589 Final Project

Anshul Sadh-Gauri and Ali Ayaan Rizvi

May 2025

1 Introduction

Predict outcomes of soccer sports games in the Premier league, specifically money line bets. Goal is to predict whether outcome is a win, draw, or loss. The 2 models we decided to use are neural networks and random forest. We picked neural networks since we thought there would be complex patterns in the noisy sports data that a neural network would be able to capture. Next, we picked a random forest since we thought it would be able to make good, understandable decisions by combining many trees and thus make accurate predictions.

1.1 Feature Selection

The feature selection procedure consisted of the following steps:

1. Eliminate all features with zero variance.
2. Run ANOVA to select the top 1 000 features.
3. Apply ElasticNet and select the most valuable features.

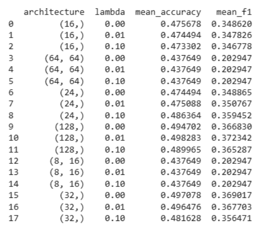
Since the dataset comprises 1 000 features for the home team and 1 000 features for the away team, only the home-team features (the first 1 000) were subjected to this selection procedure. Once the most valuable home-team features were identified, the same feature indices were used for the away team so that both teams have an equal number of features.

The final list of features is:

```
['average_attendance_home_diff', 'leaguePosition_away_diff',
 'over25CornersAgainstPercentage_home_diff',
 'leadingAtHTPercentage_away_diff', 'seasonGoalDifference_overall_diff',
 'goals_scored_min_46_to_60_home_diff', 'losePercentage_away_diff',
 'over45OffsidesPercentage_home_diff',
 'over85CornersForPercentage_away_diff',
 'over35CornersAgainstPercentage_overall_diff',
 'dangerous_attacks_avg_away_diff',
 'over75CornersForPercentage_overall_diff',
 'seasonOver25PercentageHT_overall_diff',
 'over65CornersAgainst_home_diff',
 'over35OffsidesPercentage_overall_diff',
 'corners_fh_over6_percentage_away_diff', 'GoalDifferenceHT_away_diff',
 'over85CornersForPercentage_overall_diff',
 'over65CornersAgainstPercentage_home_diff',
 'offsidesTeamOver55_away_diff',
 'over65CornersAgainstPercentage_overall_diff',
 'goals_conceded_min_41_to_50_diff',
 'over45OffsidesTeamPercentage_overall_diff',
 'over35CornersAgainstPercentage_home_diff', 'odds_home_win',
 'odds_draw', 'odds_away_win', 'elo_home', 'elo_away',
 'home_AFC Bournemouth', 'home_Arsenal FC', 'home_Aston Villa FC',
 'home_Brentford FC', 'home_Brighton & Hove Albion FC',
 'home_Burnley FC', 'home_Cardiff City FC', 'home_Chelsea FC',
 'home_Crystal Palace FC', 'home_Everton FC', 'home_Fulham FC',
 'home_Huddersfield Town FC', 'home_Leeds United FC',
 'home_Leicester City FC', 'home_Liverpool FC',
 'home_Luton Town FC', 'home_Manchester City FC',
 'home_Manchester United FC', 'home_Newcastle United FC',
 'home_Norwich City FC', 'home_Nottingham Forest FC',
 'home_Sheffield United FC', 'home_Southampton FC',
 'home_Tottenham Hotspur FC', 'home_Watford FC',
 'home_West Bromwich Albion FC', 'home_West Ham United FC',
 'home_Wolverhampton Wanderers FC', 'away_AFC Bournemouth',
 'away_Arsenal FC', 'away_Aston Villa FC', 'away_Brentford FC',
 'away_Brighton & Hove Albion FC', 'away_Burnley FC',
 'away_Cardiff City FC', 'away_Chelsea FC', 'away_Crystal Palace FC',
 'away_Everton FC', 'away_Fulham FC', 'away_Huddersfield Town FC',
 'away_Leeds United FC', 'away_Leicester City FC', 'away_Liverpool FC',
 'away_Luton Town FC', 'away_Manchester City FC',
 'away_Manchester United FC', 'away_Newcastle United FC',
 'away_Norwich City FC', 'away_Nottingham Forest FC',
 'away_Sheffield United FC', 'away_Southampton FC',
 'away_Tottenham Hotspur FC', 'away_Watford FC',
 'away_West Bromwich Albion FC', 'away_West Ham United FC',
 'away_Wolverhampton Wanderers FC']
```

