Cloudy With A Chance of Football

Georgetown University School of Continuing Studies
Professional Certificate in Data Science
Cohort 23 (Spring 2021)

Ermina Mujan
Aidan O'Connor
Griffin Taub
Yaphet Tewahade

17 89

GEORGETOWN UNIVERSITY

Agenda

I. Introduction
II. Hypothesis
III. Methodology
- Understanding the Problem
- Identifying Data Requirements
- Data Collection and Ingestion
- Architecture
- Munging, Wrangling, and Feature Selection
- Statistical Analysis
- Modeling and Application

IV. Demonstration and Conclusion

I. Introduction

Introduction to Fantasy Football

- League: 8 - 12 person league (managers)
- Managers:
  - Competitively draft players at beginning of season
  - Set lineups weekly (typically QB, 2xWR, 2xRB, TE, DEF, K, Flex)
  - Select unaffiliated players off of waiver wire as needed
  - Trade with other managers
  - Win by real-life points produced by lineup players weekly
- Decision-making:
  - Subjective
  - Data-informed, not usually data-driven

Source: National Football League

II. Hypothesis

Hypothesis

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 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.

We hypothesize that we can accurately **predict whether an NFL player will perform better or worse than they are projected to perform based on environmental data and historical performance data**.

III. Methodology

Understanding the Problem

In order to predict a player's performance, we needed to find historical performance data for each player, which consisted of position-specific stats and predicted and actual full points per reception (full PPR) data. We also incorporated gameday weather data to be able to make more accurate, holistic predictions.

Methodology

Identifying Data Requirements

Target

Our target is a binary column that is based on whether a player performed above (1) or below (0) the amount of fantasy points projected by fantasydata.com

Features

Passing
PassingAttempts
PassingCompletions
PassingYards
PassingCompletionPercentage
PassingYardsPerAttempt
PassingYardsPerCompletion
PassingTouchdowns
PassingInterceptions
PassingRating
PassingLong
PassingSacks
PassingSackYards
PassesDefended
TwoPointConversionPasses

Defense and Special Teams
Fumbles
FumblesLost
SoloTackles
AssistedTackles
TacklesForLoss
Sacks
SackYards
QuarterbackHits
FumblesForced
FumblesRecovered
FumbleReturnTouchdowns
Interceptions
InterceptionReturnTouchdowns
Safeties
TouchdownsScored
FieldGoalsAttempted
FieldGoalsMade
ExtraPointsMade
TwoPointConversionRuns
ExtraPointsAttempted
FieldGoalsMade0to19
FieldGoalsMade20to29
FieldGoalsMade30to39
FieldGoalsMade40to49
FieldGoalsMade50Plus
PointsAllowedByDefenseSpecialTeams
BlockedKickReturnTouchdowns
PointsAllowed
SpecialTeamsTouchdowns
DefensiveTouchdowns
BlockedKicks
TwoPointConversionReturns
FieldGoalReturnTouchdowns
PuntReturns
PuntReturnYards
PuntReturnTouchdowns
KickReturns
KickReturnYards
KickReturnTouchdowns

Receiving
ReceivingTargets
Receptions
ReceivingYards
ReceivingYardsPerReception
ReceivingTouchdowns
ReceivingLong
TwoPointConversionReceptions

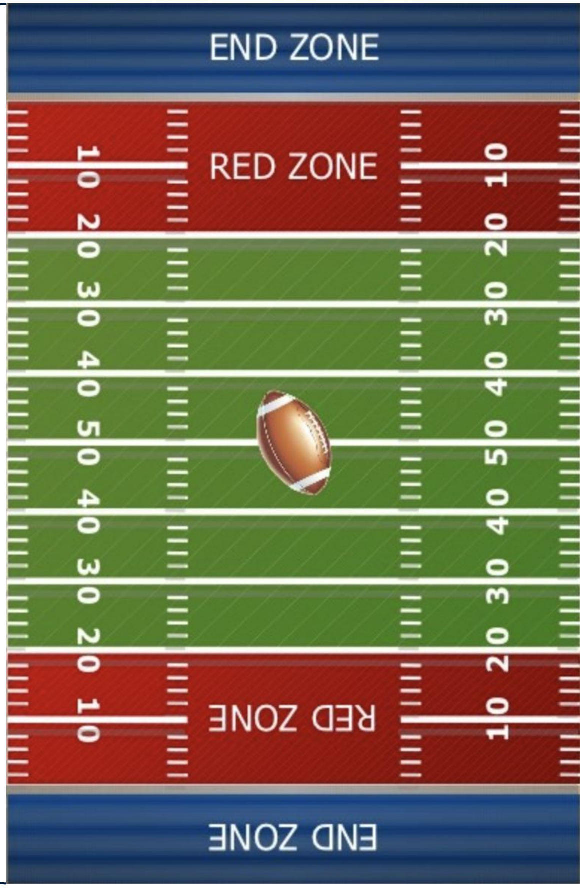
Rushing
RushingAttempts
RushingYards
RushingYardsPerAttempt
RushingTouchdowns
RushingLong

Features

Player and Game Data
PlayerID
Week
Team
Opponent
HomeOrAway
Position
PositionCategory
InjuryStatus
week_id
days_since_last_game
absolute_hours_displaced
elevation_displacement
age
Played
Started

Weather
weather_temperature
weather_wind_mph_number
weather_wind_direction
weather_cloud_cover
weather_precipitation
weather_humidity
weather_detail

Redzone
PassingYardsRZ
PassingTouchdownsRZ
PassingInterceptionsRZ
PassingYardsRZ
PassingTouchdownsRZ
PassingInterceptionsRZ
OpponentRZ
RushingYardsRZ
RushingTouchdownsRZ
ReceptionsRZ
ReceivingYardsRZ
ReceivingTouchdownsRZ
SacksRZ
InterceptionsRZ
FumblesForcedRZ
FumlbesRecoveredRZ



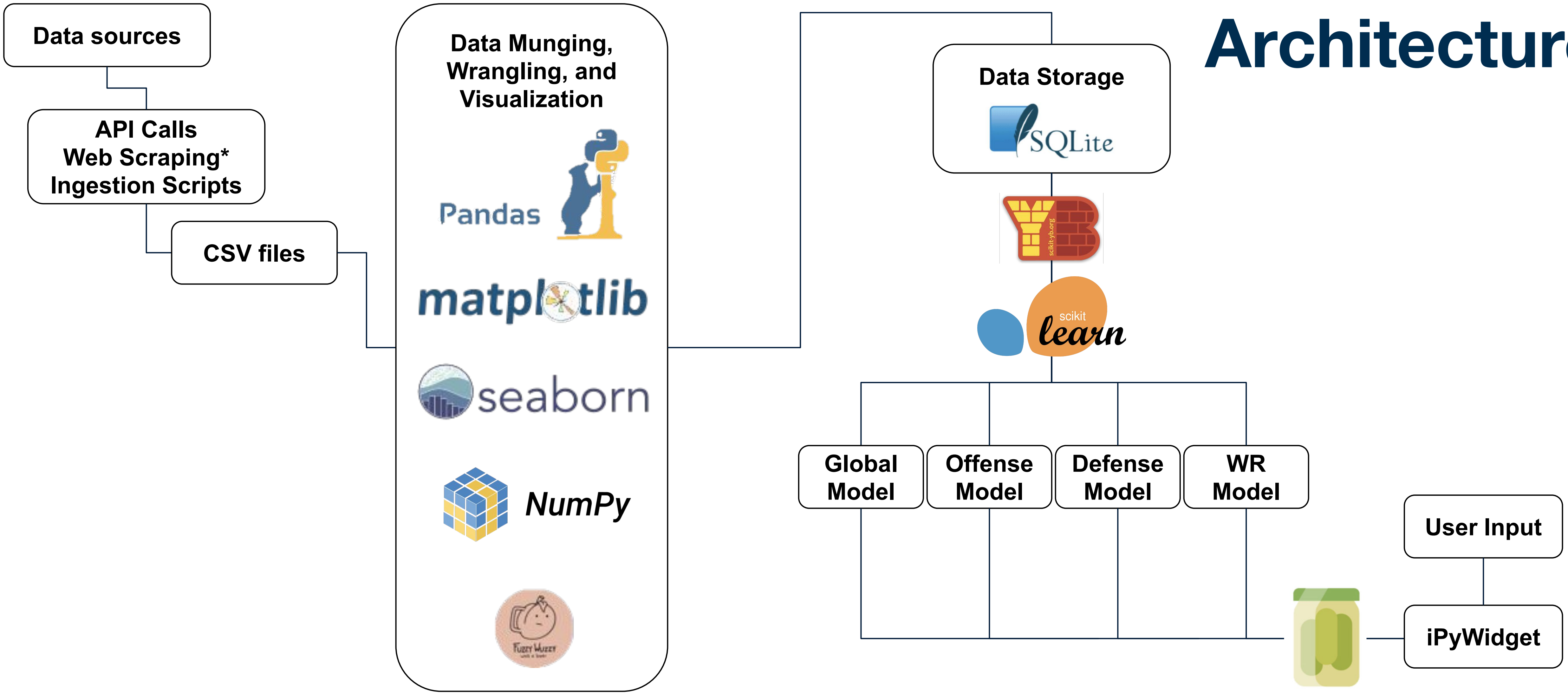
Data Collection and Ingestion

Kaggle.com	API Calls
FantasyData.com	API Calls
FantasyFootballDataPros.com	Python Ingestion Script
NFLWeather.com	Web Scraping

Methodology

Architecture

Architecture



Methodology

Munging, Wrangling, and Feature Selection

Munging, Wrangling, and Feature Selection

- De-selection of stats from non-global models
- Shifting stats ahead to predict next week performance

