Examining the Factors Influencing Injuries in the NFL

Ronan Kelly and Eoin Quinn
{ronan.kelly96, eoin.quinn55}@mail.dcu.ie

May 1, 2025

Abstract

This project explores injury prediction in the NFL using machine learning and explainable AI. The National Football League (NFL) is the highest level of American football played worldwide, a billion-dollar industry which is hindered on an annual basis by injuries causing some of its best players to miss out on game days, ultimately tarnishing the quality of entertainment for spectators. By combining publicly available play-by-play and injury datasets, we engineered a vast range of features like game situation and player fatigue, which we believed had the potential to help predict injury occurrence utilising multiple ML models. However, the most insightful outcomes came from decision tree models based on explainability, which allowed for custom feature importance scoring and weighted predictions. Although a lack of data availability limited model results, the project provided valuable insights and learning in applying explainable AI to real-world sports problems.

1 Introduction

Player safety has become one of the most topical subjects widely debated in sports in recent years. With concerning rises in concussions and soft-tissue injuries as well as high medical and rehabilitation costs, injury prediction and prevention has become a critical area of focus for teams, leagues, and sports scientists alike. As the physical demands on athletes continue to increase, so too does the urgency to develop intelligent systems that can anticipate injuries before they occur. In recent years the NFL has increased the number of matches played and are planning further increases. This emphasises the need for effective workload management to prevent more injuries. We are already seeing the effects of additional games as there was an 18% increase in concussions in 2022, just one year after adding an extra game week [1]. As two sport enthusiasts and Data Science students we believed that with adequate data we could develop a solution to this problem. By leveraging machine learning (ML) methods we hoped to build a model that could aid teams effectively in a professional environment, indicating the likelihood of any player sustaining an injury on a given play.

While building an accurate injury prediction model is important, understanding the model's decisions is equally important. In high-stakes environments like professional sports, those

involved should be able to trust the insights provided by machine learning systems to make informed decisions. With these complex 'black box' models, it's difficult for people to understand the results, limiting the usefulness of the model. By combining machine learning with explainability, we hoped to design a solution that not only helps prevent injury but also builds trust and confidence in its recommendations.

2 Related Work

The area of injury prediction and prevention has been a long-studied topic, the goal of this project was to attempt to replicate and improve upon the accuracy of previous studies.

2.1 Research Summary

A systematic review of 38 relevant studies conducted on injury prediction across different sports, most commonly soccer, found that Area-Under-Curve (AUC) results ranged from between 0.57 to 0.95. Of the 10 studies that didn't report AUC the average precision was 0.79 [2]. Another systematic review of 246 different studies, once again conducted across a variety of sports, found accuracy results ranging from 0.75–0.82, and precision ranging from 0.5–0.85 [3]. Additionally, a review conducted on 19 different studies based exclusively on soccer players found sensitivity ranging from 0.15–0.56 [4].

Across one review 74% of studies analysed factors relating to injury history and player experience. In contrast, only 18% of studies utilised features relating to load such as GPS data. Additionally, 58% of studies also incorporated musculoskeletal screening data, while 47% of studies used body composition and anthropomorphic measurement data [2]. Another review identified medical data, such as body composition, as well as injury history and training load as the most important predictive features [3]. Similarly, in another review, we found the most common key predictors to be medical data such as injury history and body composition as well as training load [5].

In one review, models with the greatest predictive accuracy were tree-based models, most notably Random Forest and XGBoost [2]. In another review, the author commented on the main limitations of the models being the class imbalance of the task at hand, as well as the fact that many events can occur between the collection of predictor variables and the time of the injury [4].

2.2 Gaps in Research

A big gap in the previous research has been the lack of explainability and understanding within what are largely regarded as the 'black box' models that are machine learning.

Other studies have emphasised the importance of utilising explainable AI as a means of gaining greater insights from your machine learning model. Explainable AI shows us how

data points contribute to the output of a model; this can be used to challenge current perspectives by presenting new information [6].

