Predicting the Spread in the English Premier League

Michael Weber

Motivation

The beautiful game aka football aka soccer is the most popular sport in the world with more than 4/10 people polled in a recent sample stating they supported and watched a particular club regularly. That more than 4/10 comes out to more than 2 billion people as supporters of a football club. The English Premier League is the most popular league in the world by viewers and tv deals worth over 1.5 Billion usd annually. Also the betting industry is valued at over 1 Trillion annually. Finding the most predictive features to generate postive results for a club would be very valuable information but the game of soccer is notoriously hard to predict.

So, can you use a linear regression model to predict the winning margin in the English Premier League?

Initial Data set

I found the last 10 seasons of EPL match results and stats from **DataHub.io** Match stats including **score, shots, shots on target, corner kicks and yellow/red card counts.**

Code to follow along: https://github.com/MichaelWeber2050/Project_2/blob/master/Project%202%20-%20EPL%20Spread%20Prediction%20-%20Code%20.ipynb (https://github.com/MichaelWeber2050/Project_2/blob/master/Project%202%20-%20EPL%20Spread%20Prediction%20-%20Code%20.ipynb)

Web scraped more data with Selenium

Through **pair plots** and **exploratory data analysis** I determined the initial dataset of match stats were **not descriptive enough** to use as features, so I used **Selenium to scrape passing and posession stats from Premierleague.com** The code worked but did not want to get through a whole season in one function call so I split the gathering into 3 equal parts, saved to csv files, then concatenated each csv to make a full season to add to DataFrame of match stats.

Also scraped club wages and promotion/relegations data

Club wages only very accurate data available for the last 4 seasons. In the EPL the bottom 3 teams at the end of the year get removed from the league and replaced with the top 3 teams from the 2nd division league so there is turnover each year and I thought this would be a good categorical feature to include because if a team has been relegated or promoted they are likely a less talented team and thus their inclusion to a game could prove to influence the spread.

Also, scraped from EA Sport the top 100 players in the world ranking and if a team in the EPL held those players I added a feature to denote that. Higher ranked players should theoretically produce influence on the match result spread.

Feature Engineering

My stats alone and even the additional information was not proving very effective at predicting the match spread result. I first used a basic OLS to check for coefficients to get a pulse of my data.

I realized the teams themselves were not being properly weighted in the model so I chose to give them a numerical rating coressponding to their place in the league at the time of the match.

But the rankings based on league position were also not being properly weighted so I gave each team Home/Away a ranking based on their Defensive and Attack/Offensive strength in realtion to the rest of the league.

I used dictionaries of the goals scored and goals conceded of each team after each match and tallied a goal per game scored and conceded measure and then divided that by each season's league running avg. **This resulted in my most predictive features and really bumped up my predictive R squared measure and showed vast improvement in the error measures RMSE and MAE.**

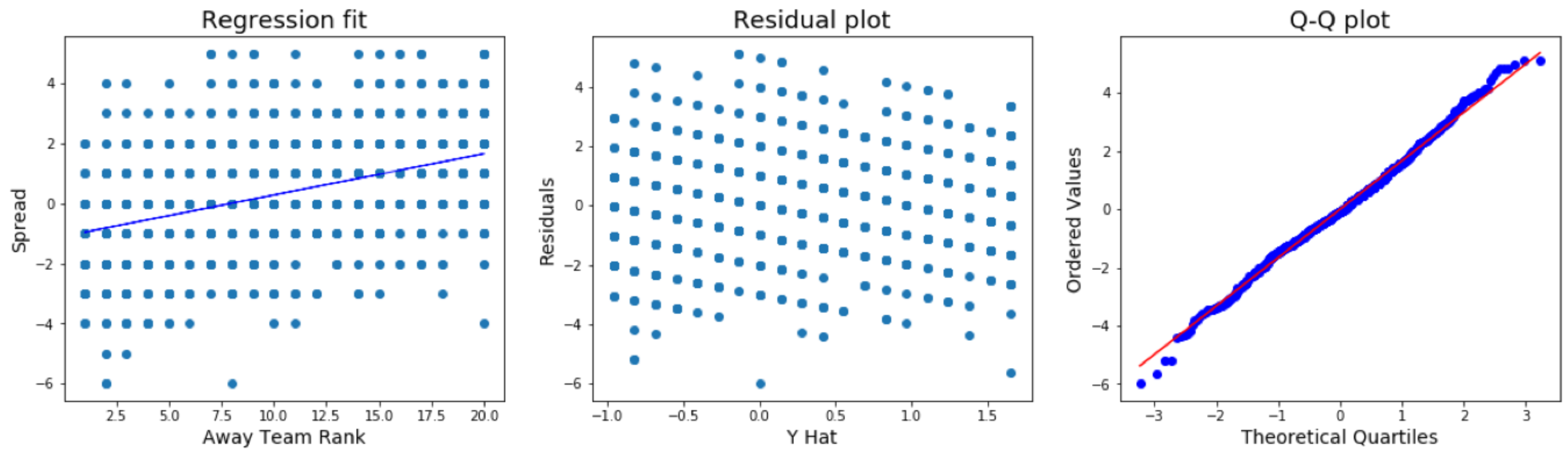
Code for reference in jupyter notebook **"Project 2 - EPL Spread Prediction - Code "**: It is rather long and clunky but it got the job done. Very open to suggestions for improvement.

