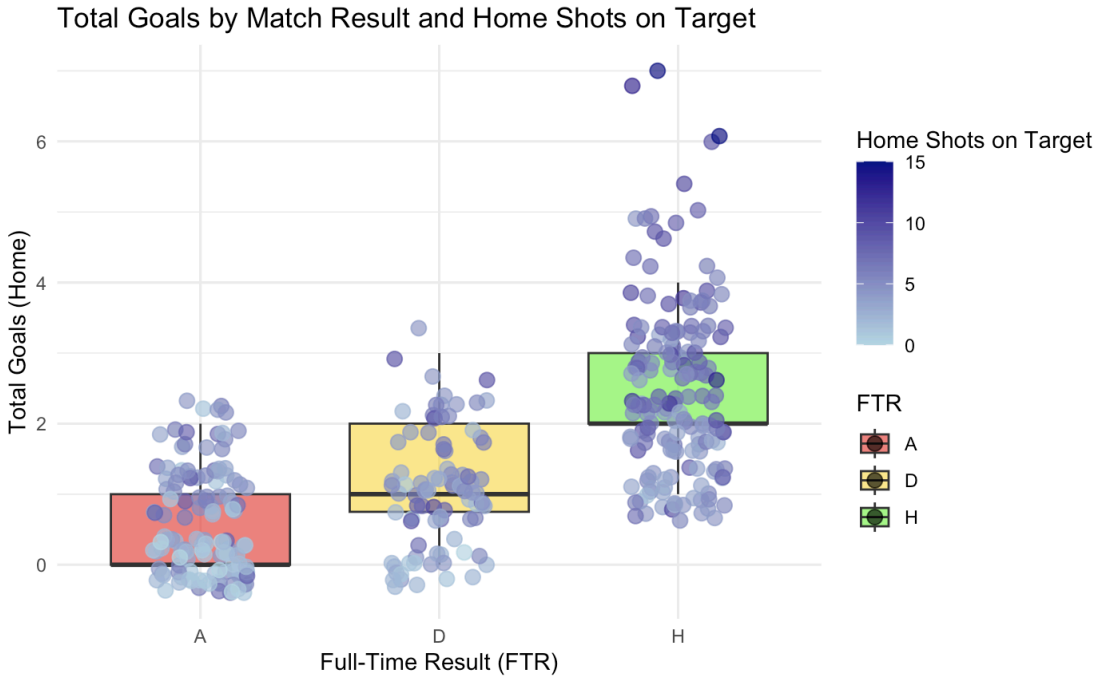
Research Goals

Playing in a competitive soccer league such as the Premier League, every advantage you can give your club to get an edge on the competition is crucial for program success. Which factors contribute to that edge by making a win more likely compared to a loss or a draw? The goal of this machine learning algorithm is to create a classification model that accurately predicts whether the home team will win, lose, or draw based on a variety of predictors. Most of the available predictors are in-game statistics, focusing this project on which features are associated with a win as the game develops, rather than which features predict a win before the game.

Data

The Premier League data we decided to use gives us information on all 380 matches from the English Premier League in the 2021-22 season. The information includes the date the match is played, the teams playing, and other general statistics about the match such as how many goals were scored by each team, their shots on net, and the amount of fouls they committed. The data was collected from the Premier League official website, cleaned prior to our usage, and found on Kaggle. It records important information about each match, which can be used to create predictive models.



This visualization shows how the number of goals scored by the home team varies by match result, with the color of each point representing the number of home shots on target. It

reveals that home wins generally correspond with higher goals and more shots on target while draw and away wins tend to have both lower home team goals and shots on target.

Model Building

We use a random forest with 500 trees, a minimum leaf size of 2, and the number of randomly selected predictors used for each split determined by the square root of the total number of predictors ($18^{1/2} \sim 4$). We chose 18 predictors from the initial 21, excluding the categorical predictors with so many unique values that they lacked actual meaning and prediction power: the date and referee name. We also excluded the “cheating” predictors that will not be useful for soccer managers because they don’t offer much new insight This way, managers can focus more on predictors that aren’t just goal scores: full-time home team goals, full-time away team goals, half-time home team goals, half time away goals, and half time result. The large number of trees in our forest ensures low variability, and the small minimum leaf size ensures low bias. The combination also does not result in super intense computation costs.

Our random forest model fits 500 unpruned trees each with resampled data points from our whole dataset, which allows us to average across all trees and predict the most common results, thus lowering variance from dataset to dataset while keeping the low bias of unpruned trees. The random forest model also uses predictors from a randomly selected predictor subset of size 4 in each split of each tree, which results in lower computational intensity, especially compared to bagging which considers all predictors in splits. Moreover, the random forest model is an eager learner—once it forms the forest by using training data, we do not need to re-fit the model again and we can use the forest built to predict new data. As we want to predict whether a home team will win, tie, or lose their game, our model uses 18 predictors with cheating and non meaningful data removed.

Besides the random forest model, we also built some other classification models. With the tree model having low bias but higher variance compared to our random forest, we decided it may be unstable and possibly overfit when we use it to evaluate the new dataset, although it has more interpretability since it can be plotted. The Bagging model is similar to our random forest value only using all predictors in each split rather than the subset of size 4 predictors. This results in slightly lower OOB prediction error (39% to 36%), but much more computational intensity. In addition, since the bagging uses all predictors in the splitting process, all the trees in the forest will be similar to each other as well as trees that use all data points, which will result in higher variance compared to our random forest. Thus, we conclude that it is not worth it to slightly increase accuracy with much higher computational cost and higher variance. We also fit the KNN classification model, which gives slightly better accuracy (62% compared to 61%). However, such small differences can not offset the fact that KNN is a lazy learner, which needs to re-calculate the distance and identify the nearest neighborhood every time to evaluate new data, leading to higher computational intensity. Combining all the facts we list above, we ended up choosing the random forest as our final model.

