Data Mining project report

BPL PREDICTION MATCHS

1. **Introduction/Motivation:**

In football, the first English league is known to be the most competitive. Indeed, many clubs have many star players which makes each match close. Thus, the goal of our project is to be able to predict future matches with technology combining Data Mining (Machine Learning) and Big Data.

We have collected our data which represents the matches of the last 10 seasons. Each match is characterized by the final results and overall statistics and we have also added the statistics during the match (KPI: corner, free kick, etc.).

Date	HomeTeam	AwayTeam	FTHG	FTAG	FTR	HTHG	HTAG	HTR	Referee	HS	AS	HST	AST	HF	AF	HC
16/08/14	Arsenal	Crystal Palace	2	1	H	1	1	D	J Moss	14	4	6	2	13	19	9
16/08/14	Leicester	Everton	2	2	D	1	2	A	M Jones	11	13	3	3	16	10	3

```
Arsenal
Games Win Percent: 0.49295774647887325
Games Loose Percent: 0.29295774647887324
Games Draw Percent: 0.2140845070422535
['Arsenal',
 684,
 2495,
 684,
 540,
 31,
 3592,
 2248,
 5315,
 3704,
 14.971830985915492,
 10.433802816901409,
 1811,
 0.7258517034068136,
 0.871307619943556,
 1.3776918829376035,
 3.6476608187134505]
```

Knowing that to answer this problem we need a lot of data, Data Mining is required. It is the most efficient for this kind of question. So our general problem is: knowing the past 10 seasons, what would be the results in the next games? This problem can thus be applied in the world of sports betting (Betclic, etc.).

2. **Data and Data Analysis:** How this type of analysis is adequate for the data, problem and questions posed?

