Group 67 Final Report

Introduction/Background

Literature Review

In American Football, the idea of the “best play” is a culmination of factors: the game score, the weather, opposing line-up, etc. Prior work has explored how play-by-play choices affect the overall win chances of a game, with one model reaching accuracy levels of 84% [1]. Other models have used neural networks to analyze the number of yards a play would gain (accuracy of 22%) or what play would be used in a certain scenario (accuracy of 72%) [2][3]. There still exists space for more in-depth exploration of predicting success of a specific play at a specific point in the game. Our team aims to answer the following question: can an ML model predict success of a down at any given point in the game?

Problem Definition

Problem

In the NFL, predicting play outcomes is crucial for making strategic decisions, yet current methods rely on basic statistics or intuition, often missing critical factors like formations, player matchups, and in-game context. This leads to less accurate predictions and suboptimal game-time decisions.

Motivation

A machine learning model can leverage vast NFL data—such as player performance, team tendencies, and situational variables (e.g., down, distance, weather)—to predict whether the offensive team will gain a first yard given the circumstances before the play starts. By delivering real-time decision-making for coaches, the model could provide insights, improve game analysis, and give competitive advantages for coaching teams.

Methods

Dataset Description

The team’s chosen dataset is regular season play of the NFL from 2009-2016 from Carnegie Melon linked here. It breaks down every play of a game with features such as game quarter, number down, name of the play, yards gained, etc. There are 102 features that can possibly be used in our model to provide information regarding the status of a game at the incidence of each play.

Cleaning and Processing

Our first step was to clean the data. To avoid predicting outcomes based on future information, all the features that would not be known before the play started (yards gained, play called, etc.) were filtered out using Python’s Pandas library. Additionally, all the plays that could not result in a first down, like when a team was punting, kicking a field goal, or taking a timeout, were filtered out. To further clean the data, we also removed any samples that had missing data or null values.

The remaining features are a mix of both numerical and categorical features. The numerical features have been normalized to prevent features like “Drive” contributing more to the model’s decision due to its higher scale than features like “Down”, which is a value only from 1-4. For the categorical features, we used OneHotEnconding functions from scikit-learn's library to convert the data into a binary 0 and 1. For example, the feature “OffensiveTeam” was broken into 32 separate features for each team (OffensiveTeam_CIN, OffensiveTeam_DEN, etc.) and the datapoint was set to 1 if the team is the offensive team for that play and 0 otherwise. Thus, the models we use will be able to make connections from the categorical data.

Furthermore, after analyzing our dataset and the large number of features present, we carried out preliminary forward feature selection so that in the future we could focus on only the most impactful ones. This will improve our model’s performance and also reduce complexity. After running our forward feature selection to select 10 features we found that all 10 consisted of a ‘passer_id”(the player passing the ball). This indicates that the ‘passer_id’ feature was highly impactful on the model’s prediction accuracy. In the future we will increase the number of features selected so that we will be able to run our decision tree classifier on only a selected number of features rather than the entire cleaned dataset.

Results and Discussion

Model 1: Decision Tree

We chose a decision tree because the dataset can be split into subsets to best separate the classifications. A decision tree is a supervised learning algorithm best used for predictive modeling. The algorithm creates a tree structure with feature nodes, where the leaf node is the final prediction or classification.

Model 1: Results

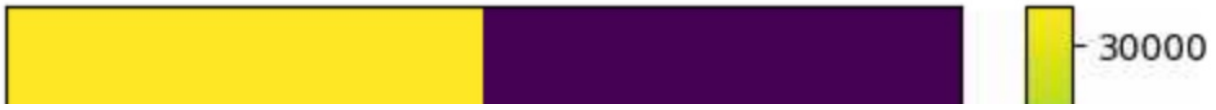
The model was trained on preprocessed and cleaned data from our dataset. Using default hyperparameter values for the decision tree, the accuracy, precision, and recall values are listed below:

