Predicting NFL Outcomes from Previous Performance Statistics

Sai Dandu, Max Friel, Dorian Tisdale
CS613 Machine Learning
Drexel University

***Abstract*—This study explores the use of machine learning techniques to predict the outcomes of NFL games using historical play-by-play data from NFLFastR. A variety of models were employed, including linear regression, logistic regression, bagging, and random forest, to assess their effectiveness for both regression and classification tasks. Ensemble methods, particularly random forest, demonstrated superior performance in capturing predictive patterns within the data, while simpler models like logistic regression showed limited accuracy in classification tasks. This work highlights the potential of machine learning in leveraging large-scale sports data for predictive analytics. Future efforts will focus on refining feature engineering and optimizing model parameters to improve predictive accuracy and practical applicability in NFL game outcome forecasting.**

I. Background and Related Work

The application of machine learning (ML) to predict sports outcomes has gained significant traction in recent years, fueled by advances in data analytics, computing power, and the availability of detailed sports statistics. Machine learning algorithms are capable of identifying complex patterns and relationships within large datasets, making them particularly ripe for forecasting the outcomes of sports games, where results are influenced by a matrix variables such as player performance, team dynamics, weather conditions, and historical trends.

AI-driven sports simulations, such as those used in the Madden NFL video game series, exemplify another application of predictive analytics. Each year, Madden's developers simulate the NFL season, including the Super Bowl, to predict outcomes based on in-game AI models trained on player ratings, team performance metrics, and historical data. These simulations have become highly publicized events, with notable accuracy in certain years.

The integration of ML in sports analytics has also extended into betting markets, where the goal is to "beat the Vegas odds"—the betting lines set by professional oddsmakers. Unlike traditional "gamblers" who may rely on intuition or limited statistical analysis, ML models can process large datasets to identify subtle trends that oddsmakers might miss. By exploiting these human inefficiencies, ML-based betting systems aim to deliver a competitive edge.

In the paper "Sports Results Prediction Model Using Machine Learning" by Alden Obradović and Dino Kečo, they explored the application of machine learning techniques to forecast football (soccer) match outcomes. The authors developed a predictive model use historical data from the English Premier League, covering the 2000 to 2020 seasons. They tested various machine learning algorithms, including logistic regression, decision trees, neural networks, and support vector machines, to classify match results into win, lose, or draw categories. The study found that these models achieved an average prediction accuracy of around 60%, with neural networks performing slightly better than other ML methods. The research highlights the potential of machine learning in sports analytics, particularly in assisting coaches and managers with performance evaluation and strategic planning, in addition to informing sports betting practices.

For our project, we are using linear regression, support vector machine, logistic regression and two ensemble methods (bagging and random forests) to attempt to predict the outcomes of future NFL Football games. We choose this challenge based on availability and the richness in details of the dataset available from the NFLFastR dataset. By doing some feature engineering and creating additional features that add classification fields to the dataset, we will test machine learning methods and their ability to beat the odds.

II. Methodology

A. Training Algorithms

Logistic Regression: Logistic regression was applied to predict binary outcomes (win / loss against spread) using historical NFL data. Logistic regression uses a sigmoid function to estimate the probability of a binary outcome (`TRUE` or `FALSE`). This can be expressed as:

$$P(y=1|\mathbf{x}) = g(\mathbf{x}) = \frac{1}{1+e^{-(\mathbf{x}\mathbf{w}+b)}}$$

Where:

- $P(y=1|x)$: Probability of a positive class given input x.
- $xw+b$: Linear combination of input features (x) and model weights (w), with b as the bias term.

The weights w were optimized using gradient descent, minimizing the binary cross-entropy loss function with L2 regularization.

$$\text{Loss} = -\frac{1}{m}\sum_{i=1}^{m}\left[y_i\log(\hat{y}_i)+(1-y_i)\log(1-\hat{y}_i)\right]+\frac{\lambda}{2}\sum_j w_j^2$$

Where:

- y_i: Actual outcome of the i-th observation.

- $\hat{y}_i$: Predicted probability for the i-th observation.
- m: Total number of observations.
- λ: Regularization parameter.