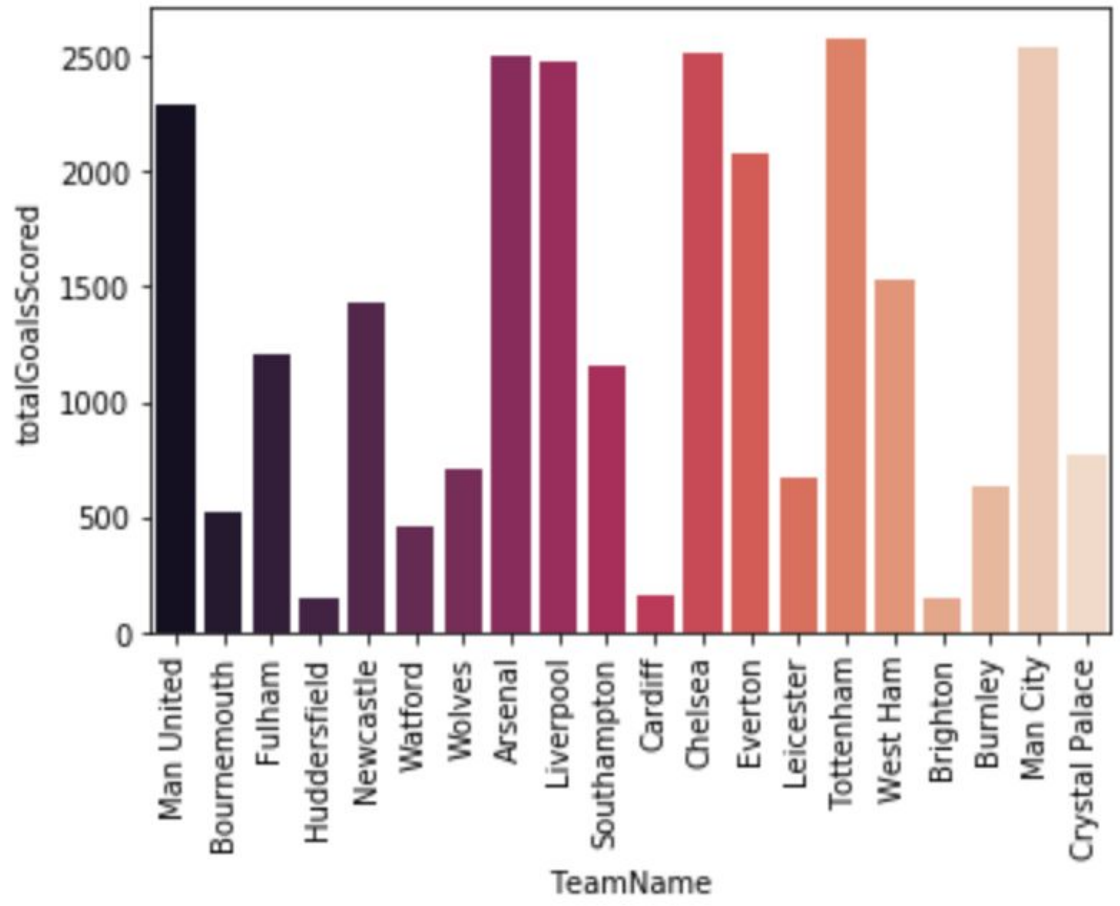
At first, we only use the 10 csv files corresponding to the matches of the last 10 seasons. We then concatenate these files to have only all seasons in our csv *allsaisons* file.

```
Date,HomeTeam,AwayTeam,FTHG,FTAG,FTR,HTHG,HTAG,HTR,Referee,HS,AS,HST,AST,HF,AF,HC,AC,HY,AY
,HR,AR
17/08/13,Arsenal,Aston Villa,1,3,A,1,1,D,A Taylor,16,9,4,4,15,18,4,3,4,5,1,0
17/08/13,Liverpool,Stoke,1,0,H,1,0,H,M Atkinson,26,10,11,4,11,11,12,6,1,1,0,0
17/08/13,Norwich,Everton,2,2,D,0,0,D,M Oliver,8,19,2,6,13,10,6,8,2,0,0,0
17/08/13,Sunderland,Fulham,0,1,A,0,0,D,N Swarbrick,20,5,3,1,14,14,6,1,0,3,0,0
17/08/13,Swansea,Man United,1,4,A,0,2,A,P Dowd,17,15,6,7,13,10,7,4,1,3,0,0
17/08/13,West Brom,Southampton,0,1,A,0,0,D,K Friend,11,7,1,2,14,24,4,8,4,0,0,0
17/08/13,West Ham,Cardiff,2,0,H,1,0,H,H Webb,18,12,4,1,10,7,4,3,0,1,0,0
18/08/13,Chelsea,Hull,2,0,H,2,0,H,J Moss,22,7,5,2,7,16,5,1,0,1,0,0
18/08/13,Crystal Palace,Tottenham,0,1,A,0,0,D,M Clattenburg,5,17,3,2,6,9,3,7,1,0,0,0
19/08/13,Man City,Newcastle,4,0,H,2,0,H,A Marriner,20,5,11,1,9,7,8,1,2,3,0,1
21/08/13,Chelsea,Aston Villa,2,1,H,1,1,D,K Friend,15,7,3,3,12,13,1,2,1,4,0,0
24/08/13,Aston Villa,Liverpool,0,1,A,0,1,A,M Clattenburg,17,5,3,1,9,8,8,2,3,3,0,0
24/08/13,Everton,West Brom,0,0,D,0,0,D,R East,22,7,8,2,14,15,11,1,1,1,0,0
24/08/13,Fulham,Arsenal,1,3,A,0,2,A,H Webb,16,18,7,7,10,8,1,8,2,2,0,0
```

With only this kind of data, we can already know the probability of winning at home (only 45%!). And of course we can have the victory and defeat percentage for each team with their match statistics (yellow and red card, corner, fouls, etc.).

```
Number of Matches: 3550
Number of Features: 21
Number of matches won by HOME: 1632
Win rate of HOME team: 45.97183098591549
```

We first wanted to know which club has scored the most in the last 10 seasons. Thus thanks to our new table on the stats of each team we have direct access to the sum of the goals of each match of the team during the last 10 years in the league.