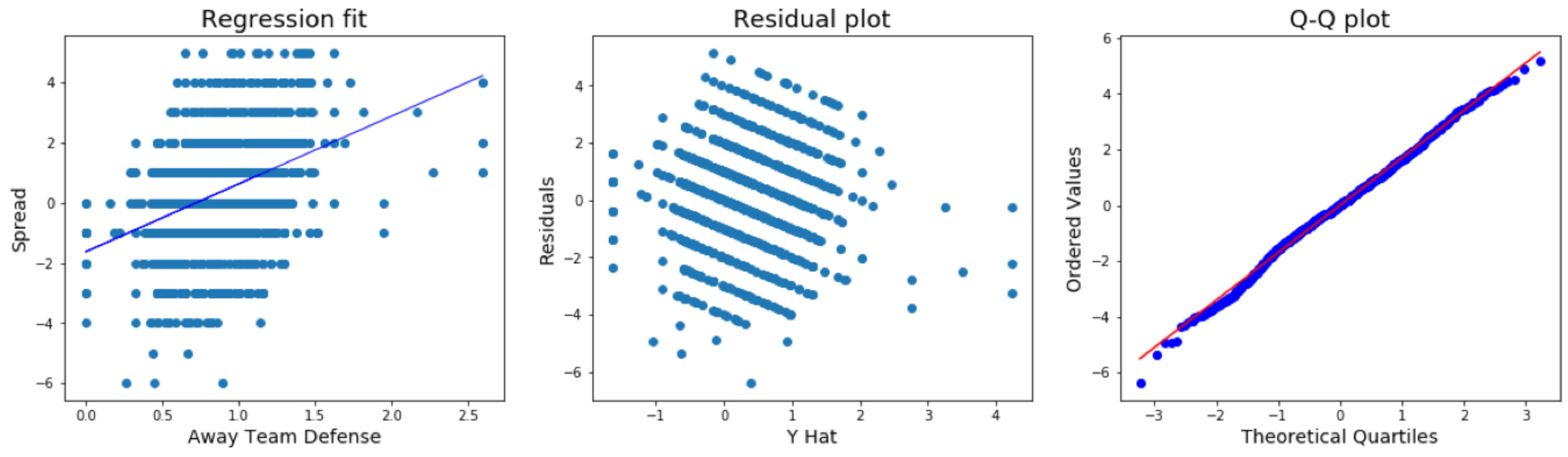
Top 3 graphs

show Team rank based on position in the league with **LITTLE PREDICTIVE POWER** but normally distributed

Bottom 3 graphs

show after feature engineering Defensive and Attack features per team with **MUCH MORE PREDICTIVE WEIGHT**



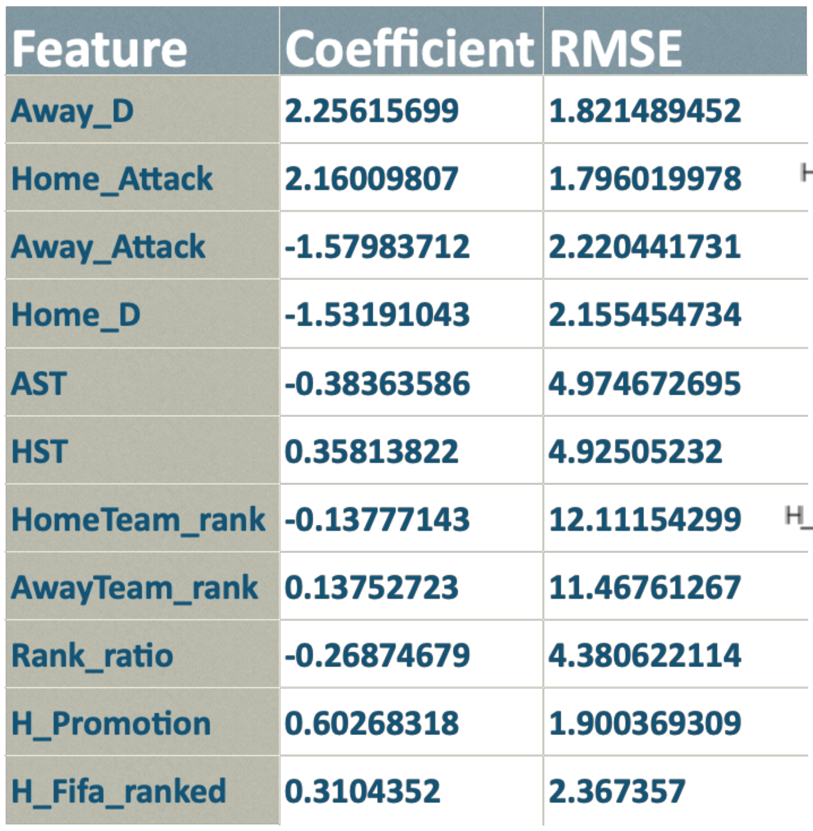


I had my Four engineered features which I knew to be my best baseline predictors from my Forward Feature selection plotting and one by one testing. Meaning I placed each potential feature into an OLS regression model and recorded the Coefficients RMSE MAE and Intercept.

I chose RMSE because I found it the best indicator for feature value and reducing the error and variance of the model.

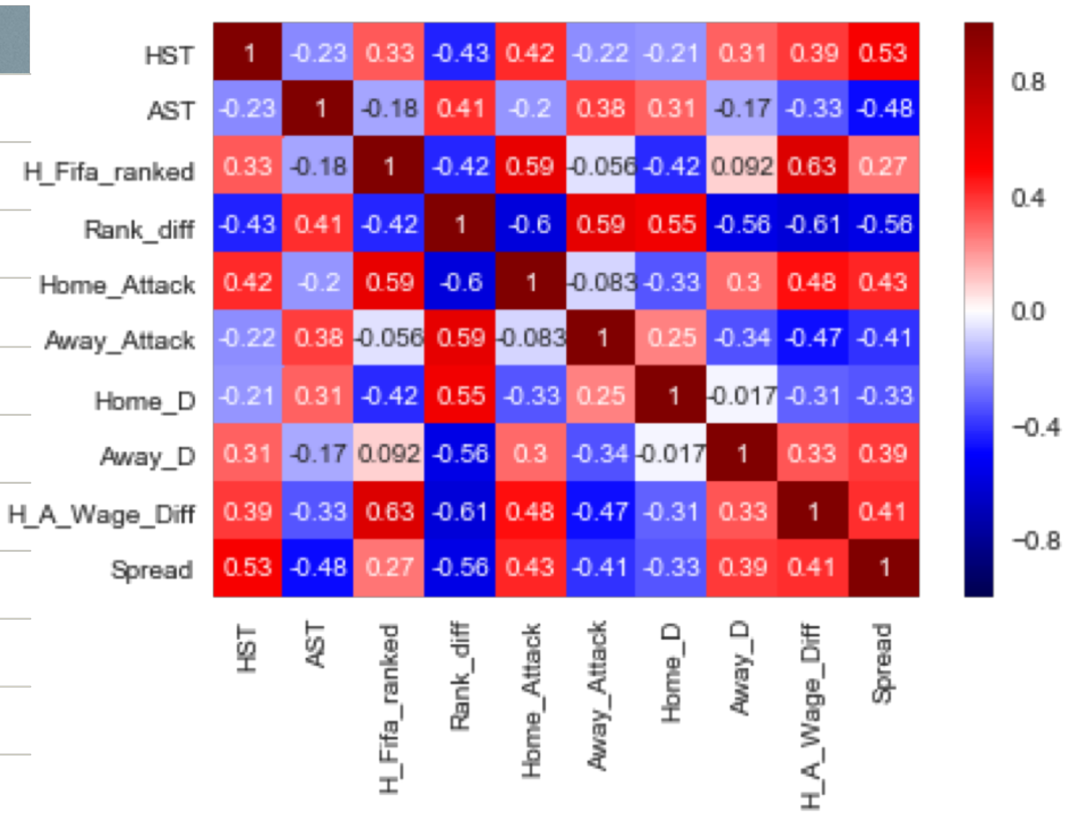
See appendix slide below Again the HomeTeam_rank and AwayTeam_rank variables and after engineering into the top four features.