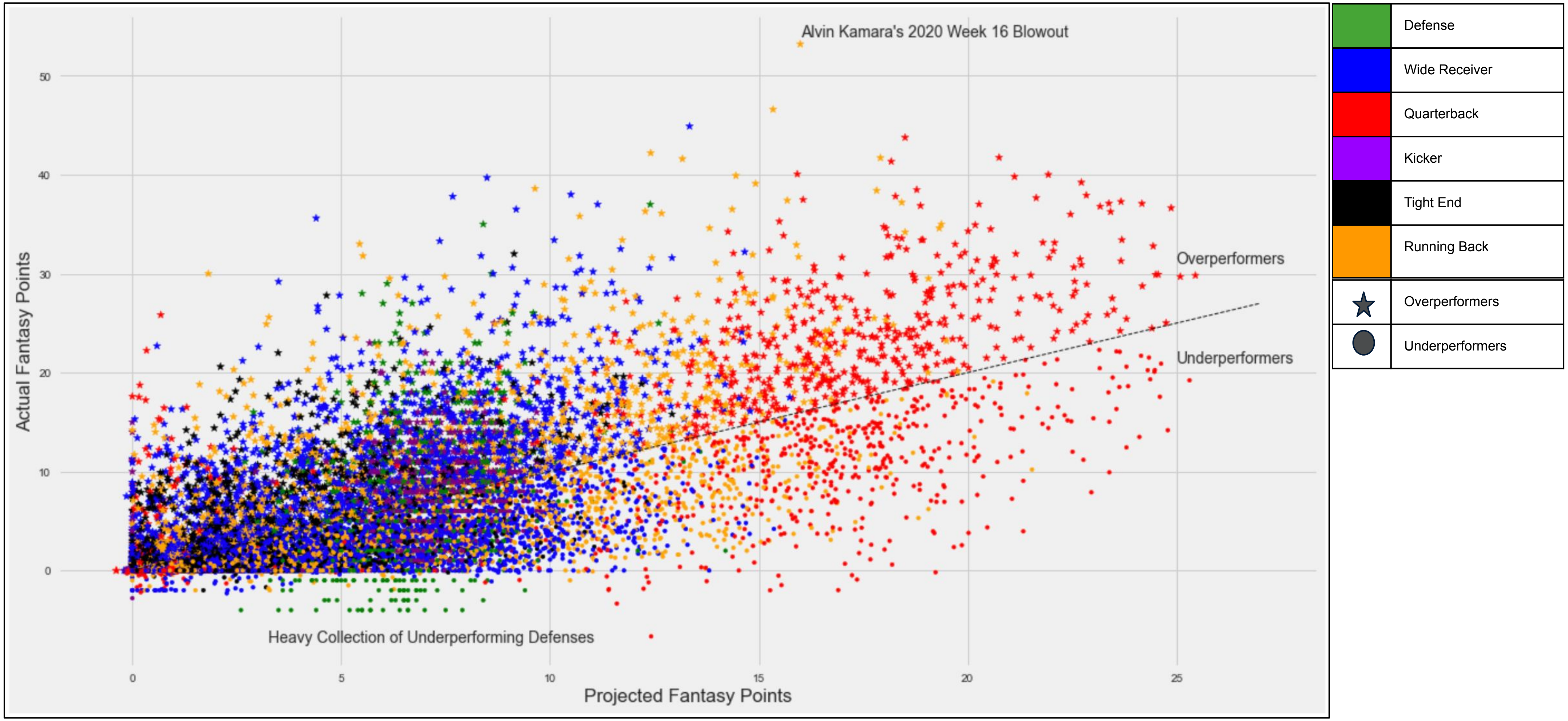
PlayerID	week_id	...	Stat1	Stat2	Stat3	Stat4	Stat5	Stat6	performance
12345	2019_1	...	1	15	2.2	0	6	8	1
12345	2019_2	...	3	8	1.5	2	9	3	0

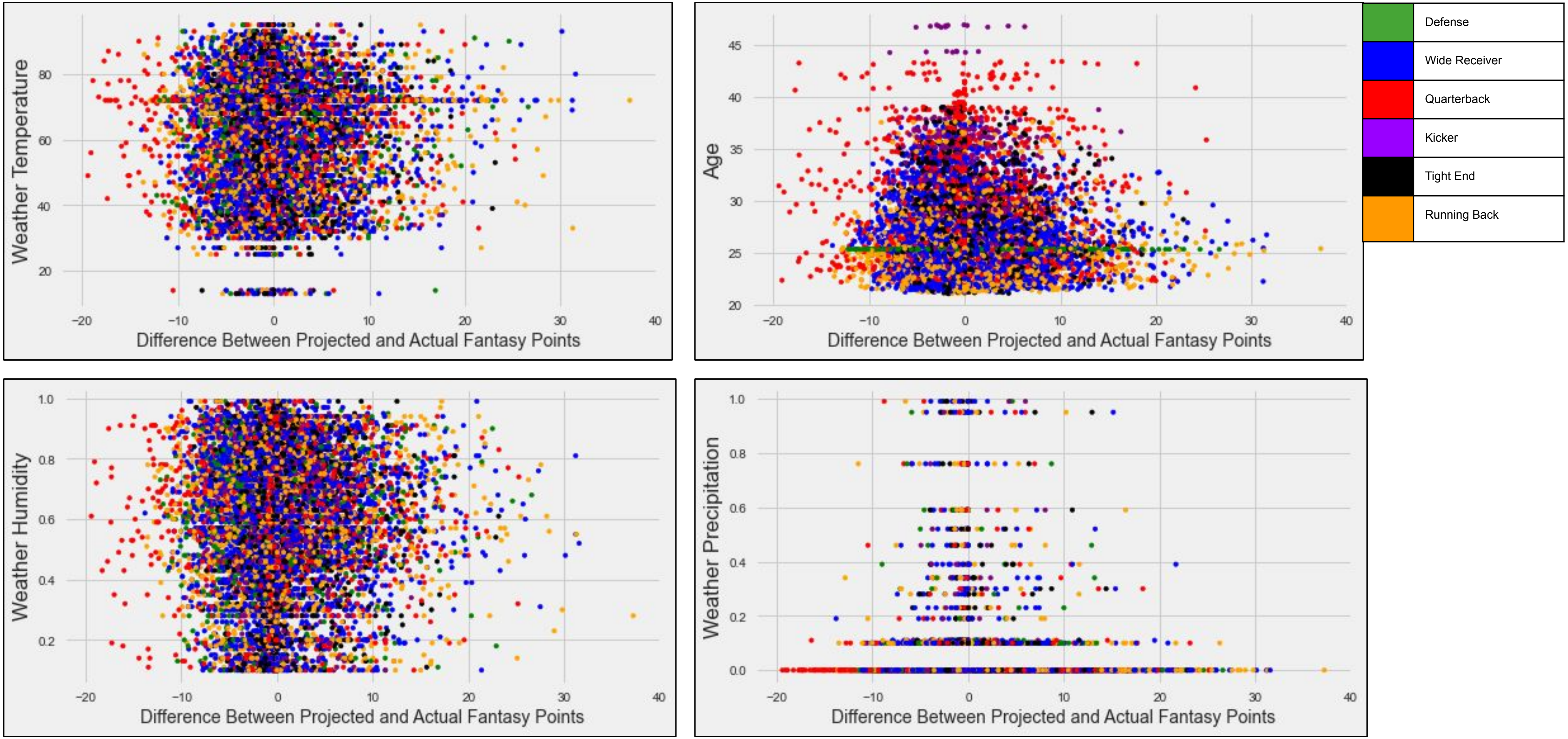
PlayerID	week_id	...	Stat1	Stat2	Stat3	Stat4	Stat5	Stat6	performance
12345	2019_1	...	0	0	0	0	0	0	1
12345	2019_2	...	1	15	2.2	0	6	8	0