The importance of explanatory models can't be understated as a means of allowing medical professionals, who are not well versed in the area of machine learning, to understand the outputs of these models, as well as the reasons for said output [7].

Our project hopes to build on previous experiments by adding a layer of explainability to the model. This will make it much more accessible to the average medical professional and make it more usable in a sporting environment.

3 Methodology

3.1 Dataset and Preprocessing

Building a dataset was a tough challenge for our project. With a goal of finding out the most influential factors in injury occurrence in the NFL, the number of possible strategies we could take were countless. Considering we had a dataset with a concrete list of injuries which occurred in NFL games in the 2019 and 2020 seasons as well as the plays these injuries happened on, we decided to start there [8]. Concatenating the 2019 and 2020 injuries dataset gave us a list of 1,513 plays (rows), providing some information about the play, and some information about the injury (player that was injured etc.) with 84 columns prior to processing. We considered this dataset as our positive class, thus making all other plays which occurred in the 2019 and 2020 seasons our negative class.

This led us to our search for NFL Play by Play datasets ideally containing some information which we could use as features. We found that the NFL Verse GitHub repository [9] provided us with the most suitable dataset for our task, with the raw data for 2019 and 2020 seasons combined having 94,963 rows and 372 columns.

In order to build a dataset for our first experiment, we had to merge this play-by-play dataset of every play with our play-by-play dataset of plays where players got injured. This required some preprocessing of each dataset, from basics of removing unwanted and unnecessary columns and improving dataset readability to dealing with null values in the most suitable way possible. The injuries dataset consisted of some frustrating errors such as inconsistent date formats and misspellings under certain labels. We also found at a later stage that there were some plays (approximately 100) that were not a part of our injuries dataset where players were in fact injured. These plays have play descriptions with "Player x was injured during the play" written at the end of the play description. We confirmed that players were injured on these plays by looking at game reports online and that they were in fact false negatives, thus we re-labelled these rows as positives. This left us with our processed dataset with 88,336 (1619 of positive class, 86717 of negative) rows and 41 columns. Not all of these columns were used as features for our first model; however some columns were necessary to keep for play descriptions and potential feature engineering at a later stage.

3.2 Exploratory Data Analysis (EDA)

The next stage of this project was to begin analysing the dataset we had created looking for correlations between our features and injuries.

Taking inspiration from a book called 'Data Mining Concepts and Techniques' [10], we began visualising our injury data. We looked at the mean, median and mode of injuries per game as well as the interquartile range, discovering on average 3 injuries per game. We then visualised our data creating box plots and histograms displaying the number of injuries per game.

The next steps looked at analysing the correlation between our features and injuries. We first began with play type, where pass and run were the highest correlated even after accounting for frequency. Next, we looked at formations, showing that shotgun and pistol formations account for most of the injuries. We were then intrigued by the correlation between pitch type and injuries, this showed us that grass pitches account for 60% of injuries but only 46% of pitch types in the league. Majority of injuries occur outdoors, with injury probability being highest on open-roofed stadiums. As for play based features, we found injury probability to be highest when gaining between 25 and 35 yards on a given play.

As our dataset is based on play-by-play information, we had to confirm there were no instances of multiple players being injured on the same play. Through our research we found no instances of such. However, we did find 38 times where the same player got injured multiple times in one match.

3.3 Feature Engineering

Feature engineering is an incredibly important step when developing a machine learning model, and as such we utilised both our knowledge of the sport as well as our knowledge of machine learning to create useful new features for our model.

We created features detailing various elements of the individual game situation, such as, features classifying play situations like 3rd and long, 4th down plays and goal line plays. We also looked at features describing the defence, such as ranking, number of defenders in the box and classifying a blitz situation. We also explored classifying specific game states utilising a close game feature to understand game intensity.

We collected a variety of weather-based features such as precipitation and temperature from Open-Meteo [11]. We also categorised these features to make them more effective for our model.

