Finding an Edge in Sports Betting

by

(Team E)

Joseph Schoenbaum;

Gabriel Kuykendall;

FINAL PROJECT REPORT

for

DATA 606 Capstone in Data Science

University of Maryland Baltimore County

2025

Copyright ©2025

ALL RIGHTS RESERVED

**ABSTRACT**

In recent decades, sports betting has skyrocketed in popularity and revenue. In this project, we try to determine how random sports really are by predicting the margin of victory for NFL matches played from 2009 to 2025. The goal of this prediction is not to guess the exact score of the game, but to have a spread bet winning percentage around or above 52.38%. To calculate our predictions, we used a Random Forest Regressor and a XGBoost model (with different variations). Our results show that the XGBoost model is able to beat the spread at a rate of 51.28%. In addition, when looking at model predictions that had a difference of one or more from the spread (true bets), our bet win percentage jumped to 55.71%. Lastly, by reducing the dataset to the most recent five years of data, the model’s bet win percentage is a bit higher for total bets at 53.85% and slightly lower for true bets at 54.29%. This shows, given the right data and model, it is possible to gain a slight edge over a spread (depending on the sportsbook) specific to NFL matches. While 55.71% might seem like an insignificant percentage over random guessing, in the world of sports, a couple percentage points can be an impactful increase in profit.

LIST OF DEFINITIONS

*TB* True bet: any bet that has a differential of 1 point or more between the spread line and the prediction of our model

*QB* Quarterback: A quarterback in American football is the offensive position responsible for leading the team's plays, initiating action, and handling the ball

*Spread* Point spread: In sports betting, a point spread is a handicap created by oddsmakers to level the playing field between two teams or competitors with different skill levels

*Total* Total line (over/under): refers to a wager on whether the combined final score (or another specific statistic) of both teams in a game will be higher or lower than a predetermined number set by the sportsbook

*Sportsbook* Sportsbook: a place or platform where individuals can place bets on the outcome of various sporting events

*Juice* Juice (vigorish): refers to the commission or fee charged by a sportsbook for taking a bet

*AP* All-Pro: signifies that a player has been recognized as one of the best in the league at their position for a particular season

*EPA* Expected Points Added: a statistical metric that measures the impact of each play on a team's scoring potential

*SOS* Strength of Schedule: a value representing the difficulty of a team's season schedule

*QBR* Quarterback Rating: a metric developed by ESPN to evaluate quarterback performance, considering various factors beyond traditional passing statistics

LOS Line of Scrimmage: an imaginary line stretching across the width of the field at the spot where the ball is placed before each play

*Sack* Sack: when a defensive player tackles the quarterback behind the line of scrimmage

*Turnover* Turnover: when a team loses possession of the football to the opposing team, disrupting the offensive team's possession and potentially leading to a change in game momentum

*Coverage Rate* Coverage Rate: refers to a metric used to evaluate a defensive player's effectiveness in pass coverage.

*3rd Down* Third Down: the third of a team's set of four downs (attempts) to advance the football 10 yards or score a touchdown

MAE Mean Absolute Error: a statistical metric that measures the average magnitude of errors in a set of predictions, without considering their direction

ACKNOWLEDGMENTS

Team E would like to thank our professor, Unal Sakoglu, and our classmates at UMBC for the support and feedback.

TEAM MEMBERS’ CONTRIBUTIONS

|  |  |
| --- | --- |
| Name | Duties |
| Joseph Schoenbaum | Dataset cleaning, Feature Building, EDA, Random Forest Regressor, Evaluation |
| Gabriel Kuykendall | Dataset cleaning, Feature Building, EDA, XGBoost model, Evaluation |