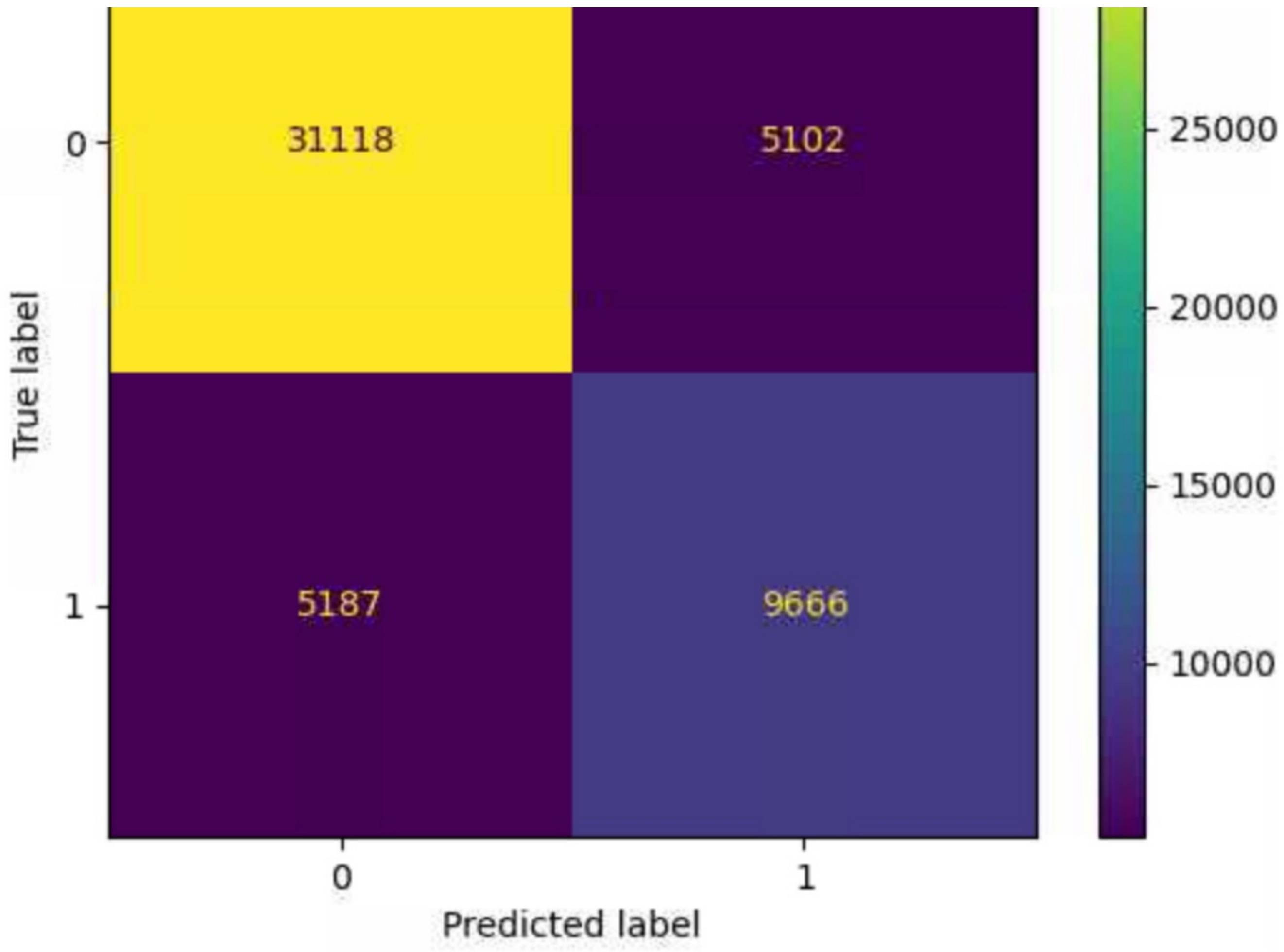
Figure 1: Decision Tree performance meterics.

	precision	recall	f1-score	support
accuracy			0.8000	51073
0	0.86	0.86	0.8600	36220
1	0.65	0.65	0.6500	14853
macro avg	0.76	0.75	0.7600	51073
weighted avg	0.8	0.8	0.8000	51073

These results indicate a high accuracy (>= 80%) for the overall model. However, these results vary based on the positive and negative predictions of the model. This can be further examined through the Confusion Matrix below:

Figure 2: Confusion Matrix



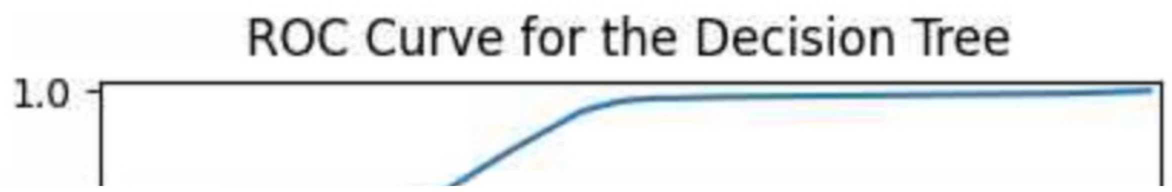


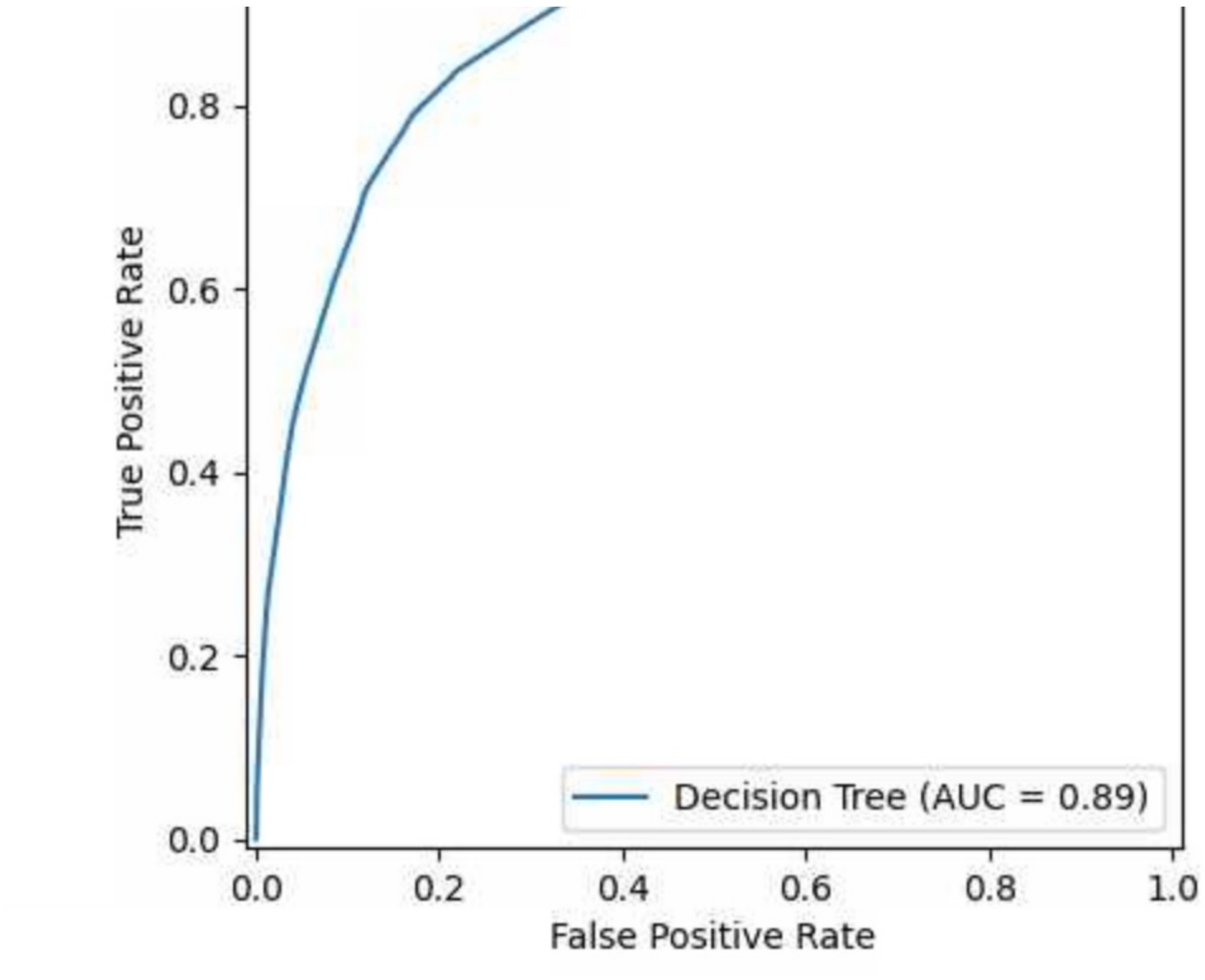
This indicates that our model is stronger in predicting when a first down did not occur in comparison to when a first down does occur. This is illustrated in the precision and recall statistics of the model:

1. True Negative Precision and Recall: 0.86
2. True Positive Precision and Recall: 0.65

This indicates that there may be some features included in the model that were most likely very strongly correlated with an unsuccessful down, but confounding features that made it more difficult to predict success.

Figure 3: Receiver operating characteristic (ROC) curve





This is reflected in the ROC curve for the decision tree. The final AUC value is 0.89, indicating a high sensitivity vs specificity ratio. This model is therefore very good at classifying between the successful and unsuccessful first down attempts.

To improve these values, our group began trying different hyperparameters to improve model accuracy. In particular, we changed the max_depth of the decision tree to be max_depth = 10 and looked at how accuracy and the f1 score changed:

Figure 4: Decision Tree performance meterics with max depth 10.

	precision	recall	f1-score	support
accuracy			0.8300	51073
0	0.88	0.88	0.8800	36220

1	0.71	0.71	0.7100	14853
macro avg	0.79	0.79	0.7900	51073
weighted avg	0.83	0.83	0.8300	51073

Based on these results, it is clear that changing the max depth from the default value improved accuracy. These results thus far are not as satisfactory as we would like, as the decision tree is still unable to classify a successful down to a satisfactory accuracy (> 80%). Even with an inital attempt at hyperparameter optimization of the max_depth, the model is still limited by the other default hyperparameters and the large number of features provided to the model. The most impactful features were the passer id's which is expected based on prior knowledge of the game, as the quarterback's passing ability is crucial in determining the success of the play.

Other ‘max_depth’ values tested out for our Decision Tree Classifier:

Figure 5: Decision Tree visualization with max depth 3.

