Fantasy Football Data for Machine Learning Prediction Algorithms

Github: georgetown-analytics/Cloudy-with-a-Chance-of-Football

Georgetown University School of Continuing Studies
Cohort 23 — Capstone Project

Authors: Griffin Taub, Yaphet Tewahade, Ermina Mujan, Aidan O'Connor
Project Coordinator: Aidan O'Connor
Capstone Advisor: Karen Belita

June 26, 2021— Capstone Project in Partial Fulfilment of the Requirements for the Certificate in Data Science

Table of Contents

1. Abstract

In fantasy football, each player has a projected score for their upcoming game. It is not uncommon for a player to have a high projected score only to produce a score below projection. On the flip side, it is not uncommon for a player to have a low projected score only to produce a score above projection. We are pursuing this project to see what influences a player's actual output score the most so that we can make data-based decisions when building a team or starting lineup week to week during the NFL season. We plan on using 2019 and 2020 NFL season data for this project.

In this model our dependent/target variable is whether the player scored above or below what was projected. We used a classification model with an 80/20 train/test split method to determine whether a player will perform above or below the projected fantasy points. This model is designed to help fantasy managers decide who to draft, start, trade for, and pick up off the waiver wire. Our ideal end state is to deploy our trained model to a web application that requests basic user input and takes certain features into account to provide a prediction (which could be taken as a recommendation) about a player's performance in the coming week.

2. Problem Framing

Our project targets the following problems:

- On any given week, will an NFL player score above or below their projected fantasy score?
- What are the factors that have influenced performance the most so that we can best predict future performance?
- If we can determine the factors/features that have the most influence on player performance then we will be able to make an informed prediction on whether a player will score above or below their projected fantasy score on any given week during the regular NFL season.

3. Hypothesis Generation

This project hypothesizes that we can accurately predict whether an NFL player will overperform or underperform based on historical gameday data, as well as historical fantasy football predictions.

Furthermore, we assess these features to be most useful in successfully predicting outcomes:

- Injury status data, weather data, elevation and time displacement, Red zone stats, player age, and applicable previous weeks statistics

As of now, the usefulness of these features is theoretical and requires further analysis and experimentation to deem the validity of this hypothesis.

4. Motivation

Before technology, fantasy sports were played on paper by a select few sports buffs who were committed enough to use pencil and paper to record their team and stats throughout the season. Technology has transformed and catapulted fantasy sports to become a multi billion dollar industry. There are over 50 million fantasy sports players and a whopping 78% of those players play fantasy football, making it the most popular fantasy sport. Whether it is bragging rights or cold hard cash on the line, fantasy football is a big deal for millions and everyone is looking for an edge.

A fantasy football league typically consists of 8-12 competitors. There is a draft held before every NFL season where each fantasy competitor drafts/picks the best players available to them to create the best team possible. Once the draft is settled and the season starts, fantasy competitors go head to head every week with the scoring of those matchups dictated by the statistical in-game performance of the players they drafted and inserted into their weekly lineup. Every week the goal is to put together the best performing team possible that will accrue the most fantasy points. The challenge here is knowing who to insert into your lineup to ensure you are assembling the best possible team every week.

We decided to build a model that can determine the factors/features that have the most influence on player performance so that we will be able to make an informed prediction on whether a player will score above or below their projected fantasy score on any given week during the regular NFL season. With this predictive information from our model, we will be able to make an informed decision on who to start in our lineup. Using data science to evaluate the potential of a player for each week of the season is more likely to give you an edge rather than using your gut feeling or emotions.

5. Methodology

5.1. Data Selection and Scope

We were very thoughtful in the data selection process and knew that if we managed the scope well and found good reliable data, that would give us a solid foundation and set us up for success as we moved through the data science lifecycle. We set some business rules and limited the scope right off the bat before we started selecting data. As a group, we agreed to focus on only 2019 and 2020 NFL season data. Since there are many different scoring formats in fantasy football, we also agreed that any data we selected would have to fall into the PPR (Points Per Reception) format, that way we knew our scoring would be consistent across different data sets we selected. Since we are interested in individual player performance, we made sure that our data selection included every NFL player, every game they played, and all the stats they accrued during the 2019 and 2020 season. We brought in as many variables as possible that we thought might influence player performance knowing that later on down the road we could drop variables we didn't need. With our data selection, we knew there were other variables at play affecting player performance such as weather, elevation and injury status.

