NFL Point Spread Prediction

Team composition:
Duy PHAM (dmpham1@wpi.edu) Wen Chiang LIM (wlim@wpi.edu)
Kevin METZLER (kjmetzler@wpi.edu) Daniel FOX (dcfox1@wpi.edu)

April 24, 2023

1. Recap of problem and dataset

In this project, we aim to predict the outcomes of American National Football League (NFL) games by estimating the final score point difference (Visitor score minus Home score). There are avid football fans in the project team, and this allows the team to have fun while working on something interesting for the project. A member of the team scraped the dataset from profootballreference.com as part of his previous hobby and included features on (a) environment (e.g., temperature, wind speed and humidity) and (b) game statistics (e.g., passing yards, rushing yards). While the original dataset featured 12,828 games from 1970 to 2022, we removed data from earlier years to start from 1991 instead as some features were only available after 1991 and there were several rule changes that meant interpretation of games in much earlier years may have a difference in meaning to the games in recent years.

1.1 Data Visualizations

Pairplots comparing the dependent variable of interest ("score difference") and the various independent variables were examined (see Appendix). While this did not include dummy coded variables such as type of surfaces or stadiums played or the day of the week, it allowed a quick investigation of what are likely important features for the prediction model as well as allowing for fine-tuning of the model if required.

2. Two Perspectives to problem

We approached the problem of prediction of score point differences from two perspectives:

a. [Section 3] In-game prediction of final score gap based on game statistics, and environmental factors
b. [Section 4] Future game prediction of final score gap based on past game statistics.

This two-pronged approach was due in part to the need for understanding the features that matter to the results of a game and, more importantly, the appreciation for the challenge and difficulty in predicting future games.

Both approaches involve prediction of a continuous variable (final score gap) and would thus largely revolve around regression-based methods.

3. In-game prediction of final score

Games statistics tell a vivid story of how a game went, but more importantly, they provide an indication of what are the more important statistics to focus on in game play. This would potentially be of value to NFL teams who would need to strategize their game play as well as identifying key players who could perform better at important metrics, as popularized in the movie "MoneyBall"[1].

3.1 Pre-Processing

To analyze the dataset, we performed minor data processing to ensure there was no missing data, by providing sensible data imputation (e.g., 0 wind speed for games inside domes, or average attendance for games with missing attendance). We also removed several columns that were either redundant or had information about the final score. For example, there were originally columns for Date, Day of the Week, Year, Week of the Season, and Time, so we removed the date block and kept the Year and Week of the Season, as well as adding one-hot encoding for the day of the week. Additionally, we found that some data that we wanted to use wasn't collected until after 1991 such as the Rush Attempts and Attendance, which is one reason why we only consider games taking place after 1991 for this Project.

Several other notable changes made to the data before beginning our analysis are that, in some cases, the attendance was blank for the 2020 Season and thus we filled those with zeros since fans were not allowed to attend due to COVID-19. We also had decided to one-hot-encode all the Teams for both home and away, but many of the teams had changed names or cities in the time between 1991 and 2023. To accomplish this, as part of his previous hobby, Dan also provided a three-letter franchise mapping code. Each team name was thus mapped to a continuous franchise as teams moved from city to city or changed their city or franchise name. For example, the Cleveland Browns moved to Baltimore and became the Baltimore Ravens. Two years later, an expansion team picked up the mantle of Browns in Cleveland, thus the original Browns are coded as 'BAL,' the same code as the Ravens, and the new Browns have their own code, 'CLE'. It may seem odd for the old Cleveland Browns to be coded as 'BAL,' but we feel it is appropriate because the team today is in Baltimore.

3.2 Best approach: Full model linear regression

Our best approach based on the R^2 and the MSE scores was the Full Model Linear Regression (FMLR), although many came quite close such as the Tuned Lasso, Tuned Ridge, Tuned Model Tree, and the Full Model after removing outliers from the features. The main difference is a lower Validation MSE for the FMLR at 24.52 whereas the difference in R^2 for the listed models here is negligible around 0.88 with the highest Validation R^2 in the FMLR at 0.8832599.

Since this is a Linear Regression, we can look at exactly what the coefficients of each variable are. It is important to note here that we are predicting the difference in score between the Visitor and Home, so a positive coefficient will indicate more points to the visiting team, and a negative coefficient will indicate more points to the home team.

[1] https://en.wikipedia.org/wiki/Moneyball_(film)

First off, the Intercept of the FMLR is –22, indicating that the Home Team has an advantage. Next, the variables with the largest influence towards the Home Score are Home Passing Touchdowns with –3.3, and Home Rush Touchdowns with –3.1, which are both obvious because they directly lead to points. For the Visitor Score, these two are also the two most important with +2.985 per Passing Touchdown and +3.02 per Rushing touchdown. What is interesting here is that a touchdown is worth 6 points so we should expect these to be plus or minus 6 each, but because there are three other ways to score points in the NFL that weren't recorded in the dataset thus throwing off these variables and having to put weight into less obvious ones that may correspond with a field goal or a safety such as the number of fourth downs a team has gotten to.

Some other large coefficients that we can observe are Baltimore being the home team, adding -1.67 to the Score Difference. Similarly, if New England is the home team, they add -1.44 to the Score Difference, and if Green Bay is the home team, they add -1.08 to the Score Difference. On the other hand, when New England is the visiting team, they increase the Score Difference by +1.49 on average. Interestingly, when the New Orleans Saints are the home team, they change the Score Difference by -1.04, meaning they provide an advantage to the opposite team.

A few more major factors are the number of Turnovers that each side gives up, pulling the Difference in Score away from them by about 1.5 per turnover. Lastly, the day of the week had a stronger effect than we were expecting in the favor of the Home Score by changing the Score Difference by -1.26 if the game was played on a Friday, and -1.18 if the game was played on a Tuesday.

Several of the features were negligible for the model, which I will interpret here. These are all features with an influence less than 0.01. Home rushing yards, temperature, attendance, the game taking place on a Wednesday, the visitor time of possession, the visitor rushing yards, Windchill, and the year. It surprised us that attendance was so non-influential because Dan had noticed in his original run through the data that the home-field advantage shrank during covid because there were no fans. Home-field advantage is already likely covered by the home and visiting team features that the remaining score difference that was left to be explained by attendance was most likely negligible. The temperature and windchill are low because teams are likely to train in conditions like the average day that they are going to play between October and February. We had initially predicted that this was going to play a larger role because if a team from Florida was playing in Seattle in December; they were likely to be at a disadvantage, but like with attendance, this is wrapped into the team one-hot-encoding as well. For the rushing yards I would have thought that these would be more significant, around the lines of 6 points per 100 yards rushed, but their coefficients suggest about 1 point per 1000 yards rushed in a game. Now this could be because there are other metrics for the touchdowns scored from rushing discussed earlier, or it could be because in most scenarios, the home rushing yards and visiting rushing yards cancel out.

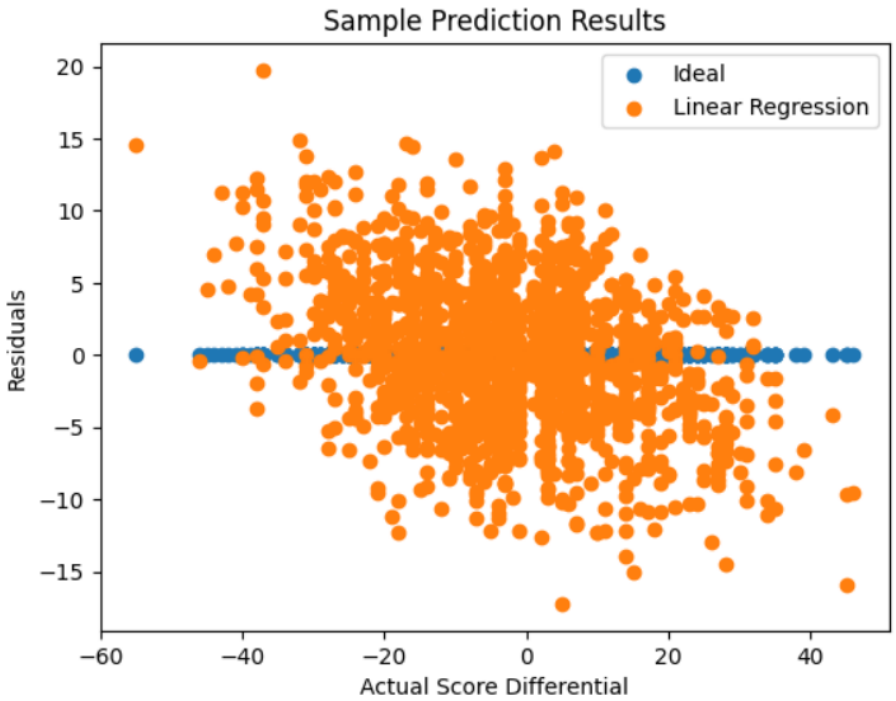


Figure 1: Residual Plot for the FMLR.

While this model does very well with an R^2 or about 0.883 and an MSE of about 24.5, we noticed that the residual plot is skewed so that it performs quite poorly on the edges of the Score Differences. We will explore our attempts to fix this in section 3.3.

3.3 Other approaches attempted

The team also attempted several other methods:

Method	**Test R2**	**Test MSE**
Full model linear regression	0.8832599	24.52052
Random Forest (Max Depth = 10)	0.7640823	49.62753
Lasso	0.6252612	78.80247
Lasso (tuned)	0.8832190	26.45348
Ridge (tuned)	0.8829164	26.52203
Model tree (tuned)	0.8817807	26.77928
Full model linear regression after removal of outliers	0.8826767	26.57632
PCA on Linear Regression	0.6041471	83.22631
Y:=Y+2*(Y)^(1/3) Transformation	0.8855100*	39.22493*
Sigmoid Transformation	0.8695056*	98.44117*

*Due to the complexity of the transformations, it was unfeasible to invert this transformation for the calculation of the MSE and R^2. The values here indicate the comparison between the prediction and a transformed test set.
We elaborate below on some of the more interesting approaches.

