Covering the Spread

Michael Sloan

Department of Computer Science and Engineering
Wright State University, Dayton, OH
Email: `sloan.40@wright.edu`

Abstract. Sports betting is one of the largest growing industries in America and continues to grow with further legalization throughout the United States. To capitalize on this growing market, it is important to pick up on trends and correlations amongst sports data. With the use of machine learning models such as the Support Vector Machine, K Nearest Neighbors and the newly introduced Bernoulli Naïve Bayes, research was able to be conducted on whether this specific data set provides enough information to accurately predict if the betting favorites in the National Football League (NFL) were able to cover the spread or not. The goal of the models is to accurately predict outcomes of future games and therefore win future bets just with basic information such as the name of teams playing, spread favorite, stadium information, weather data, etc. Throughout the process of preprocessing, model training, and evaluation it was discovered that the results from this problem were not ideal. In hopes to find out why the results did not meet expectations the feature list was split into three different iterations and compared against each other. This revealed that the home team, away team, home score, and away score were the most critical features, and the rest of the features did not contribute very much at all to the model's prediction.

Keywords: spread, NFL, betting favorite.

1 Introduction

Sports betting is one of the largest growing industries in America and continues to grow with further legalization throughout the United States. In Ohio alone, sports betting brings in a revenue of $779,140,969 per year [1]. One of the staples in sports betting is a metric called the point spread or spread. The spread is a handicap given to the favored team before the game. For example, let's say before a National Football League (NFL) game the home team is favored by 7 points on the sportsbooks. This means if someone places a bet on the home team to cover, the home team will have to beat the away by more than 7 points to cover the spread.

My motivation to look at NFL data along with sports betting data stems from a few major sources. I began playing football in the third grade and continued all throughout high school. It has always been my favorite sport and naturally I became a fan of watching games on TV as well. Sports betting allows for a new level of engagement with fans of the game, and recently there has been an influx of data used by experts to try and predict sports outcomes to get the upper hand on the sportsbooks. This provides a perfect opportunity to try and use some of this data with the help of machine learning to predict outcomes of sporting events and gain the upper hand on the sportsbooks.

The problem that I am trying to solve corresponds with the scenario that I introduced in the first paragraph, the betting favorite covering the spread. I want to be able to use various features from an NFL dataset and predict whether the team that was favorited covered the spread or not. This can be broken down into a simple binary classification problem with a one representing that the betting favorite covered the spread, or zero meaning the betting favorite did not cover the spread.

2 Dataset

2.1 Data Source and Statistics

I downloaded the NFL betting dataset from a user on Kaggle [2]. The dataset had 13,787 entries and 17 features. Amongst those entries there were 27,277 null cells. Each entry in the dataset represented one game of the NFL season. The dataset contained games ranging from 1966 up to early 2024. The features contained relevant game information such as what week in the NFL season it was, what teams were playing, the scores of the teams playing, betting favorite, spread, stadium information, and weather data.

2.2 Data Preprocessing

This dataset required a lot or preprocessing to fulfill the needs of the problem I want to solve. I started by removing the first 5,000 rows because they didn't contain any of the betting information and I didn't want older results to skew the models because the game of football has modernized exponentially since the 1960's and 1970's. Next, I removed the last 200 rows because it contained entries of null values because the games haven't occurred. Later, I added a "covered" column to the dataset that will act as my label to determine if the team favorite covered or not. To fill in the covered column I performed a simple calculation where I first determined if the "team_favorite_id" was corresponding to the home team or the away team and then I calculated the difference using the below equations respectively. If the difference was greater than 0, then it means the favorite covered because the "spread_fav" feature is a negative float. The games who had favorites cover the spread received a one in the "covered" feature and all other entries received a zero.

$$difference = score\_home - score\_away + spread\_fav \tag{1}$$

$$difference = score\_away - score\_home + spread\_fav \tag{2}$$

Additional preprocessing that was performed on the data set was filling in null data with mean values such as: temperature, wind, and over/under lines. Next, I converted the Boolean values to integers, and changed the playoff weeks from categorical to an extension of the week numbers to keep them numerical. The humidity feature that was included in my dataset had around 40% missing values, so I decided to drop that feature along with the date of each game as that has little effect on the outcome.