5.2. Data Ingestion and Extract, Transform, and Load

We completed all data pulls using Python scripts, bash scripts, and connecting to APIs. Initially, we were only using Python scripts (i.e. urllib and requests) to web scrape before understanding how to use APIs, and ran into ingestion challenges with websites, such as FantasyFootbalDataPros.com, where our IP addresses were flagged by the website when we attempted to ingest data too many times in a given period. Fortunately, we eventually learned how to incorporate APIs and were still able to scrape from websites, such as NFLWeather.com without any issues. The web scraping we performed on NFLWeather.com did not violate the terms of use. Once the data was ingested, we pushed the raw data to our GitHub repository for cleaning.[1]

5.3. Data Wrangling and Feature Selection

In order to build an accurate model, feature selection was an important consideration before modeling our data. Most of our features are player statistics, including passing, receiving, rushing, defense, and special teams. Our other features are player and game day data such as team, opponent, position, injury status, age of player, as well as game day weather data. After ingesting our data, we then standardized our column names based on the Python Enhancement Proposal 8 Style Guide and NFL player birthdays, and further cleaned our data using pandas, matplotlib, seaborn, numpy, and fuzzy wuzzy. In addition, we ensured that our final datasets had no NaN values using the missingno module. A feature that we included in this step of the data science pipeline that would be significant for our project down the road was creating a "performance_difference" column, whose values derived from subtracting the values from the "actual" and "predicted" columns; we would then use the difference from "performance_difference" to categorize a player's performance. The final step of data wrangling was shifting stats forward one week to predict current week performance (we did not shift weather, time, and location features). Once we standardized our columns and cleaned our data, we decided to use SQLite3 for our database to push our cleaned data, as it was the simplest solution that met our storage needs in terms of data size and accessibility.[2]

[1] Our ingestion techniques are described more fully in our Wiki.
[2] Our cleaning techniques are described more fully in our Wiki.

5.4. Statistical Analysis

Exploratory Data Analysis (EDA)

Exploratory analysis was used to ensure produced results were valid and relevant to our desired outcomes. It helped answer questions regarding confidence intervals, standard deviations, and categorical variables. After finishing analysis and gathering insights, we used the features for our machine learning models, and other forms of data visualization and modeling.

We used libraries such as pandas, numpy, matplotlib and seaborn to identify obvious errors, as well as better understand patterns within the data, detect outliers or anomalous events, and find interesting relationships among the variables.

Figure 5.4.1 illustrates the relationship between projected and actual fantasy points by position. The instances are colored by position and the markers are a star for overperformed and dot for underperformed. We also added a line showing the difference between the under and over performed clusters. We also called out exceptionally low and exceptionally high outliers

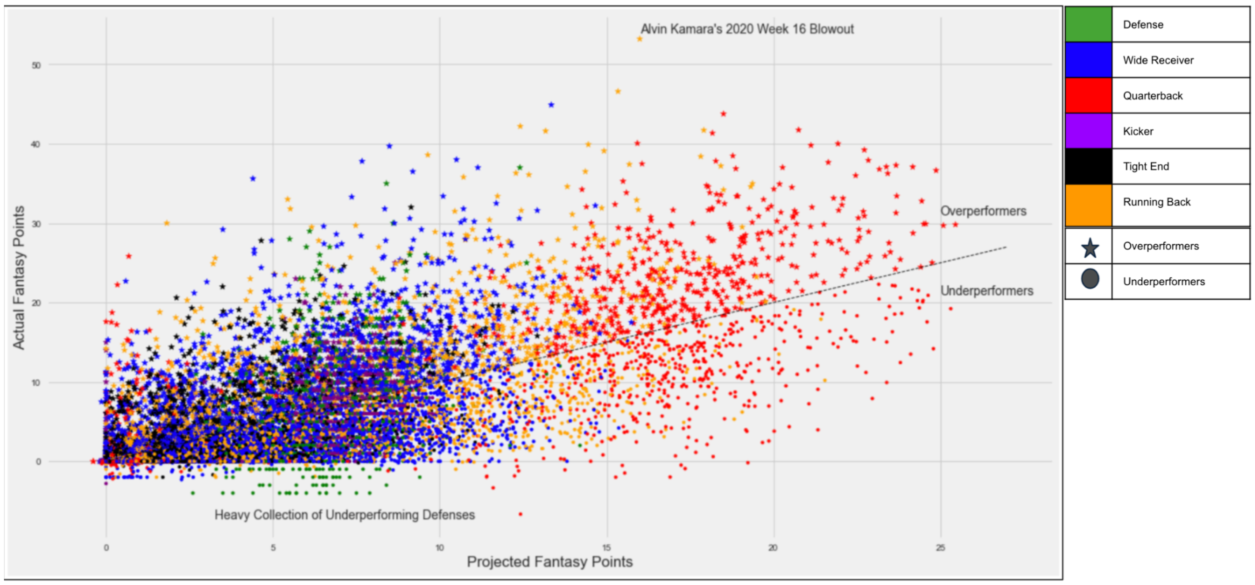

Figure 5.4.1: Projected vs Actual Fantasy Football Points

Figure 5.4.2. depicts Temperature and Humidity on the left, and Age and Precipitation on the right. This data is captured in dedicated scatter plots against the difference between projected and actual fantasy points.