TABLE OF CONTENTS

TITLE i

COPYRIGHT ii

ABSTRACT iii

LIST OF DEFINITIONS iv - v

ACKNOWLEDGMENTS (& TEAM MEMBERS’ CONTRIB.) vi

I. INTRODUCTION 1 - 4

II. METHODS 5 - 6

III. RESULTS 7 - 8

IV. DISCUSSIONS AND CONCLUSIONS 9

V. FUTURE WORK 9

VI. REFERENCES 10

VII. APPENDIX 11 - ??

1. INTRODUCTION

The goal of our project is to create a model that has a spread bet winning percentage above 52.38%. To break-even in sports betting, at standard odds of -110 (meaning you risk $11 to make $10), you need to win around 52.38% or more of your bets. This is because sportsbooks charge a fee on every bet you place that is around 4.2% (this can vary depending on the bet and sportsbook). The problem is sports are very noisy and hard to predict. Sportsbooks place lines so they always make a profit. With both those things in mind, it is difficult to produce a model that will be profitable (above 52.38%). To tackle this goal, we specifically look at NFL matches from 2009 to 2025. Our main dataset is provided from a GitHub repository called [nfl\_data\_py](https://github.com/nflverse/nfl_data_py). This GitHub repository “is a Python library for interacting with NFL data sourced from nflfastR, nfldata, dynastyprocess, and Draft Scout. Includes import functions for play-by-play data, weekly data, seasonal data, rosters, win totals, scoring lines, officials, draft picks, draft pick values, schedules, team descriptive info, combine results and id mappings across various sites.” We also have additional feature data from <https://www.pro-football-reference.com/> that were not included in the original dataset. The original dataset from nfl\_data\_py had 4345 matches and 46 features.

After adding other features and cleaning the data our dataset had 3184 matches and 90 features. Each one of these features is something you would know before the game starts (hence why it is a predictive model). Below is a list of all the features we used for our model and the reasoning behind each feature.

Target variable: game\_margin – This is what our model is trying to predict. In reality, our goal is only to beat the spread though.

Features:

* spread\_line – The margin that one team is expected to win or lose by
  + Allows for a baseline guess for the model to use as a starting point
* away\_rest, home\_rest – The number of days each team has to rest before the event
  + A rest advantage could be influential in the outcome
* total\_line – The total amount of points scored in the gaem
  + Allows for a baseline on the difference of score
* div\_game – Marks if the game is a divisional match or not
  + Divisional games are typically close since opponents know each other better
* Roof, surface, temp, wind
  + External factors that might influence game margin

Note – The features above were from the original dataset. All other features were created through adding an additional dataset or data manipulation. All additional datasets still have the same years/matches to pull from. For example, some of the additional features below are from the following datasets: A play-by-play dataset (has information about every play that occurred in a game), an AP data dataset (shows which players were the best in the NFL in a given year), an injury dataset (shows if a player is hurt and not active for the match), and quarterback rating (QBR) dataset (shows how well a quarterback has been performing). Another quick thing to mention also is any player on the Injured Reserved (IR) List is not counted within the injury dataset. The IR is a list that a team can place their players on if they have a serious injury that will force them to miss more than 4 matches. This allows the team to add other players to their roster. That means if a player tears their ACL prior to the 2023 season and are placed on IR they will not show up in the injury dataset. This causes issues in our dataset because there was no reliable way to capture players placed on IR. One final thing to note is due to rolling features (to capture recent team performance), we start our data after week 3 of any given NFL season. This allows for no NAN values when giving our model the dataset, which we hoped would give cleaner results.

