EPL Game Result Prediction

Bikash Shrestha

December 8, 2019

Contents

List of Tables

List of Figures

1 Introduction

Soccer (European Football) is one of the most popular sports in the world. Predicting the results of soccer matches are very difficult because the result of the match is dependent on many factors, such as a team's morale, skills, current score, players form, coaching, training, etc. Even for football experts, it is very hard to predict the exact results of football matches. But predicting the match results is interesting to many fans and also for the Punters. As a soccer fan myself, I am curious how Artificial Intelligence can be applied to predict the result of the matches. In this project I will try to implement neural network model to predict a result of the match between two teams.

2 Dataset

The dataset was obtained from Kaggle Data Science website called the ”EPLDataset” [2]. This dataset has been made publicly available. It contains the statistics for every match played from season 06/07 to 18/19 in the EPL (English Premier League) with statistics provided from the official EPL website. It has the statistics of each match for example goals, possessions, shots on target, total shots, touches, passes, tackles, clearances, corners, offsides, yellow cards, red cards, fouls conceded and so on.

2.1 Dataset Description

There are 4938 rows and 31 columns. So, it contains 4938 matches of the EPL teams. Among 31 columns, I will extract 12 columns as my features and add 1 more columns for the label field. The label will can 0, 1 or 2. ”1” for Home Team Win, ”0” for Draw and ”2” for Away Team Win. So my dataset will consist of 4938 rows and 13 columns. The input features are for the following fields:

- Home team possession
- Away team possession
- Home team shots on target
- Away team shots on target
- Total No. of passes of the ball for the home team
- Total No. of passes of the ball for the away team
- Total No. of tackles for the home team
- Total No. of tackles for the away team
- Total No. of corners for the home team
- Total No. of corners for the away team
- Total No. of red cards for the home team
- Total No. of red cards for the away team

2.2 Input Data Visualization

The histogram plot of every input features showing their maximum and minimum value as well as how they are distributed can be seen in the images given below.

