ORIE 4741 Final Project Report

December 4th, 2021

Eric Hall, Enya Zimecka, and Varun Maheshwari

Problem Statement

We are poor college students. The goal of this project is to fix that. If we can leverage data to beat the bookies, we can put actual nutrition in our ramen! Sports betting is a growing industry, and many individuals use betting to supplement their incomes (or to waste their incomes) and also just to have fun! Data can inform our betting strategies to make sure we make the smartest bets possible.

Our Data

We used data from two sources, the Pro Football Archives and Odds Portal. The Pro Football Archives has data from 1920 up to 2020, including team and seasonal stats such as defense, field goals, fumbles, interceptions, snap counts, and more for each team in the NFL. Odds Portal contains game results and betting odds for NFL games since 2008.

When we speak of betting odds, it's important to understand exactly what the numbers mean. For those of you unfamiliar with American-style moneyline betting, the remainder of this paragraph is an explanation. Negative betting odds indicate expected winners ("betting favorites"), and positive ones indicate expected losers ("underdogs"). The absolute value of negative betting odds indicates how much you would have had to bet to win $100 if you won (the bet "hits"). The absolute value of negative betting odds indicates how much you stand to win if you bet $100 and it hits. So, an odds of -420 means you would have to bet $420 to win $100 and walk away with $520. An odds of 420 means you would win $420 if you bet $100 and will then walk away with $520. There are some rare cases when there are two betting favorites playing each other and both of their odds are negative. There are even some cases when teams have the same odds! However, in most cases the bookmakers will set up the odds such that the implied probabilities sum to over 100%, leaving themselves an "edge" to make profit.

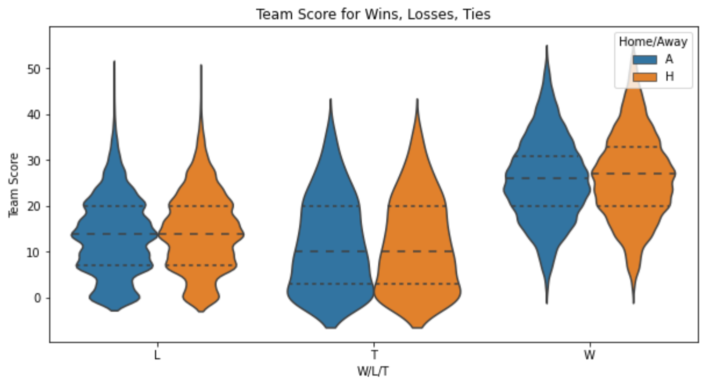
Data Exploration

Prior to running models to predict game outcomes, we did some exploration into our datasets to learn more about what they contain and to gain intuition about the data which we could leverage in our modeling.

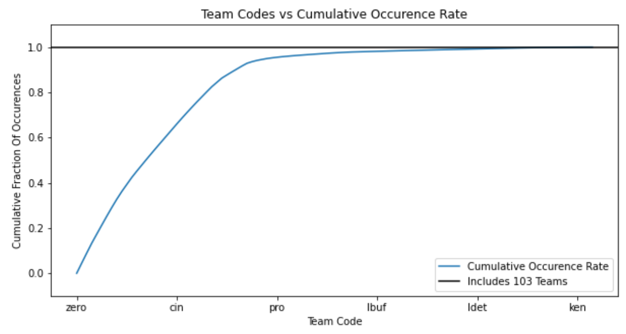
Pro Football Archives:

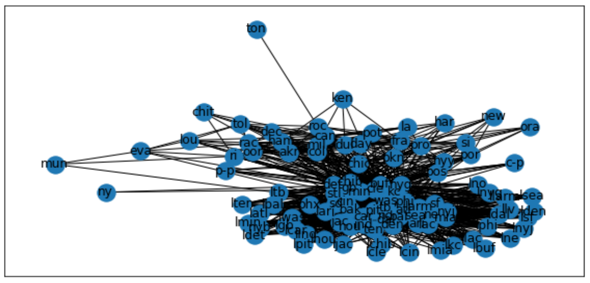
This dataset is organized by team rather than by game, with each team recording every game it plays as a home or away game, and its opponent for that game. This means that every game is represented twice in the dataset, once for the home team and once for the away team. The data columns include day of the week, the date of the game, home/away, the opponent, the results as score and win/loss/tie, overtime, location, venue, attendance, special notes, and unique codes for each team, as derived from the URL the data is scraped from. The unique codes are important, because this data runs from 1920 when the NFL was called the AFL to 2020, and contains several teams changing names—but the team code persists. There are no betting odds in this dataset, but there are some other season-granularity stats available online. However, which stats were recorded change over time, and are only available for an extremely small subset of the data, concentrated in the early 2000s, and therefore aren't especially useful for training models—we would need to sacrifice far too much data.

We first examined the occurrence rates of different team codes in the data. Some teams have been playing in the NFL since 1920, while many others have come and gone. It turns out that more modern teams have very few games as compared to old teams, so we decided to not filter the data based on frequency of team appearance. However, we still don't care about old teams with few games, so we weight games by the year they were played in with more recent games being heavier. We also analyzed the win/loss/tie score distribution of teams as a function of teams being home or away, and generated a graph visualization showing what teams played against each other.



From this chart, we can see that there is a slight home advantage for winning teams, but that the correlation doesn't hold over for losses or ties. Overall, ties and losses score significantly lower than wins for both home and away, but that's unsurprising. It is interesting to note that ties seem to be more heavily skewed towards mutual shutouts than high-scoring games, but as we'll note later, we considered ties to be losses for our model training.

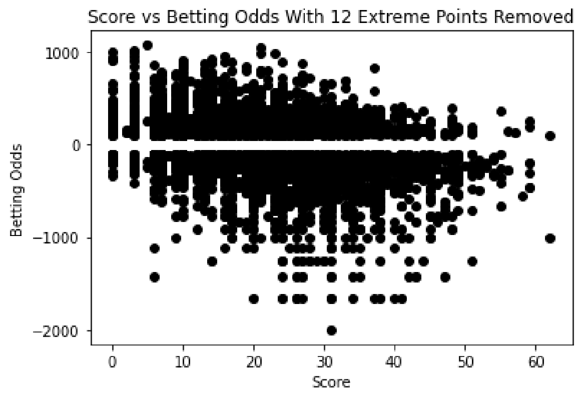




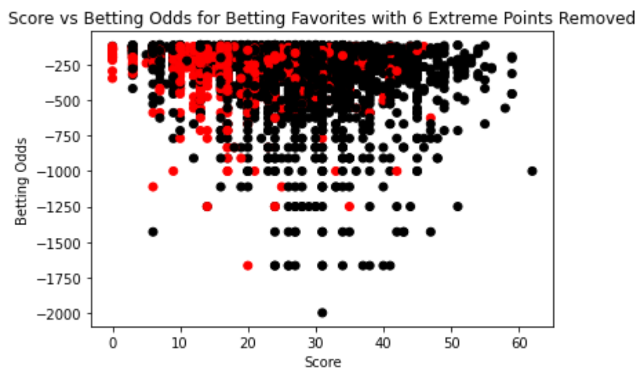
This chart highlights how a small number of teams (those ranked below `cin`) account for the vast majority of games, almost 90%. However, this correlation isn't as useful as a time weighting for predicting future games, so it wasn't used as a data filter. The network shows the matches between teams. If two teams played each other, we draw a line. The dense cluster of teams is almost completely densely connected, where every team played every other team. However, there are some notable outliers, indicating that the model would be "flying blind" if those teams occurred in the test set but not the train set. Some teams even played themselves!

