

Project report

Python for Data Analysis

Cédric HANSEN & Quentin GABOT
DIA3
2020/2021

Summary

- I. Discussions about the dataset
 - 1. The initial assumptions
 - 2. State of the art
- II. Representation & prediction
 - 1. Understand the dataset via the graphs
 - 2. Preprocessing
 - 3. The training phase
 - 4. The final tests
- III. Computing the API
 - 1. API Flask
 - 2. Results

The initial assumptions

- Our dataset is called the “Avila Bible”
 - A corpus of texts written by 12 copyists
 - The copyists are designated as a letter such as :
 $(A, B, C, D, E, F, G, H, I, W, X, Y)$
- The dataset is described by its columns & rows
 - Rows (or samples) = 4 consecutives sentences from the corpus
 - Columns (or features) = variables that distinguish and allow us to predict the patterns' authors
- The dataset is split between 2 subsets
 - The first one is the “train set” with 10430 samples
 - We will train our models on this dataset
 - The second is the “test set” with 10437 samples
 - We will make our final predictions on it : but which prediction ?



State of the art

- The nature of the prediction
 - ❑ Our objective is to assign each row from the test set to an author
 - ❑ This assignation will be made by a model trained previously on the train set
- The various prediction techniques
 - ❑ Machine Learning is divided into two categories : supervised learning & unsupervised learning
 - ❑ As we possess the output data, we are in the first situation : supervised learning
 - ❑ Supervised learning is then divided into regression & classification
 - ❑ As the response variable is qualitative, we are in the latter situation : classification
- The classification models
 - ❑ The classification is a very wide and rich field in terms of possibilities
 - ❑ For this problem, we have retained some of the most famous and effective models
 - ❑ Logistic Regression, KNN, Random Forest, Support Vectors Machines & Naïve Bayes
 - ❑ We will select the one with the best accuracy i.e., the best ratio $\frac{\text{good prediction}}{\text{total prediction}}$

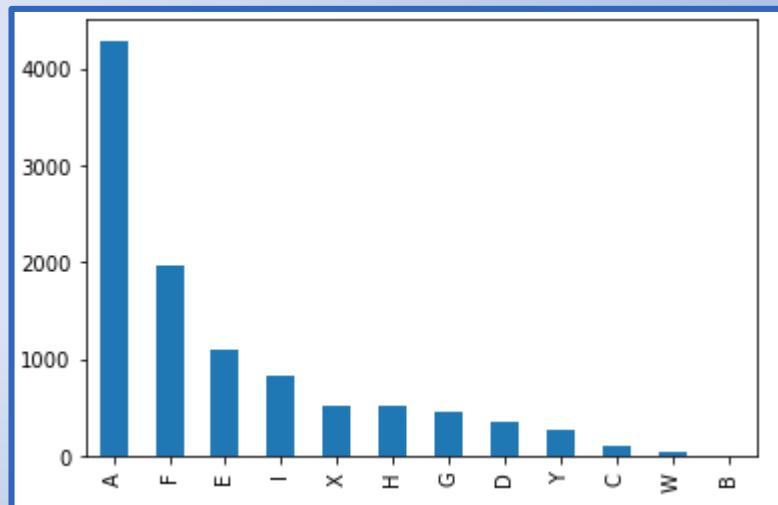
Understand the dataset via the graphs

➤ The data visualization

- ❑ We decide to help our comprehension of the dataset with some graphs
- ❑ We have decided to implement three major graphs : boxplots, heatmap & pairplot

➤ The consequences

- ❑ We have understood that the classes' distribution is unequal
- ❑ Thus, our model will provide way more precision than a random classifier
- ❑ We have discovered that there was little to no relation between variables except for the last one : "modular ratio/interlinear spacing"
- ❑ Indeed, this variable is the combination of two other variables
- ❑ As such, we have decided to drop this variable



Preprocessing

➤ The need for preprocessing

- ❑ If we directly jump into the training phase, we might skip some precious steps in order to deliver the best prediction possible
- ❑ For example, the data visualization has informed us that the variables weren't scaled
- ❑ Thus, if we would have started the training phase without scaling the variables, the prediction would have suffered from this flaw

➤ The preprocessing

- ❑ We have scaled the data like explained previously
- ❑ We have checked for any NAN values in order to treat this problem
- ❑ As there were none, we didn't have to propose a solution

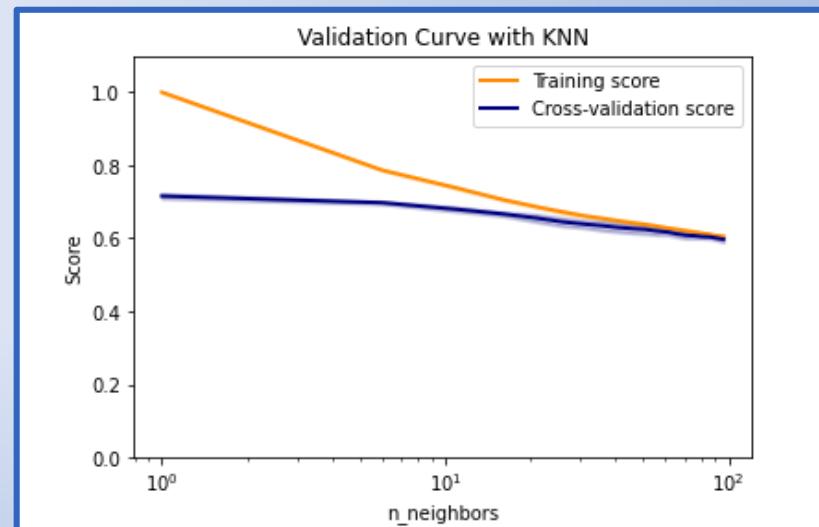
The training phase

➤ The protocol

- ❑ The training phase will be divided into 5 phases corresponding to the 5 models that we will train
- ❑ Each phase we follow the exact same process : we compute the naïve model, we tune the hyperparameters thanks to the validation curve in order to determine which one is interesting to manipulate and we finally use a RandomSearchCV to determine the best values for each hyperparameters

➤ The validation curve

- ❑ The validation curve allows us to visualize a more precise interval for each quantitative hyperparameters in order to save us time and gain in precision before the Grid Search
- ❑ This curve also helps us to avoid any case of overfitting by being vigilant of the difference between the CV score et the Train scoree
- ❑ Sometimes, the validation curve shows us that a variable remains invariant: this is the case for the “leaf_size” du model KNN
- ❑ We then drop this hyperparameter as its default value is already the best one



The final tests

➤ The final test

- ❑ The final test consists of testing every model on the test set
- ❑ As we do so, we pick the best accuracy value representing the most efficient model
- ❑ This model is the Random Forest with an impressive 99,95% accuracy

API Flask

- Setting the API
 - ❑ Create new environment
 - ❑ Library: Flask, Bootstrap, Wtform

- How it works
 - ❑ Creates form where we add each variable
 - ❑ Check the validity of each variable
 - ❑ Use our model on the input variable
 - ❑ Show the prediction

Results

From which copyist are you the closest?

intercolumnar distance :

upper margin :

lower margin :

exploitation :

row number:

modular ratio :

interlinear spacing :

weight :

peak number :

modular ration/interlinear spacing :

First page: Form

With this value, your handwriting is close to the copyist :

['A']

[Try Again With new value](#)

Second page: Result