Team ID: 32

Members:

1-عبد الرحمن ايمن فهمي

20191700332

2-تغريد سعيد عبد الفتاح

20191700194

3-سليمان سنوسي سليمان

20191700290

4-ايه هشام احمد السعدي

20191700167

5-علاء احمد عبد التواب

20191700891

Pattern Recognition

Projects

(CS)

Player value prediction

GENERAL NOTES:

- we drop ‘club_join_date ’ and ‘conract_year_date’ and merge it in one column ‘contract_duration’

- we drop ‘body_type’ column and split it into 3 columns ‘ body_type_lean’ and ‘ body_type_normal’ and ‘ body_type_stocky’

- we drop ‘wark_rating’ column and split it into 2 columns ‘wark_attacking’ and ‘wark_defensive’

- Milestone 1 part1 : preprocessing code.
- Milestone 1 part2 : Model code.

Preprocessing Techniques:

1- import libraries (numpy, pandas, matplotlib, IPython)

2- Explorality Data Analysis (read csv file , dataframe describe, data info)

3- Number of Respondents vs. Rows(Check Duplicated Data)

4- Time Stamps (convert date to datetime64 and check range of dates)

5- Handle Inconsistent data (Check up for Nans and change any Inconsistent data)

6- Numeric Check Nan (Replace with mean)

7- Fill Na(fill with mean of the column)

8- Drop useless columns(drop function in dataframe)

9- Feature Encoding(Convert categorical data to numerical)

10- Save data cleaned.

Perform Analysis:

We used built-in function “SelectFromModel”

To retrieve appropriate features (in our case 16 features only)

Some of features related with each other such as

Positions of player with columns (ST, LW, and so on) if player played in his own position he has best power.

We remove some of independency with columns

We used multiple regression and polynomial regression and we have also some models such as we used data as a whole with no selection(Approximately 79 columns) and model with Normalized data and so on .

Polynomial is the best way because it has lowest MSE.

MSE (Polynomial (with degree 3))= 9.691299e+11

Accuracy (Polynomial (with degree 3))=0.9689

Training time(Polynomial (with degree 3))= 13.9889 s

MSE (Multiple) = 1.237889e+13

Accuracy (Multiple) = 0.6032

Training time(Multiple)=0.1515 s

MSE (NO Features Selection) =1.906519e+12

Accuracy (NO Features Selection) =0.9388

Training time(NO Features Selection)=0.3178 s

MSE (Normalized features) =1.237889e+13

Accuracy (Normalized features) =0.6032

Training time(Normalized features)=0.2199 s

MSE (Standardized data)= 1.237889e+13

Accuracy (Standardized data)=0.6032

Training time(Normalized features)=0.2199 s

What features is discarded?