3.3.1 Transformation

As noted at the end of Section 3.2, the Linear model performs poorly for the edges of the Score Differences. When observing the data distribution, it was clear that our dataset was not normally distributed. In Figure 2, we see that a vast majority of games ended closer to a tie than not, but notably few games (one or two per season, on average) end in a tie, because of an overtime period, so there is a dip at 0.

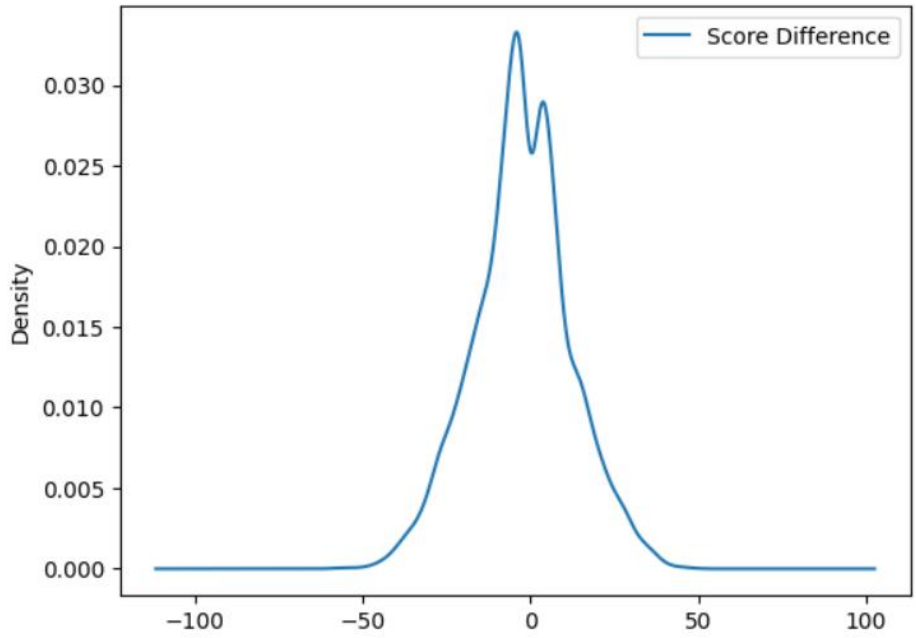


Figure 2: Density Plot of the Score Difference among all the games.

Our goal from here is to transform the data or modify it in some way such that our models can handle all ranges of the data and not just the central points. We decided on two different approaches to try, the first being a transformation of the data that would spread out the values more, and the second being removing the outliers from the data so that the model can at least make better predictions on the data we keep.

The First transformation that we tried on the test set was a sigmoid shape, with the goal of having a steeper function around a Score Difference of zero so that they spread out more, and a shallower curve towards the edges so that those stay about where they are. This would preserve the ordering but distort the magnitude of the values. After some mathematical analysis, the formula we settled on was:

$$Y_{new} = \frac{98}{1+e^{-0.1 \cdot Y_{old}}} - 49.$$

The Selection of these coefficients considers the max and min of the Score Differences and makes sure that these are still about the same in the new Y output. One downside to this transformation is that it has an upper and lower bound at plus or minus 49, but the Linear Model is still prone to

predicting outside this range, meaning that we could not realistically invert these predictions for comparison to the actual values of the set. To handle this, we decided it would be best to Transform the test set as well and compare the predictions based on transformed training data to the actual transformed values. This makes the MSE obsolete since the magnitudes are not preserved, but the R^2 is still relevant and produced a value of about 0.87, slightly below the un-transformed Linear Model that we used.

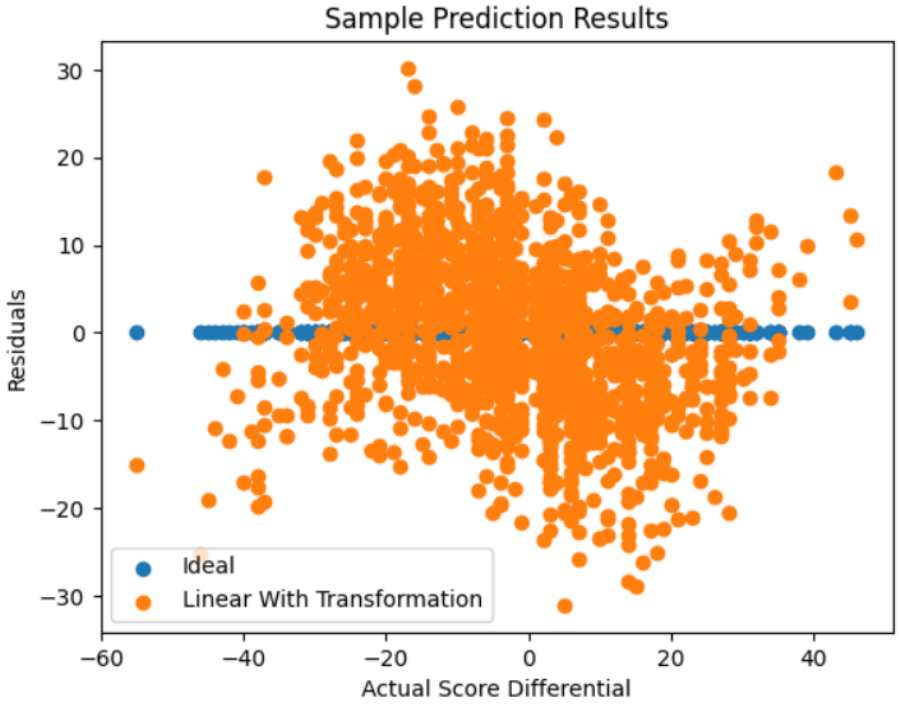


Figure 3: Residual Plot of the Sigmoid Transformed Data.

The Second Transformation we tried was a much simpler one that uses the Cube Root function to achieve a similar shape to the sigmoid, but without the horizontal asymptotes. What is nice about this is that for larger values, this transformation looks quite like the identity function, as the main feature of this is at and near zero. The equation for this transformation is:

$$Y_{new} = Y_{old} + 2\sqrt[3]{Y_{old}}.$$

While this formula is simpler, it is still not quite feasible to invert, so we used the same trick with the transformed test data to find the R^2 which comes to about 0.885 slightly higher than the untransformed data, but a statistically negligible difference.

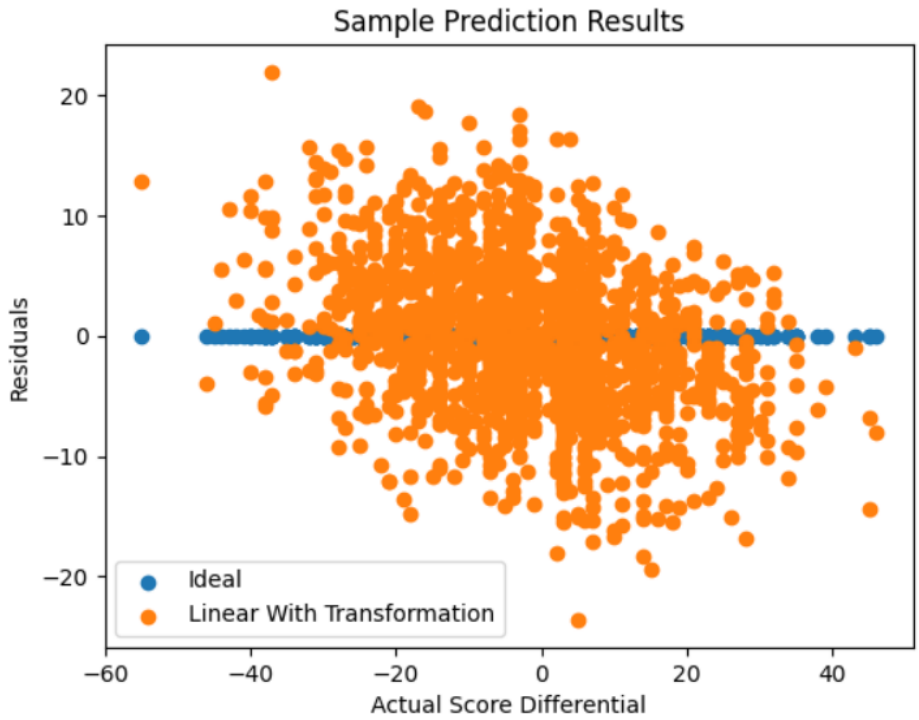
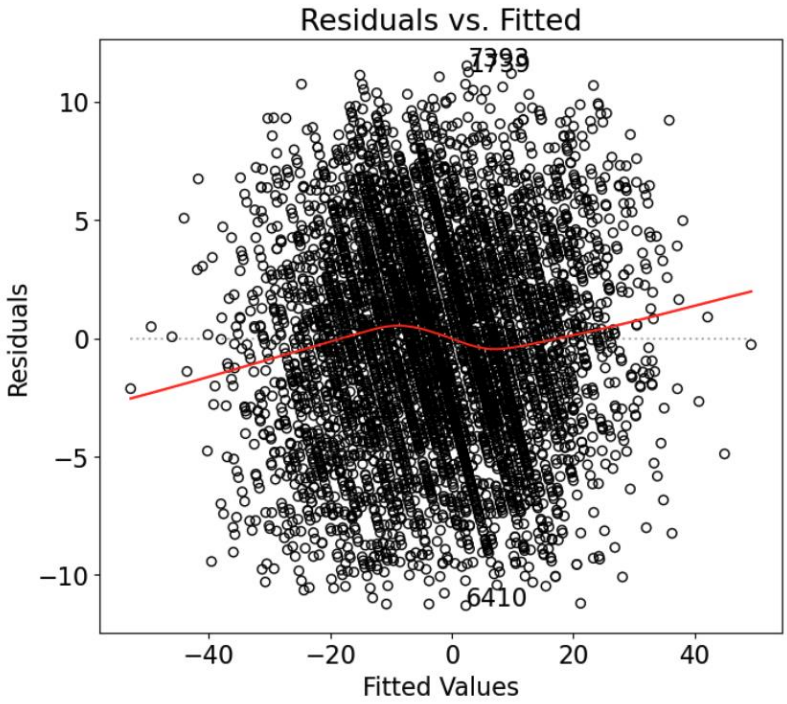


Figure 4: Residual Plot for the Cube Root Transformation.