Finally, we created features relating to the team and their specific attributes. We created features relating to team rest and season progression. We also looked at their team strategy, creating counters for the number of times the home team is on defense and offense as well as counters for different play types to gauge player fatigue levels. We finally created features which looked at the offensive predictability creating a ratio for the number of pass plays, as well as classifying how good a play is via its Expected Points Added (EPA).

These newly created features played an important role in improving our accuracy and also aided the explainability of our model.

3.4 Modelling Approach

We felt it would be best to start with a decision tree model for our first experiments for simplicity. After poor results using a standard train-test split, we pivoted to stratified k-fold cross-validation. To ensure class balance in each of the 53 folds, we oversampled the 1619 positive cases with replacement and under-sampled the negative cases without replacement, selecting 1619 from each class for training and testing. This meant that the model saw all of the positive cases in every fold and iterated through a new set of 1619 negative cases in each fold. This allowed us to train and test on a balanced dataset each time.

With the decision tree as our baseline model ultimately performing poorly on our first few experiments, we decided to try a random forests and gradient boosting models. We believed both bagging and boosting ensemble methods could potentially improve our results, and while they did lead to some improvement, the increases in our metrics were minimal. We also decided to try some hyperparameter tuning on a standard gradient boosting model.

3.5 Explainability

Considering how minimal the improvements were in the results that we achieved, we took some time to weigh up our options before continuing. We understood that our model's performance could never match that of some of the related work models, as we had no access to the data needed to replicate or improve upon results from related work. These studies had access to player training load, GPS data and player medical history data. After seeking advice from our supervisor, we ultimately decided not to look at predicting whether each player would get injured or not on each play. Instead, we looked at interpreting our current feature set (predicting whether any player on the field would get injured or not on a given play).

With this new goal in mind, we decided to go back to the decision trees model with our most recent list of features. Decision trees are best for explainability because they split data based on clear, human-readable rules, making it easy to trace how a prediction was made by the model. We examined the rule base in 3 of our 53 folds, selecting the least accurate, most accurate, and the fold closest to the mean accuracy for analysis.

We were able to label the test set for all folds to show which rule caused each prediction, allowing us to track the accuracy of each rule and how many times the rule was used in each fold. We were also able to look at the segments/terms of each rule (the branches) in each fold. This allowed us to leverage some simple maths on each term, taking into account the accuracy and usage of each rule the term appears in to show a term's "grade".

We were also able to weight the usage of each term based on how much the term is used in comparison to all other terms, allowing us to easily score each term and then rank them based on these scores.

After this, we grouped each term by the feature present in the term, taking the mean accuracy for the feature across all terms to score each feature. Having done this for each of our 3 folds we were able to take the sum of scores for each feature in each fold and normalised these final scores to add up to one. This provided us with our own feature importance algorithm by ranking these features on our normalised scores.

4 Experiments and Results Analysis

For a full list of experiment results see Appendix A.

4.1 Baseline Modelling

Our first experiments established a performance baseline.

Experiment 1 & 2:

We began with a simple Decision Tree classifier trained on the raw dataset, which contained significant class imbalance. This performed poorly, achieving 0 for both precision and recall with 98% accuracy due to class imbalance. This led to our decision to use precision as our key metric as it accurately frames the quality of our model across both classes.

Subsequently, we focused on attempting to rectify the issues with class imbalance. To do this we conducted k-fold cross-validation, where the classes are perfectly balanced.

Accuracy	Precision	Recall
52.98	52.60	41.25

Table 1: *K-fold Cross-Validation with Balanced Classes*

This became our baseline model from which we compared later experiments against.

4.2 Feature Engineering

Having established a baseline, we began experimenting with feature engineering to improve model performance.

Experiment 3 - 7:

Firstly, we created a play counter, which tracked when the home team was on either offense or defense, showing player fatigue levels. After this, we attempted to utilise strategy-based features as means of predicting injury. Our thought process was that certain formations or defensive styles may lead to increased injury risk, adding defenders in the box and offensive formation features.These additional features brought noticeable but minor improvements in each of our metrics.