The features of each game were transformed to represent the self- and opponent statistics. This allowed the logistic regression model to train on twice the number of samples compared to treating games separately. In addition, gradient descent with a learning rate of 0.01. L2 regularization was used to optimize the weights. After training, the thresholds were adjusted by evaluating the accuracy between various probabilities. The accuracy was calculated for the validation set. The regularization parameters were tuned to reduce overfitting.

After running the data, the logistic regression model achieved a validation accuracy of **54.98%**, which is an improvement over random guessing.

Linear Regression: A supervised machine learning technique used to model the linear relationship between one or more independent variables (features) and a dependent variable (target). It is one of the simplest and most interpretable methods for predictive modeling, often serving as a baseline for comparison in machine learning tasks.

For multiple independent variables $x_1, x_2, \ldots, x_n$, it generalizes to:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n + \epsilon \tag{1}$$

Where:

- y: Dependent variable (predicted output)
- $x_1, x_2, \ldots, x_n$: Independent variables (features)
- β_0: Intercept
- $\beta_1, \beta_2, \ldots, \beta_n$: Coefficients (weights) of the independent variables
- ϵ: Error term, representing the deviation of actual values from predictions.

Determine the coefficients (β) by minimizing the residual sum of squares (RSS) or using optimization methods such as Ordinary Least Squares (OLS):

$$\text{RSS} = \sum_{i=1}^{m} (y_i - \hat{y}_i)^2 \tag{2}$$

where y_i are the observed values and $\hat{y}_i$ are the predicted values.

Initially, linear regression was performed twice with one observation per game, once to calculate the home score and then again to calculate the away score. Upon review of the weights of the features, the expectation was that Home Last 4 Games Yards Gained and Away Last 4 Games Yards Gained would have a similar weight to each other respective to Home and Away Score as would all other features. It was discovered that this was not the case. To increase accuracy, linear regression was run once with two observations per game, one observation for the home team and one for the away team. To make this work, features had to be converted to Self and Opponent instead of Home and Away team for each row. For example, when looking at the Home Last 4 Games Yards Gained, for the home team observation that was Self Last 4 Games Yards Gained, and for the away team it was Opponent Last 4 Games Yards Gained. This allowed the training and validation sets to effectively double and provided a lower SMAPE value for both the training and validation sets.

Once a score was determined for both home and away teams, the difference was taken and compared to the spread for the game. The game was then given a binary classification based on if the home team would win against the spread.

As seen in the table accuracy of predictions vs point spread, if you only look at observations where the winning team beat the spread by x amount you can actually get much better results. Particularly about one fifth of the observations predicted a team to win against the spread by more than 6 points and those predictions won against the spread 66.6%.

SVM: A classification algorithm that separates data points of different classes and finds a hyperplane between them. In this case, the kernel trick was used. To determine the best-fitting polynomial degree results, we run for degrees 0-100 and the results are shown below. 47 gave the best fit, so that was used.

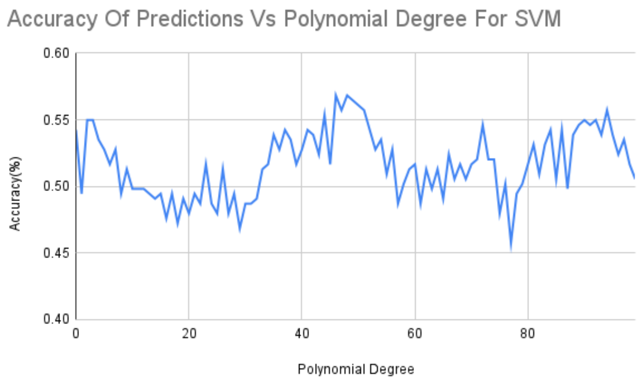

Fig. 1. Performance comparison of polynomial degrees within SVM.

SVM alpha was calculated using the following equations

$$K(a,b) = (ab^T + 1)^{47} \tag{3}$$

$$\alpha = (diag(Y)K(X,X)diag(Y))^{-1} ones(size(\alpha),1) \tag{4}$$

Where:

- Y: Dependent variable From Training Set
- X: Independent variables From Training Set(features)

Prediction was then done using the following equation

$$g(x) = K(x,X)diag(Y)\alpha \tag{5}$$

Where:

- Y: Dependent variable From Training Set
- X: Independent variables From Training Set(features)
- x: Independent variables From Validation Set(features)

Ensemble Bagging: A machine learning technique designed to improve the accuracy and stability of predictive models. It achieves this by combining the predictions of multiple base models (typically weak learners) to form a more robust and generalizable aggregated model.