3.3.2 Dropping of influential outliers

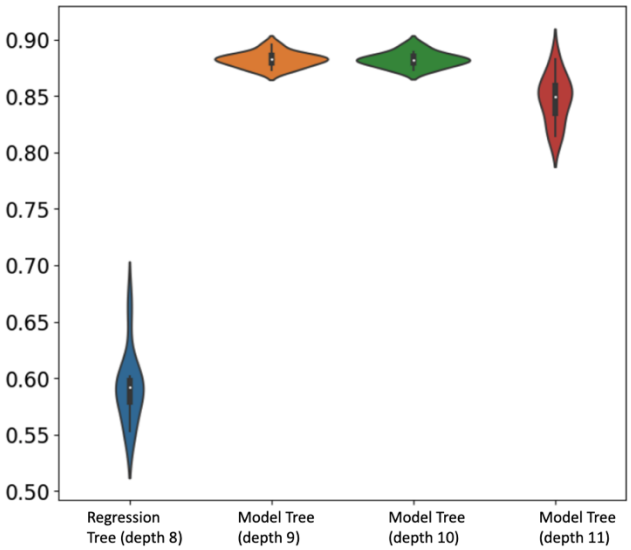
We noticed from the pairplots and the residual plots from linear regression that there were outliers in the data. As such, we attempted to remove influential outliers from the training data based on cook's distance, before fitting the linear regression. To operationalize this, we dropped points where cook's distance was greater than 4/number of data points in the training set.

Although training R2 (0.909) was now much better and the residual plot looks better than the original plot, the test R2 and MSE were still slightly poorer than that of full model linear regression (see table above).



3.3.3 Regression Tree and Model Tree

We also tried fitting a regression tree as a baseline and used a model tree. This was a method that could potentially combine regression tree with linear regression where the leaf nodes of the regression tree were not a constant but a linear regression. Using cross validation, the model tree obtained by Depth 9 and 10 after smoothing and pruning were pretty consistent in having high R2 and low training MSE. The violin plot below shows the consistently high R2 for depth 9 and 10 before dropping when we used depth 11.



On investigation, depth 9 and 10 model trees had only one leaf node (i.e., a linear regression). It was only in depth 11 which provided a more complex tree, which had slightly poorer test R2 and MSE. Nonetheless, we note the results were all slightly worse than that of the full linear regression.

Output from depth 9 and depth 10, and an example of output from the depth 11 model tree is provided in the appendix. Given that the output from depth 9 and depth 10 for a model tree was a linear regression for a single leaf node, it supported our finding that a full model linear regression was a good model for in-game prediction of the final point differential.

4. Future-game prediction of the final point differential

Predicting many aspects of the outcome of future games is essential to the $80 billion sports-betting industry, and betting on the NFL is the highest portion of that industry in the USA. First, it is essential that the oddsmakers, by and large, set the initial odds for games, because the house can lose a lot of money if those odds are not balanced with the final outcomes. And although odss-makers can adjust the odds if there is such an imbalance, the "damage" has already been done prior to the adjustment. As an example, say a point spread has been set at 3 points favoring the home team with even odds, but a lot of money might be bet in favor of the home team winning by more than 3 points (known as "covering the spread") with very little money bet in favor of the visiting team either winning or losing by less than 3 points. Now envision this happening in a lot of games. If the public is *more accurate* than the gambling establishments (known in the business as "the house") and many of those bets have to be payed out, the house would lose money and go out of business.

Therefore, the initial odds set by the house being as close as possible to the actual outcome is critical to the success of their business. Yes, the house doesn't need to be very accurate on individual games, but they do need to be at least as accurate, if not more, than the betting public in aggregate. There might be betters out there that have an algorithm that is more accurate than the oddsmakers. That is okay so long as the betting public, as a whole, is not more accurate.

4.1 Data preprocessing

To predict future games, a great deal of preprocessing of the data is required.

4.1.1 Predictor

In summary, there is one variable that digests the relative past performance of the visiting team with respect to the home team's past performance. It is the average point differential in past games, not including overtime, of the visiting team minus the same average of the home team. We made adjustments for games played in previous seasons (a hyper parameter weighting factor), and there is a limit to how many games in the past are factored into this statistic (also a hyperparameter). The rationale for this statistic is that past performance can be a predictor of future performance.

Also, there are 32 variables, one for each team, the value of which is the number of games the visiting team has played against that team minus the number of games the home team has played against that team. Again, the hyperparameters of weighting factor for previous seasons and the

number of games over which to compute each statistic are considered. Because of the weighting factor, these values may not be whole numbers. The rationale for these statistics is that the strength of teams played may skew the predictive power of past performance. Since the NFL has a very imbalanced schedule (each team plays three teams twice, and only plays once against 11 of the remaining 28 teams), some teams may end up with a much tougher schedule than others.

The details of these 33 variables, including the equations that determine them, are in the appendix.

4.1.2 Hyper Parameters

There are three hyperparameters that factor into these variables.

N, the number of previous games played by each team that are considered. The rationale for this hyperparameter is that we presume that only the most "recent" games have predictive power.

W, the weighting factor for games played in previous seasons. Games played in the current season have a weight of $W^0 = 1.0$. Games in the previous season have a weight of W. Games in the season before that one have a weight of W^2, and games in the season before that one have a weight of W^3, and so on. The rationale for this hyperparameter is that we presume that games played in previous seasons have less predictive power than ones played in the later seasons, due to the many changes from one season to the next, especially personnel changes.

S, the starting week for each season. Only games played in weeks S through to the end of the season are represented in X. The rationale for this hyperparameter is that we presume games earlier in the season are much harder to predict because of the scarcity of lookback games in the current season. And if this is not the case, then S should take on a value of 1.

4.1.3 Target

We have chosen the target, y, as the final point differential from the point-of-view of the visiting team (visiting team final score minus home team final score). This choice is largely based on the data that we have, which is the Vegas Oddsmakers' point differential prediction for each game played. These odds are set such that there should be an equal chance of the favorite winning by at least that point differential.

4.2 Model Performance

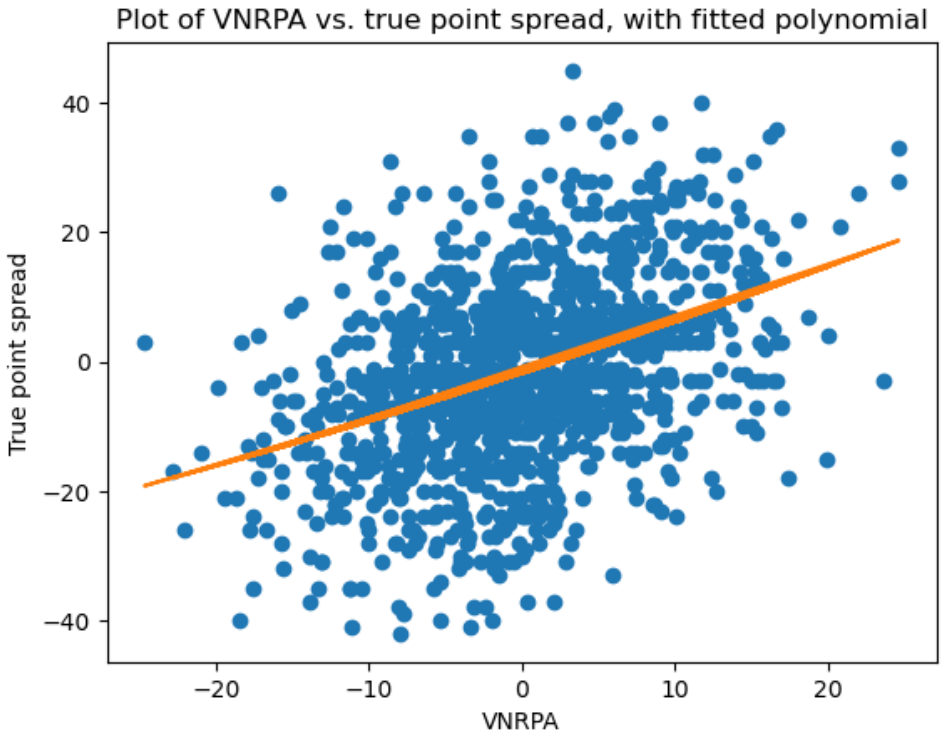
The test set for predicting NFL future games were the seasons 2017 through 2022. It was not difficult to calculate how our models performed against both the Vegas Odds and 538's ELO model. One metric used to evaluate predicted probabilities is the Brier Score, which is a function of predicted probabilities against the true outcome. Here we use 538's formula for converting an NFL predicted point spread into win probabilities:
$P_{win} = \frac{1}{\left((10^{-ps/16})+1\right)}$, where P_{win} is the probability of the favorite winning the game, and ps is the predicted positive point spread in the game.
If ps is negative, then Pwin is lower than 0.5, representing the probability of the underdog winning the game.