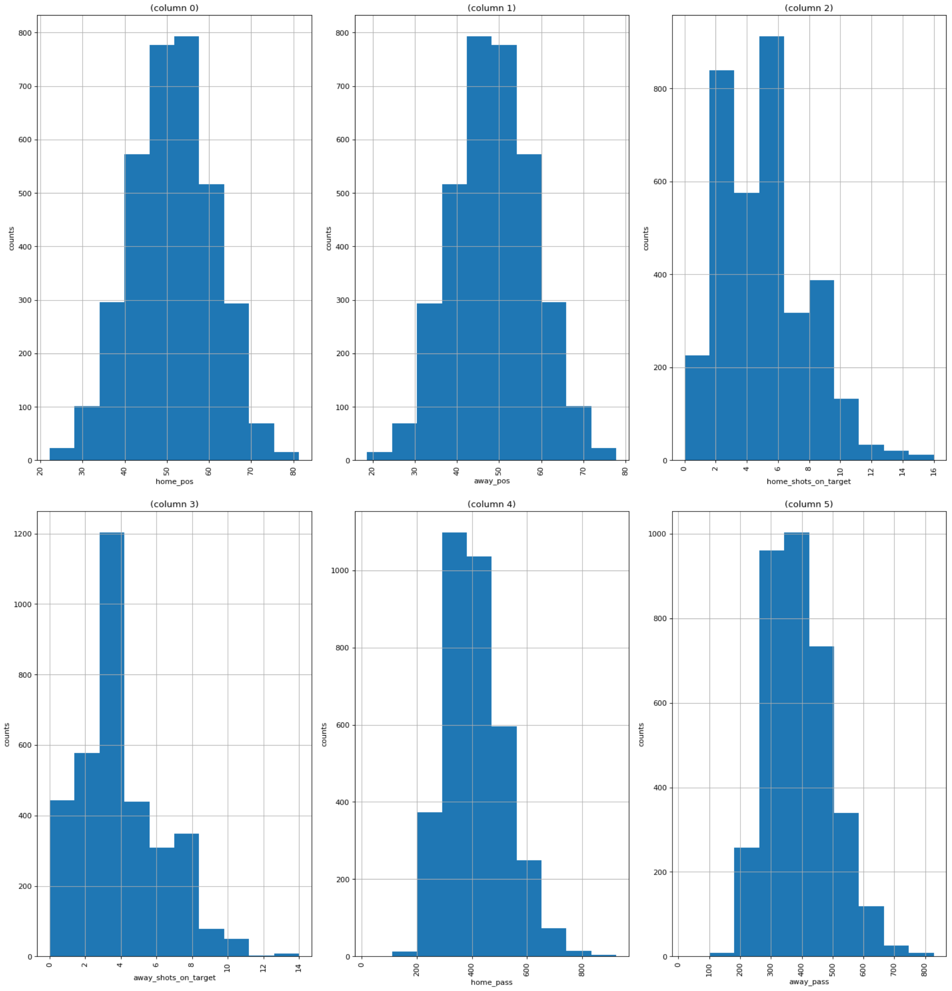

Figure 1: Input Data Distribution Histograms (Part One)

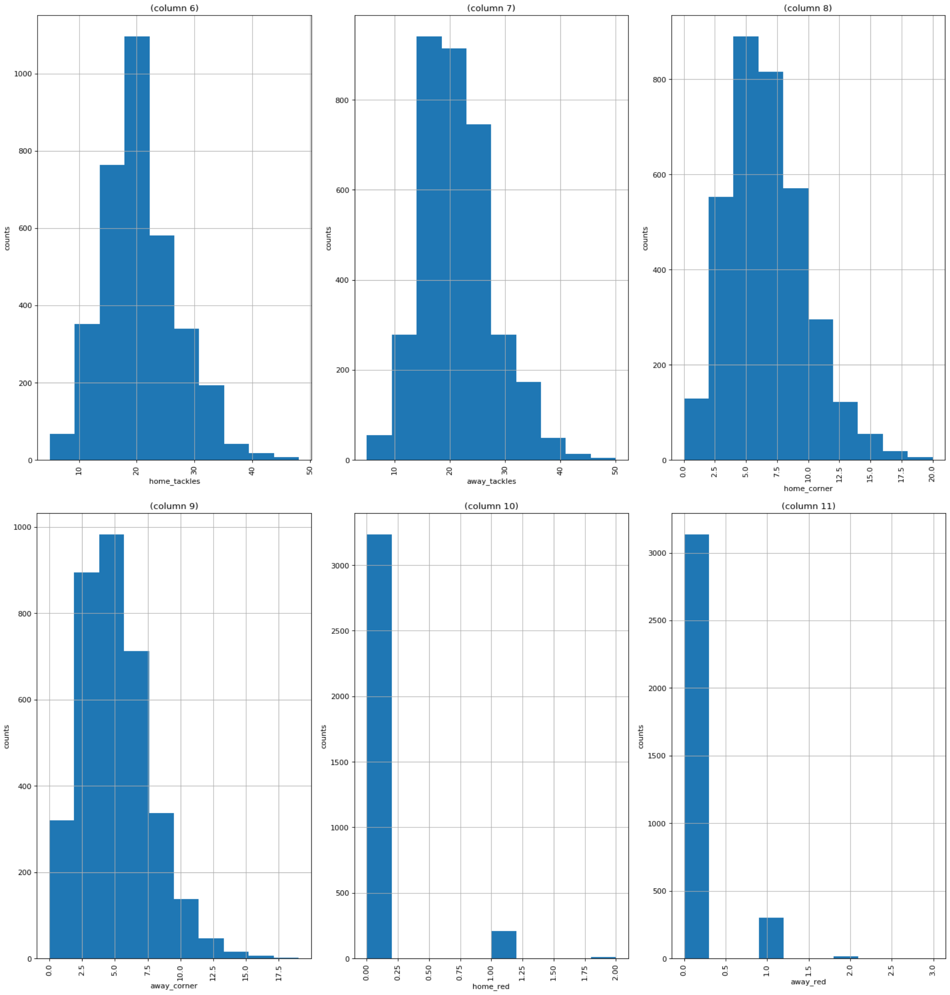

Figure 2: Input Data Distribution Histograms (Part Two)