During our EDA, we found that:

- Extreme temperatures negatively influence performance, and mid-70s play is great for good performance
- Precipitation above 20% greatly reduces high performance
- The oldest players tend to be kickers and quarterbacks, and the older the player, the less deviance from projected points
- Wide Receivers appear to have more outlying better performance between 40 and 80%, where quarterbacks are the opposite

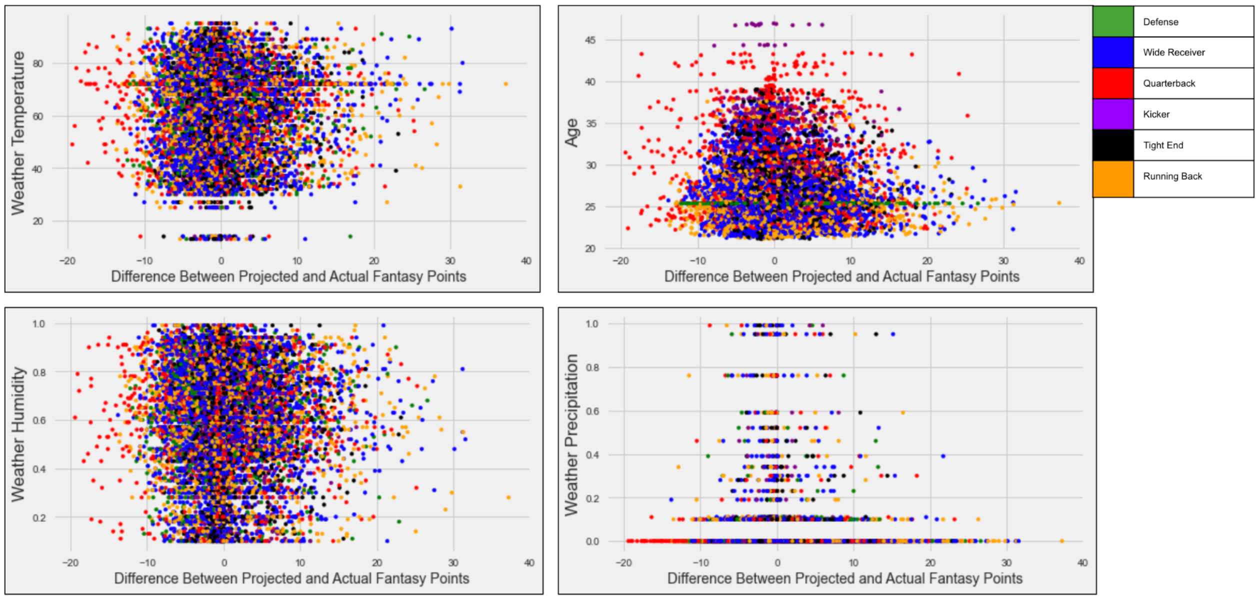

Figure 5.4.2: Temperature, Humidity, Precipitation and Age against Performance Scatter plots

Figure 5.4.3 shows a Histogram with the distribution of player age in our dataset. The average age of an NFL player is 26.5. The reason behind the high instance count is that we have 912 unique players in our dataset with a different age each week.