The Brier Score is effectively the mean squared error of the forecast, so the lower the better. Here are the scores of our various models, along with 538's ELO model, and that of the Vegas Odds, in descending order of Brier Score:

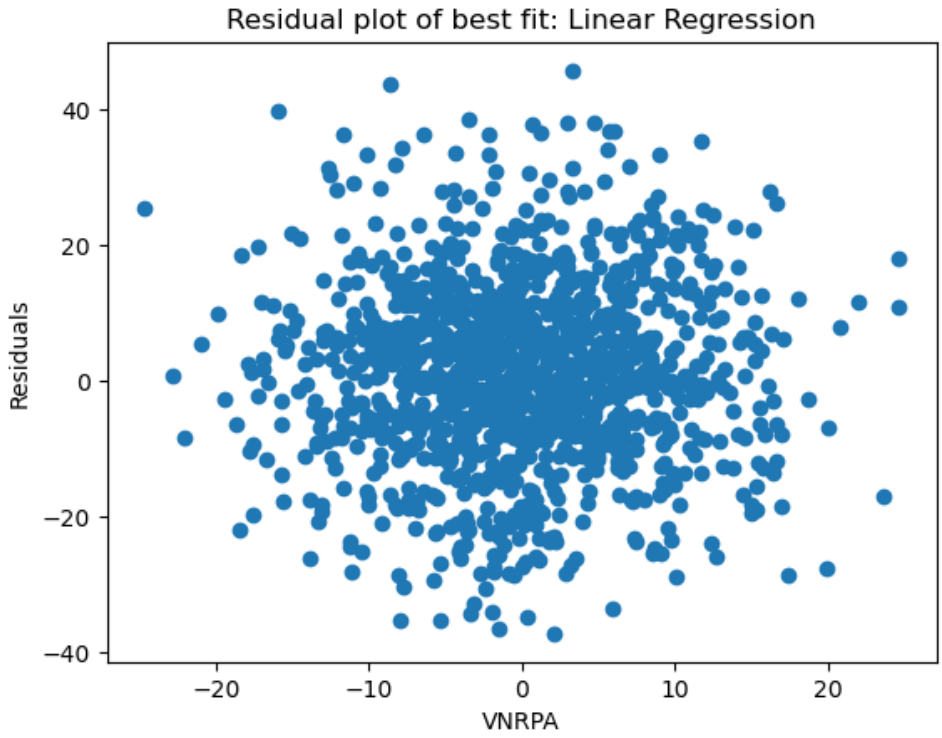
Method	Root Mean Squared Error	Mean Absolute Error	Brier Score
Vegas Odds	13.28	9.97	0.207
Linear Regression	13.29	10.30	0.219
Ridge	13.32	10.33	0.220
Random Forest	13.28	10.37	0.221
Support Vector Regressor	13.44	10.43	0.223
538 ELO	13.33	10.33	0.224
K-nearest Neighbors	14.17	11.12	0.245

You see that all but one of our models performed better than 538's ELO model, but none have done as well as the Vegas Odds.

We declare linear regression as the best model we have because of the lowest mean absolute error. Also, the following plot shows a decent linear relationship between VNRPA and the point spread outcome. Note that the fitted line is actually a degree-2 polynomial, but the coefficient for the squared component was negligible.



Here is the plot of the residuals against the main statistic, VNRPA:



We note that there is no discernable pattern in the residuals plot, lending credibility to the model.

Since the best model was linear regression, we feel it would be informative to display the coefficients here:

Feature	**Coefficient**
intercept	-2.834306
IsNeutral	7.278556
Playoffs	-1.783569
VNRPA	0.726479
MIN	-0.867163
NEP	-0.762036
SFF	-0.649918
PIT	-0.564060
NYG	-0.532154
IND	-0.511185
NOS	-0.497442
DEN	-0.496493
KCC	-0.486838

ARI	-0.471277
GBP	-0.463309
CAR	-0.390883
BAL	-0.359157
WAS	-0.350404
TEN	-0.277473
SEA	-0.222385
CIN	-0.180843
HOU	-0.169710
LAC	-0.157068
PHI	-0.154669
ATL	-0.126589
NYJ	-0.118093
LAR	-0.083770
LVR	-0.020785
DET	-0.013137
TBB	0.004281
BUF	0.024732
JAC	0.075328
CLE	0.419654
MIA	0.558725
CHI	0.565357

The interesting things to note is: 1) the intercept is clearly the visiting team's field disadvantage, which is inline with was Vegas oddsmakers and 538 use; 2) the visiting team has an extra disadvantage in the playoff, and, indeed, 538 has noted that the home team (usually the favorite) has more of an advantage in the playoffs; 3) we designed VNRPA so that it should have a coefficient of close to 1.00 if no other factors were at play, but it is actually 0.73, which suggests strong significant of the other variables; 4) the better historical teams have higher negative values as a factor for strength of schedule. We note that for 4), a model that considered the strength of the team at the time that they were played could contribute to better accuracy, and that is reserved for future work.

4.2.1 Betting Strategy

We have observed that it is best to bet on games where the model's predicted point spread differs from the Vegas Odds by more than 6 points. Although Vegas Odds appear to be more efficient in general, when there is a large discrepancy, it is difficult to reason that the Vegas Odds are superior.

It is important to note that gambling has high frictional costs. In general, when you bet against the point spread, you are putting up $115 to win $100. Thus, doing some math shows that you must consistently predict against the spread correctly by 53.5%.

It is also important to note that the fewer games for which there are this large point differential discrepancy, either you must bet more on each game or accept smaller gains. We assume in the following table that a person would put up $115 to win $100 for all games that have this large discrepancy.

The following table describes how such a betting strategy would have performed over the seasons from 2017 to 2022.

Method	Correct vs. Vegas	Incorrect	Accuracy	Dollars Gained	Dollars Invested	ROI
Random Forest	55	38	0.5914	1130	10695	10.57%
Support Vector Regressor	95	78	0.5491	530	19895	2.66%
Ridge	71	59	0.5462	315	14950	2.11%
Linear Regression	66	56	0.5410	160	14030	1.14%
K-Nearest Neighbors	220	187	0.5405	495	46805	1.06%

The authors would like to point out this is speculative, and the only good way to judge the models would be to apply it to future seasons, and probably over the next five or six. Since it is reasonable to assume that Vegas odds will only continue to get more and more efficient in their predictions as time goes on, we realize that the model that performed best over the previous six seasons may lose its efficacy in the future. And, of course, the strongest model we found in terms of mean absolute error was the Linear Regression, which only has an ROI of 1.14%. This suggests that the ROI for the Random Forest model is suspect.

5. Cross-Validation Considerations

To use our training data most effectively, we wanted to use Cross-Validation and report the mean values of the metrics. However, because the data is a time series, seasonality and trend patterns might be present or emerge – violating the independent and identically distributed (i.i.d.) assumption. More intuitively, training folds can potentially expose the models to patterns present in hold-out folds. The models will essentially cheat the evaluation process as such. For business, it also makes little sense to use past values to predict future ones.

To overcome this, we performed Cross-Validation on a rolling basis. We split the training data into K+1 sequential, equal portions. However, instead of training the models with every fold except the hold-out at each iteration, we subsequently added training data. The first fold was used to evaluate the second fold, and their combined data was used to evaluate the third. This also reflects modeling in businesses, with models being re-fitted with new data coming in.

We employed time-series cross-validation for parameter (and hyperparameter) tuning for the various models tried. We used a 5-fold model (K=5), which divided up the training set into six partitions, 0 through 5.

Partition 0: Seasons 1993 through 1996
Partition 1: Seasons 1997 through 2000
Partition 2: Seasons 2001 through 2004
Partition 3: Seasons 2005 through 2008
Partition 4: Seasons 2009 through 2012
Partition 5: Seasons 2013 through 2016

The five validations were done with the following training and validation sets:
CV1: training – partition 0; validation – partition 1
CV2: training – partitions 0-1; validation – partition 2
CV3: training – partitions 0-2; validation – partition 3
CV4: training – partitions 0-3; validation – partition 4
CV5: training – partitions 0-4; validation – partition 5

We selected he "winning" parameters by using the average mean absolute error (MAE) of all five validations. R-squared was so low, and mean squared error so high, that in all cases that we felt using either would not gain us much. The mean absolute error makes sense because we are predicting the scoring differential and comparing our predictions to the Vegas odds, which is not a squared relationship. To put it another way, the fit to the actual point differentials is poor, to say the least. But the goal is to do better than other methods out there that predict the point differentials.

Method	Hyper Parameters	Parameters	MAE CV Mean
Linear regression	N=28, W=0.3, S=5	Defaults	10.632
Ridge	N=32, W=0.3, S=5	alpha=10, tol=0.00001	10.625
Support vector regressor	N=41, W=0.2, S=5	defaults	10.605
K-nearest neighbors	N=36, W=0.2, S=5	n_neighbors=15, p=1	10.846
Random forest	N=41, W=0.2, S=5	max_depth=3, n_estimators=100	10.526

6. Conclusion

Future game prediction is much harder than in-game prediction. If the objective is to learn what is important in winning games, it is easier. But if the aim is to beat betting odds, it is much more difficult considering the information asymmetry with Vegas betting.