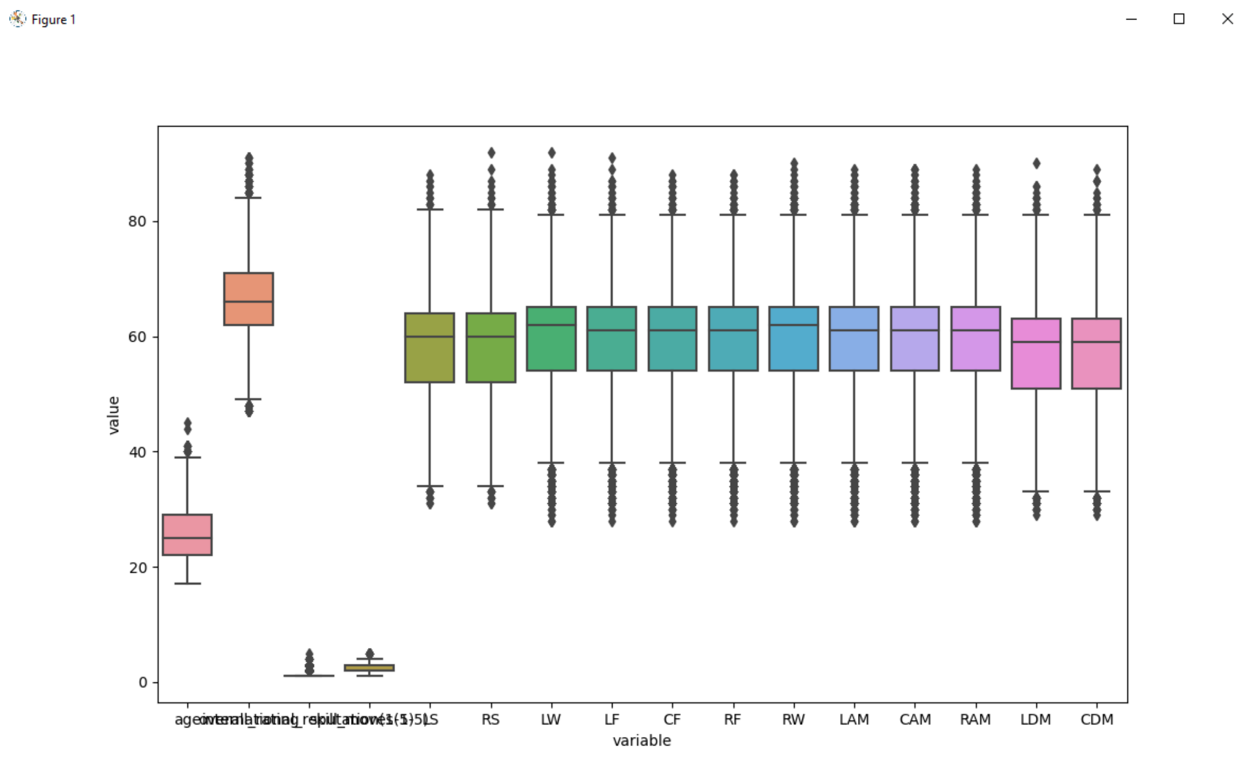
(id,name,full_name,birth_date,Positions,nationality,club_team, club_jersey_number,club_join_date,contract_end_year,national_team,club_position,national_jersey_number,national_team_position, tags, traits)

What the sizes of training, testing sets?

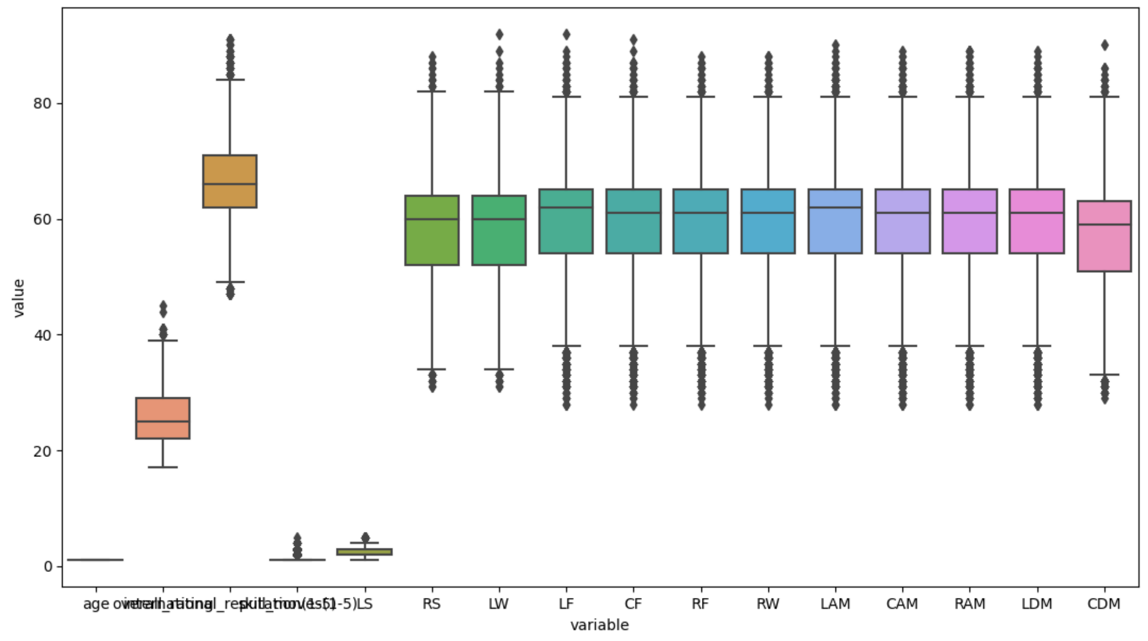
Training set: 80%(11,310 record)

Test set: 20%(2,828 record)