Table 1: Input Feature Statistics

	Min	Max	Mean	Std
home pos	22.2	81.3	51.348785	9.562747
away pos	18.7	77.8	48.651215	9.562747
home shots on target	0.0	16.0	5.016204	2.645209
away shots on target	0.0	14.0	3.942998	2.291858
home team passes	19.0	924.0	413.162326	108.280811
away team passes	20.0	828.0	392.862558	101.632971
home team tackle	5.0	48.0	20.583044	6.402427
away team tackle	5.0	50.0	21.079572	6.382935
home team corners	0.0	20.0	06.201389	3.118955
away team corners	0.0	19.0	4.828993	2.773971
home team red	0.0	2.0	0.066262	0.259034
away team red	0.0	3.0	0.097512	0.314686

2.3 Output Data Visualization

Here is the distribution of output labels.

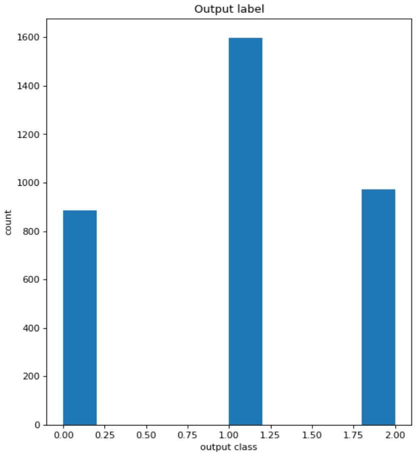

Figure 3: Output Data Distribution Histogram

3 Data Processing

3.1 Data Splitting

First of all, the data was randomly shuffled and then the dataset was split into training, validation and test. At first 70% of the dataset was allocated for training, 20% was allocated for validation and 10% for testing. But later it was changed to 60%, 30% and 10% respectively.

3.2 Data Normalization

As we can see the data was not distributed uniformly. Therefore we need to pre-process the data with normalization technique. At first Min-Max normalization because it guarantees all features will have the exact same scale although it does not handle outliers well [5]. The formula to calculate min-max value is as follow.

$$\frac{value - min}{max - min}$$

But later it was changed to Z-score normalization which increases the accuracy. There might be outliers in the training dataset. The formula to calculate Z-score is as follow.

$$\frac{value - mean}{sd}$$

4 Data Analysis

After normalizing the input data, the output data has to be preprocessed to convert them into right format. One Hot Encoding method was used to convert the output labels into a 1-dimensional numerical vector. The resulting vector has only one element equal to 1 and the rest will be 0. Since there are three output labels i.e 0 for Draw, 1 for Home Win and 2 for Away Win so the vector will contain 3 elements. So the output vector looks like [1, 0, 0] for Draw, [0, 1, 0] for Home Win and [0, 0, 1] for Away win.

4.1 Relationship between input feature and output

I plotted the stacked histogram to show the correlation of each input feature with the outputs. The figure below shows correlation of shots on target with output.

