Soccer Match Prediction

Segun Madarikan

Fall 2022

#Introduction

The purpose of this project is to use machine learning prediction techniques to predict the outcome of English soccer league premier league matches. I am a big soccer fan and thought that this would be a fun way to apply techniques learnt in class to something in the real world. Additionally, more sophisticated versions of these techniques/algorithm can be used in sports analysis and betting. I will be using different modeling techniques to classify these fixtures as wins or not wins. The premier league results dataset that I obtained from kaggle.com has data on all of the soccer matches in the league from the 2006-07 season all the way to the 2017-18 season. In this report, I will be investigating matches played by the soccer club Everton. I chose this club because it has a tendency to not perform exceptionally well or poorly in any given season. This will enable me to observe a club that has a good mix of wins, losses and draws. For features/parameters, I took into account Everton's recent perfomances before a fixture(form), the opposing team's form, the last few results between the two teams(head to head), and whether Everton were playing at Home or not. These are commonly expected to influence the outcome of games in soccer. I will use a logistic regression model and decision tree model to predict my outcome.

#Feature Creation and Data Wrangling

First I read the dataset into a table.I then added a column to give every game a unique ID based on when it was played. I also added a column that states the name of the winner.

```
epl_results_tbl <- read_csv("~/HW 341/results.csv") %>%
  mutate(winner = ifelse(home_goals>away_goals,home_team,ifelse(home_goals==away_goals,"Draw",away_team

epl_results_tbl <- epl_results_tbl %>%
  mutate(gameID = seq.int(nrow(epl_results_tbl)))
```

My first feature was the difference in form between Everton and their opposing team. This was represented as an integer in the range -15 to 15. Firstly, Everton and the opposing team's form was calculated with the help of the function below find_form. This function finds the results of the teams last 5 games and awards 3 points for a win, 1 point for a draw and 0 points for a loss. It then returns the sum of all the points obtained in these 5 games. The form difference feature is Everton's form minus the form of the opposition.

```
find_form <- function(team,game) {
  f_table <- epl_results_tbl %>%
    filter(gameID < game)%>%
    filter(home_team == team | away_team ==team )%>%
    mutate(indic = row_number())
  f_table <- f_table %>%
    filter(indic > nrow(f_table)-5)%>%
    mutate(points = ifelse(winner==team,3,ifelse(winner=="Draw",1,0)))
  sum(f_table$points)
}
v_find_form <- Vectorize(find_form)
```

There is then the head to head feature. This feature uses the find_head function to calculate the results of the last 5 games between Everton and the opposition. It collects there last 5 games and awards 3 points for every Everton win and -3 points for every Everton loss.) for a draw. The return value is the some of the points.

```r
find_head <- function(home,away,game) {
  f_table <- epl_results_tbl %>%
    filter(gameID < game)%>%
    filter(home_team == home | away_team ==home )%>%
    filter(home_team == away | away_team ==away )%>%
    mutate(indic = row_number())
  f_table <- f_table %>%
    filter(indic > nrow(f_table)-5)%>%
    mutate(points = ifelse(result=="H",3,ifelse(result=="D",0,-3)))
  sum(f_table$points)
}
v_find_head <- Vectorize(find_head)
```

I then get my data from the results.csv file and append my features onto the table. I then add an extra column for the at_home feature. The dataset already states which club is at home so my column has a value of 1 or 0 to indicate if Everton are at home or not. This format makes it easier to use this feature.