We implemented a bagging-based ensemble learning method for both classification and regression tasks. By using bagging, we were testing the ability to improve the stability and accuracy combining predictions from previously used models trained on different random subsets of the data. The dataset is preprocessed by converting categorical values ("TRUE" and "FALSE") into binary numeric representations (1 and 0). The data is then split with a 70-30 train-test split for model evaluation. For classification, the target variable is treated as binary, while for regression, it is treated as continuous.

For classification, the algorithm trains an ensemble of decision stumps, each built using a bootstrap sample of the training data. Each decision stump predicts the majority class in its bootstrap sample. Predictions are then aggregated using majority voting to generate the final classification output. The accuracy of the ensemble is evaluated on the test set.

For regression, an ensemble of simple models is trained, where each model uses the mean target value of its bootstrap sample as its prediction. Predictions from all models are aggregated using averaging to generate the final regression output. The performance of the ensemble is assessed using the Mean Squared Error (MSE) on the test set.

Ensemble Random Forests: An ensemble learning method that extends the concept of bagging to decision trees while introducing randomization in feature selection. It is a robust technique used for both classification and regression tasks, leveraging the collective performance of multiple decision trees to improve prediction accuracy and reduce overfitting.

Leveraging the binary outputs from each of the previous ML methods, we used an ensemble learning method inspired by Random Forests for both classification and regression tasks. The process begins with preprocessing the input dataset by converting categorical string values ("TRUE" and "FALSE") into binary numeric values (1 and 0). The dataset is then split into features and target values for both classification and regression. An 70-30 train-test split is performed to create training and testing datasets.

For classification, the algorithm builds an ensemble of decision trees using bootstrap sampling of the training data. At each iteration, it randomly selects a subset of features for each tree and predicts the majority class label in the sampled training data as the output for the corresponding test set. The final classification output is determined by aggregating predictions across all trees. The classification accuracy is calculated as the proportion of correct predictions on the test set.

For regression, the algorithm similarly constructs trees, but this time predicts the mean of the sampled training data as the output for each tree. The final regression output is obtained by averaging predictions across all trees. The regression performance is determined by using the mean squared error (MSE) between the predicted and actual test labels.

B. Implementation Details

We used a combination of R and SQL to create the CSV dataset. Once a CSV dataset was created, we used Python to analyze the CSV dataset

1) Pre-processing NFLFastR data using R into a local MySQL database
2) Use R to calculate the previous player statistics as outlined in the Data subsection of Evaluation and write to a csv file
3) Read CSV file into Python
4) Randomize and choose validation/training data sets
5) Complete Logistical Regression, Linear Regression and Support Vector Machine Analysis
6) Appending (3) binary columns to the end of the dataset to hold the boolean outcomes from the Complete Logistical Regression, Linear Regression and Support Vector Machine Analysis ML methods

III. Evaluation

A. Data

Data was gathered from the R library nflfastr. In order to eliminate possible variations due to trades, injuries and other personnel changes, we decided to use an average of player's stats over a period of time. The periods we chose were the last 4 games and last year. This should allow for a team both being a long term good team as well as a team that has had recent success and therefore momentum going into the game. The play by play data was only available through the 2023 season so we selected the 2020 through 2023 season. Once the player statistics were calculated they were added to team statistics for that game as described in Team Statistics table. One feature was added for per stastic for both the home and away team. The following process was used for every game.

- Collect all player ids that had a team depth of 1 or 2 for both the home and away team for that game
- Collect all plays those players were involved in over the past 4 games/year
- Average each statistic that is described in Player Statistics Table Below
- Append a feature for all 4 of the following categories for each statistic, Past 4 Games Home Team, Past 4 Games Away Team, Past Year Home Games, Past Year Away Games

B. Results

IV. Conclusions

This study demonstrates the potential of machine learning (ML) models to predict NFL game outcomes using historical play-by-play data. By employing a range of techniques, including linear regression, support vector machines, and ensemble methods such as bagging and random forests, the analysis demonstrated varying degrees of predictive accuracy. Ensemble methods, particularly random forests, consistently outperformed simpler models in identifying promising patterns in the data.

