Grady Lynch

English Premier League Financial Analysis

Team:
Grady Lynch

Subject:
The financial landscape of soccer in the English Premier League is very complicated, and very volatile. Unlike popular American sports leagues like the NFL or the NBA, the Premier League does not have a salary cap. This means that Premier League teams are not limited by the league in how much money they spend on their players' salaries. Another important financial aspect of the Premier League is the presence of player transfer fees. The most common way that a soccer player moves to a new team is when one team pays a transfer fee to the player's current team in exchange for that player. So when an ambitious team with financial muscle decides to buy all the best players and spend boatloads of money, this team should be dominant. However, this hasn't always been the case historically.

I am interested in finding out what the actual value of the money spent by Premier League teams is in terms of their performance.

Data:
I'll be using the data from 11 Premier league clubs that have avoided relegation since 2017. Relegation refers to the demotion of the three lowest-performing teams each season to the second-tier league. Including teams that have fluctuated between the Premier League and the second division would introduce inconsistencies in the dataset. Since I am interested in comparing performance and finances year over year, the first year of data I will use will be 2018, one year after 2 of the 11 clubs completed their first year in the Premier League.

Of the 7 seasons of data that I will need, I obtained the first 6 of them from kaggle. For the last season, 2023 - 2024, I will be able to research the information needed to create this myself.

Here are the steps of my data curation.

First, I modified the dataset I got from Kaggle by removing all features except "Team", "salaries", "spending", and "points". I also removed all rows for teams outside of the 11 teams that I am analyzing. Then I changed the “spending” feature in the data set from just expenditure to net expenditure. Net spending for a soccer team is their transfer expenditure minus their transfer income.

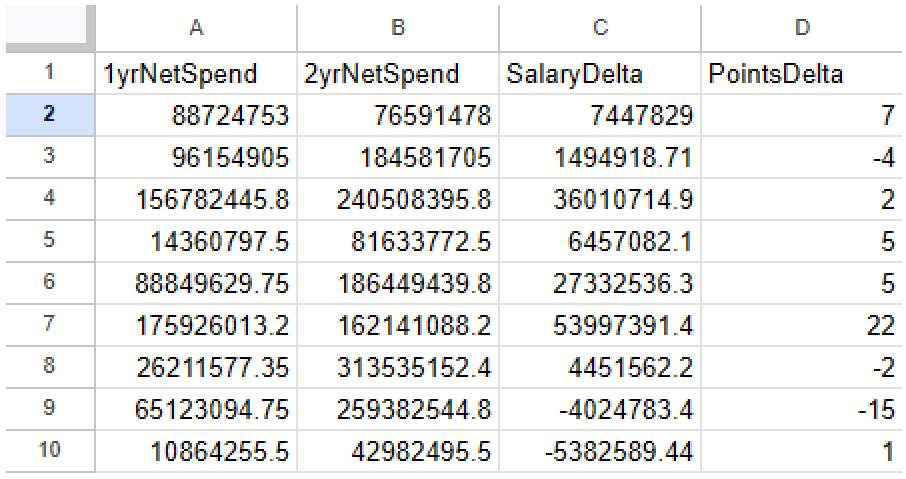
Second, I created a spreadsheet of data for the 2023-2024 season that matched my other data. I used data from the website Transfermarkt.

Third, I normalized all monetary data across my data sets for inflation. This is because I want to be able to combine all the data across teams and seasons into a dataset that I can randomly split for training and testing. To normalize for inflation, I used the Consumer Price Index numbers for the United Kingdom and the European Union, because my net spending data was in euros and my salary data was in pounds. The equation I used was: Actual value = Historical value X Current CPI / Historical CPI. Before moving on to the next step, I also converted the euros and pounds values to USD using current exchange rates.

Fourth, I calculated my features PointsDelta(outcome), 1yrNetSpend, 2yrNetSpend, and SalaryDelta. I did this for each row in each season except the first. The first season of data is just used for the calculation of my deltas and 2 year net spend attributes.

