**Europe Soccer Match Outcome Prediction**

Joo Yeong Cho

University of Kyung Hee

Whwndud92@khu.ac.kr

**Abstract**

**People always imagine getting a jackpot without labor, and most people do it. In addition, we have always endeavored to maximize the best performance and convenience by introducing human technology into it. Recently, Deep Learning has been developed to show the challenge of human limitations, and a large number of data sets and information are flooded on the Internet to make various attempts through simple models. The development of the European football market and the media is rapidly expanding and the interest of the people is increasing. As a result, a lot of people will pay attention to every game of European football. There are also a lot of game data. I use Supervised Learning to predict the victory, draw and defeat of the European football game, and to predict the final league rankings based on the predicted results of each team.**

**1. Introduction**

With the exception of the summer season, the European football game is played almost 20 times each week, with each home and away team playing twice in each of the remaining leagues.

Thus, there are 380 matches in one season, and each game has a betting odds for home team win, draw and away team win at multiple betting sites. If gamer betting on the outcome of several matches, the dividend rate of the results is multiplied by the dividend, and then proceeds to the gamer.

in this paper, I'm going to show you the process of running Supervised Learning along with more than 5,000 match data and athlete data in several representative leagues, making actual bets based on it and forecasting the final league rankings.

Learning was largely divided into two categories: data with only payout ratio and data with dividend rate and soccer players data.

Here, the team information data is too abstract and inaccurate, and it is expected that it will cause damage to the prediction of the game.

**2. Data Set**

On the Kaggle.com homepage, I obtained a data set in which the European football game results, players' information, and team information were uploaded in the form of a relational database.

The approximate schema of the database is as follows.

Player : (player ID, player name, date of birth, height, etc ...)

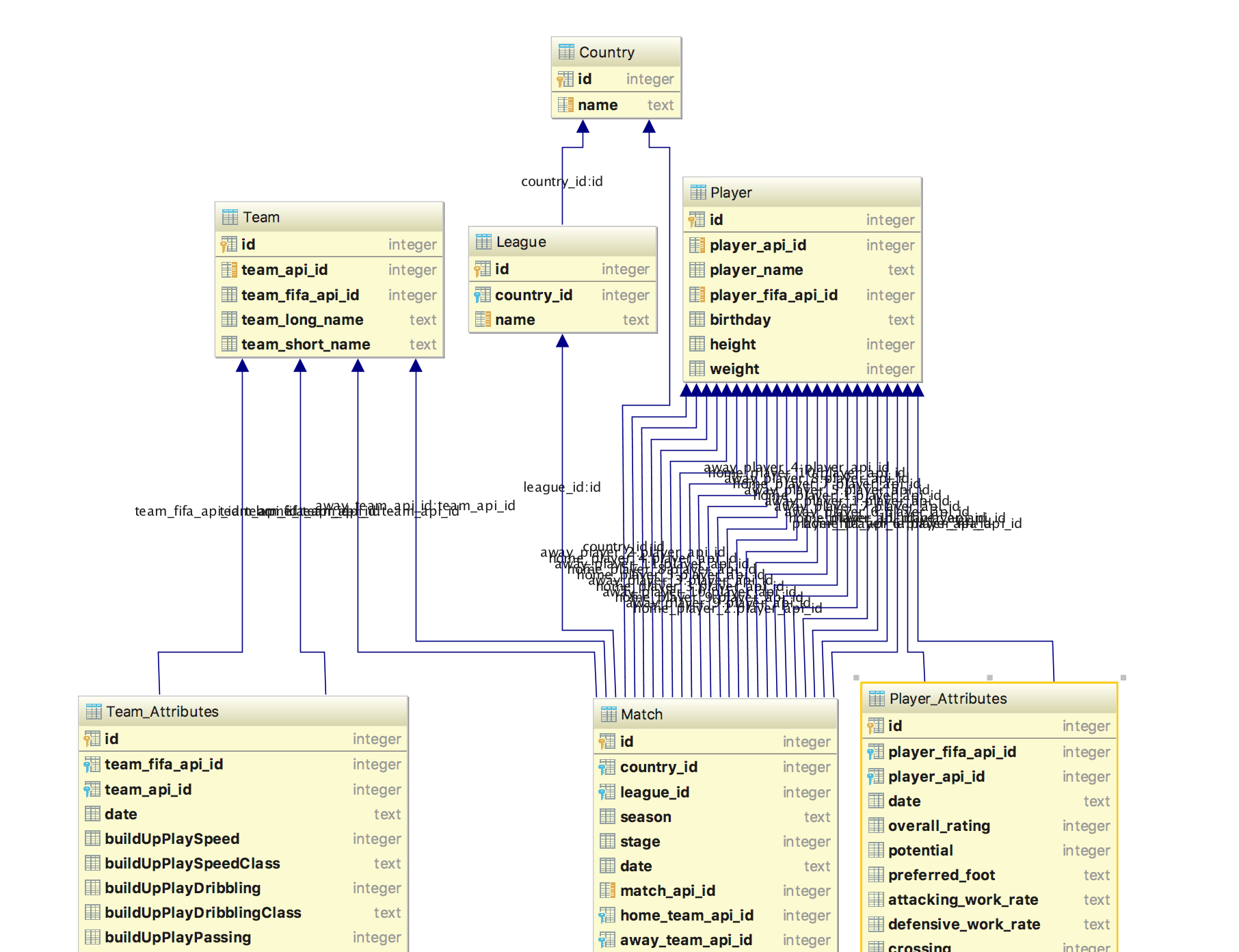
player attribute (player ID, date of ability measurement, total ability, running, shooting etc ...)