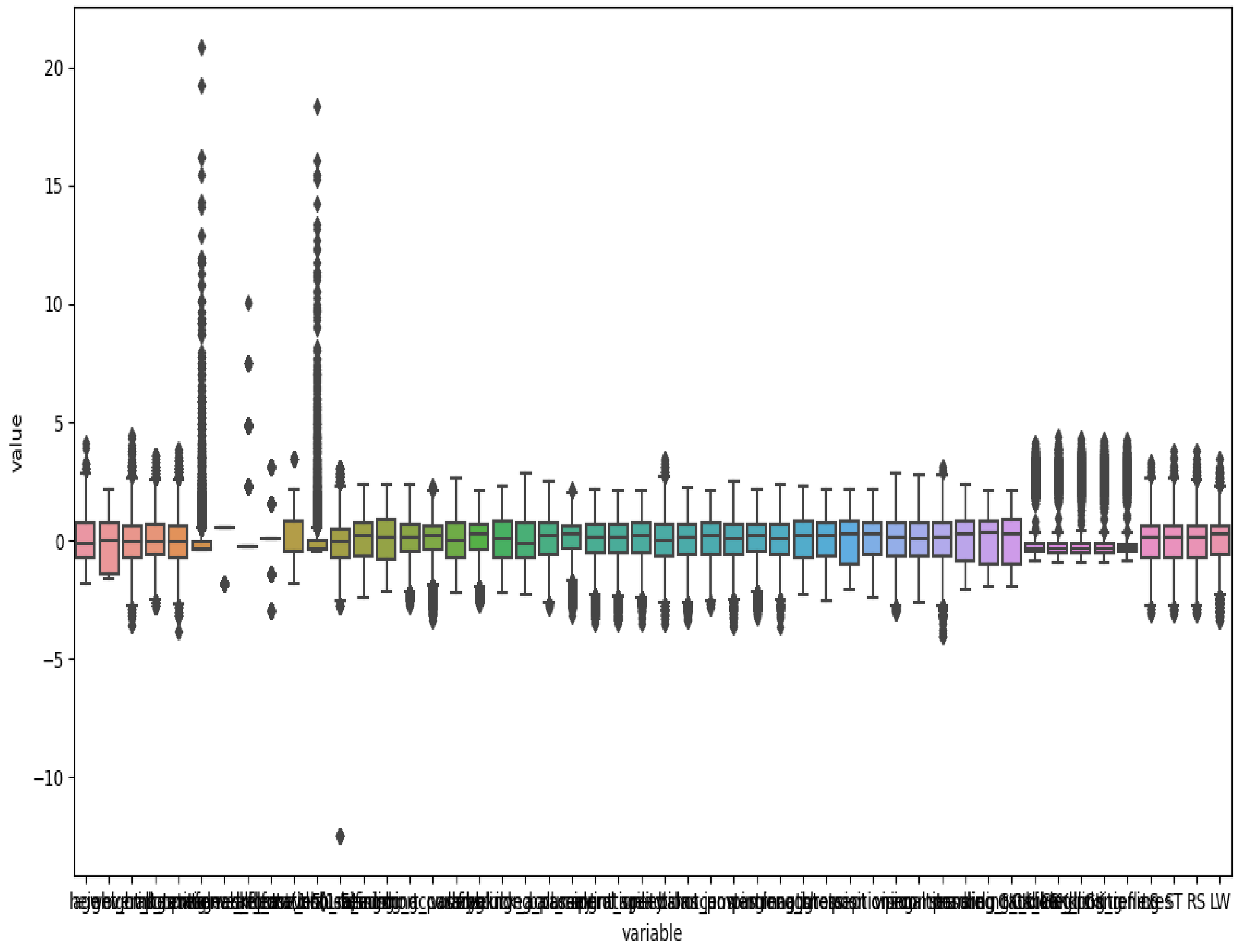
Multiple



Polynomial



