



Course CEN 359: Introduction to Machine Learning

## **Predicting Soccer Games Results with Logistic Regression**

**Student:** Emerald Podbićanin

**Student ID:** 18001732

Sarajevo, 2020

## **ABSTRACT**

I took this topic without any hesitation because I love and live sports and not only that but I really enjoy predicting outcomes of games and that is why I felt this project was a perfect choice for me, personally. I wanted this scientific approach to help me see things in a more professional manner and it did. A machine learning algorithm I used is Logistic Regression because it can classify outcomes and fits perfectly fine with my project.

## **INTRODUCTION**

As we all know, in every sports game there are mainly two possible outcomes. You either win or you lose. In some sports of course there could be a draw as a possible outcome, as well. Predicting an outcome in the real world can bring you money in a legal but an infamous way, and that is betting. Of course, by this I don't want to suggest anyone to bet but I wanted to state that if I was able to write a perfect algorithm for predicting results, then betting companies would need to exclude real life sports from their betting offers and that would stop a lot of people from losing money.

My motivation for this project is to challenge myself and try my best to do serve this topic in the best possible way. I hope that I will make an algorithm that would be able to predict outcomes in more than 70% of the time, but we will see.

The input data will be related to two teams and the efficiency in their last 50-100 games, such as „Home Team Goals Scored“ and „Home Team Goals Conceded“ as an example. The output will be a prediction that will try to guess the winner in a match between two teams, „Home Win“ or „Away Win“

## **RELATED WORK**

All related work for this particular topic is really similar. Everyone picked logistic regression for prediction because it can classify two outcomes and honestly I think there is no need for another method when this one fits fine. Also, the input and output data can't really be that much different because the topic is concrete and it is obvious, I would say, that everybody picks the team stats as input and win/lose as output.

- [https://www.researchgate.net/publication/312037695\\_Predicting\\_football\\_match\\_results\\_with\\_logistic\\_regression](https://www.researchgate.net/publication/312037695_Predicting_football_match_results_with_logistic_regression)
- <https://www.imperial.ac.uk/media/imperial-college/faculty-of-engineering/computing/public/1718-ug-projects/Corentin-Herbinet-Using-Machine-Learning-techniques-to-predict-the-outcome-of-professional-football-matches.pdf>
- <https://scholars.unh.edu/cgi/viewcontent.cgi?article=1472&context=honors>

## **METHOD**

The method I used for this project is Logistic Regression. Logistic regression is a classification algorithm used to assign observations to a discrete set of classes. Unlike linear regression which outputs continuous number values, logistic regression transforms its output using the logistic sigmoid function to return a probability value which can then be mapped to two or more discrete classes.

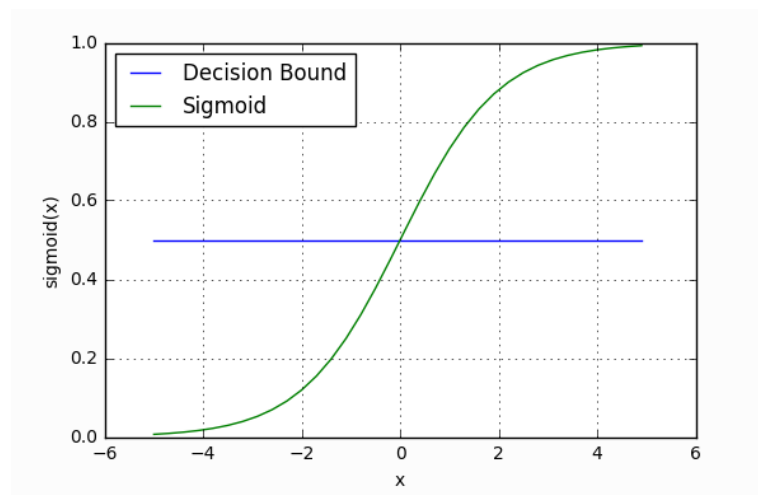
There are three types of logistic regression:

- Binary (Pass/Fail)
- Multi (Cats, Dogs, Sheep)
- Ordinal (Low, Medium, High)

