Kyle Ford

CPSC 5240

Due Date: 3/24/2023

UTC Department of Computer Science

Evaluating and Developing Models to Predict the Outcomes of NFL Games

Abstract:

The National Football League (NFL) represents one of the largest professional sports leagues in the world. Its popularity continues to grow, making it one of the most prominent sports and entertainment platforms around the globe. As such, the ways in which the game is consumed has changed dramatically with access to technology and other digital services. One of the biggest ways in which the way the NFL is consumed has changed is through the legalization of sports betting in the United States. In 2021, sports betting in the United States became legal at the federal level, opening up massive revenue opportunities for the league and sportsbook companies. A billion-dollar industry was born with the decision to legalize sports betting. With the introduction of sports betting came an increased appetite to be able to predict the outcomes of NFL games. Sports bettors and sportsbooks became extremely interested in any advantage they could gain in correctly picking the outcome of a game. Hence, it can be argued that models designed to predict the outcome of NFL games have never been more in demand, and their accuracy never of more vital importance. This paper will review several currently existing models and evaluate the efficacy with which they predict NFL match outcomes. These models vary in their approach and include linear, logistic, random forest, and stochastic models, among others. Finally, we develop our own logistic regression model using game log data from the 2010-2019 NFL seasons in an attempt to develop a high accuracy model that can predict the outcome of NFL games. Special attention is paid to specific predictive attributes which drive the model.

Introduction:

The prevalence of the National Football League (NFL) in America's popular culture ecosystem cannot be overstated. Simply put, the NFL dominates the American sports landscape, and football constitutes one of the most popular sports in American society. As such, the prevalence of the preeminent professional league provides a barometer of the sport's overall popularity. As the NFL is quite popular, there is a large emphasis on being able to predict the winner of an upcoming match. Fans want to be able to predict how their favorite team will perform and media personalities and outlets create revenue and content by predicting the outcome of games. Furthermore, the federal legislation legalizing sports betting in America in 2019 provided another massive revenue stream to state governments and the NFL. It is estimated by ESPN that the NFL makes an estimated $2.3 billion per year from sports betting. As such, the ability to accurately predict the outcome of a game has never been more important from a

financial perspective. At the time this paper is being written, thirty-six out of 50 states plus Washington D.C. have legal sports betting. Simply put, there is a lot of money on the line when it comes to predicting the outcomes of NFL games.

Current research on the topic yields greatly different model outcomes. Literature for this study includes linear models featuring a range of accuracy values from 80.65% to 90.32% and features one study reporting an absolute error of 10.68 [1] [4] [10]. Furthermore, literature detailing logistic regression models reports accuracy scores ranging from 63.33% to 90.2%, depending on the model used [2] [3] [7]. Other studies reviewed for this project include a random forest model which updates a team's win probability depending on certain game factors [5], a Gaussian model [6], a stochastic model [8], and a comparative analysis of current deep learning models [9]. While this paper will focus on developing linear and logistic regression models to predict the outcomes of NFL games, we review the results of these studies to gain a more holistic understanding of the landscape of the field.

Finally, using game log data from the 2010-2019 seasons, we seek to create a logistic regression model capable of categorizing which team, home or away, will be the winner of a game based on a variety of input features. These features include home and away rushing attempts and yards gained, home and away passing yards gained, and home and away turnovers. The training set contains the binary classifiers -1 and 1, with -1 denoting the home team won the match, and 1 denoting the away team won the match. These classifiers compose the output labels on which our logistic model will be trained. The goal is to develop a logistic regression model capable of predicting the outcome of an NFL game with high accuracy and a small number of input feature values.

Methodology:

i. Defining key terminology

Before we begin with our analysis, let us first define some key semantics useful to this study. First, in football, one team is said to be in possession of the ball. That team is said to be on offense. While on offense, the team in possession has four plays to gain ten yards or more, which results in a first down. The length of the field is one-hundred yards, and if the team on offense traverses the entire length of the field, they score a touchdown, and earn six points. In the event of a touchdown, the scoring team has the option to kick an extra point, which is worth one point, or attempt to score another touchdown from two yards out, which earns them two points. Teams alternate possession of the ball until the conclusion of the game. At the conclusion of the game, the team with the most points is declared the winner, and the other team is the loser.