No feature selection



Normalized

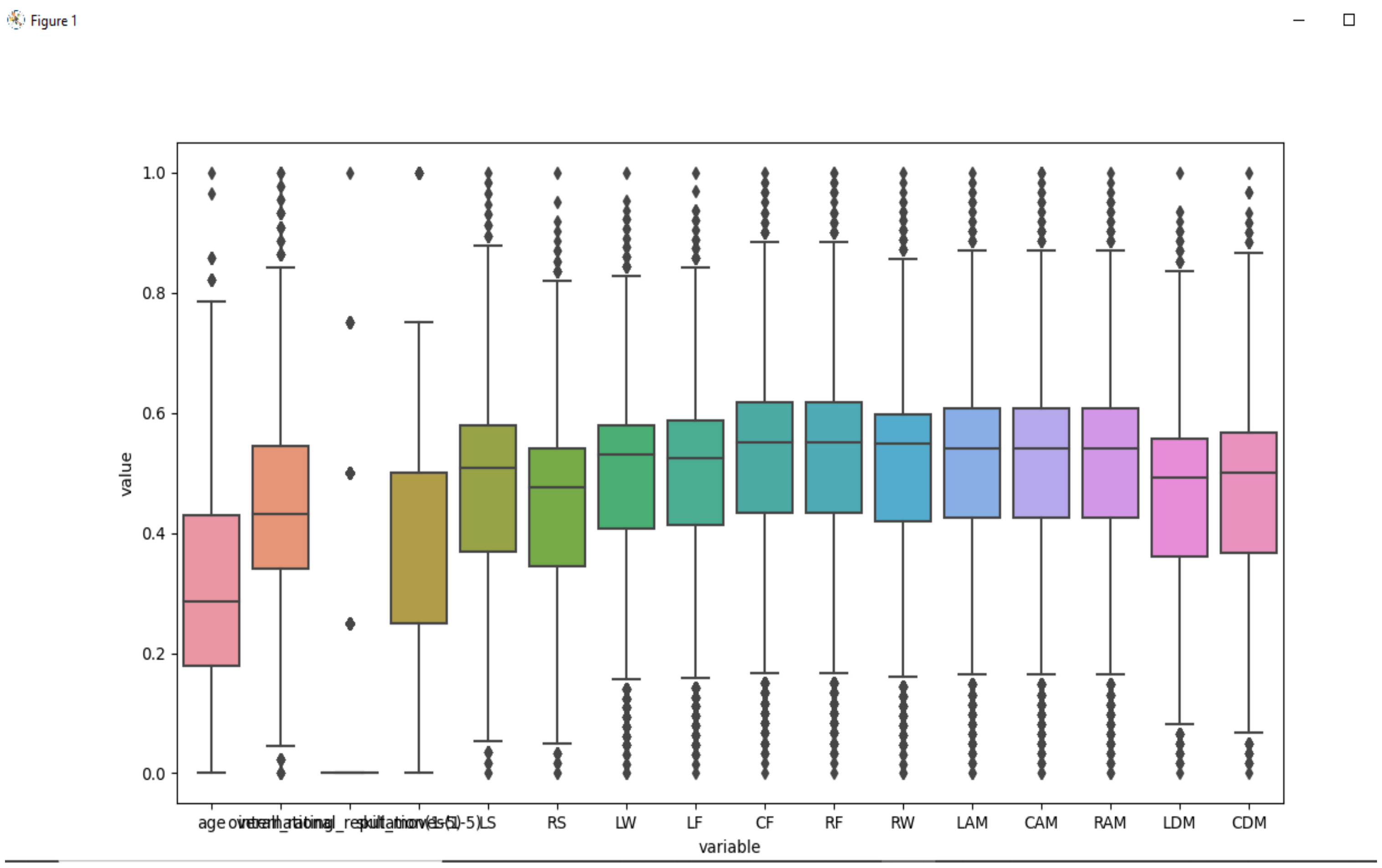