In order to map predicted values to probabilities, we use the **sigmoid** function. The function maps any real value into another value between 0 and 1. In machine learning, we use sigmoid to map predictions to probabilities. Sigmoid function:

$$S(z) = \frac{1}{1 + e^{-z}}$$

- $S(z)$  = output between 0 and 1 (probability estimate)
- $z$  = input to the function (your algorithm's prediction e.g.  $mx + b$ )
- $e$  = base of natural log



So let's say we have two possible outcomes: Home Win or Away Win. This is how we classify:

- $p \geq 0.5$ , class = 1 (Home Win)
- $p < 0.5$ , class = 0 (Away Win)

For example, if our threshold was .5 and our prediction function returned .7, we would classify this observation as positive. If our prediction was .2 we would classify the observation as negative. For logistic regression with multiple classes we could select the class with the highest predicted probability.

Now to make a prediction we will need to write a prediction function. A prediction function in logistic regression returns the probability of our observation being positive, True, or “Yes”. We call this class 1 and its notation is  $P(\text{class}=1)$ . As the probability gets closer to 1, our model is more confident that the observation is in class 1.

Finally, we use a Cost function to make a graph at different output levels and we use Gradient Descent to find a local minimum as quickly as possible by picking a particular learning rate. More details about this will be covered as I go on and begin with the full process, later on.

## **EXPERIMENTS**

### **Dataset**

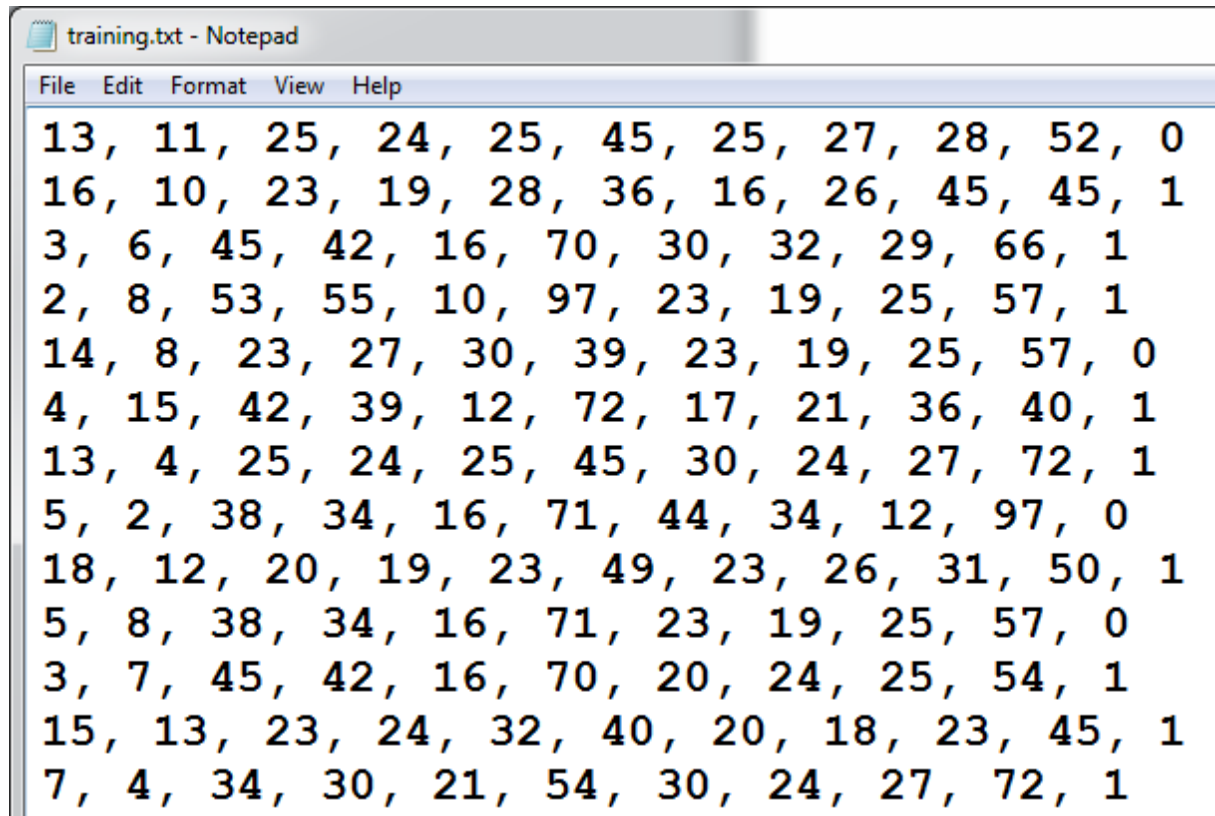
Looking and examining around sports data, I finally decided how my dataset will be established. I used official data from <https://www.rezultati.com> where I could find everything I needed. Just to mention that I manually wrote my data files because I wasn't comfortable with the datasets I found online. So, I wanted to train my algorithm on matches from the English Premier League because I watch them every week when I am at home. I searched for the full team stats in the past season (2018-2019) and using that official data I wrote the first dataset below, by myself. I used the following features:

- How many goals did a team score at home (Home Goals Scored)
- How many goals did a team concede at home (Home Goals Conceded)
- How many total points at home did a team win (Total Home Points)
- How many goals did a team score away (Away Goals Scored)
- How many goals did a team concede away (Away Goals Conceded)
- How many total points at home did a team win (Total Away Points)
- How many points in total did a team have (Total Points)

<b>Team Name</b>	<b>Total Home Points</b>	<b>Home Goals Scored</b>	<b>Home Goals Conceded</b>	<b>Total Away Points</b>	<b>Away Goals Scored</b>	<b>Away Goals Conceded</b>	<b>Total Team Points</b>
Manchester City	54	57	12	44	38	11	98
Liverpool	53	55	10	44	34	12	97
Arsenal	45	42	16	25	31	35	70
Chelsea	42	39	12	30	24	27	72
Tottenham	38	34	16	33	33	23	71
Manchester United	36	33	25	30	32	29	66
Everton	34	30	21	20	24	25	54
Wolves	34	28	21	23	19	25	57
West Ham	31	32	27	21	20	28	52
Bournemouth	29	30	25	16	26	45	45
Leicester	27	24	20	25	27	28	52
Watford	27	26	28	23	26	31	50
Newcastle	25	24	25	20	18	23	45
Southampton	23	27	30	16	18	35	39
Burnley	23	24	32	17	21	36	40
Brighton	23	19	28	13	16	32	36
Fulham	21	22	36	5	12	45	26
Crystal Palace	20	19	23	29	32	30	49
Cardiff	20	21	38	14	13	31	34
Huddersfield	9	10	31	7	12	45	16

I used these particular features because I personally think they should be enough efficient in training my algorithm to predict the score of a match. But, this isn't the only dataset as I need another one that I will directly use to teach my algorithm.

The second dataset is extracted from the first and once again I did it manually. This is my training set of examples, as well. It contains 52 training examples that consist of direct encounters between teams. My training set looks like this (I only showed 13 rows but it has 47 more and I couldn't use a table for this because of the larger number of columns) :



```
training.txt - Notepad
File Edit Format View Help
13, 11, 25, 24, 25, 45, 25, 27, 28, 52, 0
16, 10, 23, 19, 28, 36, 16, 26, 45, 45, 1
3, 6, 45, 42, 16, 70, 30, 32, 29, 66, 1
2, 8, 53, 55, 10, 97, 23, 19, 25, 57, 1
14, 8, 23, 27, 30, 39, 23, 19, 25, 57, 0
4, 15, 42, 39, 12, 72, 17, 21, 36, 40, 1
13, 4, 25, 24, 25, 45, 30, 24, 27, 72, 1
5, 2, 38, 34, 16, 71, 44, 34, 12, 97, 0
18, 12, 20, 19, 23, 49, 23, 26, 31, 50, 1
5, 8, 38, 34, 16, 71, 23, 19, 25, 57, 0
3, 7, 45, 42, 16, 70, 20, 24, 25, 54, 1
15, 13, 23, 24, 32, 40, 20, 18, 23, 45, 1
7, 4, 34, 30, 21, 54, 30, 24, 27, 72, 1
```