Finally, I added my rows to a final dataset without the team or season labels. From this point I had normalized and anonymous data with my three features and my outcome.

Here is a snapshot of a few rows from my final dataset:



	A	B	C	D
1	1yrNetSpend	2yrNetSpend	SalaryDelta	PointsDelta
2	88724753	76591478	7447829	7
3	96154905	184581705	1494918.71	-4
4	156782445.8	240508395.8	36010714.9	2
5	14360797.5	81633772.5	6457082.1	5
6	88849629.75	186449439.8	27332536.3	5
7	175926013.2	162141088.2	53997391.4	22
8	26211577.35	313535152.4	4451562.2	-2
9	65123094.75	259382544.8	-4024783.4	-15
10	10864255.5	42982495.5	-5382589.44	1

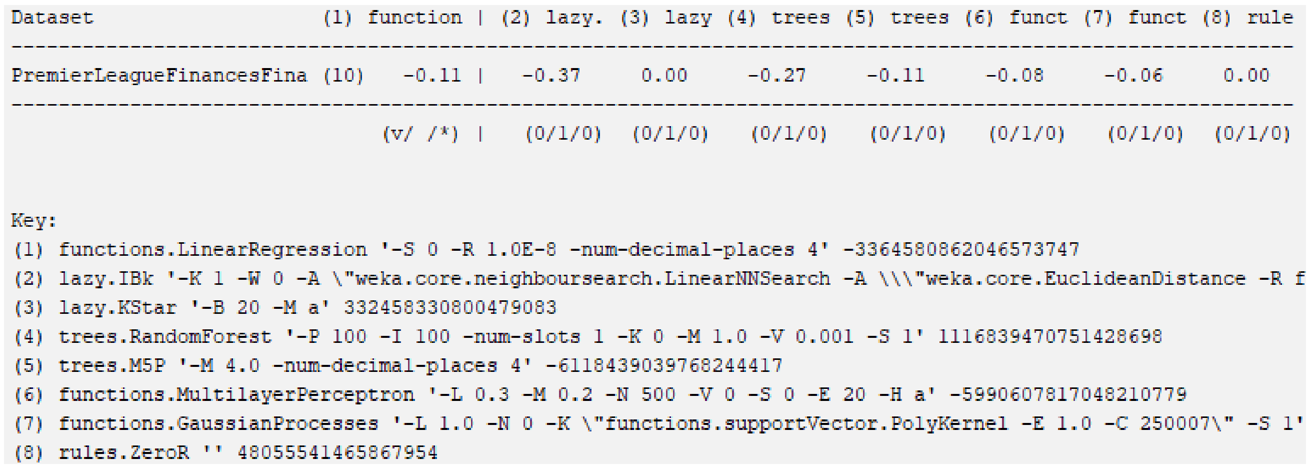
Features:

My features were PointsDelta(outcome), 1yrNetSpend, 2yrNetSpend, and SalaryDelta. All of these features I derived from the raw data except for 1yrNetSpend. The features were derived very simply by either calculating the sum or the difference of one of the raw features across 2 consecutive seasons of data.

The reason that I chose these features is because they are comparative. The question that I am interested in is how does a change in investment level in a soccer team affect the team's performance. To see the changes in investment level you have to compare the values across at least two seasons.

Experiments & Results:

I loaded my csv file into Weka and converted it to an arff file in the preprocessing tab. I used the Weka Experiment Environment to run all of my models and algorithms at once and get easy to interpret output. Because my dataset's output was numeric, I was limited in what algorithms and models Weka would be able to use. For my first experiment I used a 75/25 train test split and 8 different models and algorithms. Here is a sample of the Weka output followed by a comparative table of the most significant data for each of the model's results.