Team (Team ID, Team Name)

Team attributes (team ID, date the team measured the ability, attack, buildup, defense etc ...)

Match (Match ID, home team, away team, season, match result, league ID, main player list, betting rate of several betting sites)

Country (country name, affiliation league name)

League (league ID, country name, league name)

The diagram of the provided database is as above.

**3. Data processiong**

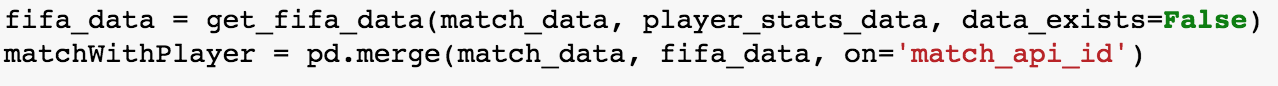
All of the data preprocessing work was done by connecting to a locally stored sqlite dataset, converting it to memory, using python's pandas and numpy libraries. Then, the relational data was classified for each purpose and stored in a flat file of csv format.

3.1 Select Data / Feature Selection

Once I did not get all the property information stored in the match table to reduce memory requirements and calculation execution time and improve performance.

Only the odds of wins, draws, and losses from the six most influential betting sites were brought using queries. We have already concluded that the results of the analysis by many people have already been mixed in the dividend rate.

In addition, player attributes that match the player lineup of match data were also brought in to use 22 player information.

Here, we merge the attribute values of the two tables,

Using the get\_fifa\_data function, the total ability data of the 22 players measured at the most recent date on the date the match was held are fetched and merged with the current match\_data.

The two types of odds data and odds + player data prepared in this way were prepared as 5000 records, divided into 8 : 2 and stored as 4000 and 1000 train.csv and test.csv.

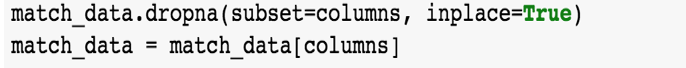
3.2 Preprocess Data

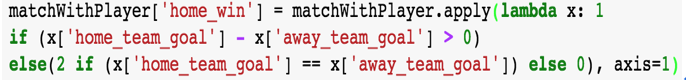
Because there are a lot of missing data in the data of the selected table, the record with the missing data is discarded. This is because we judged that there is enough data compared to the feature value.