6.1 In-Game Predictions

Given the number of various methods tested, it was surprising that linear regression provided the best fit in terms of test MSE and R2, even after the various attempts to focus on the weak areas of

linear regression (transformation, deletion of outliers). It was interesting how even the model tree method provided a single-leaf node linear regression when we were training up to a depth 9 and 10, providing further evidence how a linear regression was a good model for this context. Even subsequent attempts to include quadratic terms were only able to improve the test R2 and test MSE very marginally, suggesting our model is "saturated" as is.

We acknowledge there are things we could still have tried, such as ensemble methods to focus on the outlier points that were not estimated well by our model. This can be possible future work that can be embarked on to improve on the model fit. Nonetheless, given the already high R2 and low MSE for our existing model, it was very likely we would not have gotten much higher improvement unless a lot more time was spent on finetuning the model.

6.2 Future Game Predictions

Predicting future game point differentials is very difficult because the variance in the error terms is so high. While good teams consistently perform well, and bad teams consistently poor, the performance of all teams in individual games fluctuates wildly. Therefore, we expect all metrics to reflect a poor performance. Moreover, while the data offers a lot of information about what happens in the games, it says nothing about what happens outside of them. For example, rosters could change on a seasonal basis as managers acquire new players or trade old ones. They could also change from game to game due to injuries, suspensions, or other issues. This leads to a lot of variation in a team's strength over time – not accounting for factors like budgets for training and coaching.

But while overall performance against game outcomes is desirable, what is more important is how any model performs against other prominent models, as well as the Vegas odds.
We believe that in recent times, as well as into the future, the Vegas odds are becoming more and more efficient. By efficient, we mean coming as close to predicting the outcome of the game. Yet still, as we demonstrate below, the Vegas Odds are still well off on individual outcomes.
One prominent model out there is the ELO model developed by a website called fivethirtyeight.com (538 is the number of electoral votes for President of the United States, and the site originally had just one model for predicting the outcome of the US presidential race based on adjusted polling averages). They do not claim to be better than Vegas Odds, but they do claim to come close.

We are encouraged that we did better than the very popular fivethirtyeight.com.

6.2.2 Future Work

There is a lot of future work that could be applied to a model that predicts future games. As we noted above, a model that considered the strength of a team at the time they were played could

help the efficacy. More historical variables could be added to the model, although we note that the cross-validation performed worse when net yardage, over the course of N games with a weighting factor, W, in previous seasons, was added as a variable. It was not clear to us that adding any more statistics would help, but admittedly, we only attempted net yardage. Certainly, more parameter tuning could help, and perhaps someone could try other algorithms.

7. Group Dynamics:

During the project, the team met five times and was able to split the work up effectively before meeting to consolidate ideas and plan.

a. Duy:
- Prepared final project proposal
- Processed dataset to allow time-series modeling
- Reviewed future game prediction methodologies
- Future game prediction modelling
- Worked on final project report and presentation

b. Dan:
- Prepared initial writeup
- Scraped dataset and updated any discrepancies and inaccuracies
- Processed dataset to allow time-series modeling
- Developed lookback variables
- Reviewed future game prediction methodologies
- Future game prediction modelling
- Worked on final project report and presentation

c. Kevin:
- Prepared initial writeup
- Final project proposal presentation
- Processed dataset for analysis for in-game prediction
- In-game prediction modelling using full model linear regression, random forest, lasso, transformations
- Worked on final project report and presentation

d. Wen Chiang Ivan:
- Prepared initial writeup
- Data visualisations for in-game prediction
- In-game prediction modelling using full model linear regression, ridge, lasso, removal of outliers, model tree
- Structured final report and presentation

- Worked on final project report and presentation

8. Appendix

8.1 Future Game Prediction Variable Equations

8.1.1 Predictor

The predictor, X, that is constructed for predicting future games is the same list of games (observations) as in the raw data set, but though we added many variables, we deleted most.

Every row in X refers to a particular game played in the past. Rows are in the order of the games played, with the first row being the earliest game played and the final row being the most recent game played. The "current game" refers to the game in the specified row being populated. The “ith previous game played by team 1" refers to the game played i games ago by team 1. Note this is not the game that is i rows back.

8.1.2 Deleted variables

All but two variables from the original data set eventually get dropped out of the new X. We dropped variables that contain information about the future (usually in-game statistics), for the obvious reason. We dropped other pre-game environmental variables to simplify the model--future efforts may include them. Finally, we dropped identification information about the game (Year/Season designation, Week, Date Played, Home Team, Visiting Team, Home Score, Visitor Score, etc.) after that identifying information was used to construct other predictive variables in X.

8.1.3 Carried-over variables

Variables that remain in the new X, unchanged from the data set, are 'Playoffs' (1=game is a playoff game, 0=not) and 'IsNeutral' (1=games played at a neutral site, 0=not).

8.1.4 Added lookback variables

33 variables that we call *lookback variables* have been added. L*ookback variables* are a function of past games played by both home and visiting teams and are always determined from the visiting team's point-of-view. That is, values that contribute to each "lookback variable" are positive for the visiting team, and negative for the home team.

8.1.5 Hyperparameters

There are three hyperparameters that factor into *lookback variables:*

N, the number of previous games played by each team that are considered. The rationale for this hyperparameter is that we presume that only the most "recent" games have predictive power.

W, the weighting factor for games played in previous seasons. Games played in the current season have a weight of $W^0 = 1.0$. Games in the previous season have a weight of W. Games in the season

before that one have a weight of W^2, and games in the season before that one have a weight of W^3, and so on. The rationale for this hyper parameter is that we presume that games played in previous seasons have less predictive power than ones played in the later seasons, because of the many changes from one season to the next, especially personnel changes.

S, the starting week for each season. Only games played in weeks S through to the end of the season are represented in X. The rationale for this hyper parameter is that we presume games earlier in the season are much harder to predict because of the scarcity of lookback games in the current season. And if this is not the case, then S should take on a value of 1.

8.1.6 Lookback variable 1

VNRPA (visitor net regulation points average). VNRPA is an amalgamation of the regulation scores (overtime points are not included) obtained by both teams over the previous N games. Essentially, VNRPA is the average point differential of the visiting team over the past N games minus the average point differential of the home team over the past N games. These point differentials are weighted according to what season each game was played during. The rationale for this variable is that past performance can be a predictor of future performance.

It is calculated using the following formulas:

The visiting team of the current game is designated as team 1, and the home team as team 2. We use these designations so as not to confuse them with the home and visiting teams for any past game (which we effectively ignore).

$g1_i$ is an object that represents the ith previous game (i varies from 0 to N) played by team 1, and $g2_i$ represents the ith previous game played by team 2. Example: $g1_{13}$ represents the game played team 1 13 games ago. $g1_0$ and $g2_0$ both represent the current game.

$ts(g1_i)$ equals team 1's score (points in overtime do not count) in game $g1_i$, and $ts(g2_i)$ equals team 2's score (points in overtime do not count) in game $g2_i$. Example: if team 1 scored 7 points (no overtime played) 13 games ago, then $ts(g1_{13}) = 7$. Example: if team 2 scored 20 points 8 games ago, but scored 6 points in overtime, then $ts(g1_8) = 14$.

$os(g1_i)$ represents team 1's opponents' score in game $g1_i$, and $os(g2_i)$ represents team 2's opponents' score in game $g2_i$. Example: if team 2's opponents scored 20 points (no overtime played) 9 games ago, then $os(g2_9)$.

$s(g1_i)$ equals the season during which game $g1_i$ was played, and $s(g2_i)$ equals the season during which game $g2_i$ was played. s takes on a numerical value equal to the first year that games were played in that season, which lasts from September to February. $s(g1_o)$ and $s(g2_o)$ both equal the season of the current game. Example: if $g1_6$ was played in October of 2017, then $s(g1_6) = 2017$. Example: if $g2_6$ was played in January 2019, then $s(g2_6) = 2018$.

$$\text{VNRPA} = \frac{\sum_{i=1}^{N}\left(W^{\left(s(g1_i)-s(g1_0)\right)}\right)\left(ts(g1_i)-os(g1_i)\right)}{\sum_{i=1}^{N}\left(W^{\left(s(g1_i)-s(g1_0)\right)}\right)} - \frac{\sum_{i=1}^{N}\left(W^{\left(s(g2_i)-s(g2_0)\right)}\right)\left(ts(g2_i)-os(g2_i)\right)}{\sum_{i=1}^{N}\left(W^{\left(s(g2_i)-s(g2_0)\right)}\right)}$$

8.1.7 Lookback variables 2 through 33

Each of the two teams faced different opponents over the past N games, and these 32 variables account for this. Every team is a variable of how many games team 1 has played against that team (weighted by season) minus how many games team 2 has played against that team (weighted by season). The rationale for these variables, as stated above, is to correct for the skew in past performance based on strength of schedule.