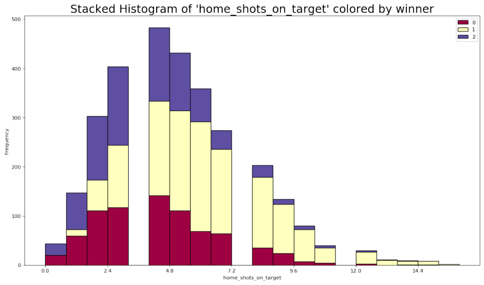

Figure 4: Stacked Histogram for home shots on target vs output frequency

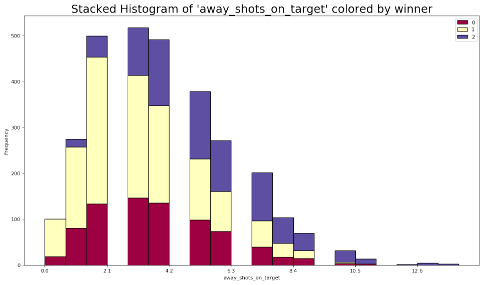

Figure 5: Stacked Histogram for away shots on target vs output frequency

Here in Fig 4, we can see that there is increase in the frequency of Home wins (i.e 0) as the home shots on target is increasing and when there is very high shots on target (i.e greater than 12) there is only Home win no Away win. The same can be seen in Fig 5 where there is increase in the frequency of Away wins as the away shots on target is increasing.

The figures given below are for other input features.