Methodology

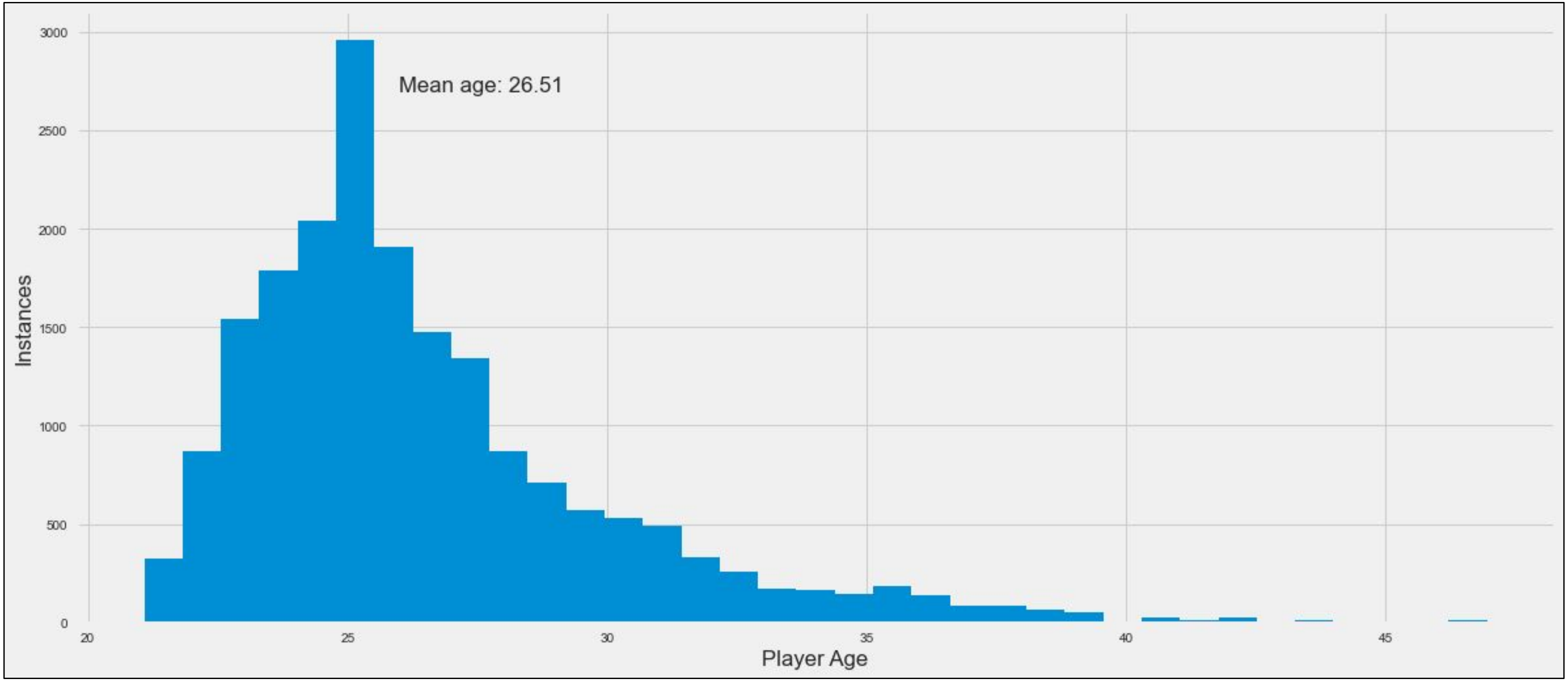
Statistical Analysis

EDA (Exploratory Data Analysis)

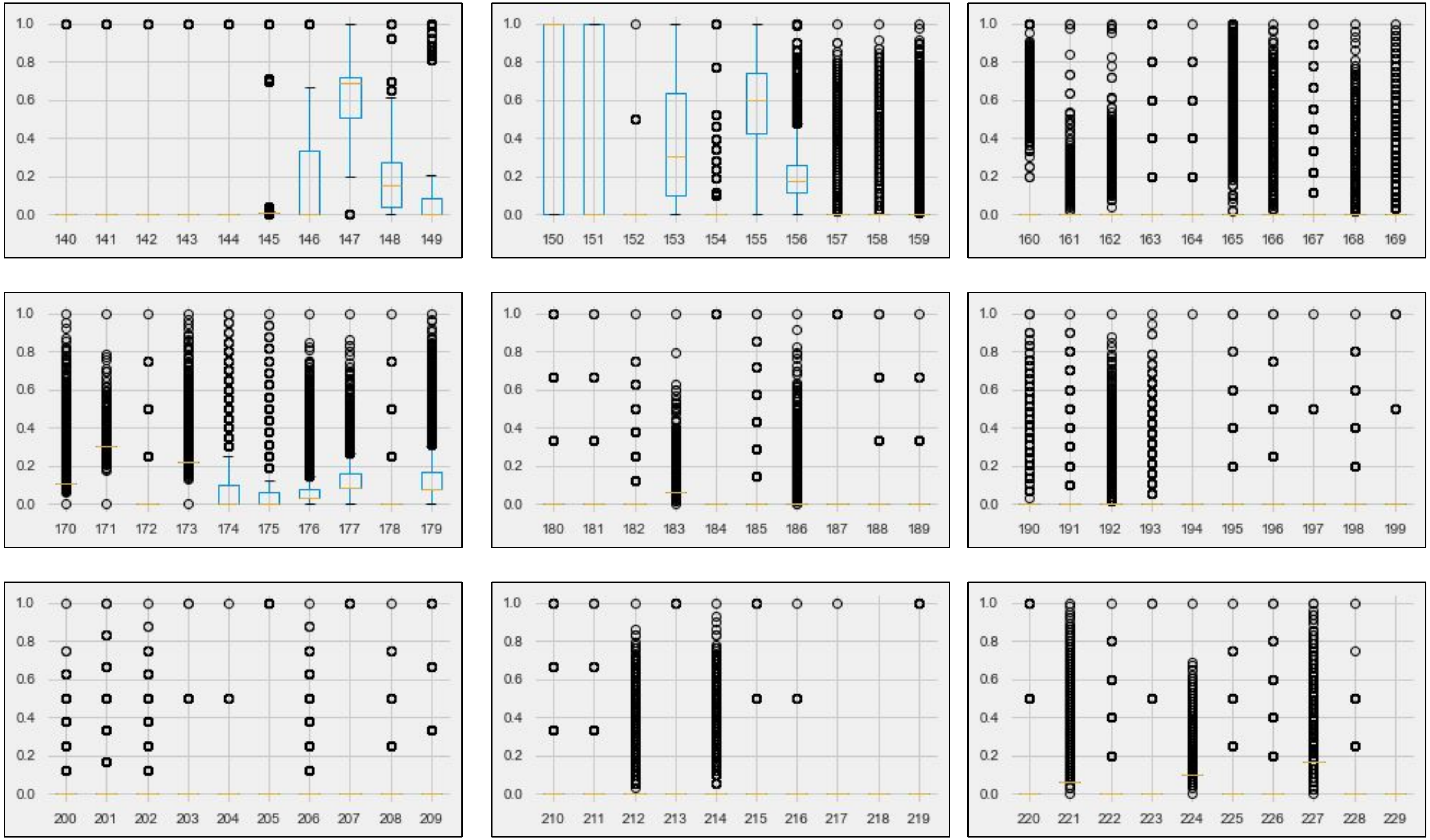




GEORGETOWN UNIVERSITY


Mean age: 26.51
Instances
3000
2500
2000
1500
1000
500
0
20
25
30
35
40
45
Player Age

Non-Binary Columns After Application of Min-Max Scaler



Methodology

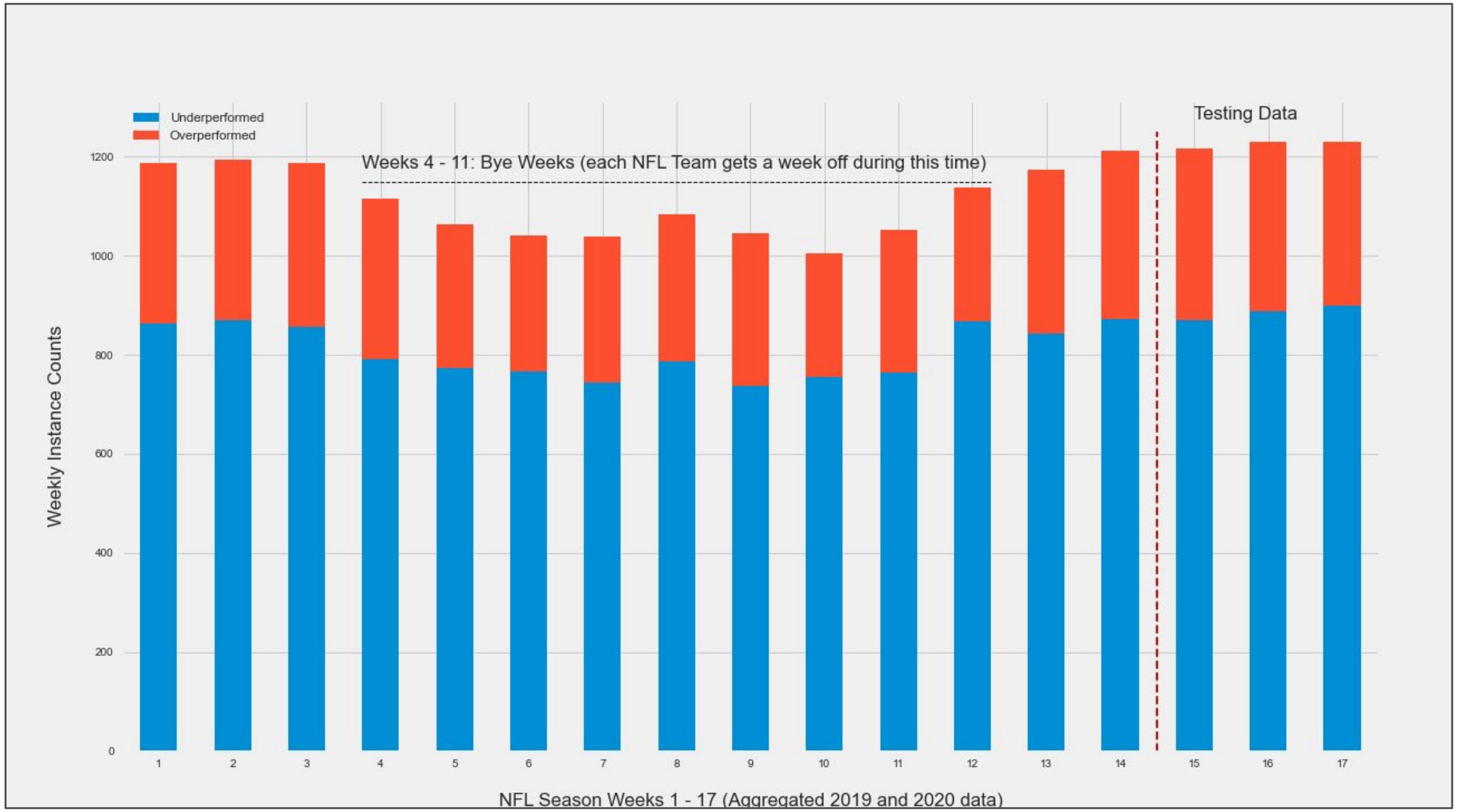
Modeling and Application

Modeling and Application

Initial Assumptions and Plan:

- Class imbalance, but consistent imbalance across weeks
  - ➢ Penalize mistakes on minority class selection
- Time-aware model
  - ➢ Train on weeks 1 - 14
  - ➢ Test on weeks 15 - 17
- Many instances, but variety of positions
  - ➢ Global model
  - ➢ Position-based models
- Desired more options from which to choose than fewer, more accurate options
  - ➢ Recall is the desired measure

** With the exception of the Defense Model*



Modeling and Application

Model	Reason for Selection
`SVC()`	Effective in high dimensional spaces as a binary classifier, and effectively handles unbalanced classes
`NuSVC()`	Similar to SVC, but includes a parameter to control the number of support vectors
`LinearSVC()`	Similar to SVC and scales better to large numbers of samples. Also has more flexibility in the choice of penalties
`SGDClassifier()`	Efficient model for evaluating many thousands of training data points
`KNeighborsClassifier()`	Unlike linear models (which try to draw distinctive lines between classes), classifies based on nearest neighbors in dataset
`LogisticRegression()`	A binary classifier that handles imbalanced classes well
`LogisticRegressionCV()`	Similar to LogisticRegression, but uses cross validation as well
`BaggingClassifier()`	Uses base estimators, then aggregates them, which may help find signal in the noise better than the others
`ExtraTreesClassifier()`	Controls over-fitting by fitting randomized decision trees on the data (which may help with the class imbalance)
`RandomForestClassifier()`	Unlike ExtraTreesClassifier, this chooses the optimal decision tree split

Modeling and Application: Global Model

Step 1:
Initial models with default parameters

```
models = [
    SVC(),
    NuSVC(),
    LinearSVC(),
    SGDClassifier(),
    KNeighborsClassifier(),
    LogisticRegression(),
    LogisticRegressionCV(),
    BaggingClassifier(),
    ExtraTreesClassifier(),
    RandomForestClassifier()
]
```

Step 2:
Same initial models with some parameters specified

```
modified_models = [
    SVC(gamma = 'auto'),
    NuSVC(gamma = 'auto'),
    LinearSVC(max_iter = 2000),
    SGDClassifier(max_iter = 100, tol = 1e-3),
    KNeighborsClassifier(n_neighbors = 10),
    LogisticRegression(solver = 'lbfgs'),
    LogisticRegressionCV(cv=3, max_iter=100),
    BaggingClassifier(n_estimators = 15),
    ExtraTreesClassifier(n_estimators = 300),
    RandomForestClassifier(n_estimators = 300)
]
```

Step 3a:
Models with below 0.50 recall for performance

```
other_models = [
    LinearSVC(max_iter = 6000),
    KNeighborsClassifier(n_neighbors = 10),
    LogisticRegression(solver='lbfgs', max_iter = 2000),
    LogisticRegressionCV(cv=3, max_iter=600),
    BaggingClassifier(n_estimators = 20),
    ExtraTreesClassifier(n_estimators = 600),
    RandomForestClassifier(n_estimators = 600)
]
```

19,215 instances
105 features

Modeling and Application: Global Model

Step 3c:
Models with above 0.50 recall for overperformance

```
good_models = [
    SVC(gamma='auto'),
    NuSVC(gamma='auto'),
    SGDClassifier(max_iter=100, tol=1e-3),
    BaggingClassifier(n_estimators = 40)
]
```

Step 3b:
“Wild Card” model added to good_models

```
good_models.append(BaggingClassifier(n_estimators = 20))
```

Step 4:
Final pre-GridSearch model selection

```
good_models = [
    SVC(
        gamma = 'auto',
        class_weight = 'balanced'
    )
    NuSVC(
        gamma = 'auto',
        class_weight = 'balanced'
    ),
    SGDClassifier(
        max_iter = 100,
        class_weight = 'balanced'
    ),
    BaggingClassifier(
        n_estimators = 40
    )
]
```

Modeling and Application: Global Model

Step 5a1:
Set SVC GridSearch parameters

```
param_grid_SVC = {
    'C' : [1, 10],
    'gamma' : ['scale','auto'],
    'kernel' : ['linear', 'rbf'],
    'class_weight' : ['balanced']
}
```

Step 5a2:
Run gridsearch on SVC

```
grid = GridSearchCV(
    SVC(),
    param_grid_SVC,
    refit = True,
    verbose = 2,
    n_jobs = -1
)
```

Step 5a3:
Run SVC with best parameters

```
{
    'C': 1,
    'class_weight': 'balanced',
    'gamma': 'scale',
    'kernel': 'rbf'
}
```

Step 5b1:
Set SGDClassifier GridSearch parameters

```
param_grid_SGDClassifier = {
    'loss': ['hinge','log'],
    'max_iter' : [100, 500],
    'penalty': ['l1','l2'],
    'n_jobs': [-1],
    'class_weight' : ['balanced']
}
```

Step 5b2:
Run gridsearch on SGDClassifier

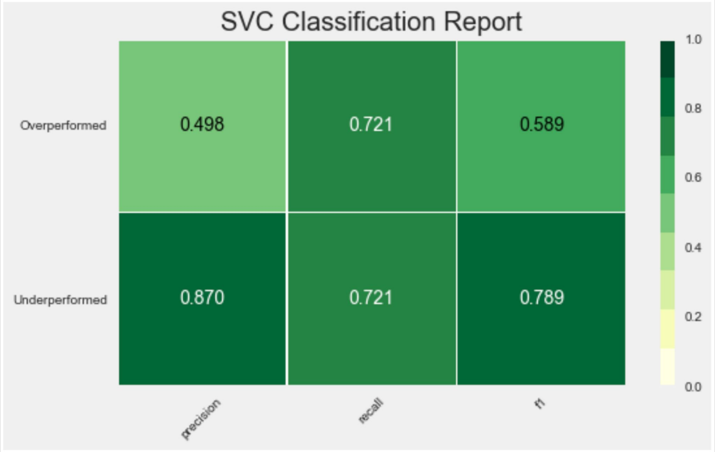
```
grid = GridSearchCV(
    SGDClassifier(),
    param_grid_SDGClassifier,
    refit = True,
    verbose = 2
)
```

Step 5b3:
Run SGDClassifier with best parameters

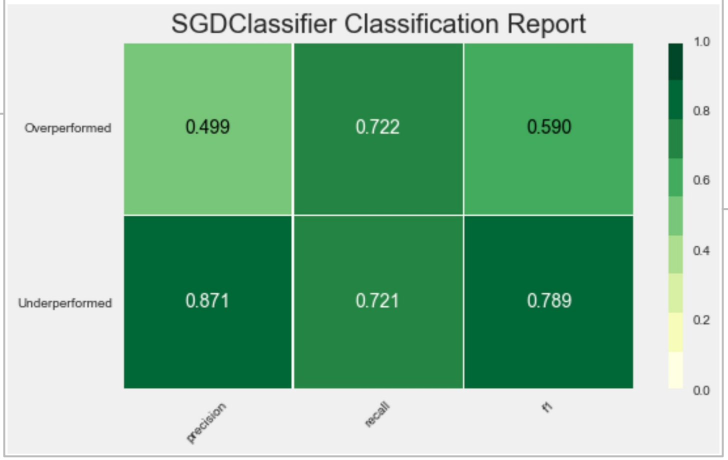
```
{
    'class_weight': 'balanced',
    'loss': 'hinge',
    'max_iter': 500,
    'n_jobs': -1,
    'penalty': 'l1'
}
```

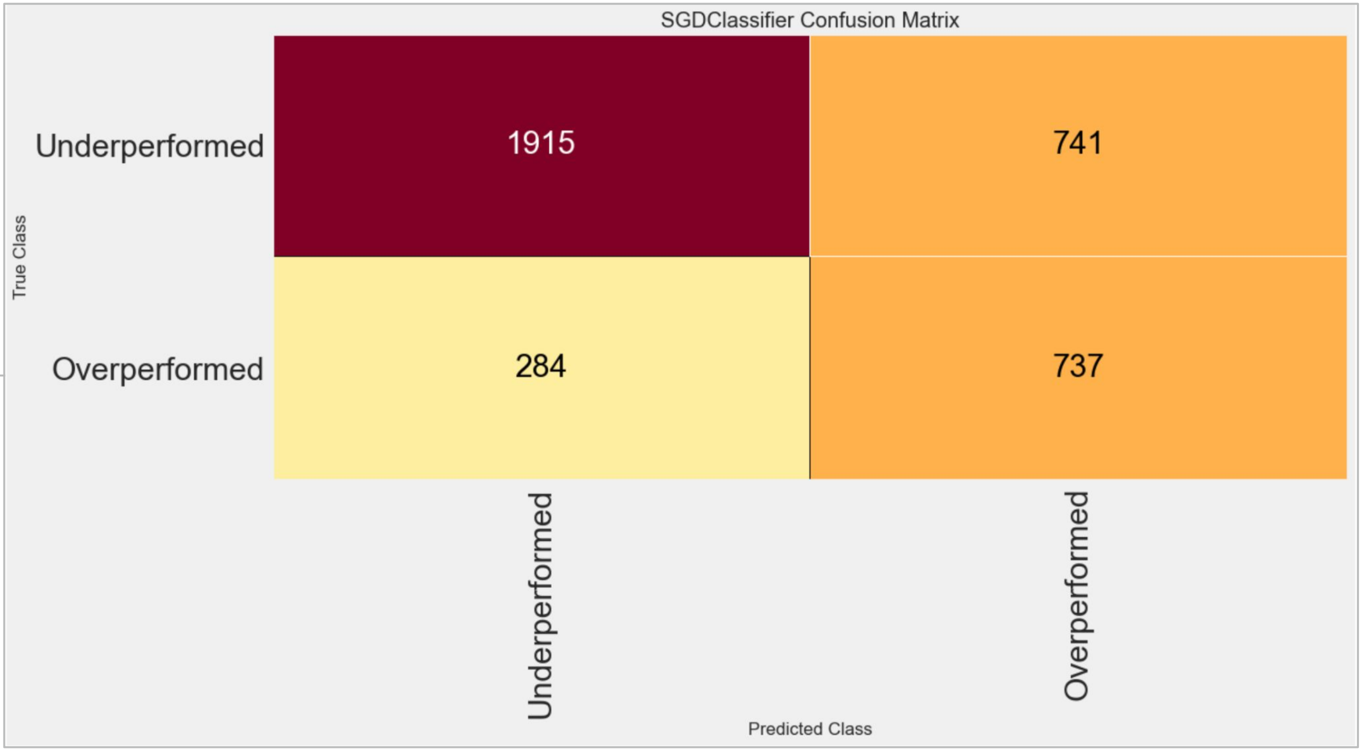
Modeling and Application: Global Model

```
F1 SCORE SVC: 0.589271417133707
```