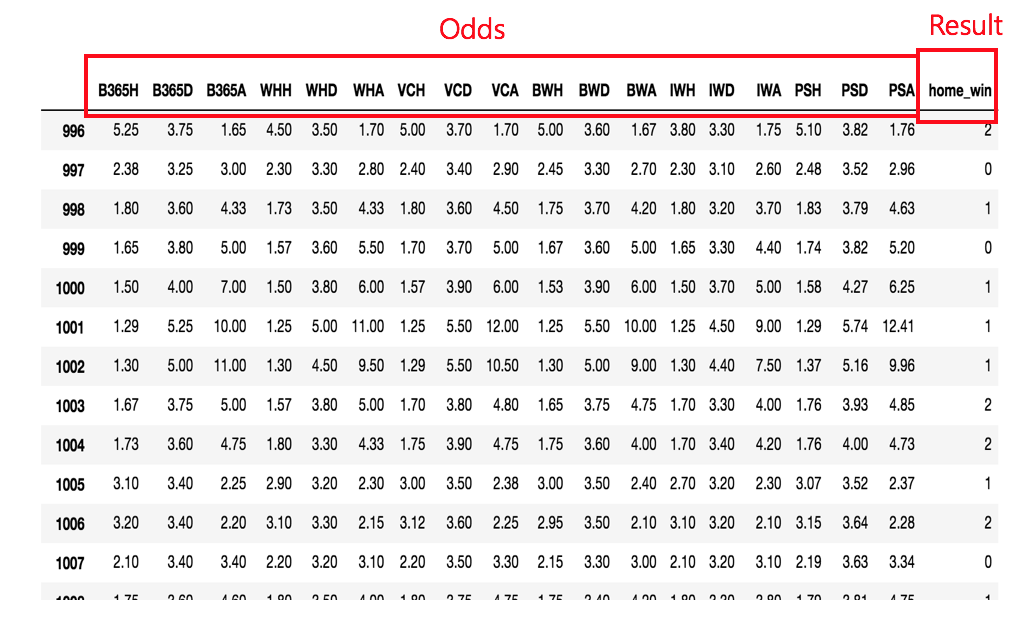
3.3 Transform Data

All data was transformed from 0 to 1 using the MinMaxScalar algorithm.

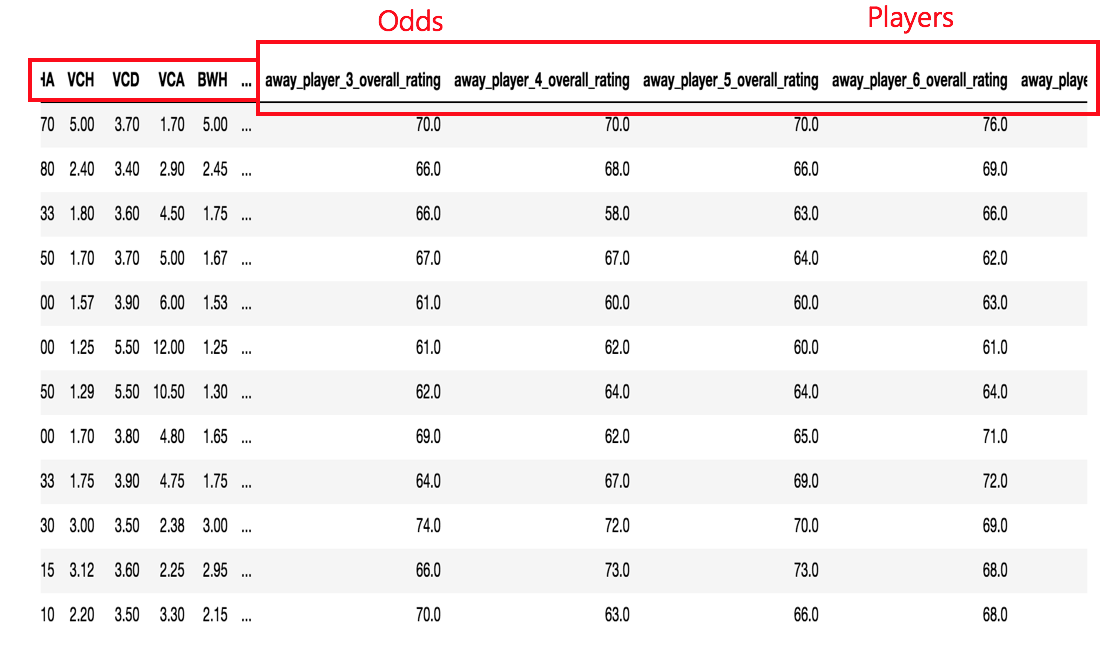
And, in the home win attribute corresponding to the result value, it transforms to 1 if the home team wins, draws 2, and 0 to the victory of the away team.