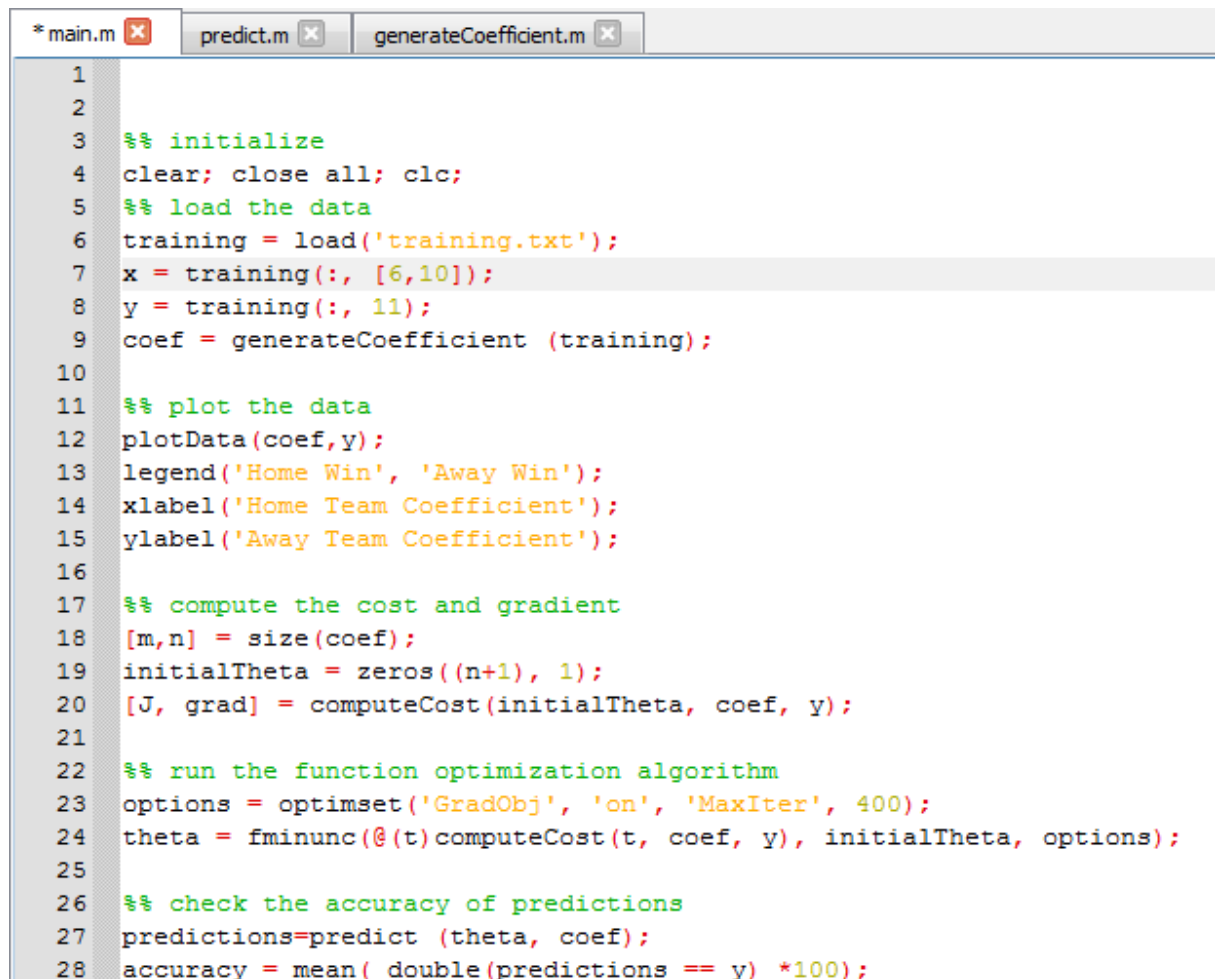
The features in this training datasets are (from left to right):

- Home Team ID (auto-incremented from the order in the first set)
- Away Team ID (auto-incremented from the order in the first set)
- Home Team Total Points Won at Home
- Home Team Goals Scored at Home
- Home Team Goals Conceded at Home
- Home Team Total Points
- Away Team Total Points Won Away
- Away Team Goals Scored Away
- Away Team Goals Conceded Away
- Away Team Total Points
- Outcome ( 1 = Home Win, 2 = Away Win)

Hopefully this training set will provide my desired results. Considering that I follow football as one of my biggest hobbies, I truly believe I made the right decisions when it comes to datasets.