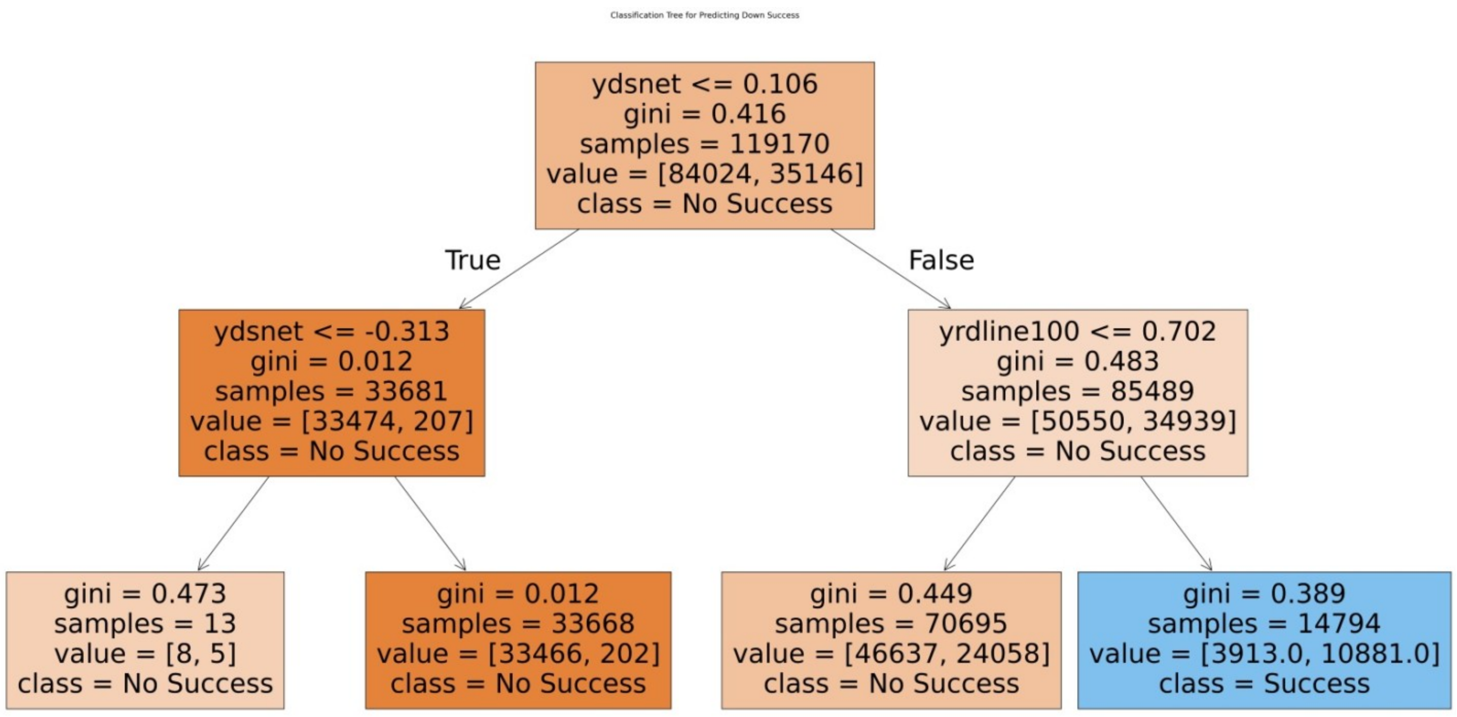
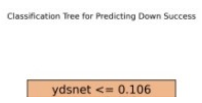
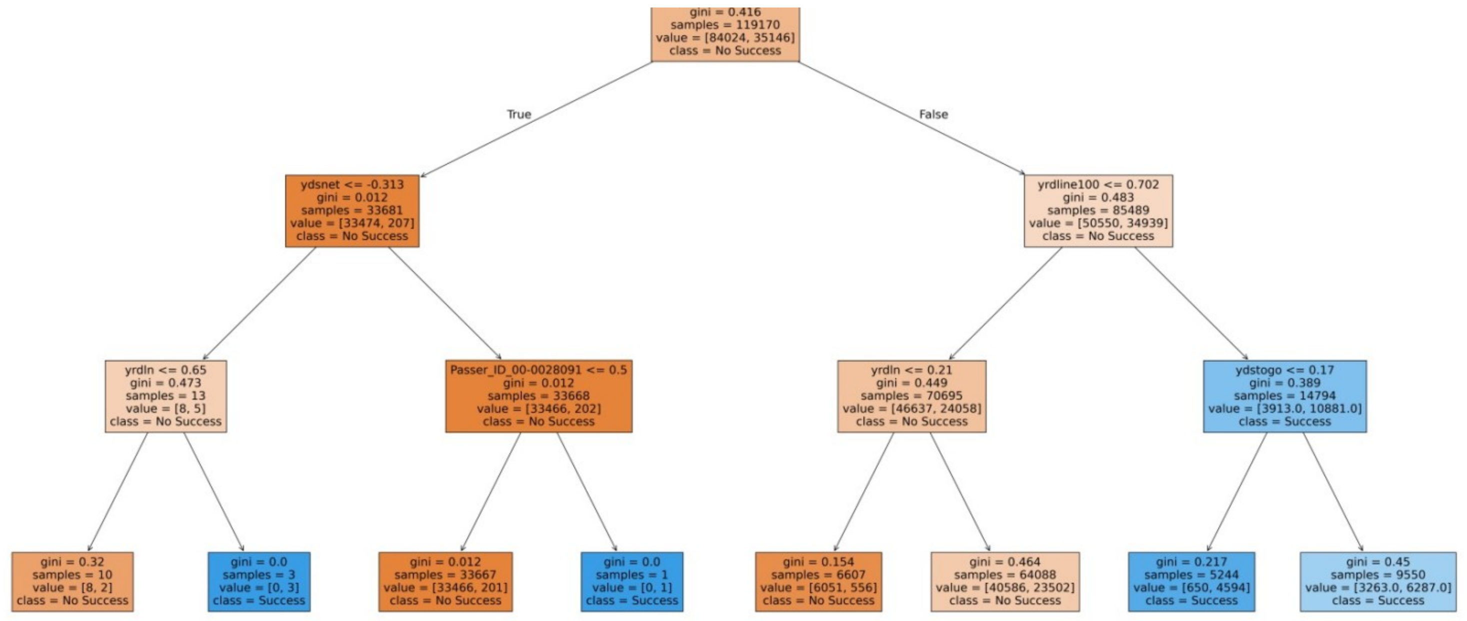


Figure 6: Decision Tree visualization with max depth 4.





This approach begins to assess hyperparameter optimization in order to choose model parameters that are best suited to the data. However, manually changing the max_depth to an arbitrary value is most likely not the best approach, as it can limit the range of explored hyperparameter values and reduces the likelihood of finding the best combination of hyperparameters. This approach only maximizes accuracy by tuning a single hyperparameter when there are many other parameters that can be tuned. Therefore, conducting a hyper parameter search that tests various combinations of hyperparameter values and selects the most optimal set will be an important next step when trying to improve model performance. Further work can be done to tune more hyperparameters for a decision tree, such as min_samples_split and max_features.

Currently, the visualizations method for the decision tree is showing the tree with the features that are used to split each branch. This method is effective in showing what features are considered important by the model and which features are the most significant in classifying the trials. We would argue that this visualization method is relevant, as it can guide feature selection and indicate which features are more significant for determining the success of a down. This is helpful in better understanding the model's goals, as it allows for a graphical representation of what features and quantitative metrics are deemed useful to a model.

Model 1: Discussion

To improve this approach, more hyperparameters should be tuned to see how to optimize the model.

Furthermore, we should do more feature engineering, such as backward feature selection, to only provide the model with important features. Despite the data cleaning we did to remove any ‘future-looking' features, there may be features that are introducing noise to the model when it attempts to classify a successful down. Performing more feature engineering may yield a more accurate model.

