League of Legends Competitive Match Data

- **See the main project notebook for instructions to be sure you satisfy the rubric!**
- See Project 03 for information on the dataset.
- A few example prediction questions to pursue are listed below. However, don't limit yourself to them!
  - Predict if a team will win or lose a game.
  - Predict which role (top-lane, jungle, support, etc.) a player played given their post-game data.
  - Predict how long a game will take before it happens.
  - Predict which team will get the first Baron.

Be careful to justify what information you would know at the "time of prediction" and train your model using only those features.

Summary of Findings

Introduction

In this project, our goal was to predict whether any given team was going to win a League of Legends match after inputting various predictive features. In order to accomplish this, we decided to use a classification model, which would take in the input data and predict whether a specific team was going to win the match or not. This is because the problem we sought to solve was a binary classification problem, or a problem where, given some input data, the classifier would output 1 or 0, essentially returning a predictive class label based on a KNearestNeighbors model. In this case, our target variable, was 'result', a column in the League of Legends dataset that contained a binary variable that was 1 if the team won the match, and 0 if the team was defeated. In addition, we set our evaluation metric to be accuracy, which was calculated by finding the probability of correctly predicting the outcome of the test set matches using a classification model fitted on the training set of matches.

Baseline Model

We decided that, given our extensive background knowledge of the League of Legends video game, the optimal way of choosing predictive features was to parse through all the column names, and determine logically which of them would have the largest impact on match outcome. Ultimately, we decided on including 8 features in our KNearestNeighbors model, which were 'side','dragons','heralds', 'barons', 'towers','vspm','earned gpm', and 'golddiffat15'. After closely scrutinizing this subset of the data, we came across multiple

NaN values, which stemmed from the fact that some of our data had 'partial' under the 'datacompleteness' column of the dataframe. Since we knew that a KNearestNeighbors binary classifier model would not run without data imputation performed on the NaN values of the dataframe, and we were unable to impute those values with satisfactory alternatives in Project 3, we decided that the most reasonable decision would be to filter out all entries with partial datacompleteness in order to increase functionality and help the model run smoothly.

In order to create our predictive model, we first had to input the necessary data into train_test_split, in order to split our data into a training test and a test set for our pipeline. We inputted a 0.25 test size and tried to create our train_test_split. Upon starting implementation, however, we realised that there could be a case where the training set of data includes only wins or losses, which would severely hinder the accuracy of our model, because it would have a disproportionate exposure to match wins or losses. Because of this, we decided to perform train_test_split after grouping by 'gameid' in order to partition the data in pairs, where one included the win for that particular match, and the other included the loss. This would guarantee that the model had exposure to equal numbers of wins and losses, and increase accuracy of binary predictions.

Since the majority of the data that was chosen was quantitative, the only major transformation we implemented within our column transformer was OneHotEncoding, in order to convert the nominal 'side' column (which was either 'Red' or 'Blue') into functional numerical data. We also considered using standard scalar in order to standardise select variables, but after plotting multiple histograms for our personal analysis, we found that 'barons', 'towers', 'dragons', 'heralds', etc. all contained skewed discrete data that did not follow a Gaussian distribution. Therefore, we concluded that it was not appropriate to include standard scalar in our KNearestNeighbors pipeline. Since the rest of our data was quantitative, all we had to do to create a pipeline was OneHotEncode 'side'. We then fit the pipeline on our training set, and ran the predict function on our sklearn pipeline using our test set. Surprisingly, we found that at the default number of neighbours, which was 5, our model correctly predicted around 93% of match outcomes from the training set in which it was used to fit the model. More importantly, on the test set, our model also correctly predicted around 93% of match outcomes, which is reasonably good given that it was a baseline model. Now, we wanted to figure out which pipeline hyperparameter optimized our model accuracy. Using a wide range of neighbours in the pipeline hyperparameter, we found that the accuracy ranged from approximately 90% to 94%, with the highest accuracy being 93.72% at 15 neighbours. In general, there is a possibility of having an even higher accuracy as the number of neighbours is increased beyond the scope of our testing, but the increases after a certain point are very minimal, and we believe 15 neighbours is a good middle point to prevent overfitting the data (with a value that is too low), or underfitting the data (with a value that is too high).

Final Model

In order to improve upon our model, we decided to engineer two new features to include that could potentially improve classification accuracy: 'totalgolddiff' and 'killdiff'. In our initial model, we had included 'golddiffat15', but after more deliberation, we concluded that even after 15 minutes had elapsed, there could still be a reasonable exchange of gold, so a team that had more gold than the other team at 15 minutes might not have more gold at the conclusion of the match. For this reason, we replaced 'golddiffat15' with 'totalgolddiff'. We created a helper function that subtracted one column's total gold from the opposing team's total gold for a given gameid, and created an additional column in the dataframe. We also decided to include 'killdiff', a column that calculated the difference in kills between a team and their opponent. We believe that in a given game, difference in kills is a better feature to predict outcome accuracy when compared with total number of kills. We also realised that one team's deaths was the same as the other team's kills, so subtracting deaths from kills would result in a signed difference of kills.

We decided to keep our current KNearestNeighbors model and use a classifier, because we felt our current model was already doing a good job of predicting match outcome, and improvement of features would be enough to boost our accuracy. Then, we created a pipeline, repeating the same steps as our baseline, but replaced the 'golddiffat15' feature with 'totalgolddiff', and included 'killdiff'. To our astonishment, the accuracy of our model went up, outputting a probability between 97% and 98% on the testing set. This was a substantial improvement from the 92-93% range we were getting on our baseline model, which indicated that our engineered features were important factors in determining match outcome. We also noticed, after running the model using multiple neighbors, that the accuracy was still best when 15 neighbors were included as a hyperparameter.

However, it is important to include that it would be presumptuous to assume that the true accuracy of our model would be 98%. In reality, if we were to include the League of Legends datasets for multiple years, and expand it to include more teams and therefore more data, the accuracy would decrease. In addition, the KNearestNeighbors classifier is not optimal for large datasets, so inputting a larger dataset would likely degrade the performance of the algorithm.

Fairness Evaluation

We decided to evaluate our model for "fairness" based on gamelength of the data, using a permutation test. We determined that the distribution of gamelengths was roughly normal, so the mean would be a reasonably summary statistic, which we could use for our threshold. This threshold would classify a game as either being short or long. We first computed the proportion of teams that were predicted to win in each group of game lengths. Since these two numbers were roughly the same, our model achieves demographic parity.

Next, we checked to see if our model achieved accuracy parity by computing the accuracy in each game length group. Accuracy parity would be the best measure of fairness, given

our dataset, because the impact of false positives and false negatives are similar. In other words, there is no greater negative repercussion of wrongly predicting a match win than wrongly predicting a match loss. We noticed our model more accurately predicted the outcomes of short games when compared with long games, with a difference of about 4%. We wanted to see if this difference was significant enough to conclude that our model might not acheive accuracy parity, so we performed a permutation test. Our null hypothesis was that the classifier's accuracy is the same for both short and long games, and any difference is due to random chance. Our alternative hypothesis was that the classifier's accuracy is higher for shorter games. Then, we performed a permutation test, and graphed the result. We saw that despite the difference in accuracy being only around 4%, the difference in accuracy across the two groups seems to be significant enough to conclude that the model likely does not achieve accuracy parity. Therefore, our model must be improved upon to account for the inherent bias in predictions.

Code

In [1]:
```
import matplotlib.pyplot as plt
import numpy as np
import os
import pandas as pd
import seaborn as sns
%matplotlib inline
%config InlineBackend.figure_format = 'retina'  # Higher resolution figures
```

In [64]:
```
#Imports
from sklearn.preprocessing import OneHotEncoder
from sklearn.pipeline import Pipeline
from sklearn.compose import ColumnTransformer
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split
```

Baseline Model

In [65]:
```
#Dataset we are working with
lol = pd.read_csv('lol.csv', low_memory = False)
lol.head()
```

Out[65]:

	gameid	datacompleteness	url	league	year	split	playoffs	date
0	ESPORTSTMNT01_2690210	complete	NaN	LCK CL	2022	Spring	0	2022-01-10 07:44:08
1	ESPORTSTMNT01_2690210	complete	NaN	LCK CL	2022	Spring	0	2022-01-10 07:44:08
2	ESPORTSTMNT01_2690210	complete	NaN	LCK CL	2022	Spring	0	2022-01-10 07:44:08
3	ESPORTSTMNT01_2690210	complete	NaN	LCK CL	2022	Spring	0	2022-01-10 07:44:08
4	ESPORTSTMNT01_2690210	complete	NaN	LCK CL	2022	Spring	0	2022-01-10 07:44:08

5 rows × 123 columns

In [4]:
```
#We only care about team stats so we filter out dataset to include only rows wi
team_stats = lol[(lol['participantid'] == 100) | (lol['participantid'] == 200)]
team_stats.head()
```

Out[4]:

	gameid	datacompleteness	url	league	year
10	ESPORTSTMNT01_2690210	complete	NaN	LCK CL	2022
11	ESPORTSTMNT01_2690210	complete	NaN	LCK CL	2022
22	ESPORTSTMNT01_2690219	complete	NaN	LCK CL	2022
23	ESPORTSTMNT01_2690219	complete	NaN	LCK CL	2022
34	8401-8401_game_1	partial	https://lpl.qq.com/es/stats.shtml?bmid=8401	LPL	2022

5 rows × 123 columns

In [5]:
```
#Split data into training and test sets
team_stats = team_stats[team_stats['datacompleteness'] == 'complete']

#Get unique gameid
unique_games = team_stats.groupby('gameid').count()[['datacompleteness']]

#Split data into train and testing based on gameid
X = unique_games.drop('datacompleteness', axis = 1)
```

```
y = unique_games['datacompleteness']
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size = 0.25)

#Merge to get data for both teams in that game
train = team_stats.merge(X_train.reset_index(), right_on='gameid', left_on='gar
test = team_stats.merge(X_test.reset_index(), right_on='gameid', left_on='game

#Keep relevant columns for features and result to get final training and testi
X_train = train.drop('result', axis=1)
y_train = train['result']
X_test = test.drop('result', axis=1)
y_test = test['result']
```

In [8]:
```
#Column transformer
preproc = ColumnTransformer(transformers = [('side_ohe', OneHotEncoder(), ['sid

#Create pipeline using k-nearest neighbors
pl = Pipeline([('preprocessor', preproc),('knn',KNeighborsClassifier())])

#Fit on training set
pl.fit(X_train, y_train)
```

Out[8]:
```
Pipeline(steps=[('preprocessor',
                 ColumnTransformer(remainder='passthrough',
                                   transformers=[('side_ohe', OneHotEncoder(),
                                                  ['side'])])),
                ('knn', KNeighborsClassifier())])
```

In [9]:
```
#Test model accuracy on training set
pred = pl.predict(X_train)
(pred == y_train).mean()
```

```
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
```

Out[9]:
```
0.9468687122736419
```

In [10]:
```
#Test model accuracy on test set
pred = pl.predict(X_test)
(pred == y_test).mean()
```

```
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
```

