Player Value Prediction

(CS_31)

20191700215	○ حسن وليد حسن السيد
20191700140	○ أميره أيمن محمد على
20191700353	○ عبدالله رمضان السيد ابراهيم
20191700883	○ ريهام صلاح على حسن
20191700387	○ عصام الدين شريف جاب الله

Preprocessing techniques:

1) **we made feature encoder by the label encoding on the string data , we try use (one hot encoding) but the (one hot encoding) leads to error because the value of column (nationality) is more than 7 different values**

2) **we made feature scaling of data by sklearn preprocessing minmax scaler because it preserves the shape of the original distribution, it doesn't meaningfully change the information embedded in the original data and it doesn't reduce the importance of outliers Unlike RobustScaler.**

3) **In function Filling Data:**
   - **We fill the empty cells in String columns to "no"**
   - **We are filling the empty cells in numeric columns to the median of each column.**
   - **We fill the empty cells in these columns ('LS', 'ST', 'RS', 'LW', 'LF', 'CF', 'RF', 'RW', 'LAM', 'CAM', 'RAM', 'LM', 'LCM', 'CM', 'RCM', 'RM', 'LWB', 'LDM', 'CDM', 'RDM', 'RWB', 'LB', 'LCB', 'CB', 'RCB', 'RB') to "0+0"**
   - **We fill the empty cells in date columns to "1/1/1990"**

4) **The number of feature selection is 5 in regression model**
   and number of feature selection is 84 in classification model by the correlation function , and we try use the
   chi squared but it causes the accuracy reduction and increase the run time.

5) **we split columns that have multi values by special characters like (',' , '/' , '+') using split function**

Classification Models:

- ## Decision Tree:

  It looks at the variables in a data set, determines which are most important.
  Tree of decisions which best partitions the data.

  ### Hyper parameter:

  - **(max_depth)** is one way to combat overfitting because when the tree is deeper, it's be more complex. Because there will be more splits and it captures more information about the data, the model will fit perfectly for the training data and will not be able to generalize well on test set. So, if your model is overfitting, so it can be controlled by max_depth.

    Best max_depth in the model is 12.
  - **(Criterian)** is Information Gain. Information Gain is used for splitting the nodes when the target variable is categorical. It works on the concept of the entropy. Entropy is used for calculating the purity of a node. Lower the value of entropy, higher is the purity of the node.

    Best criterian in the model is entropy.

- ## Adaboost Algorithm:

  It is an iterative ensemble method that build a strong classifier by combining multiple poorly performing classifiers.

  ### Hyper parameter:

  - **(n_estimators)** is the maximum number of estimators at which boosting is terminated. In case of perfect fit, the learning procedure is stopped early.

    Best n_estimators in the model is 100
  - (**algorithm**)

    If 'SAMME.R' then use the SAMME.R real boosting algorithm. base_estimator must support calculation of class probabilities.
    If 'SAMME' then use the SAMME discrete boosting algorithm. The SAMME.R algorithm typically converges faster than SAMME, achieving a lower test error with fewer boosting iterations.

    Best algorithm in the model is 'SAMME'
  - (**max_depth**) **the same thing in decision tree.**

    Best max_depth in the model is 12.

- ## Logistic Regression:

  Used to assign observations to a discrete set of classes.

  ### Hyper parameter:

  - **(Solver):**

    Algorithm to use in the optimization problem. Default is 'lbfgs'. To choose a solver, you might want to consider the following aspects