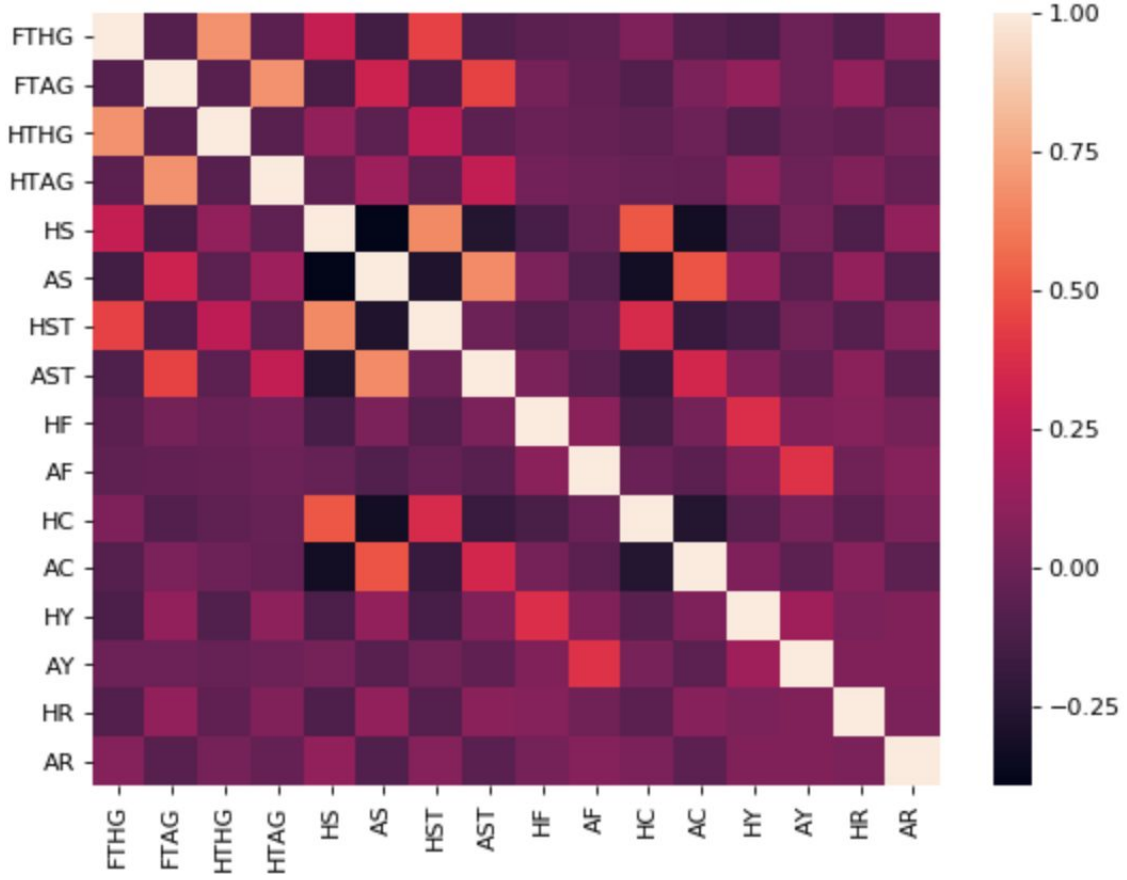


we then filter 3 columns of the match table: the match result (half and full time) and the referee (which do not seem very important for our problem).

When we display the correlation table of the different columns, we see that the variable corresponding to the half-time result seems useless for our processing.

