# Fantasy Football Model Project

ADAN 8888

Dr. Savas

8 December 2024

## Introduction

This report outlines the development of a machine learning model designed to predict fantasy football points for the 2023 NFL season using historical performance data from 2019-2022. Fantasy football, a popular game where participants draft real NFL players to form teams and score points based on individual player metrics, benefits from accurate data-driven predictions to enhance draft strategies. Following the Model Development Life Cycle (MDLC), the project encompassed defining the problem, collecting and preprocessing data, selecting and training models, and evaluating and deploying the final solution. Key steps included addressing missing data, engineering features to boost predictive power, and assessing models using metrics like Root Mean Squared Error (RMSE) and $R^2$.

The development process involved evaluating multiple machine learning algorithms, from linear regression to advanced models like XGBoost, with hyperparameter tuning to optimize performance. The final model, deployed in a cloud environment, provides fantasy football managers with actionable insights to make informed draft decisions. This report provides a detailed account of each phase, explaining methodologies, decision rationales, and outcomes, offering a comprehensive view of the challenges addressed and solutions implemented.

## Step 1: Problem Definition and Requirements

The problem addressed in this project is predicting fantasy football points for the 2023 NFL season using historical data from the 2019-2022 seasons. The primary goal is to create an accurate machine learning model that forecasts player performance metrics such as rushing yards, receptions, touchdowns, and passing yards. These predictions are designed to assist fantasy football managers in making informed decisions during drafts and throughout the season, ultimately improving their competitive edge. By focusing on key contributors to fantasy scoring, such as yards from scrimmage and total touchdowns, the project aims to minimize prediction errors while maximizing the model's reliability.

The dataset includes the top 200 players from each year, reflecting the structure of a typical 12-person fantasy league. Compiled into a comprehensive "master" workbook, the data encompasses variables like player names, team affiliations, and performance metrics such as rushing attempts, receptions, fumbles, and interceptions. This extensive dataset provides a robust foundation for model development, ensuring the inclusion of critical features that influence fantasy scoring. Data preprocessing involved addressing missing values, creating engineered features, and ensuring consistency across the dataset to enhance the model's predictive power.

Machine learning techniques were employed to analyze patterns in historical player performance and predict future fantasy points. Various algorithms, including linear regression and advanced models like XGBoost, were tested and optimized through hyperparameter tuning. Evaluation metrics such as Root Mean Squared Error (RMSE) and $R^2$ guided the selection of the best-performing model. The resulting predictions empower fantasy managers with data-driven insights, enabling them to draft undervalued players, optimize lineups, and improve trade decisions. By leveraging machine learning, the project offers a more objective and detailed approach compared to traditional analysis.

Beyond individual benefits, the project has broader implications for the fantasy football ecosystem. Accurate predictive models enhance content quality for analysts and media platforms, improve decision-making tools for participants, and open opportunities for innovation in daily fantasy sports and sports betting markets. As a $7.22 billion industry with millions of players, even incremental improvements in prediction accuracy translate into significant economic and user engagement gains. By providing accessible and actionable insights, this project supports the continued growth and accessibility of fantasy football for casual and competitive players alike.

**Step 2: Data Collection and Understanding**

The dataset for this project was sourced from publicly available data on Football Reference, specifically the scrimmage statistics pages for each year. Data was collected for the top 200 players across the 2019–2022 seasons to ensure a representative sample of players relevant to fantasy football. This dataset reflects key metrics tied directly to fantasy scoring, such as rushing yards, receiving yards, total touchdowns, and fantasy points. Given that fantasy football league rosters often consist of 12 teams with 16 players each, this sample size provides sufficient depth for analysis while maintaining relevance to the league format.

To prepare the dataset for analysis, four separate yearly CSV files were combined into a single master file. This consolidation allowed for seamless cross-year analysis and ensured that the data was consistent across seasons. During this process, specific attention was given to standardizing player names, as inconsistencies in formatting, such as special characters or differing spellings, could lead to duplicate entries. These duplicates were resolved by removing special characters and ensuring uniform naming conventions.

Once the data was merged, the first step was to clean and standardize the dataset. One major issue was the presence of missing values, particularly in the "Position" column, which is essential for distinguishing how players generate their fantasy points. Missing positions were filled using official data from Football Reference, ensuring that every player had a properly assigned position. For other missing values, imputation techniques were used. Categorical data, such as position, was imputed using the mode, while numerical data, such as games played, was imputed with the mean.

