Machine Learning Engineer Nanodegree

Capstone Project

Dhrubojyoti Jasu
October 17th, 2018

I. Definition

Project Overview

For this capstone project, I will use an English Premier League Football dataset from http://football-data.co.uk/englandm.php . This is a binary classification problem similar to the CharityML in Supervised Learning section of the MLND. The legal sports-betting market in the U.S. was worth an estimated USD270 million in 2017 -- with another USD2.5 billion to USD3 billion in black market betting, according to research firm Eilers & Krejcik Gaming, LLC.

Even though it is not legalized in many other countries like India, but for my own interest, I want to build a predictive model capable of predicting if the home team will win a football match. Usually betting is conducted with human instincts but now we can use some machine learning algorithm to predict the result of the future matches also.

There are reports related to sports prediction using machine learning. Some of them I have listed below:

- Using Machine Learning to Predict the Outcome of English County twenty over Cricket Matches
- How I Used Machine Learning to Predict Soccer Games for 24 Months Straight
- Predicting Football Results With Statistical Modelling

I am a regular viewer of football around the globe, my favorite club is Manchester United and I follow English Premier League religiously and now I am excited I can use my knowledge of machine learning to have some fun with data.

Problem Statement

The problem is to use the existing dataset of EPL obtained through http://football-data.co.uk/englandm.php and use it to train some supervised learning algorithms to predict the matches of EPL. I want to predict whether a home team is gonna win the match by training some supervised learning algorithms.

Metrics

I will be using accuracy_score & fbeta_score from sklearn.metrics to evaluate both the benchmark and my final model. The goal of this project is to predict the win of 'Home Team' accurately. So accuracy as a metric to evaluate a model's performance is appropriate. However, predicting a team is not going to win is not that much important, hence, a model's ability to precisely predict the winner of a 'home team' is *more important* than the model's ability to recall those teams.

For this reason, we can use the F-beta score as a metric which considers both precision & recall