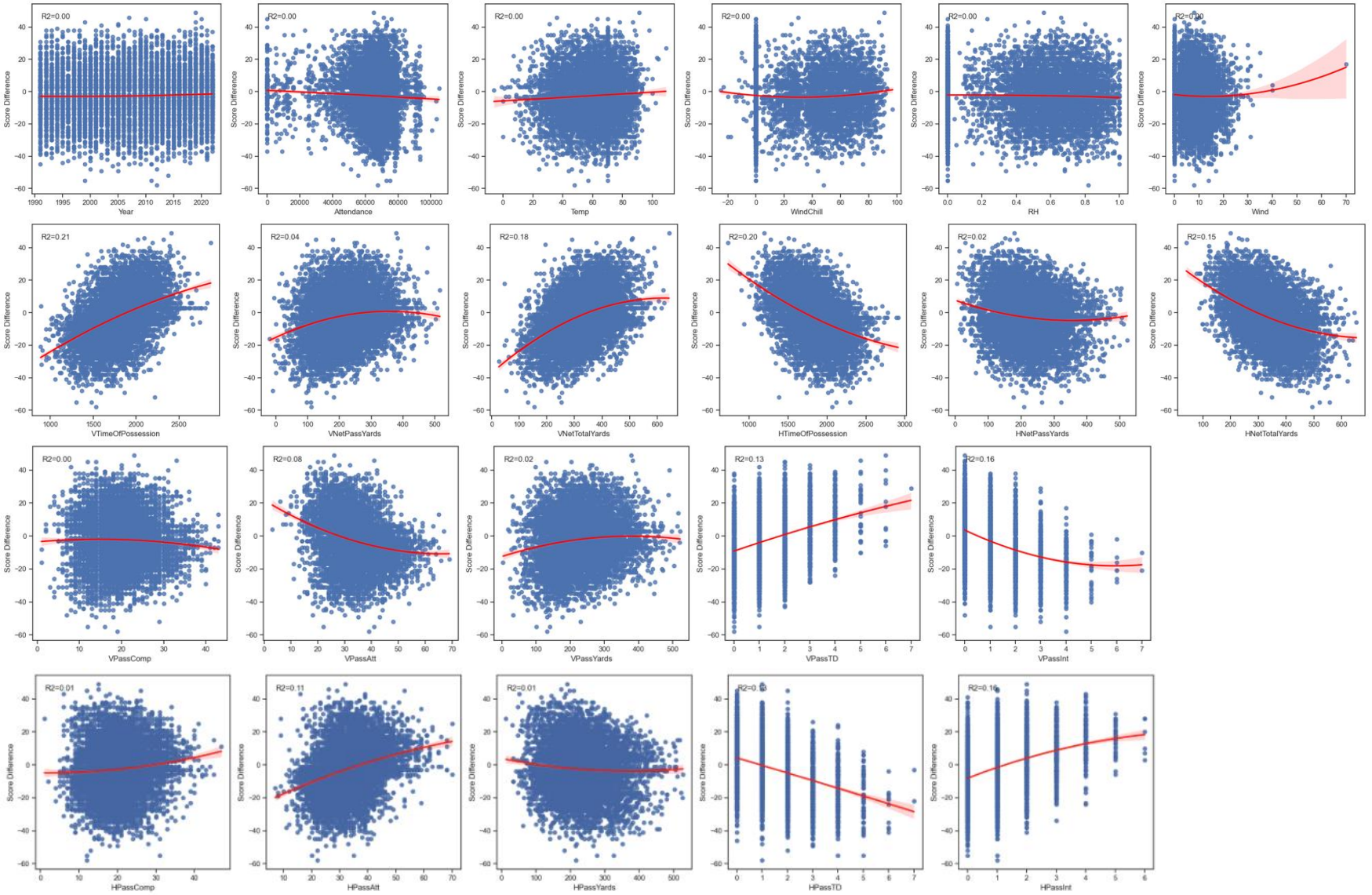
Let TEAM be a column name that designates a particular team. For example, DEN is the column name for the Denver Broncos.

Let gp(g_i) be 1.0 if team i played game g_i against team TEAM, and 0.0 otherwise.

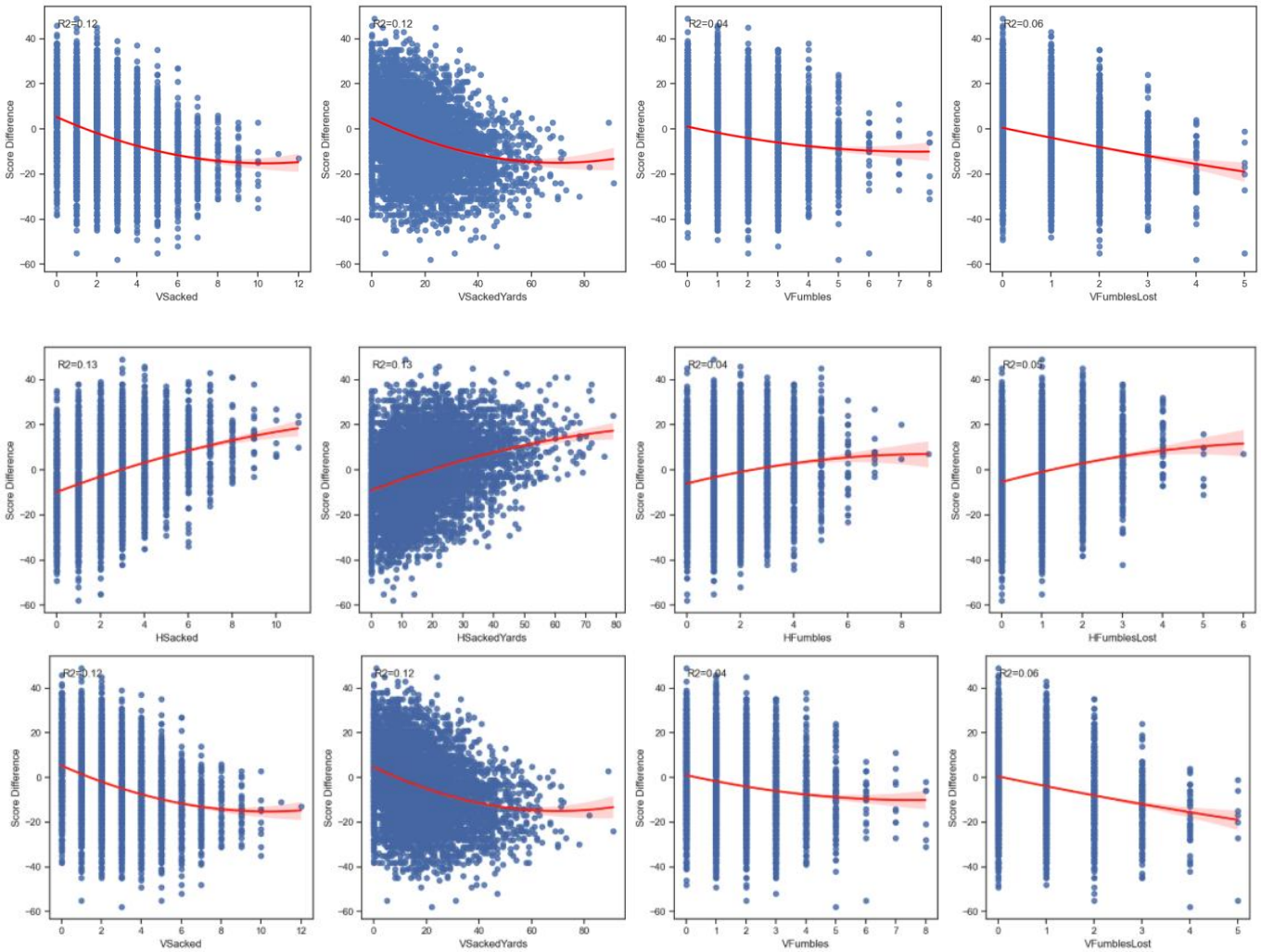
The value in the TEAM column for a training/predicted game is:

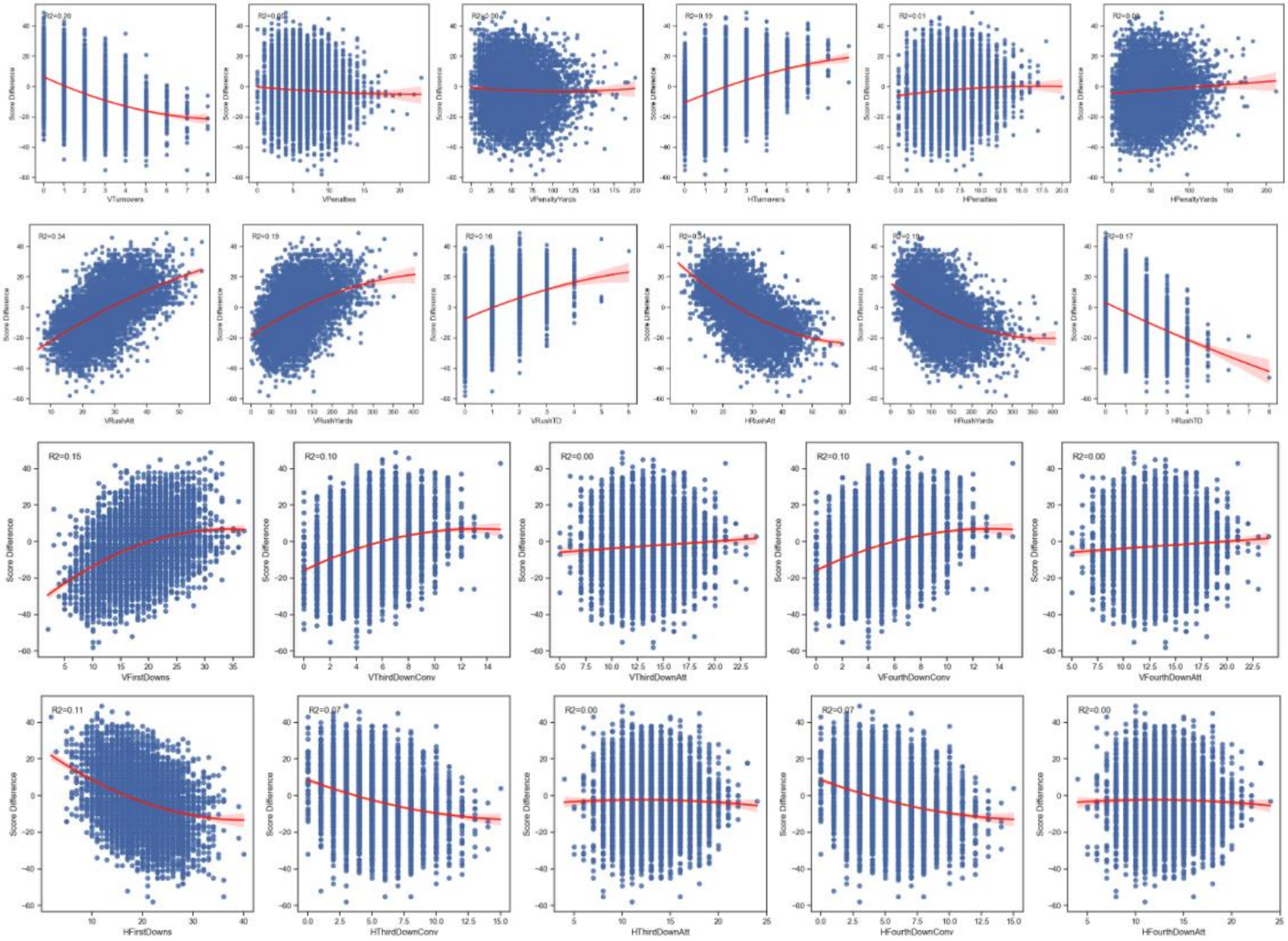
$$\text{TEAM} = \sum_{i=1}^{N}\left(W^{\left(s(g1_i)-s(g1_0)\right)}\right)gp(g1_i) - \sum_{i=1}^{N}\left(W^{\left(s(g2_i)-s(g2_0)\right)}\right)gp(g2_i)$$