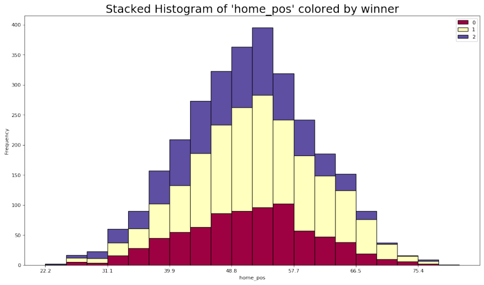

Figure 6: Stacked Histogram for home possession vs output frequency

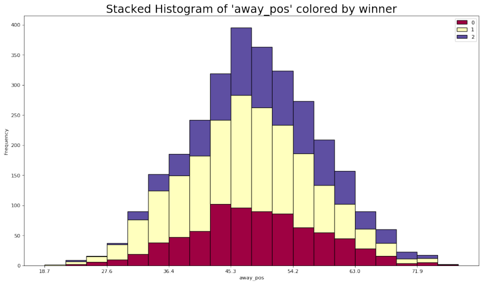

Figure 7: Stacked Histogram for away possession vs output frequency

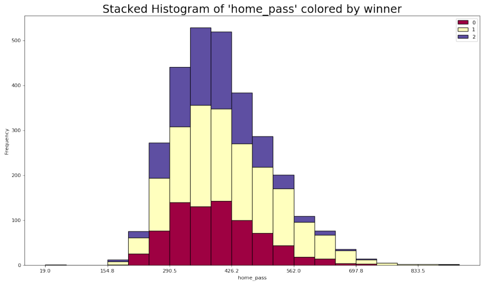

Figure 8: Stacked Histogram for home pass vs output frequency

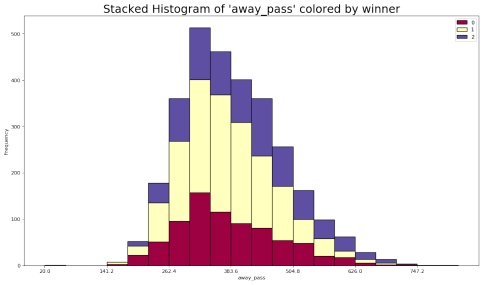

Figure 9: Stacked Histogram for away pass vs output frequency

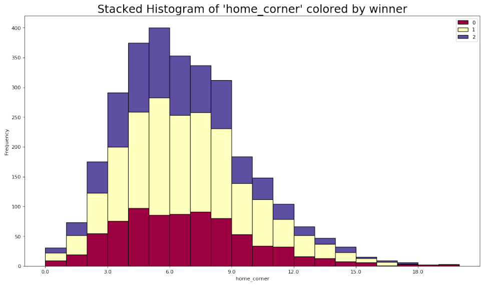

Figure 10: Stacked Histogram for home corner vs output frequency

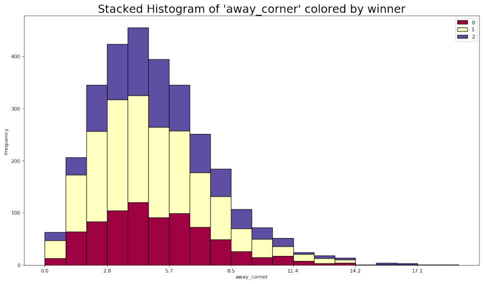

Figure 11: Stacked Histogram for away corner vs output frequency

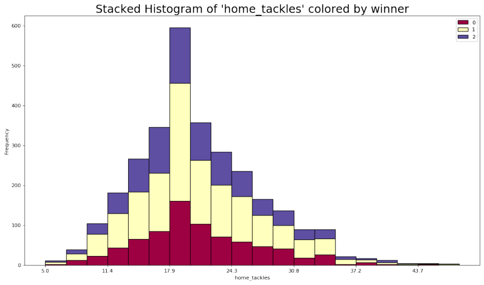

Figure 12: Stacked Histogram for home tackles vs output frequency

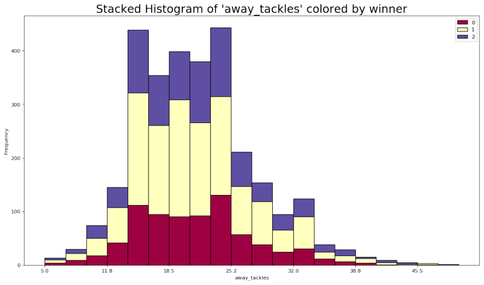

Figure 13: Stacked Histogram for away tackles vs output frequency

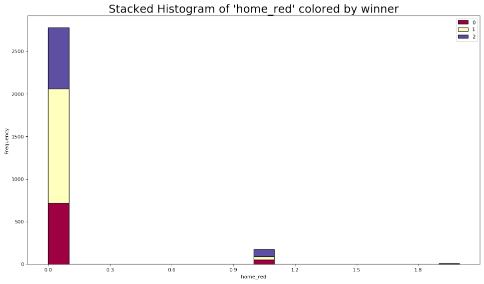

Figure 14: Stacked Histogram for home red card vs output frequency

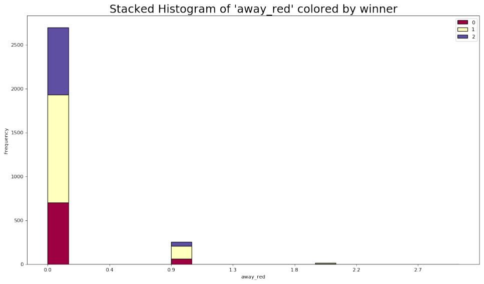

Figure 15: Stacked Histogram for away red card vs output frequency

5 Modelling

Artificial Neural Network (ANN) was used to create the model. I used a feed forward neural network.

5.1 Varying Neural Network Architecture

First I tried with basic architecture with one input layer, one hidden layer and one output layer. Later I increased the hidden layer from one to two and three. But there was no significant improvement even when increasing the hidden layers.

5.1.1 Performance comparison

The table given below shows the accuracy percentage for training, validation and test sets for varying hidden layers.

Table 2: Performance comparison for varying hidden layers

	Training Acc	Validation Acc	Test Acc
One layer	63.0105 %	60.7698 %	58.7045 %
Two layer	64.0904 %	58.2714 %	56.8826 %
Three layer	64.0904 %	59.8920 %	55.2632 %

As we can see from above table that the basic architecture with just one hidden layer in overall is performing better than other architecture with multiple hidden layer for all dataset.

5.2 Changing Activation function

The linear activation function performed pretty worse in comparison with sigmoid activation function. I also tried ”softmax” activation function which is generally used for multi-class classification [3]. I used these activation function to the final output layer and the performance comparison can be seen in the table shown below.

5.2.1 Performance comparison

Table 3: Performance comparison for different activation function in output layer

	Training Acc	Validation	Test Acc
Softmax	64.7317 %	58.4740 %	58.5020 %
Sigmoid	63.0105 %	60.7698 %	58.7045 %
Linear	25.4134 %	25.8609 %	24.6963 %

The table shown below shows the performance when changing activation function to tanh of just hidden layers and also of all layers except output.