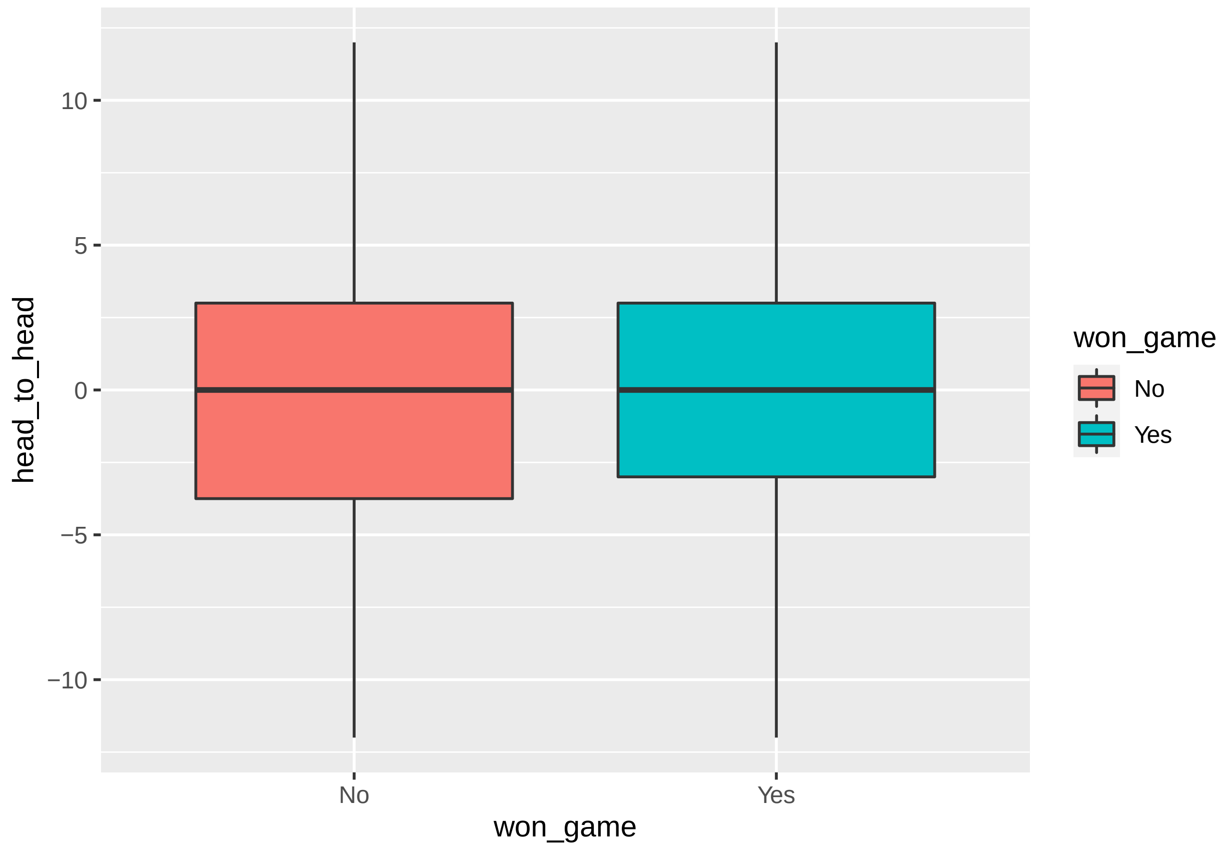
```r
epl_results_tbl <- epl_results_tbl %>%
  mutate(home_team_form = v_find_form(home_team,gameID))%>%
  mutate(away_team_form = v_find_form(away_team,gameID))%>%
  mutate(head_to_head = v_find_head(home_team,away_team,gameID))
```

To make the head to head and form features more accurate, I filtered out results from seasons before 2013-14 to reduce the chances of teams having incomplete form or head to head data(not having up to 5 matches).