The last data preprocessing performed was one-hot encoding on the categorical features in my data set. This type of encoding was chosen to prohibit any type or ordinance amongst these features such as team names, stadium names, etc. There should be no ranking given to any team names, or stadium names as that would skew the data and lead to the model making unnecessary and inaccurate assumptions. I chose to use the 80/20 data split for my training and testing data respectively.

3 Methodology

To solve the binary classification problem of predicting if a team covered the spread or not data needed to be trained and tested with common binary classification models before the newly introduced model. The baseline models that I chose to utilize were K Nearest Neighbors, Support Vector Machine, Logistic Regression, and Random Forest. The new model that I introduced was the Bernoulli Naïve Bayes classifier.

3.1 K Nearest Neighbors

K Nearest Neighbors (KNN) works just as the name suggests, it is a classification method which looks at the distances between test data and the training data and uses that to predict the class for the data. In context of my problem, the 20% of my test data was dispersed and the model will look at the K nearest plots and categorize the test data according to majority of the K neighbors. I found the optimal value of K for my dataset to be 25.

3.2 Support Vector Machine

The concept of Support Vector Machines (SVM) is to find the points that are closes to the hyperplane and put the decision boundary in the middle of those two points. This creates a clear separation between classes allowing for the model to predict based on which side of the hyperplane the test data points are on. For my problem, the RBF kernel had the best performance in conjunction with the SVM.

3.3 Logistic Regression

Logistic Regression is a classification model that combines the features as input with the dot product of the weight vector to get the output. It then uses this output to determine if it is a 1 or 0, for example:

$$\textit{Output} >= 0.5;\ \textit{class} = 1 \tag{2}$$

$$\textit{Output} < 0.5;\ \textit{class} = 0 \tag{3}$$

3.4 Random Forest

The Random Forest model is like the decision tree, but with the caveat of bagging. Bagging is a type of ensemble method that looks for a diverse group by varying training data or breaking it up into smaller groups of training data. For my specific data set I got the best results by using 500 estimators and 16 leaf nodes. This model was also able to help me distinguish what my most significant features were in my dataset.

3.5 Bernoulli Naïve Bayes

Bernoulli Naïve Bayes is a subset of the Naïve Bayes Algorithm which is derived from Bayes' Theorem. Bayes' Theorem uses sequential events that calculate an initial probability, but the more data that is gained then the initial probability is impacted. A synonym for initial probability is prior probability and once the data is observed, the posterior probability is determined by building off the prior probability [3].

The difference between the Bernoulli Naïve Bayes Algorithm and the Bayes' Theorem are the assumptions that are made before evaluation. The Bernoulli Naïve Bayes classifier assumes that the features are conditionally independent and have equal importance. The Bernoulli Naïve Bayes predicts the posterior probability with the help of this equation [3]:

$$\text{Posterior probability} = (\text{conditional probability} * \text{prior probability}) / \text{evidence} \tag{4}$$

4 Experimental Results

4.1 Iterations of Experiment

While training and evaluating my models I realized the results were a poorer than expected. To correct this, I changed various parameters on the models and tried further preprocessing some data. Eventually I concluded that the model parameters weren't the problem, and no further preprocessing could be done to my features. This led me to create three different set tests of features: Test 1, Test 2, and Test 3. Test 1 contained the original feature list that I described in the introductory section. Next, with the help of the Random Forest model I determined what the four most important features of the data set was, and I evaluated the models solely on that feature list. For the last test, I used nearly all the original features that were present in Test 1, except I removed the scores for the home and away teams, because in a real-life scenario when predicting whether a team will cover the spread or not, the score will be unavailable. A comprehensive list of all the features for each test is shown below in Table 1.

Table 1. List of each test's features

Test 1	Test 2	Test 3
schedule_season	score_home	schedule_season
schedule_week	team_home	schedule_week
schedule_playoff	score_away	schedule_playoff
team_home	team_away	team_home
score_home		team_favorite_id
score_away		spread_favorite
team_away		team_away
team_favorite_id		over_under_line
spread_favorite		stadium
over_under_line		stadium_neutral
stadium		weather_temperature
stadium_neutral		weather_wind_mph
weather_temperature		weather_deatail
weather_wind_mph		
weather_detail		

See Figure 1. to compare the three different tests model accuracies to each other.