This is the home, draw, and away odds data for the six selected betting sites.

The Home win attribute is divided into 0, 1, and 2 classes.

The odds of six betting sites and the overalls of 22 players are stored.

**4. Single layer Logistic Classification**

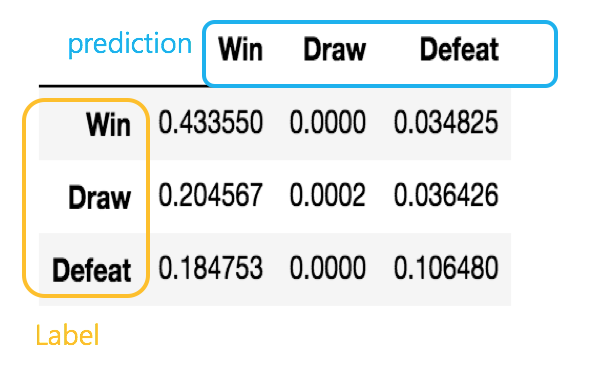
Basic multinomial logistic regression was trained using single layer and softmax functions.

For the 4000 training data and 1000 testing data divided by the ratio of 8 : 2, learning rate was set to 0.01 and 30,000 steps were learned.

And training / testing 5 times and measuring the mean value were performed to generalize the results that changed every training

4.1 For odds

Let's look at the results when training about 4000 match\_train\_odd.csv and 1000 match\_test\_odd.csv.



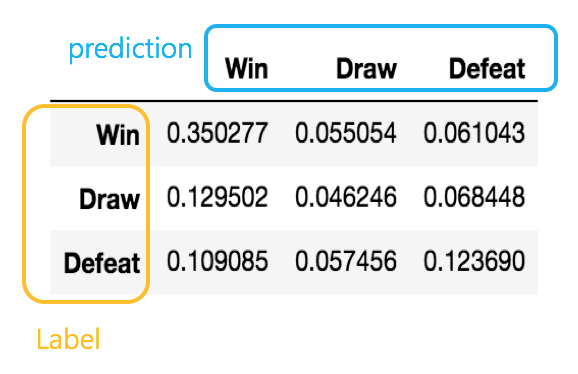
Based on the home team, calculate the result table for Win, Draw, and Defeat as shown above, so you can get the precision and recall.

Where each index is label and column is prediction.

In addition, the final accuracy of the total result is 53.98%.

4.2. For odds and players

The training results for 4000 match\_train\_odd\_player.csv and 1000 match\_test\_odd\_player.csv are as follows



The result table and the prediction / recall chart are shown as above, and the final accuracy of the total result is 51.98%.

**5. Multi-layer perceptron**

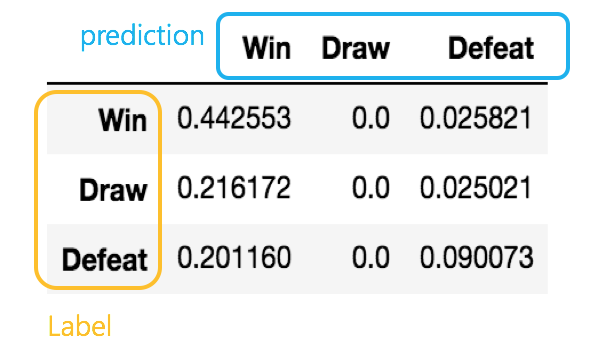
This time, instead of softmax, I used ReLu and dropout to stack layers and deeply learn match\_test\_odd.csv and match\_test\_odd\_player.csv.

The learning rate was 0.001, and the batchsize was 500 and 300 epochs. and then, to increase the width of the hidden layer, increase the value of the node to 500 and set it to the 9 Layer.