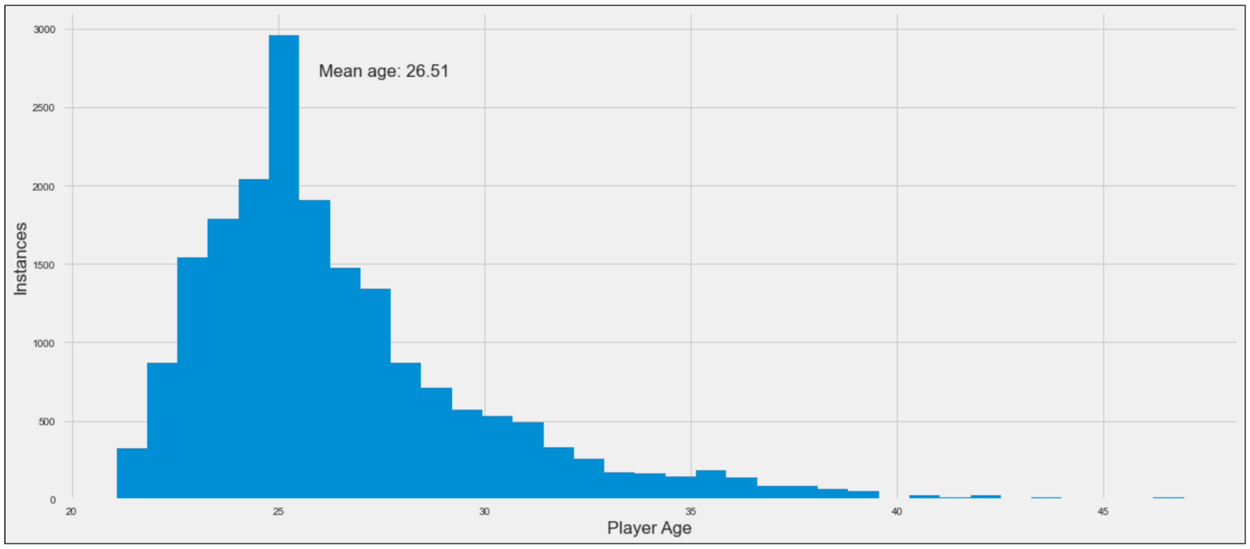

Figure 5.4.3: Distribution of Player Age

We learned that many machine learning algorithms perform better when features are on a relatively similar scale and/or close to normal distribution. Figure 5.4.4 demonstrates our non-binary columns following application of the MinMaxScaler, which we found to perform better for our dataset than the RobustScaler.

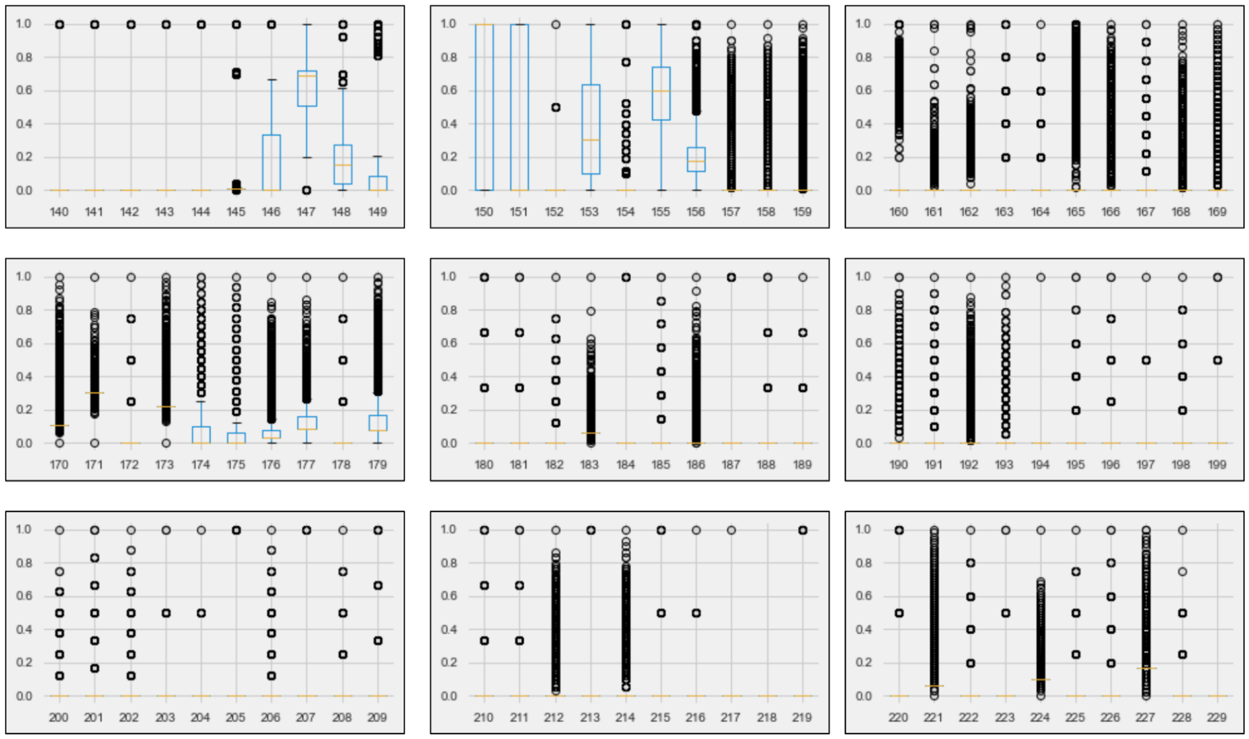
Figure 5.4.4: Non-Binary Columns After Application of Min-Max Scaler

5.5. Modeling and Application

After we completed our wrangling and EDA, we applied our data sets to several models to begin the data modeling and application step of the data pipeline. In doing this, we decided to do four different models, three of which were based on positions:

1. Global model
2. Offense model
3. Defense model
4. Wide Receiver model

The Global, Offense, and Wide Receiver models focused on optimizing for recall scores as there are more options for fantasy owners to choose from, while for the Defense model, we focused on precision as there are fewer options (only 32 teams to decide from, with 1 to 2 teams already being aligned to other managers); fantasy managers are more likely to worry about whether or not a prediction would be accurate, or a "true positive," when deciding on a defense (as we saw in figure 5.4.1, for example, defenses are typically projected to score very low, and there is a significant cluster that performed in the negatives, worse than projected).

