Machine Learning Engineer Nanodegree

Capstone Proposal

Felipe Quirce May 2nd, 2017

Proposal

Football results prediction.

Domain Background

Betting is massive domain and there plenty of companies looking to make predictions based on football data since a good prediction allows you to price bets. I believe that most of the companies are doing a good job predicting results since they make a lot of money from bets.

This project will be based in [https://www.kaggle.com/hugomathien/soccer] data, is data about 25.000 scores and 10.000 players, using this data will try to predict the outcome of the match. If we consider the amount of data and the quality of the data we should be able to predict results based on the squads and the stats.

I like football and this could be my first iteration on this data, and later on, I could search for other prediction like the which is the most important player per squad, which are the best odds based on the prediction, check if the odds are right based on the predictions.

I've found a few articles where some people try to predict results of sports games like An Artificial Neural Network Approach to College Football Prediction and Rankings and Using Machine Learning to Predict the Outcome of English County twenty over Cricket Matches which are final projects.

Problem Statement

I want to use the historical match data to predict more recent match results. The problem is supervised learning problem, where the labels are the different possible results of the match and the possible features are the players and their skill level.

The skill level of the players is based on the FIFA skill set, those are relevant because they get updated quite often on the relation with the player's performance. The result of the match is going to be a classification problem since we will try to know how will win or if there is going to be a draw.

Datasets and Inputs

This project will be based on https://www.kaggle.com/hugomathien/soccer data, is data about 25.000 matches and 10.000 players and all the teams. This data also contains the players for every match, If we only had the names of the teams it will be difficult to give a prediction but since we have the squads will be able to give a proper prediction based on the squads.

There are 3 possible sources of data that could be used to solve the problem:

1. The FIFA skill set of the players that are on the 16 that have been called for the match. There are around 20+ skills that for each player that could be used:

- Crossing
- Finishing
- Heading Accuracy
- Short Passing
- Volleys
- Dribbling
- Curve
- Free Kick Accuracy
- Long Passing
- Ball Control
- etc…

I will use PCA in order to get one feature that summarizes all of them, the 16 players per team will be used the players in the match

2. The stats of the teams in the n-previous games, this will give us a good prediction on how the team was doing before the game. The stats collected include goals, faults, cards, shots off, shots in, etc…

3. The bet prices before the match can give us a hint on the prediction of the match. The bet prices include features like:

    - B365H = Bet365 home win odds
    - B365D = Bet365 draw odds
    - B365A = Bet365 away win odds
    - BbMxH = Betbrain maximum home win odds
    - BbMx>2.5 = Betbrain maximum over 2.5 goals

Solution Statement

The first step is to preprocess the data, to do this I will use three main techniques:

1. Data selection: Looking for the data that contains the relevant features for example matches without starting squad, players without stats

2. Scaling: The features can have very different ranges and that can affect the classifier, so a scaling seems necessary, minMaxScaler will be the algorithm selected to apply this technique.
3. Decomposition: Decomposition is a method that reduces the features in orthogonal components that explain a maximum amount of variance, PCA will be the algorithm selected to apply this technique.

There are 2 main algorithms that would be a good fit for this problem:

1. DecisionTreeClassifier could be a good algorithm since:
   - The data requires less preparation that when you use other algorithms.
   - The tree will be easy to analyze afterward.
2. Neural Network could be a good algorithm since:
   - The data requires less preparation (scaling) that when you use other algorithms.
   - The dataset is quite large 25.000 scores.
   - Can approximate any function, regardless of its linearity

Given those features will try to train different algorithms using the results that we have in the database. To train we and test the will use the data of the previous years and we will use as validation set the latest data available in the database

Benchmark Mode

My benchmark model will be comparing with the result obtained in predict-winners-big-games-machine-learning, where the writer archived an accuracy of 70%.

Evaluation Metrics

The f1__beta score on the validation dataset will be metric used to evaluate the model and compare the different models used since this is a multi-label problem we will average of the F1__beta score of each class.

$$F_\beta = (1+\beta^2)\cdot\frac{\text{precision}\cdot\text{recall}}{(\beta^2\cdot\text{precision})+\text{recall}} = \frac{(1+\beta^2)\cdot\text{true positive}}{(1+\beta^2)\cdot\text{true positive}+\beta^2\cdot\text{false negative}+\text{false positive}}$$

Project Design

the first steps that I will use to solve the problem are going to be common for the different algorithms: * clean the data ex: discard the matches that don't have squads, remove matches which have players without info, etc…

- Normalize the data ex: label the data, extract the Avg of skills for each player, etc.... Using PCA in order to reduce dimensionality on the player skills will be totally necessary since 16 players * 2 teams * 20+ skills makes 640 features.
- Divide the data in training, test, and validation, using for validation the recent matches which should be about of 5% of the matches. GroupKFol will be the method used to split data between training and testing since it will allow us to keep the balance between classes.

Then I will try to solve in using 2 different approaches:

- Use a Multiple-Layer Feedforward Architectures Neural network to predict the label, using tensorflow.
- Apply pca or similar to reduce the features and apply decision tree using the sklearn as a library.

After implement the solution will have to tune the hyper parameters of each algorithm in order the get the best possible result.