Table 4: Performance comparison for tanh activation function for different layers

	Training Acc	Validation Acc	Test Acc
Only Hidden layers	65.3392 %	59.4868 %	59.1093 %
All layer except output	64.0904 %	60.0945 %	56.0729 %

5.3 Learning Curve of Neural Network

The given below images show the learning curve for the Neural Network model.

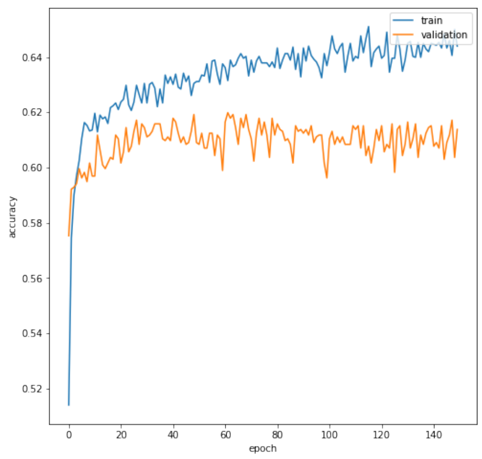

Figure 16: Curve showing change in accuracy vs epoch

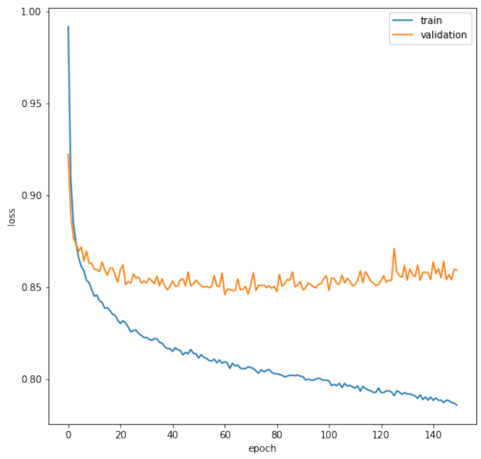

Figure 17: Curve showing change in loss vs epoch

5.4 Logistic Regression Model

Later I also tried Logistic Regression Model and it performed pretty well in comparison with our Neural Network Model. It performed even better than neural network model in the validation and test data sets.

The table given below shows the performance of the logistic regression model.

Table 5: Performance of Logistic Regression Model

Training Acc	Validation Acc	Test Acc
60.7155 %	61.5125 %	59.9190 %

The given below images show the learning curve for the logistic regression model.

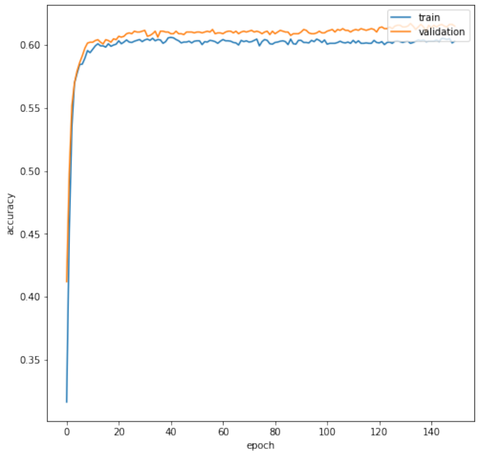

Figure 18: Curve showing change in accuracy vs epoch

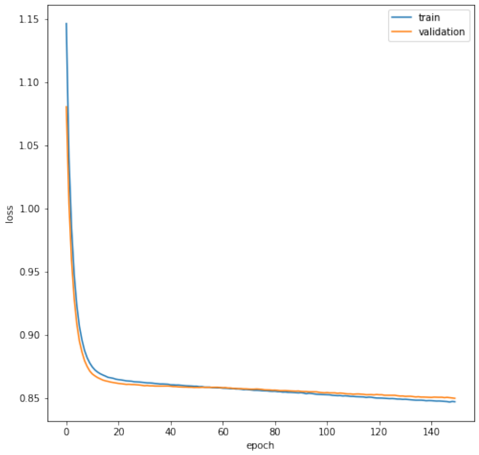

Figure 19: Curve showing change in loss vs epoch

6 Model Evaluation

Since, it is the classification, I used confusion matrix to evaluate the model [1]. I used three essential classification model metrics to evaluate:

- Precision
- Recall
- F1 score

The table given below shows the precision, recall and f1 score for the neural network model for validation and test data sets.