To begin our modeling pipeline across all models, we first had to fit our test and train data, ensure that the models are trained on data from weeks 1 through 14, and tested on weeks 15 through 17 (satisfying the 80:20 train:test split), used LabelEncoder and OneHotEncoder to encode categorical values, scaled our data using MinMaxScaler, and imputed any null values with SimpleImputer. We then used a scoring and visualization function to produce scoring results through classification reports, which we used to evaluate our models and adjust model parameters.

We optimized our scores for our respective models by feeding our test and train data multiple times through the pipeline and scoring/visualization functions and tweaking model parameters. As we narrowed down the best scoring models, we finally used GridSearchCV to find which parameters would yield the best scores, and fed those parameters into our top scoring models to find the best responding models. Then, we produced one final classification report for each model, as well as a confusion matrix, to further evaluate our models' performance. Finally, we performed a time series split cross validation, using each successive week as new training data with the following week used as testing data. For the Global, Defense, and Wide Receiver models, the best scores were seen by the SGDClassifier. Refer to *Figures 1-3* below for the scoring results.

5.5.1. Global Model

The focus metric for the global model is recall for overperformance to cast a wider net in order to have more options available for fantasy managers.

We ran an initial set of models using default parameters, intending to maximize effectiveness and efficiency in high dimensional spaces, penalize incorrect minority class selection, find the most efficient predictors from ensemble methods, perform binary classification, compare similar instances, and optimize to control overfitting. After iteratively adjusting the parameters, dropping models, and running a grid search for the top two models (SVC and SGDClassifier). Grid search yielded results that showed SGDClassifier to be the most effective model for overperformance recall, with a score of 0.722. Finally, I ran the SGDClassifier through a 17-fold time series split cross validation (indices were weeks, so the splits were unequal), which showed the model's accuracy to be 0.73. The confusion matrix for this model showed that the model performed very well at identifying underperformance and detected true positives and false positives at nearly the same rate. Our focus on recall, or casting a wider net of players to offer fantasy managers more options for overperformance, shows that this model is a success.

```
F1 SCORE SGDClassifier: 0.589271417133707
```

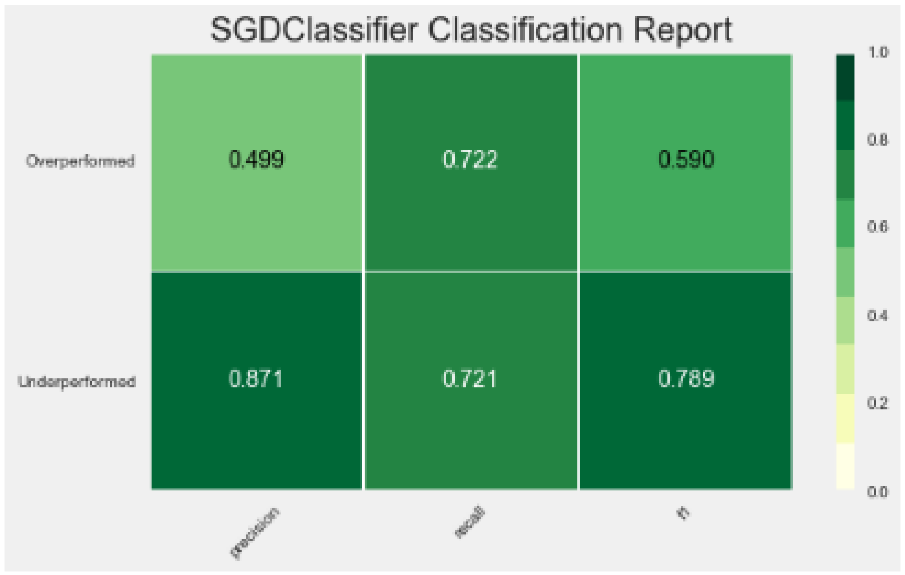

Figure 5.5.1a: Global Model Classification Report (SGDClassifier)