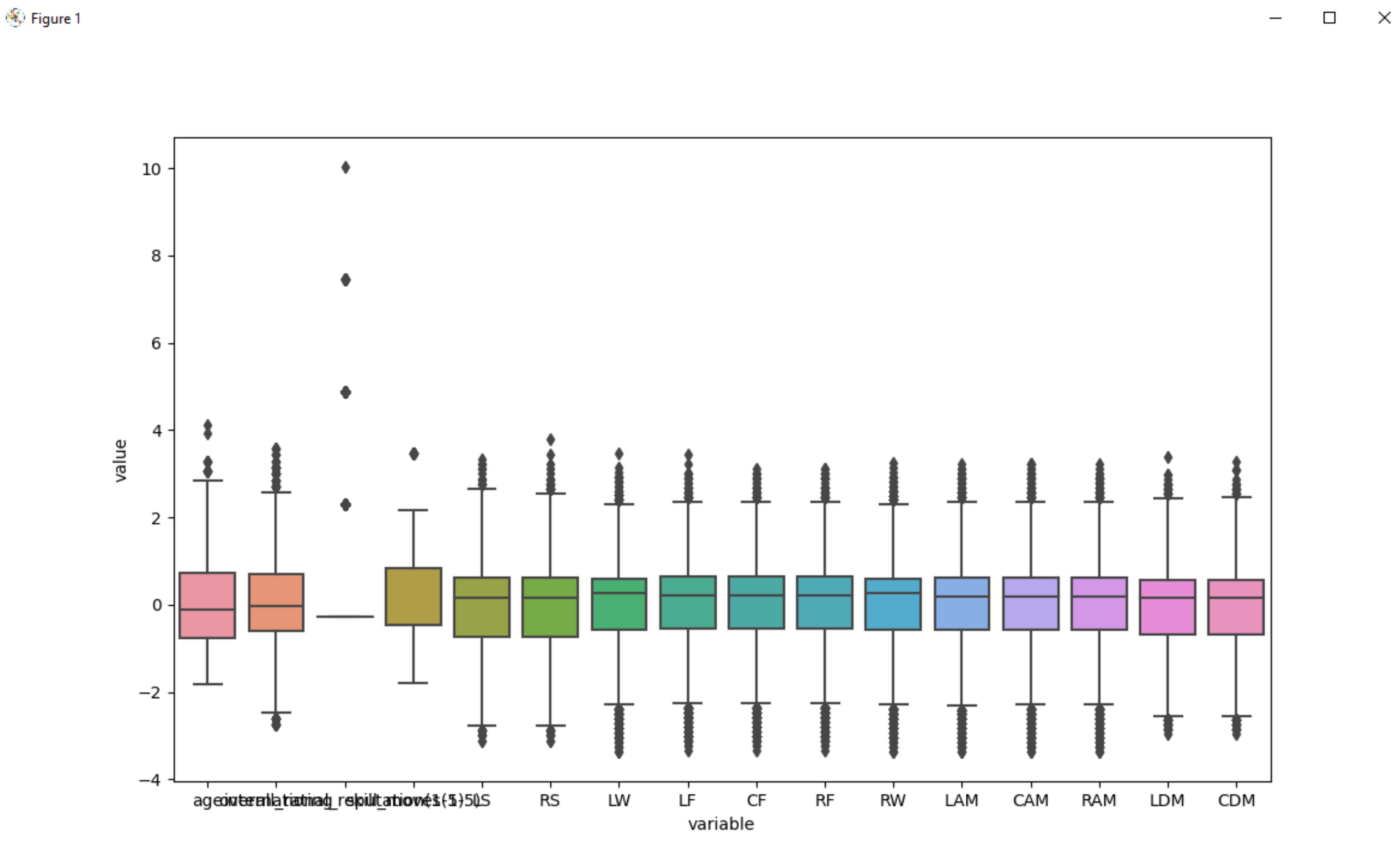