In terms of the decision tree as a model, it is a good introductory model to begin classifying data in terms of a successful down. A decision tree allows features to direct splits in model decision making based on feature importance and requires limited feature engineering in order to produce a good result (as seen by our basic cleaning of data prior to training the model). However, decision trees are prone to overfitting and as small changes in how data is split can lead to large changes in model classification, this tends to produce larger variance in the model, which can limit model accuracy.

This can be overcome by using other types of models to better understand the data. For example, a random forest model generates multiple decision trees to classify data based on ‘majority vote’. This harnesses the basics of the basic decision tree model implemented here but will also allow the model to be more robust to noise in the data and the variance of one singular decision tree. This model also will direct feature selection, as the random forest model takes into account the relative contributions of different features. Furthermore, a more complex model like a Neural Network may be better suited for this data, which tends to improve accuracy and handle large variations in features. Implementing these models will be the next step of our group to achieve our goal of 80% accuracy or higher in down prediction for both a successful and unsuccessful play.

Model 2: Random Forest

Random Forest was chosen because of its resistance to overfitting and ability to handle large datasets. Random forest creates multiple decision trees from a random subset of features in the dataset, reducing the risk of overfitting and improving overall prediction performance. During prediction, the algorithm aggregates the results of all trees to make the final classification.

Model 2: Results

Since Random Forest is unaffected by cardinal ordering, the model has trained the same dataset with binary encoding instead of one hot, reducing the total dimensionality of the dataset. Using default hyperparameter values for Random Forest, the accuracy, precision, and recall values are listed below:

Figure 7: Random Forest performance meterics

Figure 7: Random Forest performance meterics.

	precision	recall	f1-score	support
accuracy			0.8500	51073
0	0.87	0.92	0.8900	36220
1	0.77	0.67	0.7200	14853
macro avg	0.82	0.79	0.8100	51073
weighted avg	0.84	0.85	0.8400	51073

With Random Forest, accuracy increased by 2%. Despite the increase in accuracy, the model become worst at predicting true positives. This can be further examined through the Confusion Matrix below:

Figure 8: Random Forest confusion matrix.

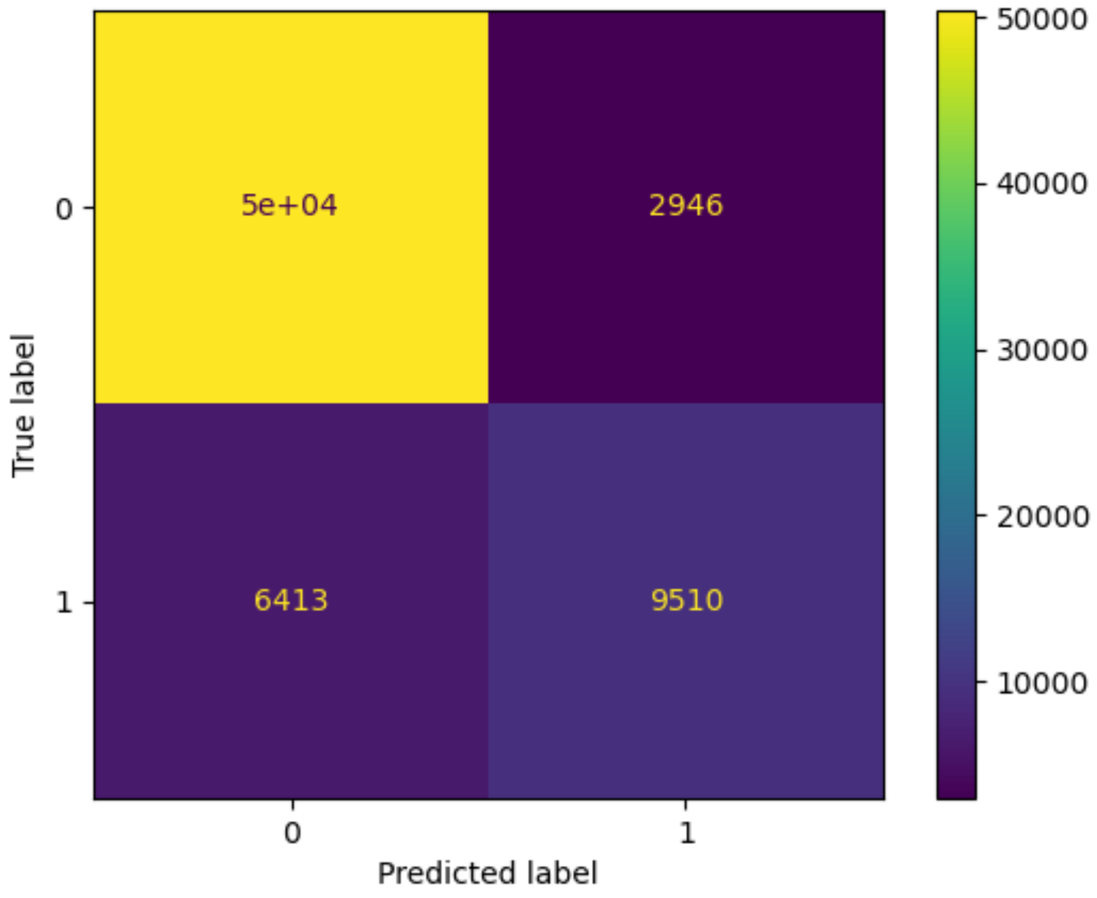
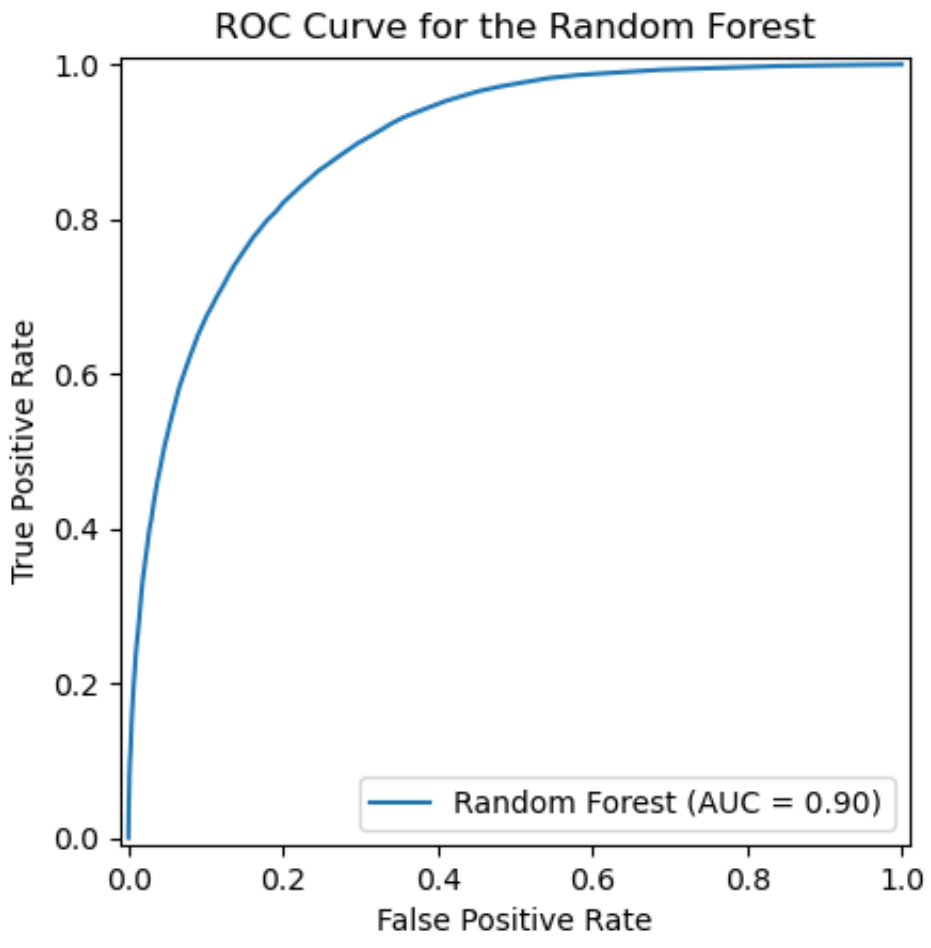