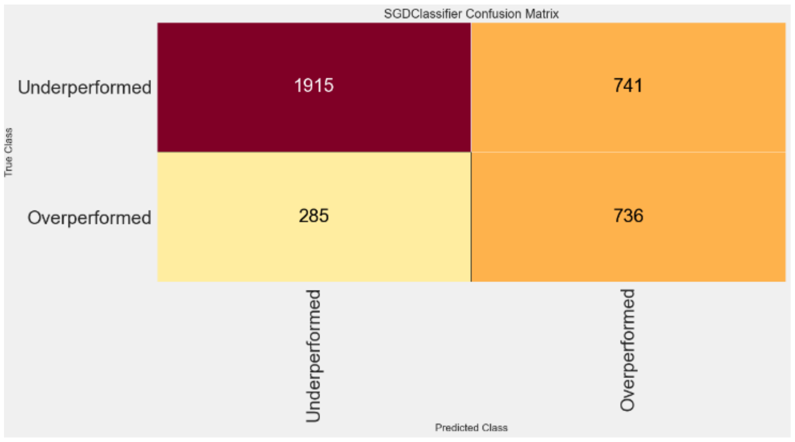

Figure 5.5.1b: Global Model Confusion Matrix (SGDClassifier)

5.5.2. Offensive Model

With the offensive model, only offense position and offensive player stats were included in analysis. The focus metric for the offensive model is recall for overperformance to cast a wider net in order to have more options available for fantasy managers. Running default parameters on several models -- SVC, NuSVC, and SGDClassifier were the top performing models with recall scores all above .5 in the overperformed class. These were the models I moved forward with in parameter tuning. After some basic tuning, SVC noticeably improved from 0.58 recall to 0.65 recall. Before considering this model final, it was run through grid search to get an output of the best combination of parameters. After applying the output of the grid search, recall improved slightly to 0.653. Finally, the SVC Model was run through a 17-fold time series split cross validation (indices were weeks, so the splits were unequal), which showed the model's accuracy to be 0.77. The confusion matrix for this model showed that the model performed very well at identifying underperformance and detected true positives and false positives at nearly the same rate. Our focus on recall, or casting a wider net of players to offer fantasy managers more options for overperformance, shows that this model is a success.

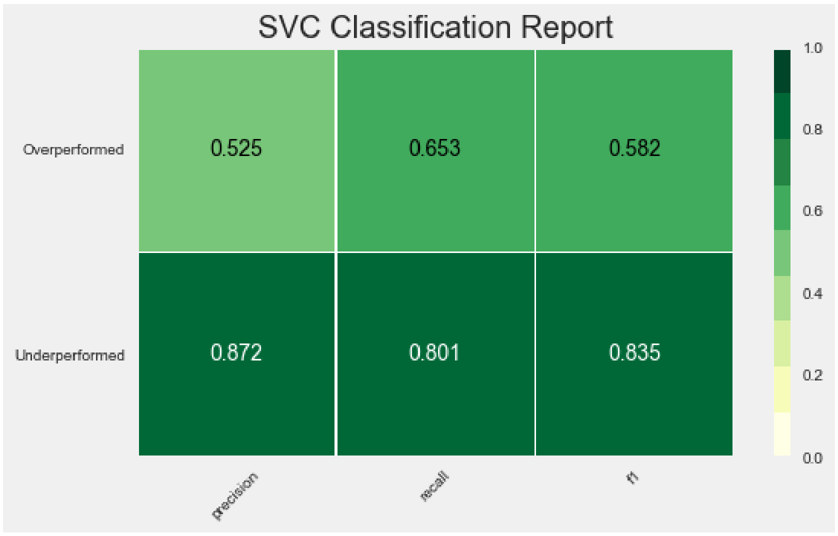

Figure 5.5.2a: Offensive Model Classification Report (SVC)

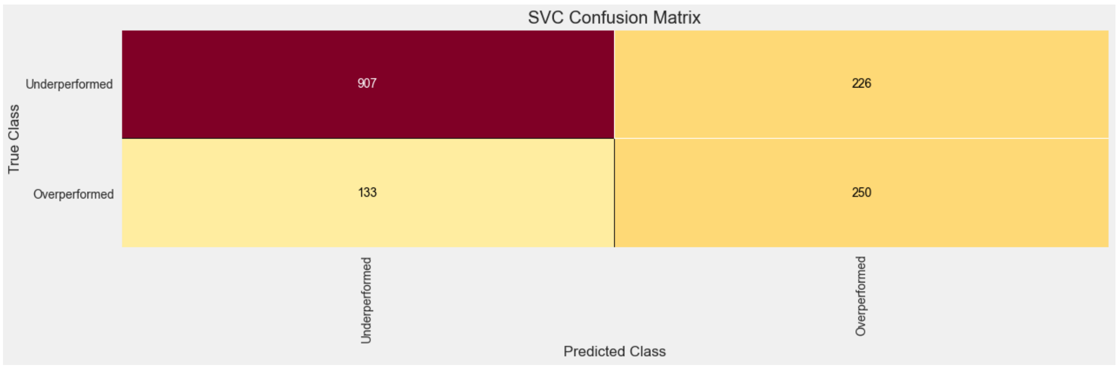

Figure 5.5.2b: Offensive Model Confusion Matrix (SVC)