Considering the improvement "play count" added to our model, we felt counters of specific play types (run/pass) could also help. However, this saw decreases of roughly 0.2% in both

accuracy and precision and 1% for recall, this is clearly adding unnecessary noise to the model which we left out for future experiments.

Given the relatively poor performance of our decision trees model, we decided to test both Random Forests and XGBoost models, still using k-fold cross validation and the same feature set used in experiment 4. The random forest model showed significant improvements in both precision and accuracy, however, looking at recall we saw a significant decline.

Accuracy	**Precision**	**Recall**
56.22	59.21	37.85

Table 2: *Random Forest on Formation-Based Dataset*

The XGBoost model also saw improvements in precision and accuracy, as well as a minor decrease in recall compared to our decision trees model, but overall the XGBoost model did not perform as well as the Random Forest. While improvements were minor, these experiments clearly demonstrated the valuable improvements ensemble methods can make through bagging and boosting techniques.

4.3 Advanced Features & Improvements

At this point we began testing more advanced techniques such as hyperparameter tuning and ensemble methods alongside additional feature engineering to improve the model.

Experiment 8-13:

Experiments 8, 9 and 10 only contained additional feature engineering. We added defensive rankings due to the physical nature of the sport, hoping that teams playing against better defences would be more likely to have someone get injured. We then added environmental/weather data such as temperature and precipitation levels in hopes that slippery surfaces or cold conditions would show increased likelihoods of injury occurrence. Despite logical reasoning behind the inclusion of these features, in reality these features only added noise to the models' seeing decreases in our metrics across the board by about 2% to 3% compared to experiment 6.

With the addition of entirely new features not working, we decided to adjust some of the features we already had, categorising weather and creating game situation features. These included categorising down and distance, position of the ball on the field, rest differential and play style related features. Also given our plateauing model performance, we decided to test another new model, Gradient Boosting, yielding our results for experiment 10 below which saw significant improvements in all metrics.

Accuracy	**Precision**	**Recall**
59.25	58.29	62.94

Table 3: Gradient Boosting with Extensive Feature Engineering

Following on from this we decided to conduct a hyper-parameter tuning experiment, we hoped this would provide the best possible performance from our Gradient Boosting model. However, this was not the case, with a decrease in all metrics by between 1-3% compared with our original Gradient Boosting model using default parameters. For our final feature engineering experiment we decided to add EPA (Expected Points Added). This is a measure how much a play improves the team's scoring chance, so we believed plays with higher EPA would lead to more injuries as we will see greater risks taken from both sides, for example the defence sending an extra defender to tackle the quarterback.

Our 13th experiment focused on attempting to use a stacked ensemble model. We believed multiple models combined could maximise each of their strengths to build the best possible model. We utilised the scikit-learns built-in stacking method on three models, Random Forest, XGBoost and Gradient Boosting, using majority vote with no weighting. Unfortunately, this experiment did not yield the results we were expecting, seeing decreases of between 1-4% in precision and accuracy, and over 20% for recall. This verified to us that our task was extremely difficult, as a significant reduction in these metrics shows that when individual models struggle, combining them may amplify their weaknesses rather than improve performance.

4.4 Explainability Models

As explained in the methodology, we decided to go back to a decision tree model which enabled us to score our features in the most recent model by looking at rule usage and accuracy. This helped us understand which features were a help to the model and which were more of a hindrance. We decided to run 3 more experiments with our new information from our explainability. Firstly, we ran a decision tree model with the feature set we had most recently used for random forests and gradient boosting models to give us a new baseline (Experiment 14).

Experiment 15:

After this, we wanted to test the validity of our feature importance algorithm by comparing it to scikit-learn's built-in function. We were able to do this by taking only our top 10 features from our feature importance algorithm and comparing it to the top 10 listed features from scikit-learn's built-in function, which we retrieved in experiment 14, and running each model. Ultimately, the model consisting of our top 10 features performed slightly better, with an accuracy of 57.45% (0.75 greater than the model using scikit-learn's features), precision of 55.82% (0.57 greater than the model using scikit-learn's features), and recall of 69.54% (1.67 greater than the model using scikit-learn's features).