Figure 9: Random Forest ROC curve.



The ROC curve for our Random Forest model, shown in Figure 9, demonstrates the model's effectiveness in classifying whether a football play will result in a first down. The curve displays a sharp ascent towards the top-left corner, indicating high sensitivity (true positive rate) while maintaining a low false positive rate across various classification thresholds. The Area Under the Curve (AUC) is 0.90, signifying that the model has excellent discriminatory ability, with a 90% likelihood of correctly ranking a randomly chosen positive instance (first down) higher than a randomly chosen negative instance (not a first down). This high AUC value shows the robustness of the model in separating the two classes effectively. From this figure, we can show that our Random Forest model has successfully captured meaningful patterns and features in the data, making it well-suited for this classification task. The model's strong ability to balance sensitivity and specificity means that our predictions have a higher probability of being used by football analysts, rather than be discarded due to its unreliability.

Model 2: Discussion

Model 2: Discussion

The Random Forest model achieved an accuracy of 85%, a modest improvement over the Decision Tree's performance. This increase can be attributed to the ensemble approach of Random Forest, which reduces overfitting by averaging predictions from multiple decision trees. However, the model still struggled with predicting true positives, as indicated by a precision of 0.77 and a recall of 0.67 for successful plays. A critical issue identified in the dataset is the inherent imbalance, as it contains a higher proportion of unsuccessful plays. This imbalance skews the model's predictions toward the majority class (unsuccessful plays), leading to reduced effectiveness in predicting the minority class (successful plays). Addressing this imbalance by applying techniques such as oversampling (e.g., SMOTE) for successful plays or undersampling the majority class could help the model better generalize to both classes.

Additionally, increasing the number of decision trees in the Random Forest did not significantly improve accuracy. This suggests that the model's performance is limited by other factors, such as the class imbalance and feature selection, rather than the number of trees. Fine-tuning other hyperparameters, such as max_depth, min_samples_split, and max_features, might yield better results than simply increasing the ensemble size.

Future work should focus on rebalancing the dataset, optimizing hyperparameters through systematic approaches like GridSearchCV, and exploring alternative metrics (e.g., F1-score or AUC) that better reflect the model's performance on imbalanced data. These steps will help ensure the model is equally effective at predicting both successful and unsuccessful plays.

Model 3: Neural Network

A neural network can learn from historical data and patterns to make predictions. Neurons receive inputs, governed by thresholds and activation functions. The final neuron determines the final classification.

Model 3: Results

The Neural Network was set with (64, 32) hidden layers, Adam optimizer, and 0.001 learning rate. The hidden layer activation function uses rectified linear unit function (relu) and the output layer uses logistic sigmoid function. Below are the results of the model.