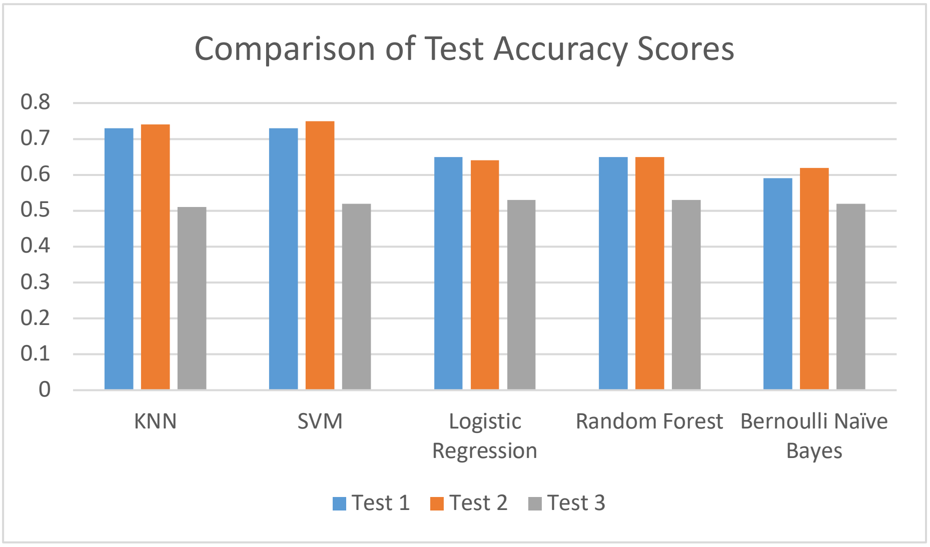

Fig. 1. A visualization of the various accuracy scores for each model on the three different tests

Based on the results of Figure 1., Test 2 has the best performing model, therefore in Table 2 and Table 3, I am going to compare classification reports between Test 2’s best and worst performing model respectively.

Table 2. Classification Report of Test 2 SVM, which is the best performing model of all the different tests.

	Precision	Recall	F1 Score	Support
0	0.73	0.84	0.78	914
1	0.79	0.65	0.71	822
Accuracy			0.75	1736
Macro average	0.76	0.74	0.75	1736
Weighted average	0.75	0.75	0.75	1736

Table 3. Classification Report of Test 2 Bernoulli Naïve Bayes, which is the worst performing model of Test 2.

	Precision	Recall	F1 Score	Support
0	0.61	0.76	0.68	914
1	0.63	0.46	0.53	822
Accuracy			0.62	1736
Macro average	0.62	0.61	0.60	1736
Weighted average	0.62	0.62	0.61	1736

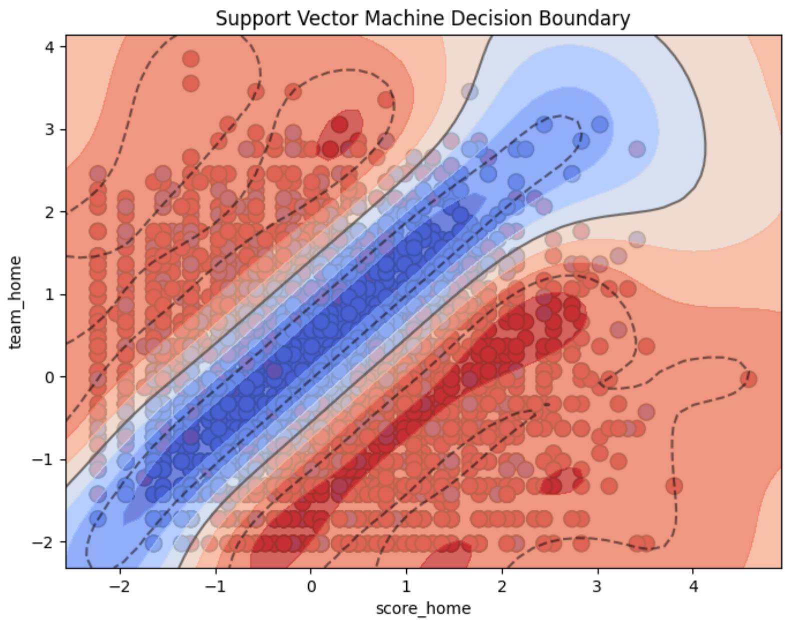

Fig. 2. Visualization of the best performing model from all three tests, Test 2's SVM.