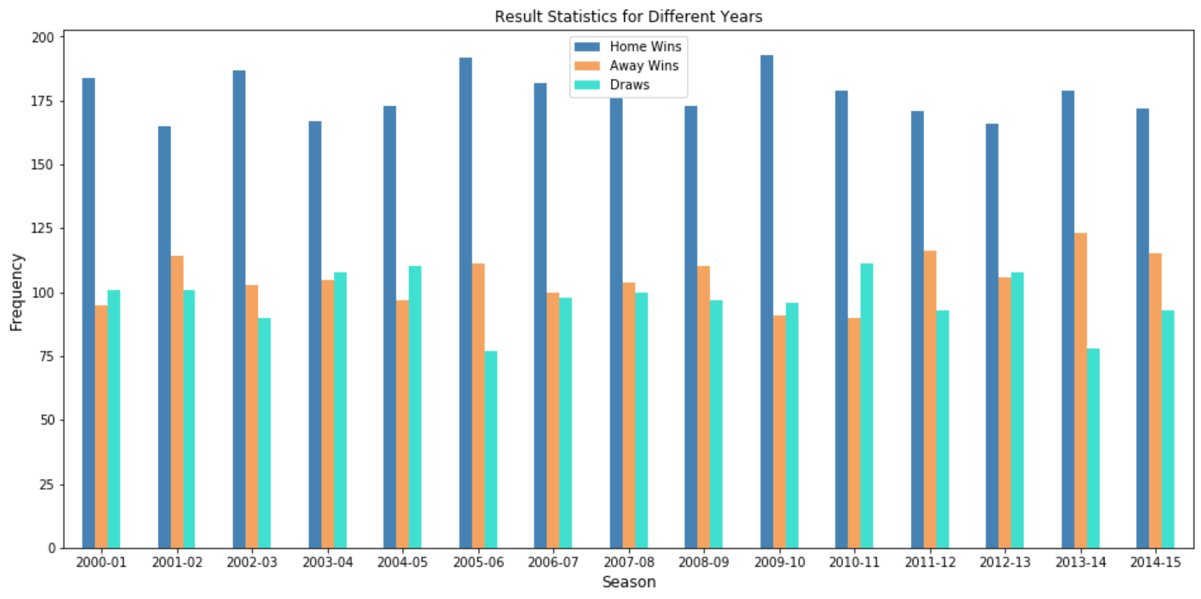
II. Analysis

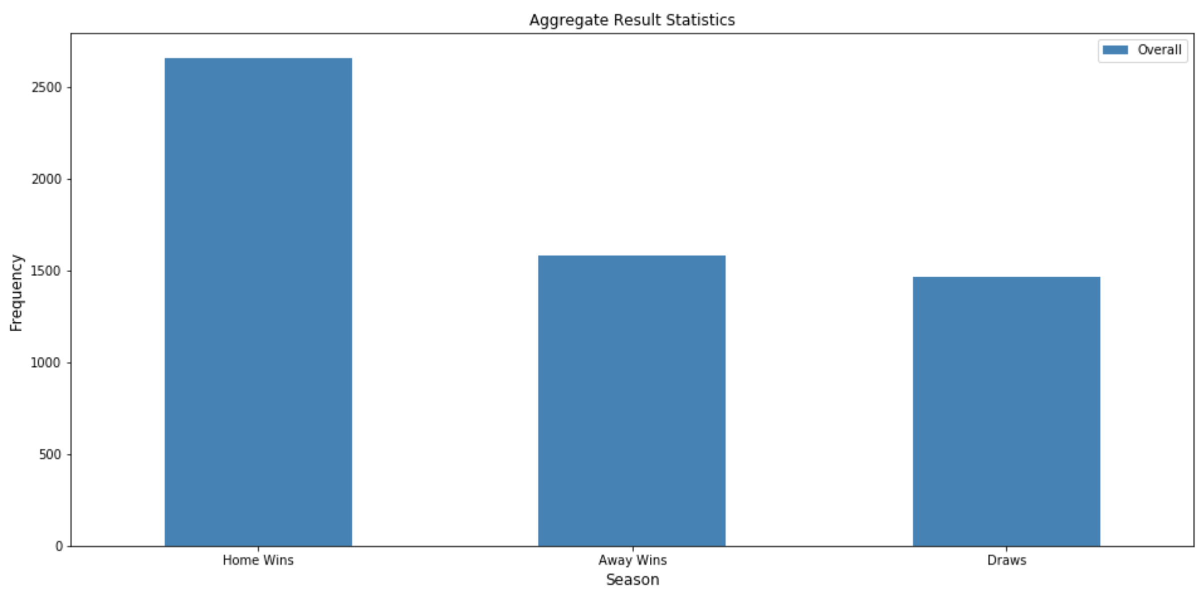
Exploratory Data Analysis of EPL sessions

In the Datasets I have downloaded all the sessions data separately from the http://football-data.co.uk/englandm.php . Then I consolidated all the datasets into a final dataset called My_Capstone_Dataset.csv file. Now I will use all separate sessions CSV files to find answers to some interesting questions listed below:

- How many matches have been won/drawn by home teams?
- What is the win percentage for home teams & away teams?

While searching for above two answers I will create some visualization for clear understanding.



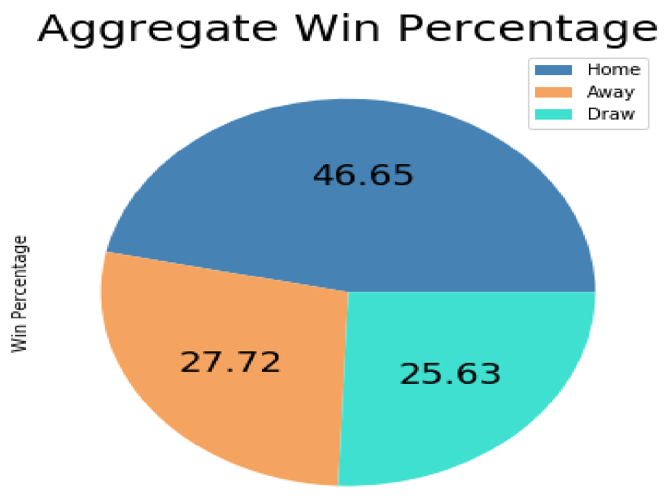


Displaying the overall results

	Home wins	Away Wins	Draws
Overall	2659	1580	1461

So our first question was *How many matches have been won/drawn by home teams?*

From the above visualization, we can clearly see more matches have won by the home teams only. It is too common in football matches as home team can feel advantage while the home crowd is supporting behind them and pitch conditions and other playing factors are also in favor of them. Now, let's visualize the win percentage for Home & Away teams.