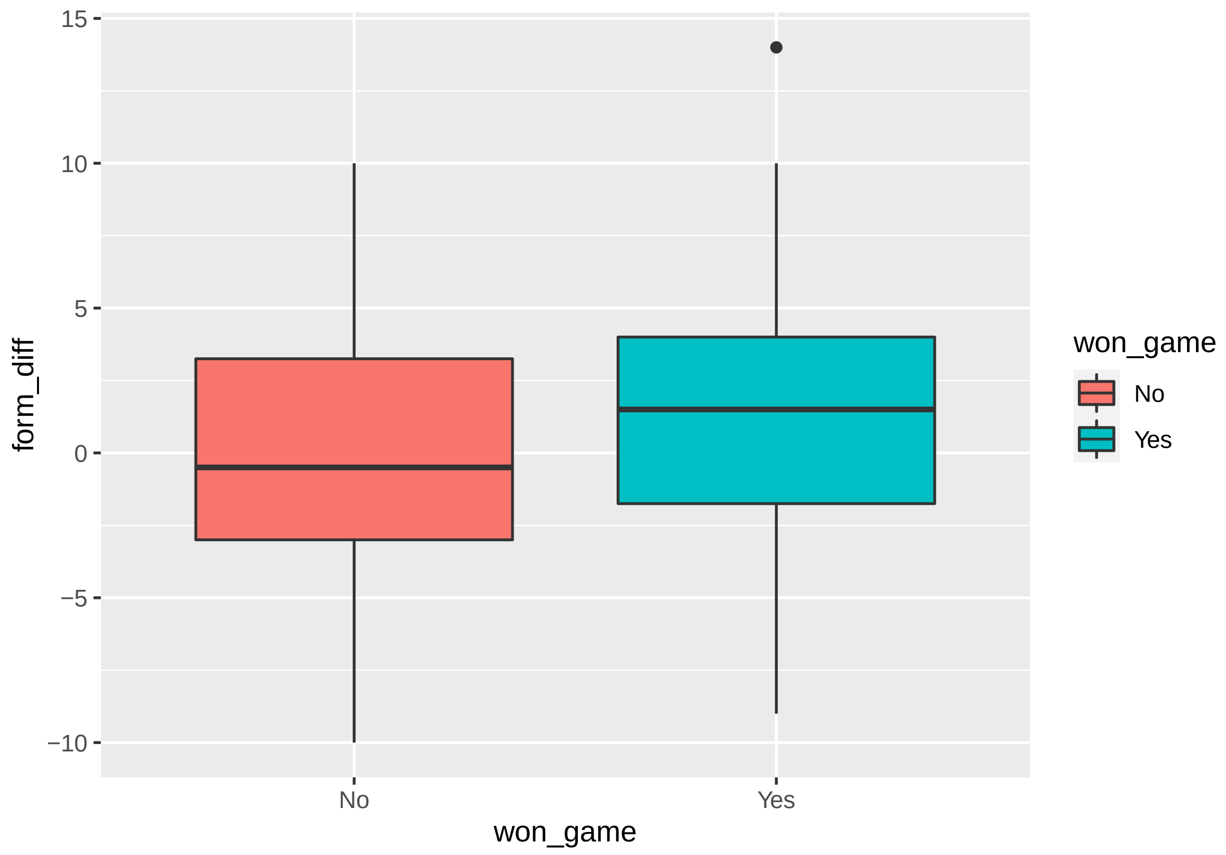
```r
everton_tbl <- epl_results_tbl %>%
  filter(home_team=="Everton" | away_team == "Everton" )%>%
  filter(gameID > 2654)%>%
  mutate(won_game = ifelse(winner == "Everton" ,"Yes","No"))%>%
  mutate(current_form = ifelse(home_team=="Everton",home_team_form,away_team_form))%>%
  mutate(opp_form = ifelse(home_team=="Everton",away_team_form,home_team_form))%>%
  mutate(head_to_head = ifelse(home_team=="Everton",head_to_head,0-head_to_head))%>%
  mutate(form_diff = current_form - opp_form)%>%
  mutate(at_home = ifelse(home_team=="Everton",1,0))
```

Here are boxplots of how the different features affect the outcome of games. From these we can infer that head_to_head only varies slightly with the outcome of games as the boxplots for Yes and No are similar other than No results being slightly more inclined to have a worse head to head for Everton. There is a relatively high correlation between Everton having better form than the opposing side and winning, and there is a strong correlation between Everton being at home and them winning games.