Feature engineering played a key role in enhancing the dataset's predictive power. Two key new features were created: "Total Yards from Scrimmage," a combination of rushing and receiving yards, and "Touchdown Efficiency," calculated as touchdowns per game. These features were chosen because they offer a more holistic view of player contributions and are highly correlated with fantasy football performance. Total Yards from Scrimmage provides a single metric that encapsulates a player's overall yardage production, while Touchdown Efficiency highlights a player's ability to convert opportunities into points.

Feature selection focused on identifying variables most relevant to predicting fantasy points. Metrics such as total yards, touchdowns, rushing attempts, and receiving targets were retained due to their direct impact on fantasy scoring. Irrelevant or redundant features were removed to streamline the dataset, avoiding issues such as multicollinearity that could affect model performance. By focusing on these impactful features, the dataset became more efficient and ready for machine learning modeling.

The exploratory data analysis (EDA) phase revealed important trends and correlations within the data. Strong relationships were observed between Total Yards from Scrimmage and Fantasy Points, as well as between Touchdowns and Fantasy Points. Visualizations such as scatter plots and correlation heatmaps validated these patterns, confirming the relevance of the engineered and selected features. EDA also uncovered positional differences in scoring patterns. For example, quarterbacks predominantly score through passing yards and touchdowns, while running backs rely more on rushing statistics. Understanding these distinctions was vital for guiding the modeling approach.

In summary, data collection and understanding involved merging four years of data from Football Reference into a single dataset, addressing missing values, creating new features, and selecting the most relevant variables. This foundational work ensured that the dataset was well-prepared for subsequent modeling efforts.

**Step 3: Data Preparation and Preprocessing**

Data preparation and preprocessing transformed the cleaned and consolidated dataset into a format suitable for machine learning. The first step was to standardize numerical features such as yards from scrimmage and touchdowns. Standardization ensured that each feature had a mean of 0 and a standard deviation of 1, allowing the model to weigh all variables equally during training. This was particularly important given the varying scales of the original metrics.
Missing values were further addressed during preprocessing to ensure the dataset's integrity. For the "Position" column, which had already been partially cleaned during the data understanding phase, any remaining gaps were filled using mode imputation, ensuring consistency. For numerical data, the mean of each respective feature was used for imputation. These steps ensured that no rows were excluded due to missing values, maximizing the dataset's utility.

The dataset was then split into training and test sets, using an 80-20 split. The training set included data from the 2019–2022 seasons, while the test set consisted of data reserved for evaluating predictions on the 2023 season. This approach ensured that the model was trained on historical trends while being tested on unseen data, providing a realistic assessment of its predictive capabilities. Through these preprocessing steps, the dataset was finalized for modeling, with clean, standardized features and an optimal structure for training and evaluation. This preparation laid the groundwork for effective machine learning and accurate predictions of fantasy football performance.

**Step 4: Model Approach and Justification**

In this phase of the project, I selected two primary models—Random Forest and XGBoost—to predict fantasy football points. Both models were chosen for their capabilities in handling complex, high-dimensional datasets with non-linear relationships, which are inherent in the nature of fantasy football statistics.

The Random Forest model was selected due to its ensemble learning approach, which constructs multiple decision trees and averages their results. This allows Random Forest to manage complex datasets effectively and prevent overfitting through techniques like bagging. It is particularly suited for the diverse feature set of fantasy football data, such as Total Yards, Touchdown Efficiency, and Fantasy Points per Game. The Random Forest model can handle both continuous and categorical variables, making it a versatile choice. Additionally, its ability to assess variable importance helps understand the relationships within the dataset. One of the model's primary strengths is its resilience to overfitting, particularly when using a large number of estimators. However, tuning key hyperparameters, such as the number of estimators, maximum depth, and minimum samples for splitting a node, is necessary to avoid potential overfitting and achieve optimal performance.

For XGBoost, a powerful gradient boosting algorithm, I chose it based on its efficiency and ability to model complex relationships in the data. XGBoost builds decision trees sequentially, where each tree corrects the errors made by previous trees, thereby boosting its predictive power. This makes it particularly effective for tasks where feature interactions are important, like predicting fantasy football performance. The model also incorporates regularization techniques, like L1 and L2 penalties, which help prevent overfitting and improve generalization. XGBoost's inherent complexity arises from the need to fine-tune hyperparameters like the learning rate, max_depth, and subsample ratio. This can lead to improved performance, but also requires careful attention to avoid overfitting. Given the complexity of the fantasy football dataset, XGBoost was a fitting choice for capturing intricate patterns that other models might miss.