Hover to magify

	FTHG	FTAG	HTHG	HTAG	HS	AS	HST	AST	HF	AF	HC	AC	HY	AY	HR	AR
FTHG	1	-0.078	0.69	-0.059	0.29	-0.15	0.44	-0.097	-0.057	-0.046	0.054	-0.079	-0.11	-0.0051	-0.085	0.075
FTAG	-0.078	1	-0.073	0.69	-0.13	0.32	-0.1	0.45	0.022	-0.033	-0.088	0.051	0.12	-0.0063	0.12	-0.069
HTHG	0.69	-0.073	1	-0.068	0.11	-0.052	0.26	-0.052	-0.0084	-0.026	-0.051	-0.0016	-0.091	-0.02	-0.042	0.027
HTAG	-0.059	0.69	-0.068	1	-0.051	0.15	-0.056	0.28	0.019	-0.0054	-0.022	-0.027	0.1	0.0016	0.065	-0.028
HS	0.29	-0.13	0.11	-0.051	1	-0.39	0.66	-0.26	-0.14	-0.022	0.51	-0.32	-0.12	0.029	-0.11	0.11
AS	-0.15	0.32	-0.052	0.15	-0.39	1	-0.28	0.67	0.05	-0.09	-0.32	0.5	0.12	-0.07	0.12	-0.092
HST	0.44	-0.1	0.26	-0.056	0.66	-0.28	1	8.7e-05	-0.084	-0.029	0.36	-0.19	-0.14	0.014	-0.082	0.077
AST	-0.097	0.45	-0.052	0.28	-0.26	0.67	8.7e-05	1	0.046	-0.07	-0.18	0.34	0.067	-0.041	0.099	-0.062
HF	-0.057	0.022	-0.0084	0.019	-0.14	0.05	-0.084	0.046	1	0.097	-0.12	0.022	0.38	0.067	0.076	0.025
AF	-0.046	-0.033	-0.026	-0.0054	-0.022	-0.09	-0.029	-0.07	0.097	1	-0.012	-0.059	0.061	0.39	0.012	0.078
HC	0.054	-0.088	-0.051	-0.022	0.51	-0.32	0.36	-0.18	-0.12	-0.012	1	-0.26	-0.076	0.034	-0.059	0.051
AC	-0.079	0.051	-0.0016	-0.027	-0.32	0.5	-0.19	0.34	0.022	-0.059	-0.26	1	0.057	-0.056	0.084	-0.056
HY	-0.11	0.12	-0.091	0.1	-0.12	0.12	-0.14	0.067	0.38	0.061	-0.076	0.057	1	0.16	0.049	0.059
AY	-0.0051	-0.0063	-0.02	0.0016	0.029	-0.07	0.014	-0.041	0.067	0.39	0.034	-0.056	0.16	1	0.058	0.066
HR	-0.085	0.12	-0.042	0.065	-0.11	0.12	-0.082	0.099	0.076	0.012	-0.059	0.084	0.049	0.058	1	0.05
AR	0.075	-0.069	0.027	-0.028	0.11	-0.092	0.077	-0.062	0.025	0.078	0.051	-0.056	0.059	0.066	0.05	1



After this we make a scale on each features name with their role in the match (home or away):

```
      HomeTeam_Arsenal  HomeTeam_Bournemouth  ...        HR        AR
0                    1                     0  ... -0.250749  3.103158
1                    0                     0  ... -0.250749 -0.297376
6                    0                     0  ...  3.661985  3.103158
7                    0                     0  ... -0.250749 -0.297376
8                    0                     0  ... -0.250749 -0.297376
...                ...                   ...  ...       ...       ...
3540                 0                     0  ... -0.250749 -0.297376
3541                 0                     0  ... -0.250749 -0.297376
3543                 0                     0  ... -0.250749  6.503692
3544                 0                     0  ... -0.250749 -0.297376
3546                 0                     0  ... -0.250749 -0.297376
```

This pre-processing method is the most suitable for our question because we have centered our new variables at 0 (combining the teams with their role in the match). For example for the club manchest united we will have two new variables: home manchester united and away manchester united. we also see this scaling here:

```
      HomeTeam_Arsenal  HomeTeam_Bournemouth  ...        HR        AR
0                    1                     0  ... -0.250749  3.103158
1                    0                     0  ... -0.250749 -0.297376
6                    0                     0  ...  3.661985  3.103158
7                    0                     0  ... -0.250749 -0.297376
8                    0                     0  ... -0.250749 -0.297376
...                ...                   ...  ...       ...       ...
3540                 0                     0  ... -0.250749 -0.297376
3541                 0                     0  ... -0.250749 -0.297376
3543                 0                     0  ... -0.250749  6.503692
3544                 0                     0  ... -0.250749 -0.297376
3546                 0                     0  ... -0.250749 -0.297376

[1498 rows x 56 columns]
0       H
1       D
6       A
7       H
8       A
       ..
3540    A
3541    D
3543    H
3544    H
3546    D
Name: FTR, Length: 1498, dtype: object
```