So from above visualization also we can see 46% times Home team wins, 28% time Away team wins and 25% time matches are drawn. So the majority of the times home team usually wins.

Algorithms & Techniques

Some questions I think are right can be asked which are listed below:

- What model should we use?
- What are the features (the aspects of a game) that matter the most to predicting a team wins?
- Does being the home team give a team the advantage?

I will use three *supervised learning* algorithm listed below to train and predict on my dataset. I will deploy a train_test_split to shuffle & split data into a training & testing set and will compare the metrics of *three* learning algorithms and will choose the best one amongst them. Finally, I will perform a hyperparameter tuning to optimize my final classifier. Then I will try some prediction on my test data set.

- **Support Vector Machine(SVM):** I will use an SVC classifier from sklearn.svm.SVC with 'rbf' kernel to train and predict on my dataset.
- **Random Forest:** A random forest is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and use averaging to improve the predictive accuracy and control over-fitting.
- **xgboost:** XGBoost is an algorithm that has recently been dominating applied machine learning and Kaggle competitions for structured or tabular data. XGBoost is an implementation of gradient boosted decision trees designed for speed and performance. XGBoost stands for eXtreme Gradient Boosting.

As Tianqi Chen said:

> The name xgboost, though, actually refers to the engineering goal to push the limit of computations resources for boosted tree algorithms. Which is the reason why many people use xgboost

Generally, XGBoost is fast. Really fast when compared to other implementations of gradient boosting.

Benchmark

I will use an untuned Logistic Regression classifier from 'sklearn' to benchmark my final result. When the outcome (dependent variable) has only a limited number of possible values (in our cases hometeam win or loss), Logistic Regression is used when the response variable is categorical in nature.

III. Methodology

Feature Set Explanation

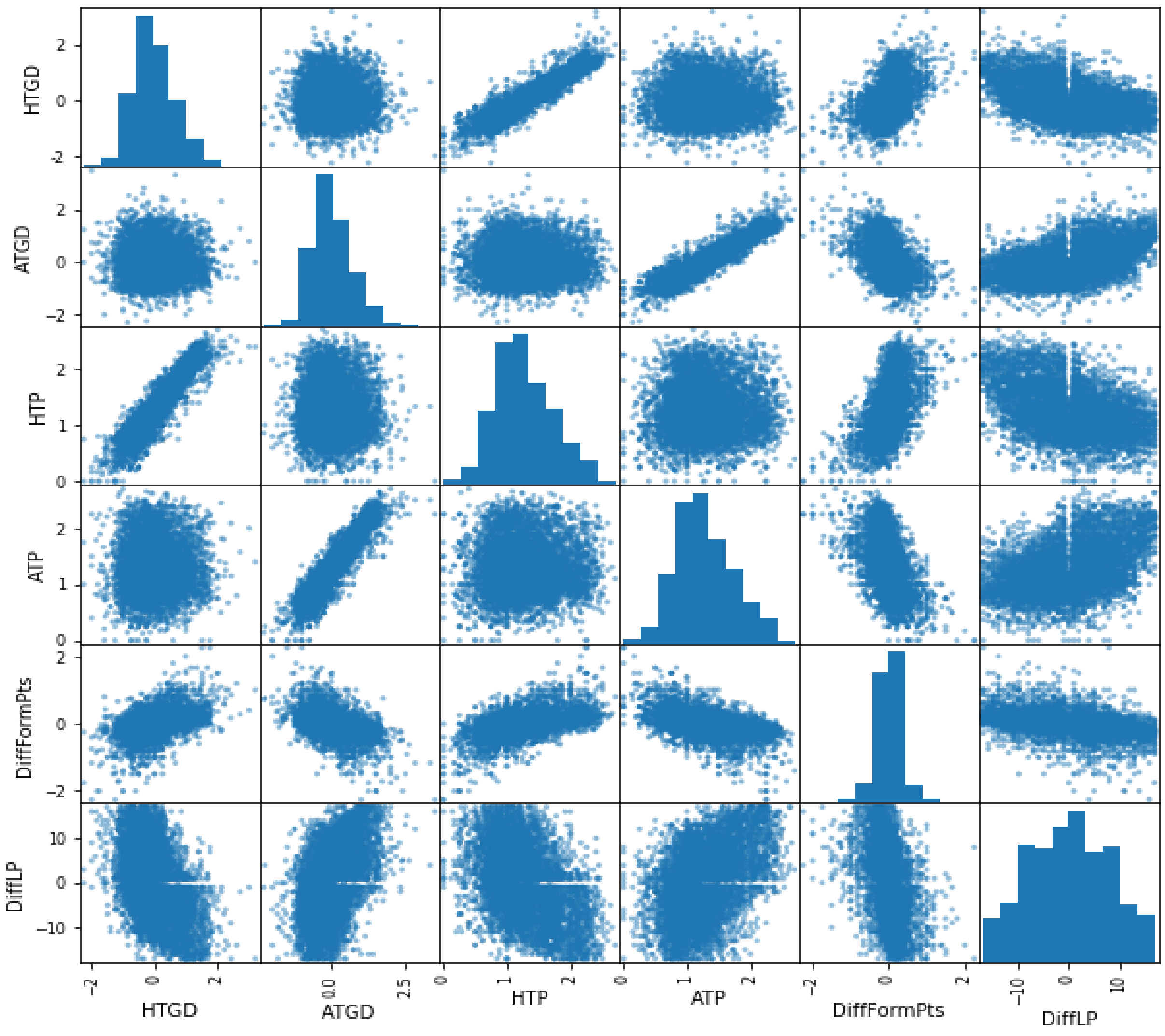
- FTR: Full Time Result (H=Home Win, D=Draw, A=Away Win)
- HTP - Home team points
- ATP - Away team points
- HTGD - Home team goal difference
- ATGD - away team goal difference
- DiffFormPts - Difference in points
- DiffLP - Difference in last years prediction
- HM - Home Match
- AM - Away Match