**Step 5: Hyperparameter Tuning, Model Performance, and Selection**

The hyperparameter tuning process played a critical role in optimizing the performance of both models. For Random Forest, I evaluated three key hyperparameters: the number of estimators, the maximum depth of the trees, and the minimum samples required to split a node. The number of estimators controls the ensemble size, impacting both prediction accuracy and computational cost. A higher number of trees generally improves generalization but increases the risk of overfitting, while fewer trees might underfit the model. The maximum depth defines how deep the trees can grow, and deeper trees capture more intricate patterns but can overfit. The minimum samples for splitting a node control the model's complexity by restricting how finely the trees split the data. By tuning these parameters using GridSearchCV, I was able to achieve a good balance between model complexity and accuracy. The tuned Random Forest model consistently produced the best results on both training and validation datasets.

For XGBoost, I focused on four hyperparameters: the learning rate, max_depth, alpha (L1 regularization), and lambda (L2 regularization). The learning rate controls the contribution of each tree to the final prediction, and tuning it helped prevent overfitting by ensuring that the model didn't learn too quickly. The max_depth parameter determines how deep the individual trees can grow, capturing more complex patterns. Regularization parameters like alpha and lambda help prevent overfitting by penalizing large coefficients. After testing these parameters across three variations, the model tuned with the learning rate and max_depth achieved the best results, with the lowest validation RMSE and highest $R^2$.

The performance metrics used to evaluate the models were Root Mean Squared Error (RMSE) and $R^2$ (coefficient of determination). RMSE measures the average magnitude of prediction errors, making it an ideal metric for regression tasks like predicting fantasy points, where smaller errors are crucial. $R^2$, on the other hand, quantifies the proportion of variance in the target variable that is explained by the model. A higher $R^2$ indicates that the model is better at capturing the underlying patterns in the data, which is important for selecting a model that generalizes well.

The results of the three model variations—base, GridSearchCV-tuned, and PCA-augmented Random Forest, along with the base and tuned XGBoost models—were evaluated on both training and validation datasets.

- Random Forest Results: The base model showed strong performance on the training dataset, with low RMSE and high $R^2$, but slightly overfit on the validation set. The GridSearchCV-tuned model improved performance on the validation set, reducing RMSE and maintaining a strong $R^2$. However, the addition of PCA in the third variation led to worse performance, likely due to dimensionality reduction that discarded useful features.
- XGBoost Results: The base XGBoost model performed reasonably well, but tuning the learning rate and max_depth (Variation 1) reduced the validation RMSE and increased $R^2$, indicating an improvement in model performance. The other variations, tuning regularization (Variation 2) and the subsample feature (Variation 3), had less significant effects, with Variation 1 consistently outperforming the others.

The final selection of the best model was based on the GridSearchCV-tuned Random Forest for Week 7 and XGBoost with the learning rate and max_depth tuned for Week 8. The Random Forest model showed a slight advantage in generalization after tuning, while XGBoost's sequential tree-building process proved to be more efficient in capturing complex interactions. Both models showed strong predictive capabilities, but the selection of the best model was based on which one achieved the lowest RMSE and highest R² on the validation set.

| Model | Training RMSE | Validation RMSE | Training R² | Validation R² |
|---|---|---|---|---|
| Base (Random Forest) | 18.49 | 30.64 | 0.995 | 0.989 |
| GridSearchCV (Random Forest) | 16.95 | 30.68 | 0.996 | 0.989 |
| PCA + GridSearchCV (Random Forest) | 30.30 | 159.27 | 0.987 | 0.702 |
| Base (XGBoost) | 3.58 | 4.01 | 0.776 | 0.711 |
| Variation 1 (XGBoost) | 3.29 | 3.85 | 0.822 | 0.739 |
| Variation 2 (XGBoost) | 3.44 | 3.90 | 0.800 | 0.725 |
| Variation 3 (XGBoost) | 3.35 | 3.88 | 0.812 | 0.732 |

The GridSearchCV-tuned Random Forest emerged as the best model for Week 7, and XGBoost with the learning rate and max_depth tuned emerged as the best for Week 8. These models provided the most accurate predictions while balancing model complexity and performance.