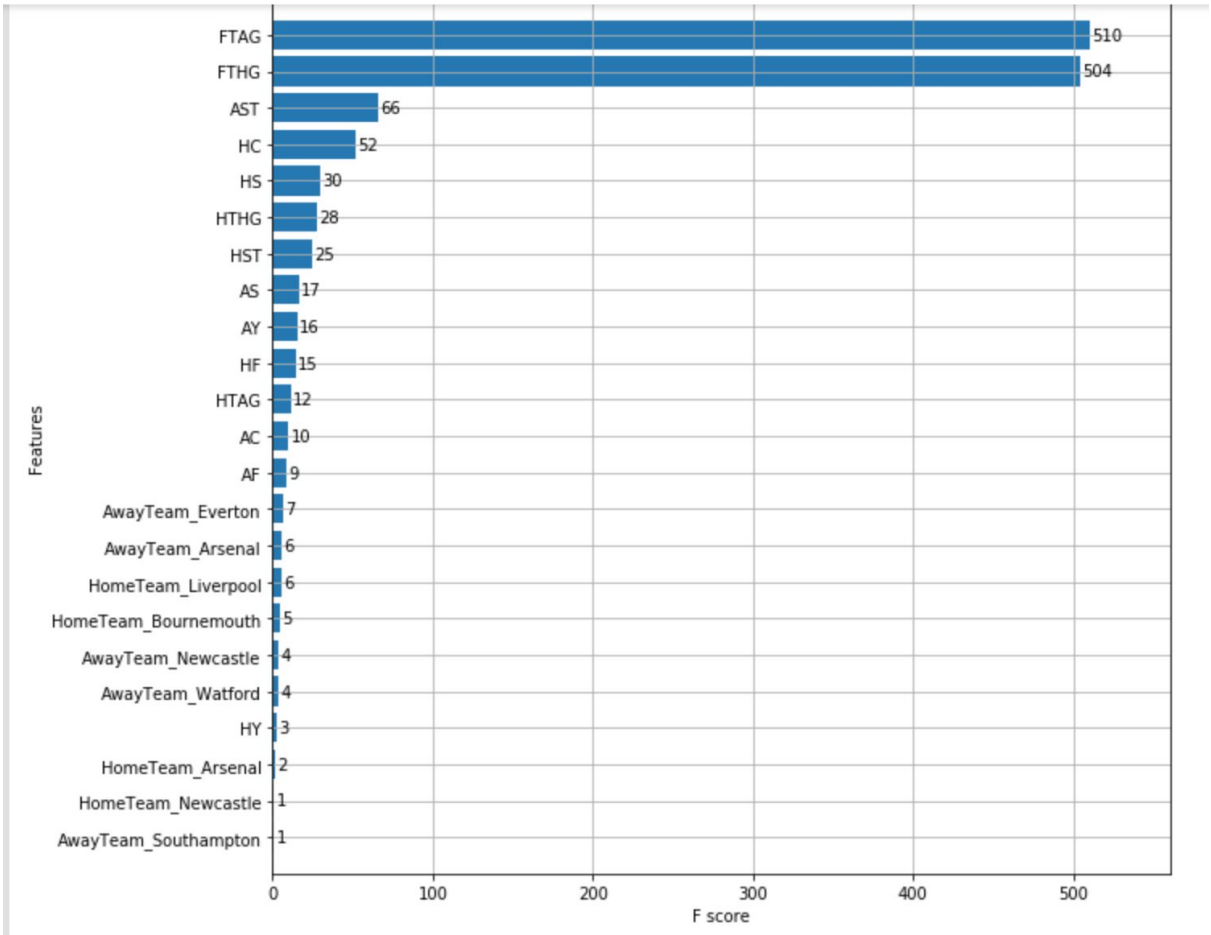
3. **Architecture of Proposed Solution:**

for the processing phase, we decided to predict 20% of all matches in our match table. We tested 5 different methods to predict these 20% of total matches: Random Forest, Logistic Regression, SVM, KNeighbor and XGBOOST