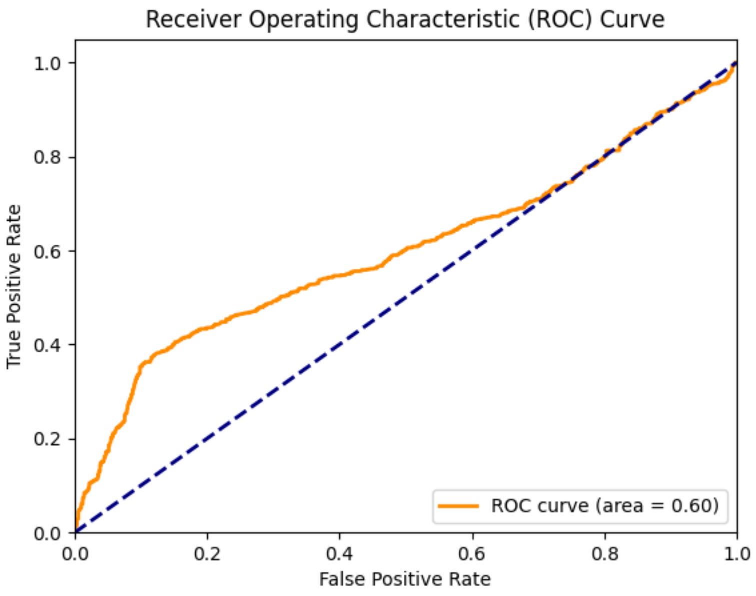

Fig. 3. Receiver Operating Characteristic Curve to visualize the Bernoulli Naïve Bayes from Test 2.

5 Discussion

5.1 Test Comparisons

Between the three tests that were conducted with various sets of features Test 2 was the highest performing across all the models. This should be the case because according to the Random Forest model, the four most significant features were chosen for Test 2's feature list, allow for minimal obscurity from unnecessary features. Test 1 was the next highest performing model with the original list of features. As you can see in Figure 1 above, Test 1's performance wasn't much worse than Test 2's, but there is a slight drop off due to unnecessary features that don't effectively contribute to the overall predictions. Lastly, Test 3 had very poor performance for each model, while it contained nearly all the same features as Test 1 besides the scores for the home and away teams.

5.2 Explanation for Test Performances

Test 3's performance is as expected once I began training the models on the data. Initially, when I sought out this problem, my goal was to use Test 3's features as they are the most realistic application. When a person places a bet on an NFL team to cover the spread, they are not aware of the final score of the game as that diminishes the point of betting in the first place. However, the top two most crucial features in the data set are each of the scores for the respective home and away teams. This would make sense because there is obviously a direct correlation between score of the game and the favorite covering the spread; see equations 1 and 2. This fact further justifies why Test 2 performs so well because it has the least number of features, and the features it does contain are the most crucial by far. Lastly, Test 1 is in the middle of Test's 2 and 3 because it is a combination of both of those scores, therefore the models can recognize the importance of the home and away scores, but the other features obscure the home and away score's importance, enabling the model to look for other correlations.

5.3 Bernoulli Naïve Bayes Performance

Across each of the three tests Naïve Bayes performance remained the worst across all the metrics. Initially, I was confused to why that was because I was under the impression that the Bernoulli Naïve Bayes is an excellent binary classification model. Further investigation into how the Bernoulli Naïve Bayes works showed me exactly why it doesn't perform nearly as well as any of the other binary classification models that I used as a baseline. As stated previously the Bernoulli Naïve Bayes makes two important assumptions before training and testing any of the data: the features are conditionally independent and have equal importance. In the data set used to train these models, these assumptions are false. Specifically, the features for each team's score and the team's name are much more significant than any of the other features in the data set. This led to a very poor result from the Bernoulli Naïve Bayes model because before training commenced, I have already broken the most important rule.

This is further proven in the fact that Test 2 used the 4 most important features that were closely ranked together compared to any of the other test's feature list. The home and away team's score were given a 0.25 importance and 0.26 importance score respectively. These two features have very similar importance and there are only two other features to skew the equal importance scores in the feature list which is likely why the Bernoulli Naïve Bayes scores were the best for Test 2.