1.2 Neural Network

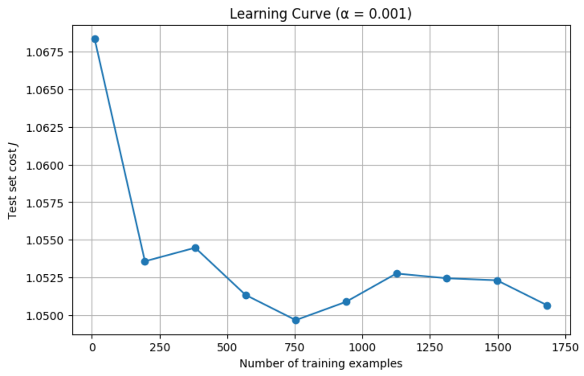
In picture below, mean f1 and accuracy are from cross val.



```
    architecture  lambda  mean_accuracy   mean_f1
0          (16,)    0.00       0.475678  0.348620
1          (16,)    0.01       0.474494  0.347826
2          (16,)    0.10       0.473302  0.346778
3       (64, 64)    0.00       0.437649  0.202947
4       (64, 64)    0.01       0.437649  0.202947
5       (64, 64)    0.10       0.437649  0.202947
6          (24,)    0.00       0.474494  0.348865
7          (24,)    0.01       0.475088  0.350767
8          (24,)    0.10       0.486364  0.359452
9         (128,)    0.00       0.494702  0.366830
10        (128,)    0.01       0.498283  0.372342
11        (128,)    0.10       0.489965  0.365287
12       (8, 16)    0.00       0.437649  0.202947
13       (8, 16)    0.01       0.437649  0.202947
14       (8, 16)    0.10       0.437649  0.202947
15         (32,)    0.00       0.497078  0.369017
16         (32,)    0.01       0.496476  0.367703
17         (32,)    0.10       0.481628  0.356471
```

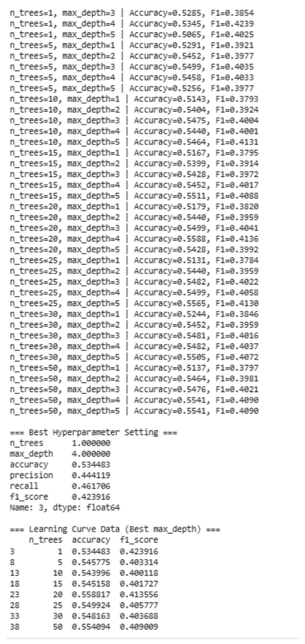
The model with 1 hidden layer and 128 neurons performed the best.

The following graph is how the cost curve changes as number of samples increases.



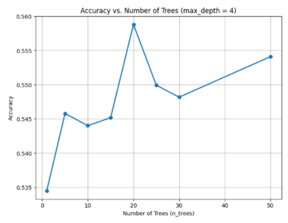
1.3 Random Forest

Following table shows results from different number of trees and max depth of random forest.