Second, there are a few key semantics which must be defined for when a team is in possession of the football. As mentioned, when a team is on offense, they have four attempts at gaining ten or more yards in order to gain a first down. The different play types a team can choose to run can be divided into various categories. First, a team can choose a rushing play. This involves handing the ball off to another player on the team behind the line of scrimmage. The line of scrimmage is a horizontal plane extending from one side of the field

to the other denoting the position of the ball at the start of a play. A rushing attempt, therefore, is defined as any play where an offensive player attempts to gain yards by advancing the ball from behind the line of scrimmage. Consequently, the number of yards a player advances the ball from the line of scrimmage is calculated in a statistic known as rushing yards. Second, a team on offense may attempt a passing play. This involves a player, generally the quarterback, attempting to throw the ball to a player on his team beyond the line of scrimmage. Such a play is known as a passing attempt. As such, the number of yards the ball is advanced upon a successful passing play is encapsulated in a statistic known as passing yards. Both rushing and passing yards will be input features for our developed logistic model. These statistics are further classified into home and away rushing and passing yards, with home designating the team that is playing in their local stadium, and away designating the team that has traveled to take part in the game.

Finally, one key defensive statistic that must be defined is turnovers. A defensive team earns a turnover when they take the ball away from the team on offense. Generally, turnovers come in the form of interceptions or fumbles. An interception occurs on a passing play where the team on offense throws the ball to a player on defense. A fumble occurs when the team in possession drops the football before being declared down by contact. In this event, the defending team can pick the ball up and claim possession, resulting in a recovered fumble. Turnovers are included as input features because they play a drastic role in determining where on the field a team takes over control of the ball, which can lead to high probability of scoring scenarios.

ii. Examining and Evaluating the Dataset

The data used to conduct this study comes from the URL https://www.kaggle.com/datasets/davidsasser/nfl-game-stats-20102019. The complete data set is comprised of ten .csv files. Each .csv file contains the complete game log data from a single NFL season going back to the year beginning in 2010. Since there are ten total .csv files, there is complete data on ten seasons, specifically the 2010 through 2019 seasons. One of the data preprocessing steps that will be taken will be aggregating data from all of the seasons into a single data structure. In this way, we examine data from the ten seasons independently of the season during which the game occurred.

With the data being contained in a .csv file, the data is stored in a simple table structure. As such, each row represents a different entry in the table, or a different game in the season in our specific case, and each column represents an attribute of that game. There are some important attributes of each game that will be focused on more than others for the purposes of our study. Specifically, the columns H-RushYards, H-PassYards, H-Turnover, A-RushYards, A-PassYards, and A-Turnover will serve as the feature columns for our logistic regression model. This means that we will attempt to predict the winner of NFL matches based only on the number of rushing and passing yards a team earns, and how many turnovers they force the other team to commit. In this way, we define a simple logistic regression model that is dependent on only a small number of input feature values. It is also important that the home and away score columns be omitted as feature values, as the score of

the game does not hold any predictive value. Using these columns as feature variables would be to define a model that already has the correct answers. Thus, these columns are omitted and used only for preprocessing techniques.

Finally, the column Result will serve as the target variable of our model. This is a binary column containing -1 and 1. The column shows -1 when the home team is the winner, and 1 when the away team is the winner. This serves as a simple binary classifier by which we can classify the outcome of our experiment. Note, however, that there are certain rows in the table that have the value 0 in this column. As another data preprocessing step, we will convert these values to either -1 or 1, whichever is appropriate depending on the score of the game.

Review of Current Models

i. Linear Models

First, let us review linear models predicting the outcomes of NFL games. A traditional linear system uses a Multiple Regression approach with more than one predictor variable and the coefficients of each regression variable being additive. [1] summarizes this process and describes a hybrid linear model that also incorporates the K-Nearest Neighbors technique to develop a new model. As a frame of reference, [1] also reports traditional linear models accurately predicting the results of NFL games with roughly 60%, depending on the feature variables used. This study incorporates a total of twenty-one feature variables in its analysis, which are to be used as input variables for the regression model. This is substantially more than is fitted for our study to be examined later and includes feature variables such as points scored by each team, win/loss record of each team in games played in a season prior to the game currently being played, team rating using various indexes to rank teams, and the betting point spread of a game [1]. This model combines a traditional linear approach by adding a K-Nearest Neighbors algorithm which classifies attributes together and computes the weights according to each cluster's similarities. This study reports an 80.65% prediction accuracy, a significant improvement over traditional linear models.