Feature Selection Table



Feature	Coefficient	RMSE
Away_D	2.25615699	1.821489452
Home_Attack	2.16009807	1.796019978
Away_Attack	-1.57983712	2.220441731
Home_D	-1.53191043	2.155454734
AST	-0.38363586	4.974672695
HST	0.35813822	4.92505232
HomeTeam_rank	-0.13777143	12.11154299
AwayTeam_rank	0.13752723	11.46761267
Rank_ratio	-0.26874679	4.380622114
H_Promotion	0.60268318	1.900369309
H_Fifa_ranked	0.3104352	2.367357

Feature Selection Heat Map



	HST	AST	H_Fifa_ranked	Rank_diff	Home_Attack	Away_Attack	Home_D	Away_D	H_A_Wage_Diff	Spread
HST	1	-0.23	0.33	-0.43	0.42	-0.22	-0.21	0.31	0.39	0.53
AST	-0.23	1	-0.18	0.41	-0.2	0.38	0.31	-0.17	-0.33	-0.48
H_Fifa_ranked	0.33	-0.18	1	-0.42	0.59	-0.056	-0.42	0.092	0.63	0.27
Rank_diff	-0.43	0.41	-0.42	1	-0.6	0.59	0.55	-0.56	-0.61	-0.56
Home_Attack	0.42	-0.2	0.59	-0.6	1	-0.083	-0.33	0.3	0.48	0.43
Away_Attack	-0.22	0.38	-0.056	0.59	-0.083	1	0.25	-0.34	-0.47	-0.41
Home_D	-0.21	0.31	-0.42	0.55	-0.33	0.25	1	-0.017	-0.31	-0.33
Away_D	0.31	-0.17	0.092	-0.56	0.3	-0.34	-0.017	1	0.33	0.39
H_A_Wage_Diff	0.39	-0.33	0.63	-0.61	0.48	-0.47	-0.31	0.33	1	0.41
Spread	0.53	-0.48	0.27	-0.56	0.43	-0.41	-0.33	0.39	0.41	1

Feature engineering done, time to model

I had a group of 16 features with varying importance from the RMSE MAE and Coefficients + my own intution, **Now time to model**

I placed my top 4 features into the **OLS model to get a baseline for my R^2 and RMSE** going forward before other models. R^2 was .48 up substantially from my first of .23 when I ran with my initial not web scraped and non feature engineered data.

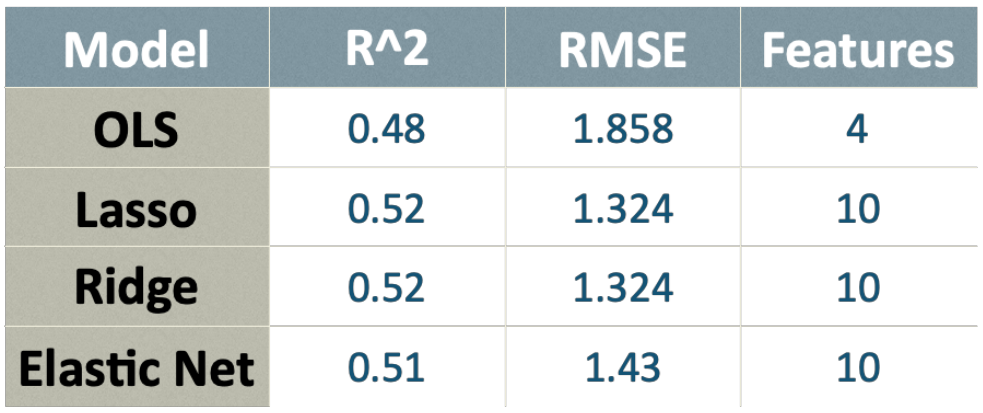
Then I **ran a Lasso with Cross Validation to find a basis point for my alpha** to then use Lasso for some further feature selection tuning the alpha up from the suggested low to zero out features and determine my best fits.

This got my to about 8-12 best features down from 16 using only the training data.

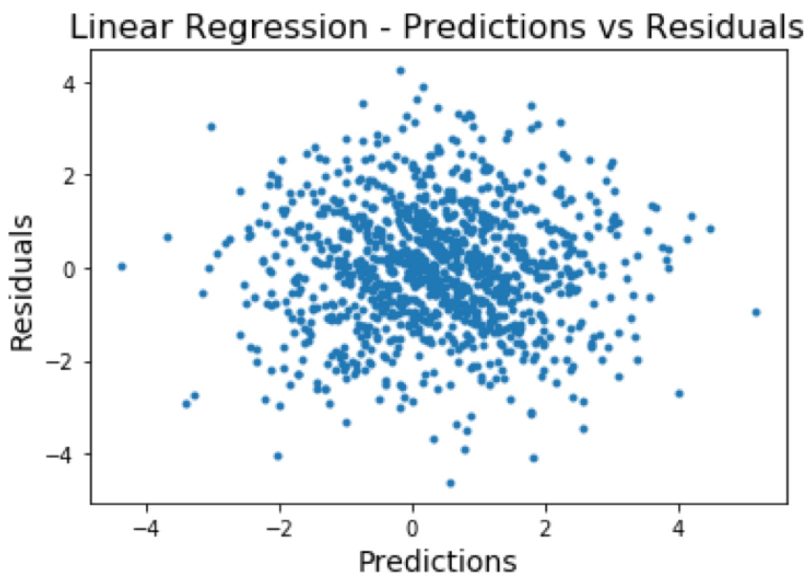
Train Test Split the data I set up 3 models Ridge with Cross Validation, Lasso with Cross Validation and Elastic Net Each model would see 3 feature sets, 8 features, 10 features and 12 features and record scores with Cross Validation alphas and apply each model to the training validation and test sets. **Applied standard scalers** to the data for each model

I did perform a **log10 transform** to one feature I had constructed but ultimately it was collinear with another feature which proved a better fit for the model so the log10 transformed feature was left out.