Out[10]:
```
0.9251225952470766
```

In [11]:
```
#Test to see what hyperparameter is best to use (number of neighbors)

neigh_acc = {}
for i in range(1,30):
    pl = Pipeline([('preprocessor', preproc),('knn',KNeighborsClassifier(n_neig
    pl.fit(X_train, y_train)
    pred = pl.predict(X_test)
    neigh_acc[str(i) + ' neighbors'] = str(round((pred == y_test).mean(), 4) *
```

```
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
```

```
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
```

```
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
```

```
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
```

```
keepdims` will become False, the `axis` over which the statistic is taken will be eliminated, and the value None will no longer be accepted. Set `keepdims` to True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
```

In [12]:
```
#Model accuracy based on number of neighbors used
neigh_acc
```

Out[12]:
```
{'1 neighbors': '90.74%',
 '2 neighbors': '90.32%',
 '3 neighbors': '92.08%',
 '4 neighbors': '92.02%',
 '5 neighbors': '92.51%',
 '6 neighbors': '92.85%',
 '7 neighbors': '92.86%',
 '8 neighbors': '92.78%',
 '9 neighbors': '92.74%',
 '10 neighbors': '92.91%',
 '11 neighbors': '92.89%',
 '12 neighbors': '93.02%',
 '13 neighbors': '92.95%',
 '14 neighbors': '93.19%',
 '15 neighbors': '93.08%',
 '16 neighbors': '93.0%',
 '17 neighbors': '92.97%',
 '18 neighbors': '93.19%',
 '19 neighbors': '92.93%',
 '20 neighbors': '93.04%',
 '21 neighbors': '93.08%',
 '22 neighbors': '93.28%',
 '23 neighbors': '93.25%',
 '24 neighbors': '93.28%',
 '25 neighbors': '93.23%',
 '26 neighbors': '93.30%',
 '27 neighbors': '93.25%',
 '28 neighbors': '93.25%',
 '29 neighbors': '93.17%'}
```

Final Model

In [44]:
```
#Dataset containing only team stats
team_stats.head()
```

Out[44]:

	gameid	datacompleteness	url	league	year	split	playoffs	dat
10	ESPORTSTMNT01_2690210	complete	NaN	LCK CL	2022	Spring	0	2022 01-1 07:44:0
11	ESPORTSTMNT01_2690210	complete	NaN	LCK CL	2022	Spring	0	2022 01-1 07:44:0
22	ESPORTSTMNT01_2690219	complete	NaN	LCK CL	2022	Spring	0	2022 01-1 08:38:2
23	ESPORTSTMNT01_2690219	complete	NaN	LCK CL	2022	Spring	0	2022 01-1 08:38:2
46	ESPORTSTMNT01_2690227	complete	NaN	LCK CL	2022	Spring	0	2022 01-1 09:51:1

5 rows × 125 columns

In [34]:
```
#Create dictionary containing totalgold based on gameid and side
dic = team_stats.groupby(['gameid', 'side']).agg({'totalgold':'first'}).to_dict
dic
```