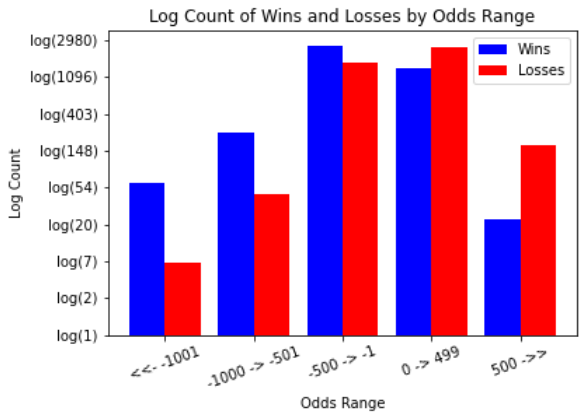
Odds Portal:
This dataset contains betting odds for each game as well as the final results of the game, including score and whether the game went into overtime. The below are a number of graphs that we have used to make interesting observations about the relationship between this data.

The first avenue of exploration we followed was simply plotting the relationship between betting odds and score. The following graph relates score and betting odds for both home teams and away teams. Twelve "extreme points" have been removed. They are outliers, but they were not excluded from the data by a formal outlier removal process. These data points were wins in the 30s with betting odds of up to -10,000 and losses in the 20s with odds of up to 2,000, so they were removed so that the majority of the data could be better visible.



There clearly seems to be a relationship between betting odds and score. It looks like they are negatively correlated, which makes sense with the understanding that negative odds are for betting favorites. In order to dig deeper into this relationship, we made the following plot. The same extreme data points are left out. Ultimate losses are colored in red and wins are in black.





However, from this graph, it looks like the seemingly apparent correlation is more of an optical illusion than anything else. There is no linear relationship visible when we control for betting favorites, which implies there is no correlation. There is a relationship between score and winning, as demonstrated by the concentrations of colors, but it looks like score and odds are not strongly related. Odds and wins are related, however, with 67% of betting favorites ultimately winning their matches. Following that observation, we plotted wins and losses by odds range, above on the right. This plot shows that odds are reflective of confidence in a win or a loss and that a low absolute value in odds indicates a significant lack of confidence, as shown by how high both win and loss counts are here. Higher odds, however, have significant count difference increases between the two.

Data Modeling

We tried training a linear SVM classifier on our betting data using a 70-30 test-train split. We input the home and away teams' identifiers and the betting odds for them to win. Having this linear classifier choose a winner yielded about a 66% accuracy rate, which is better than 50-50 but but dishearteningly close to the number of betting favorites who end up winning. We tried another simple linear model, but this one to predict team scores at the end of the game, implicitly predicting wins. This model ultimately had about a 66% accuracy rate as well. The average difference in model predictions and actual final score was 0, so we know that the model was unbiased in its prediction. However, the variance was very high as the standard deviation of the prediction error was 9 points. We believe the reason such a simple model had such high variance may be because of limitations with the data: if the model's inner workings are making decisions based off of specific match-ups, then there will be very few times team A played team B to learn from.

When leveraging our larger dataset, we didn't have the odds data[1]. Instead, we generated more columns by searching backwards in time for the previous games played, and taking columns from those games. This allows us to easily generate a very wide dataset, although the connection between very distant games and current games is often tenuous. After some testing, we settled on 5 previous games and the current game as a good balance between accuracy and width, without leaning into overfitting too much. This particular algorithm gave us a lot of trouble, due to the doubled-up nature of our dataset. Data leaks inside it could put the current game's score into the previous game's position, artificially inflating our performance on some games. Other synthetic columns included the difference in dates between each of the games, the difference in score between the team and opponent in previous games, and the percent score to team or opponent in the previous games.