I **plotted each model's Predictions vs Residuals** to check for random noise meaning the realtionship between features and Residuals(Yhat-Y) and thus Y values is reasonably linear. It suggests error variances are equal because the points are generally around the horizontal line at zero. And it does show some outliers that occur but after data cleaning I determined those are true occurences. ie Man City beating Stoke 7-2 Oct 2017



Model	R^2	RMSE	Features
OLS	0.48	1.858	4
Lasso	0.52	1.324	10
Ridge	0.52	1.324	10
Elastic Net	0.51	1.43	10

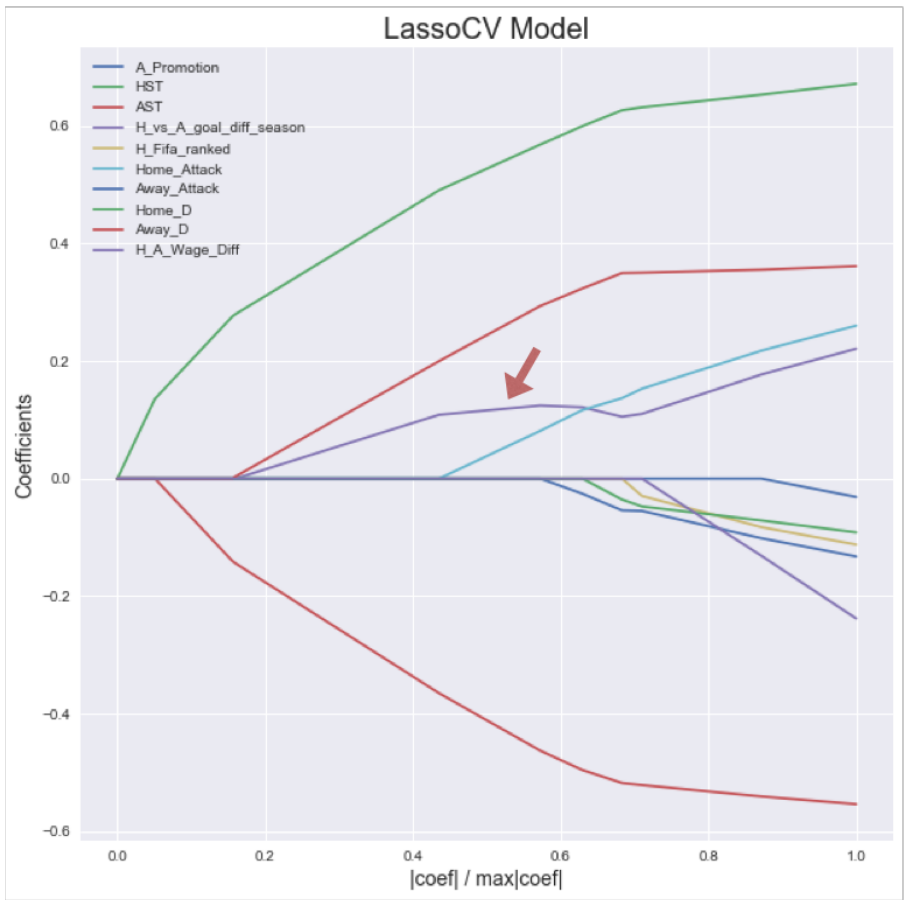


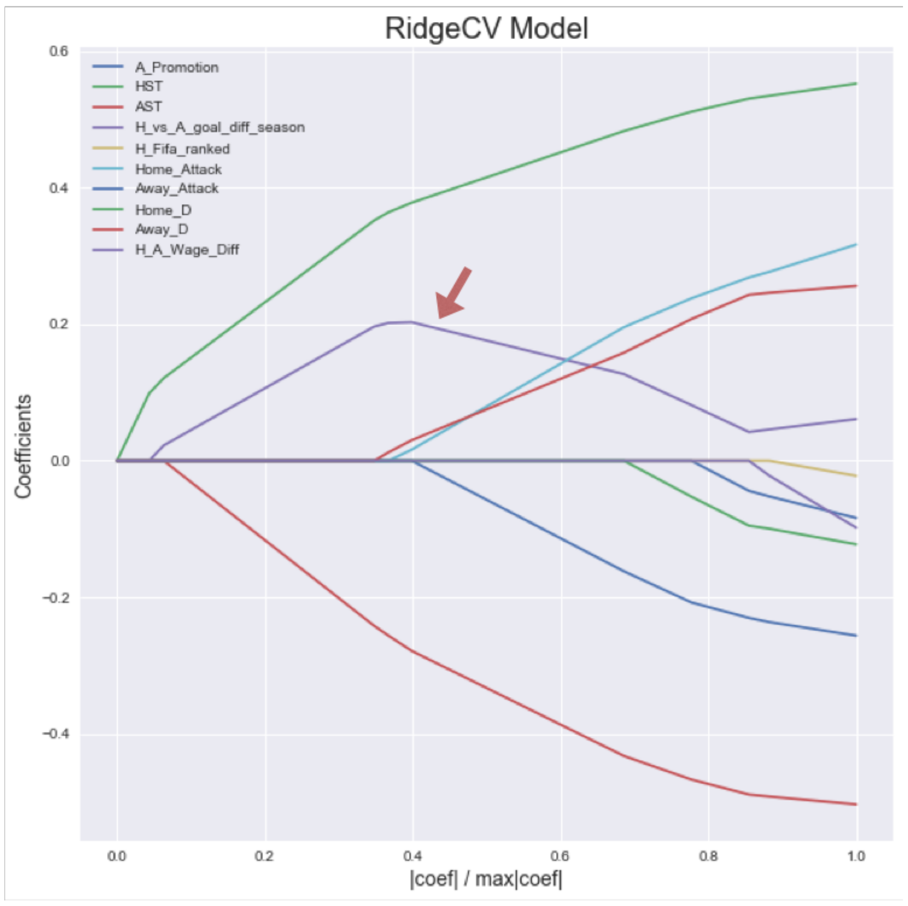
The Lasso and Ridge performed very similar over the cross validation trials and nearly identical in the final models.

And when I compared the plot of each to see the features I saw **Ridge** kept the (Home - Away wages) feature with a low coeffient value early that spikes higher as other variables get zeroed out. It feels suspect to use the Ridge with a coefficient value that starts very low and then has much more sway later on.