Appendix


R2=0.00
Score Difference
Year
R2=0.00
Attendance
R2=0.00
Temp
R2=0.00
WindChill
R2=0.00
RH
R2=0.00
Wind
R2=0.21
VTimeOfPossession
R2=0.04
VNetPassYards
R2=0.18
VNetTotalYards
R2=0.20
HTimeOfPossession
R2=0.02
HNetPassYards
R2=0.15
HNetTotalYards
R2=0.00
VPassComp
R2=0.08
VPassAtt
R2=0.02
VPassYards
R2=0.13
VPassTD
R2=0.16
VPassInt
R2=0.01
HPassComp
R2=0.11
HPassAtt
R2=0.01
HPassYards
HPassTD
HPassInt

Appendix


R2=0.12
Score Difference
VSacked
R2=0.12
Score Difference
VSackedYards
R2=0.04
Score Difference
VFumbles
R2=0.06
Score Difference
VFumblesLost
R2=0.13
Score Difference
HSacked
R2=0.13
Score Difference
HSackedYards
R2=0.04
Score Difference
HFumbles
R2=0.05
Score Difference
HFumblesLost
R2=0.12
Score Difference
VSacked
R2=0.12
Score Difference
VSackedYards
R2=0.04
Score Difference
VFumbles
R2=0.06
Score Difference
VFumblesLost


R2=0.20
VTurnovers
R2=0.00
VPenalties
R2=0.00
VPenaltyYards
R2=0.19
HTurnovers
R2=0.01
HPenalties
R2=0.00
HPenaltyYards
R2=0.34
VRushAtt
R2=0.19
VRushYards
R2=0.16
VRushTD
R2=0.34
HRushAtt
R2=0.19
HRushYards
R2=0.17
HRushTD
R2=0.15
VFirstDowns
R2=0.10
VThirdDownConv
R2=0.00
VThirdDownAtt
R2=0.10
VFourthDownConv
R2=0.00
VFourthDownAtt
R2=0.11
HFirstDowns
R2=0.07
HThirdDownConv
R2=0.00
HThirdDownAtt
R2=0.07
HFourthDownConv
R2=0.00
HFourthDownAtt
Score Difference

Results of Model Tree

Depth 9 (smoothed, pruned)	Depth 10 (smoothed, pruned)
----- ModelTree (smooth True, prune True, max depth: 9) M5Prime (pre-smoothed with constant 15): LEAF [friedman_mse=206.477, samples=6641] : LM1 (err=4.874, params=59) LM1: 1.641e-01 * Year - 2.753e-01 * Week - 1.316e-02 * Attendance - 9.612e-02 * Temp + 2.246e-01 * WindChill + 4.757e-01 * RH - 5.261e-02 * Wind + 13.237 * VFirstDowns - 4.738 * VRushAtt - 95072.35 * VRushYards + 18.371 * VRushTD + 5.272 * VPassComp - 25.071 * VPassAtt - 77208.09 * VPassYards + 21.366 * VPassTD + 158193.69 * VPassInt + 13558.165 * VSacked - 46465.824 * VSackedYards + 145569.61 * VNetPassYards + 3.567e-02 * VNetTotalYards + 112993.9 * VFumbles - 180809.72 * VFumblesLost - 5.518e-01 * VTurnovers - 4.889 * VPenalties + 1305.894 * VPenaltyYards + 36942.18 * VThirdDownConv - 1296.602 * VThirdDownAtt - 36948.184 * VFourthDownConv + 4.527e-01 * VFourthDownAtt - 11.788 * VTimeOfPossession + 10.302 * HFirstDowns - 95098.75 * HRushAtt - 24.743 * HRushYards - 3.711 * HRushTD + 24.889 * HPassComp + 62481.344 * HPassAtt - 23.141 * HPassYards + 771404.1 * HPassTD + 3.523 * HPassInt - 9599.5 * HSacked - 184189.53 * HSackedYards + 144281.78 * HNetPassYards + 1.797 * HNetTotalYards + 771403.3 * HFumbles - 1028520.75 * HFumblesLost + 4.258e-01 * HTurnovers + 5.267 * HPenalties - 296527.12 * HPenaltyYards + 288653.0 * HThirdDownConv + 296517.84 * HThirdDownAtt - 288647.66 * HFourthDownConv - 3.039 * HFourthDownAtt + 8.447e-02 * HTimeOfPossession + 2.344e-02 * DayOfWeek_Friday - 2.285e-01 * DayOfWeek_Monday - 1.594 * DayOfWeek_Saturday - 7.812e-03 * DayOfWeek_Sunday + 7.617e-02 * DayOfWeek_Thursday + 15096.932	----- ModelTree (smooth False, prune True, max depth: 10) M5Prime (pre-smoothed with constant 15): LEAF [friedman_mse=206.477, samples=6641] : LM1 (err=4.865, params=68) LM1: 2.125e-01 * Year - 1.960e-01 * Week - 1.400e-01 * Attendance + 1.314e-01 * Temp - 4.340e-02 * WindChill + 1.969e-01 * RH - 2.575e-01 * Wind + 14.312 * VFirstDowns - 6.154 * VRushAtt + 77843.36 * VRushYards + 18.195 * VRushTD + 5.356 * VPassComp - 26.851 * VPassAtt - 12748.89 * VPassYards + 21.441 * VPassTD + 298808.84 * VPassInt - 3.858 * VSacked + 2237.038 * VSackedYards + 116512.46 * VNetPassYards - 119167.23 * VNetTotalYards - 1.855e-01 * VFumbles + 213433.38 * VFumblesLost - 341512.03 * VTurnovers - 4.738e-01 * VPenalties - 4.867 * VPenaltyYards - 79821.484 * VThirdDownConv + 59738.016 * VThirdDownAtt + 79829.83 * VFourthDownConv - 59742.457 * VFourthDownAtt + 1.259 * VTimeOfPossession - 12.125 * HFirstDowns + 10.945 * HRushAtt + 15619.812 * HRushYards - 24.578 * HRushTD - 3.594 * HPassComp + 25.406 * HPassAtt + 320482.38 * HPassYards - 23.039 * HPassTD - 80834.44 * HPassInt + 3.422 * HSacked - 49253.062 * HSackedYards - 301781.53 * HNetPassYards - 23714.764 * HNetTotalYards + 1.215 * HFumbles - 80834.44 * HFumblesLost + 107797.3 * HTurnovers + 2.812e-01 * HPenalties + 5.156 * HPenaltyYards + 222227.53 * HThirdDownConv + 39312.633 * HThirdDownAtt - 222236.53 * HFourthDownConv - 39307.992 * HFourthDownAtt - 2.672 * HTimeOfPossession + 7.812e-02 * DayOfWeek_Friday - 1.797e-01 * DayOfWeek_Monday + 7.812e-02 * DayOfWeek_Saturday + 5.625e-01 * DayOfWeek_Sunday - 0.000e+00 * DayOfWeek_Thursday - 2.031e-01 * DayOfWeek_Tuesday - 3.906e-01 * DayOfWeek_Wednesday - 3.203e-01 * Roof_dome + 5.156e-01 * Roof_outdoors - 1.643 * Roof_retractable roof (closed) + 3.125e-01 * Roof_retractable roof (open) - 6.250e-02 * Surface_a_turf + 4.297e-01 * Surface_astroplay + 1.719e-01 * Surface_astroturf - 5902.45