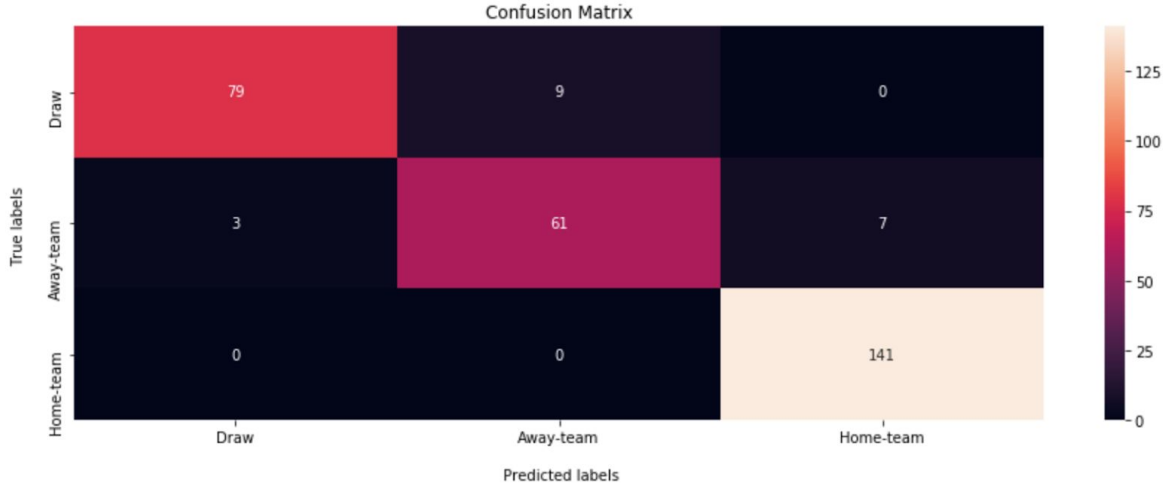
For each method we decided to compare them thanks to their accuracy score on the predictions of the test data.

	Model	accuracy
1	logistic regression	1.000000
4	XGB	1.000000
2	SVM	0.976667
0	random forest	0.936667
3	KNN	0.726667

then we try to see the importance of each features name. It is calculated according to the F score of each feature for the prediction of XGBOOST classifier.



knowing that random forest and XGB have the same accuracy score, we also want to display the random forest confusion matrix:



as we see for predicted draw, 79 are truly Draw and only 3 are true Away winer.

for the predicted away wins, 61 are truly Away wins and 9 are truly Draw

for the predicted home wins, 141 are truly Home wins and 7 Away wins.

But we can also say that in the total 141 Real Home wins matches, we have predicted all Home wins, so 100% for real home wins matches.

At this point we create a csv file and change the H, D A value of the winer of each match by number (Home:1, Draw: 0 and Away: 1). This csv file is to prepare Pyspark part of our project. We couldn't implement pyhive with our project so we implement PySpark.

Now, we create with PySpark a vector to combine all features names. We concatenate also the full time results of each match. These are our input (features) and our output (FTR) for our spark model.

```
root
 |-- Attributes: vector (nullable = true)
 |-- FTR: double (nullable = true)
```

We create these 2 vectors because Spark support only 2 columns during testing one for features and the other for labels.

After that we can now implement our model with the previous prediction method which is the most efficient for our model : Random Forest. And we show the different Test errors and accuracy:

```
+----------+---+--------------------+
|prediction|FTR|            features|
+----------+---+--------------------+
|       0.0|0.0|(56,[0,25,40,41,4...|
|       2.0|2.0|(56,[0,25,40,41,4...|
|       1.0|1.0|(56,[0,25,40,41,4...|
|       0.0|0.0|(56,[0,27,40,41,4...|
|       1.0|1.0|(56,[0,27,40,41,4...|
|       0.0|0.0|(56,[0,28,40,41,4...|
|       1.0|1.0|(56,[0,28,40,41,4...|
|       1.0|1.0|(56,[0,29,40,41,4...|
|       1.0|1.0|(56,[0,30,40,41,4...|
|       2.0|2.0|(56,[0,31,40,41,4...|
+----------+---+--------------------+
only showing top 10 rows

Test Error = 0.146974
Accuracy = 0.853026
```

Then we test our model with Kneighbor classifier to predict the future scores and show the probability of each possibility (home wins, away wins or draw) :