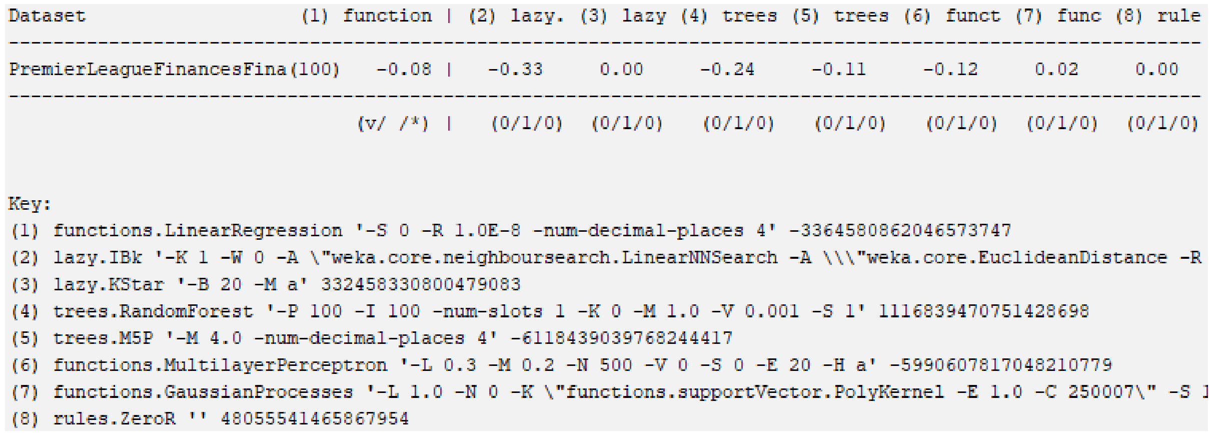
```
Dataset                   (1) function | (2) lazy. (3) lazy (4) trees (5) trees (6) funct (7) funct (8) rule
--------------------------------------------------------------------------------------------------------------
PremierLeagueFinancesFina (10)   -0.11 |   -0.37     0.00     -0.27     -0.11     -0.08     -0.06     0.00
--------------------------------------------------------------------------------------------------------------
                               (v/ /*) |   (0/1/0)  (0/1/0)   (0/1/0)   (0/1/0)   (0/1/0)   (0/1/0)  (0/1/0)


Key:
(1) functions.LinearRegression '-S 0 -R 1.0E-8 -num-decimal-places 4' -3364580862046573747
(2) lazy.IBk '-K 1 -W 0 -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.EuclideanDistance -R f
(3) lazy.KStar '-B 20 -M a' 332458330800479083
(4) trees.RandomForest '-P 100 -I 100 -num-slots 1 -K 0 -M 1.0 -V 0.001 -S 1' 1116839470751428698
(5) trees.M5P '-M 4.0 -num-decimal-places 4' -6118439039768244417
(6) functions.MultilayerPerceptron '-L 0.3 -M 0.2 -N 500 -V 0 -S 0 -E 20 -H a' -5990607817048210779
(7) functions.GaussianProcesses '-L 1.0 -N 0 -K \"functions.supportVector.PolyKernel -E 1.0 -C 250007\" -S 1'
(8) rules.ZeroR '' 48055541465867954
```

75/25 Train Test Split Results

	Correlation Coefficient	Mean Absolute Error	Root Mean Squared Error
Linear Regression	-0.11	10.80	13.45
Lazy IBK	-0.37	16.55	21.30
Lazy KStar	0.00	10.32	12.93
Random Forest	-0.27	12.22	15.51
M5P Tree	-0.11	10.80	13.45
Multilayer Perceptron	-0.08	12.30	16.35
Gaussian Processes	-0.06	10.41	13.10
ZeroR	0.00	10.36	13.03

