Avila Data Set Presentation

The Avila data set has been extracted from 800 images of the "Avila Bible", a giant Latin copy of the whole Bible produced during the XII century between Italy and Spain.

Preemptive analysis

The prediction task consists in associating each pattern to one of the 12 copyists.

Data Features + Copyists Target

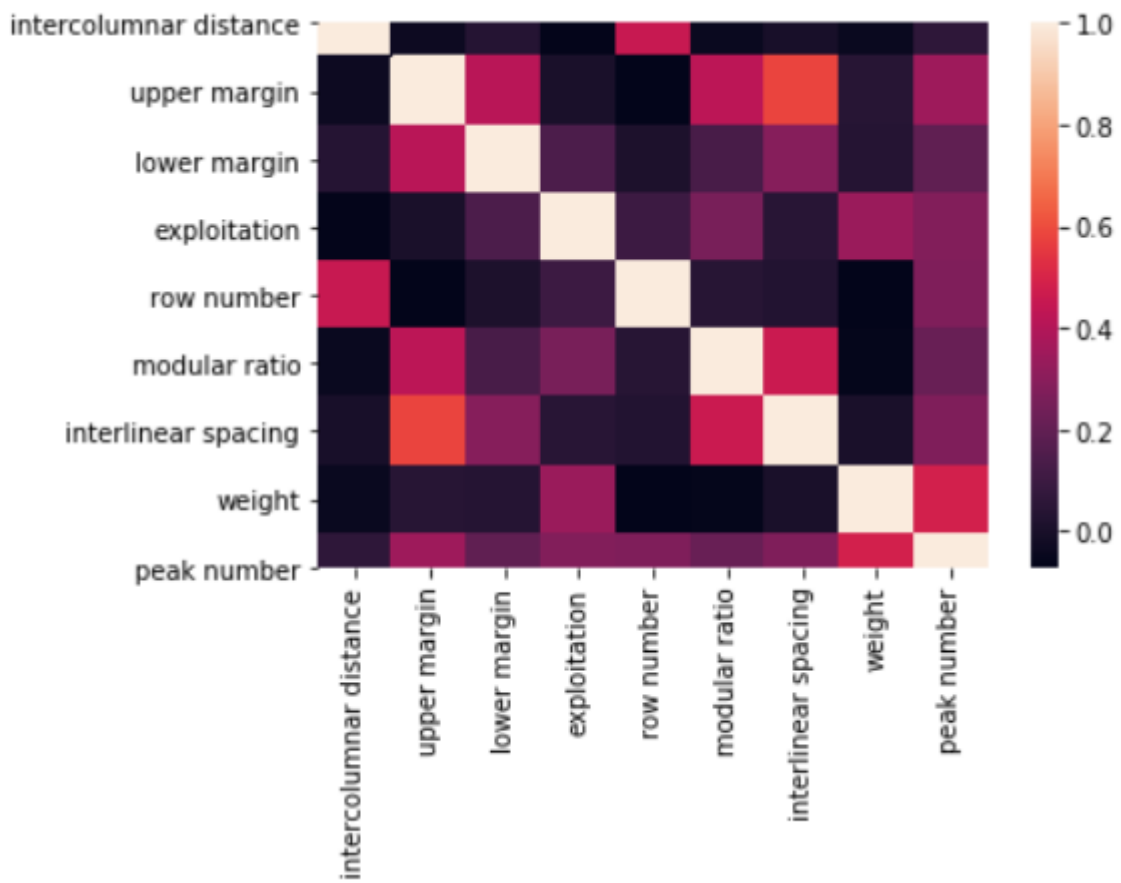
Attributes	Datatypes	Missing values
intercolumnar distance	float64	0
upper margin	float64	0
lower margin	float64	0
exploitation	float64	0
row number	float64	0
modular ratio	float64	0
interlinear spacing	float64	0
weight	float64	0
peak number	float64	0
modular ratio/ interlinear spacing	float64	0
copyists	object	0