	Away Team	Draw	Home Team
0	0.0	40.0	60.0
1	80.0	20.0	0.0
2	0.0	20.0	80.0
3	0.0	80.0	20.0
4	20.0	40.0	40.0
...	...	...	...
295	60.0	40.0	0.0
296	0.0	20.0	80.0
297	0.0	20.0	80.0
298	100.0	0.0	0.0
299	0.0	0.0	100.0

300 rows × 3 columns

Then like scikit learn, we create 2 different features for each team. For example, Manchester United becomes Home manchester united and Away manchester united

Then we concatenate the predicted result, probabilities for each possibility and features to show our results

4. **Conclusions:**

After showing the Random Forest accuracy score (85%), we can say that our model answers majoritary to the problematic. We can predict the futures matches at 85ù accuracy. It's not perfect, but we can choose other features variables to improve the accuracy score (player stats for example).

With the player stats, we should answer also to another problematic like : who will score for each future match?

5. **References: The final report should include the full reference of the libraries, papers, code, tutorials that you have based your**

project on, your approach to solve the problems, and the tools and datasets that you have employed.

Datasets :

- https://datahub.io/sports-data/english-premier-league
- https://fixturedownload.com/results/epl-2018

Librairies :

- https://numpy.org/
- https://matplotlib.org/
- https://scikit-learn.org/stable/
- https://ipython.readthedocs.io/en/stable/api/generated/IPython.display.html
- https://seaborn.pydata.org/
- https://spark.apache.org/docs/latest/api/python/index.html
- https://pypi.org/project/sasl/

Tutorials :

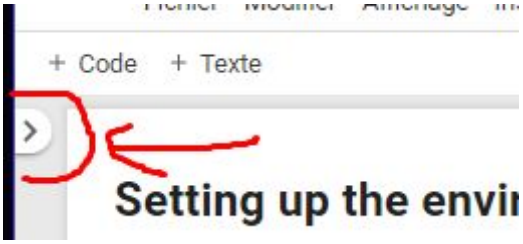
- https://github.com/Rajesh426/Google-Colab-Pyspark-Classification/blob/master/Installing_Spark_Classification_Task_Linear_Decision_Gradient_RandomForest_Models.ipynb?fbclid=IwAR1Jst2Q0BybB25xrcrvDCgWmIA280OtLmzNSBuCe0dGlx-4g0q5MmEjynE
- https://stackoverflow.com/questions/21370431/how-to-access-hive-via-python
- https://medium.com/@mcamara89/quick-hive-commands-and-tricks-3aa515b77a48
- https://community.cloudera.com/t5/Support-Questions/How-to-join-multiple-csv-files-in-folder-to-one-output-file/td-p/181377

ANNEXE:

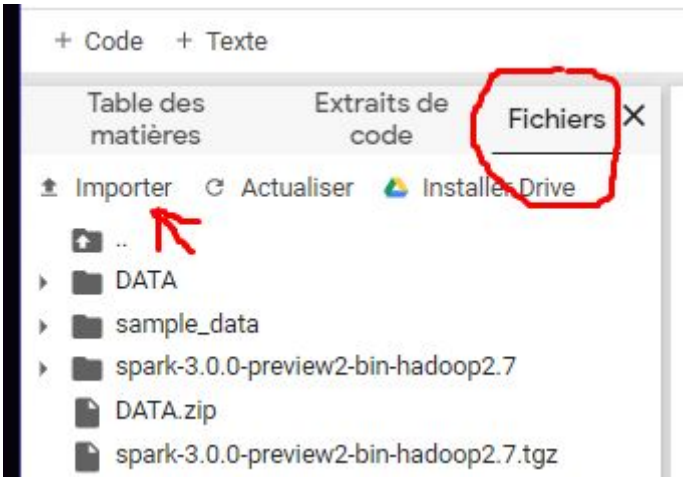
To run our code in google collabs, you will have to import DATA.zip on the files of the project.

To do that here is the steps :

1) you click here to open the location of the files:



2) you make sure to be on the window of the files and the you click in import:



3) Then you click in actualiser and you will see the Zip file added.

4) You can now run the Code