Out[34]:
```
{('ESPORTSTMNT01_2690210', 'Blue'): 47070,
 ('ESPORTSTMNT01_2690210', 'Red'): 52617,
 ('ESPORTSTMNT01_2690219', 'Blue'): 57629,
 ('ESPORTSTMNT01_2690219', 'Red'): 71004,
 ('ESPORTSTMNT01_2690227', 'Blue'): 62868,
 ('ESPORTSTMNT01_2690227', 'Red'): 52177,
 ('ESPORTSTMNT01_2690255', 'Blue'): 69897,
 ('ESPORTSTMNT01_2690255', 'Red'): 78701,
 ('ESPORTSTMNT01_2690264', 'Blue'): 64564,
 ('ESPORTSTMNT01_2690264', 'Red'): 53693,
 ('ESPORTSTMNT01_2690302', 'Blue'): 86016,
 ('ESPORTSTMNT01_2690302', 'Red'): 90593,
 ('ESPORTSTMNT01_2690328', 'Blue'): 70143,
 ('ESPORTSTMNT01_2690328', 'Red'): 56806,
 ('ESPORTSTMNT01_2690351', 'Blue'): 54722,
 ('ESPORTSTMNT01_2690351', 'Red'): 64849,
 ('ESPORTSTMNT01_2690370', 'Blue'): 45316,
 ('ESPORTSTMNT01_2690370', 'Red'): 59013,
 ('ESPORTSTMNT01_2690390', 'Blue'): 40628,
 ('ESPORTSTMNT01_2690390', 'Red'): 53833,
 ('ESPORTSTMNT01_2690439', 'Blue'): 54926,
 ('ESPORTSTMNT01_2690439', 'Red'): 46712,
 ('ESPORTSTMNT01_2690444', 'Blue'): 41654,
 ('ESPORTSTMNT01_2690444', 'Red'): 52411,
 ('ESPORTSTMNT01_2690460', 'Blue'): 45776,
 ('ESPORTSTMNT01_2690460', 'Red'): 55240,
 ('ESPORTSTMNT01_2690464', 'Blue'): 68485,
 ('ESPORTSTMNT01_2690464', 'Red'): 78378,
 ('ESPORTSTMNT01_2690479', 'Blue'): 67795,
 ('ESPORTSTMNT01_2690479', 'Red'): 52032,
 ('ESPORTSTMNT01_2690695', 'Blue'): 60674,
 ('ESPORTSTMNT01_2690695', 'Red'): 67152,
 ('ESPORTSTMNT01_2690705', 'Blue'): 66455,
 ('ESPORTSTMNT01_2690705', 'Red'): 58806,
 ('ESPORTSTMNT01_2690725', 'Blue'): 38495,
 ('ESPORTSTMNT01_2690725', 'Red'): 51002,
 ('ESPORTSTMNT01_2690815', 'Blue'): 45192,
 ('ESPORTSTMNT01_2690815', 'Red'): 58383,
 ('ESPORTSTMNT01_2690835', 'Blue'): 75300,
 ('ESPORTSTMNT01_2690835', 'Red'): 61542,
 ('ESPORTSTMNT01_2690862', 'Blue'): 61887,
 ('ESPORTSTMNT01_2690862', 'Red'): 49366,
 ('ESPORTSTMNT01_2690882', 'Blue'): 49815,
 ('ESPORTSTMNT01_2690882', 'Red'): 37852,
 ('ESPORTSTMNT01_2690906', 'Blue'): 65109,
 ('ESPORTSTMNT01_2690906', 'Red'): 71563,
 ('ESPORTSTMNT01_2691079', 'Blue'): 57962,
 ('ESPORTSTMNT01_2691079', 'Red'): 62924,
 ('ESPORTSTMNT01_2691080', 'Blue'): 56379,
 ('ESPORTSTMNT01_2691080', 'Red'): 67109,
 ('ESPORTSTMNT01_2691081', 'Blue'): 44154,
 ('ESPORTSTMNT01_2691081', 'Red'): 56859,
 ('ESPORTSTMNT01_2691082', 'Blue'): 44238,
 ('ESPORTSTMNT01_2691082', 'Red'): 52653,
 ('ESPORTSTMNT01_2691269', 'Blue'): 44671,
 ('ESPORTSTMNT01_2691269', 'Red'): 55905,
 ('ESPORTSTMNT01_2691284', 'Blue'): 62784,
```

```
('ESPORTSTMNT01_2691284', 'Red'): 72834,
('ESPORTSTMNT01_2691301', 'Blue'): 44928,
('ESPORTSTMNT01_2691301', 'Red'): 55686,
('ESPORTSTMNT01_2691334', 'Blue'): 63602,
('ESPORTSTMNT01_2691334', 'Red'): 49392,
('ESPORTSTMNT01_2691349', 'Blue'): 58718,
('ESPORTSTMNT01_2691349', 'Red'): 50408,
('ESPORTSTMNT01_2691376', 'Blue'): 51693,
('ESPORTSTMNT01_2691376', 'Red'): 67333,
('ESPORTSTMNT01_2691392', 'Blue'): 39691,
('ESPORTSTMNT01_2691392', 'Red'): 50423,
('ESPORTSTMNT01_2691491', 'Blue'): 70191,
('ESPORTSTMNT01_2691491', 'Red'): 60618,
('ESPORTSTMNT01_2691492', 'Blue'): 57959,
('ESPORTSTMNT01_2691492', 'Red'): 62272,
('ESPORTSTMNT01_2691493', 'Blue'): 56724,
('ESPORTSTMNT01_2691493', 'Red'): 72476,
('ESPORTSTMNT01_2691494', 'Blue'): 52845,
('ESPORTSTMNT01_2691494', 'Red'): 44916,
('ESPORTSTMNT01_2691507', 'Blue'): 45102,
('ESPORTSTMNT01_2691507', 'Red'): 58528,
('ESPORTSTMNT01_2691515', 'Blue'): 57160,
('ESPORTSTMNT01_2691515', 'Red'): 40047,
('ESPORTSTMNT01_2691518', 'Blue'): 76283,
('ESPORTSTMNT01_2691518', 'Red'): 74244,
('ESPORTSTMNT01_2691546', 'Blue'): 86941,
('ESPORTSTMNT01_2691546', 'Red'): 79989,
('ESPORTSTMNT01_2691557', 'Blue'): 64683,
('ESPORTSTMNT01_2691557', 'Red'): 78702,
('ESPORTSTMNT01_2691713', 'Blue'): 38256,
('ESPORTSTMNT01_2691713', 'Red'): 55245,
('ESPORTSTMNT01_2692054', 'Blue'): 38318,
('ESPORTSTMNT01_2692054', 'Red'): 49858,
('ESPORTSTMNT01_2692264', 'Blue'): 59877,
('ESPORTSTMNT01_2692264', 'Red'): 64447,
('ESPORTSTMNT01_2692407', 'Blue'): 50856,
('ESPORTSTMNT01_2692407', 'Red'): 38303,
('ESPORTSTMNT01_2692425', 'Blue'): 53824,
('ESPORTSTMNT01_2692425', 'Red'): 64719,
('ESPORTSTMNT01_2692441', 'Blue'): 54346,
('ESPORTSTMNT01_2692441', 'Red'): 41949,
('ESPORTSTMNT01_2692497', 'Blue'): 62137,
('ESPORTSTMNT01_2692497', 'Red'): 49293,
('ESPORTSTMNT01_2692529', 'Blue'): 47182,
('ESPORTSTMNT01_2692529', 'Red'): 56437,
('ESPORTSTMNT01_2692548', 'Blue'): 64060,
('ESPORTSTMNT01_2692548', 'Red'): 49682,
('ESPORTSTMNT01_2692567', 'Blue'): 47377,
('ESPORTSTMNT01_2692567', 'Red'): 54623,
('ESPORTSTMNT01_2692672', 'Blue'): 61318,
('ESPORTSTMNT01_2692672', 'Red'): 52683,
('ESPORTSTMNT01_2692687', 'Blue'): 48892,
('ESPORTSTMNT01_2692687', 'Red'): 58632,
('ESPORTSTMNT01_2692705', 'Blue'): 66091,
('ESPORTSTMNT01_2692705', 'Red'): 52464,
('ESPORTSTMNT01_2692722', 'Blue'): 61868,
('ESPORTSTMNT01_2692722', 'Red'): 53259,
```

```
('ESPORTSTMNT01_2692750', 'Blue'): 56794,
('ESPORTSTMNT01_2692750', 'Red'): 46151,
('ESPORTSTMNT01_2692918', 'Blue'): 48621,
('ESPORTSTMNT01_2692918', 'Red'): 38931,
('ESPORTSTMNT01_2692951', 'Blue'): 46166,
('ESPORTSTMNT01_2692951', 'Red'): 61462,
('ESPORTSTMNT01_2692980', 'Blue'): 47036,
('ESPORTSTMNT01_2692980', 'Red'): 60305,
('ESPORTSTMNT01_2693004', 'Blue'): 75477,
('ESPORTSTMNT01_2693004', 'Red'): 67901,
('ESPORTSTMNT01_2693026', 'Blue'): 59300,
('ESPORTSTMNT01_2693026', 'Red'): 69976,
('ESPORTSTMNT01_2693034', 'Blue'): 54696,
('ESPORTSTMNT01_2693034', 'Red'): 46726,
('ESPORTSTMNT01_2693035', 'Blue'): 72622,
('ESPORTSTMNT01_2693035', 'Red'): 81137,
('ESPORTSTMNT01_2693060', 'Blue'): 59555,
('ESPORTSTMNT01_2693060', 'Red'): 64879,
('ESPORTSTMNT01_2693070', 'Blue'): 56852,
('ESPORTSTMNT01_2693070', 'Red'): 47853,
('ESPORTSTMNT01_2693089', 'Blue'): 49916,
('ESPORTSTMNT01_2693089', 'Red'): 61311,
('ESPORTSTMNT01_2693098', 'Blue'): 46628,
('ESPORTSTMNT01_2693098', 'Red'): 60904,
('ESPORTSTMNT01_2693100', 'Blue'): 50437,
('ESPORTSTMNT01_2693100', 'Red'): 62753,
('ESPORTSTMNT01_2693125', 'Blue'): 77899,
('ESPORTSTMNT01_2693125', 'Red'): 69603,
('ESPORTSTMNT01_2693131', 'Blue'): 87886,
('ESPORTSTMNT01_2693131', 'Red'): 78554,
('ESPORTSTMNT01_2693136', 'Blue'): 50963,
('ESPORTSTMNT01_2693136', 'Red'): 38592,
('ESPORTSTMNT01_2693351', 'Blue'): 62010,
('ESPORTSTMNT01_2693351', 'Red'): 59172,
('ESPORTSTMNT01_2693353', 'Blue'): 76017,
('ESPORTSTMNT01_2693353', 'Red'): 67374,
('ESPORTSTMNT01_2693410', 'Blue'): 64530,
('ESPORTSTMNT01_2693410', 'Red'): 54278,
('ESPORTSTMNT01_2693423', 'Blue'): 55418,
('ESPORTSTMNT01_2693423', 'Red'): 48796,
('ESPORTSTMNT01_2693486', 'Blue'): 66194,
('ESPORTSTMNT01_2693486', 'Red'): 70379,
('ESPORTSTMNT01_2693488', 'Blue'): 69281,
('ESPORTSTMNT01_2693488', 'Red'): 67736,
('ESPORTSTMNT01_2693537', 'Blue'): 46476,
('ESPORTSTMNT01_2693537', 'Red'): 60006,
('ESPORTSTMNT01_2693557', 'Blue'): 66907,
('ESPORTSTMNT01_2693557', 'Red'): 59675,
('ESPORTSTMNT01_2693619', 'Blue'): 50067,
('ESPORTSTMNT01_2693619', 'Red'): 57774,
('ESPORTSTMNT01_2693789', 'Blue'): 72494,
('ESPORTSTMNT01_2693789', 'Red'): 62148,
('ESPORTSTMNT01_2693854', 'Blue'): 66654,
('ESPORTSTMNT01_2693854', 'Red'): 63454,
('ESPORTSTMNT01_2693886', 'Blue'): 57793,
('ESPORTSTMNT01_2693886', 'Red'): 68414,
('ESPORTSTMNT01_2694027', 'Blue'): 64325,
```

```
('ESPORTSTMNT01_2694027', 'Red'): 46354,
('ESPORTSTMNT01_2694052', 'Blue'): 73030,
('ESPORTSTMNT01_2694052', 'Red'): 63375,
('ESPORTSTMNT01_2694058', 'Blue'): 59523,
('ESPORTSTMNT01_2694058', 'Red'): 68123,
('ESPORTSTMNT01_2694069', 'Blue'): 58494,
('ESPORTSTMNT01_2694069', 'Red'): 51185,
('ESPORTSTMNT01_2694087', 'Blue'): 38920,
('ESPORTSTMNT01_2694087', 'Red'): 55838,
('ESPORTSTMNT01_2694092', 'Blue'): 60702,
('ESPORTSTMNT01_2694092', 'Red'): 50023,
('ESPORTSTMNT01_2694094', 'Blue'): 72647,
('ESPORTSTMNT01_2694094', 'Red'): 64157,
('ESPORTSTMNT01_2694120', 'Blue'): 58216,
('ESPORTSTMNT01_2694120', 'Red'): 69290,
('ESPORTSTMNT01_2694124', 'Blue'): 59640,
('ESPORTSTMNT01_2694124', 'Red'): 48907,
('ESPORTSTMNT01_2694155', 'Blue'): 65319,
('ESPORTSTMNT01_2694155', 'Red'): 66434,
('ESPORTSTMNT01_2694167', 'Blue'): 44939,
('ESPORTSTMNT01_2694167', 'Red'): 37726,
('ESPORTSTMNT01_2694174', 'Blue'): 60390,
('ESPORTSTMNT01_2694174', 'Red'): 48819,
('ESPORTSTMNT01_2694175', 'Blue'): 55971,
('ESPORTSTMNT01_2694175', 'Red'): 67136,
('ESPORTSTMNT01_2694177', 'Blue'): 58367,
('ESPORTSTMNT01_2694177', 'Red'): 65837,
('ESPORTSTMNT01_2694197', 'Blue'): 79729,