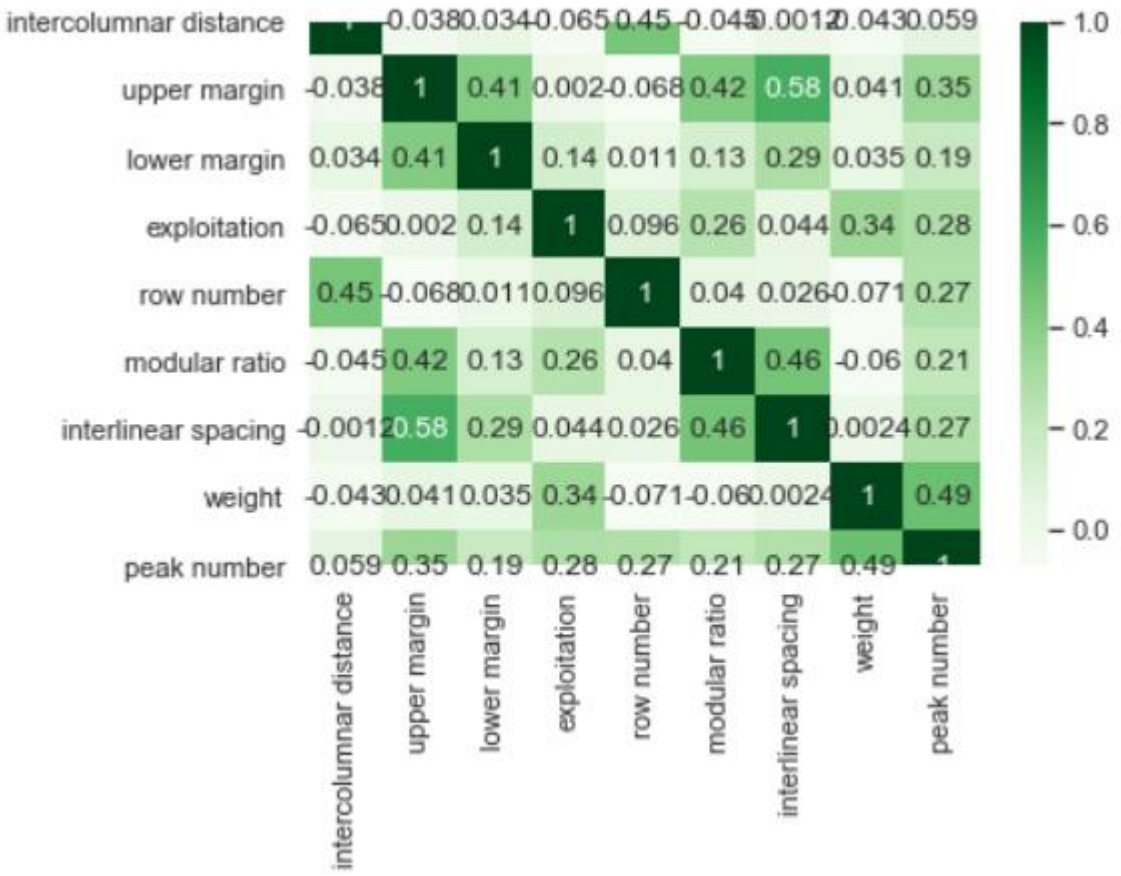
dtype: object

TrainingSet Shape

(10429, 11)