Lasso zeroed it out sooner and had the two features I engineered Away D and Home D as the most important variables to carry to end the for prediction.

Overall I think at this stage I like the **Lasso Model is best** because: 1) Lasso has a slightly lower RMSE, here the better model has the least error. 2) Lasso emphasizes the Defensive measures nearly eaqual and in sports the old adage "Defense Wins Championships" proves true time and time again.





Lasso and Ridge with 10 features + scaler transform R^2 .523 RMSE 1.324

Conclusion

So, can this model predict the spread of a match in the EPL?

Not well with an average error of 1.3 goals off the mark but I think it did allow me to find some interesting insights like the importance of defense to positive results much more so than even precise attacking play and the wages paid to players by a club.

Lastly I found the promotion sides (teams promoted to the top league) are much more successful at home than away and top fifa ranked players follow the same pattern, producing little effective difference away from home but show a decent effective weight when playing in front of their home crowd.

Future Work

1) For future work I would like to add more seasons to train the model on more data and see how it performs.

2) Also I would like to engineer more features from the raw stats that could help increase the performance of the model.

3) Lastly I would like to explore more model types and a more Bayesian perspective, try to use weighted probabilities that could help with a different result.

Thank you!

Michael Weber

Predicting the Spread in the English Premier League

Michael Weber

Motivation

The beautiful game aka football aka soccer is the most popular sport in the world with more than 4/10 people polled in a recent sample stating they supported and watched a particular club regularly. That more than 4/10 comes out to more than 2 billion people as supporters of a football club. The English Premier League is the most popular league in the world by viewers and tv deals worth over 1.5 Billion usd annually. Also the betting industry is valued at over 1 Trillion annually. Finding the most predictive features to generate postive results for a club would be very valuable information but the game of soccer is notoriously hard to predict.

So, can you use a linear regression model to predict the winning margin in the English Premier League?

Initial Data set

I found the last 10 seasons of EPL match results and stats from **DataHub.io** Match stats including **score, shots, shots on target, corner kicks and yellow/red card counts.**

```
In [ ]:
```

```
In [ ]:
```

Web scraped more data with Selenium

Through **pair plots** and **exploratory data analysis** I determined the initial dataset of match stats were **not descriptive enough** to use as features, so I used **Selenium to scrape passing and posession stats from Premierleague.com** The code worked but did not want to get through a whole season in one function call so I split the gathering into 3 equal parts, saved to csv files, then concatenated each csv to make a full season to add to DataFrame of match stats.

Also scraped club wages and promotion/relegations data

Club wages only very accurate data available for the last 4 seasons. In the EPL the bottom 3 teams at the end of the year get removed from the league and replaced with the top 3 teams from the 2nd division league so there is turnover each year and I thought this would be a good categorical feature to include because if a team has been relegated or promoted they are likely a less talented team and thus their inclusion to a game could prove to influence the spread.

Also, scraped from EA Sport the top 100 players in the world ranking and if a team in the EPL held those players I added a feature to denote that. Higher ranked players should theoretically produce influence on the match result spread.

Feature Engineering

My stats alone and even the additional information was not proving very effective at predicting the match spread result. I first used a basic OLS to check for coefficients to get a pulse of my data.

I realized the teams themselves were not being properly weighted in the model so I chose to give them a numerical rating coressponding to their place in the league at the time of the match.

But the rankings based on league position were also not being properly weighted so I gave each team Home/Away a ranking based on their Defensive and Attack/Offensive strength in realtion to the rest of the league.

I used dictionaries of the goals scored and goals conceded of each team after each match and tallied a goal per game scored and conceded measure and then divided that by each season's league running avg. **This resulted in my most predictive features and really bumped up my predictive R squared measure and showed vast improvement in the error measures RMSE and MAE.**

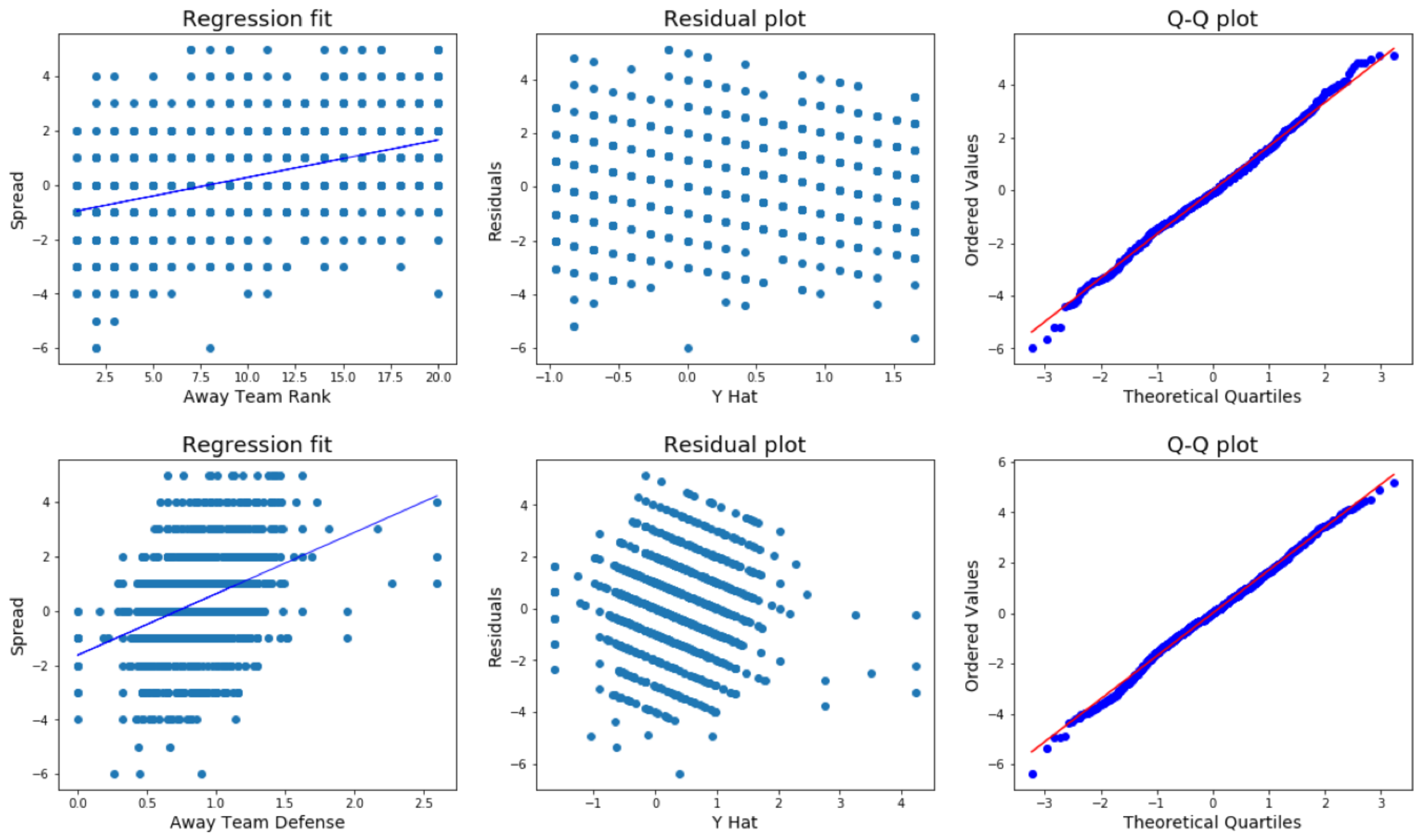
Code for reference in jupyter notebook **"Project 2 - EPL Spread Prediction - Code "**: It is rather long and clunky but it got the job done. Very open to suggestions for improvement.