5.5.3. Defense Model

For the Defense model, only defensive stats and team defense were fed into the models. The target score used for Defense was precision. Initially, after some very basic changes to the tuning, the top three performing models were SGD Classifier, Bagging Classifier and Logistic Regression. Out of the three, SGD Classifier saw the biggest improvement, as well as greatest precision score; pre-parameter tuning, SGD Classifier had a precision score of 0.641, but after using the parameters produced by gridsearch CV, the precision increased to 0.796. Using those same tuned parameters, the confusion matrix also showed that the model did very well, in terms of precision, in determining which teams would underperform; the confusion matrix showed lower precision scored for the "overperformed" class, which was likely due the class imbalance from the data showing that more teams underperformed..

```
F1 SCORE SGDClassifier: 0.44578313253012053
```

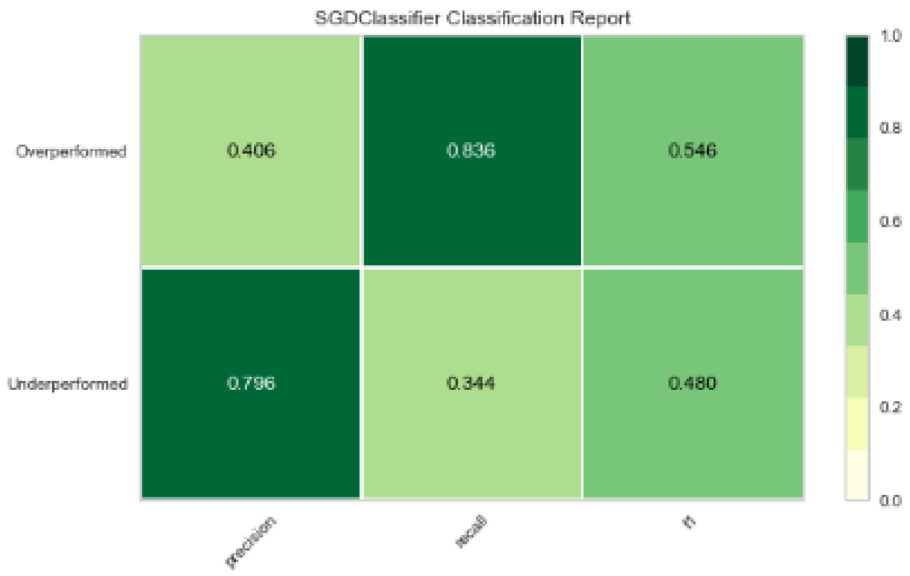

Figure 5.5.3a: Defense Model Classification Report (SGDClassifier)

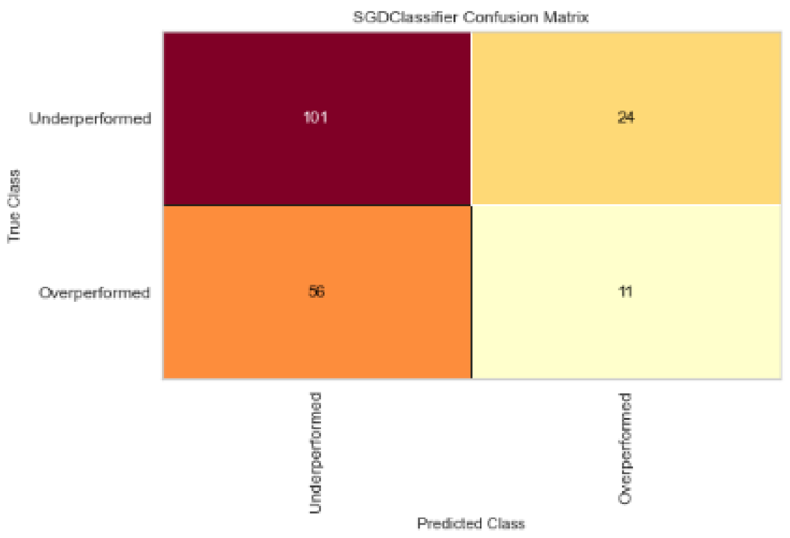

Figure 5.5.3b: Defense Model Confusion Matrix (SGDClassifier)

5.5.4. Wide Receiver Model

Just like in the global and offensive model, we are looking into recall for overperformance to cast a wider net in order to have more options available for fantasy managers.