And dropout was used to reduce learning time and prevent overfitting.

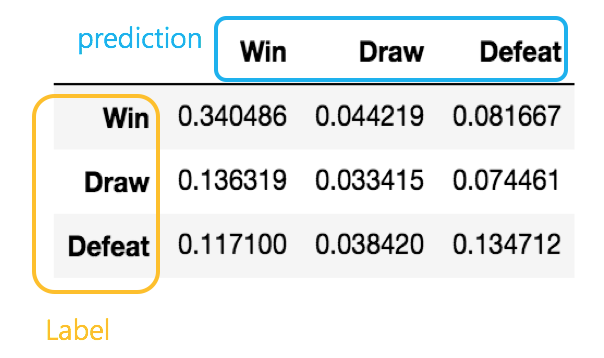
We also used AdamOptimizer and oneHot encoding the value of the class.

5.1 For Odds

The results are as follows

When we train about 300 epochs, we make the result table and chart as the cost changes at 1.0. The final accuracy is 53.22%.

5.2 For Odd and Players



When you include player data in the same way, record the above chart.

The final accuracy is 50.82%.

**6. Rank of League**

How will the rankings of league I enjoy this season be determined?

Based on the result of the victory prediction, we multiply each victory point to predict the final ranking.

In this experiment, using the single layer classfication and softmax functions with the highest final accuracy,

I have trained the 16/17 season game data of Spain's Lariga.

In order to reflect the characteristics of the league, we have trained only the odds of past season except 16/17 season. The results are as follows.



<Real Rank> <Predicted Rank>

As you can see above, ranking of the top and bottom teams are similar.

But for mid-level teams, you can see that the predictions are slightly missed.

You can also see that top and bottom teams are predicted to get too many or fewer points in comparison to reality.

**6. Result**

In the same way, we stacked five layers and skipped the learning process using the ReLu function. Learning rate and other parameters follow the multilayer perceptron method described above.

Now you can get the above chart from the results obtained from the experiment. The final accuracies were classified by input data and modeling, respectively.

The Y-axis shows the average final accuracy learned for each model and input data five times.

The results show that the Prediction and Recall of draws are significantly lower than those of Win and Defeat which are extreme results when training and testing are performed.

In reality, even if there is a large difference in dividend rate, there are a lot of draws while overturning and reversing due to various realistic variables. On the other hand, it seems that the model learned has difficulty in drawing out the result of the draw from the dividend rate. The extreme characteristics of the training model are also revealed in ranking predictions, Looking at the predicted leaderboards, you can see that the difference between the top and bottom points is much bigger than the actual leaderboard.

This shows that the model does not yet reflect the unusual outcome and reflects the power difference seen with the given data. And in the final accuracy of various learning models, we find that the results of learning odds only are higher than those of odds and player attributes.

That is, it is best to learn simple data through feature selection. And it can be seen that the predicted result gets worse as the model builds up the layer.

This also shows that learning with a complex model can lead to rather bad results and It may also be overfitting.

**7. Conclusion**

I learned the results of the European football 's win / loss through machine learning techniques, predicted the results, and tried to show the best predictability with the given data. The result is more than 50%, which means there are many realistic variables that can not be predicted.

It is because the soccer game shows that it is a sports that is difficult to predict.

That is, the learning model is confused because the y value is not consistent, rather than having a consistent y value for the set of x values. In other words, it can be concluded that it is difficult to expect high accuracy by predicting the victory of the football game. But I am confident that if I could gather real-time, sophisticated and critircal data, I could get better results.

**8. Reference**

kaggle home page : [www.kaggle.com](http://www.kaggle.com)

Europe soccer matchup data : <https://www.kaggle.com/hugomathien/soccer>

DeepLearningZeroToAll giithub page : <https://github.com/hunkim/DeepLearningZeroToAll>

Tensorflow Tutorials : <https://github.com/nlintz/TensorFlow-Tutorials>

Betting odds : [http://www.football-data.co.uk](http://www.football-data.co.uk/)

Machine learning youtube : https://hunkim.github.io/ml