('ESPORTSTMNT01_2694197', 'Red'): 84750,
('ESPORTSTMNT01_2694200', 'Blue'): 50727,
('ESPORTSTMNT01_2694200', 'Red'): 38218,
('ESPORTSTMNT01_2694212', 'Blue'): 43482,
('ESPORTSTMNT01_2694212', 'Red'): 56909,
('ESPORTSTMNT01_2694214', 'Blue'): 44249,
('ESPORTSTMNT01_2694214', 'Red'): 54057,
('ESPORTSTMNT01_2694223', 'Blue'): 49532,
('ESPORTSTMNT01_2694223', 'Red'): 40840,
('ESPORTSTMNT01_2694224', 'Blue'): 69231,
('ESPORTSTMNT01_2694224', 'Red'): 62539,
('ESPORTSTMNT01_2694225', 'Blue'): 60736,
('ESPORTSTMNT01_2694225', 'Red'): 69835,
('ESPORTSTMNT01_2694252', 'Blue'): 70460,
('ESPORTSTMNT01_2694252', 'Red'): 75268,
('ESPORTSTMNT01_2694254', 'Blue'): 57649,
('ESPORTSTMNT01_2694254', 'Red'): 67259,
('ESPORTSTMNT01_2694272', 'Blue'): 64662,
('ESPORTSTMNT01_2694272', 'Red'): 56052,
('ESPORTSTMNT01_2694273', 'Blue'): 62081,
('ESPORTSTMNT01_2694273', 'Red'): 51125,
('ESPORTSTMNT01_2694280', 'Blue'): 59283,
('ESPORTSTMNT01_2694280', 'Red'): 41652,
('ESPORTSTMNT01_2694285', 'Blue'): 57801,
('ESPORTSTMNT01_2694285', 'Red'): 44302,
('ESPORTSTMNT01_2694293', 'Blue'): 42464,
('ESPORTSTMNT01_2694293', 'Red'): 54186,
('ESPORTSTMNT01_2694317', 'Blue'): 57417,
('ESPORTSTMNT01_2694317', 'Red'): 42534,
```

```
('ESPORTSTMNT01_2694375', 'Blue'): 67741,
('ESPORTSTMNT01_2694375', 'Red'): 74537,
('ESPORTSTMNT01_2694425', 'Blue'): 56180,
('ESPORTSTMNT01_2694425', 'Red'): 43350,
('ESPORTSTMNT01_2694428', 'Blue'): 62370,
('ESPORTSTMNT01_2694428', 'Red'): 74783,
('ESPORTSTMNT01_2694459', 'Blue'): 56307,
('ESPORTSTMNT01_2694459', 'Red'): 66704,
('ESPORTSTMNT01_2694589', 'Blue'): 49605,
('ESPORTSTMNT01_2694589', 'Red'): 65963,
('ESPORTSTMNT01_2694656', 'Blue'): 61493,
('ESPORTSTMNT01_2694656', 'Red'): 50326,
('ESPORTSTMNT01_2694704', 'Blue'): 51164,
('ESPORTSTMNT01_2694704', 'Red'): 31998,
('ESPORTSTMNT01_2694761', 'Blue'): 50626,
('ESPORTSTMNT01_2694761', 'Red'): 43327,
('ESPORTSTMNT01_2694771', 'Blue'): 48161,
('ESPORTSTMNT01_2694771', 'Red'): 57666,
('ESPORTSTMNT01_2694807', 'Blue'): 48215,
('ESPORTSTMNT01_2694807', 'Red'): 59232,
('ESPORTSTMNT01_2694837', 'Blue'): 54941,
('ESPORTSTMNT01_2694837', 'Red'): 43344,
('ESPORTSTMNT01_2694867', 'Blue'): 59534,
('ESPORTSTMNT01_2694867', 'Red'): 69524,
('ESPORTSTMNT01_2694903', 'Blue'): 60669,
('ESPORTSTMNT01_2694903', 'Red'): 61574,
('ESPORTSTMNT01_2694921', 'Blue'): 60176,
('ESPORTSTMNT01_2694921', 'Red'): 72776,
('ESPORTSTMNT01_2694932', 'Blue'): 34838,
('ESPORTSTMNT01_2694932', 'Red'): 48547,
('ESPORTSTMNT01_2694943', 'Blue'): 42727,
('ESPORTSTMNT01_2694943', 'Red'): 58567,
('ESPORTSTMNT01_2694991', 'Blue'): 49992,
('ESPORTSTMNT01_2694991', 'Red'): 57758,
('ESPORTSTMNT01_2694992', 'Blue'): 70187,
('ESPORTSTMNT01_2694992', 'Red'): 82853,
('ESPORTSTMNT01_2695078', 'Blue'): 54859,
('ESPORTSTMNT01_2695078', 'Red'): 44050,
('ESPORTSTMNT01_2695092', 'Blue'): 48688,
('ESPORTSTMNT01_2695092', 'Red'): 61059,
('ESPORTSTMNT01_2695387', 'Blue'): 58379,
('ESPORTSTMNT01_2695387', 'Red'): 67534,
('ESPORTSTMNT01_2695415', 'Blue'): 59651,
('ESPORTSTMNT01_2695415', 'Red'): 45351,
('ESPORTSTMNT01_2695435', 'Blue'): 46767,
('ESPORTSTMNT01_2695435', 'Red'): 58635,
('ESPORTSTMNT01_2695445', 'Blue'): 46924,
('ESPORTSTMNT01_2695445', 'Red'): 61802,
('ESPORTSTMNT01_2695450', 'Blue'): 57042,
('ESPORTSTMNT01_2695450', 'Red'): 45558,
('ESPORTSTMNT01_2695456', 'Blue'): 50991,
('ESPORTSTMNT01_2695456', 'Red'): 43410,
('ESPORTSTMNT01_2695459', 'Blue'): 55441,
('ESPORTSTMNT01_2695459', 'Red'): 41390,
('ESPORTSTMNT01_2695465', 'Blue'): 57153,
('ESPORTSTMNT01_2695465', 'Red'): 69514,
('ESPORTSTMNT01_2695467', 'Blue'): 73183,
```

```
('ESPORTSTMNT01_2695467', 'Red'): 65851,
('ESPORTSTMNT01_2695530', 'Blue'): 59464,
('ESPORTSTMNT01_2695530', 'Red'): 64701,
('ESPORTSTMNT01_2695541', 'Blue'): 51782,
('ESPORTSTMNT01_2695541', 'Red'): 60800,
('ESPORTSTMNT01_2695573', 'Blue'): 74328,
('ESPORTSTMNT01_2695573', 'Red'): 61324,
('ESPORTSTMNT01_2695586', 'Blue'): 43591,
('ESPORTSTMNT01_2695586', 'Red'): 54888,
('ESPORTSTMNT01_2695682', 'Blue'): 61646,
('ESPORTSTMNT01_2695682', 'Red'): 66766,
('ESPORTSTMNT01_2695708', 'Blue'): 71210,
('ESPORTSTMNT01_2695708', 'Red'): 64612,
('ESPORTSTMNT01_2695722', 'Blue'): 59571,
('ESPORTSTMNT01_2695722', 'Red'): 56124,
('ESPORTSTMNT01_2695774', 'Blue'): 72544,
('ESPORTSTMNT01_2695774', 'Red'): 59163,
('ESPORTSTMNT01_2695805', 'Blue'): 48476,
('ESPORTSTMNT01_2695805', 'Red'): 38662,
('ESPORTSTMNT01_2695807', 'Blue'): 51114,
('ESPORTSTMNT01_2695807', 'Red'): 40224,
('ESPORTSTMNT01_2695835', 'Blue'): 71641,
('ESPORTSTMNT01_2695835', 'Red'): 71408,
('ESPORTSTMNT01_2695850', 'Blue'): 61008,
('ESPORTSTMNT01_2695850', 'Red'): 64302,
('ESPORTSTMNT01_2696159', 'Blue'): 54937,
('ESPORTSTMNT01_2696159', 'Red'): 43579,
('ESPORTSTMNT01_2696199', 'Blue'): 71939,
('ESPORTSTMNT01_2696199', 'Red'): 65014,
('ESPORTSTMNT01_2696249', 'Blue'): 81381,
('ESPORTSTMNT01_2696249', 'Red'): 82072,
('ESPORTSTMNT01_2696303', 'Blue'): 57061,
('ESPORTSTMNT01_2696303', 'Red'): 63007,
('ESPORTSTMNT01_2696307', 'Blue'): 66390,
('ESPORTSTMNT01_2696307', 'Red'): 48757,
('ESPORTSTMNT01_2696545', 'Blue'): 73397,
('ESPORTSTMNT01_2696545', 'Red'): 73373,
('ESPORTSTMNT01_2696601', 'Blue'): 58084,
('ESPORTSTMNT01_2696601', 'Red'): 46064,
('ESPORTSTMNT01_2696635', 'Blue'): 57517,
('ESPORTSTMNT01_2696635', 'Red'): 48394,
('ESPORTSTMNT01_2696675', 'Blue'): 36790,
('ESPORTSTMNT01_2696675', 'Red'): 53349,
('ESPORTSTMNT01_2696704', 'Blue'): 72025,
('ESPORTSTMNT01_2696704', 'Red'): 60723,
('ESPORTSTMNT01_2696752', 'Blue'): 39586,
('ESPORTSTMNT01_2696752', 'Red'): 51659,
('ESPORTSTMNT01_2696754', 'Blue'): 62122,
('ESPORTSTMNT01_2696754', 'Red'): 63480,
('ESPORTSTMNT01_2696769', 'Blue'): 43864,
('ESPORTSTMNT01_2696769', 'Red'): 51565,
('ESPORTSTMNT01_2696771', 'Blue'): 62025,
('ESPORTSTMNT01_2696771', 'Red'): 54120,
('ESPORTSTMNT01_2696789', 'Blue'): 41427,
('ESPORTSTMNT01_2696789', 'Red'): 56475,
('ESPORTSTMNT01_2696795', 'Blue'): 53399,
('ESPORTSTMNT01_2696795', 'Red'): 39394,
```

```
('ESPORTSTMNT01_2696802', 'Blue'): 64921,
('ESPORTSTMNT01_2696802', 'Red'): 72002,
('ESPORTSTMNT01_2696804', 'Blue'): 32774,
('ESPORTSTMNT01_2696804', 'Red'): 51001,
('ESPORTSTMNT01_2696809', 'Blue'): 64720,
('ESPORTSTMNT01_2696809', 'Red'): 57240,
('ESPORTSTMNT01_2696852', 'Blue'): 49154,
('ESPORTSTMNT01_2696852', 'Red'): 34109,
('ESPORTSTMNT01_2696871', 'Blue'): 40616,
('ESPORTSTMNT01_2696871', 'Red'): 57430,
('ESPORTSTMNT01_2696975', 'Blue'): 49296,
('ESPORTSTMNT01_2696975', 'Red'): 40015,
('ESPORTSTMNT01_2697010', 'Blue'): 65384,
('ESPORTSTMNT01_2697010', 'Red'): 71793,
('ESPORTSTMNT01_2697020', 'Blue'): 61442,
('ESPORTSTMNT01_2697020', 'Red'): 50673,
('ESPORTSTMNT01_2697054', 'Blue'): 55906,
('ESPORTSTMNT01_2697054', 'Red'): 43148,
('ESPORTSTMNT01_2697080', 'Blue'): 61285,
('ESPORTSTMNT01_2697080', 'Red'): 50872,
('ESPORTSTMNT01_2697096', 'Blue'): 48198,
('ESPORTSTMNT01_2697096', 'Red'): 60765,
('ESPORTSTMNT01_2697106', 'Blue'): 63939,
('ESPORTSTMNT01_2697106', 'Red'): 55574,
('ESPORTSTMNT01_2697116', 'Blue'): 60072,
('ESPORTSTMNT01_2697116', 'Red'): 70235,
('ESPORTSTMNT01_2697165', 'Blue'): 43481,
('ESPORTSTMNT01_2697165', 'Red'): 57030,
('ESPORTSTMNT01_2697166', 'Blue'): 71903,
('ESPORTSTMNT01_2697166', 'Red'): 60669,
('ESPORTSTMNT01_2697167', 'Blue'): 59082,
('ESPORTSTMNT01_2697167', 'Red'): 49471,
('ESPORTSTMNT01_2697196', 'Blue'): 48633,
('ESPORTSTMNT01_2697196', 'Red'): 55104,
('ESPORTSTMNT01_2697203', 'Blue'): 49616,
('ESPORTSTMNT01_2697203', 'Red'): 60518,
('ESPORTSTMNT01_2697251', 'Blue'): 62938,
('ESPORTSTMNT01_2697251', 'Red'): 55298,
('ESPORTSTMNT01_2697253', 'Blue'): 57753,
('ESPORTSTMNT01_2697253', 'Red'): 52488,
('ESPORTSTMNT01_2697277', 'Blue'): 76143,
('ESPORTSTMNT01_2697277', 'Red'): 65794,
('ESPORTSTMNT01_2697278', 'Blue'): 47170,
('ESPORTSTMNT01_2697278', 'Red'): 64007,
('ESPORTSTMNT01_2697284', 'Blue'): 42397,
('ESPORTSTMNT01_2697284', 'Red'): 30382,
('ESPORTSTMNT01_2697288', 'Blue'): 57631,
('ESPORTSTMNT01_2697288', 'Red'): 48628,
('ESPORTSTMNT01_2697304', 'Blue'): 39889,
('ESPORTSTMNT01_2697304', 'Red'): 27177,
('ESPORTSTMNT01_2697305', 'Blue'): 47770,
('ESPORTSTMNT01_2697305', 'Red'): 59201,
('ESPORTSTMNT01_2697320', 'Blue'): 55624,
('ESPORTSTMNT01_2697320', 'Red'): 68837,
('ESPORTSTMNT01_2697328', 'Blue'): 61525,
('ESPORTSTMNT01_2697328', 'Red'): 50871,
('ESPORTSTMNT01_2697333', 'Blue'): 45098,
```

```
('ESPORTSTMNT01_2697333', 'Red'): 56645,