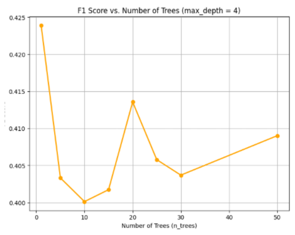


```
n_trees=1, max_depth=3 | Accuracy=0.5285, F1=0.3854
n_trees=1, max_depth=4 | Accuracy=0.5345, F1=0.4239
n_trees=1, max_depth=5 | Accuracy=0.5065, F1=0.4025
n_trees=5, max_depth=1 | Accuracy=0.5291, F1=0.3921
n_trees=5, max_depth=2 | Accuracy=0.5452, F1=0.3977
n_trees=5, max_depth=3 | Accuracy=0.5499, F1=0.4035
n_trees=5, max_depth=4 | Accuracy=0.5458, F1=0.4033
n_trees=5, max_depth=5 | Accuracy=0.5256, F1=0.3977
n_trees=10, max_depth=1 | Accuracy=0.5143, F1=0.3793
n_trees=10, max_depth=2 | Accuracy=0.5404, F1=0.3924
n_trees=10, max_depth=3 | Accuracy=0.5475, F1=0.4004
n_trees=10, max_depth=4 | Accuracy=0.5440, F1=0.4001
n_trees=10, max_depth=5 | Accuracy=0.5464, F1=0.4131
n_trees=15, max_depth=1 | Accuracy=0.5167, F1=0.3795
n_trees=15, max_depth=2 | Accuracy=0.5399, F1=0.3914
n_trees=15, max_depth=3 | Accuracy=0.5428, F1=0.3972
n_trees=15, max_depth=4 | Accuracy=0.5452, F1=0.4017
n_trees=15, max_depth=5 | Accuracy=0.5511, F1=0.4088
n_trees=20, max_depth=1 | Accuracy=0.5179, F1=0.3820
n_trees=20, max_depth=2 | Accuracy=0.5440, F1=0.3959
n_trees=20, max_depth=3 | Accuracy=0.5499, F1=0.4041
n_trees=20, max_depth=4 | Accuracy=0.5588, F1=0.4136
n_trees=20, max_depth=5 | Accuracy=0.5428, F1=0.3992
n_trees=25, max_depth=1 | Accuracy=0.5131, F1=0.3784
n_trees=25, max_depth=2 | Accuracy=0.5440, F1=0.3959
n_trees=25, max_depth=3 | Accuracy=0.5482, F1=0.4022
n_trees=25, max_depth=4 | Accuracy=0.5499, F1=0.4058
n_trees=25, max_depth=5 | Accuracy=0.5565, F1=0.4130
n_trees=30, max_depth=1 | Accuracy=0.5244, F1=0.3846
n_trees=30, max_depth=2 | Accuracy=0.5452, F1=0.3959
n_trees=30, max_depth=3 | Accuracy=0.5481, F1=0.4016
n_trees=30, max_depth=4 | Accuracy=0.5482, F1=0.4037
n_trees=30, max_depth=5 | Accuracy=0.5505, F1=0.4072
n_trees=50, max_depth=1 | Accuracy=0.5137, F1=0.3797
n_trees=50, max_depth=2 | Accuracy=0.5464, F1=0.3981
n_trees=50, max_depth=3 | Accuracy=0.5476, F1=0.4021
n_trees=50, max_depth=4 | Accuracy=0.5541, F1=0.4090
n_trees=50, max_depth=5 | Accuracy=0.5541, F1=0.4090

=== Best Hyperparameter Setting ===
n_trees      1.000000
max_depth    4.000000
accuracy     0.534483
precision    0.444119
recall       0.461706
f1_score     0.423916
Name: 3, dtype: float64

=== Learning Curve Data (Best max_depth) ===
    n_trees  accuracy  f1_score
3         1  0.534483  0.423916
8         5  0.545775  0.403314
13       10  0.543996  0.400118
18       15  0.545158  0.401727
23       20  0.558817  0.413556
28       25  0.549924  0.405777
33       30  0.548163  0.403688
38       50  0.554094  0.409009
```

A simple random forest with 1 tree and max depth 4 performed the best. The next chart shows the random forest accuracy vs number of trees.



This chart shows F1 score vs number of trees.



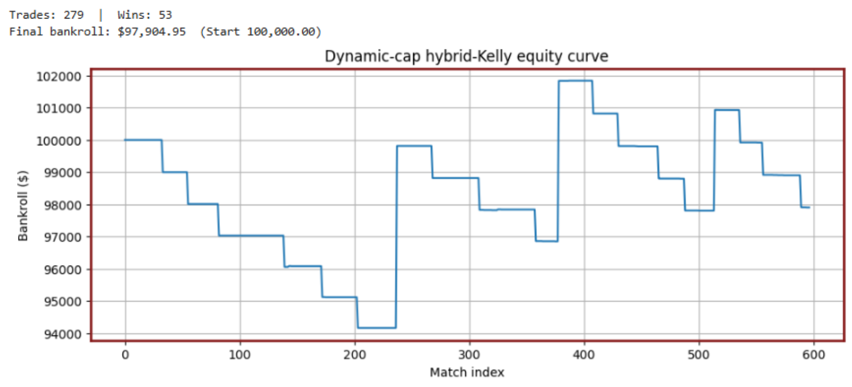
1.4 Analysis

The neural network performed best with just one hidden layer of 128 neurons. This could be because adding another hidden layer could lead to overfitting.

It had accuracy of .498 and f1 score of .37. These results aren't the best but at the same time predicting match outcomes isn't an easy task either. The random forest performed better with achieving an accuracy score of .54 and f1 score of .42. The best hyperparameters consisted of a tree depth of 1 and max depth of 4. This is a very simple tree, Perhaps this algorithm performed better because it was simpler whereas the neural network just ended up memorizing the training data. Currently max epochs was used for the neural network but if that is changed to stopping once the validation loss stops decreasing for over 5 epochs can fix that issue.

1.5 Portfolio Optimization

The following graph illustrates the results of applying a Kelly-Criterion–based portfolio optimization to our sports bets. Unfortunately, the strategy produced an overall loss, highlighting just how volatile and risky sports betting can be. Notably, the neural network model performed even worse than the random forest.



```
Trades: 279  |  Wins: 53
Final bankroll: $97,904.95  (Start 100,000.00)
```

1.6 Summarized Results

	Dataset 1	
	Accuracy	F1-Score
Algorithm 1	.498	.372
Algorithm 2	.535	.424