    Best n_estimators in the model is liblinear

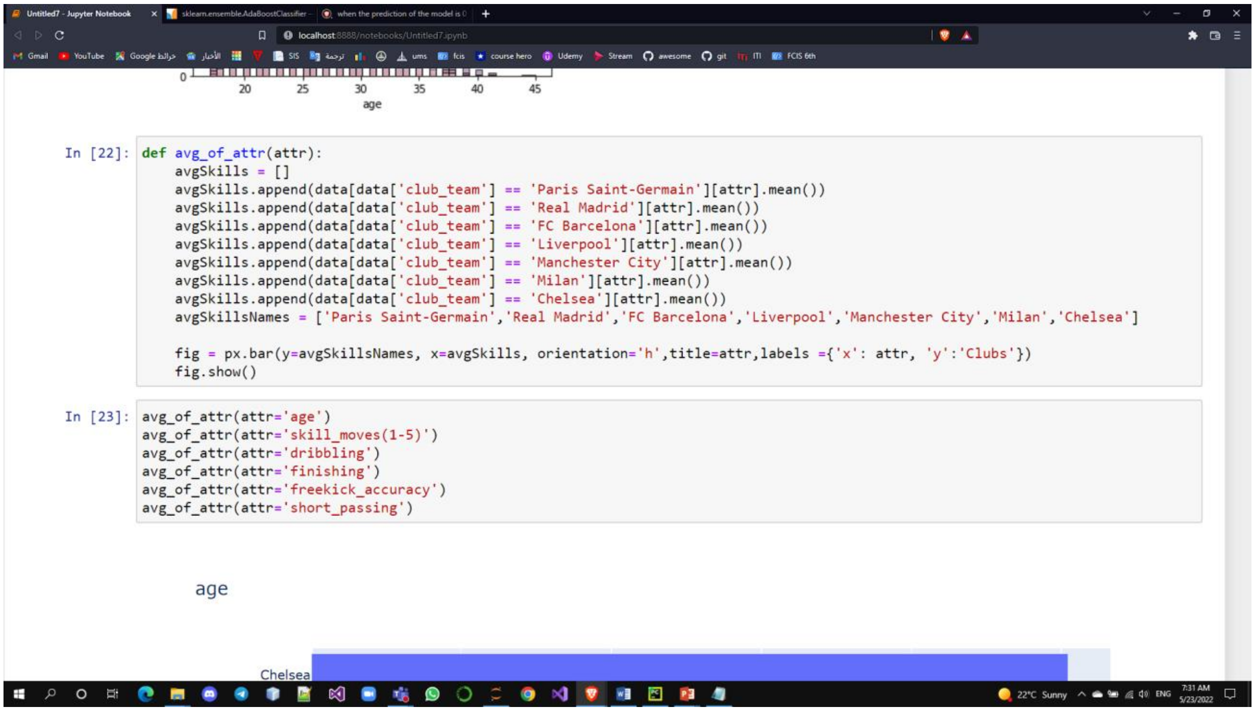
- **(C)**:
  Inverse of regularization strength; must be a positive float. Like in support vector machines, smaller values specify stronger regularization.
  Best C in the model is 10000

- **SVM:**
  - **Hyper parameter: (C)**:
    While increasing the regularization parameter (C) the accuracy is increasing too until it comes to 0.8 it has no effect.
    Best C in the model is 0.95

Time:

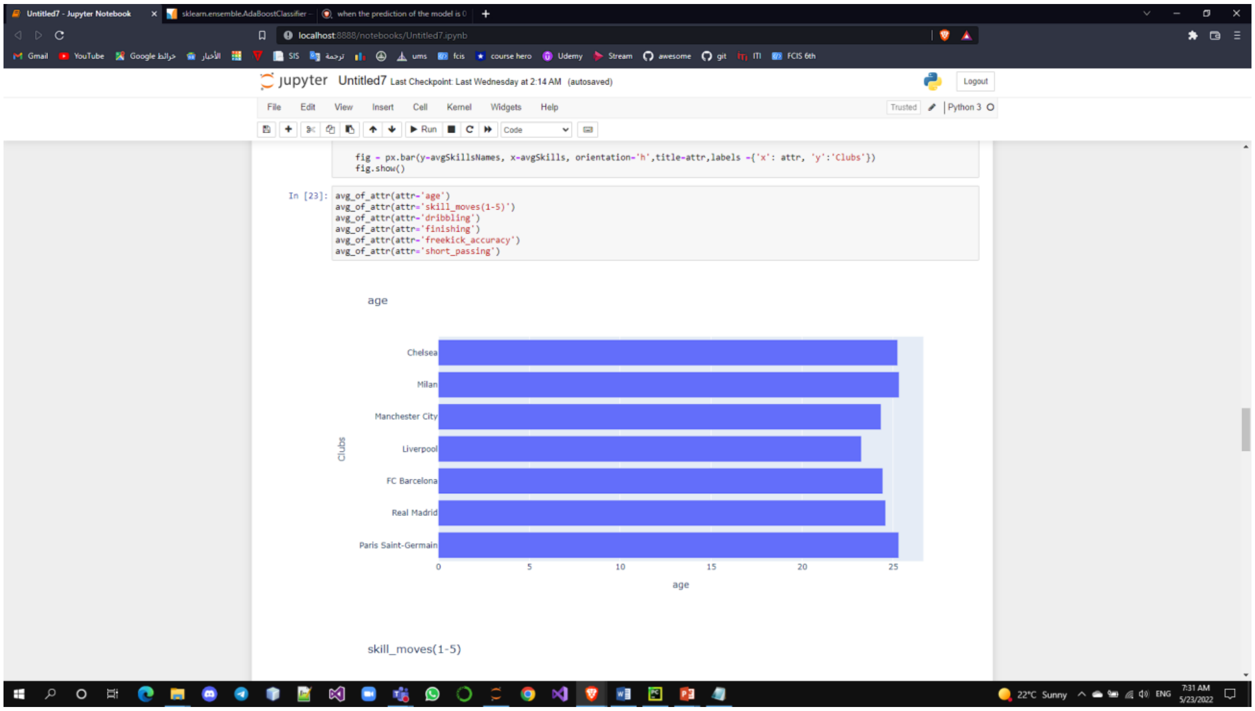
Model	**Training time**	**Test time**
Decision tree	**0.27**	**0.001**
Adaboost classifier	**35.7**	**0.099**
SVM polynomial	**13.03**	**0.65**
SVC Linear	**7.75**	**1.21**
SVC rbf	**12.61**	**4.35**
Linear SVC	**1.13**	**0.01**
Logistic Regression	**24.12**	**0.001**