## TOOLS

The environment that I worked in is OCTAVE. I chose OCTAVE because I wanted to get more familiar with it and to improve my skills when it comes to this kind of programming. I used all the libraries available by default. This is how my main.m code looks like (more on this in the following sections):



```
1
2
3 %% initialize
4 clear; close all; clc;
5 %% load the data
6 training = load('training.txt');
7 x = training(:, [6,10]);
8 y = training(:, 11);
9 coef = generateCoefficient (training);
10
11 %% plot the data
12 plotData(coef,y);
13 legend('Home Win', 'Away Win');
14 xlabel('Home Team Coefficient');
15 ylabel('Away Team Coefficient');
16
17 %% compute the cost and gradient
18 [m,n] = size(coef);
19 initialTheta = zeros((n+1), 1);
20 [J, grad] = computeCost(initialTheta, coef, y);
21
22 %% run the function optimization algorithm
23 options = optimset('GradObj', 'on', 'MaxIter', 400);
24 theta = fminunc(@(t)computeCost(t, coef, y), initialTheta, options);
25
26 %% check the accuracy of predictions
27 predictions=predict (theta, coef);
28 accuracy = mean( double(predictions == y) *100);
```

*Code snippet 1.1.*



## RESULTS AND DISCUSSION

Well, you could see my approaches in the sections above but in this section I will be more specific about the results. In my Code snippet 1.1. above there are multiple functions that I used to reach my goal. Firstly I loaded the training data using the load function. I divided the training data into x and y (x was used to check the syntax, y was used for outcome data). Then comes the function that I figured out would be really helpful and it is called generateCoefficient().

```
function c1 = generateCoefficient (t)

    c1 = t(:, [6,10]);
    for i=1:length(t)
        c1(i,1) = t(i,3) + t(i,4) - t(i,5) + t(i,6) + 5;
        c1(i,2) = t(i,7) + t(i,8) - t(i,9) + t(i,10);
    endfor

endfunction
```

*Code snippet 1.2.*

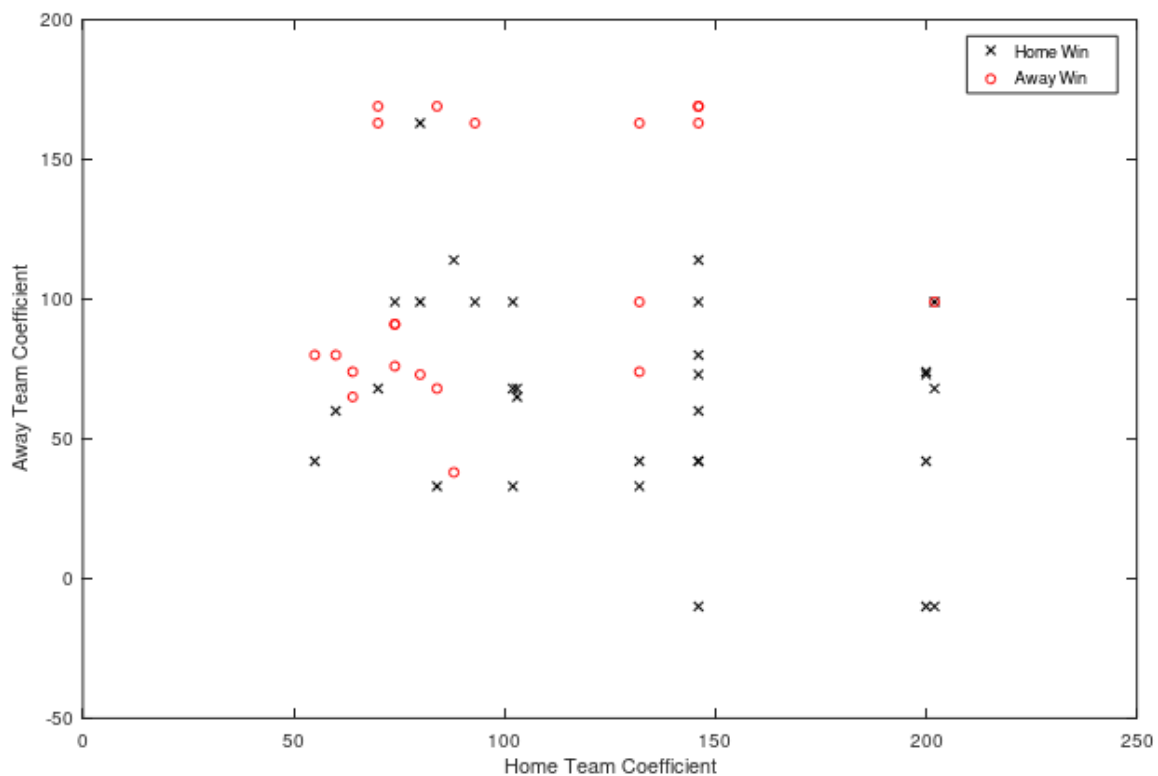
I used this function to combine all data from the dataset and make manually generated coefficients for both the Home and Away team. This is the formula that I came up with and I think that it serves justice for this whole project as it takes everything into consideration and outputs a coefficient that makes a lot of sense.