('ESPORTSTMNT01_2697336', 'Blue'): 88165,
('ESPORTSTMNT01_2697336', 'Red'): 79099,
('ESPORTSTMNT01_2697337', 'Blue'): 58503,
('ESPORTSTMNT01_2697337', 'Red'): 44001,
('ESPORTSTMNT01_2697394', 'Blue'): 50488,
('ESPORTSTMNT01_2697394', 'Red'): 65010,
('ESPORTSTMNT01_2697410', 'Blue'): 47312,
('ESPORTSTMNT01_2697410', 'Red'): 61076,
('ESPORTSTMNT01_2697430', 'Blue'): 43796,
('ESPORTSTMNT01_2697430', 'Red'): 54650,
('ESPORTSTMNT01_2697431', 'Blue'): 48992,
('ESPORTSTMNT01_2697431', 'Red'): 32990,
('ESPORTSTMNT01_2697444', 'Blue'): 45758,
('ESPORTSTMNT01_2697444', 'Red'): 55577,
('ESPORTSTMNT01_2697472', 'Blue'): 76184,
('ESPORTSTMNT01_2697472', 'Red'): 72105,
('ESPORTSTMNT01_2697476', 'Blue'): 55249,
('ESPORTSTMNT01_2697476', 'Red'): 67386,
('ESPORTSTMNT01_2697480', 'Blue'): 67688,
('ESPORTSTMNT01_2697480', 'Red'): 63042,
('ESPORTSTMNT01_2697508', 'Blue'): 45268,
('ESPORTSTMNT01_2697508', 'Red'): 58258,
('ESPORTSTMNT01_2697518', 'Blue'): 58137,
('ESPORTSTMNT01_2697518', 'Red'): 42659,
('ESPORTSTMNT01_2697524', 'Blue'): 59513,
('ESPORTSTMNT01_2697524', 'Red'): 52203,
('ESPORTSTMNT01_2697526', 'Blue'): 43039,
('ESPORTSTMNT01_2697526', 'Red'): 52581,
('ESPORTSTMNT01_2697549', 'Blue'): 66843,
('ESPORTSTMNT01_2697549', 'Red'): 63315,
('ESPORTSTMNT01_2697556', 'Blue'): 61407,
('ESPORTSTMNT01_2697556', 'Red'): 62501,
('ESPORTSTMNT01_2697580', 'Blue'): 68577,
('ESPORTSTMNT01_2697580', 'Red'): 58803,
('ESPORTSTMNT01_2697592', 'Blue'): 82185,
('ESPORTSTMNT01_2697592', 'Red'): 72544,
('ESPORTSTMNT01_2697602', 'Blue'): 78681,
('ESPORTSTMNT01_2697602', 'Red'): 73712,
('ESPORTSTMNT01_2697616', 'Blue'): 59289,
('ESPORTSTMNT01_2697616', 'Red'): 64803,
('ESPORTSTMNT01_2697640', 'Blue'): 63334,
('ESPORTSTMNT01_2697640', 'Red'): 49436,
('ESPORTSTMNT01_2697651', 'Blue'): 56554,
('ESPORTSTMNT01_2697651', 'Red'): 43261,
('ESPORTSTMNT01_2697652', 'Blue'): 64655,
('ESPORTSTMNT01_2697652', 'Red'): 50031,
('ESPORTSTMNT01_2697660', 'Blue'): 53600,
('ESPORTSTMNT01_2697660', 'Red'): 62670,
('ESPORTSTMNT01_2697663', 'Blue'): 49870,
('ESPORTSTMNT01_2697663', 'Red'): 64961,
('ESPORTSTMNT01_2697669', 'Blue'): 69272,
('ESPORTSTMNT01_2697669', 'Red'): 59157,
('ESPORTSTMNT01_2697682', 'Blue'): 56236,
('ESPORTSTMNT01_2697682', 'Red'): 62599,
('ESPORTSTMNT01_2697747', 'Blue'): 59707,
('ESPORTSTMNT01_2697747', 'Red'): 50047,
```

```
('ESPORTSTMNT01_2697748', 'Blue'): 52464,
('ESPORTSTMNT01_2697748', 'Red'): 65565,
('ESPORTSTMNT01_2697749', 'Blue'): 70743,
('ESPORTSTMNT01_2697749', 'Red'): 76929,
('ESPORTSTMNT01_2697763', 'Blue'): 61236,
('ESPORTSTMNT01_2697763', 'Red'): 65937,
('ESPORTSTMNT01_2697798', 'Blue'): 63847,
('ESPORTSTMNT01_2697798', 'Red'): 56907,
('ESPORTSTMNT01_2697803', 'Blue'): 51926,
('ESPORTSTMNT01_2697803', 'Red'): 65589,
('ESPORTSTMNT01_2697844', 'Blue'): 53661,
('ESPORTSTMNT01_2697844', 'Red'): 39020,
('ESPORTSTMNT01_2697850', 'Blue'): 56660,
('ESPORTSTMNT01_2697850', 'Red'): 47256,
('ESPORTSTMNT01_2697872', 'Blue'): 60755,
('ESPORTSTMNT01_2697872', 'Red'): 49939,
('ESPORTSTMNT01_2697879', 'Blue'): 50724,
('ESPORTSTMNT01_2697879', 'Red'): 60342,
('ESPORTSTMNT01_2697895', 'Blue'): 54708,
('ESPORTSTMNT01_2697895', 'Red'): 44145,
('ESPORTSTMNT01_2697897', 'Blue'): 39532,
('ESPORTSTMNT01_2697897', 'Red'): 48217,
('ESPORTSTMNT01_2697916', 'Blue'): 57207,
('ESPORTSTMNT01_2697916', 'Red'): 47593,
('ESPORTSTMNT01_2697927', 'Blue'): 56337,
('ESPORTSTMNT01_2697927', 'Red'): 72288,
('ESPORTSTMNT01_2697959', 'Blue'): 83257,
('ESPORTSTMNT01_2697959', 'Red'): 75361,
('ESPORTSTMNT01_2697965', 'Blue'): 49929,
('ESPORTSTMNT01_2697965', 'Red'): 42893,
('ESPORTSTMNT01_2697975', 'Blue'): 66404,
('ESPORTSTMNT01_2697975', 'Red'): 56079,
('ESPORTSTMNT01_2697976', 'Blue'): 71541,
('ESPORTSTMNT01_2697976', 'Red'): 69731,
('ESPORTSTMNT01_2697985', 'Blue'): 73985,
('ESPORTSTMNT01_2697985', 'Red'): 60314,
('ESPORTSTMNT01_2698077', 'Blue'): 44745,
('ESPORTSTMNT01_2698077', 'Red'): 57550,
('ESPORTSTMNT01_2698089', 'Blue'): 41694,
('ESPORTSTMNT01_2698089', 'Red'): 51897,
('ESPORTSTMNT01_2698102', 'Blue'): 60550,
('ESPORTSTMNT01_2698102', 'Red'): 72934,
('ESPORTSTMNT01_2698115', 'Blue'): 62990,
('ESPORTSTMNT01_2698115', 'Red'): 50705,
('ESPORTSTMNT01_2698121', 'Blue'): 53812,
('ESPORTSTMNT01_2698121', 'Red'): 63474,
('ESPORTSTMNT01_2698125', 'Blue'): 34120,
('ESPORTSTMNT01_2698125', 'Red'): 47176,
('ESPORTSTMNT01_2698126', 'Blue'): 39797,
('ESPORTSTMNT01_2698126', 'Red'): 50199,
('ESPORTSTMNT01_2698139', 'Blue'): 78448,
('ESPORTSTMNT01_2698139', 'Red'): 83047,
('ESPORTSTMNT01_2698145', 'Blue'): 59116,
('ESPORTSTMNT01_2698145', 'Red'): 69021,
('ESPORTSTMNT01_2698151', 'Blue'): 61592,
('ESPORTSTMNT01_2698151', 'Red'): 49063,
('ESPORTSTMNT01_2698191', 'Blue'): 54266,
```

```
('ESPORTSTMNT01_2698191', 'Red'): 42169,
('ESPORTSTMNT01_2698230', 'Blue'): 58762,
('ESPORTSTMNT01_2698230', 'Red'): 51420,
('ESPORTSTMNT01_2698240', 'Blue'): 47626,
('ESPORTSTMNT01_2698240', 'Red'): 59504,
('ESPORTSTMNT01_2698245', 'Blue'): 61170,
('ESPORTSTMNT01_2698245', 'Red'): 55531,
('ESPORTSTMNT01_2698252', 'Blue'): 50811,
('ESPORTSTMNT01_2698252', 'Red'): 59767,
('ESPORTSTMNT01_2698259', 'Blue'): 78905,
('ESPORTSTMNT01_2698259', 'Red'): 69880,
('ESPORTSTMNT01_2698302', 'Blue'): 54095,
('ESPORTSTMNT01_2698302', 'Red'): 62315,
('ESPORTSTMNT01_2698304', 'Blue'): 51385,
('ESPORTSTMNT01_2698304', 'Red'): 62052,
('ESPORTSTMNT01_2698306', 'Blue'): 51704,
('ESPORTSTMNT01_2698306', 'Red'): 39551,
('ESPORTSTMNT01_2698310', 'Blue'): 62299,
('ESPORTSTMNT01_2698310', 'Red'): 55413,
('ESPORTSTMNT01_2698350', 'Blue'): 39713,
('ESPORTSTMNT01_2698350', 'Red'): 53805,
('ESPORTSTMNT01_2698440', 'Blue'): 61444,
('ESPORTSTMNT01_2698440', 'Red'): 64715,
('ESPORTSTMNT01_2700416', 'Blue'): 58066,
('ESPORTSTMNT01_2700416', 'Red'): 64325,
('ESPORTSTMNT01_2700432', 'Blue'): 68792,
('ESPORTSTMNT01_2700432', 'Red'): 58260,
('ESPORTSTMNT01_2700456', 'Blue'): 65563,
('ESPORTSTMNT01_2700456', 'Red'): 58516,
('ESPORTSTMNT01_2700475', 'Blue'): 34790,
('ESPORTSTMNT01_2700475', 'Red'): 48084,
('ESPORTSTMNT01_2700491', 'Blue'): 64674,
('ESPORTSTMNT01_2700491', 'Red'): 74260,
('ESPORTSTMNT01_2700685', 'Blue'): 58552,
('ESPORTSTMNT01_2700685', 'Red'): 44481,
('ESPORTSTMNT01_2700707', 'Blue'): 56811,
('ESPORTSTMNT01_2700707', 'Red'): 66627,
('ESPORTSTMNT01_2700755', 'Blue'): 69897,
('ESPORTSTMNT01_2700755', 'Red'): 71308,
('ESPORTSTMNT01_2700783', 'Blue'): 44347,
('ESPORTSTMNT01_2700783', 'Red'): 55302,
('ESPORTSTMNT01_2700815', 'Blue'): 63747,
('ESPORTSTMNT01_2700815', 'Red'): 67669,
('ESPORTSTMNT01_2701100', 'Blue'): 50329,
('ESPORTSTMNT01_2701100', 'Red'): 51058,
('ESPORTSTMNT01_2701108', 'Blue'): 47907,
('ESPORTSTMNT01_2701108', 'Red'): 58935,
('ESPORTSTMNT01_2701116', 'Blue'): 67437,
('ESPORTSTMNT01_2701116', 'Red'): 58453,
('ESPORTSTMNT01_2701117', 'Blue'): 55938,
('ESPORTSTMNT01_2701117', 'Red'): 48180,
('ESPORTSTMNT01_2701131', 'Blue'): 60771,
('ESPORTSTMNT01_2701131', 'Red'): 51951,
('ESPORTSTMNT01_2701132', 'Blue'): 67353,
('ESPORTSTMNT01_2701132', 'Red'): 55241,
('ESPORTSTMNT01_2701145', 'Blue'): 66803,
('ESPORTSTMNT01_2701145', 'Red'): 63599,
```

```
('ESPORTSTMNT01_2701152', 'Blue'): 56610,
('ESPORTSTMNT01_2701152', 'Red'): 64956,
('ESPORTSTMNT01_2701158', 'Blue'): 58243,
('ESPORTSTMNT01_2701158', 'Red'): 45034,
('ESPORTSTMNT01_2701159', 'Blue'): 46691,
('ESPORTSTMNT01_2701159', 'Red'): 58522,
('ESPORTSTMNT01_2701183', 'Blue'): 31918,
('ESPORTSTMNT01_2701183', 'Red'): 51005,
('ESPORTSTMNT01_2701230', 'Blue'): 42974,
('ESPORTSTMNT01_2701230', 'Red'): 54398,
('ESPORTSTMNT01_2701248', 'Blue'): 67791,
('ESPORTSTMNT01_2701248', 'Red'): 57396,
('ESPORTSTMNT01_2701277', 'Blue'): 61659,
('ESPORTSTMNT01_2701277', 'Red'): 52660,
('ESPORTSTMNT01_2701282', 'Blue'): 60614,
('ESPORTSTMNT01_2701282', 'Red'): 72967,
('ESPORTSTMNT01_2701289', 'Blue'): 68723,
('ESPORTSTMNT01_2701289', 'Red'): 76622,
('ESPORTSTMNT01_2701333', 'Blue'): 40836,
('ESPORTSTMNT01_2701333', 'Red'): 49166,
('ESPORTSTMNT01_2701375', 'Blue'): 55095,
('ESPORTSTMNT01_2701375', 'Red'): 68797,
('ESPORTSTMNT01_2701400', 'Blue'): 56693,
('ESPORTSTMNT01_2701400', 'Red'): 42560,
('ESPORTSTMNT01_2701412', 'Blue'): 54536,
('ESPORTSTMNT01_2701412', 'Red'): 45344,
('ESPORTSTMNT01_2701429', 'Blue'): 74961,
('ESPORTSTMNT01_2701429', 'Red'): 75753,
('ESPORTSTMNT01_2701456', 'Blue'): 66576,
('ESPORTSTMNT01_2701456', 'Red'): 48487,
('ESPORTSTMNT01_2701489', 'Blue'): 51115,