Let's explore this data a little bit

- Total number of matches: 5600
- Number of features: 12
- Number of matches won by home team: 2603
- Win rate of home team: 46.48%

Visualizing distribution of data

The scatter matrix is plotting each of the columns specified against each other column. You would have observed that the diagonal graph is defined as a histogram, which means that in the section of the plot matrix where the variable is against itself, a histogram is plotted. Scatter plots show how much one variable is affected by another. The relationship between two variables is called their correlation negative vs positive correlation.



Data Preprocessing:

- Here the data is prepared by Separate into feature set and the target variable FTR = Full-Time Result (H=Home Win, D=Draw, A=Away Win)
- I have preprocessed the football data and converts categorical variables into dummy variables

Let's see the output visualization:

```
Processed feature columns (24 total features):
['HTP', 'ATP', 'HM1_D', 'HM1_L', 'HM1_W', 'HM2_D', 'HM2_L',
'HM2_W', 'HM3_D', 'HM3_L', 'HM3_W', 'AM1_D', 'AM1_L', 'AM1_W',
'AM2_D', 'AM2_L', 'AM2_W', 'AM3_D', 'AM3_L', 'AM3_W', 'HTGD',
'ATGD', 'DiffFormPts', 'DiffLP']
```

Implementation: Initial Model Evaluation

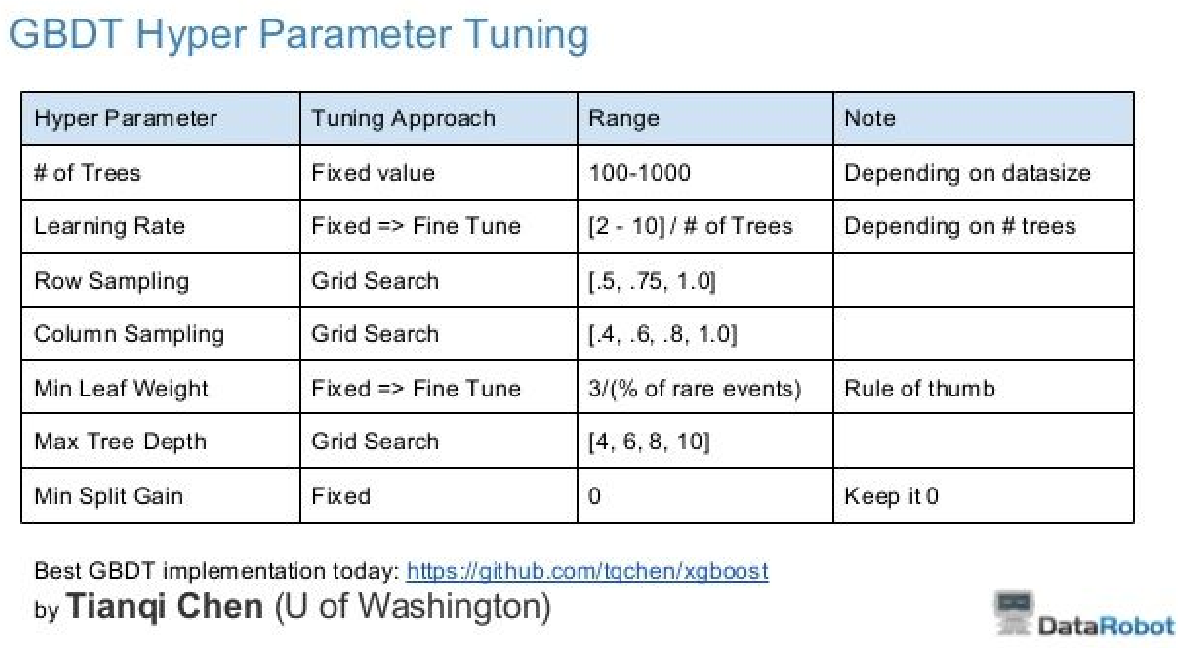
Training a LogisticRegression using a training set size of 5550. . . Trained model in 0.0260 seconds Made predictions in 0.0040 seconds. 0.6215610352557571 0.6654054054054054 F1 score and accuracy score for training set: 0.6216 , 0.6654. Made predictions in 0.0000 seconds. F1 score and accuracy score for test set: 0.6957 , 0.7200.	Training a SVC using a training set size of 5550. . . Trained model in 1.8139 seconds Made predictions in 1.0184 seconds. 0.6204535729567822 0.6803603603603604 F1 score and accuracy score for training set: 0.6205 , 0.6804. Made predictions in 0.0110 seconds. F1 score and accuracy score for test set: 0.6818 , 0.7200.	Training a RandomForestClassifier using a training set size of 5550. . . Trained model in 0.0799 seconds Made predictions in 0.0120 seconds. 0.9853789919199692 0.9863063063063063 F1 score and accuracy score for training set: 0.9854 , 0.9863. Made predictions in 0.0020 seconds. F1 score and accuracy score for test set: 0.6923 , 0.6800.	