**Coefficient formula (Home Team) = TotalHomePoints + HomeGoalsScored – HomeGoalsConceded + TotalPoints + 5**

*note: 5 is added to provide additional advantage for the home team like home fans, home stadium etc.*

**Coefficient formula (Away Team) = TotalAwayPoints + AwayGoalsScored – AwayGoalsConceded + TotalPoints**

Then I plotted the data with the axis labels representing the home and away coefficients:



After plotting, I searched through LMS material and Google to find how to compute data and implement cost functions and the gradient descent (due to this being my first time doing a project like this). And in the end I tried to calculate my precision and the accuracy after 52 training set was **73.077**.

For me, this is the most interesting part because I was inserting training examples one by one and tried to notice how the accuracy of my algorithm would change. And as I expected, when the training example showcased an obvious winner (obvious as a favorite to win), the accuracy was much higher. But, here comes the tricky part. When I inserted the first underdog winner (e.g. Brighton beating Arsenal at Arsenal's home stadium) the accuracy fell down for about 8-10 percent. And then after I added another underdog win, the accuracy fell down even more. Of course, a small team beating a big team occurs very rarely but that shows that in sports, nothing can be really surely predicted. And I think that this accuracy in the end of my work is fair, to be honest. Because I think that the favorite team to win will win in about 73% of

the time, but in the rest 27% of the time, the underdog will cause an upset. This also proves that betting is not safe and why underaged teens are not allowed to bet. You could think that Manchester City is going to win against Brighton at home but every now and then, there is an upset that nobody could have predicted.

## **CONCLUSION/FUTURE WORK**

To sum it all up, I personally am proud of my work, considering again that this is my first time dealing with these kinds of algorithms. I expected that the accuracy won't be that high because I know from my personal experience that sports are sometimes way more unpredictable than some might think. I proved that by adding not only the obvious wins but also the wins that cause the most upset during the past Premier League season. I also proved that betting or concretely said, predicting football games outcomes for money isn't safe at all. I must admit that sometimes I place bets in my freetime but I do it carefully and accordingly so that if I lose, I don't lose significantly.

Of course, I could have had an accuracy above 90% with only adding matches where the favorite team won, but I, myself, am always rooting for the smaller team so I really enjoyed putting in the smaller wins, although my accuracy fell down a bunch each time. 73% accuracy for a football match, is pretty fair, in my opinion.

After this project, I am way more familiar with OCTAVE/MATLAB syntax and know I am able to approach better to future projects of this kind. Of course, if I had an experienced team working on this, the accuracy or the algorithm itself would have been better. Next time, if I work on this particular topic, I would dig deeper and build an even more reliable coefficient that would include maybe 5-10 more features. I am also open to doing this project but on another sport, maybe tennis or basketball because I love watching them too. My final word is that this was one of the most enjoyable projects during the educational path to my Bachelor Degree and I'm genuinely grateful for it.