These results show that all of the models and algorithms performed very poorly. The correlation coefficients were very far from either 1 or -1, and the error values were very high relative to the numerical values of my output data for PointsDelta. These results are disappointing, but my expectations were quite low so I wasn't surprised. The reason that I was interested in this data project is that I know how random and unexpected teams performances in the English Premier League can be. Every season there are huge surprises with teams over performing and underperforming. This creates large variance in most teams' PointsDelta value. I think that one of the reasons for my poor results is the small size of my dataset, so what I wanted to do next is try cross validation instead of train test split. Cross validation is supposed to be preferable to train test split for small datasets. Here are the results using cross validation.



```
Dataset                   (1) function | (2) lazy. (3) lazy (4) trees (5) trees (6) funct (7) func (8) rule
----------------------------------------------------------------------------------------------------------
PremierLeagueFinancesFina(100)   -0.08 |   -0.33     0.00     -0.24     -0.11     -0.12     0.02     0.00
----------------------------------------------------------------------------------------------------------
                               (v/ /*) |   (0/1/0)  (0/1/0)   (0/1/0)   (0/1/0)   (0/1/0)  (0/1/0)  (0/1/0)


Key:
(1) functions.LinearRegression '-S 0 -R 1.0E-8 -num-decimal-places 4' -3364580862046573747
(2) lazy.IBk '-K 1 -W 0 -A \"weka.core.neighboursearch.LinearNNSearch -A \\\"weka.core.EuclideanDistance -R
(3) lazy.KStar '-B 20 -M a' 332458330800479083
(4) trees.RandomForest '-P 100 -I 100 -num-slots 1 -K 0 -M 1.0 -V 0.001 -S 1' 1116839470751428698
(5) trees.M5P '-M 4.0 -num-decimal-places 4' -6118439039768244417
(6) functions.MultilayerPerceptron '-L 0.3 -M 0.2 -N 500 -V 0 -S 0 -E 20 -H a' -5990607817048210779
(7) functions.GaussianProcesses '-L 1.0 -N 0 -K \"functions.supportVector.PolyKernel -E 1.0 -C 250007\" -S 1
(8) rules.ZeroR '' 48055541465867954
```

Cross Validation Results

	Correlation Coefficient	Mean Absolute Error	Root Mean Squared Error
Linear Regression	-0.08	10.19	12.43
Lazy IBK	-0.33	16.82	20.95
Lazy KStar	0.00	9.59	11.71
Random Forest	-0.24	11.24	13.93
M5P Tree	-0.11	10.19	12.46
Multilayer Perceptron	-0.12	11.62	14.55
Gaussian Processes	0.02	9.88	12.06
ZeroR	0.00	9.71	11.90

With cross validation, there is slight improvement in the Mean Absolute Error and Root Mean Squared Error results, but the Correlation Coefficient results are just as bad if not worse than the train test split results. Overall, these results are also poor, and I think that there are a couple reasons for this. I think one reason is, like I already mentioned, the randomness of the English Premier League. Another reason that I think my results are so poor is the size of my dataset. The curation and derivation of my final dataset took a very long time, and because of the English Premier League's rules of relegation and promotion, I was limited in the number of teams and number of seasons of data that was usable. I think that if there were a way to have more seasons and teams data, my results might improve. However, I think that the most significant reason for my poor results was the weakness of my chosen features. The results were so bad that even though dataset size probably played a role, with stronger and possibly more features the results would have at least showed some promise.

Next Steps:

The next steps of this project will be for me to reassess my features, curate better data, and consider changing my output value to a nominal one.

Curating better data will be a challenge because I have to come up with a way to keep the consistency of the financial data despite possibly including data for teams who spend time in both the Premier League, and the secondary professional English soccer division.

The next step of the project that I am considering will also be difficult. I need to find a way to add more features that have to do with a team's trend of financial investment in the medium to short term. It is also possible that I modify my current features. I could normalize the financial data by converting it into a percentage of the corresponding teams overall value. This would make the comparison between more wealthy and less wealthy teams better.

The final thing that I have planned for this project right now is to change my numerical value of PointsDelta, into a nominal value. This nominal value would likely be related to if the team improved their points total, stayed the same, or regressed. I think that this change would be the easiest to implement, and it would open the door for me to use classifiers in Weka that I wasn’t able to before.