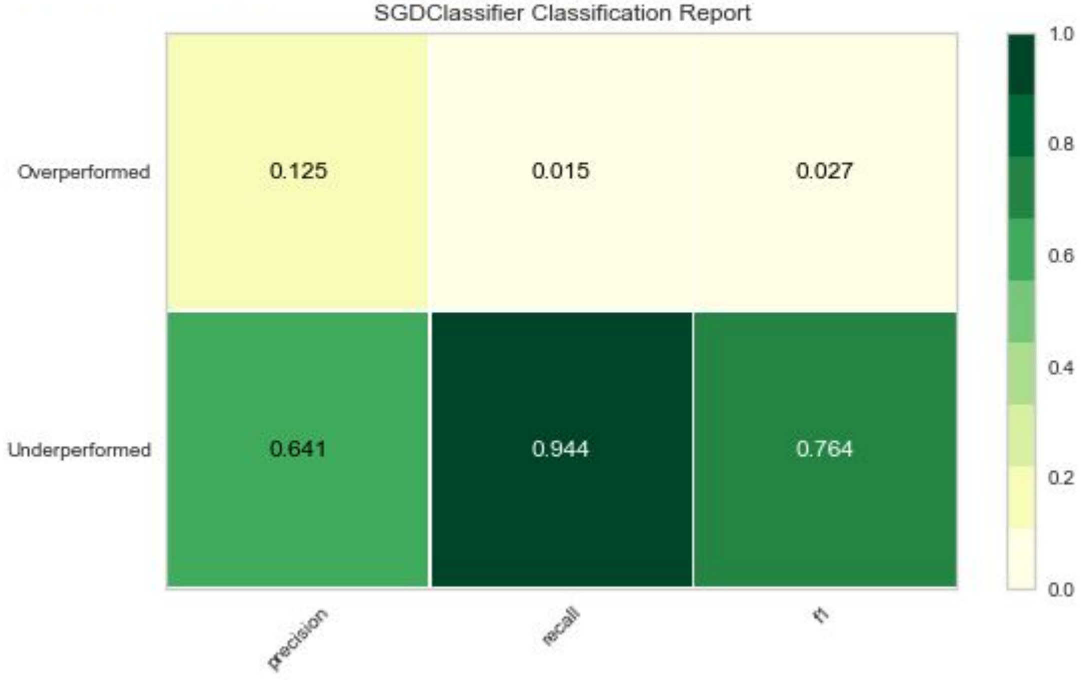

```
F1 SCORE SGDClassifier: 0.589271417133707
```



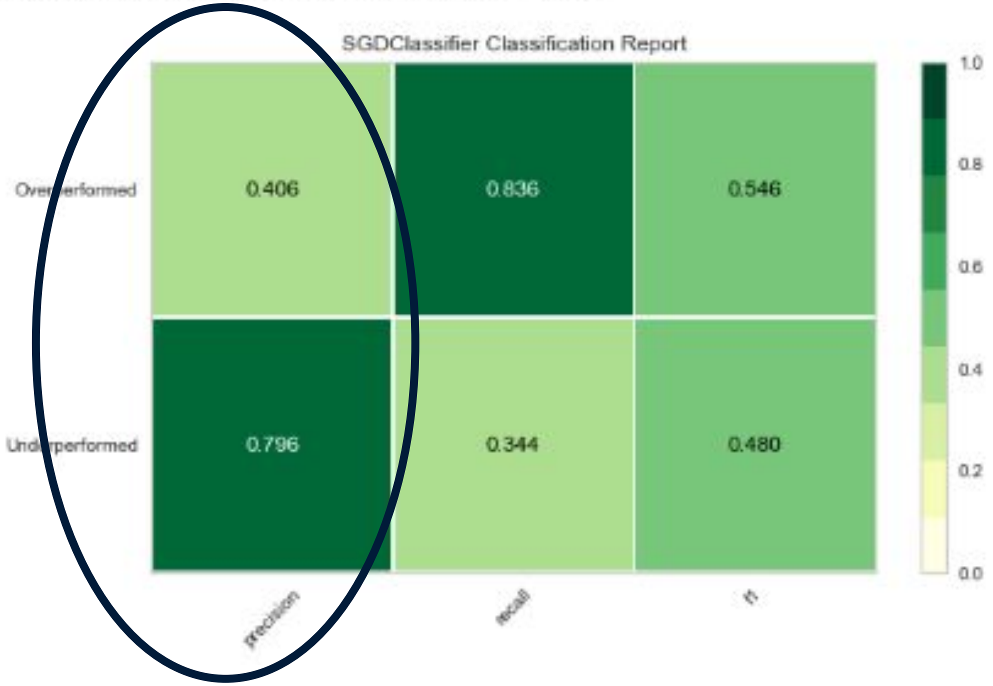


Modeling and Application: Defense Model

```
F1 SCORE SGDClassifier: 0.28070175438596495
```

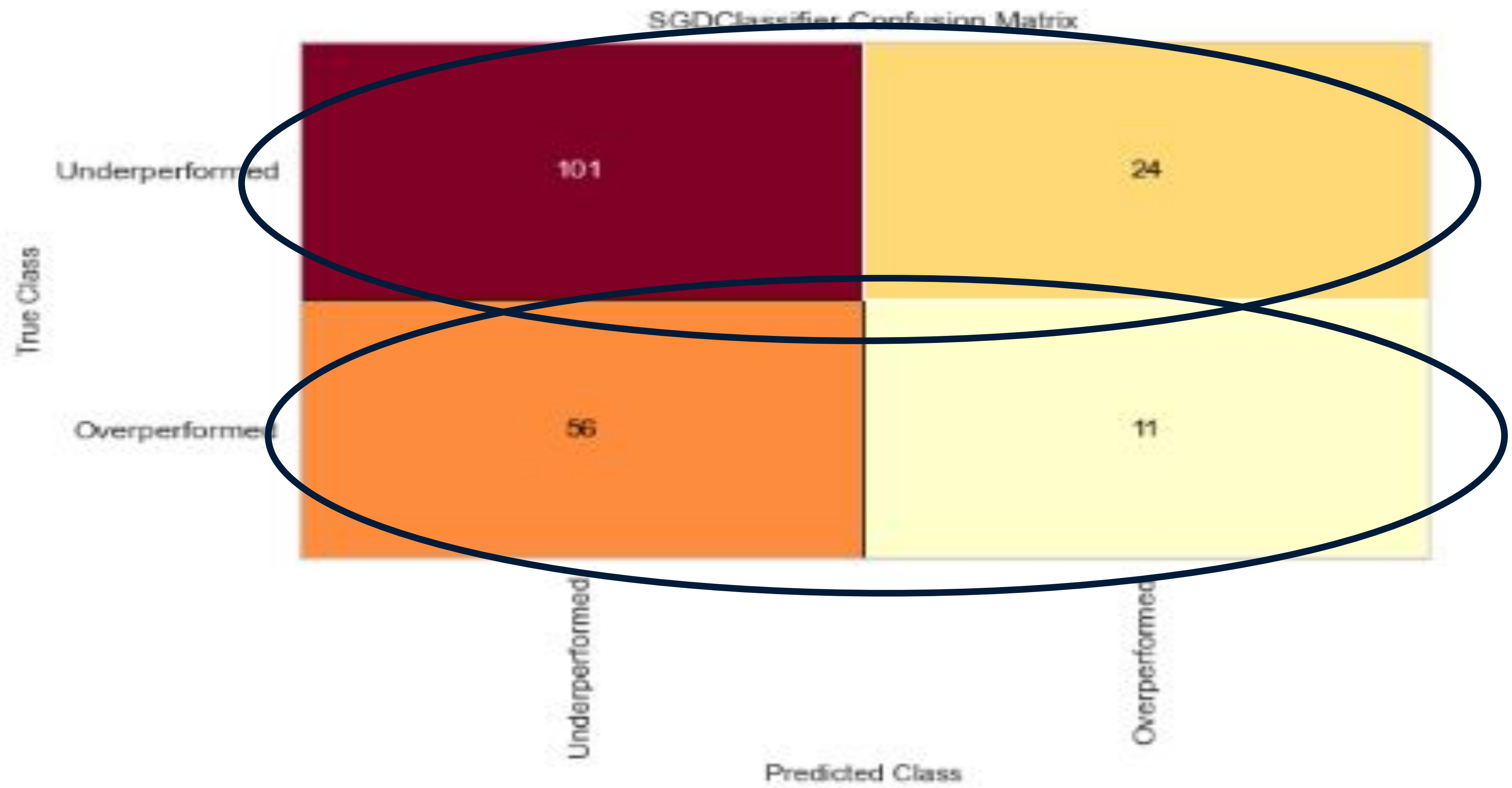



```
F1 SCORE SGDClassifier: 0.4457831325301205З
```




```
Tuned SGDClassifier Parameters: {'alpha': 0.0001, 'class_weight': None, 'loss': 'log', 'max_iter': 200}
Best score is 0.5649953105836519
```