* is\_playoff – Marks if the game is a playoff matchup
  + Playoff games tend to be close contests
* is\_final\_week – Marks if the match is in the final week of the regular season
  + Many teams sit their starters in the last week of the regular season so the data becomes even noisier
    - Ex: In week 18 of 2024 the Kansas City Chiefs sat all their starters against the Denver Broncos because they had nothing to play for after already locking up the 1st seed for the playoffs. They lost 38-0. This score is not a data point that is not only irrelevant to us but actively hurts our model for most cases
* home\_qb\_switch, away\_qb\_switch – Shows if there was a quarterback change from the previous match to the current one
  + Allows us to see if the quarterback has been injured or benched from one game to the next
  + Quarterback is the most important position in football so it will have a huge impact on the match
* home\_rolling\_avg\_epa, away\_rolling\_avg\_epa – It shows the impact of each play on a team’s scoring potential. This is a team’s average EPA over the previous 3 matches
* home\_rolling\_avg\_yards, away\_rolling\_avg\_yards – The average amount of yards a team has over the last 3 matches
* home\_rolling\_play\_count, away\_rolling\_play\_count – The average amount of plays a team has over the last 3 matches
* home\_rolling\_allowed\_avg\_epa, away\_rolling\_allowed\_avg\_epa – The average amount of EPA a team’s defense has allowed over the last 3 weeks
* home\_rolling\_allowed\_avg\_yards, away\_rolling\_allowed\_avg\_yards – The average amount of yards a team’s defense has allowed in the last 3 weeks
* home\_rolling\_allowed\_play\_count, away\_rolling\_allowed\_play\_count – The average amount of plays a team’s defense has allowed in the last 3 weeks
* epa\_home\_off\_away\_def\_rolling\_diff, epa\_home\_def\_away\_off\_rolling\_diff, avg\_yards\_home\_off\_away\_def\_rolling\_diff, avg\_yards\_home\_def\_away\_off\_rolling\_diff, play\_count\_home\_off\_away\_def\_rolling\_diff, play\_count\_home\_def\_away\_off\_rolling\_diff – The differential between the offenses and defenses of both the home team and away team in relation to EPA, yards gained, and play count
  + All of these features are attempts to capture recent team performance leading up to a match
* home\_recent\_sos\_opponent\_avg, away\_recent\_sos\_opponent\_avg – average record (Wins vs. Losses) a team’s opponents have had over the past 3 matches
* home\_season\_sos\_opponent\_avg, away\_season\_sos\_opponent\_avg – The total record (Wins vs. Losses) a team’s opponents have had over the course of the season prior to the match
* sos\_diff, season\_sos\_diff - The differential between the SOS of the home and away team
  + These capture team performance in relation to who they played
  + It’s important to note if their stats only look good because they have played bad opponents or if they are having bad stats only because they have played really elite opponents
* home\_allpro\_last\_3\_years\_weighted, away\_allpro\_last\_3\_years\_weighted, diff\_allpro\_last\_3\_years\_weighted – AP is an honor that a player is awarded for being the best at his position. It is reserved for the elite tier of players in the NFL. Our AP count is captured from the previous season. So an AP player for 2024 was named an AP player at the end of his 2023 season. This captures the amount of AP a team has for the current season (weighted based on how recent they were listed as an AP player).
* home\_allpro\_prev\_year, away\_allpro\_prev\_year, diff\_allpro\_prev\_year – This is a count of the AP players on the team
* home\_offense\_allpro\_3\_years, away\_offense\_allpro\_3\_years, home\_defense\_allpro\_3\_years, away\_defense\_allpro\_3\_years, allpro\_diff\_home\_off\_away\_def\_3\_years, allpro\_diff\_home\_def\_away\_off\_3\_years, home\_offense\_allpro\_prev\_year, away\_offense\_allpro\_prev\_year, home\_defense\_allpro\_prev\_year away\_defense\_allpro\_prev\_year, allpro\_diff\_home\_off\_away\_def\_prev\_year, allpro\_diff\_home\_def\_away\_off\_prev\_year – These are all more variations on capturing All Pro player data
  + Elite players have a huge impact on the game. By using AP data, we are attempting to capture the number of elite players a team has
* diff\_active\_allpro\_weighted,diff\_active\_allpro\_prev\_year – Shows how many All Pro players are active for the current match (not injured)
  + Obviously, we do not want to capture injured AP players within the count for our given match
* league\_rolling\_avg\_abs\_margin\_by\_week – The average game margin in a given week of season across the league
  + Throughout the years scoring can fluctuate up and down
* home\_qbr\_prev\_year, away\_qbr\_prev\_year, diff\_qbr\_prev\_year – This captures recent quarterback performance
* is\_home\_qb\_new is\_away\_qb\_new – Shows if the quarterback is experienced or not
  + Quarterback is the most important position in football so it is important to capture how well they are playing and how much experience they have
* home\_injured\_count, away\_injured\_count, diff\_injured\_count – shows how many players are injured in the match
  + Injuries are important to understand if a team is playing at an optimal condition
* home\_rolling\_win\_pct, away\_rolling\_win\_pct – Shows each team’s winning percentage in recent weeks
  + If a team has won recently, they may be playing better
  + If a team is on a losing streak, they may be playing worse
* sack\_diff, sack\_diff\_reverse, turnover\_diff, turnover\_diff\_reverse, third\_down\_diff, third\_down\_diff\_reverse, cover\_rate\_diff
* scoring\_diff, scoring\_diff\_reverse – The difference between the total points a team scores and the total points they allow
  + These features are all stats that are typically indicative of winning football games
* home\_coach\_win\_pct\_prior, away\_coach\_win\_pct\_prior – The percentage of games a head coach has won prior to the match
  + A head coach is a very important part of a team

We are not the first to try and find an edge when it comes to sports betting. While some may use their sports knowledge to make bets, others have used machine learning models to improve their earnings. A project called ‘Using Python to Mimic NFL Sportsbooks Odds Determination’ was able to create a basic model using the same dataset we used, nfl\_data\_py. They were trying to better understand how the sportsbooks set their odds and try and get an advantage. Another project submitted to Kaggle called, ‘NFL Betting Model’ by Ty Walters, he took a similar approach to our model and tried to use past data to determine future outcomes and how to bet accordingly. He used an XGBoost model to determine if the favored team would cover the spread and make bets accordingly. His results yielded higher percentages of successful bets with a more focused scope, but would have had fewer games eligible to bet on. We were hoping to capture both works in gaining an edge with a high return on investment potential because we can evaluate every game by identifying the game margin.