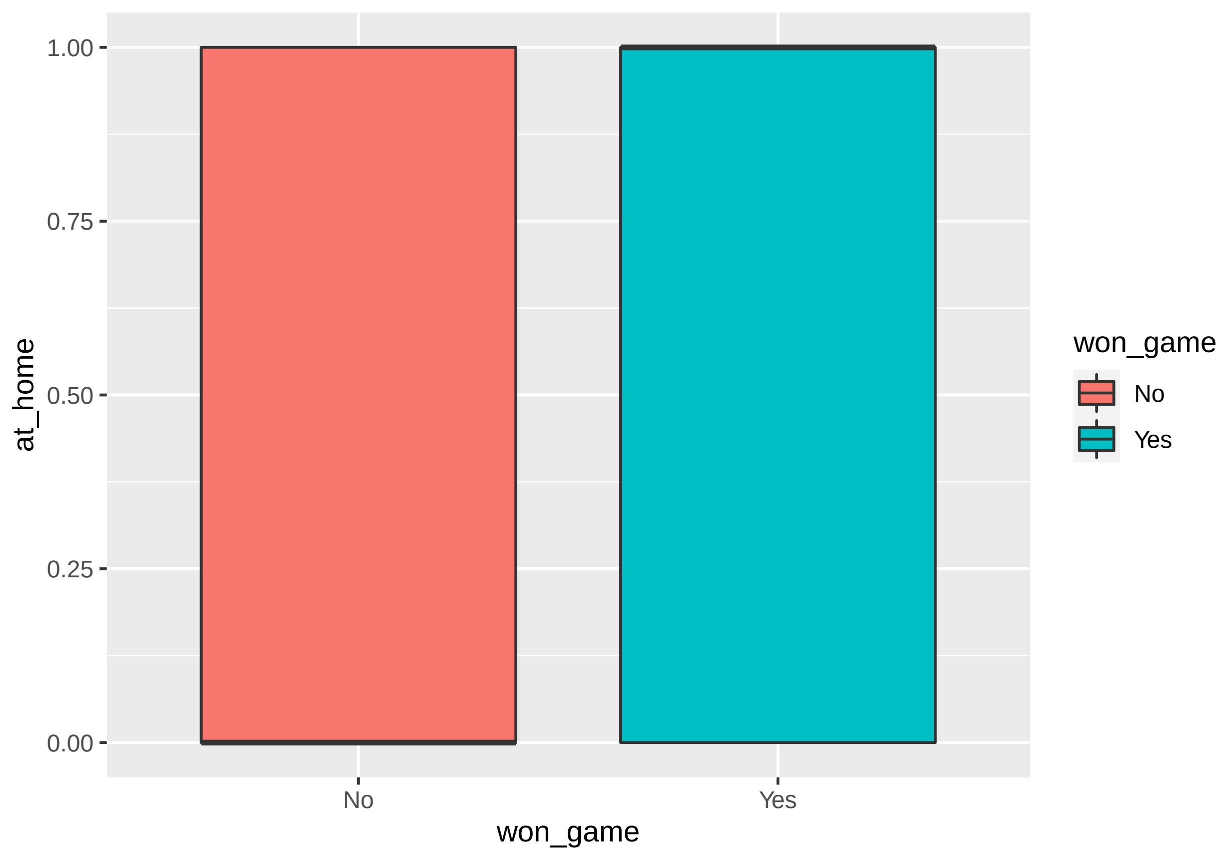
```r
ggplot(everton_tbl, aes(x=won_game, y=head_to_head, fill = won_game)) +
  geom_boxplot()
```



```
ggplot(everton_tbl, aes(x=won_game, y=form_diff, fill = won_game)) +
  geom_boxplot()
```



```
ggplot(everton_tbl, aes(x=won_game, y=at_home, fill = won_game)) +
  geom_boxplot()
```



#Logistic Regression Model

Now we split the data into training and testing datasets

```
set.seed(123456789)
results.split <- initial_split(everton_tbl, prop=0.8)
results.train <- training(results.split)
results.test <- testing(results.split)
```

For our first model, we made use of a logistic regression as this model finds the conditional probability of an event occuring given that certain variables are in place.

```
logit_model <- logistic_reg() %>%
  set_engine("glm") %>%
  set_mode("classification")

result_recipe <-
  recipe(won_game ~ head_to_head+form_diff+at_home, data=results.train)

logit_wflow <- workflow() %>%
  add_recipe(result_recipe) %>%
  add_model(logit_model)

logit_fit <- fit(logit_wflow, results.train)
```

To evaluate this model, we calculated the accuracy, and confusion matrix. As shown by the matrix, the model tends to predict that Everton will lose when they win(over half the time). This means that the model is more inclined to predict that Everton will not win. However, it has an ccuracy of 63 percent so correctly predicts the outcome more often than not. THis is better than blindly picking an option which would have a

50 percent accuracy.