Correlation analysis

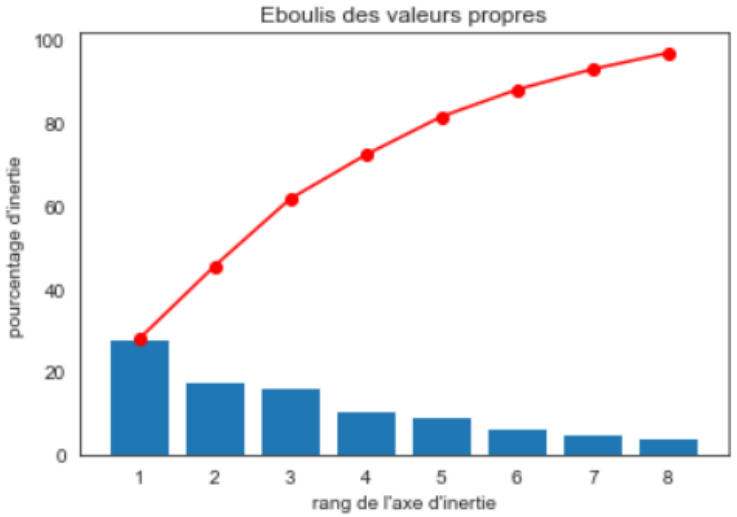


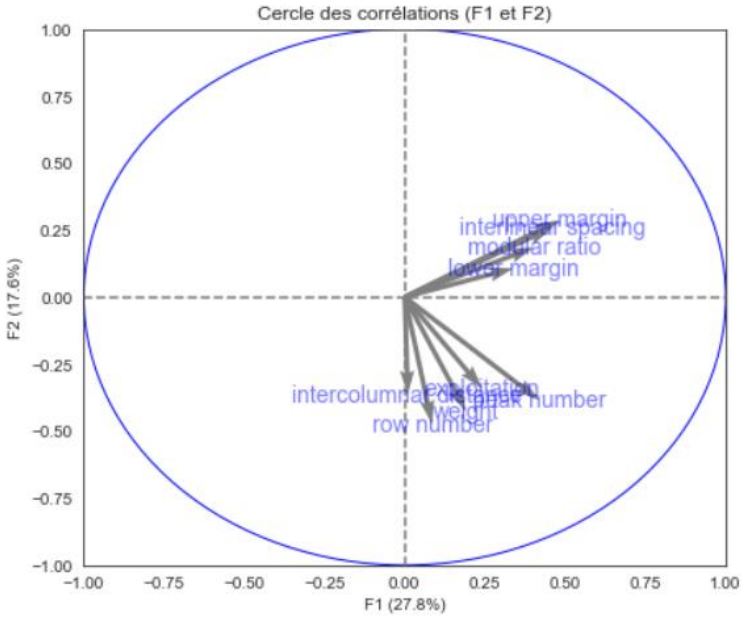


As we can see, there is no correlation higher than 0,49 between variables, all of them are going to be useful in our models

PCA analysis

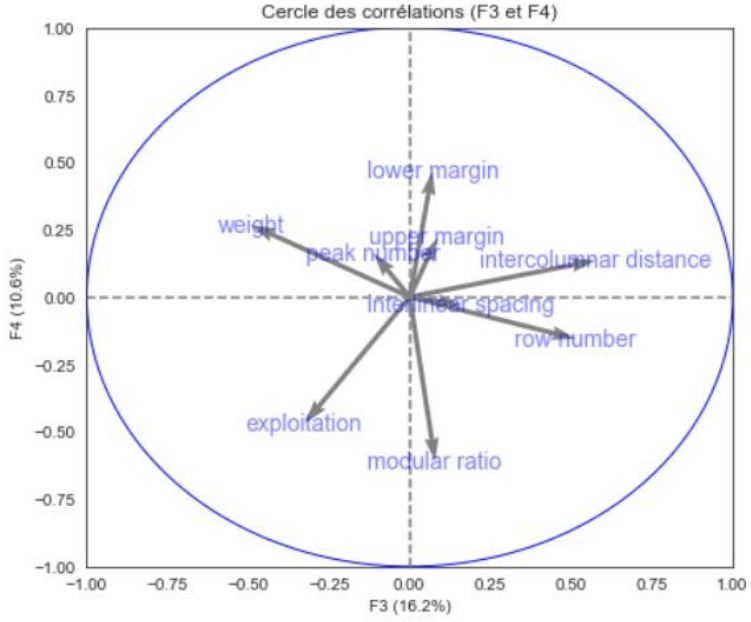
After the PCA analysis, we choose to keep the first 4 eigen values, as they explained most of the data





From the first correlation circle (F1 vs F2) we can deduce that their is 2 pack of anti colinear variables