1. METHODS

**Random Forest Regressor:**

We decided to use a random forest regressor because it handles nonlinear data well, works well with a broad range of features, is less prone to overfitting, and allows for feature importance. During the modeling part of our project, specific to the random forest regressor, there were various iterations of the model. Throughout the process we removed and created various features to attempt to properly fit our model. In addition, we used cross-validation and grid-search to help make our model as optimal as we can, given the features we have.

**1st Iteration** (all 90 features included, no hyperparameter tuning):

Overall bet winning percentage - 42.96%, TB - 42.75%, Train MAE - 3.97, Train R² - 87.79%, Test MAE - 10.69, Test R² - 9.54%, Takeaway - clear overfitting.

**2nd Iteration** (removing redundant features, adding max depth):

Overall bet winning percentage - 42.54%, TB - 41.59%, Train MAE - 6.70, Train R² - 67.58%, Test MAE - 10.82, Test R² - 7.38, Takeaway - decreased overfitting, slight loss of test performance.

**3rd Iteration** (removing features with less than 1% importance from 2nd iteration, using more hyperparameter tuning):

Overall bet winning percentage - 44.84%, TB - 43.38%, Train MAE - 10.27, Train R² - 15.87%, Test MAE - 10.65, Test R² - 9.22%, Takeaway - marginal overfitting (potential underfitting), Performance not as good as 1st iteration, but similar (or slightly better) than the model from iteration 2.

**4th Iteration** (1st iteration model with hyperparameter tuning):

Overall bet winning percentage - 47.51%, TB - 47.12%, Train MAE - 7.50, Train R² - 51.87%, Test MAE - 10.29, Test R² - 13.45%, Takeaway - Hyperparameter tuning was critical to reduce overfitting. Keeping all features gives best results, unsurprisingly. Gain is still marginal, but every percent difference matters.

**5th Iteration** (4th iteration model with spread line included):

Overall bet winning percentage - 49.88% (best value), TB - 47.74%, Train MAE - 7.32, Train R² - 53.85%, Test MAE - 10.09, Test R² - 16.70%, Takeaway - Adding the spread line as a feature improves are model slightly.

**XGBoost Model:**

We decided to use an XGBoost model to compare with our random forest regressor. XGBoost also handles nonlinear data well, which is perfect for NFL and sports data. It has built-in parameters to prevent overfitting, handle missing values, and provide weight to desired outcomes. During the modeling part of our project, we were able to create several iterations of our XGBoost model. Similarly to the random forest model, we removed and created various features to attempt to properly fit our model. In addition, we used cross-validation and grid-search to help make our model as optimal as we can, and finally attempted to weight successful betting attempts compared to failed attempts during testing.

**1st Iteration** (all 90 features included, no hyperparameter tuning):

MAE - 10.71, R² - 14%, Takeaway - baseline before hyperparameter tuning.

**2nd Iteration** (Used GridsearchCV to tune hyperparameters):

Overall bet winning percentage - 47.72%, TB - 47.07%, MAE - 10.43, R² - 18%, Takeaway - Improved the MAE and gave a baseline for the bet and TB percentage.

**3rd Iteration** (Used the top 25 features from Iteration 2 to create a reduced feature model with the same hyperparameters):

Overall bet winning percentage - 48.19%, TB - 50.55%, MAE - 10.37, R² - 18%, Takeaway - MAE and betting percentage improved, top features are very strong

**4th Iteration** (Created a weight for game margins that beat the spread, 2 to 1 for a win, used same hyperparameters as before):

Overall bet winning percentage - 47.25%, TB - 46.34%, MAE - 10.46, R², Takeaway - The weighted approach yielded similar outcomes to the previous iterations, tuning is needed.