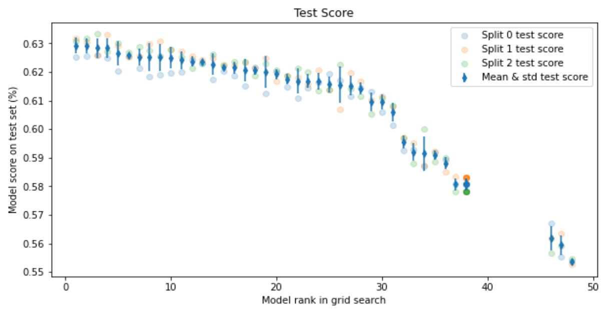
Before training, we reserved the odds data for final evaluation, this being the games in years 2008-2020, about 11% of the data. While this does deflate our final model performance somewhat, because in production it would be updated weekly, it significantly reduces the strain of evaluating our models on the betting data. After this, we created a 60-20-20 train-val-test split for model selection. For k-fold cross validation, we combined the train and val sets for an 80-20 k-fold cross-validation and test split. After evaluating our models on the test split, we trained them again on the full data up to 2008, and evaluated their betting performance across several different strategies vs a baseline model that always picks the preferred team. In every case, our trained models outperformed the baseline model by almost 2x, though none turned an overall profit.

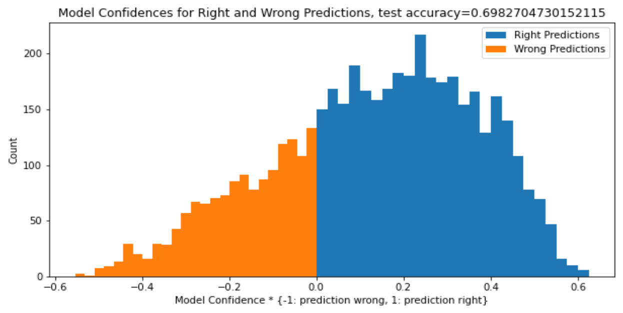
All of our models except CatBoost (more on that later) were trained in a pipeline with a custom scikit-learn ColumnTransformer object, to ensure that we treat data preprocessing in the same way between each cross-validation and across test sets. The ColumnTransformer applied one-hot encoding to categorical data like team codes, and applied mean & std normalization to numerical data.

The first models we trained were SVCs with different kernels, and three different Cs. These models did not converge in the iterations we provided, but even still the poly and rbf kernels overfit. Each of the validation scores for these models hovered between 50 and 65 percent accuracy, which we thought was unsatisfactory. After our brief survey of SVCs, we shifted to decision trees and bagged, boosted, and fancier ensembles of them. A deep decision tree scored about 55% accuracy on the validation set, not especially better than a linear SVC, but over two cross-validated

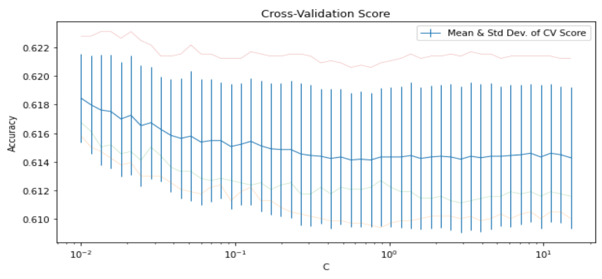
[1] As a reminder, this dataset is organized by "team" and "opponent" rather than by game with home and away, which changes how we talk about games.

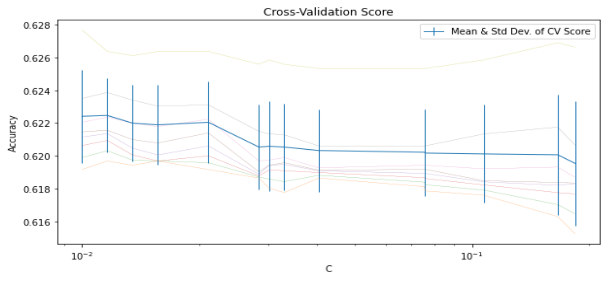
grid searches of bagging classifiers with decision tree classifiers as their base estimator, the validation score rises to over 62%, and over 69% on the test set.