Data Visualization:

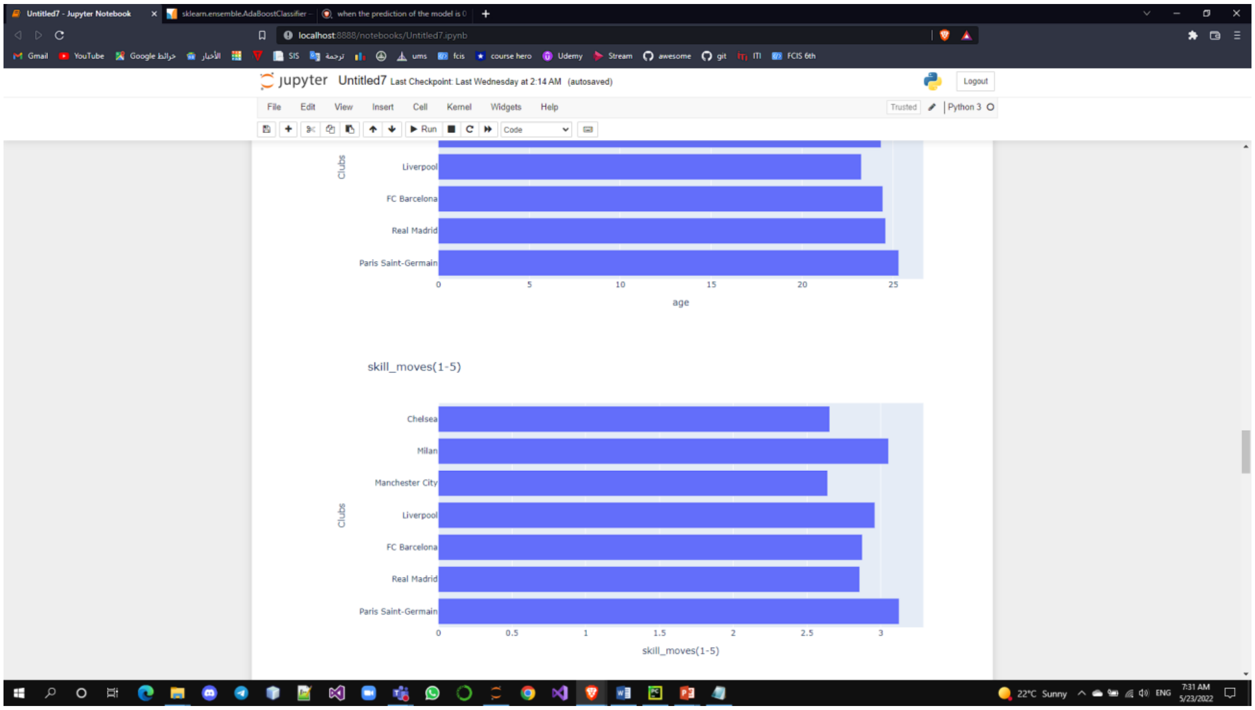


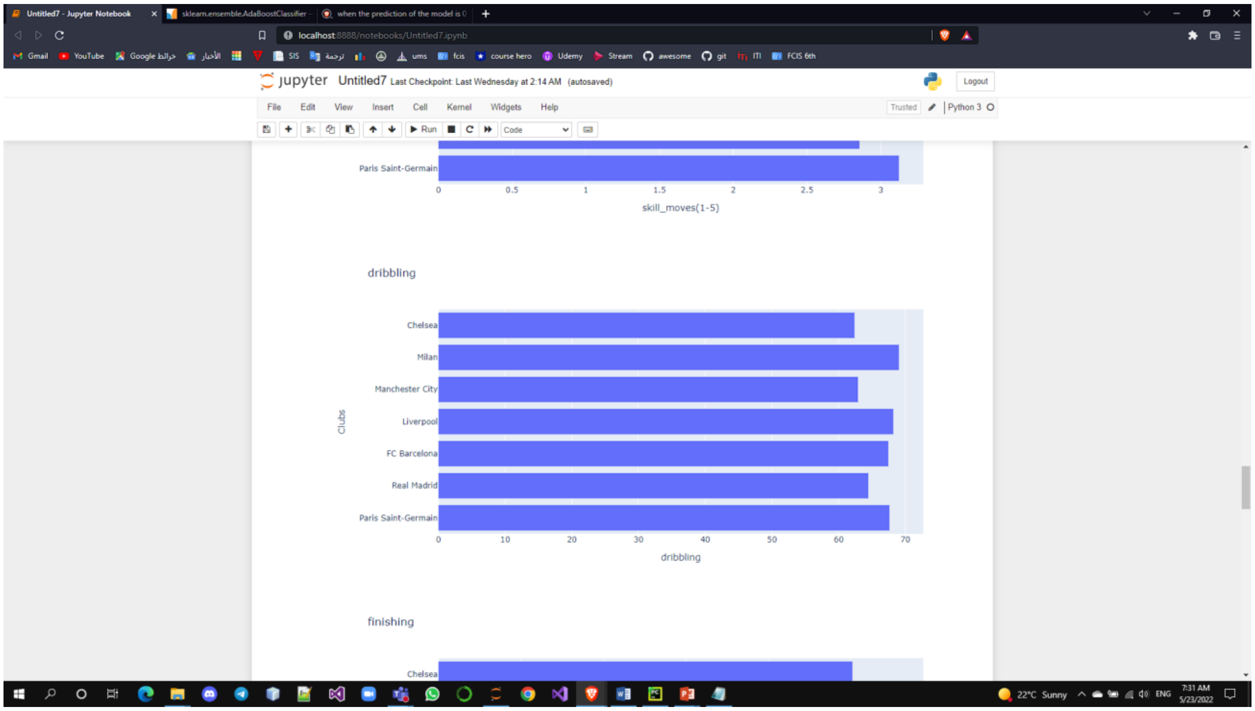
```
In [22]: def avg_of_attr(attr):
             avgSkills = []
             avgSkills.append(data[data['club_team'] == 'Paris Saint-Germain'][attr].mean())
             avgSkills.append(data[data['club_team'] == 'Real Madrid'][attr].mean())
             avgSkills.append(data[data['club_team'] == 'FC Barcelona'][attr].mean())
             avgSkills.append(data[data['club_team'] == 'Liverpool'][attr].mean())
             avgSkills.append(data[data['club_team'] == 'Manchester City'][attr].mean())
             avgSkills.append(data[data['club_team'] == 'Milan'][attr].mean())
             avgSkills.append(data[data['club_team'] == 'Chelsea'][attr].mean())
             avgSkillsNames = ['Paris Saint-Germain','Real Madrid','FC Barcelona','Liverpool','Manchester City','Milan','Chelsea']

             fig = px.bar(y=avgSkillsNames, x=avgSkills, orientation='h',title=attr,labels ={'x': attr, 'y':'Clubs'})
             fig.show()
```