Experiment 16:

With the knowledge that our feature importance ranking was valid, we wanted to test if we could increase our model performance metrics by applying a weighting to our decision tree model used in experiment 14 based on our feature importance ranking. After applying weights to some of the more influential features, considering thresholds from the rule-based splits observed in our prioritised folds, we were able to achieve our highest recall score of 71.1%, as well as solid accuracy and precision scores of 58.07% and 56.28%, respectively.

This truly verified to us that our feature importance algorithm was extremely insightful for our model's performance.

5 Final Results

5.1 Modelling Results

As mentioned, the goal of this project was to create a model that could compete on a standard that would be useful in a professional environment. That, however, is a difficult task and unfortunately our model performance was not at the level of studies carried out in the past.

That being said, each of our models had their own strengths and performed well in their own regard. Our Random Forest model from experiment 6 achieved the best results with respect to precision at 59.21%, the Gradient Boosting model from experiment 12 struck a good balance across all evaluation metrics, most notably our best accuracy at 59.59%. However, overall, our best performing model was the weighted Decision Trees model created in experiment 16, although the accuracy and precision were slightly down on previous experiments, the recall of 71.1% is a massively significant improvement when compared with past experiments. Recall is particularly important in injury prediction because minimising false negatives means the model rarely misses actual injuries. This indicates a more cautious approach, as it rarely predicts 'no injury' when one has actually occurred.

We also conducted feature importance across our various models, utilising scikit-learn's built-in feature importance algorithm. We averaged the scores across every model in our k-fold cross validation. Across our best models, defenders in the box was significantly our most important feature, followed by EPA.

5.2 Explainability Results

After transitioning to explainability, we began by extracting the rule bases for our three prioritised folds. By assigning a number to each rule, we were able to track the rule classifying each prediction. From there, we broke down each rule into its terms (e.g. `defenders_in_box > 2.5`) and scored them based on their accuracy and usage. Subsequently, we combined these scores to create a rule score from which we could rank the rule quality. Finally, we ranked our features for comparison against scikit-learn's built-in function.

Rule Insights

Since we based our scoring on accuracy and usage, we wanted to gain insights into the misclassifications in each fold. With approximately 40 rules per fold, we found a concentration of our misclassifications within only a few rules.

In our mean fold:

- There were 1390 misclassified instances (57% accuracy)

- 3 rules accounted for 60% of these errors
- 6 rules accounted for 77% of these errors

In our top 2 most prevalent rules, `EPA` $\leq$ `0.47` was one of our worst performing terms. As most EPA values fall between -1 and 1, our model was clearly struggling to differentiate between instances. On some occasions, it was only one term causing all the issues for a specific rule, where the rest of the terms performed adequately.

In our best fold:

- There were 1093 misclassified instances (66% accuracy)
- 3 rules accounted for 63% of these errors
- 8 rules accounted for 90% of these errors

Across our best fold, terms appeared to perform well with most terms achieving accuracies greater than 50%.

Based on our rule scores, our top two rules (rule 24 and rule 31) accounted for 48% of the errors in this fold. This was because they classifed the most instances.

In our worst fold:

- There were 1756 misclassified instances (46% accuracy)
- 5 rules accounted for 82% of these errors
- 1 rule accounted for 38% of these errors – Rule 20

Based on our scoring system, Rule 20 was our highest ranked rule in this fold. This was likely because it classified 42% of the fold with 51% accuracy.

The term `offensive_formation` $\leq$ `1` caused all the errors for one of our rules. The rule accuracy dropped from 44% to 20% after applying this condition.

Overall, term accuracies were very poor, causing issues with the quality of the rules.