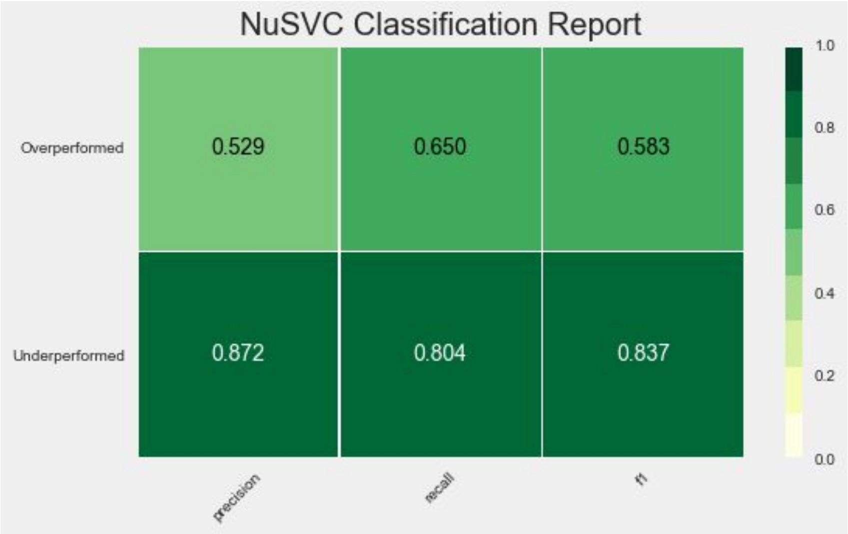
Modeling and Application: Defense Model



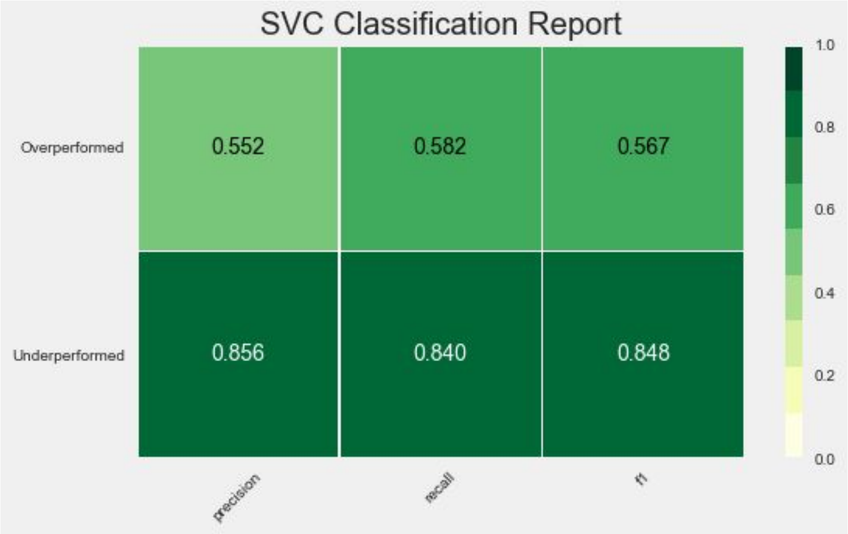
Modeling and Application: Offense Model

F1 SCORE SGDClassifier: 0.5854241338112306

Top performing models with default parameters

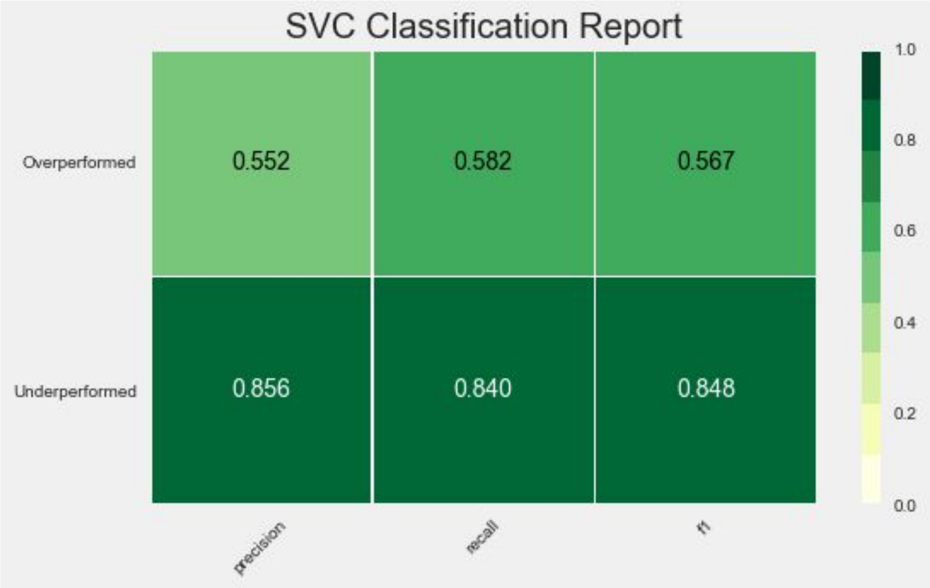


F1 SCORE NuSVC: 0.5831381733021078



F1 SCORE SVC: 0.5667090216010167

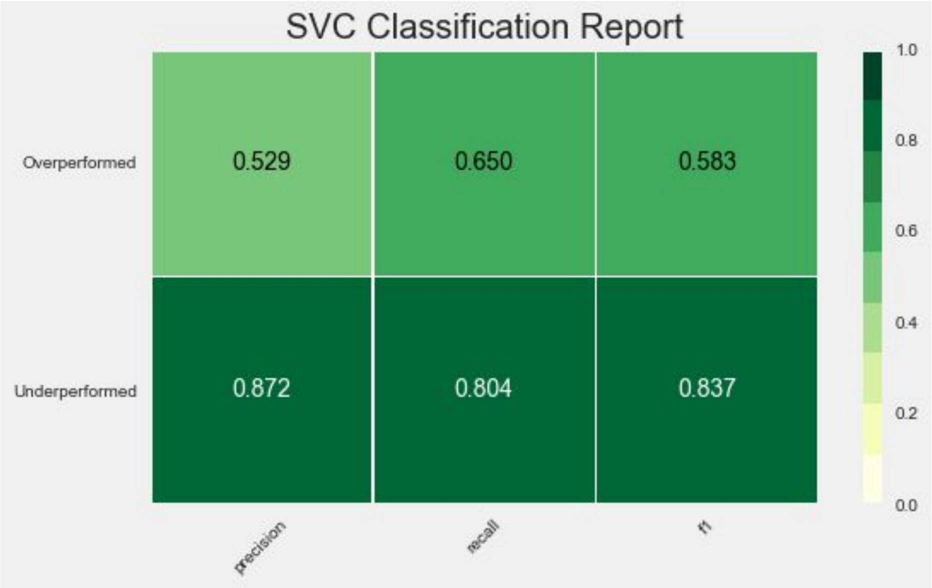
Modeling and Application: Offense Model



F1 SCORE SVC: 0.5667090216010167

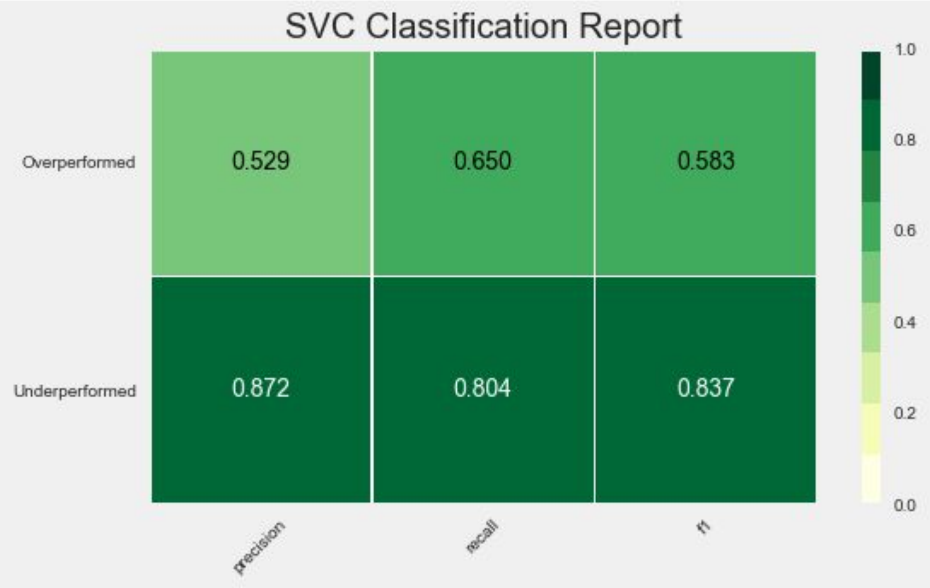
Basic tuning of SVC model

SVC(gamma = 'auto')