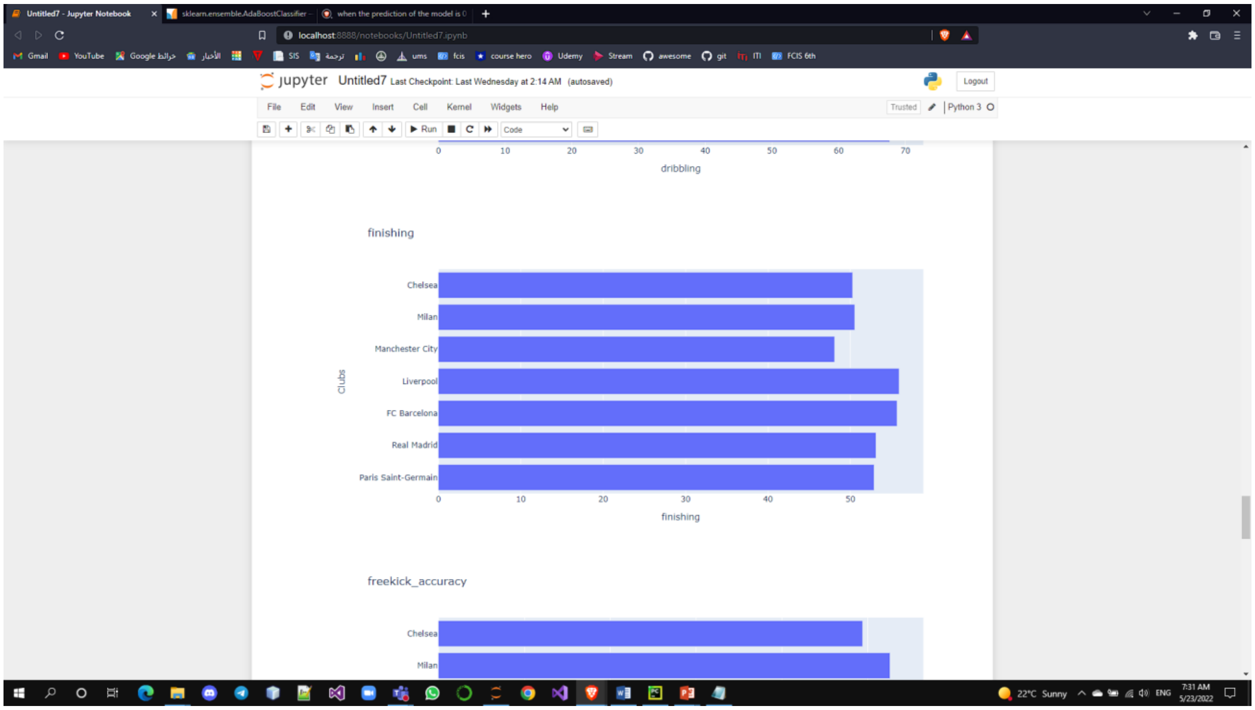
```
In [23]: avg_of_attr(attr='age')
         avg_of_attr(attr='skill_moves(1-5)')
         avg_of_attr(attr='dribbling')
         avg_of_attr(attr='finishing')
         avg_of_attr(attr='freekick_accuracy')
         avg_of_attr(attr='short_passing')
```

