Notes on: ”Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow”

Alexandre De Spiegeleer

November 26, 2020

1 The machine learning landscape

ML is nothing more than fitting a model to the known data!

1.1 Sturcture of a project

1. study the problem
2. Train the ML algo (with lots of data)
3. Solution
4. Check the solution and better understand the problem

→ try to improve model from the observed possible problems

Examples:

- Detection of signals in images using CNNs
- Classifying articles: NLP

1.2 Types of ML

Different categories of ML algorithms:

Properties	**Descrition**	**Examples**
Supervised	Need laballed training data	kNN, Linear Reg., Logistic Reg., SVM, Decision Tree, Random Forest, NN...
Unsupervised	No labelled data, the algorithm puts similar data together	Clustering: k-means, DBSCAN, HCA. Dimensional reduction: PCA, kernel PCA, LLE, t-SNE. Anomaly detection: One-class SVM, Isolation Forest
Semi-Supervised	Both labelled and not labelled data	Deepd belief network, restricted Boltzman machine
Reinforcment Learning	In the learning process give rewards or penalites depending on the action done	
Batch Learning	Trains using ALL data → If needs to retrain, need to train on all the data again	
Online Learning	Train by mini-batches: Train incrementally and can start again from a previous minibatch run	
Instance Based	On new data, check distance to known data and assign same output	
Model-based	Use the data to make predictions / interpolates to new data	

1.2.1 Unsupervised Learning

Feature Extraction Combines related features into a better one (e.g. using PCA)

Anomaly Detection Find anomalies in dataset(e.g. removing some outliers)

Association rule learning Discover relationships in large datasets

1.2.2 semi-supervised Learning

Often much data but only a few are labelled

→ Often combinaison of supervised and unsupervised algo.

1.2.3 Reinforcement Learning

Trains an agent by giving rewards and penalties to obtained the best policy

1.3 Challenges

1.3.1 data

Lack of data ML requires lots of data (mininmum thousands of examples).
Note that models performances increase with number of data.
→ Choice to work on model or gather more data!

Non representative data Data used for training must include similar data to those for predictions. (Poor extrapolation capacity)

Sampling Bias non representative data because the sampling is flawed.
Bias can appear if the proportion of data in different classes are not representative.

Poor data quality if errors, outliers and noise in data

Irrelevant features There should be enough relevant features and not much crap.

1.3.2 Fitting

Overfitting The model fits too well the training data. Occurs because model too complex compared to data → lose predictability.

Underfitting Model too simple compared to the data to be represented e.g. Linear fit on a non-linear problem

Solutions to overfitting:

- *Regularization*
  → Making the model simpler to avoid by adjusting hyperparameter
- *Hyperparameter*
  → Parameters of the learning algorithm that dictates the amount of regularization.

Solutions to underfitting

- Select a better model
- Have better features
- reduce the constraints on the model (e.g. hyperparameters)

1.4 Evaluating performance

Split the data into training set (80%) and testing set (20%). Evaluation on the test set gives an estimate of the error on unseen data. When training multiple models on the training set and testing them on the test sets, our selection of the "best" model is which model best fits the test data set. → Model cannot necessarily be good on new data. Thus, keep a validation set.

Thus, there are 3 sets of data: Training, Validation and Test

1. Train multiple models with different Hyperparameter on the training set
2. Select the model that performs best on the validation set.
3. Verify that the model is indeed good on the test set
4. Deploy

It is common to split the training set to train several models on sub-sets of the training set and train one final model on the whole training set.

If the validation set is too small → imprecise evaluation of performances. The validation set cannot be too large either (compared to training set).
→ use *cross-validation*: it uses several small validation sets. Each model is evaluated once per validation set. You can then average the results of the model on the different smaller validation sets.

1.5 Data mismatch

If there are two types of data in the training set (e.g. not from the same source). This may cause problem in the predictions. How to know if overfitting or problem with the data?

Hold part of the training set (data from one of the source) and see how the model performs on that. If it performs well → the error comes from the mismatched data If it performs poorly → the error comes from the overfitting

If it is because of the data mismatched, is it possible to preprocess these to be more like the rest of the data?

End-to-end machine learning project