Top 3 graphs

show Team rank based on position in the league with **LITTLE PREDICTIVE POWER** but normally distributed

Bottom 3 graphs

show after feature engineering Defensive and Attack features per team with **MUCH MORE PREDICTIVE WEIGHT**

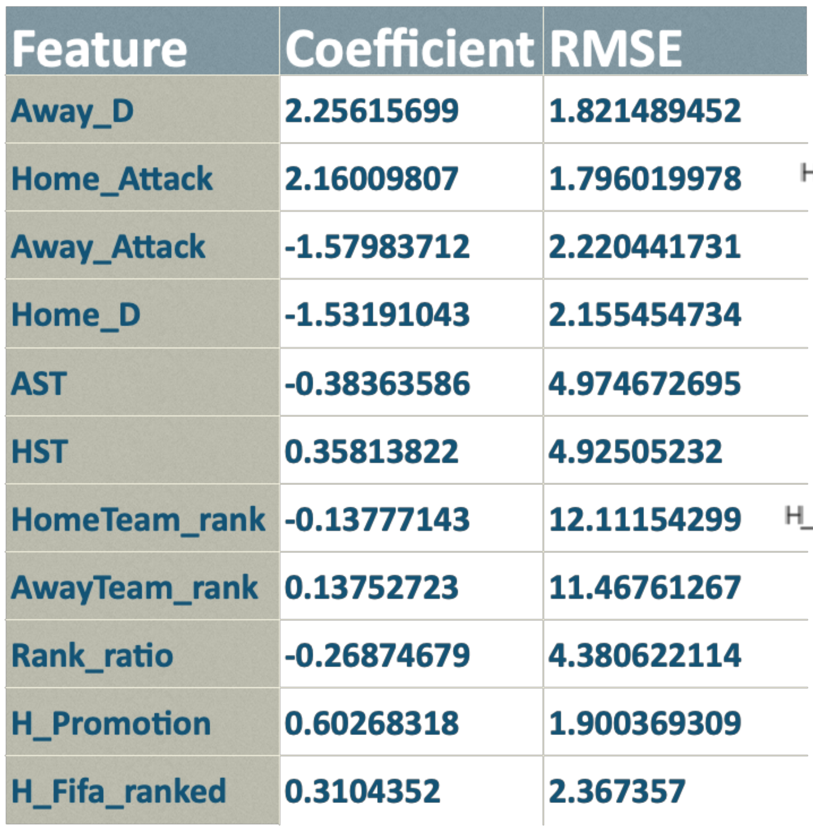


I had my Four engineered features which I knew to be my best baseline predictors from my Forward Feature selection plotting and one by one testing. Meaning I placed each potential feature into an OLS regression model and recorded the Coefficients RMSE MAE and Intercept.

I chose RMSE because I found it the best indicator for feature value and reducing the error and variance of the model.

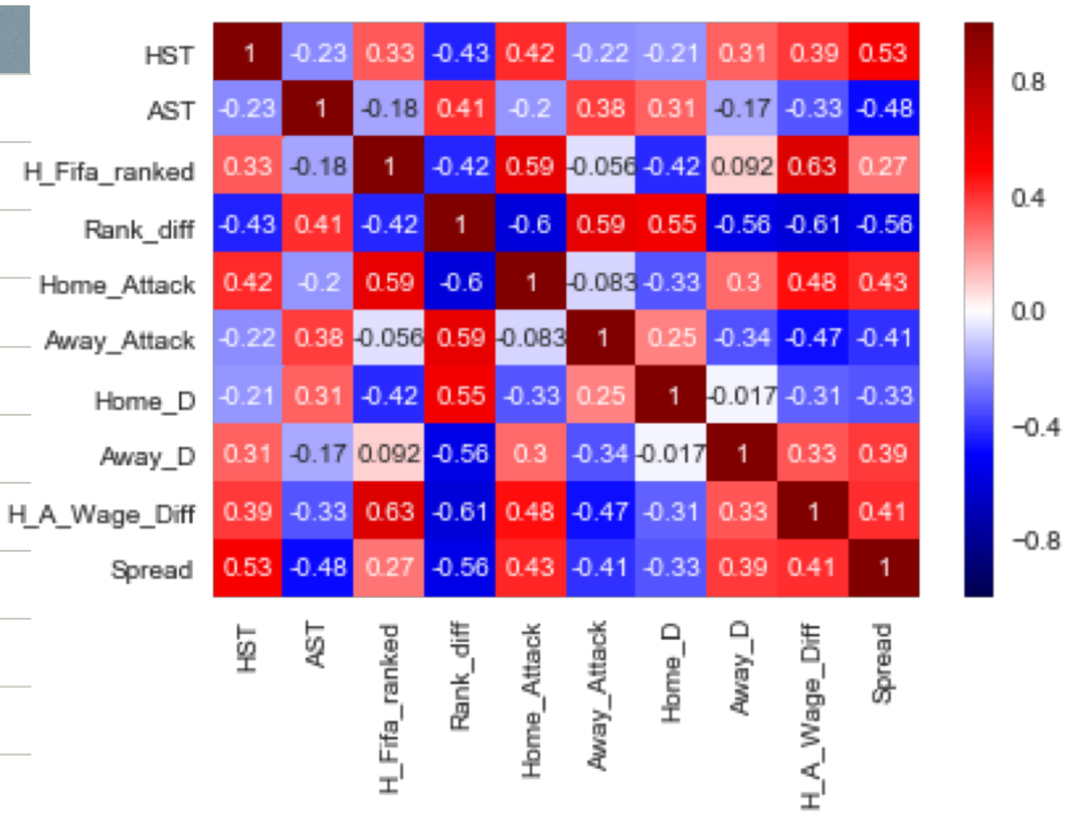
See appendix slide below Again the HomeTeam_rank and AwayTeam_rank variables and after engineering into the top four features.