[10] takes another hybridized approach at predicting the outcomes of NFL games. This model combines traditional linear regression techniques with an artificial neural network to obtain its results. This model takes in much fewer overall feature variables than [1] describes. The features included for this model include total yardage differential, rushing yardage differential, time of possession differential, turnover differential, and home or away scores [10]. The model also emphasizes additional feature variables including players performance index, bookmakers betting spread, and moving averages. Note that this model focuses on the differential in statistics between the two teams, rather than the raw value of each team's output. The neural network is responsible for finding relationships between the feature variables and weighting them appropriately for the linear regression techniques. The artificial neural network uses a sigmoid function as its activation function, which is appropriate since it is dealing with probabilities. Overall, the model produced a 90.32% accuracy rate on the testing set, which,

again, is an improvement over traditional linear models. Note, however, that this model featured far fewer over feature variables than the model described in [1].

ii. Logistic and Machine Learning Models

Now, let us consider logistic models predicting the outcomes of NFL games. These are the models that we will compare our results to once we describe our proposed model. [3] describes a study where multiple models are studied, both logistic regression techniques and machine learning techniques. Specifically, the study builds out an SVM, Random Forest, ANN, LSTM, and RNN model and compares each model's accuracy. While all of the different models have similar accuracy, the logistic regression model performs the best in this study, with an accuracy rate of 63.33% [3]. This study was more inclusive in terms of the number of features it took in. in total, it considered a wide array of feature values, including home and away point differential, home and away total wins in the season, and home and away yards gained.

These results are consistent with other results that are obtained using machine learning methodology. [9] is another quantitative study performed by comparing different machine learning models and their accuracy scores. This study uses a train-test split of 70%-30% and performed 10-fold cross-validation for all of the methods tested. The methods tested include SVM, Nearest Neighbor, Gaussian, Decision Tree, Random Forest, AdaBoost, Naïve Bayes, QDA, and Neural Network [9]. Overall, the Naïve Bayes model performed the best in terms of accuracy at 67.53%, followed by AdaBoost at 66.35%. At the low ends of the accuracy spectrum were Gaussian Processes at 44.64%.

Another study that was evaluated for the purpose of fitting a logistic regression model to our data was [7]. One unique feature of this study is it was performed on Canadian football teams rather than American NFL teams. A logistic regression model is fit based on a number of features, including the difference in rushing and passing yards, the difference in total turnovers, and the difference in the number of sacks [7]. These serve as the inputs for the logistic regression. In sum, the author measured the model's performance based on the strength of different teams. The author hypothesized that different teams would perform differently under the model due to having different strengths and weaknesses [7]. Stronger teams might rely more on having a higher difference in rushing and passing yardage, which would influence those coefficients differently. This is something that is not assumed for our study. We assume that all teams perform equally based on the distribution of the input variables. In this case, the author finds that the model performs 85.9% for Calgary, 90.2% for Saskatchewan, and 78.8% for Ottawa [7]. We take the average of these scores, 84.97%, and use that as a comparison point for our study.

Experiment

i. Data Preprocessing

The first step in the data preprocessing for this study is to resolve the presence of inappropriate binary classifiers in the 'Results' column of the dataset. This column stores the

winner of the game, encoded as a -1 if the home team wins, and a 1 if the away team wins. However, upon inspection of the data, we find the presence of a small set of games with a 0 classification. This is inappropriate to the study and represents incorrect data. In order to resolve this, we edit our source .csv files such that the zero values are removed and replaced with the appropriate -1 or 1 classifier. In this way, when the logistic regression is run, we will get tidy, representative results.

The next step we take in processing the data is importing the pandas library into Python. This enables us to create data frames to manipulate our data. We import the pandas library as shown in figure one, under the traditional alias pd.

```
#Import libraries
import pandas as pd
```

Figure 1: Importing the Pandas library

Once the pandas library has been imported, we can begin to import our specific data. We use the .read_csv function to import each of the ten .csv files that comprise our complete data set. Each data set is saved in a data frame corresponding to the year of the season it holds data for. Figure 2 illustrates how these data frames are created.

```
#Import datasets

df_2010 = pd.read_csv("game_stats_2010.csv")
df_2011 = pd.read_csv("game_stats_2011.csv")
df_2012 = pd.read_csv("game_stats_2012.csv")
df_2013 = pd.read_csv("game_stats_2013.csv")
df_2014 = pd.read_csv("game_stats_2014.csv")
df_2015 = pd.read_csv("game_stats_2015.csv")
df_2016 = pd.read_csv("game_stats_2016.csv")
df_2017 = pd.read_csv("game_stats_2017.csv")
df_2018 = pd.read_csv("game_stats_2018.csv")
df_2019 = pd.read_csv("game_stats_2019.csv")

#print head of first dataframe (2010 season)
df_2010.head()
```

Figure 2: Importing the Data Sets for the 2010-2019 NFL Seasons

Once all of the .csv files have been read into Python and stored as data frames, we need to combine them into a single data frame. This is accomplished using the pd.concat function, with the list of our ten data frames passed in as a list. In effect, this concatenates the various data frames into one data frame, df_combined.