Figure 1
1.0
0.8
0.6
0.4
0.2
0.0
value
age
overall_rating
international_reputation(1-5)
skill_moves(1-5)
LS
RS
LW
LF
CF
RF
RW
LAM
CAM
RAM
LDM
CDM
variable

Standardized



Conclusion

In this phase our team do Pre-Processing in various Forms (feature scaling, feature selection. And so on)

We had some problems, but we got over it, such as handling Nans, We remove Nans and re fill it with mean value of columns technique. We remove some useless columns and substitute some inconsistence data, in the final we moved the work to another file to be ready for Models.

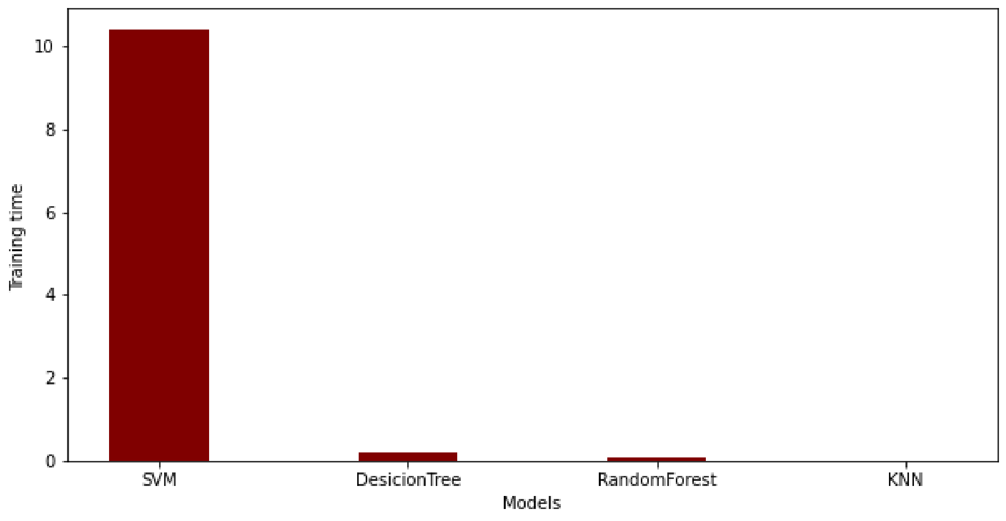
We have different models, we tried polynomial and, multiple regression and some models such as (no feature selection, normalized, standardized)

Some problems also, that MSE is too Large about 35 Trillion

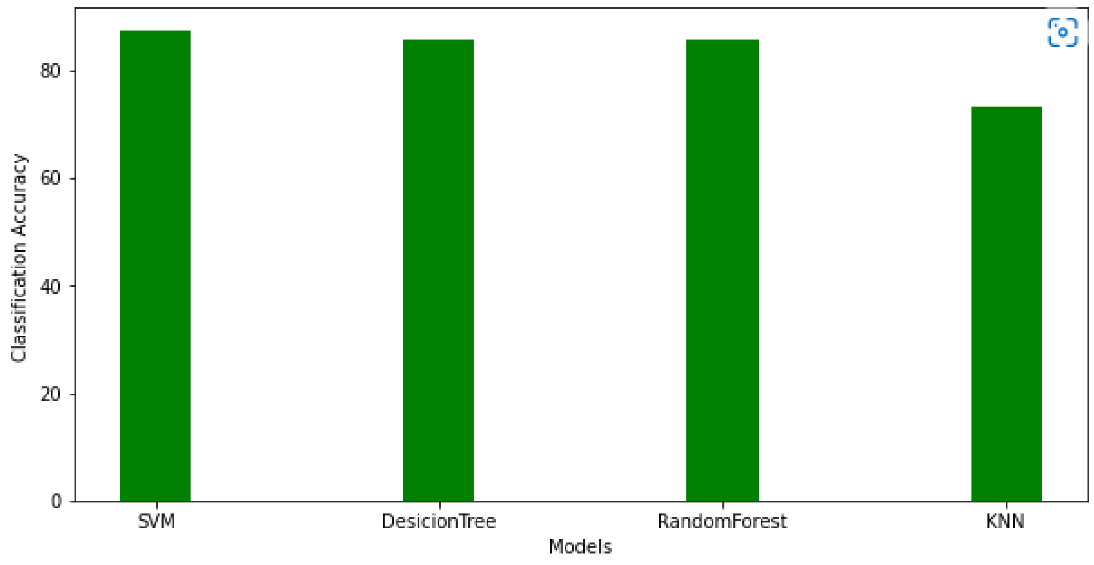
We reduced it to 960 billion (in polynomial case!) by increase degree of polynomial

CLASSIFICATION:

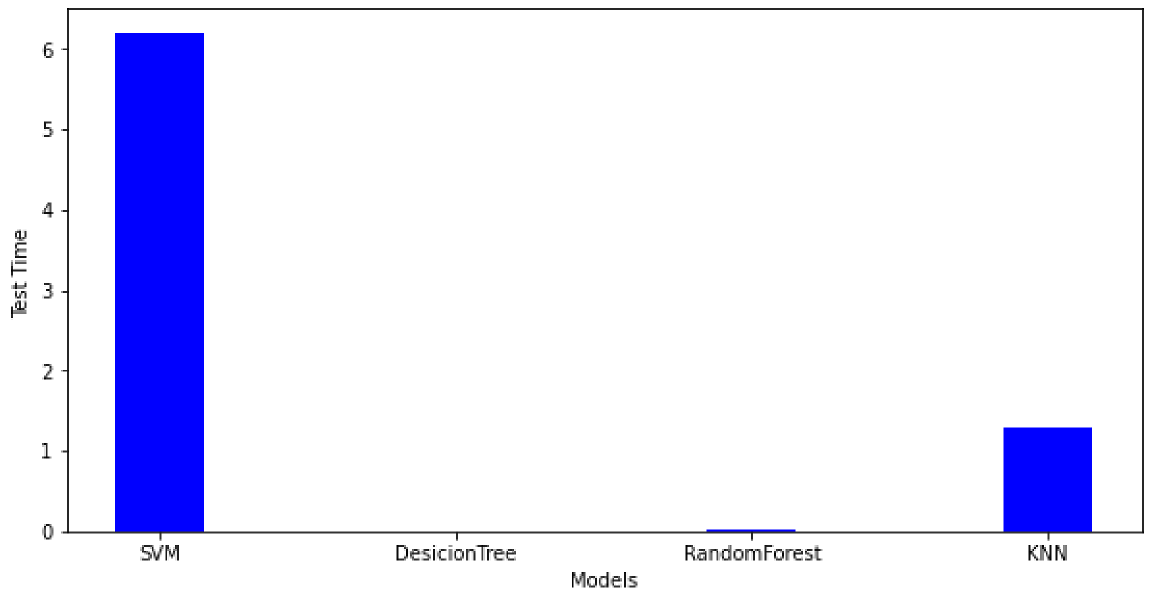
Total Training Time



Classification Accuracy



Total Test Time



Feature selection

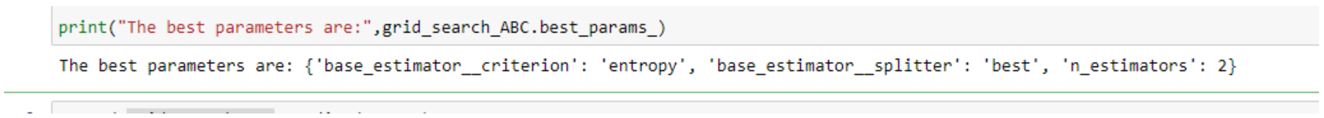
Probably no change in feature selection for classification (comparing with regression)

We approximately have 78 features after pre processing (it was 94 features)and drop some of non-important columns and we apply some of feature selection techniques such as

Correlation with Kendall method and Pearson Method . In the final we have approximately 70-73 features with Max Accuracy 87.44%

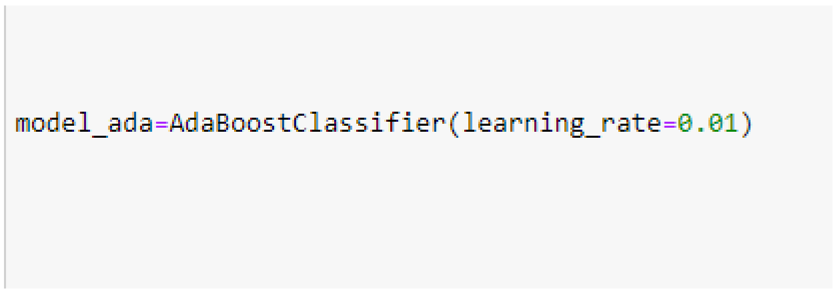
Hyperparameter Effect

We used the so-called " GridSearchCV ", it determines what the appropriate "paramater" can be used automatically, and the following image shows some of these paramaters



```
print("The best parameters are:",grid_search_ABC.best_params_)
The best parameters are: {'base_estimator__criterion': 'entropy', 'base_estimator__splitter': 'best', 'n_estimators': 2}
```

We use also Learning Rate “alpha” in ADABOOST Classifier



```
model_ada=AdaBoostClassifier(learning_rate=0.01)
```

The more we reduce its value, Accuracy is reduced also , and the more we increase it, the better performance and accuracy , but NOTE THAT , not always because at some point its Accuracy   decline again.

Conclusion

In the end, we made about five models ("SVM","DesicionTree","RandomForest","KNN",”AdaBoost”) and we modified some characteristics in the Pre-Processing stage and certainly we faced some difficulties in the beginning, such as how to deal with multiclass case and how to receive a new file and new data in order to test it and include the predicted values into it,we solved problems efficiently .The best score was SVM with Accuracy about 87.44%