Capstone Project

Architectural Decisions Document

by Torsten Bölke

# Architectural Components Overview



IBM Data and Analytics Reference Architecture. Source: IBM Corporation

## Data Source

### Technology Choice

My data is source is the well known Iris-Dataset obtainable for example from the machine learning repository at UCI: <https://archive.ics.uci.edu/ml/datasets/iris>

### Justification

It’s a well described dataset in csv-format and you can easily download it (free of charge).

This famous machine learning dataset perfectly fits the purpose of my capstone project. Choosing this dataset helps to efficiently reach the goal of my project.

## Enterprise Data

### Technology Choice

There was no Enterprise Data used.

### Justification

It just doesn’t fits the purpose of my project to use Enterprise data.

## Streaming analytics

### Technology Choice

There was no streaming used in this project.

### Justification

It just doesn’t fits the purpose of my project to use streamed data..

## Data Integration

### General Choises

#### Choises made

For data integration I downloaded the raw data and processed it using a self-written ETL script.

As environment IBM Watson Studio with Python 3.6 and Spark 2.3 was choosen.

#### Justification

To just download it is the simplest way to get the data. Using an ETL script for data integration is always a good idea, as this is a critical task and should be well documented.

Spark was chosen in order to evaluate how good my coding skills in spark ML are already. The project itself has no special requirements which point to a special framework.

### Methods for cleaning

#### Choises made

For a very first look the following methods were used:

* df.head(5)
* df.info()
* df.describe()
* plot3D = plt.figure().gca(projection='3d')
* plot3D.scatter(df[[1]],df[[2]], df[[3]])
* df.boxplot()
* df.corr()

A descriptive Header was added to the data:

df.columns = ["Feature\_1", "Feature\_2", "Feature\_3", "Feature\_4", "Label"]

#### Justification

Using the methods above, I made sure, that:

* there are no missing (null) values
* data types are matching the content: numerical data is stored as float64 and class labels are stored as object (pandas type for text)
* there are no undesired duplicates
* that ranges within each column makes sense
* min, max, mean, median, std for each column makes sense

### Methods for Feature Engineering

#### Choises made

I constructed a pipeline containing:

* String Indexer
* One Hot Encoder
* Vector Assembler
* Normalizer

Later a PCA for dimension reduction and feature scaling were added to the pipeline.

#### Justification

If you want to use ML methods from the usually used libraries you have to make sure that your data is in an appropriate format which you can provide by using these methods.

The PCA is a good way to create new features which have one or more dimensions less and are in most cases lesser correlated than the original features. Feature scaling was just another try to improve model performance, which is quite common in data science.

## Data Repository

### Technology Choice

A data repository on github was used and syncronized with my local machine.

### Justification

Using github for exactly this purpose is common practice in software development and data science and according to this project there are no reasons to decide for another repository.

## Discovery and Exploration

### Algorithm Choices

#### Choices made

The dataset has 4 numeric features and a label which has 3 distinct entries – so I had to deal with a multiclass classification task. I decided to use these 3 algorithms in this project:

* MultiLayerPerceptron
* DecisionTree
* RandomForest

#### Justification

As we had to choose at least one deep learning algorithm I had to pick the MultilayerPerceptron, as this was the only deep learning algorithm provided by the spark ML library suitable for multilayer classification.

In addition to that I had chosen DecisionTree and RandomForest because I was interested in how a single tree performs against a whole forest.

### Training choices

#### Choices made

First I splat the data into training and test data:

splits = spark\_df.randomSplit([0.85, 0.15])

df\_train = splits[0]

df\_test = splits[1]

Then the training was done in two ways:

* Training on the whole training data the normal way
* Training on the training data using crossvalidation

#### Justification

The main goal of my project is a critical review of the properties and usefulness of crossvalidation – does it really performs as good as many data scientist believe?

So, its quite obvious that I had to use both training methods in order to compare each other.

### Evaluation Choises Round 1

#### Choices made

The evaluation was done by determining the accuracy.

In order to evaluate the performance of crossvalidation the evaluation was done on both:

* Training data
* Test data

#### Justification

As it makes no difference at which class false positives or false negatives occur accuracy is the suitable measure. In other case you may have to weight a false positive more the a false negative or the other way around ( for example in health care situations). In that cases accuracy will not be a suitable measure.

In order to evaluate the results of both training methods we need to know how the differently trained models perform on before unseen data.

### Modell improvement Round 1:

#### Choices made

As first try to improve the model performance I chose hyper parameter training using grid search.

#### Justification

We want to know how good our model could perform, so it’s obviously to tune the hyper parameter first, before we have a look changing the model.

#### Modell improvements Round 2:

#### Choices made

As second try to improve the model performance I gone back to the feature engineering part and included a PCA combined with grid search. Only the 3 most important out of the 4 new features were used as input of the model.

#### Justification

Well, we were told to go back to the feature engineering part, so I did this. Without experience from similar projects its just a try and error game… And just of interest I picked out PCA first. The idea was to remove some noise by reducing the dimensionality to the most important 3 new features, which should on top of that be less correlated to each other than the original features.

### Modell improvement Round 3:

#### Choices made

As third try to improve the model performance I tried out feature scaling combined with grid search.

#### Justification

This was just another try and kind of a bonus to the requirements of this submission. And as said before without very experienced knowledge from many very similar projects this is mostly try and error without a deeper justification. Beside of that, from a mathematical point of view I would be surprised if feature scaling (don’t mistake scaling with normalization!) would have result better results – but on the other hand: deep neural networks are just black boxes. That’s why I just gave it a run…

## Actionable Insights

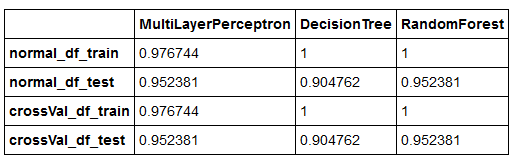
### Actionable Insight 1

#### Recommendation 1

As shown by my investigation, crossvalidation is no fully equivalent replacement for splitting data into train and test set. While it reduces the chances and the extent of overfitting, it is just no adequate replacement of an evaluation on completely unseen data. So you should always tell your stakeholders that the evaluation was based on crossvalidation only (if you really have to use crossvalidation without any other evaluation method because of a very small amount of available data).

#### Justification

As shown in the figure below, the overfitting was the same for all three algorithms for training normally on the whole data set and on training by crossvalidation:



In other words: (1) using crossvalidation I got exactly the same very good results as on (2) training on the whole training set and testing on the training data. While nobody would accept an evaluation based on method (2) only, there are still many people around who believe that a validation based on crossvalidation gives you a realistic measure of modell performance. My findings show that this is at least not always the case…

### Actionable Insight 2

#### Recommendation 2

I clearly recommend this IBM advanced data science course.

#### Justification

By taking this course a really learned a lot, especially the spark parts were totally new for me and still I was able to realize the capstone project using spark ML. And last but not least it was a lot of fun.

## Applications / Data Products

### Technology Choice

As the type of the data product I choose the presentation format.

### Justification

In order to convince people a presentation is never a bad idea. Depending on the complexity of the project further modules like a report document and deployed model could be necessary as well.

## Security, Information Governance and Systems Management

### Technology Choice

I used a PC-System with updated security patches, firewall, and virus scanner behind a hardware firewall. In addition the secure environments of IBM Cloud and GitHub were used.

### Justification

This security technology was already there and as I used open and freely available data and the results have to made public there is simply no need for a more advanced security technology.