**5th Iteration** (4th iteration model with new hyperparameters from GridsearchCV):

Overall bet winning percentage - 52.95% , TB -54.11%, MAE - 10.38, R² - 18.%, Takeaway - The weighted approach got us over the 52.38% threshold to do better than breaking even on bets

**6th Iteration** (4th iteration model with new hyperparameters from GridsearchCV using the top features from iteration 2):

Overall bet winning percentage - 54.39% , TB -53.03%, MAE - 10.37, R² - 17.%, Takeaway - Focusing on the top performing features raised the overall winning percentage but lowered the TB %, additional features or feature cuts could help improve this model, it has lost percentage on R², so it could benefit from further testing

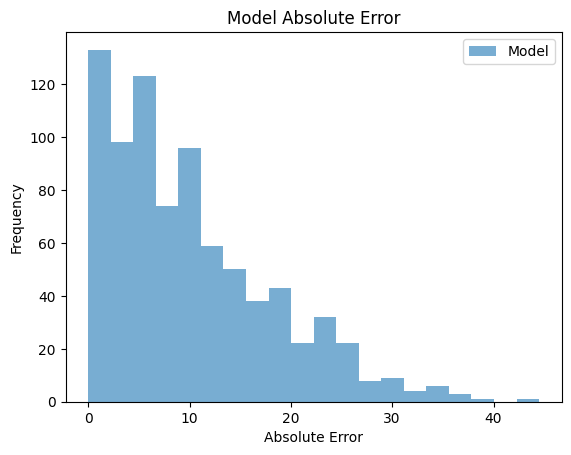
1. RESULTS

**Random Forest Regressor:**

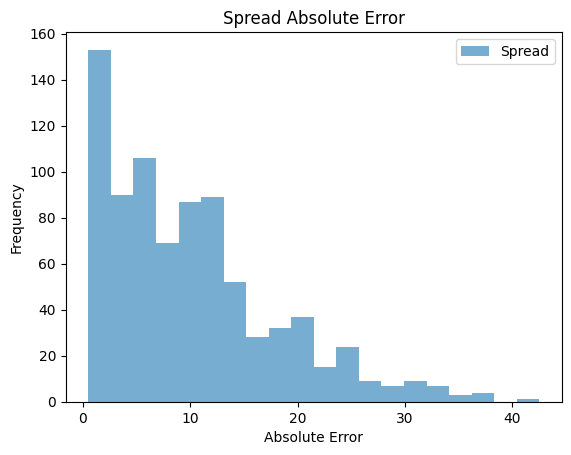
Feature importance % is relative.

* Model hyperparameters – criterion: 'absolute\_error, max\_depth: 15, max\_features: 20, min\_samples\_leaf: 5, min\_samples\_split: 10, n\_estimators: 500
* Cross-validation used
* 70/30 train/test split
* Best bet winning percentage – 49.88%
* Best true bet winning percentage – 47.74%
* Best MAE – 10.09
* Best R² - 16.70%
* Influential Features – spread\_line (~14%), home\_rolling\_win\_pct (~4%), home\_rolling\_avg\_epa (~4%), home\_coach\_win\_pct\_prior (~3%), away\_rolling\_avg\_epa (~3%), away\_coach\_win\_pct\_prior (~3%), away\_rolling\_avg\_yards (~2%), third\_down\_diff (~2%), diff\_qbr\_prev\_year (~2%)

Model absolute error to game margin value:



Spread absolute error to game margin value:

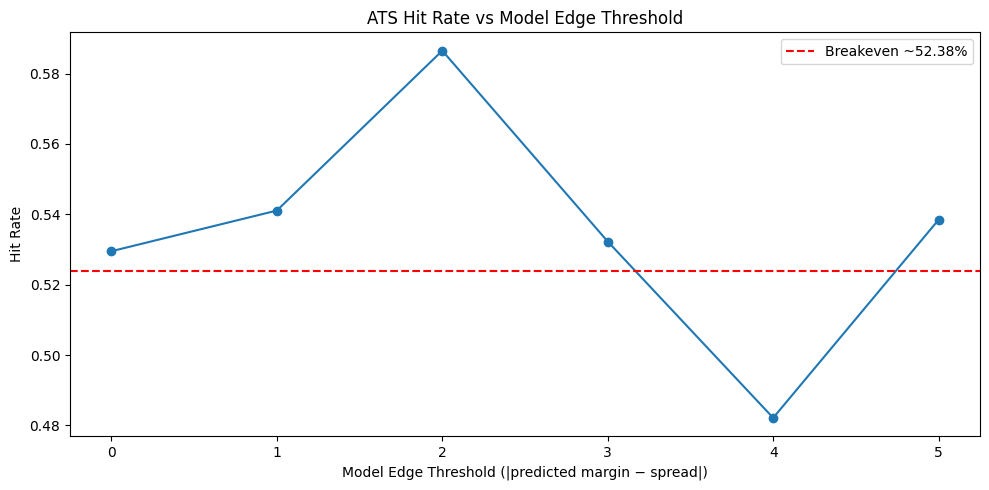


Analysis of charts: Very similar error between spread and model.