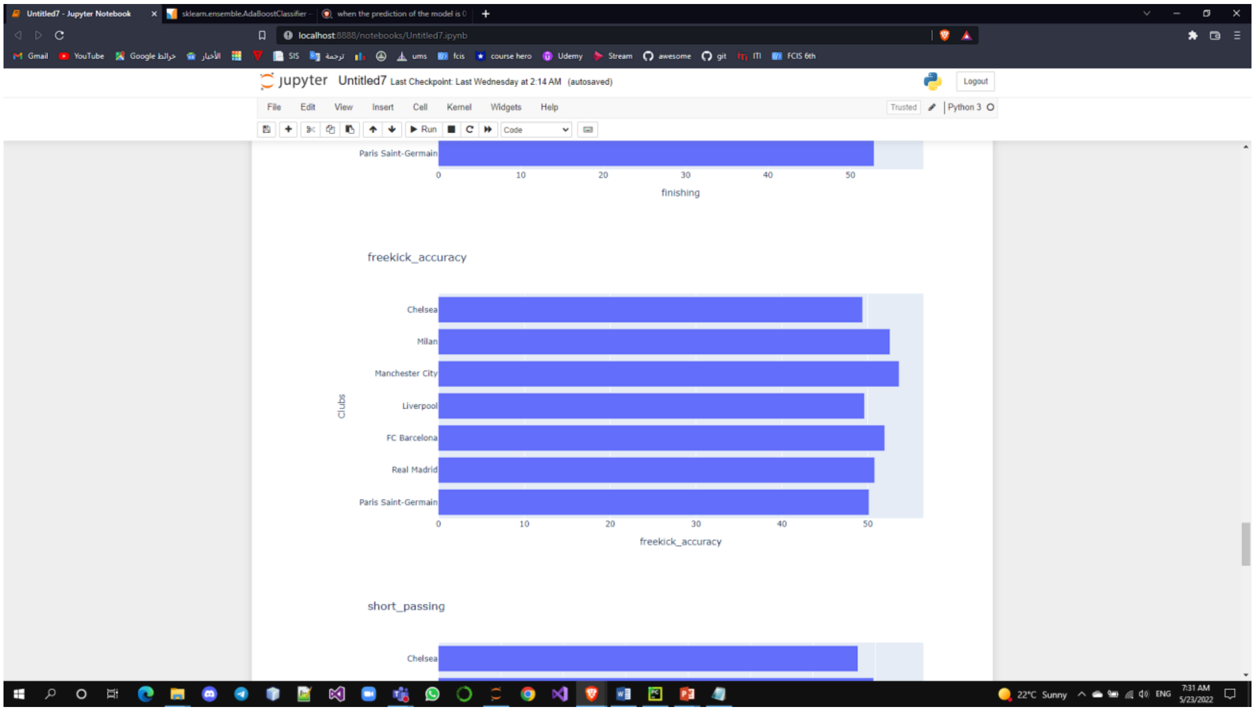


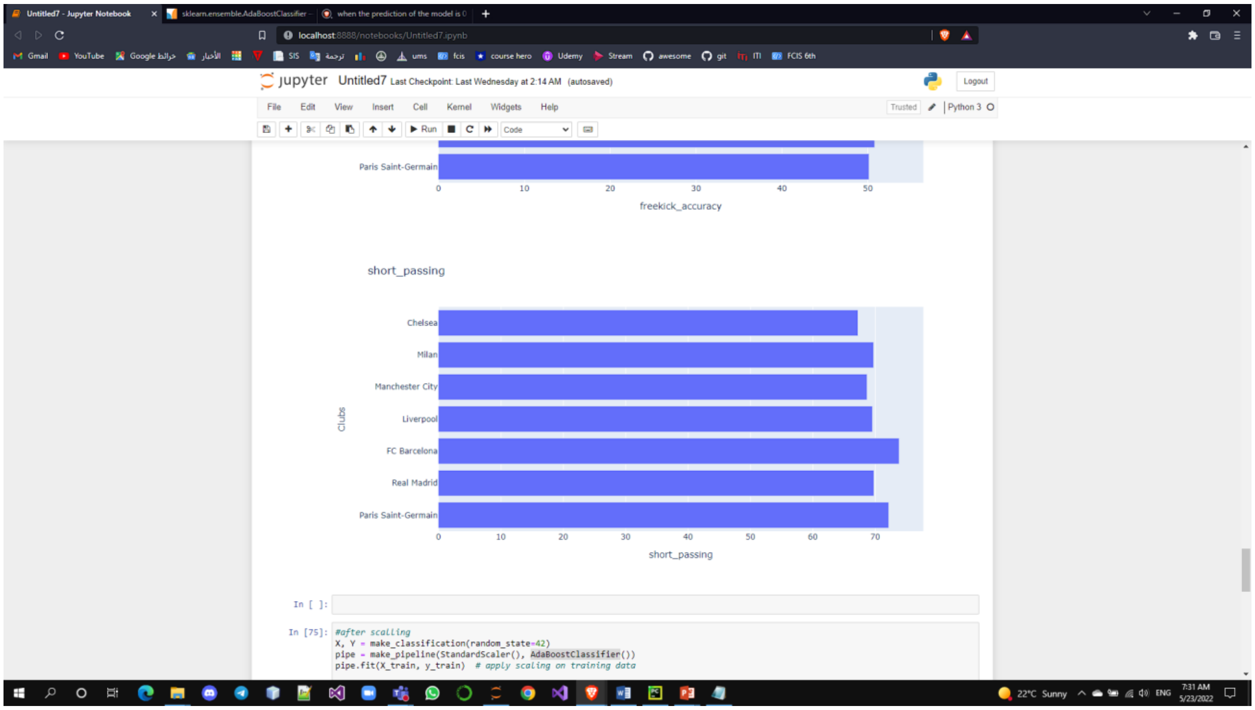
```
             fig = px.bar(y=avgSkillsNames, x=avgSkills, orientation='h',title=attr,labels ={'x': attr, 'y':'Clubs'})
             fig.show()

In [23]: avg_of_attr(attr='age')
         avg_of_attr(attr='skill_moves(1-5)')
         avg_of_attr(attr='dribbling')
         avg_of_attr(attr='finishing')
         avg_of_attr(attr='freekick_accuracy')
         avg_of_attr(attr='short_passing')
```