Feature Importance & Weighted Model:

We noticed from running multiple experiments that scikit-learn's feature importance function was flawed, as features with no importance affected our model performance when removed. Because of this, we aimed to manually analyse our features to produce our own ranking. By creating feature scores based on term scores, we were able to rank each feature in our dataset by its importance to the model in each fold. By summing the importance of each feature in each fold and then normalising, we retrieved an overall feature importance score.

Our Feature Ranking
defenders_in_box
game_seconds_remaining
epa
offense_formation
HOME_day_since_last_game
home_team_on_defense
yardline_100
run_count
stadium
surface_type

Scikit-learn Feature Ranking
defenders_in_box
epa
game_seconds_remaining
yardline_100
score_differential
OffenseTeam
stadium
offensive_predictability
current_defense_rank
home_team_on_offense

As we can see, there are a lot of crossovers between the feature rankings. Despite the similarities, we deemed it important to evaluate these features through modelling. We set up an experiment utilising only the top 10 features from both methods on two separate models to compare outputs. We saw noticeable increases in all metrics when using our feature rankings, further justifying our method of calculating importance.
For visual comparisons of these rankings, see Appendix B.

With this validation of features, we attempted to improve our base decision trees model by utilising weighted features derived from these rankings. Specifically, we adjusted instance weights based on key feature thresholds: increasing weights when `defenders_in_box` exceeded 2.5, `epa` was less than or equal to 1, or `play_type` was above 7.5, and decreasing them when `game_seconds_remaining` was greater than 2300. These thresholds were determined due to their presence in rule bases. After running this test, we saw significant improvements in performance, further verifying our feature scores.

6 Conclusion

While our model may not be able to compete with the performance benchmarks set by prior studies with access to more detailed datasets containing medical and workload data; our transition to explainability provided us with great insights into our model's shortcomings. The task we set out to complete was extremely ambitious, predicting the exact point in a game at which an injury occurs is not a common avenue for injury prediction and prevention. The scarcity of injury occurrence relative to the number of plays, as well as the influence of external factors makes this problem all the more difficult. Despite the creation of logical, engineered features, capturing nuances within a play such as rest, game situation and weather conditions, we did not see a remarkable improvement in model performance. That being said, we still saw gradual improvement from our baseline to our most advanced model, of 6.5% in precision and accuracy, and even 30% in recall.

The development of our own feature importance scores shows that even with limited data, we can begin to discover which factors most significantly influence injury risk. It gave us the ability to compare our findings with that of the scikit learn function and improve our model with weighted features. Explainability also enables professionals to adjust strategies, such

as workload management, based on the important features within our model. This will help to somewhat bridge the gap between data science and on-field decision making.

Throughout this project we have gained valuable knowledge in the area of explainable AI as well as dealing with setbacks and adversity which will definitely stand to us as we progress in our careers. Although, our models showed some promise in predicting NFL injuries we recognise that further improvements could be made by incorporating more player-specific features. Additionally, expanding the dataset to include multiple seasons could further enhance model skill.

While our model has limitations, it marks a valuable step toward applying explainable AI to injury prediction and supporting player safety in professional sports.

References

[1] ESPN. (2022). NFL says regular season concussions increased 18% in 2022. Retrieved from https://www.espn.com/nfl/story/_/id/35582897/nfl-says-regular-season-concussions-increased-18-2022

[2] BMJ Sports Medicine. (2023). Machine learning approaches to injury risk prediction in sport: a scoping review with evidence synthesis. *BJSM*, 59(7), 491. Retrieved from https://bjsm.bmj.com/content/59/7/491

[3] JEO-ESSKA. (2021). Machine learning methods in sport injury prediction and prevention: a systematic review. *JEO-ESSKA*. Retrieved from https://jeo-esska.springeropen.com/articles/10.1186/s40634-021-00346-x

[4] Sportsmith. (2022). Machine learning applications in soccer. Retrieved from https://www.sportsmith.co/reviews/may-2022/machine-learning-applications-in-soccer/