**XGBoost:**

This model was the most stable, as well as providing consistent betting wins over 52.38%. The reduced model performed better on true bets but overall this model is more impactful.

* Best Model hyperparameters – n\_estimators: 384, max\_depth: 2, min\_child\_weight: 3, learning\_rate: 0.01, subsample: 0.5, colsample\_bytree: 1, reg\_alpha: 5.5, reg\_lambda: 3.5
* Cross-validation used
* 80/20 train/test split
* Best bet winning percentage – 52.95%
* Best true bet winning percentage – 54.11%
* Best MAE – 10.38
* Best R² - 18.0%



This chart provides the winning bet percentage as the game margin got further from the spread.

1. DISCUSSIONS AND CONCLUSIONS

Sports data is extremely noisy and can be random at times. Many factors influence every second of every game. We tried to capture that for our models by creating creative and obvious features to capture the many nuances of NFL football. We tried to evaluate the talent level of teams, recent success and struggles, as well as environmental factors in the games in our features, with some success. We realized that being able to create robust features was more important than the amount of data for our project, so we went from looking at data from 2000 to 2024 to 2009 to 2024. This allowed us to use more data from our dataset, like injuries that had not been kept track of week by week in our dataset before 2009. We also used the spread line to train our model because we have access to that data prior to the games being played. We also did a test for each model only using the last 5 years, this showed improvement in the XGBoost model, but the random forest model got worse. This is a reminder that sports are always evolving, and sports science, rule changes, and other innovations will keep the sport evolving. We need to be conscious of that when evaluating the sport over long periods.

Our best model was able to place 414 true bets at 54.11%, which will create a positive return on investment for the bettor, and therefore, we had some success in our project. There is more room for improvement in each model and potential to create more interesting features, even after having 90 features available to use for our model.

1. FUTURE WORK

In the future the most important thing to address is having more features. More specifically, adding features that are extremely influential to the game margin. For example, adding something such as spread line has such a significant impact on our model. While small percent improvements may not seem like a major improvement for some tasks, these small adjustments are very impactful for sports betting.

Another potential future project would be to look at more niche situations within the sports betting and football event space. For example, what if we only look at events specific to team’s located on the west coast playing a game on the east coast with a coach winning percentage above a certain threshold. In short, it would be interesting to look at specific match variables and see if we find any trends or patterns that lead to higher-than-average bet winning percentage. Of course, sportsbooks are keen on any trends like that as well, so it is hard to stay one step ahead.

Lastly, a potential project could be to implement a front-end UI to host the ability to input a bunch of match variables and spit out a prediction value to help show if you should bet on the match at hand or not. In order to do this, we would also need to update the models to take the current season into account as well.

1. REFERENCES

NFL-Data-Py. (n.d.). *NFL\_Data\_py* [Python package]. PyPI. Retrieved August 11, 2025, from<https://pypi.org/project/nfl-data-py/>

Pro Football Reference. (2024). *2024 All-Pro Team*. Sports Reference LLC. Retrieved August 11, 2025, from<https://www.pro-football-reference.com/years/2024/allpro.htm>

Crabtree, T. (n.d.). *NFL scores and betting data* [Dataset]. Kaggle.<https://www.kaggle.com/datasets/tobycrabtree/nfl-scores-and-betting-data?select=spreadspoke_scores.csv>

Analytics Vidhya. (2016, March 15). *Complete guide to parameter tuning in XGBoost (with codes in Python)*. Retrieved August 11, 2025, from<https://www.analyticsvidhya.com/blog/2016/03/complete-guide-parameter-tuning-xgboost-with-codes-python/#What_Is_XGBoost>?

Walters, T. (n.d.). *NFL betting model*. Kaggle. Retrieved August 11, 2025, from<https://www.kaggle.com/code/twalters20/nfl-betting-model#Feature-and-Model-Testing>

*Using Python to mimic NFL sportsbook odds determination*. (2023, October 6). *Plain English*.<https://python.plainenglish.io/using-python-to-mimic-nfl-sportsbook-odds-determination-2d96dc84ecab>

1. APPENDICES

Provide your codes here. And also put all the relevant information (what programming language and version the code is written for, any environment setup variables, if so what are they set to, etc). No limit on appendices, but do not make it toooo long (e.g. 20 pages is tooo long). Make sure your code is copy/pasteable so we can copy and run it if we want to.