('ESPORTSTMNT01_2701489', 'Red'): 56249,
('ESPORTSTMNT01_2701519', 'Blue'): 70061,
('ESPORTSTMNT01_2701519', 'Red'): 52518,
('ESPORTSTMNT01_2701535', 'Blue'): 68849,
('ESPORTSTMNT01_2701535', 'Red'): 79952,
('ESPORTSTMNT01_2701537', 'Blue'): 49303,
('ESPORTSTMNT01_2701537', 'Red'): 38076,
('ESPORTSTMNT01_2701547', 'Blue'): 68167,
('ESPORTSTMNT01_2701547', 'Red'): 60805,
('ESPORTSTMNT01_2701548', 'Blue'): 55710,
('ESPORTSTMNT01_2701548', 'Red'): 53436,
('ESPORTSTMNT01_2701556', 'Blue'): 44149,
('ESPORTSTMNT01_2701556', 'Red'): 32993,
('ESPORTSTMNT01_2701557', 'Blue'): 50539,
('ESPORTSTMNT01_2701557', 'Red'): 60630,
('ESPORTSTMNT01_2701622', 'Blue'): 48371,
('ESPORTSTMNT01_2701622', 'Red'): 55475,
('ESPORTSTMNT01_2701805', 'Blue'): 70737,
('ESPORTSTMNT01_2701805', 'Red'): 80624,
('ESPORTSTMNT01_2701892', 'Blue'): 64405,
('ESPORTSTMNT01_2701892', 'Red'): 58116,
('ESPORTSTMNT01_2701898', 'Blue'): 80479,
('ESPORTSTMNT01_2701898', 'Red'): 92555,
('ESPORTSTMNT01_2701905', 'Blue'): 67197,
('ESPORTSTMNT01_2701905', 'Red'): 52458,
('ESPORTSTMNT01_2701917', 'Blue'): 71714,
```

```
('ESPORTSTMNT01_2701917', 'Red'): 60955,
('ESPORTSTMNT01_2701941', 'Blue'): 59460,
('ESPORTSTMNT01_2701941', 'Red'): 49296,
('ESPORTSTMNT01_2701966', 'Blue'): 52525,
('ESPORTSTMNT01_2701966', 'Red'): 41451,
('ESPORTSTMNT01_2701968', 'Blue'): 58171,
('ESPORTSTMNT01_2701968', 'Red'): 72584,
('ESPORTSTMNT01_2701997', 'Blue'): 48618,
('ESPORTSTMNT01_2701997', 'Red'): 59968,
('ESPORTSTMNT01_2702177', 'Blue'): 49557,
('ESPORTSTMNT01_2702177', 'Red'): 37977,
('ESPORTSTMNT01_2702184', 'Blue'): 70883,
('ESPORTSTMNT01_2702184', 'Red'): 75024,
('ESPORTSTMNT01_2702196', 'Blue'): 64210,
('ESPORTSTMNT01_2702196', 'Red'): 66533,
('ESPORTSTMNT01_2702203', 'Blue'): 35188,
('ESPORTSTMNT01_2702203', 'Red'): 51398,
('ESPORTSTMNT01_2702254', 'Blue'): 62902,
('ESPORTSTMNT01_2702254', 'Red'): 54301,
('ESPORTSTMNT01_2702297', 'Blue'): 69908,
('ESPORTSTMNT01_2702297', 'Red'): 67756,
('ESPORTSTMNT01_2702309', 'Blue'): 67073,
('ESPORTSTMNT01_2702309', 'Red'): 52857,
('ESPORTSTMNT01_2702317', 'Blue'): 68151,
('ESPORTSTMNT01_2702317', 'Red'): 62844,
('ESPORTSTMNT01_2702340', 'Blue'): 85927,
('ESPORTSTMNT01_2702340', 'Red'): 83643,
('ESPORTSTMNT01_2702406', 'Blue'): 46504,
('ESPORTSTMNT01_2702406', 'Red'): 59245,
('ESPORTSTMNT01_2702451', 'Blue'): 68491,
('ESPORTSTMNT01_2702451', 'Red'): 63449,
('ESPORTSTMNT01_2702518', 'Blue'): 63402,
('ESPORTSTMNT01_2702518', 'Red'): 53927,
('ESPORTSTMNT01_2702884', 'Blue'): 63980,
('ESPORTSTMNT01_2702884', 'Red'): 57966,
('ESPORTSTMNT01_2702905', 'Blue'): 55958,
('ESPORTSTMNT01_2702905', 'Red'): 59946,
('ESPORTSTMNT01_2702921', 'Blue'): 77386,
('ESPORTSTMNT01_2702921', 'Red'): 72588,
('ESPORTSTMNT01_2702939', 'Blue'): 69734,
('ESPORTSTMNT01_2702939', 'Red'): 70597,
('ESPORTSTMNT01_2702946', 'Blue'): 56792,
('ESPORTSTMNT01_2702946', 'Red'): 47913,
('ESPORTSTMNT01_2702954', 'Blue'): 51522,
('ESPORTSTMNT01_2702954', 'Red'): 66152,
('ESPORTSTMNT01_2702962', 'Blue'): 69960,
('ESPORTSTMNT01_2702962', 'Red'): 60456,
('ESPORTSTMNT01_2702995', 'Blue'): 53798,
('ESPORTSTMNT01_2702995', 'Red'): 67225,
('ESPORTSTMNT01_2703006', 'Blue'): 53619,
('ESPORTSTMNT01_2703006', 'Red'): 44452,
('ESPORTSTMNT01_2703011', 'Blue'): 62565,
('ESPORTSTMNT01_2703011', 'Red'): 49559,
('ESPORTSTMNT01_2703020', 'Blue'): 44547,
('ESPORTSTMNT01_2703020', 'Red'): 48584,
('ESPORTSTMNT01_2703026', 'Blue'): 72680,
('ESPORTSTMNT01_2703026', 'Red'): 62644,
```

```
 ('ESPORTSTMNT01_2703046', 'Blue'): 64232,
 ('ESPORTSTMNT01_2703046', 'Red'): 68495,
 ('ESPORTSTMNT01_2703056', 'Blue'): 48439,
 ('ESPORTSTMNT01_2703056', 'Red'): 58494,
 ('ESPORTSTMNT01_2703062', 'Blue'): 64296,
 ('ESPORTSTMNT01_2703062', 'Red'): 58146,
 ('ESPORTSTMNT01_2703103', 'Blue'): 58703,
 ('ESPORTSTMNT01_2703103', 'Red'): 44636,
 ('ESPORTSTMNT01_2703104', 'Blue'): 48600,
 ('ESPORTSTMNT01_2703104', 'Red'): 58208,
 ('ESPORTSTMNT01_2703128', 'Blue'): 51179,
 ('ESPORTSTMNT01_2703128', 'Red'): 63254,
 ('ESPORTSTMNT01_2703174', 'Blue'): 38171,
 ('ESPORTSTMNT01_2703174', 'Red'): 53216,
 ('ESPORTSTMNT01_2703192', 'Blue'): 54941,
 ('ESPORTSTMNT01_2703192', 'Red'): 39273,
 ('ESPORTSTMNT01_2703264', 'Blue'): 73537,
 ('ESPORTSTMNT01_2703264', 'Red'): 64982,
 ('ESPORTSTMNT01_2703280', 'Blue'): 63604,
 ('ESPORTSTMNT01_2703280', 'Red'): 53242,
 ('ESPORTSTMNT01_2703294', 'Blue'): 66622,
 ('ESPORTSTMNT01_2703294', 'Red'): 72787,
 ('ESPORTSTMNT01_2703309', 'Blue'): 66933,
 ('ESPORTSTMNT01_2703309', 'Red'): 55967,
 ('ESPORTSTMNT01_2703335', 'Blue'): 54072,
 ('ESPORTSTMNT01_2703335', 'Red'): 42864,
 ('ESPORTSTMNT01_2703490', 'Blue'): 45694,
 ('ESPORTSTMNT01_2703490', 'Red'): 58233,
 ('ESPORTSTMNT01_2703516', 'Blue'): 65231,
 ('ESPORTSTMNT01_2703516', 'Red'): 76845,
 ('ESPORTSTMNT01_2703585', 'Blue'): 46591,
 ('ESPORTSTMNT01_2703585', 'Red'): 53683,
 ('ESPORTSTMNT01_2703605', 'Blue'): 54070,
 ('ESPORTSTMNT01_2703605', 'Red'): 63303,
 ('ESPORTSTMNT01_2703652', 'Blue'): 40852,
 ('ESPORTSTMNT01_2703652', 'Red'): 60038,
 ('ESPORTSTMNT01_2703653', 'Blue'): 80007,
 ('ESPORTSTMNT01_2703653', 'Red'): 85489,
 ('ESPORTSTMNT01_2703669', 'Blue'): 66113,
 ('ESPORTSTMNT01_2703669', 'Red'): 51058,
 ('ESPORTSTMNT01_2703674', 'Blue'): 56656,
 ('ESPORTSTMNT01_2703674', 'Red'): 63698,
 ('ESPORTSTMNT01_2703696', 'Blue'): 41393,
 ('ESPORTSTMNT01_2703696', 'Red'): 56391,
 ('ESPORTSTMNT01_2703721', 'Blue'): 51920,
 ('ESPORTSTMNT01_2703721', 'Red'): 61970,
 ('ESPORTSTMNT01_2703805', 'Blue'): 70724,
 ('ESPORTSTMNT01_2703805', 'Red'): 61626,
 ('ESPORTSTMNT01_2703811', 'Blue'): 67031,
 ('ESPORTSTMNT01_2703811', 'Red'): 57621,
 ('ESPORTSTMNT01_2703952', 'Blue'): 46025,
 ('ESPORTSTMNT01_2703952', 'Red'): 60679,
 ('ESPORTSTMNT01_2704054', 'Blue'): 54151,
 ('ESPORTSTMNT01_2704054', 'Red'): 65548,
 ('ESPORTSTMNT01_2704267', 'Blue'): 68532,
 ('ESPORTSTMNT01_2704267', 'Red'): 62163,
 ('ESPORTSTMNT01_2704268', 'Blue'): 54466,
```

```
('ESPORTSTMNT01_2704268', 'Red'): 61553,
('ESPORTSTMNT01_2704270', 'Blue'): 54500,
('ESPORTSTMNT01_2704270', 'Red'): 61100,
('ESPORTSTMNT01_2704279', 'Blue'): 50299,
('ESPORTSTMNT01_2704279', 'Red'): 58390,
('ESPORTSTMNT01_2704285', 'Blue'): 46838,
('ESPORTSTMNT01_2704285', 'Red'): 56513,
('ESPORTSTMNT01_2704485', 'Blue'): 59012,
('ESPORTSTMNT01_2704485', 'Red'): 63470,
('ESPORTSTMNT01_2704495', 'Blue'): 56375,
('ESPORTSTMNT01_2704495', 'Red'): 48444,
('ESPORTSTMNT01_2704544', 'Blue'): 46824,
('ESPORTSTMNT01_2704544', 'Red'): 59408,
('ESPORTSTMNT01_2704545', 'Blue'): 65938,
('ESPORTSTMNT01_2704545', 'Red'): 52363,
('ESPORTSTMNT01_2704600', 'Blue'): 52297,
('ESPORTSTMNT01_2704600', 'Red'): 59956,
('ESPORTSTMNT01_2704602', 'Blue'): 57987,
('ESPORTSTMNT01_2704602', 'Red'): 67355,
('ESPORTSTMNT01_2704633', 'Blue'): 49633,
('ESPORTSTMNT01_2704633', 'Red'): 57794,
('ESPORTSTMNT01_2704649', 'Blue'): 47657,
('ESPORTSTMNT01_2704649', 'Red'): 58746,
('ESPORTSTMNT01_2704672', 'Blue'): 56598,
('ESPORTSTMNT01_2704672', 'Red'): 66524,
('ESPORTSTMNT01_2704690', 'Blue'): 74982,
('ESPORTSTMNT01_2704690', 'Red'): 56484,
('ESPORTSTMNT01_2704707', 'Blue'): 55822,
('ESPORTSTMNT01_2704707', 'Red'): 65483,
('ESPORTSTMNT01_2704726', 'Blue'): 53863,
('ESPORTSTMNT01_2704726', 'Red'): 60567,
('ESPORTSTMNT01_2704760', 'Blue'): 52177,
('ESPORTSTMNT01_2704760', 'Red'): 39194,
('ESPORTSTMNT01_2704778', 'Blue'): 68133,
('ESPORTSTMNT01_2704778', 'Red'): 72493,
('ESPORTSTMNT01_2704797', 'Blue'): 45645,
('ESPORTSTMNT01_2704797', 'Red'): 56810,
('ESPORTSTMNT01_2704815', 'Blue'): 52422,
('ESPORTSTMNT01_2704815', 'Red'): 40409,
('ESPORTSTMNT01_2704883', 'Blue'): 53902,
('ESPORTSTMNT01_2704883', 'Red'): 44312,
('ESPORTSTMNT01_2704885', 'Blue'): 67674,
('ESPORTSTMNT01_2704885', 'Red'): 56128,
('ESPORTSTMNT01_2704886', 'Blue'): 53997,
('ESPORTSTMNT01_2704886', 'Red'): 66450,
('ESPORTSTMNT01_2704900', 'Blue'): 51343,
('ESPORTSTMNT01_2704900', 'Red'): 62310,
('ESPORTSTMNT01_2704920', 'Blue'): 53670,
('ESPORTSTMNT01_2704920', 'Red'): 59347,
('ESPORTSTMNT01_2704929', 'Blue'): 55192,
('ESPORTSTMNT01_2704929', 'Red'): 65576,
('ESPORTSTMNT01_2705039', 'Blue'): 61583,
('ESPORTSTMNT01_2705039', 'Red'): 52912,
('ESPORTSTMNT01_2705072', 'Blue'): 68848,
('ESPORTSTMNT01_2705072', 'Red'): 79756,
('ESPORTSTMNT01_2705102', 'Blue'): 61413,
('ESPORTSTMNT01_2705102', 'Red'): 50248,
```

```
('ESPORTSTMNT01_2705156', 'Blue'): 43933,
('ESPORTSTMNT01_2705156', 'Red'): 59623,