Feature Selection Table



Feature	Coefficient	RMSE
Away_D	2.25615699	1.821489452
Home_Attack	2.16009807	1.796019978
Away_Attack	-1.57983712	2.220441731
Home_D	-1.53191043	2.155454734
AST	-0.38363586	4.974672695
HST	0.35813822	4.92505232
HomeTeam_rank	-0.13777143	12.11154299
AwayTeam_rank	0.13752723	11.46761267
Rank_ratio	-0.26874679	4.380622114
H_Promotion	0.60268318	1.900369309
H_Fifa_ranked	0.3104352	2.367357

Feature Selection Heat Map



Feature engineering done, time to model

I had a group of 16 features with varying importance from the RMSE MAE and Coefficients + my own intution, **Now time to model**

I placed my top 4 features into the **OLS model to get a baseline for my R^2 and RMSE** going forward before other models. R^2 was .48 up substantially from my first of .23 when I ran with my initial not web scraped and non feature engineered data.

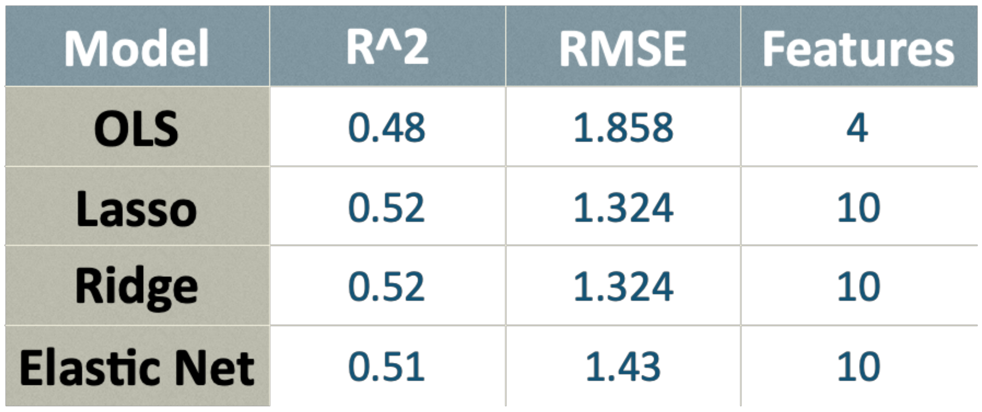
Then I **ran a Lasso with Cross Validation to find a basis point for my alpha** to then use Lasso for some further feature selection tuning the alpha up from the suggested low to zero out features and determine my best fits.

This got my to about 8-12 best features down from 16 using only the training data.

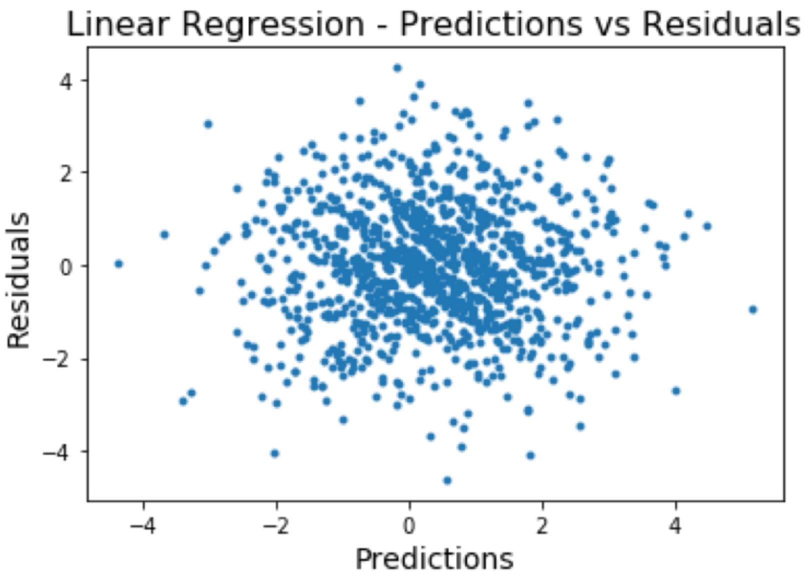
Train Test Split the data I set up 3 models Ridge with Cross Validation, Lasso with Cross Validation and Elastic Net Each model would see 3 feature sets, 8 features, 10 features and 12 features and record scores with Cross Validation alphas and apply each model to the training validation and test sets. **Applied standard scalers** to the data for each model

I did perform a **log10 transform** to one feature I had constructed but ultimately it was collinear with another feature which proved a better fit for the model so the log10 transformed feature was left out.

I **plotted each model's Predictions vs Residuals** to check for random noise meaning the realtionship between features and Residuals(Yhat-Y) and thus Y values is reasonably linear. It suggests error variances are equal because the points are generally around the horizontal line at zero. And it does show some outliers that occur but after data cleaning I determined those are true occurences. ie Man City beating Stoke 7-2 Oct 2017



Model	R^2	RMSE	Features
OLS	0.48	1.858	4
Lasso	0.52	1.324	10
Ridge	0.52	1.324	10
Elastic Net	0.51	1.43	10