[5] Sports Medicine Open. (2022). Predictive modelling and injury risk in athletes. Retrieved from https://sportsmedicine-open.springeropen.com/articles/10.1186/s40798-022-00465-4#Sec7

[6] Sportsmith. Predictive modelling and its use to identify injury risk. Retrieved from https://www.sportsmith.co/articles/predictive-modelling-and-its-use-to-identify-injury-risk/

[7] MDPI. (2020). Title of the article. *MDPI*, 24(1), 119. Retrieved from https://www.mdpi.com/1424-8220/24/1/119

[8] Sammie Erne. NFL Injury Analysis. Retrieved from https://github.com/sammieerne/NFL-Injury-Analysis

[9] NFLVerse. NFLVerse Data Release. Retrieved from https://github.com/nflverse/nflverse-data/releases?page=1

[10] Han, J., Kamber, M., & Pei, J. (2011). Data Mining: Concepts and Techniques (3rd ed.). *Morgan Kaufmann*. Retrieved from https://myweb.sabanciuniv.edu/rdehkharghani/files/2016/02/The-Morgan-Kaufmann-Series-in-Data-Management-Systems-Jiawei-Han-Micheline-Kamber-J-Concepts-and-Techniques-3rd-Edition-Morgan-Kaufmann-2011.pdf

[11] Open Meteo. Weather forecast API. Retrieved from https://open-meteo.com/

Appendix

A Experiment Results

Experiment	Desc	Model(s)	Accuracy	Precision	Recall	Notes
E1	basic DT predicting entire dataset	DT	98.12	0	0.00	poor precision data imbalance
E2	k-fold DT (53 folds). Removed pointless features	DT	52.98	52.60	41.25	Significant increase in precision when balancing
E3	k-fold DT. (drive play counter feature added)	DT	53.03	53.20	42.55	Minor increase across the board
E4	Adding defenders in the box and offensive formation features	DT	53.61	54.03	43.41	
E5	k-fold. (run & pass play count added)	DT	53.34	53.81	42.64	metrics decrease, will leave out of future experiments
E6	k-fold RF (same features as E4)	RF	56.22	59.21	37.85	RF better with additional features
E7	k-fold XG Boost(same features as E4)	XGB	55	55.85	41.63	significant decrease in precison
Quick Check	RF & XGB with/without engineered features (RF improves, XGB has gets slightly worse)					
E8	RF with Defense rankings on top of E6 features	RF	55.99	58.82	37.12	slight decrease, needs some refining
E9	Same as E8 just with additional weather features - Temp, Precipitation	RF	54.92	57.20	34.72	Another slight decrease however temp becomes 3rd most important variable
E10	Conducted additional Feature engineering	RF/ **GB**	59.25	58.29	62.94	Significant increase in overall accuracy, however decrease in precision
E11	Same features as E11, conducted hyperparameter tuning	RF/ **GB**	56.37	57.42	45.81	Decrease in comparison in E10
E12	Adding in an EPA feature to see improvement	GB	59.59	58.81	61.73	Nice improvement in both metrics, slight decrease compared to best recall
E13	Conducting a test using a stacked model	RF/GB/XGB	55.84	57.81	39.50	slight decrease, not ideal, very poor recall
E14	Explainability Experiment with additions from above	DT	55.43	56.96	68.53	Base DT kfold model significantly best recall
E15	Testing the valididty of our FI against python FI, top 10 features	DT (Our Model)	56.7(+0.75)	55.25(+0.57)	67.87(+1.67)	Our model shows noticeable signs of improvement from the python FI
E16	Use of weighted features - Df_in_box, run_play, EPA, Game_secs_Rem	DT	58.07	56.28	71.10	Noticeable improvements from base model