**Step 6: Model Refinement and Bias Mitigation**

**Bias Identification:** During the model development process, several potential biases were identified, particularly in relation to player positions. In fantasy football, different player positions, such as running backs (RBs), wide receivers (WRs), and quarterbacks (QBs), typically score at varying levels due to differences in scoring opportunities. For example, RBs and WRs often accumulate more points because they are involved in multiple scoring categories, such as rushing yards, receiving yards, touchdowns, and receptions. On the other hand, positions like offensive linemen or tight ends may score fewer points due to their more limited contribution to scoring plays. The initial model was prone to over-representing the impact of high-scoring positions like RBs and WRs, which created an imbalance in the predictions.

Furthermore, factors such as the number of games played and injuries could lead to further biases, with players who played more games having higher chances of accruing fantasy points. This resulted in an unintentional favoring of players with more opportunities to score, rather than those who were more efficient or consistent.

**Bias Mitigation:** To address these biases, several mitigation strategies were implemented. First, player statistics were normalized per game played to account for players who missed games or had shorter seasons. By doing this, we ensured that players who played fewer games were not unfairly penalized for their lower total fantasy points. This adjustment allowed the model to focus on efficiency rather than simply the volume of games played.

Additionally, the feature selection process was revisited. Initially, the model included positional categories that could inadvertently amplify position-based biases. By carefully selecting features that were more closely tied to player performance and removing overly positional data, we ensured that the model gave fairer consideration to all player types. For example, we prioritized metrics like total yards, touchdowns, and fantasy points per game over positional classifications, which helped reduce the impact of positions that naturally accumulate more fantasy points.

**Model Improvement:** After the identification and mitigation of biases, the model was retrained using the updated features and normalized data. The retrained model exhibited a more balanced distribution of predictions across player positions, with no single position disproportionately influencing the outcomes. While the accuracy of the model remained high, the fairness of its predictions was significantly improved. This refinement process ensured that the model's predictions were not only reliable but also equitable across various player types, ensuring that fantasy football managers would have a fairer basis for making decisions in their drafts.

**Step 7: Deployment and Monitoring**

**Model Deployment:** Once the model was finalized and its performance validated, it was saved using Python's pickle library. This allowed for easy serialization of the trained model, enabling straightforward saving and loading for future use. The serialized model was then uploaded to my GitHub repository, making it accessible for others to use, collaborate on, or for personal use in the future. Hosting the model on GitHub ensures that it is version-controlled and easily reproducible for anyone interested in downloading and running the model locally.

GitHub serves as a simple and effective solution for deploying the model in the context of a school project. It allows other users (such as classmates, instructors, or future contributors) to download the necessary files, run the model on their own machines, and examine the model's performance using the code and dataset provided.

**Deployment Mode:** Since this project was intended to predict fantasy football points, batch inference was chosen as the deployment method. Batch inference allows the model to process predictions for a group of players at once, which is especially useful when reviewing many players' rankings at once during a draft. Fantasy football managers typically review large numbers of players at once, and the batch mode provides a way to process all players' predictions at once efficiently.

This method is well-suited for the scale of the project, where real-time processing of predictions isn't necessary. By selecting this approach, users can download the model from GitHub, process predictions in batches, and then use the outputs to assist with fantasy football decisions.

**Monitoring and Performance Tracking:** While the deployment on GitHub doesn't support automatic real-time monitoring like cloud platforms, performance tracking is still essential. I tracked the model's performance on the training and validation data during the model development phase, using metrics such as Root Mean Square Error (RMSE) and business-specific metrics (e.g., accuracy of top 10 player predictions). These metrics were included in the documentation on GitHub to provide context for the model's effectiveness.

In the future, for a more advanced deployment, drift detection could be set up using additional tools or periodic re-evaluations of the model's performance, especially as new player data becomes available. However, for the scope of this project, performance tracking will be manually conducted by revisiting the model with new data and analyzing its predictions against actual fantasy points.

**Model Maintenance:** To maintain the relevance of the model after deployment, updates will be necessary as new player data becomes available each NFL season. The model will be regularly updated to reflect changes in player performance and to incorporate new features if required. Additionally, regular retraining of the model with post-2023 season data will help to ensure that it continues to provide accurate and timely predictions for fantasy football managers.

In terms of model maintenance, the project repository on GitHub will be updated with new versions of the model as updates and improvements are made. Future users will be able to access the latest version of the model, ensuring that they are working with the most current version of the predictions.