Table 6: Neural Network Model evaluation using precision, recall and f1 score

	Precision	Recall	F1-score
Validation	67.3449 %	46.9277 %	0.5531
Test	65.3333 %	48.7854 %	0.5677

Table 7: Logistic Regression Model evaluation using precision, recall and f1 score

	Precision	Recall	F1-score
Validation	62.2435 %	30.7225 %	0.4113
Test	61.6071 %	27.9352 %	0.3844

6.1 Comparison between Custom prediction function and Keras prediction function

I build my own custom function to predict the outputs. I extracted the weights from the trained model and used them on my custom function to predict the outputs. The table given below shows the outputs comparison.

Table 8: Custom prediction vs keras prediction

Custom			Keras		
Output1	Output2	Output3	Output1	Output2	Output3
0.24506184	0.23038296	0.5245552	0.24506187	0.23038308	0.5245551
0.12145221	0.84463965	0.03390814	0.1214522	0.8446396	0.03390815
0.17067145	0.06975986	0.75956869	0.17067145	0.06975986	0.7595687
0.01994902	0.97423905	0.00581193	0.01994902	0.9742391	0.00581194
0.24237966	0.26135145	0.49626889	0.24237967	0.26135138	0.49626896

7 Feature Importance Analysis

By looking in the Fig 12 and Fig 13, we can see that home tackles and away tackles do not show significant relationships with the output so I tried removing them from the input feature sets. And there was no significant drop in the performance after removing them. We can see the table below for the performance.

Table 9: Performance after removing home_tackle and away_tackle from the input dataset

	Training Acc	Validation Acc	Test Acc
Two features removed	62.4705 %	60.1621 %	58.0972 %

Two more features (away_pass and home_pass) were removed from the input sets which reduces the performance of the model. There was a noticeable drop in the accuracy of training set but there was no significant drop in validation and test sets as compared to the above case. Here we can see the performance in the given table below.

Table 10: Performance after removing home_pass and away_pass from the input dataset

	Training Acc	Validation Acc	Test Acc
Four features removed	59.8380 %	58.6090 %	59.5141 %

8 Challenges Faced

I faced some difficulty while converting the output to the right format. Since the actual output was categorical (i.e 0, 1 and 2) but the output from the model will be in between 0 and 1, I have to convert the actual output in that range. So for that I have to use One Hot

Encoding method.

I also faced some challenges while finding relationships between input and output. I have to find suitable visualization technique which shows the relationship between input features and output.

9 Code Access

All the results here can be reproduced by using the google colab notebook. Here's the link to the notebook: AI Project Notebook

10 Future Improvement

Performance may be further improved by following ways [4].

- Cross Validation
- Using Ensemble Method
- Adding player ratings or team ratings in the input features

11 Conclusion

In this project, I developed a neural network model to predict the result of the English Premier League soccer match using match statistics. Here I tested different activation functions and their effects on the performance of the model. It was found that tanh activation function performed better than other activation functions in the hidden layers. I also tested the importance of each feature in the performance of the model by finding the relationships between input features and output. Shots on target, possession and red cards are the important features for better performance of the model.

References

[1] Aleksey Bilogur. Evaluating Keras neural network performance using Yellowbrick visualizations, 2019.

[2] Englader. EPLDataset, 2019.

[3] Pius Gadosey. A beginner's guide to NumPy with Sigmoid, ReLu and Softmax activation functions, 2019.

[4] Rohith Gandhi. Improving the Performance of a Neural Network, 2018.

[5] Timo Stöttner. Why Data should be Normalized before Training a Neural Network, 2019.