Once df_combined has been created, the next step in preprocessing the data is to shuffle the rows in the data frame. This ensures that our training and testing sets are representative samples, and not swayed by the season that they come from. In order to do this, we call the .sample() method on our df_combined data frame. Figure 3 shows both the concatenation and the shuffling steps.

```
#data preprocessing

#concatenate all dataframes to combine data from all seasons
df_combined = pd.concat([df_2010, df_2011, df_2012, df_2013, df_2014, df_2015, df_2016, df_2017, df_2018, df_2019])

#shuffle rows in dataframe to ensure proper training and testing samples
df_combined = df_combined.sample(frac=1, random_state=42).reset_index(drop = True)

#print head of merged dataframe
df_combined.head()
```

Figure 3: Concatenating and Shuffling the Data

ii. Model Fitting

Now that the data has been preprocessed, we can begin to fit our logistic regression model. The first thing we will do to define our model is define our feature variables and our target variable. We define a list of the columns that will serve as feature variables for this study. As defined earlier, they are home and away rush yards, home and away pass yards, and home and away turnovers. We define these columns as X for our feature variables. Also, we define the result column, which contains our binary classifier, as y, or our target variable. Figure 4 shows the process by which this is accomplished.

```
#split dataset into features and target variable
feature_cols = ['H-RushYards', 'H-PassYards', 'H-Turnover', 'A-RushYards', 'A-PassYards', 'A-Turnover']
X = df_combined[feature_cols] #features
y = df_combined.Result #Target Variable
```

Figure 4: Splitting the Dataset into Feature and Target Variables

Now that our X and y data has been defined, we can create our training and testing sets. The first step in this process is to import the train_test_split function from the sklearn.model_selection package. For the purposes of this study, we will use a 75%-25% train test split. We implement our X_train, X_test, y_train, and y_test variables using the train_test_split function, and passing in X, y, and our test_size as arguments. Figure 5 shows this process.

```
#Split X and y into training and testing sets
from sklearn.model_selection import train_test_split

#75% of data used for model training, 25% for model testing
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25, random_state = 16)
```

Figure 5: Split X and y into training and Testing sets

The final step in creating our model is to now define a logistic regression object and fit it using our training data sets. In order to do this, we first import the LogisticRegression function from sklearn and instantiate an object using the LogisticRegression function. We call our logistic regression object logreg for clarity. We use the .fit() method to fit the data to the model, passing in both our X and y training data sets as arguments to the function. At this point, the model is complete and has been trained. We can now use it to generate predictions using the .predict() method. We define y_pred using this method and passing in our X_test dataset as an argument. Figure 6 shows this process.

```
#import LogisticRegression function from sklearn
from sklearn.linear_model import LogisticRegression

#instantiate the model using default parameters
logreg = LogisticRegression(random_state = 16)

#Fit the model
logreg.fit(X_train, y_train)

#use the model to predict
y_pred = logreg.predict(X_test)
```

Figure 6: Fit the Logistic Regression Model

iii. Model Evaluation

```
#confusion matrix

#import metrics class
from sklearn import metrics

cnf_matrix = metrics.confusion_matrix(y_test, y_pred)
cnf_matrix
```

Figure 7: Define Confusion Matrix to Evaluate Model

```python
#visualize confusion matrix

#import classes
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

class_names = [-1,1] #name of classes
fig, ax = plt.subplots()
tick_marks = np.arange(len(class_names))
plt.xticks(tick_marks, class_names)
plt.yticks(tick_marks, class_names)

#create heatmap
sns.heatmap(pd.DataFrame(cnf_matrix), annot=True, cmap = "YlGnBu", fmt = 'g')
ax.xaxis.set_label_position("top")
plt.tight_layout()
plt.title("Confusion Matrix", y = 1.1)
plt.ylabel("Actual Label")
plt.xlabel("Predicted Label")
```