Figure 10: Neural Network performance meterics.

	precision	recall	f1-score	support
accuracy			0.8600	69307
0	0.89	0.91	0.9000	53384
1	0.62	0.64	0.6600	15923
macro avg	0.78	0.77	0.7800	69307
weighted avg	0.84	0.85	0.8400	69307

The Neural Network's accuracy ranges from 84% to 86%. Similar to model 2, Random Forest, the Neural Network was better at identifying true negatives and worse at identifying true positives. This can be further analyzed in the confusion matrix below.

Figure 11: Neural Network confusion matrix.

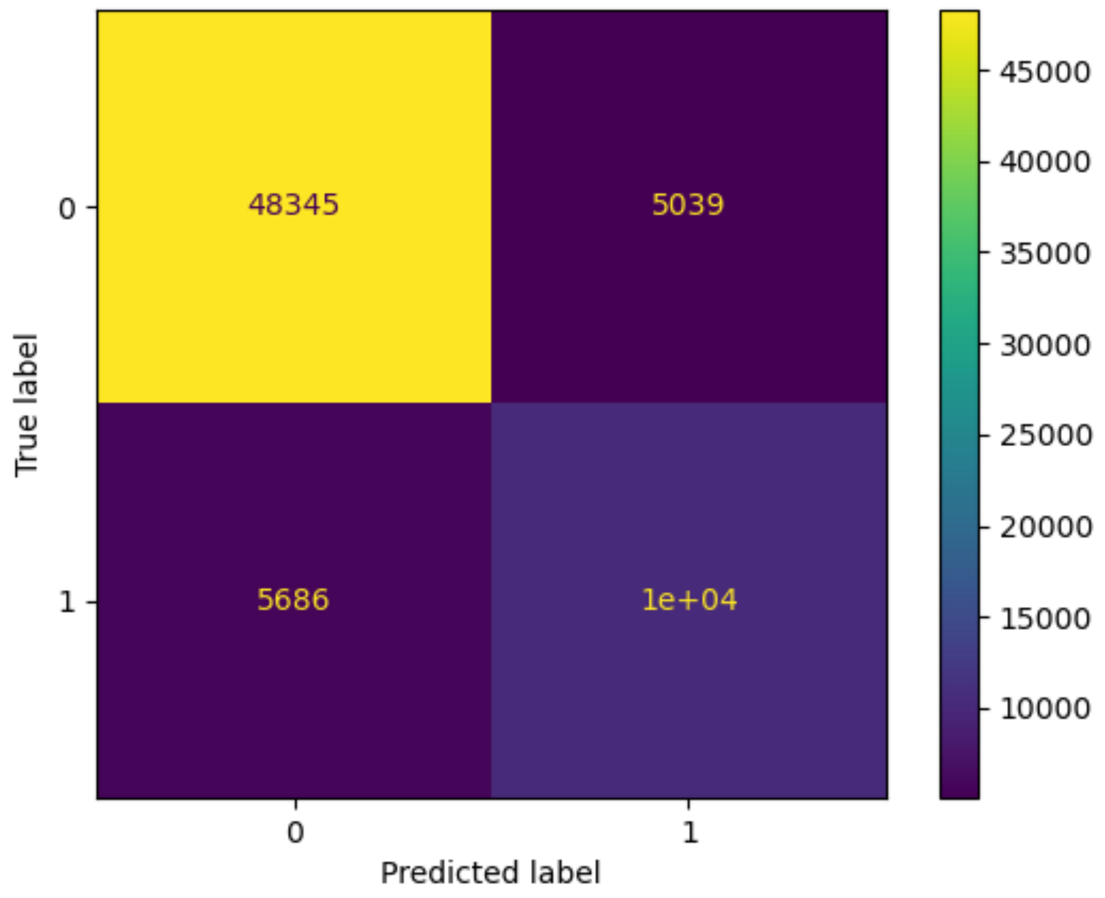
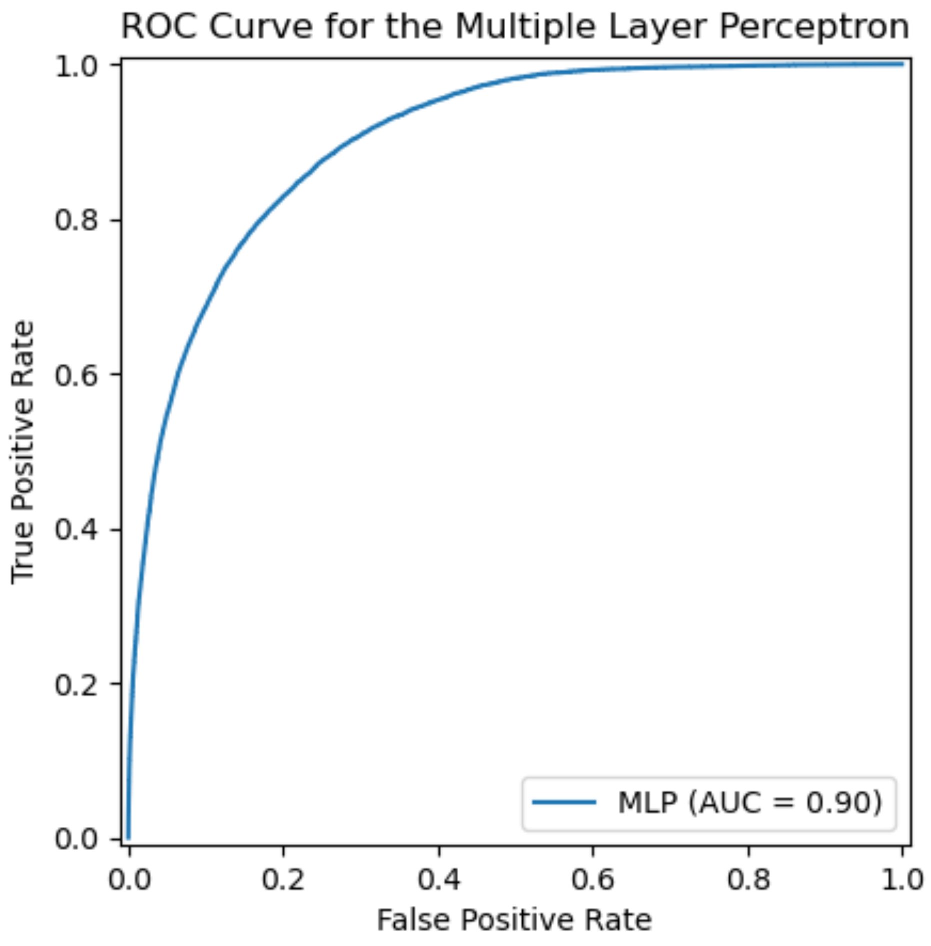