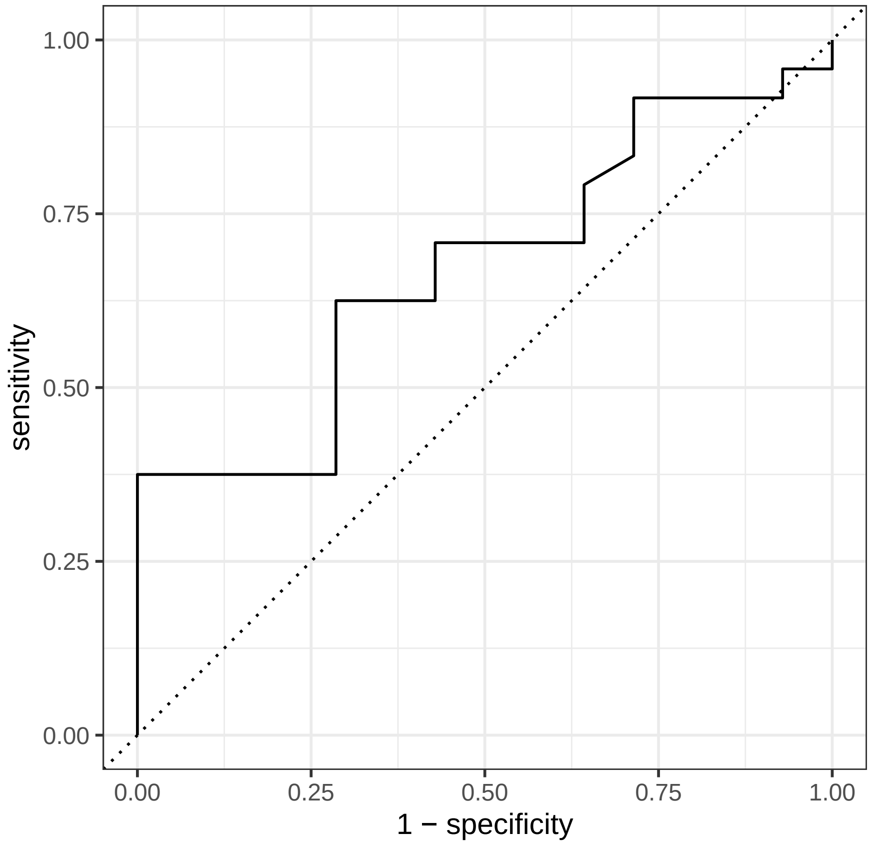
```r
new.results.test <- augment(logit_fit, results.test)
new.results.test$won_game <- as.factor(new.results.test$won_game)

new.results.test %>%
  conf_mat(truth = won_game, estimate = .pred_class)
```

```
##           Truth
## Prediction No Yes
##        No  20  10
##        Yes  4   4
```

```r
new.results.test %>%
  accuracy(truth = won_game, estimate = .pred_class)
```

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.632
```

```r
model_curve <- roc_curve(new.results.test, won_game, .pred_No)
autoplot(model_curve)
```



```r
roc_auc(new.results.test, won_game, .pred_No)
```

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.671
```