Figure 8: Confusion Matrix Visualization Code

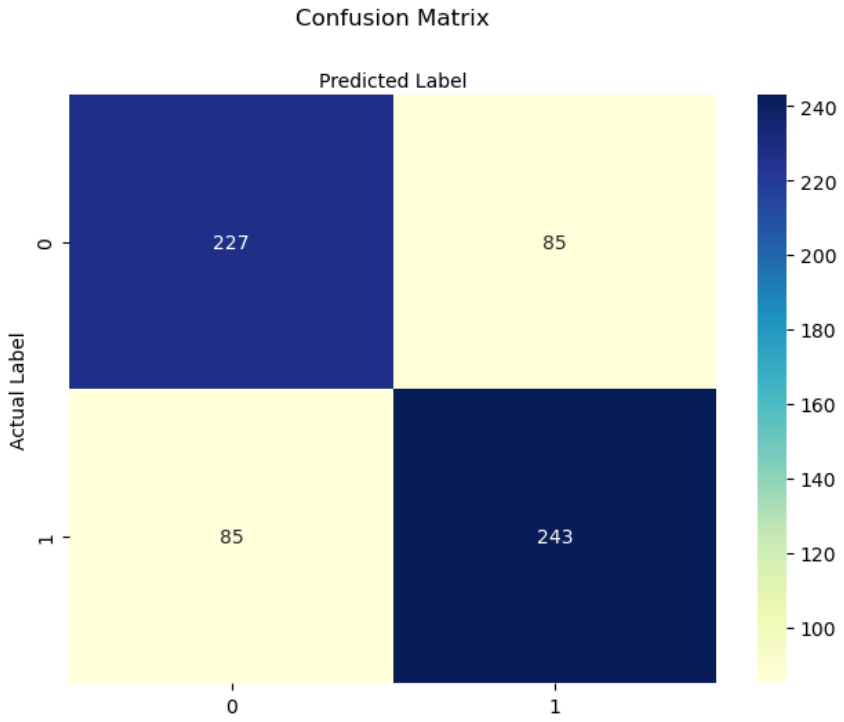

Figure 9: Confusion Matrix

```
#Evaluate confusion matrix
from sklearn.metrics import classification_report

target_names = ['Home team won', 'Away team won']
print(classification_report(y_test, y_pred, target_names = target_names))
```

```
              precision    recall  f1-score   support

Home team won       0.73      0.73      0.73       312
Away team won       0.74      0.74      0.74       328

     accuracy                           0.73       640
    macro avg       0.73      0.73      0.73       640
 weighted avg       0.73      0.73      0.73       640
```

Figure 10: Classification Report

```
#Evaluate accuracy score
from sklearn.metrics import accuracy_score

print("Accuracy score: ", accuracy_score(y_test, y_pred))
```

```
Accuracy score:  0.734375
```

Figure 11: Evaluating the Model's Accuracy Score

Discussion

Using the results obtained from the confusion matrix, classification report, and accuracy score in Figures 9, 10, and 11, we achieve adequate results compared to similar models found in the literature. The model obtains an accuracy score of 73.43% on the test dataset, which is in line with similar studies. At the low end of the spectrum, [9] reports accuracy scores between 44.64% and 67.53% and [3] reports an accuracy score of 63.33% on the test dataset. On the high end of the spectrum, [1] reports an accuracy score of 80.65%, [7] reports an accuracy score of 84.97%, and [10] reports an accuracy score of 90.32%. In terms of accuracy, the model designed in this study represents the median value of models studied in the literature.

One of the areas of improvement inherent to this model over others studied is feature selection. Certain models studied considered a wide range of feature variables. This large number of variables contributes to computational inefficiency and data dependency that impact their overall efficacy. One of the goals of our implementation was to limit the number of input features such that no data dependencies could develop in our data. Simultaneously, we sought to maintain an appropriate level of testing accuracy such that the model is useful in predicting the outcome of NFL matches. Considering the results, we can conclude that this model achieves both of these goals. Feature selection is limited to six input variables, a low value compared with other models studied for comparison. Furthermore, we achieve a testing accuracy comparable to

other models studied. Thus, we can conclude that this model reasonably improves upon existing methods for predicting the outcomes of NFL matches.