TABLE I
PLAYER STATISTICS

Statistic	Meaning
Yards Gained	The number of yards the play finished compared to the previous line of scrimmage
Touchdown	Binary value representing if a touchdown was scored on the play
Sack	Binary value representing if the quarterback was tackled behind the line of scrimmage
Penalty Yards	If a penalty was called the number of yards the penalty was for
Fumble Lost	Binary value representing if the offensive team fumbled the ball and the defense recovered
Interception	Binary value representing if there was an interception thrown on the play

TABLE II
TEAM STATISTICS

Statistic	Meaning
Rest	The number of days since the team last played
Past 4 Time Of Possession	A sum of the time of possession for every drive the team has had over the last 4 games
Past Season Time Of Possession	A sum of the time of possession for every drive the team has had over the last season

The results highlight the value of feature engineering and algorithm selection in sports analytics. While the inclusion of recent performance statistics provided a strong foundation for prediction, the study also revealed limitations in certain models, such as linear regression's sensitivity to feature weighting. Despite these challenges, the models showed promise, achieving competitive prediction rates that could inform strategic decision-making for teams, analysts, and more importantly, the betting markets.

Future work should explore advanced feature weighting techniques, such as opponent strengths, and refine ensemble methods to leverage probability-based predictions. Additionally, integrating cross-validation strategies could enhance the robustness of the models, addressing the evolving rules and strategies of the NFL. By building on these results, further research can advance the application of machine learning in sports, providing deeper insights and more accurate forecasts.

V. FUTURE WORK

- Performance data over the past 4 games/season does not take into account a weighting for the opponents. Adding a weight may give more accurate results.
- Ensemble methods were performed based on binary results. Using the probabilities rather than a simple win/loss may improve accuracy.
- Due to the ever changing landscape of the NFL strategy and rules there is only so much data that is relevant to current outcomes. Given that, using s-fold cross validation may be useful to improve accuracy.
- Research further into why the weights for linear regression were different when calculating the home and away score separately

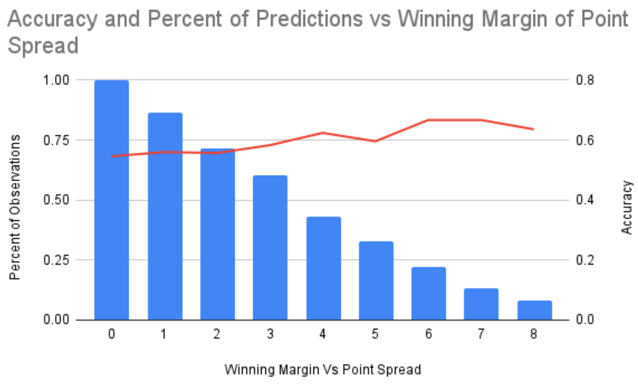

Fig. 2. Performance comparison of our algorithm with baselines.

REFERENCES

[1] Sebastian Carl and Ben Baldwin. *nflfastR: Functions to Efficiently Access NFL Play by Play Data*. R package version 5.0.0.9000. 2024. URL: https://www.nflfastr.com.

[2] Raymond Goode. *Predicting the Super Bowl 57 winner with 100 Madden NFL 23 simulations*. 2023. URL: https://www.si.com/videogames/features/madden-nfl-23-super-bowl-57-simulations-prediction.

[3] Serafeim Moustakidis et al. "Predicting Football Team Performance with Explainable AI: Leveraging SHAP to Identify Key Team-Level Performance Metrics". In: *Future internet* (2023). URL: https://doi.org/10.3390/fi15050174.

[4] Alden Obradović and Dino Kečo. "Sports Results Prediction Model Using Machine Learning". In: *SAR Journal* 7.3 (Sept. 2024). Creative Commons Attribution-NonCommercial-NoDerivs 4.0 License, pp. 184–189. ISSN: 2619-9955. DOI: 10.18421/SAR73-03. URL: https://doi.org/10.18421/SAR73-03.

[5] Legal Sports Report. *What Is the Vig (Vigorish)?* Accessed: 2024-12-11. 2024. URL: https://www.legalsportsreport.com/how-to-bet/vigorish/#:~:text=To%20fully%20cover%20the%20vig,enough%20to%20beat%20the%20vig.

([5]) ([1]) ([3]) ([2]) ([4])