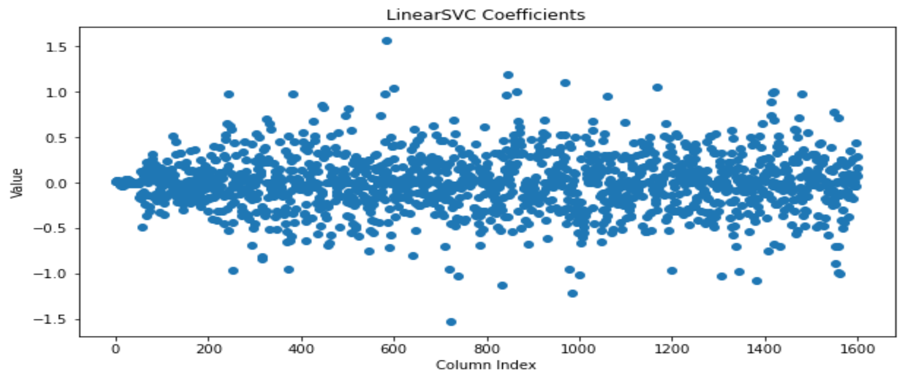


After seeing the success of a grid search, we created a novel search over Cs in linear SVCs, chosen for their decent classification power and lack of overfitting in our first tests. First, we conducted a coarse grid search with 3 cross-validations, then ran a random perturbation of the best-performing Cs with 5 cross-validations.



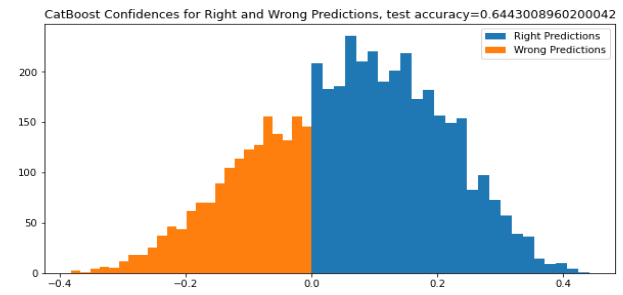


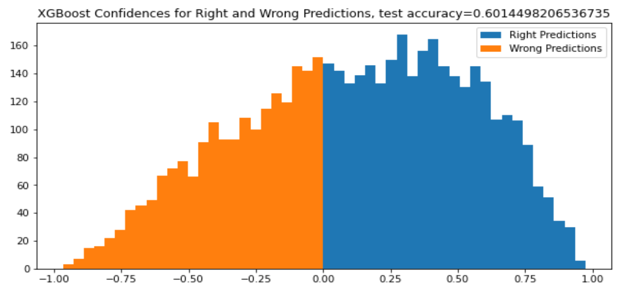
The best performing SVC had C=0.0116, and scored 0.62 on average in the validations. When we examined the coefficients, we noticed a noisy association by column index.



Some notable characteristics include the centering around 0, semi-regular outliers indicating repeated emphasized features (remember that the data is generated from a pattern!), and very low importance of features at the front-end: the model cares a lot about the past games, even all the way out 5 years!

Finally, we explored some state-of-the-art ensembles: XGBoost and CatBoost. XGBoost is a gradient-boosted decision tree ensemble. CatBoost is the same thing, but it makes all input data categorical by choosing the buckets as a part of the learning process! Gradient-boosted ensembles train multiple models in a similar way to regular decision tree ensembles, but they weight hard-to-classify features more through a gradient descent algorithm. This leads to forests of decision trees that cover for each others weaknesses, thereby boosting classification performance. Correspondingly, both XGBoost and CatBoost perform very strongly, with 60% and 64% accuracy on the test set respectively, and very little overfitting.