Untitled7 - Jupyter Notebook
sklearn.ensemble.AdaBoostClassifier
when the prediction of the model is 0
localhost:8888/notebooks/Untitled7.ipynb
jupyter Untitled7 Last Checkpoint: Last Wednesday at 2:14 AM (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
Clubs
Liverpool
FC Barcelona
Real Madrid
Paris Saint-Germain
0
5
10
15
20
25
age
skill_moves(1-5)
Chelsea
Milan
Manchester City
Liverpool
FC Barcelona
Real Madrid
Paris Saint-Germain
0
0.5
1
1.5
2
2.5
3
skill_moves(1-5)
22°C Sunny
ENG
7:31 AM
5/23/2022


Untitled7 - Jupyter Notebook
sklearn.ensemble.AdaBoostClassifier
when the prediction of the model is 0
localhost:8888/notebooks/Untitled7.ipynb
jupyter Untitled7 Last Checkpoint: Last Wednesday at 2:14 AM (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
Paris Saint-Germain
0
0.5
1
1.5
2
2.5
3
skill_moves(1-5)
dribbling
Clubs
Chelsea
Milan
Manchester City
Liverpool
FC Barcelona
Real Madrid
Paris Saint-Germain
0
10
20
30
40
50
60
70
dribbling
finishing
Chelsea
22°C Sunny
ENG
7:31 AM
5/23/2022


Untitled7 - Jupyter Notebook
sklearn.ensemble.AdaBoostClassifier
when the prediction of the model is 0
localhost:8888/notebooks/Untitled7.ipynb
jupyter Untitled7 Last Checkpoint: Last Wednesday at 2:14 AM (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
0 10 20 30 40 50 60 70
dribbling
finishing
Chelsea
Milan
Manchester City
Liverpool
FC Barcelona
Real Madrid
Paris Saint-Germain
Clubs
0 10 20 30 40 50
finishing
freekick_accuracy
Chelsea
Milan
22°C Sunny
ENG
7:31 AM
5/23/2022


Untitled7 - Jupyter Notebook
sklearn.ensemble.AdaBoostClassifier
when the prediction of the model is 0
localhost:8888/notebooks/Untitled7.ipynb
jupyter Untitled7 Last Checkpoint: Last Wednesday at 2:14 AM (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
Paris Saint-Germain
0 10 20 30 40 50
finishing
freekick_accuracy
Chelsea
Milan
Manchester City
Liverpool
FC Barcelona
Real Madrid
Paris Saint-Germain
Clubs
0 10 20 30 40 50
freekick_accuracy
short_passing
Chelsea
22°C Sunny
ENG
7:31 AM
5/23/2022



```
In [ ]:

In [75]: #after scalling
         X, Y = make_classification(random_state=42)
         pipe = make_pipeline(StandardScaler(), AdaBoostClassifier())
         pipe.fit(X_train, y_train)  # apply scaling on training data
```

Conclusion:

- In this phase we used 5 classification models(Adaboost, Decision tree, SVM[SVC polynomial, Linear SVC, SVC linear, SVC rbf] and Logistic Regression)
- The objective from using these models is trying to get the highest accuracy in dataset.
- The Accuracy of each model is:

Model	Accuracy
Decision Tree	94.29%
Adaboost Classifier	93.1%
SVC Linear	88%
Linear SVC	88.7%
SVM polynomial	99.2%
SVC rdf	88%
Logistic Regression	81.9%