Training a XGBClassifier using a training set size of 5550. . . Trained model in 0.4477 seconds Made predictions in 0.0220 seconds. 0.6521471132114238 0.694954954954955 F1 score and accuracy score for training set: 0.6521 , 0.6950. Made predictions in 0.0020 seconds. F1 score and accuracy score for test set: 0.7451 , 0.7400.

Refinement

- From above result my benchmark model *Logistic Regression's* f1 score & accuracy score on test set is 0.6957 , 0.7200 respectively.
- Even though Random Forest performed very well on training data but failed on test data
- It is clearly visible from the above result that XgBoost is the best model for this problem.

IV. Results

Tuning the parameters of XGBoost



GBDT Hyper Parameter Tuning

Hyper Parameter	Tuning Approach	Range	Note
# of Trees	Fixed value	100-1000	Depending on datasize
Learning Rate	Fixed => Fine Tune	[2 - 10] / # of Trees	Depending on # trees
Row Sampling	Grid Search	[.5, .75, 1.0]	
Column Sampling	Grid Search	[.4, .6, .8, 1.0]	
Min Leaf Weight	Fixed => Fine Tune	3/(% of rare events)	Rule of thumb
Max Tree Depth	Grid Search	[4, 6, 8, 10]	
Min Split Gain	Fixed	0	Keep it 0

Best GBDT implementation today: https://github.com/tqchen/xgboost
by **Tianqi Chen** (U of Washington)

Below is the output of the optimized XGBoost Classifier:

```
XGBClassifier(base_score=0.5, colsample_bylevel=1,
colsample_bytree=0.8,
       gamma=0.4, learning_rate=0.1, max_delta_step=0,
max_depth=3,
       min_child_weight=3, missing=None, n_estimators=40,
nthread=-1,
       objective='binary:logistic', reg_alpha=1e-05,
reg_lambda=1,
       scale_pos_weight=1, seed=2, silent=True, subsample=0.8)
Made predictions in 0.0150 seconds.
F1 score and accuracy score for training set: 0.6365 , 0.6827.
Made predictions in 0.0000 seconds.
F1 score and accuracy score for test set: 0.7826 , 0.8000.
```

Final Model Evaluation

Let's find out the difference between optimized & unoptimized metrics of my final XGBoost classifier on the test set

Metric	Unoptimized Model	Optimized Model
Accuracy Score	0.74	0.80
F-Score	0.74	0.78

My **benchmark model *Logistic Regression*** f_score & Accuracy Score was 0.6957, 0.7200 respectively on the test set.

