Team Kay Model Write-up

Jonggoo Kang, Ellie Jeon, Sungil Kim

In order to build the best model to predict the probability of winning in a match per team, we first evaluated the classifiers by tuning parameters to find hyperparameters after pre-processing. Then we attempted to find the best model with the highest accuracy by comparing the classifiers. The top three models were utilizing Logistic Regression, AdaBoost, and Gradient Boosting. Logistic regression was picked out of the three because the binary response 0 and 1 utilized the linear relationships in the model to explain winning and losing. We classified 0,1 binary target variable 'win?' to demonstrate what features explains each class the most. This also led us to realize that the columns that started with 'Opp' was in fact the opponent's statistics during the game ('gameid'). All columns with 'Opp' was removed to begin our feature selection methods. The top absolute correlation chart illustrated that some variables had extremely high correlation coefficients. Therefore, variables with high correlations were removed. Then the variables that were left including our response 'Win?' were:

```
'MP', '2P%', '3P%', 'FT%', 'ORB', 'DRB', 'AST', 'STL', 'BLK', 'TOV', 'PF',
'PTS', 'Win?'
```

The first logistic model was built with all variables above and 'Win?' as the response. After the model evaluation, the variable 'PTS' (p-value above 0.05) was dropped from the model to create the final model. The formula of the final logistic regression was the following:

```
logit[win=1] = -6.6799 + (-0.0417*MP) + (10.3005*2P%) + (8.9374*3P%) +
(2.4784*FT%) + (0.1241*ORB) + (0.2638*DRB) + (0.0183*AST) + (0.2543*STL) +
(0.1355*BLK) + (-0.1587*TOV) + (-0.1260*PF)
```

In the final model, the most significant variable per unit change is '2p%' given that all other variables stay constant. All variables have a linear relationship with logit[win=1]. Every 1 unit increase in '2p%' results in 10.3005 increase in log odds of winning (1 or 0) when other variables remain constant. This suggests that the estimated odds of winning multiply by e^10.3005. To test the model, the Team 17 and Team 18 dataset were combined to generate a master data with maximum observation counts. Then with this model, I inputted the desired team names with the master data to feed off of in the Predicted_Winner() function (located in the Final.ipynb file). This illustrates that ever matchup was treated differently. The win or loss was selected by comparing the assigned possibilities of winning in the team records. Each team record had the form of the array [losing probability, winning probability]. This function did not account for a team with a lower win probability to win against a team of higher probability. Higher winning probability always won the match against the lower. Inputting the team names in the last line in the Final.ipynb will split out the win probabilities in each matchup.

In making of all the models included, outside sources were not used. All models/functions generated were by only our team members. Among the datasets provided for this Bracketlytics Challenge, our team only utilized TeamStats17 and TeamStats18. Our model had extremely low p-values (almost all 0.00). We believe that there is overfitting in our final model. We also believe certain variables such as ‘Opp Pts’ might have been important. However, due to time constraints, we decided to go with the model to predict winners of each match. The scripts of code provided includes the final model, as well as other models we attempted with in the beginning process. However, all predictions for matches were generated by the final model using the Predicted_Winner(model, teamdata, teamname_A, teamname_B, prob_arr = False) function.