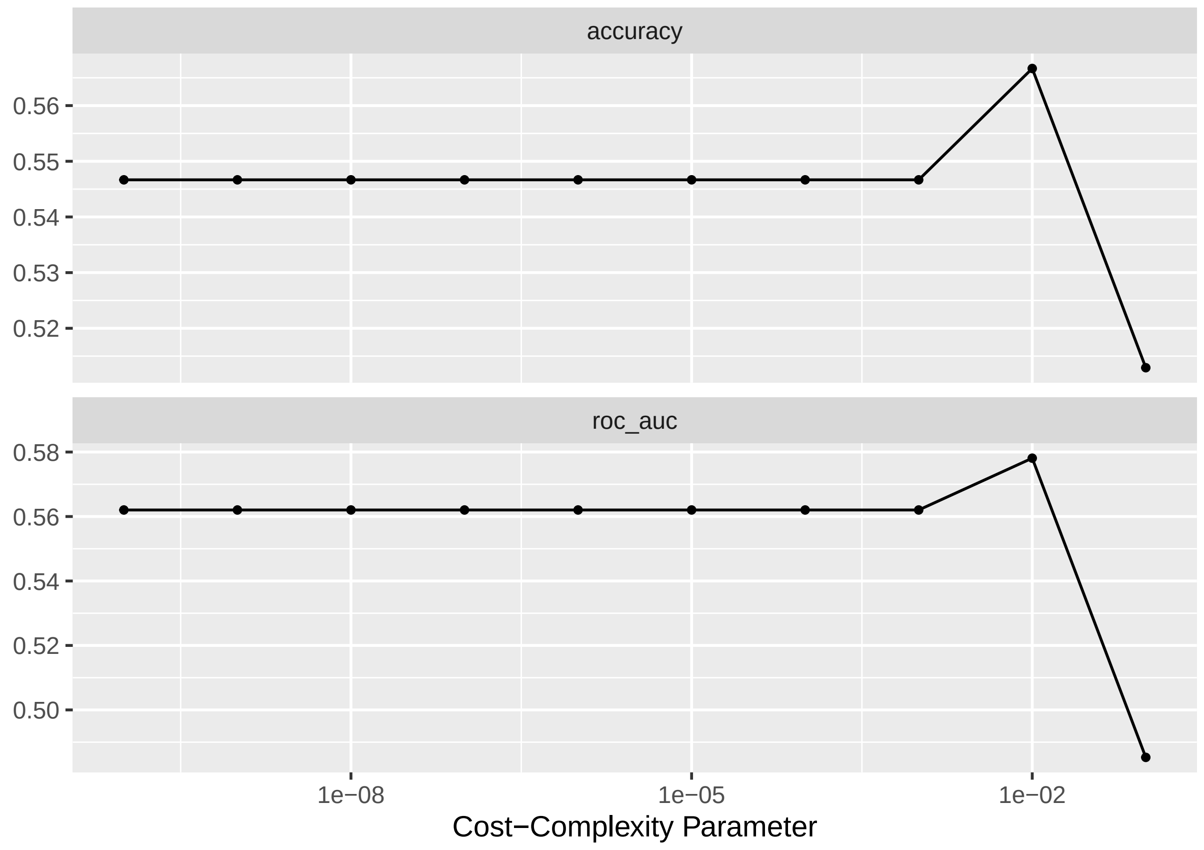
#Decision Tree Model

Next we utilize a decision tree to predict the outcome of events.

```r
decision_model <-
  decision_tree(cost_complexity = tune()) %>%
  set_mode("classification") %>%
  set_engine("rpart")

decision_recipe <-
  recipe(won_game ~
           head_to_head+
           form_diff+
           at_home,
         data=results.train)
decision_wflow<-workflow()%>%
  add_model(decision_model)%>%
  add_recipe(decision_recipe)
```

We will then use 10 fold cross validation to tune the cost complexity

```r
decision_folds<-vfold_cv(results.train,v=10)
decision_grid<-grid_regular(cost_complexity(),levels=10)
decision_res<-
  tune_grid(
    decision_wflow,
    resamples=decision_folds,
    grid=decision_grid)
autoplot(decision_res)
```



Results appear to be best at about 1e-02 cost complexity based on the graph. And the table below confirms it. As accuracy is now at 65.2 percent. This is a little bit better than the accuracy obtained from the logistic regression but not by too much.

```
(best.penalty<-select_by_one_std_err(decision_res, metric="accuracy",-cost_complexity))
```