6 Challenges and Future Steps

Considering the poor results, specifically the accuracy scores from the various tests and models that were evaluated there were multiple challenges that I faced throughout this project and therefore also some future steps that can be arranged to improve this problem and future machine learning problems.

6.1 Lack of Available Datasets

The most crucial part of a machine learning problem is first finding a good data set and problem to solve. When I began this project, I wanted to look for sports related problems because watching sports is one of my favorite hobbies and I wanted a problem that I had passion to solve. This goal was met, and I got to work on a problem that dealt with the NFL, but I should have invested much more time into searching for NFL related datasets. Initially I believed that the dataset I chose had various important features for determining which team won a game, and how lopsided the margin of victory was. This belief turned out false and part of the blame belongs to the dataset that I chose.

In future projects I will spend much more time examining a chosen dataset before I begin preprocessing and training to ensure that the features are valuable, and that the data is mostly complete. Another problem that I had with my dataset was that an entire column had to be removed due to 42% of its features being null and I had to fill in a small percentage of three other features with mean values.

The evaluation of certain statistics and their contribution to whether an NFL team won a game will be another avenue of future research for this problem. For example, one advanced analytic that could be considered for a future feature added to this dataset would be the efficiency versus expected model (EVE) [4]. The EVE model is complex, but it basically measures a team's efficiency by comparing their success on each play to a league average based on situation and opponent. This could be an important metric used to determine how good a team is which could possibly lead to determining whether a team won the game or not and by how much which contributes to predicting whether a team covered the spread or not.

6.2 Adjustment of Target Variable

Another future step I could take with this dataset is to change the target variable and problem that I want to solve. The scope of my problem lends itself nicely to a regression-based problem as well as a classification problem. To do this, I would need to change the target variable to the scores of the home and away team and try to predict those with the help of the other features. This would probably become a much more difficult problem using the exact same dataset because I would basically be using the Test 3 list of features and the performance of Test 3's models were the worst of the three tests by far. I would most likely have to implement my above improvement in adding more significant features to my dataset before I changed the target variable, and then I could begin to reevaluate the problem and hopefully get better results.

7 Conclusion

The results of my project were less than ideal, but through the poor performance I gained a lot of knowledge about machine learning and how to design a proper machine learning problem. Before I began this project, I would have guessed that the hardest part of this project would be the training and testing of the models, but I was completely wrong. I spent most of the labor hours with this project in choosing what problem I wanted to solve, finding a corresponding dataset, and then preprocessing this dataset to evaluate it.

Training the models for this project didn't require much labor or code at all, just simple application programming interface (API) calls with a few lines of code and then evaluating the models with classification reports. Once I got the results for the initial feature list, I decided to split the features into various test feature lists to see which test performed the best and why they performed that way. This process taught me a lot about my dataset and how important some features were, and how miniscule some were in the prediction.

The final part of this project was comparing each of the test's model results to one another and finding the correlation across tests. For example, across all three tests the SVM performed best, and the Bernoulli Naïve Bayes performed worst. Once I discovered this, I was able to investigate what caused models to perform better than the other models. In the case of the Bernoulli Naïve Bayes, it performed so poorly because it assumes all features are of equal importance and that was not a correct assumption for this dataset at all. With all this information garnered, I am now able to revise this problem for the future and tweak the dataset, chosen models, or even the target variables in hopes to gain better results and more accurately predict if the NFL favorite covered the spread.

References

1. Ramsey, Eric. US Sports Betting Revenue & Handle. LSR. https://www.legalsportsreport.com/sports-betting/revenue/. Last accessed 2023/12/8.
2. Spreadspoke. NFL scores and betting data. Kaggle. https://www.kaggle.com/datasets/tobycrabtree/nfl-scores-and-betting-data/data. Last accessed 2023/12/8.
3. What are Naïve Bayes classifiers? IBM. https://www.ibm.com/topics/naive-bayes#:~:text=The%20Naïve%20Bayes%20classifier%20is,a%20given%20class%20or%20category. Last accessed 2023/12/8.
4. NFL Advanced Stats Zone. The Analyst. https://theanalyst.com/na/2023/06/nfl-advanced-stats-zone/?tab=teams&sortOrder=desc. Last accessed 2023/12/8.