Lower Margin, Upper Margin, Interlinear spacing, Modular ratio
Vs the others



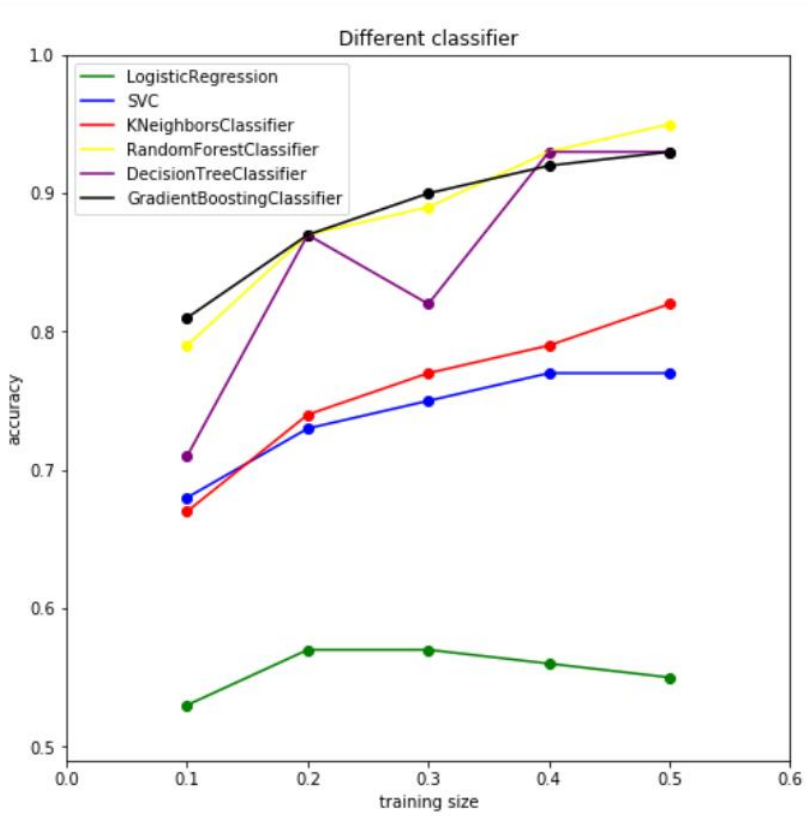
Modelisation

1) We drop of the copyist column
(name of the copyist)

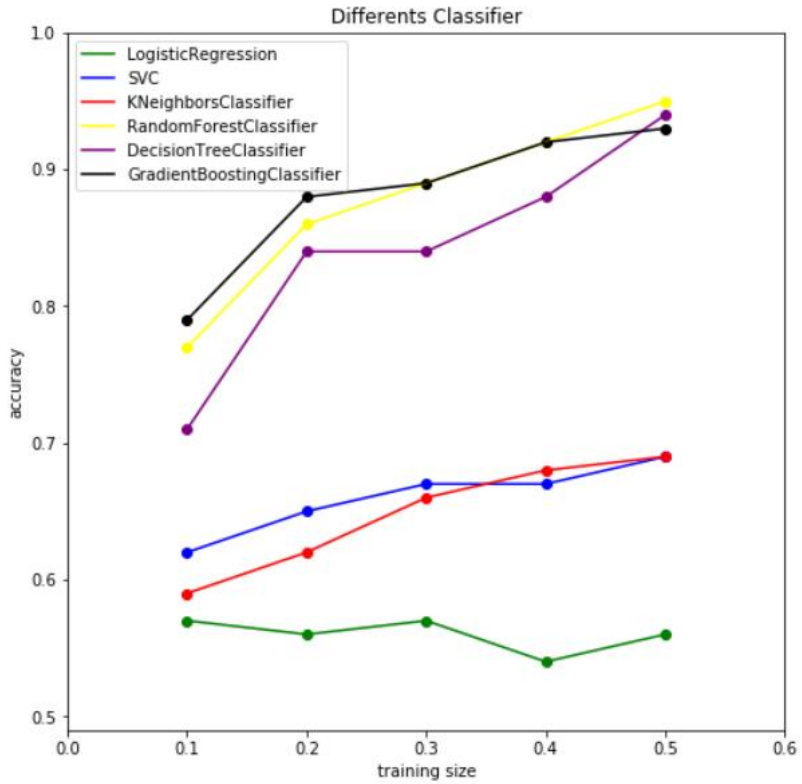
2)We create the different models testing :
- Logistic regression
- SVC
- KNeighborsClassifier
- Random Forest Classifier
- Decision Tree Classifier
- Gradiant Boosting Classifier

3) We split the data and we test the differents models with
differents training size (0.1, 0.2, 0.3, 0.4, 0.5)
- We find the accuracy for each models for each training size

Comparaison of models



scaled



No scaled

We have chosen the Gradient Boosting Classifier because this is the best accuracy for the minimal training size

GradientBoostingClassifier GridSearchCV:

Variation of Hyper parameters → Fitting 5 folds for each of 25 candidates, totalling 125 fits

```
parameters = {
    'n_estimators': [400,450,500,550,600],
    'learning_rate' :[0.1, 0.2, 0.3, 0.4, 0.5]}
gbc = GradientBoostingClassifier()
clf = GridSearchCV(gbc, parameters, cv=2, scoring='accuracy', verbose=5, n_jobs=-1)
clf.fit(X,y)
```

Best score of accuracy → 0.9965480870649152

Best parameters → {'learning_rate': 0.1, 'n_estimators': 600}

Computation time →

```
[Parallel(n_jobs=-1)]: Done 125 out of 125 | elapsed: 19.8min finished
```

Predictions

We call predict on the estimator with the best found parameters
on TestingSet

TestingSet vs Trainingset → result of prediction → `0.9985626676887697`

The result is better than the accuracy with
the Gradiant Boosting Clasifier

API

We try to do an API with the tuto : https://rubikscode.net/2020/02/10/deploying-machine-learning-models-pt-1-flask-and-rest-api/

We got a problem to charge the model with tensorflow, so the API doesn’t load.

We ask to our comrade (who use the same tuto) to help us, but nobody found the solution.