Implementation

See code

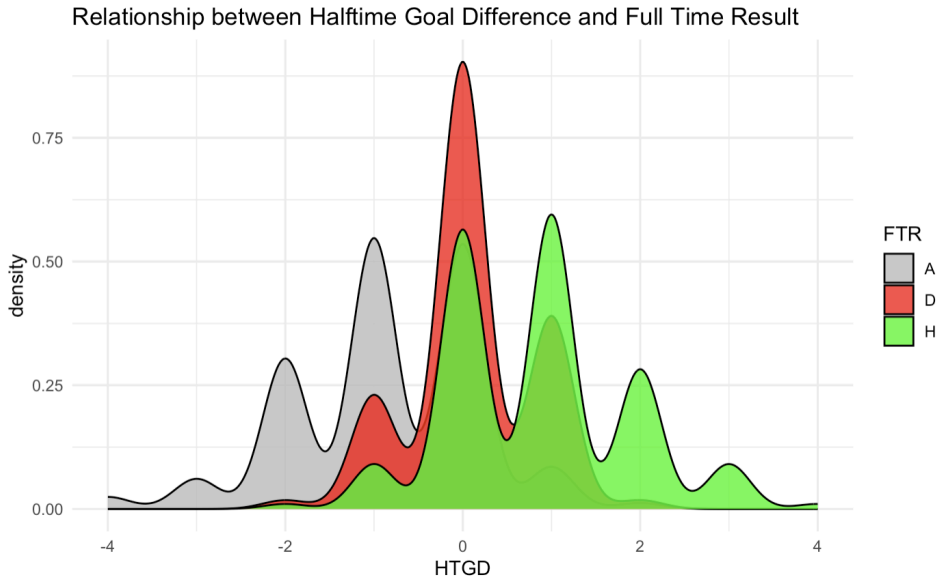
Model Evaluation

Our final random forest model has an “out of bag” (OOB) prediction error of 38.95%. In our random forest, a given tree’s out of bag data points are not used to train it, making them natural test cases for that tree. This means that given our selection of predictors, we expect our model to correctly classify 61.05% of games as home wins, away wins, or draws for new data. We can compare this to the “no information rate”, the accuracy rate we would get if we simply predicted the dominant category for all games. Our no information rate would predict a home win for every game, since that’s the most common game result. It would correctly classify 42.89% of games. It is clear that our random forest model is a significant improvement on the no information rate.

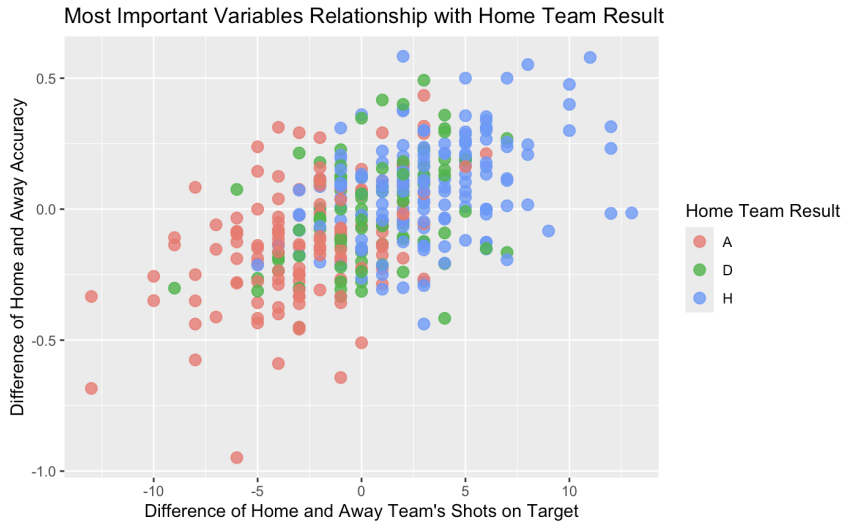
	true		
predicted	A	D	H
A	84	27	23
D	18	20	12
H	27	41	128

Out of Bag (OOB) Confusion Matrix
Note: A = away win, D = draw, H = home win

In order to evaluate within class accuracy, we built the OOB confusion matrix seen above. As mentioned earlier, using OOB data allows us to evaluate how our model performs on new data. For each true game result class in this new data, our OOB confusion matrix tells us how our game result predictions break down. We can use these numbers to calculate our model’s accuracy for that class. For example, we can see that our model correctly predicted 84 out of 129 (84 + 18 + 27) true away wins for a within class accuracy of 65.12%. We find a within class accuracy of 22.73% for true draws and 78.53% for true home wins. In order to maximize overall accuracy, our model clearly prioritizes within class accuracy for classes that are more frequent in the data.



This visualization shows why we chose to exclude halftime data from our analysis. There's a strong correlation between halftime scores and full-time outcomes, meaning the halftime result is often a direct reflection of the final result. For a coach or manager, relying on halftime scores to predict the final outcome isn't particularly useful because it doesn't offer new insight. Instead, it's more valuable to focus on other predictors that can explain or influence match outcomes beyond what the halftime score already indicates.



After wrangling our data, the four most important predictors left were Home and Away team shot accuracy and Home and Away team shots on target. The above visualization shows the relationship between the home team's result and these predictors. There is a clear trend that if the home team has more shots on target and a higher shot accuracy, the game is more likely

to result in a win for them. On the contrary if the away team has more shots on target and a higher shot accuracy, the home team is more likely to lose.