While this model generally improves upon other models studied, there are a few areas where further study could improve the model. First, consider the nature of the feature variables selected for the model. They include descriptive statistics regarding each teams' performance in the game. As such, their values are not known prior to the game's conclusion. For this model to hold reasonable predictive power, they would need to be known prior to the match. Thus, we conclude that further research to develop metrics approximating expected values for these input variables is appropriate. This would undoubtedly introduce an extra source of bias to the model, however; such a metric would increase the overall predictive power of the model. Second, the model architecture is quite simplistic in its implementation. We consider a classification model using a basic logistic regression to classify the winner and loser of the game. While we achieve acceptable test accuracy using this method, the model could be improved by expanding the architecture and introducing more advanced machine learning methodology. Specifically, introducing an LSTM (long short-term memory) component to the architecture could specifically aid in classifying the winning and losing team with respect to previous game results. In this way, we can expect to improve our test accuracy with more advanced methodology.

Conclusion

In conclusion, models predicting the outcomes of NFL games are in high demand in the current market. Fans and teams have always sought a competitive edge in being able to predict which team was more likely to win, but the introduction of federal legislation in the United States legalizing sports betting created an immense market for predicting the outcomes of sports matches. The NFL represents the most popular professional sports league in the country, and, as such, creates a massive demand for these models. In reviewing the literature of existing models aimed at predicting the winner of an NFL match, we make a few observations. First, there are a plethora of models in existence that achieve high test accuracy and have adequate predictive potential. Second, while these models achieve a high testing accuracy, they utilize a large number of input variables to achieve their results. Thus, we establish the goal of maintaining high test accuracy while reducing the number of input feature variables.

In designing the model, we observe in the literature that, generally speaking, logistic and machine learning models tend to feature higher testing accuracy than linear models. As such, we design a logistic regression model meant to classify the winner and loser of an NFL match. The model is fit using game log data from NFL matches over a decade long period. From our dataset, we define six input feature variables that serve as inputs to our model. They include home and away pass yards, home and away rush yards, and home and away turnovers. The model is designed using Python's sklearn library.

In evaluating the model's efficacy, we note that the model achieves a testing accuracy of 73.43%. Thus, we conclude that we have achieved our goal of maintaining testing accuracy while significantly reducing the number of feature variables. These results are confirmed through a number of analytical metrics, including the confusion matrix (figure 9), classification report

(figure 10), and accuracy score (figure 11). While this model accomplishes the goals set forth for this study, we note that feature selection can serve as an impediment to the overall predictive potential of the model. One area for further research is developing predictive metrics for the feature values that can serve as input variables for the model. Furthermore, we note that the model can be improved through the expansion of the architecture and inclusion of more advanced machine learning techniques.

References

[1] - A. O. Uzoma and N. E. O., “A Hybrid Prediction System for American NFL Results,” International Journal of Computer Applications Technology and Research, vol. 4, no. 1, pp. 42–47, 2015.

[2] - B. L. Boulier and H. O. Stekler, “Predicting the outcomes of National Football League Games,” International Journal of Forecasting, vol. 19, no. 2, pp. 257–270, Apr. 2003.

[3] - Bosch, Pablo. "Predicting the winner of NFL-games using Machine and Deep Learning." Vrije universiteit, Amsterdam Feb. 2018

[4] - D. Harville, “Predictions for National Football League Games via Linear-Model Methodology,” Journal of the American Statistical Association, pp. 516–524, Mar. 2012.

[5] - D. Lock and D. Nettleton, “Using random forests to estimate win probability before each play of an NFL game,” Journal of Quantitative Analysis in Sports, Mar. 2014.

[6] - H. Stern, “On the Probability of Winning a Football Game,” The American Statistician, vol. 45, no. 3, pp. 179–183, Feb. 2012.

[7] - K. A. Willoughby, “Winning Games in Canadian Football: A Logistic Regression Analysis,” The College Mathematics Journal, vol. 33, no. 3, pp. 215–220, Jan. 2018.

[8] - M. Mohsin and A. Gebhardt, “A stochastic model for NFL games and point spread assessment,” Journal of Applied Statistics, Sep. 2022.

[9] - R. Beal, T. J. Norman, and S. D. Ramchurn, “A Critical Comparison of Machine Learning Classifiers to Predict Match Outcomes in the NFL,” International Journal of Computer Science in Sport, vol. 19, no. 2, 2020.

[10] - O. U. Anyama and C. P. Igirl, “An Application of Linear Regression & Artificial Neural Network Model in the NFL Result Prediction ,” International Journal of Engineering Research & Technology (IJERT), vol. 4, no. 1, Jan. 2015.