('ESPORTSTMNT01_2705190', 'Blue'): 53217,
('ESPORTSTMNT01_2705190', 'Red'): 66470,
('ESPORTSTMNT01_2705385', 'Blue'): 48660,
('ESPORTSTMNT01_2705385', 'Red'): 44931,
('ESPORTSTMNT01_2705675', 'Blue'): 45897,
('ESPORTSTMNT01_2705675', 'Red'): 57837,
('ESPORTSTMNT01_2705762', 'Blue'): 47182,
('ESPORTSTMNT01_2705762', 'Red'): 59451,
('ESPORTSTMNT01_2705832', 'Blue'): 54168,
('ESPORTSTMNT01_2705832', 'Red'): 42745,
('ESPORTSTMNT01_2705845', 'Blue'): 60212,
('ESPORTSTMNT01_2705845', 'Red'): 46864,
('ESPORTSTMNT01_2705859', 'Blue'): 47212,
('ESPORTSTMNT01_2705859', 'Red'): 61003,
('ESPORTSTMNT01_2705912', 'Blue'): 52397,
('ESPORTSTMNT01_2705912', 'Red'): 63409,
('ESPORTSTMNT01_2705943', 'Blue'): 39175,
('ESPORTSTMNT01_2705943', 'Red'): 45382,
('ESPORTSTMNT01_2705951', 'Blue'): 63992,
('ESPORTSTMNT01_2705951', 'Red'): 69709,
('ESPORTSTMNT01_2705978', 'Blue'): 81313,
('ESPORTSTMNT01_2705978', 'Red'): 83529,
('ESPORTSTMNT01_2706102', 'Blue'): 55485,
('ESPORTSTMNT01_2706102', 'Red'): 38615,
('ESPORTSTMNT01_2706138', 'Blue'): 46308,
('ESPORTSTMNT01_2706138', 'Red'): 54656,
('ESPORTSTMNT01_2706200', 'Blue'): 58901,
('ESPORTSTMNT01_2706200', 'Red'): 64089,
('ESPORTSTMNT01_2706227', 'Blue'): 48703,
('ESPORTSTMNT01_2706227', 'Red'): 39325,
('ESPORTSTMNT01_2706258', 'Blue'): 43782,
('ESPORTSTMNT01_2706258', 'Red'): 53909,
('ESPORTSTMNT01_2706264', 'Blue'): 42288,
('ESPORTSTMNT01_2706264', 'Red'): 53792,
('ESPORTSTMNT01_2706274', 'Blue'): 62177,
('ESPORTSTMNT01_2706274', 'Red'): 50620,
('ESPORTSTMNT01_2706287', 'Blue'): 72128,
('ESPORTSTMNT01_2706287', 'Red'): 60085,
('ESPORTSTMNT01_2706306', 'Blue'): 58817,
('ESPORTSTMNT01_2706306', 'Red'): 46752,
('ESPORTSTMNT01_2706355', 'Blue'): 40532,
('ESPORTSTMNT01_2706355', 'Red'): 54337,
('ESPORTSTMNT01_2706360', 'Blue'): 63554,
('ESPORTSTMNT01_2706360', 'Red'): 46327,
('ESPORTSTMNT01_2706368', 'Blue'): 75262,
('ESPORTSTMNT01_2706368', 'Red'): 63567,
('ESPORTSTMNT01_2706386', 'Blue'): 54881,
('ESPORTSTMNT01_2706386', 'Red'): 45043,
('ESPORTSTMNT01_2706472', 'Blue'): 52139,
('ESPORTSTMNT01_2706472', 'Red'): 48212,
('ESPORTSTMNT01_2706474', 'Blue'): 58372,
('ESPORTSTMNT01_2706474', 'Red'): 70145,
('ESPORTSTMNT01_2706477', 'Blue'): 52094,
('ESPORTSTMNT01_2706477', 'Red'): 41755,
('ESPORTSTMNT01_2706506', 'Blue'): 59952,
```

```
('ESPORTSTMNT01_2706506', 'Red'): 50419,
('ESPORTSTMNT01_2706524', 'Blue'): 59348,
('ESPORTSTMNT01_2706524', 'Red'): 67265,
('ESPORTSTMNT01_2706543', 'Blue'): 47747,
('ESPORTSTMNT01_2706543', 'Red'): 35852,
('ESPORTSTMNT01_2706561', 'Blue'): 59978,
('ESPORTSTMNT01_2706561', 'Red'): 71437,
('ESPORTSTMNT01_2706565', 'Blue'): 55681,
('ESPORTSTMNT01_2706565', 'Red'): 46702,
('ESPORTSTMNT01_2706572', 'Blue'): 32508,
('ESPORTSTMNT01_2706572', 'Red'): 42630,
('ESPORTSTMNT01_2706576', 'Blue'): 68729,
('ESPORTSTMNT01_2706576', 'Red'): 56334,
('ESPORTSTMNT01_2706603', 'Blue'): 75514,
('ESPORTSTMNT01_2706603', 'Red'): 68679,
('ESPORTSTMNT01_2706605', 'Blue'): 63264,
('ESPORTSTMNT01_2706605', 'Red'): 65598,
('ESPORTSTMNT01_2706607', 'Blue'): 58741,
('ESPORTSTMNT01_2706607', 'Red'): 46631,
('ESPORTSTMNT01_2706611', 'Blue'): 48440,
('ESPORTSTMNT01_2706611', 'Red'): 63835,
('ESPORTSTMNT01_2706637', 'Blue'): 59125,
('ESPORTSTMNT01_2706637', 'Red'): 49783,
('ESPORTSTMNT01_2706640', 'Blue'): 73628,
('ESPORTSTMNT01_2706640', 'Red'): 64041,
('ESPORTSTMNT01_2706652', 'Blue'): 56586,
('ESPORTSTMNT01_2706652', 'Red'): 40746,
('ESPORTSTMNT01_2706657', 'Blue'): 73255,
('ESPORTSTMNT01_2706657', 'Red'): 69487,
('ESPORTSTMNT01_2706674', 'Blue'): 44665,
('ESPORTSTMNT01_2706674', 'Red'): 60489,
('ESPORTSTMNT01_2706703', 'Blue'): 49823,
('ESPORTSTMNT01_2706703', 'Red'): 62834,
('ESPORTSTMNT01_2706723', 'Blue'): 62589,
('ESPORTSTMNT01_2706723', 'Red'): 48632,
('ESPORTSTMNT01_2706724', 'Blue'): 51678,
('ESPORTSTMNT01_2706724', 'Red'): 56674,
('ESPORTSTMNT01_2706729', 'Blue'): 56316,
('ESPORTSTMNT01_2706729', 'Red'): 49723,
('ESPORTSTMNT01_2706743', 'Blue'): 47168,
('ESPORTSTMNT01_2706743', 'Red'): 34526,
('ESPORTSTMNT01_2706746', 'Blue'): 36253,
('ESPORTSTMNT01_2706746', 'Red'): 50804,
('ESPORTSTMNT01_2706748', 'Blue'): 46229,
('ESPORTSTMNT01_2706748', 'Red'): 56080,
('ESPORTSTMNT01_2706768', 'Blue'): 49097,
('ESPORTSTMNT01_2706768', 'Red'): 58370,
('ESPORTSTMNT01_2706770', 'Blue'): 55259,
('ESPORTSTMNT01_2706770', 'Red'): 43090,
('ESPORTSTMNT01_2706774', 'Blue'): 27380,
('ESPORTSTMNT01_2706774', 'Red'): 42846,
('ESPORTSTMNT01_2706776', 'Blue'): 53391,
('ESPORTSTMNT01_2706776', 'Red'): 39940,
('ESPORTSTMNT01_2706813', 'Blue'): 67114,
('ESPORTSTMNT01_2706813', 'Red'): 61818,
('ESPORTSTMNT01_2706815', 'Blue'): 64442,
('ESPORTSTMNT01_2706815', 'Red'): 69648,
```

```
('ESPORTSTMNT01_2707081', 'Blue'): 51467,
('ESPORTSTMNT01_2707081', 'Red'): 59694,
('ESPORTSTMNT01_2707138', 'Blue'): 58911,
('ESPORTSTMNT01_2707138', 'Red'): 68791,
('ESPORTSTMNT01_2707147', 'Blue'): 47055,
('ESPORTSTMNT01_2707147', 'Red'): 36366,
('ESPORTSTMNT01_2707168', 'Blue'): 47417,
('ESPORTSTMNT01_2707168', 'Red'): 55144,
('ESPORTSTMNT01_2707174', 'Blue'): 40810,
('ESPORTSTMNT01_2707174', 'Red'): 53102,
('ESPORTSTMNT01_2707191', 'Blue'): 46451,
('ESPORTSTMNT01_2707191', 'Red'): 55605,
('ESPORTSTMNT01_2707195', 'Blue'): 42546,
('ESPORTSTMNT01_2707195', 'Red'): 53118,
('ESPORTSTMNT01_2707199', 'Blue'): 49218,
('ESPORTSTMNT01_2707199', 'Red'): 39999,
('ESPORTSTMNT01_2707204', 'Blue'): 51042,
('ESPORTSTMNT01_2707204', 'Red'): 60516,
('ESPORTSTMNT01_2707208', 'Blue'): 48529,
('ESPORTSTMNT01_2707208', 'Red'): 61537,
('ESPORTSTMNT01_2707209', 'Blue'): 57389,
('ESPORTSTMNT01_2707209', 'Red'): 49422,
('ESPORTSTMNT01_2707330', 'Blue'): 58405,
('ESPORTSTMNT01_2707330', 'Red'): 41716,
('ESPORTSTMNT01_2707372', 'Blue'): 46522,
('ESPORTSTMNT01_2707372', 'Red'): 63886,
('ESPORTSTMNT01_2707382', 'Blue'): 58336,
('ESPORTSTMNT01_2707382', 'Red'): 66969,
('ESPORTSTMNT01_2707411', 'Blue'): 63384,
('ESPORTSTMNT01_2707411', 'Red'): 73695,
('ESPORTSTMNT01_2707434', 'Blue'): 59434,
('ESPORTSTMNT01_2707434', 'Red'): 47205,
('ESPORTSTMNT01_2707441', 'Blue'): 59403,
('ESPORTSTMNT01_2707441', 'Red'): 63261,
('ESPORTSTMNT01_2707442', 'Blue'): 87579,
('ESPORTSTMNT01_2707442', 'Red'): 77759,
('ESPORTSTMNT01_2707450', 'Blue'): 55922,
('ESPORTSTMNT01_2707450', 'Red'): 66873,
('ESPORTSTMNT01_2707466', 'Blue'): 67989,
('ESPORTSTMNT01_2707466', 'Red'): 58360,
('ESPORTSTMNT01_2707515', 'Blue'): 56608,
('ESPORTSTMNT01_2707515', 'Red'): 69310,
('ESPORTSTMNT01_2707529', 'Blue'): 44566,
('ESPORTSTMNT01_2707529', 'Red'): 57113,
('ESPORTSTMNT01_2707537', 'Blue'): 65817,
('ESPORTSTMNT01_2707537', 'Red'): 53840,
('ESPORTSTMNT01_2707557', 'Blue'): 62639,
('ESPORTSTMNT01_2707557', 'Red'): 66810,
('ESPORTSTMNT01_2707568', 'Blue'): 62787,
('ESPORTSTMNT01_2707568', 'Red'): 46223,
('ESPORTSTMNT01_2707629', 'Blue'): 58442,
('ESPORTSTMNT01_2707629', 'Red'): 48778,
('ESPORTSTMNT01_2707660', 'Blue'): 64371,
('ESPORTSTMNT01_2707660', 'Red'): 71811,
('ESPORTSTMNT01_2707669', 'Blue'): 63003,
('ESPORTSTMNT01_2707669', 'Red'): 75376,
('ESPORTSTMNT01_2707693', 'Blue'): 46560,
```

```
 ('ESPORTSTMNT01_2707693', 'Red'): 61007,
 ('ESPORTSTMNT01_2707715', 'Blue'): 67236,
 ('ESPORTSTMNT01_2707715', 'Red'): 76878,
 ('ESPORTSTMNT01_2707748', 'Blue'): 72433,
 ('ESPORTSTMNT01_2707748', 'Red'): 80351,
 ('ESPORTSTMNT01_2707756', 'Blue'): 50062,
 ('ESPORTSTMNT01_2707756', 'Red'): 37460,
 ('ESPORTSTMNT01_2707791', 'Blue'): 79905,
 ('ESPORTSTMNT01_2707791', 'Red'): 78536,
 ('ESPORTSTMNT01_2707857', 'Blue'): 62753,
 ('ESPORTSTMNT01_2707857', 'Red'): 45539,
 ('ESPORTSTMNT01_2707893', 'Blue'): 81271,
 ('ESPORTSTMNT01_2707893', 'Red'): 82554,
 ('ESPORTSTMNT01_2707926', 'Blue'): 30386,
 ('ESPORTSTMNT01_2707926', 'Red'): 46418,
 ('ESPORTSTMNT01_2707931', 'Blue'): 48339,
 ('ESPORTSTMNT01_2707931', 'Red'): 56781,
 ('ESPORTSTMNT01_2707936', 'Blue'): 54344,
 ('ESPORTSTMNT01_2707936', 'Red'): 54278,
 ('ESPORTSTMNT01_2708021', 'Blue'): 48222,
 ('ESPORTSTMNT01_2708021', 'Red'): 53447,
 ('ESPORTSTMNT01_2708024', 'Blue'): 72258,
 ('ESPORTSTMNT01_2708024', 'Red'): 67708,
 ('ESPORTSTMNT01_2708032', 'Blue'): 64362,
 ('ESPORTSTMNT01_2708032', 'Red'): 66855,
 ('ESPORTSTMNT01_2708036', 'Blue'): 59833,
 ('ESPORTSTMNT01_2708036', 'Red'): 49229,
 ('ESPORTSTMNT01_2708038', 'Blue'): 70509,
 ('ESPORTSTMNT01_2708038', 'Red'): 63421,
 ('ESPORTSTMNT01_2708044', 'Blue'): 39925,
 ('ESPORTSTMNT01_2708044', 'Red'): 58936,
 ...}
```