Depth 11 (smoothed, pruned) Subset of model

```
----- ModelTree (smooth True, prune True, max depth: 11) M5Prime (pre-smoothed with constant 15):

VRushAtt <= 0.402 [friedman_mse=206.477, samples=6641] (err=4.792, params=6012)
|  HRushAtt <= 0.427 [friedman_mse=147.231, samples=3419] (err=4.798, params=5946)
|  |  HTurnovers <= 0.188 [friedman_mse=139.197, samples=1095] (err=4.662, params=5886)
|  |  |  VTurnovers <= 0.188 [friedman_mse=115.91, samples=621] (err=4.702, params=3229)
|  |  |  |  VPassTD <= 0.357 [friedman_mse=92.366, samples=319] (err=4.344, params=1003)
|  |  |  |  |  HNetTotalYards <= 0.569 [friedman_mse=77.837, samples=250] (err=4.382, params=476)
|  |  |  |  |  |  VNetTotalYards <= 0.545 [friedman_mse=65.495, samples=167] (err=4.246, params=26)
```

| | | | | | | LEAF [friedman_mse=57.527, samples=116] : LM1 (err=4.392, params=12)
| | | | | | | HNetTotalYards <= 0.397 [friedman_mse=49.696, samples=51] (err=3.894, params=13)
| | | | | | | HPassComp <= 0.402 [friedman_mse=88.512, samples=11] (err=6.655, params=5)
| | | | | | | | LEAF [friedman_mse=19.25, samples=6] : LM2 (err=3.604, params=2)
| | | | | | | | LEAF [friedman_mse=53.76, samples=5] : LM3 (err=9.047, params=2)
| | | | | | | Year <= 0.694 [friedman_mse=20.44, samples=40] (err=2.674, params=7)
| | | | | | | LEAF [friedman_mse=15.349, samples=18] : LM4 (err=3.076, params=3)
| | | | | | | LEAF [friedman_mse=15.225, samples=22] : LM5 (err=2.293, params=3)

LM1: 7.193e-01 * Year - 1.367e-01 * Week - 1.449e-02 * Attendance - 2.223e-01 * Temp - 9.853e-03 * WindChill + 1.743e-02 * RH + 14.101 * Wind + 1.451 * VFirstDowns - 1.053 * VRushAtt + 101482.89 * VRushYards + 2.638 * VRushTD - 2.493e-01 * VPassComp - 39.135 * VPassAtt - 2492.74 * VPassYards + 4.44 * VPassTD - 68675.97 * VPassInt - 5.431e-01 * VSacked + 437.402 * VSackedYards + 137330.53 * VNetPassYards - 155368.73 * VNetTotalYards - 8.547e-03 * VFumbles - 49054.406 * VFumblesLost + 78485.29 * VTurnovers + 1.071e-01 * VPenalties - 9.171e-01 * VPenaltyYards - 2163.215 * VThirdDownConv + 49479.54 * VThirdDownAtt + 2165.576 * VFourthDownConv - 49485.117 * VFourthDownAtt + 9.589e-01 * VTimeOfPossession - 3.026e-01 * HFirstDowns - 29.05 * HRushAtt + 25815.629 * HRushYards - 2.876 * HRushTD - 9.392 * HPassComp + 22.357 * HPassAtt + 22352.232 * HPassYards - 3.006 * HPassTD + 363.456 * HPassInt + 4.071e-01 * HSacked - 3438.825 * HSackedYards + 10508.833 * HNetPassYards - 39172.785 * HNetTotalYards + 1.431e-01 * HFumbles + 363.398 * HFumblesLost - 483.83 * HTurnovers - 1.591e-01 * HPenalties + 1.244e-01 * HPenaltyYards + 210494.25 * HThirdDownConv + 14008.072 * HThirdDownAtt - 210508.86 * HFourthDownConv - 13993.78 * HFourthDownAtt + 7.289e-02 * HTimeOfPossession + 4.778e-04 * DayOfWeek_Friday + 3.757e-02 * DayOfWeek_Monday + 4.683e-03 * DayOfWeek_Saturday - 5.803e-03 * DayOfWeek_Sunday + 2.202e-03 * DayOfWeek_Thursday - 1.842e-03 * DayOfWeek_Tuesday - 2.704e-03 * DayOfWeek_Wednesday - 1.096e-03 * Roof_dome + 4.877e-01 * Roof_outdoors - 8.803e-03 * Roof_retractable roof (closed) - 1.036 * Roof_retractable roof (open) + 5.119e-05 * Surface_a_turf - 3.117e-02 * Surface_astroplay - 6.463e-02 * Surface_astroturf + 2.013e-03 * Surface_dessograss - 4.394e-04 * Surface_fieldturf + 2.252e-03 * Surface_grass + 2.771e-02 * Surface_matrixturf - 12852.807

LM2: 2.401 * Year - 1.367e-01 * Week - 1.449e-02 * Attendance - 2.223e-01 * Temp - 9.853e-03 * WindChill + 1.743e-02 * RH + 5.524 * Wind + 1.451 * VFirstDowns - 1.053 * VRushAtt + 101482.89 * VRushYards + 2.638 * VRushTD - 2.493e-01 * VPassComp - 11.398 * VPassAtt - 2492.74 * VPassYards + 4.44 * VPassTD - 68675.97 * VPassInt - 5.431e-01 * VSacked + 437.402 * VSackedYards + 137304.78 * VNetPassYards - 155367.58 * VNetTotalYards - 8.547e-03 * VFumbles - 49054.324 * VFumblesLost + 78485.29 * VTurnovers + 1.071e-01 * VPenalties - 9.171e-01 * VPenaltyYards - 2157.905 * VThirdDownConv + 49480.055 * VThirdDownAtt + 2165.576 * VFourthDownConv - 49484.6 * VFourthDownAtt + 9.589e-01 * VTimeOfPossession - 3.026e-01 * HFirstDowns - 6.114 * HRushAtt + 25815.629 * HRushYards - 2.876 * HRushTD + 15.372 * HPassComp + 7.68 * HPassAtt + 22369.373 * HPassYards - 3.006 * HPassTD + 363.456 * HPassInt + 4.071e-01 * HSacked - 3438.825 * HSackedYards + 10508.833 * HNetPassYards - 39189.04 * HNetTotalYards + 1.431e-01 * HFumbles + 363.398 * HFumblesLost - 483.83 * HTurnovers - 1.591e-01 * HPenalties + 1.244e-01 * HPenaltyYards + 210503.98 * HThirdDownConv + 14008.072 * HThirdDownAtt - 210508.48 * HFourthDownConv - 14001.328 * HFourthDownAtt + 7.289e-02 * HTimeOfPossession + 4.778e-04 * DayOfWeek_Friday - 2.573 * DayOfWeek_Monday + 4.683e-03 * DayOfWeek_Saturday - 5.803e-03 * DayOfWeek_Sunday + 2.202e-03 * DayOfWeek_Thursday - 1.842e-03 * DayOfWeek_Tuesday - 2.704e-03 * DayOfWeek_Wednesday - 1.096e-03 * Roof_dome + 4.877e-01 * Roof_outdoors - 8.803e-03 * Roof_retractable roof (closed) - 1.036 * Roof_retractable roof (open) + 5.119e-05 * Surface_a_turf - 3.117e-02 * Surface_astroplay - 6.463e-02 * Surface_astroturf + 2.013e-03 * Surface_dessograss - 4.394e-04 * Surface_fieldturf + 2.252e-03 * Surface_grass + 2.771e-02 * Surface_matrixturf - 12865.17

LM3: 3.001 * Year - 1.367e-01 * Week - 1.449e-02 * Attendance - 2.223e-01 * Temp - 9.853e-03 * WindChill + 1.743e-02 * RH + 5.524 * Wind + 1.451 * VFirstDowns - 1.053 * VRushAtt + 101482.89 * VRushYards + 2.638 * VRushTD - 2.493e-01 * VPassComp - 11.398 * VPassAtt - 2492.74 * VPassYards + 4.44 * VPassTD - 68675.97 * VPassInt - 5.431e-01 * VSacked + 437.402 * VSackedYards + 137304.78 * VNetPassYards - 155367.58 * VNetTotalYards - 8.547e-03 * VFumbles - 49051.094 * VFumblesLost + 78485.29 * VTurnovers + 1.071e-01 * VPenalties - 9.171e-01 * VPenaltyYards - 2157.905 * VThirdDownConv + 49480.055 * VThirdDownAtt + 2165.576 * VFourthDownConv - 49484.6 * VFourthDownAtt + 9.589e-01 * VTimeOfPossession - 3.026e-01 * HFirstDowns - 6.114 * HRushAtt + 25815.629 * HRushYards - 2.876 * HRushTD + 15.956 * HPassComp + 7.68 * HPassAtt + 22369.373 * HPassYards - 3.006 * HPassTD + 363.456 * HPassInt + 4.071e-01 * HSacked - 3438.825 * HSackedYards + 10508.833 * HNetPassYards - 39189.04 * HNetTotalYards + 1.431e-01 * HFumbles + 363.398 * HFumblesLost - 483.83 * HTurnovers - 1.591e-01 * HPenalties + 1.244e-01 * HPenaltyYards + 210503.98 * HThirdDownConv + 14008.072 * HThirdDownAtt - 210508.48 * HFourthDownConv - 14001.328 * HFourthDownAtt + 7.289e-02 * HTimeOfPossession + 4.778e-04 * DayOfWeek_Friday - 2.573 * DayOfWeek_Monday + 4.683e-03 * DayOfWeek_Saturday - 5.803e-03 * DayOfWeek_Sunday + 2.202e-03 * DayOfWeek_Thursday - 1.842e-03 * DayOfWeek_Tuesday - 2.704e-03 * DayOfWeek_Wednesday - 1.096e-03 * Roof_dome + 4.877e-01 * Roof_outdoors - 8.803e-03 * Roof_retractable roof (closed) - 1.036 * Roof_retractable roof (open) + 5.119e-05 * Surface_a_turf - 3.117e-02 * Surface_astroplay - 6.463e-02 * Surface_astroturf + 2.013e-03 * Surface_dessograss - 4.394e-04 * Surface_fieldturf + 2.252e-03 * Surface_grass + 2.771e-02 * Surface_matrixturf - 12865.158