Figure 12: Neural Network ROC curve.



The ROC curve, depicted in Figure 12, reaffirms the model's strong discriminatory capability, with an AUC of 0.90. This indicates that the model can reliably distinguish between plays that result in a first down and those that do not. However, the disparity in class-specific performance raises concerns about class imbalance or insufficient feature representation for plays resulting in first downs. Addressing these challenges through techniques like oversampling the minority class, tuning hyperparameters, or adding domain-specific features could help improve recall and overall performance for identifying first downs.

Model 3: Discussion

From the confusion matrix, we can see that the proportion of true negatives to false negatives (48,345:5,039) is much higher compared to the proportion of true positives to false positives (10,000:5,686). One possible cause for this imbalance might be explained by the dataset having a higher sample size of non-first downs compared to first downs in NFL plays, which could have influenced the model to prioritize classifying a play as a non-first down.

We can see a similar pattern observed in the neural network performance metrics (Figure 10). The model achieves high precision (89%) and recall (91%) for predicting non-first downs but struggles with lower precision (62%) and recall (64%) for predicting first downs. To address this, techniques such as applying class-weight adjustments, introducing a penalty for false positives, or conducting a detailed analysis of scenarios that lead to false positives could help improve the model's performance, particularly for predicting first downs.

Neural Network Advantages

Neural networks are one of the most popular machine learning techniques for a multitude of reasons. Neural networks can evaluate non-linear relationships in the data, while other models might require data conditioning techniques or transformations. They also have strong predictive capabilities and ability to handle complex data, which makes them very good at classifying data with unclear relationships.

Neural Network Disadvantages

While neural networks are powerful tools for classifying data due to its ability to finetune weights and create non-linear relationships, one main disadvantage is their inability to understand the main factors that they use for prediction. They are essentially "black-boxes" when it comes to understanding which features influence the outputted classification. For the NFL specifically, the actual prediction whether a play will result in a first down is not the most important for teams, as running a model to output a prediction is not reasonable during the fast-paced environment of the game. An NFL team will most likely care more about understanding the factors that influence whether a team will achieve first down rather than the ability to predict the first down.

Model Comparison

The decision tree provided a solid baseline, offering simplicity and interpretability. However, it struggles with overfitting and limited generalization. While effective at identifying key features, it lacks robustness in handling complex feature relationships or noisy data. In contrast, the random forest demonstrated better accuracy by mitigating overfitting through ensemble learning. Nonetheless, it faced challenges due to dataset imbalance, with an abundance of failure data points compared to successes. Increasing the number of trees in the random forest did not improve accuracy, highlighting class imbalance and feature selection as key limiting factors. The neural network achieved the highest overall accuracy by capturing complex, non-linear relationships. However, it struggled to predict true positives effectively, likely due to the dataset imbalance. Additionally, the neural network demands significant computational resources and requires meticulous hyperparameter tuning for optimal performance. Future improvements could include employing feature selection techniques or utilizing Synthetic Minority Oversampling Technique (SMOTE) to augment the dataset, reducing overfitting in models like the random forest and neural network. Exploring newer datasets with more detailed and relevant features may also address the class imbalance issue, enabling a deeper understanding of the factors contributing to successful outcomes.

References

[1]K. Pelechrinis and E. Papalexakis, “The Anatomy of American Football: Evidence from 7 Years of NFL Game Data,” PLOS ONE, vol. 11, no. 12, p. e0168716, Dec. 2016, doi: https://doi.org/10.1371/journal.pone.0168716.

[2] X. Guo, “Neural Network Models for Predicting NFL Play Outcomes.” Available: https://cs230.stanford.edu/projects_spring_2020/reports/38964602.pdf

[3] Joash Fernandes, Craig, et al. “Predicting Plays in the National Football League.” Journal of Sports Analytics, vol. 6, no. 1, 1 Jan. 2020, pp. 35–43, content.iospress.com/articles/journal-of-sports-analytics/jsa190348, https://doi.org/10.3233/JSA-190348. Accessed 29 Sept. 2024.

Gantt Chart

linked here

or url here:

`https://gtvault.sharepoint.com/:x:/s/Fall2024MLGroup/EfY4R2bddEFBpLBHulTtZOoBZAyw6e7NpzgmQ1JhD3fz-g?e=u6JpgE`

Contributions

Name	Contribution

Vibha	Methods, Results & Discussion, Model 1, Forward feature Selection, Final Video
Brandon	Data Cleaning and Processing, Gantt Chart, Discussion
Neja	Methods, Results & Discussion, Proposal Video, Streamlit, Model 1, Forward feature Selection
Aiden	Data Cleaning and Processing, Results & Discussion
Alex	Model 2, Model 3, Streamlit, Data Processing, Results & Discussion