!).

Stock photos of a woman threatening a goldfish with a gun

What course reader can finish without them? There's a whole slew of these. Amazing. That's life for ya. No ML algorithm can hope to explain why this is the case. Note: I am against torturing goldfish, but have reason to believe this particular specimen was photoshopped in.