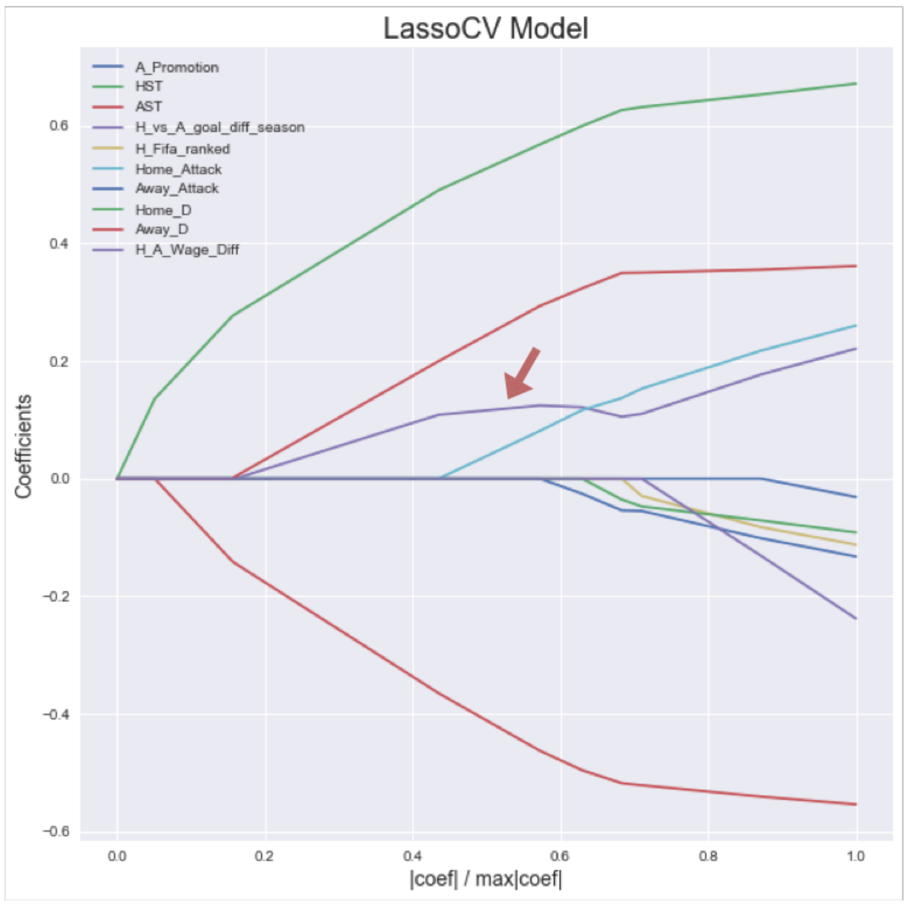


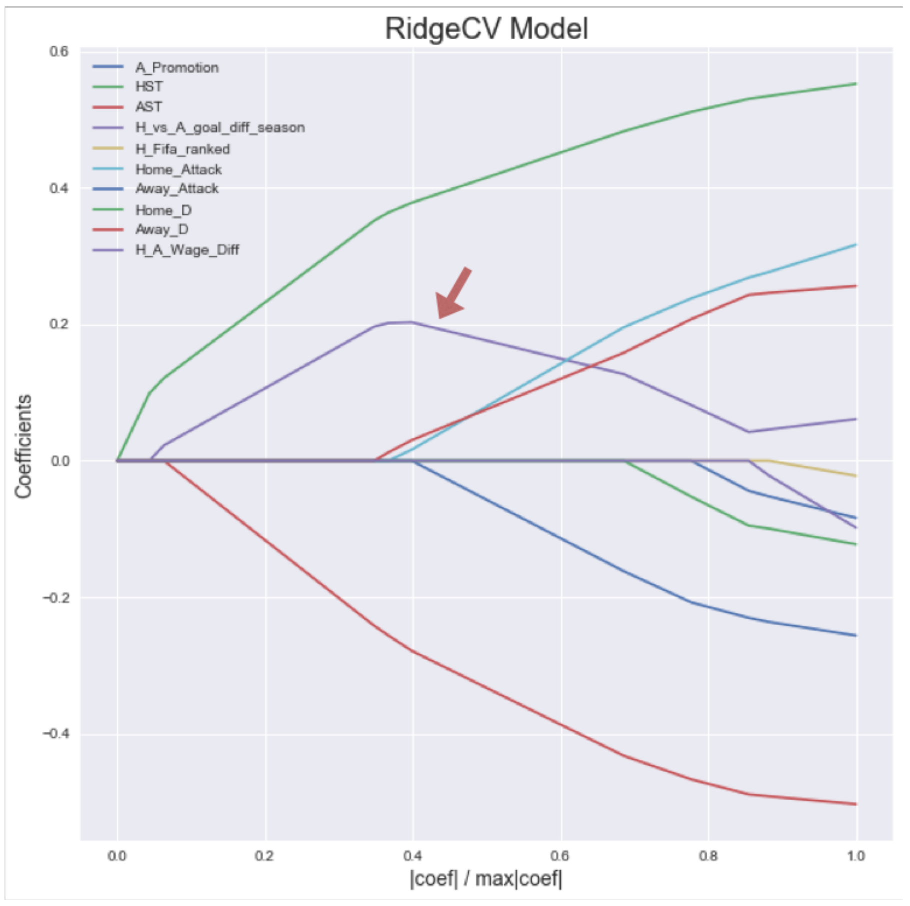
The Lasso and Ridge performed very similar over the cross validation trials and nearly identical in the final models.

And when I compared the plot of each to see the features I saw **Ridge** kept the (Home - Away wages) feature with a low coeffient value early that spikes higher as other variables get zeroed out. It feels suspect to use the Ridge with a coefficient value that starts very low and then has much more sway later on.

Lasso zeroed it out sooner and had the two features I engineered Away D and Home D as the most important variables to carry to end the for prediction.

Overall I think at this stage I like the **Lasso Model is best** because: 1) Lasso has a slightly lower RMSE, here the better model has the least error. 2) Lasso emphasizes the Defensive measures nearly eaqual and in sports the old adage "Defense Wins Championships" proves true time and time again.





Lasso and Ridge with 10 features + scaler transform R^2 .523 RMSE 1.324

Conclusion

So, can this model predict the spread of a match in the EPL?

Not well with an average error of 1.3 goals off the mark but I think it did allow me to find some interesting insights like the importance of defense to positive results much more so than even precise attacking play and the wages paid to players by a club.

Lastly I found the promotion sides (teams promoted to the top league) are much more successful at home than away and top fifa ranked players follow the same pattern, producing little effective difference away from home but show a decent effective weight when playing in front of their home crowd.

Future Work

1) For future work I would like to add more seasons to train the model on more data and see how it performs.

2) Also I would like to engineer more features from the raw stats that could help increase the performance of the model.

3) Lastly I would like to explore more model types and a more Bayesian perspective, try to use weighted probabilities that could help with a different result.

Thank you!

Michael Weber