After running the initial set of models with the default parameters, best recall scores for overperformance were shown in SVC and NuSVC models with the same score of 0.565, and SGDClassifier with the recall score of 0.661. With basic parameter tuning for all models, slight differences were shown in SVC and SGDClassifier. I chose 2 best performing models and ran it through a grid search. After applying the best parameters found, *SGDClassifier performed the best with a 0.845 recall score for overperformance.*

```
F1 SCORE SGDClassifier: 0.5244338498212158
```

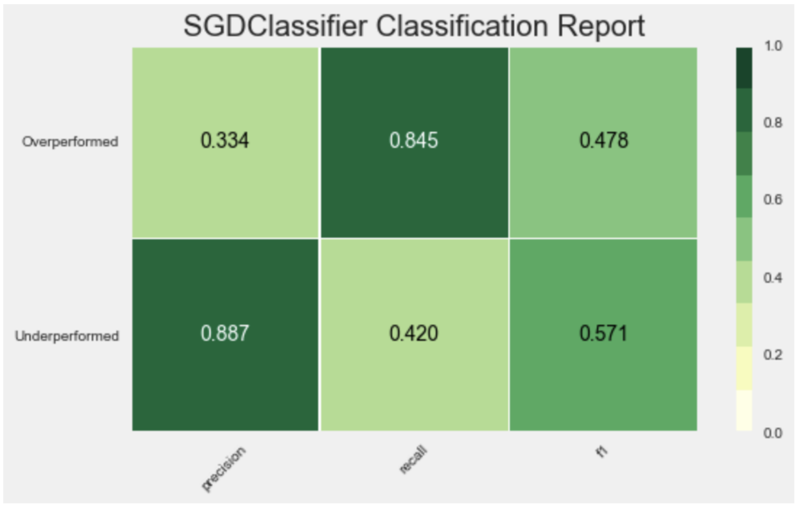

Figure 5.5.4a: Wide Receiver Model Classification Report (SGDClassifier)

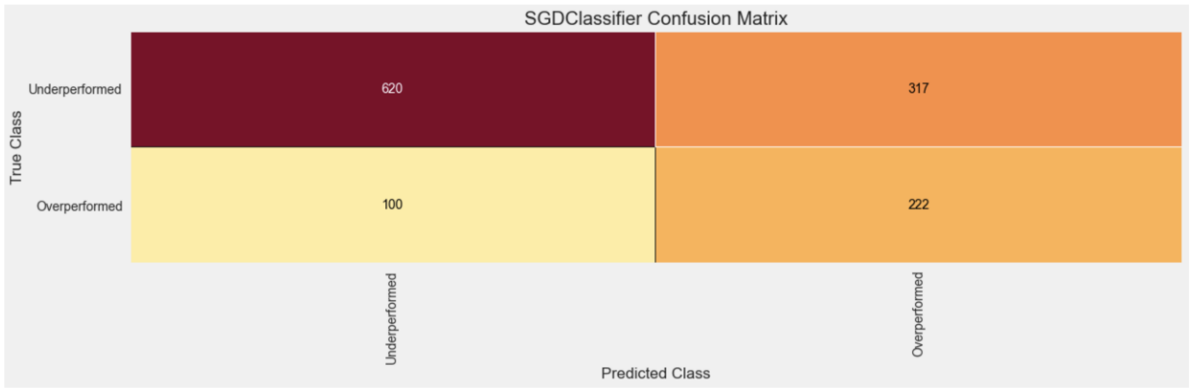

Figure 5.5.4b: Wide Receiver Model Confusion Matrix (SGDClassifier)

6. Implications on Future Work

Given more time to perform this project, we would have pursued the following areas of study, which we believe are promising areas of future study:

- Multi-class classification or logistic regression of over or underperformance to determine how much a player will deviate from projected fantasy points, and to what extent they will do so
- Analysis of individual defensive players instead of entire team defenses
- Use of play-by-play data to analyze and, ideally, predict team's offensive strategy, and determine and optimum strategy
- Application of this model's infrastructure to other sports (with the most applicable other professional sports being baseball and lacrosse, since they are typically played outdoors). This would require significant extra work for basketball and hockey, given the different characters of those leagues.

7. Conclusion

We decided to go with four different models (global, offense, defense and wide receivers) to gain more hands-on experience, and to be able to compare the differences between each model. Our data responded well to SVC and SGDClassifier models. More specifically SGDClassifier proved to be the most performant for the global, defense and wide receiver models, while SVC was performant for the offense model. We had a sufficient number of important features to produce our models, including player and game data, team, opponent, position, injury status, days since last game, hours and elevation displacement if away from home, age, and playing status. We also used weather data and red zone stats.

Keys to our success in building the four predictive models were organized team coordination, detailed plan of project milestones, frequent meetings, and live debugging.