F1 SCORE SVC: 0.5831381733021078

Modeling and Application: Offense Model



F1 SCORE SVC: 0.5831381733021078

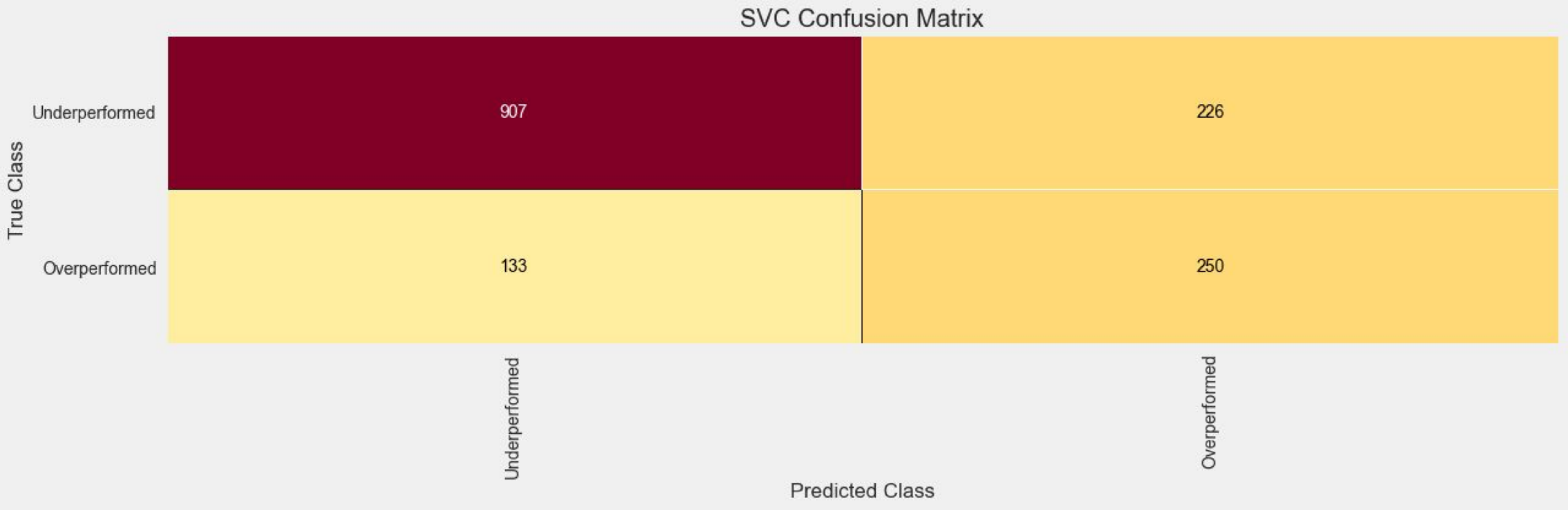
Grid search, best parameters identified

In: grid.best_params_

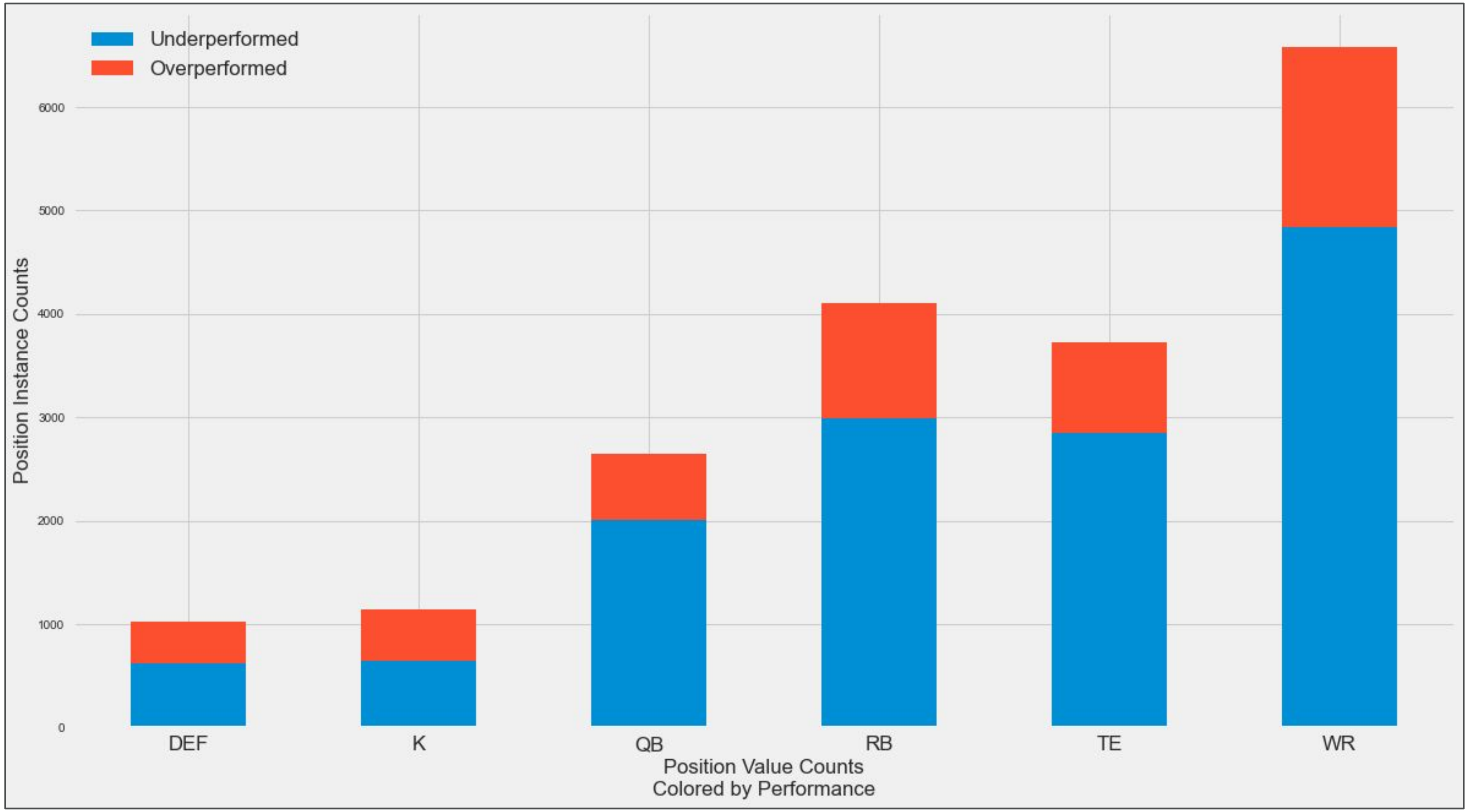
Out: {'C': 1, 'class_weight': 'balanced', 'gamma': 'scale', 'kernel': 'rbf'}