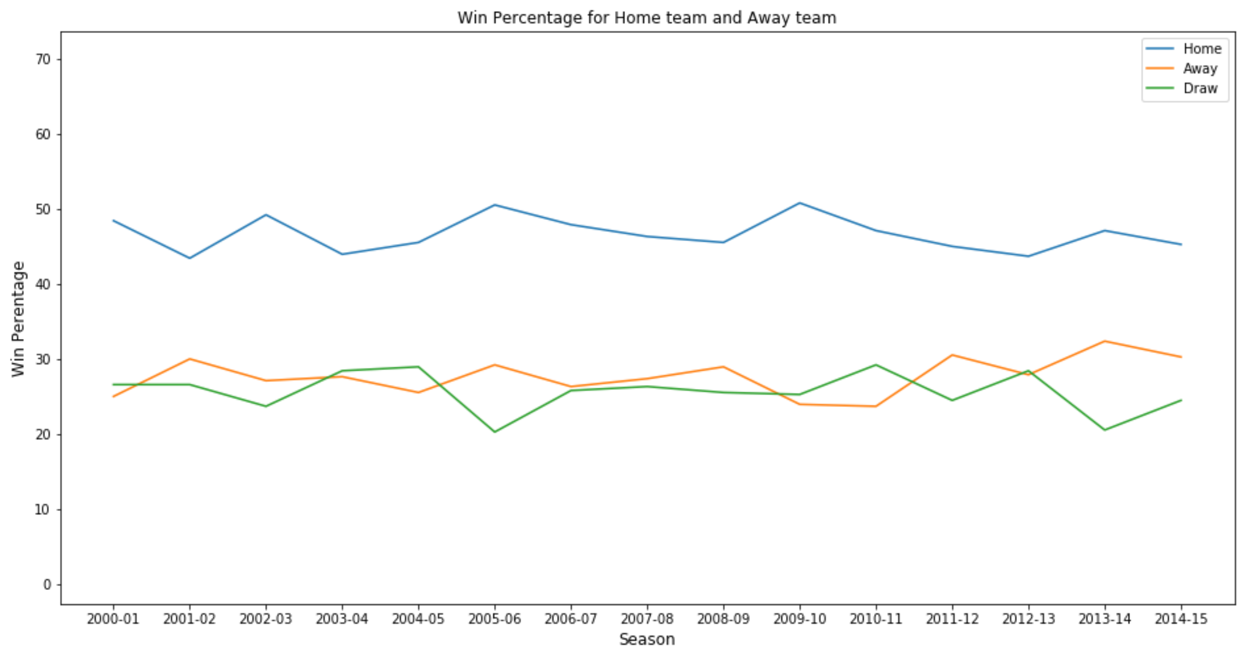
Justification

By using my final model we can predict the results of an EPL game which is statistically more significant than pure guessing the result of a match with human instinct.

Conclusion

Free Form Visualization

Let's visualize a line plot of the win percentages for different parameters for different seasons



From the above line plot also we can clearly see home team wins more in EPL while draws and losses are equally common for the home team.

Reflection

- In this project first I performed exploratory data analysis(EDA) with some simple visualization where it was clearly displayed that in EPL home team wins proportion is higher than draw & loss.
- I have chosen three classifiers to train & predict my data. Surprisingly Random Forest gave me a very high f1-score & accuracy score on training data but performed worst relatively on testing data. This is a very common scenario of *high train score, low test score*. RandomForest learned well from the training data but didn't perform well on testing data as compared tp XGBoost classifier. So I choose XGBoost as my final classifier.
- Pure guessing also can give some good result while predicting match results, but using my final trained model is more statistically significant for better prediction.

Improvement

- Possible improvement can be a web app taking the input of the two teams & predicting the winner based on the model trained above.
- We can implement more data features like twitter sentiment analysis on pre-match analysis by training a deep neural network. For this purpose, we can use millions of data points affecting a football match result. In FIFA 2018 world cup bing tried a similar thing by predicting some matches result accurately.
- Possibly a neural network with a much bigger feature rich dataset can predict more accurately than my final model.