Figure 1: Overview of Model Experimentation Results

B Feature Importance Visualisations

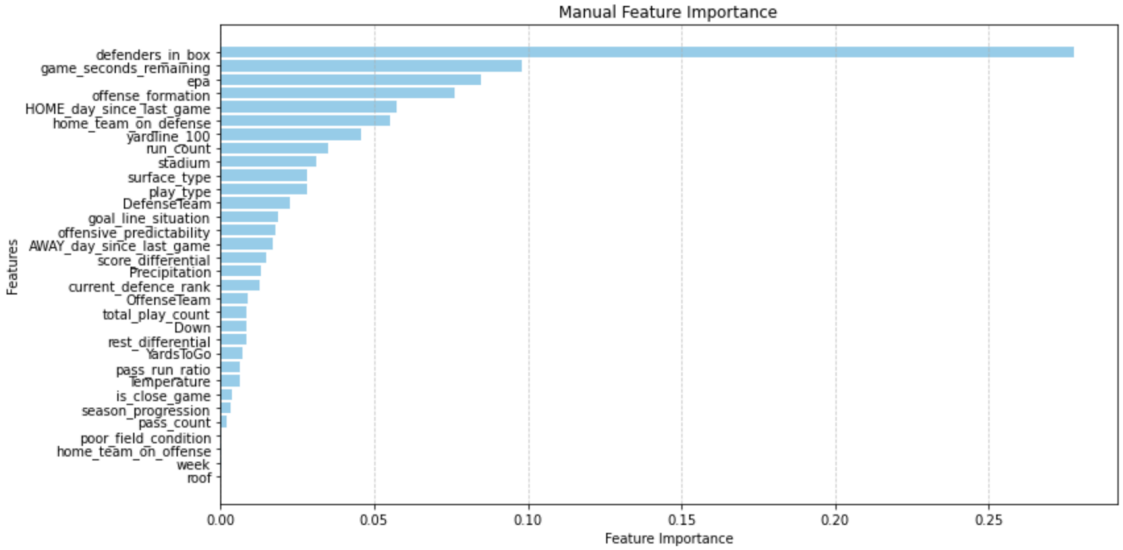

Figure 2: Feature Importance Based on Manual Weighting

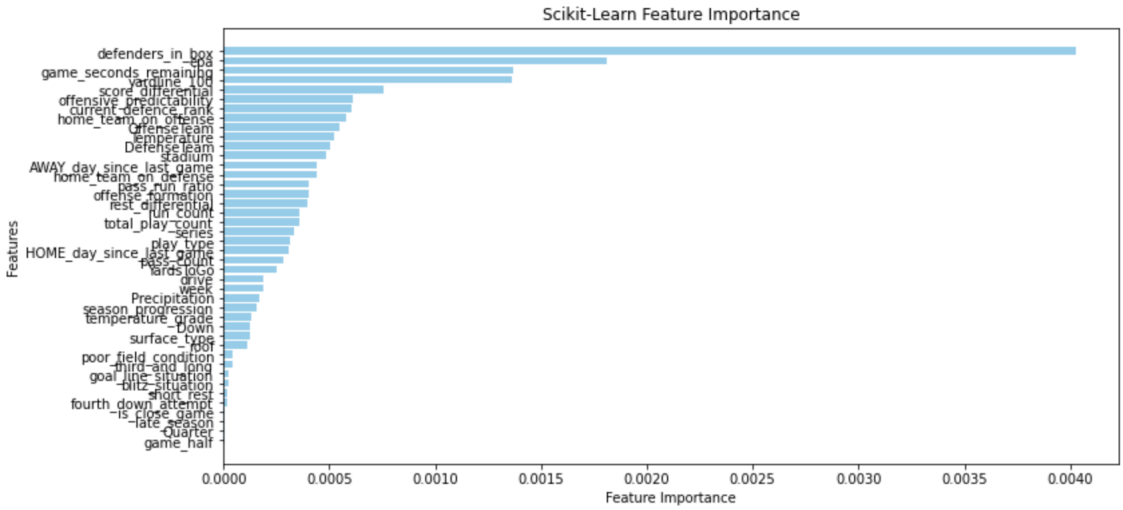

Figure 3: Feature Importance from Scikit-Learn