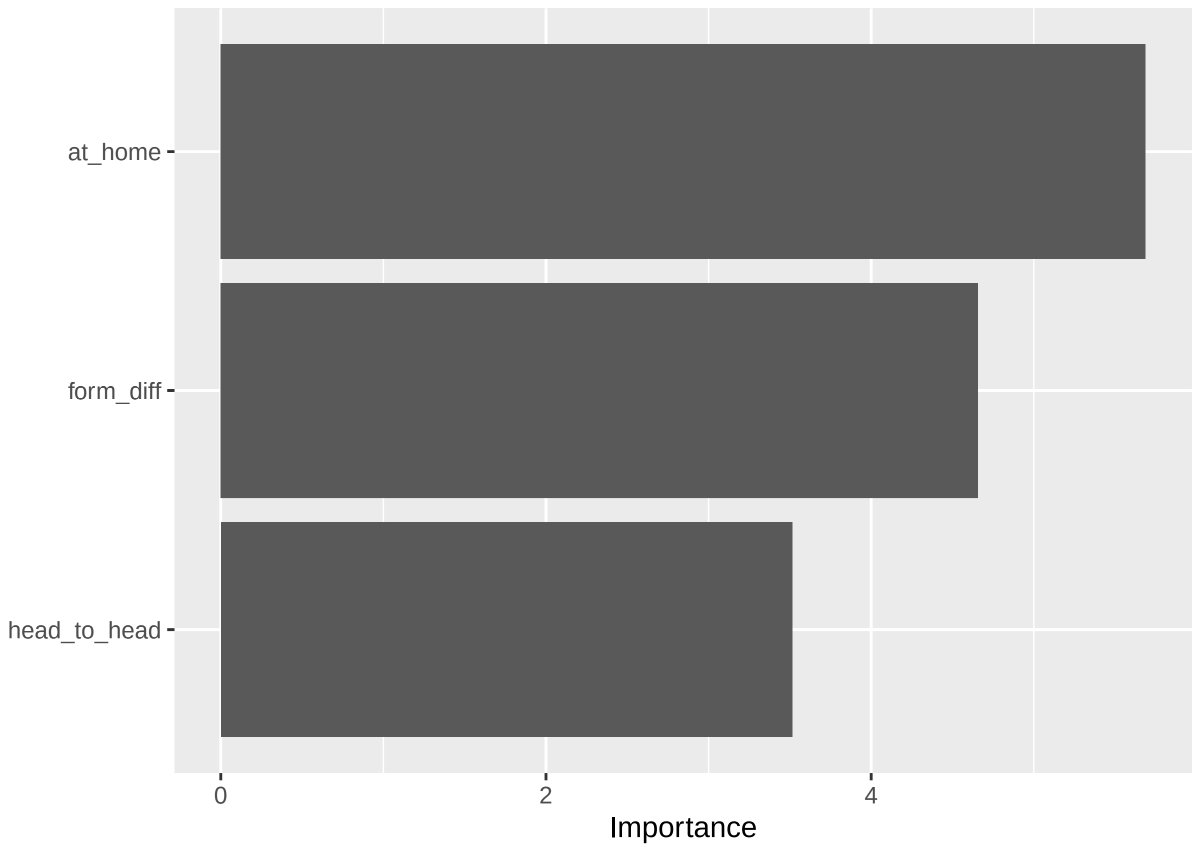
```
## # A tibble: 1 x 9
##   cost_complexity .metric  .estimator  mean     n std_err .config   .best .bound
##             <dbl> <chr>    <chr>      <dbl> <int>   <dbl> <chr>     <dbl>  <dbl>
## 1            0.01 accuracy binary     0.567    10  0.0263 Preproce~ 0.567  0.540
```

Finally, we can then use this to see which feature is mot important in predicting the outcome of Everton games. It appears that playing at home is the most important feature in this prediction. It is not surprising that this feature influences the outcome of games as it is a known fact that soccer teams perform better when playing at their home stadium. The extent to which this benefits a team varies but it appears to help Everton a lot. Head to head was the next most important and the current form of both teams seems to be the least. This could be because the difficulty of teams that Everton faces varies greatly. Meaning that they may have to play much better to win one game compared to another.

```
decision_final_wf<-finalize_workflow(decision_wflow, best.penalty)
decision_final_fit<-fit(decision_final_wf,results.train)
decision_final_res<-last_fit(decision_final_wf, results.split)
collect_metrics(decision_final_res)
```

```
## # A tibble: 2 x 4
##   .metric  .estimator .estimate .config
##   <chr>    <chr>          <dbl> <chr>
## 1 accuracy binary         0.605 Preprocessor1_Model1
## 2 roc_auc  binary         0.598 Preprocessor1_Model1
```

```
decision_final_fit%>%
  extract_fit_engine()%>%
  vip()
```



#Conclusion

The objective of this project was to create models to predict the outcome of soccer matches. Predicting the outcome of events is never easy and soccer is a sport that has a lot of variables at play in deciding which team will win. This is not an easy task even for the sports most recognized pundits. I made use of the epl statistiscs and results found from the source provided. Using the data provided by the dataset, I created features that I felt would be useful in predicting the outcome of games based on my experience watching soccer. These included whether the team was at home, their current form and the head to head between the two teams. I first created a logistic regression model which yielded an accuracy of 63 percent. This was a considerable amount higher than a blind prediction possibility of 50 percent. Then, I used a decision tree model, yielding a classification accuracy of 65 percent.

Some ways that I could improve results in the future would be to come up with more features to further improve prediction. One feature I could have included was the current position of the team although this would have required a great deal of processing. Another way would be to use the random forest model which could have lead to slightly better results. All in all, an accuracy of 65 percent is roughly 2 in every 3 predictions so this would be relatively useful in predicting the outcome of matches.

#Dataset Source

https://www.kaggle.com/datasets/zaeemnalla/premier-league