F1 SCORE SVC: 0.5820721769499418

Modeling and Application: Offense Model



Modeling and Application: Wide Receivers Model

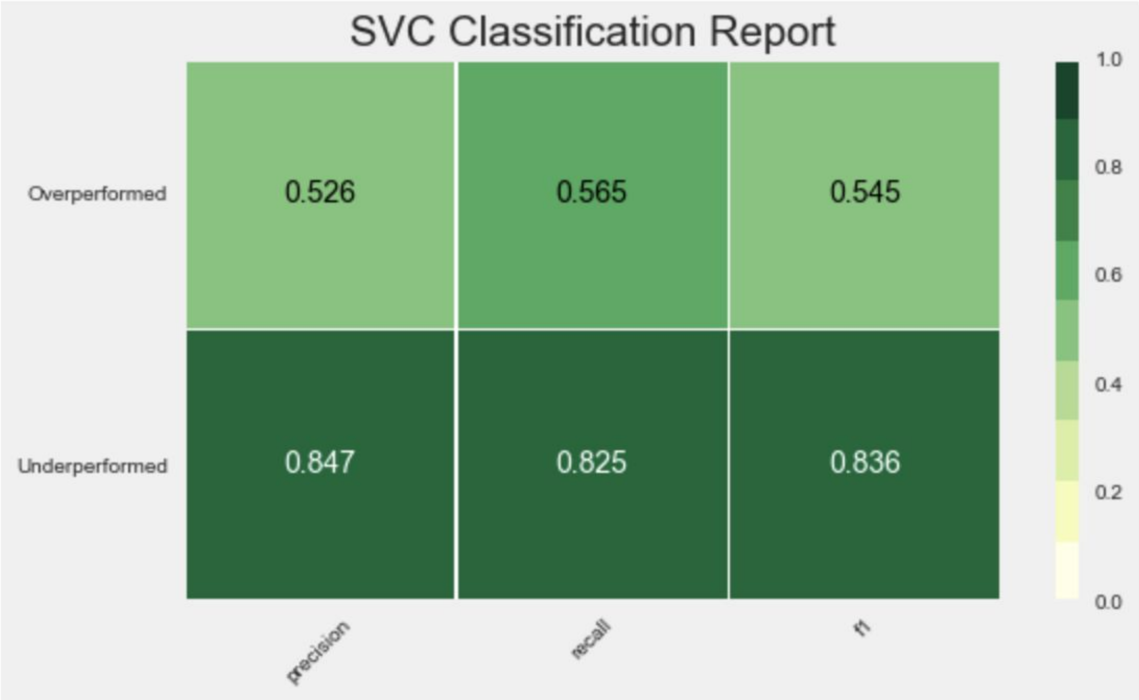


<u>Why Wide Receivers?</u>

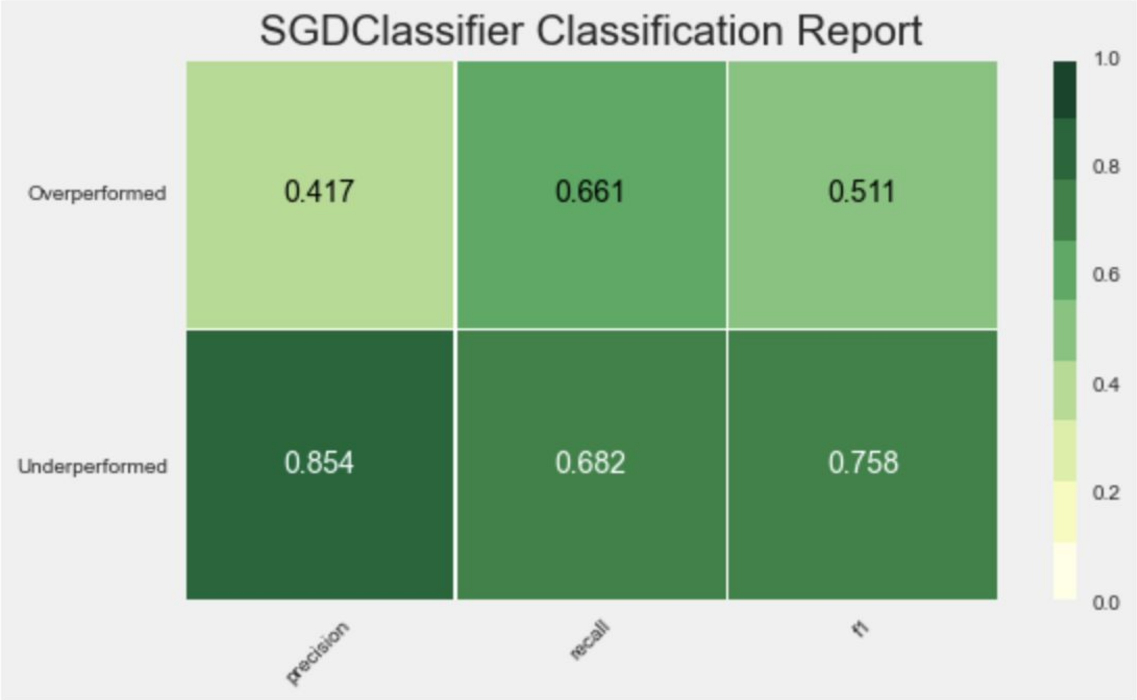
- majority of the players are WR
- less injury prone than running backs
- managers tend to worry more about them

Modeling and Application: Wide Receivers Model

```
F1 SCORE SVC: 0.5449101796407185
```




```
F1 SCORE SGDClassifier: 0.45303030303030303
```




```
grid.best_params_
```

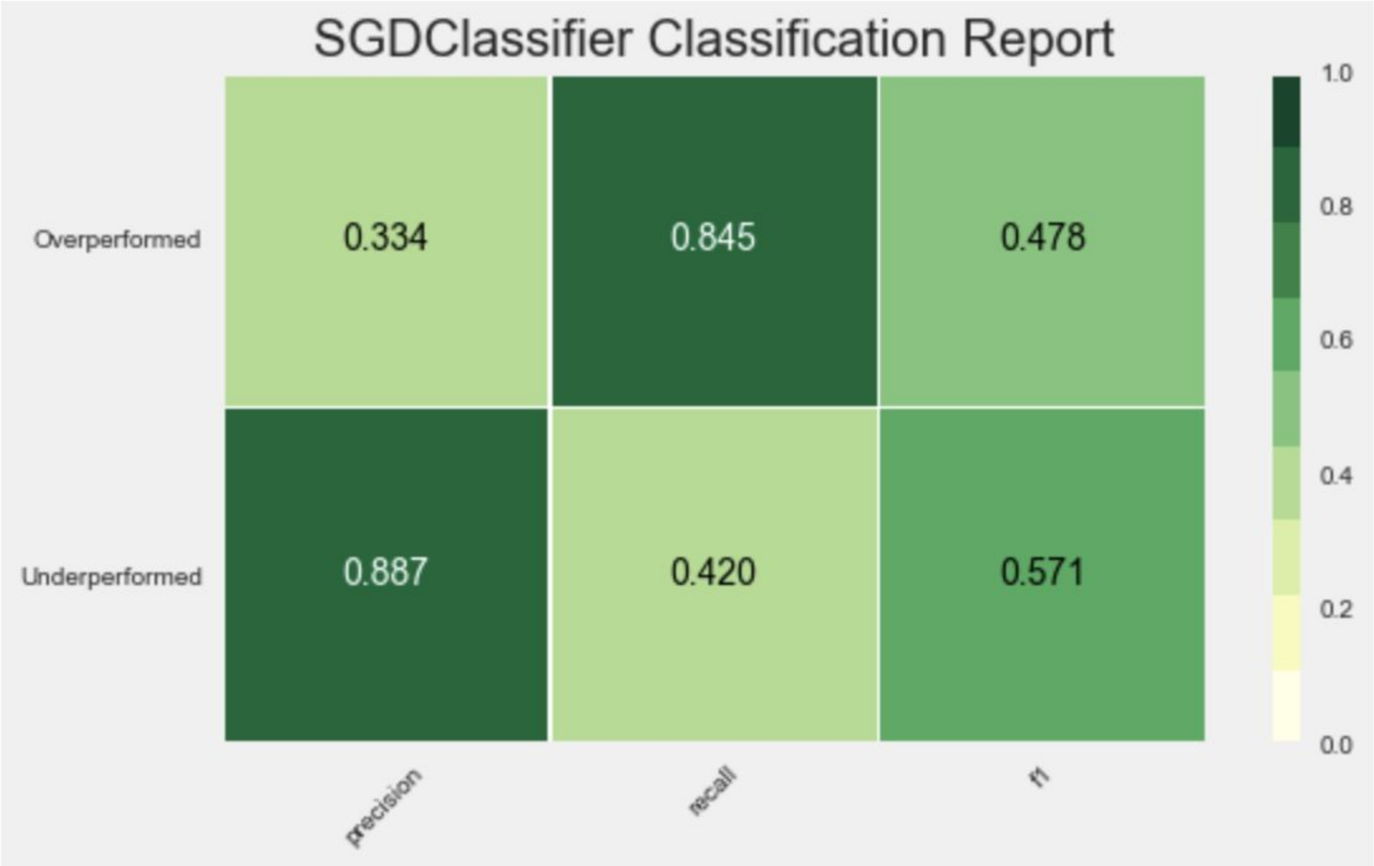
```
{'C': 1, 'class_weight': 'balanced', 'gamma': 'auto', 'kernel': 'rbf'}
```

Modeling and Application: Wide Receivers Model

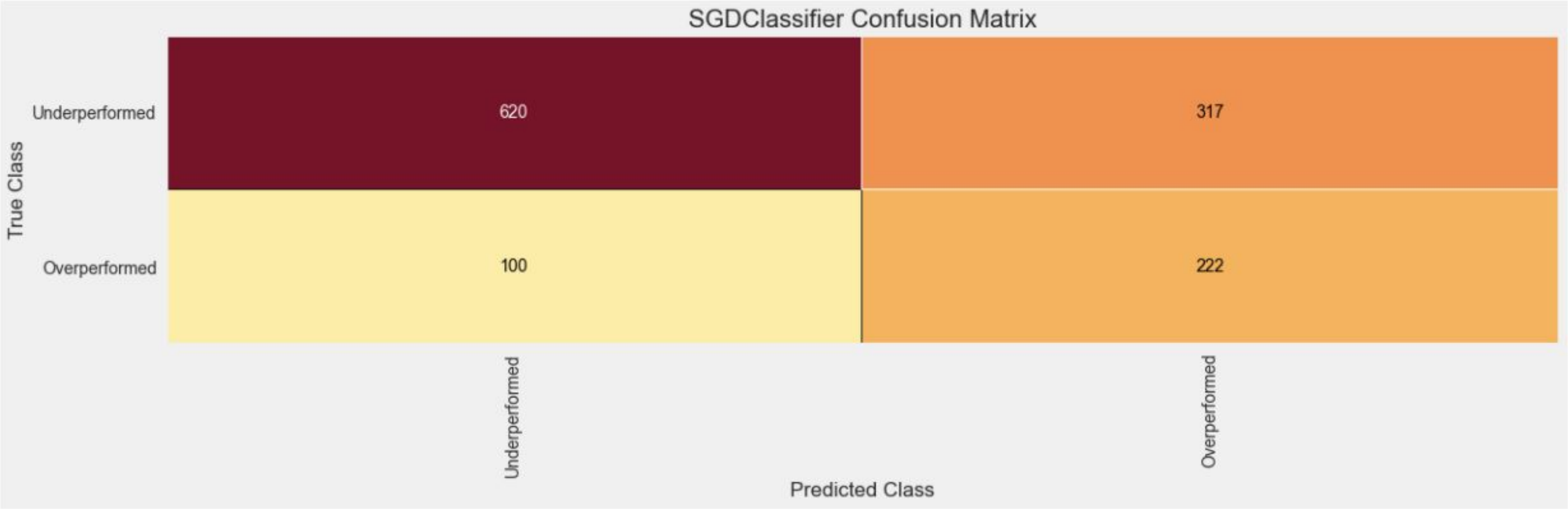
```
grid.best_params_
```

```
{'class_weight': 'balanced',
 'loss': 'log',
 'max_iter': 100,
 'n_jobs': -1,
 'penalty': 'l2'}
```

```
F1 SCORE SGDClassifier: 0.5244338498212158
```



Modeling and Application: Wide Receivers Model



IV. Brief Demonstration and Conclusion

Conclusion

Areas of future study:

- Individual Defensive Players
- Multi-class classification
- Regression
- Play-by-play data
- Application to other sports