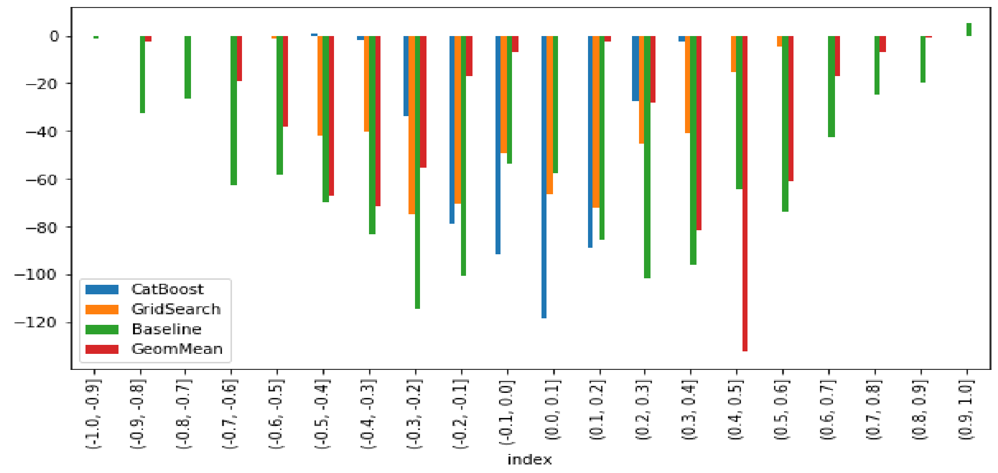
Notably, the CatBoost model (on the left) has a bigger gap between low-confidence predictions that it gets correctly and low-confidence predictions that it gets wrong than the XGBoost model. This translates directly into better performance on the test set.

Of course, the test set isn't sufficient to evaluate our models over our challenge. For this, we need to examine how the models would have won us money over the odds test set. When examining bets, we considered four betting strategies: straight bets, proportional bets, quadratic bets, and risky bets. Straight bets make for the easiest comparison, because the strategy takes the model prediction and submits the same bet to the odds every time. Proportional bets submit the model confidence directly to the odds, while quadratic bets square the confidence to further devalue low-confidence predictions. Risky bets are the most volatile, because they multiply the payout of the selected bet by the model confidence and the payout of the unselected bet by the complement of the model confidence, then submits the bet with a higher expected payout. In this way, the model confidences are taken into account, but we still let the model take risks on games with large payouts.

Before evaluating the models over the odds test set, all models were retrained with their best parameters over the full dataset from 1920 to 2007. While they would be kept up to date in production with some form of online training after each match or season, for speed of computation we trained the models once before evaluating them over 13 seasons. Notably, the 13 seasons' games' features were formed in the same way as the training set, with the actual results forming the rows rather than some kind of time-series prediction-on-prediction pattern, despite our not training the models online.

Conclusions

After evaluating each of the models, and an ensemble formed by taking the geometric mean of the models (which favors more confident predictions), the risky betting strategy far outperformed all other strategies, especially for the CatBoost model. All trained models outperformed the baseline in straight betting, which lost on average 0.36 units per game, by at least twice. Catboost did the best, losing only 0.138 on average in straight betting, and had an exceptionally tiny loss of 0.09 on average in a risky betting strategy. Notably, no models made profit on average, under any betting strategy. Therefore, we can at best recommend this strategy as a handicap to take into account when placing bets, rather than a profit-making machine. Below is a plot of the baseline, CatBoost, grid-searched bagged decision tree, and ensemble models' wins or losses by confidence in the prediction for a straight betting strategy, lower is worse. The most important justification for our fancier betting strategies is observing that the models tend to lose much more money with low confidence bets.



Our model is not yet a weapon of math destruction, as it doesn't yet turn a profit, though if it ever does we would proceed to immediately put bookmakers out of business by using it as a weapon of math destruction. It also is not a fairness concern, because it is not being used to make any ethical decisions, such as referee calls or draft decisions. If we incorporated roster or foul information into the model, then it could be used to impute those kinds of decisions, but we aren't, so we have no ethical concerns. While there are other ethical correlations, such as team funding and win rate, other correlations like team location are probably too obscured by the generic data we use to be useful in ethical decisions.

References

CatBoost:
https://catboost.ai/

XGBoost:
https://xgboost.readthedocs.io/en/stable/
https://machinelearningmastery.com/gentle-introduction-xgboost-applied-machine-learning/

Moneyline Odds:
https://www.oddsshark.com/sports-betting/betting-money-line

Pro Football Archives:
https://www.profootballarchives.com/nfl.html

OddsPortal:
https://www.oddsportal.com/american-football/usa/nfl/results/