In [37]:
```
#Helper function to calculate difference between team's totalgold and opponent
def helper(gameid, side):
    total_gold = dic[(gameid, side)]
    if side == 'Blue':
        opp_gold = dic[(gameid, 'Red')]

    else:
        opp_gold = dic[(gameid, 'Blue')]

    return total_gold - opp_gold
```

In [43]:
```
#Add column for each team's totalgolddiff in the game
team_stats['totalgolddiff'] = team_stats.apply(lambda x: helper(x['gameid'], x
team_stats[['totalgolddiff']]
```

Out[43]:

	totalgolddiff
10	-5547
11	5547
22	-13375
23	13375
46	10691
...	...
148115	-18584
148126	2038
148127	-2038
148138	9960
148139	-9960

21206 rows × 1 columns

In [42]:
```
#Add column calculating difference between team's kills and opposing team's ki
team_stats['killdiff'] = team_stats['kills'] - team_stats['deaths']
team_stats[['killdiff']]
```

Out[42]:

	killdiff
10	-10
11	10
22	-13
23	13
46	9
...	...
148115	-23
148126	7
148127	-7
148138	15
148139	-15

21206 rows × 1 columns

In [78]:
```
#Features we care about for our model
team_stats[['gamelength', 'side','dragons','heralds', 'barons', 'towers','vspm
```

Out[78]:

	gamelength	side	dragons	heralds	barons	towers	vspm	earned gpm	killdiff	total
10	1713	Blue	1.0	2.0	0.0	3.0	6.9002	988.5114	-10	
11	1713	Red	3.0	0.0	0.0	6.0	7.1804	1182.8021	10	
22	2114	Blue	1.0	1.0	0.0	3.0	7.8619	984.5222	-13	
23	2114	Red	4.0	1.0	2.0	11.0	9.8202	1364.1343	13	
46	1972	Blue	4.0	1.0	1.0	11.0	8.0325	1258.7830	9	
...	...	...	...	...	...	...	...	...	...	
148115	1501	Red	0.0	1.0	0.0	0.0	5.9161	723.4777	-23	
148126	2117	Blue	4.0	1.0	1.0	8.0	6.3769	1141.2187	7	
148127	2117	Red	1.0	1.0	1.0	8.0	5.1582	1083.4577	-7	
148138	1388	Blue	2.0	0.0	1.0	8.0	4.5821	1367.8963	15	
148139	1388	Red	1.0	2.0	0.0	2.0	4.7983	937.3487	-15	

21206 rows × 11 columns

In [81]:
```
#Split data into training and test sets
team_stats = team_stats[team_stats['datacompleteness'] == 'complete']

#Get unique gameid
unique_games = team_stats.groupby('gameid').count()[['datacompleteness']]

#Split data into train and testing based on gameid
X = unique_games.drop('datacompleteness', axis = 1)
y = unique_games['datacompleteness']
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size = 0.25)

#Merge to get data for both teams in that game and only get relevant columns fo
train = team_stats.merge(X_train.reset_index(), right_on='gameid', left_on='gam
test = team_stats.merge(X_test.reset_index(), right_on='gameid', left_on='game

#Keep relevant columns for features and result to get final training and testi
X_train = train.drop('result', axis=1)
y_train = train['result']
X_test = test.drop('result', axis=1)
y_test = test['result']
```

In [83]:
```
#Column transformer
preproc = ColumnTransformer(transformers = [('side_ohe', OneHotEncoder(), ['sid

#Create pipeline using k-nearest neighbors
pl = Pipeline([('preprocessor', preproc),('knn',KNeighborsClassifier())])

#Fit on training set
pl.fit(X_train.drop('gamelength', axis=1), y_train)
```

Out[83]:
```
Pipeline(steps=[('preprocessor',
                 ColumnTransformer(remainder='passthrough',
                                   transformers=[('side_ohe', OneHotEncoder(),
                                                  ['side'])])),
                ('knn', KNeighborsClassifier())])
```

In [87]:
```
#Test new model accuracy on training set
train_pred = pl.predict(X_train)
(train_pred == y_train).mean()
```

```
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
```

Out[87]:
```
0.9837147887323944
```

In [88]:
```
#Test new model accuracy on test set
test_pred = pl.predict(X_test)
(test_pred == y_test).mean()
```

```
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
```

Out[88]:
```
0.97736703130894
```

In [89]:
```
#Test to see what hyperparameter is best to use (number of neighbors)

neigh_acc = {}
for i in range(1,30):
    pl = Pipeline([('preprocessor', preproc),('knn',KNeighborsClassifier(n_neig
    pl.fit(X_train.drop('gamelength', axis=1), y_train)
    pred = pl.predict(X_test)
    neigh_acc[str(i) + ' neighbors'] = str(round((pred == y_test).mean(), 4) *
```

```
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
```

```
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
```

```
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
```

```
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
/opt/anaconda3/envs/dsc80/lib/python3.8/site-packages/sklearn/neighbors/_class
ification.py:230: FutureWarning: Unlike other reduction functions (e.g. `skew
`, `kurtosis`), the default behavior of `mode` typically preserves the axis it
acts along. In SciPy 1.11.0, this behavior will change: the default value of `
```

```
keepdims` will become False, the `axis` over which the statistic is taken will
be eliminated, and the value None will no longer be accepted. Set `keepdims` t
o True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
```

In [90]:
```
#Model accuracy based on number of neighbors used
neigh_acc
```

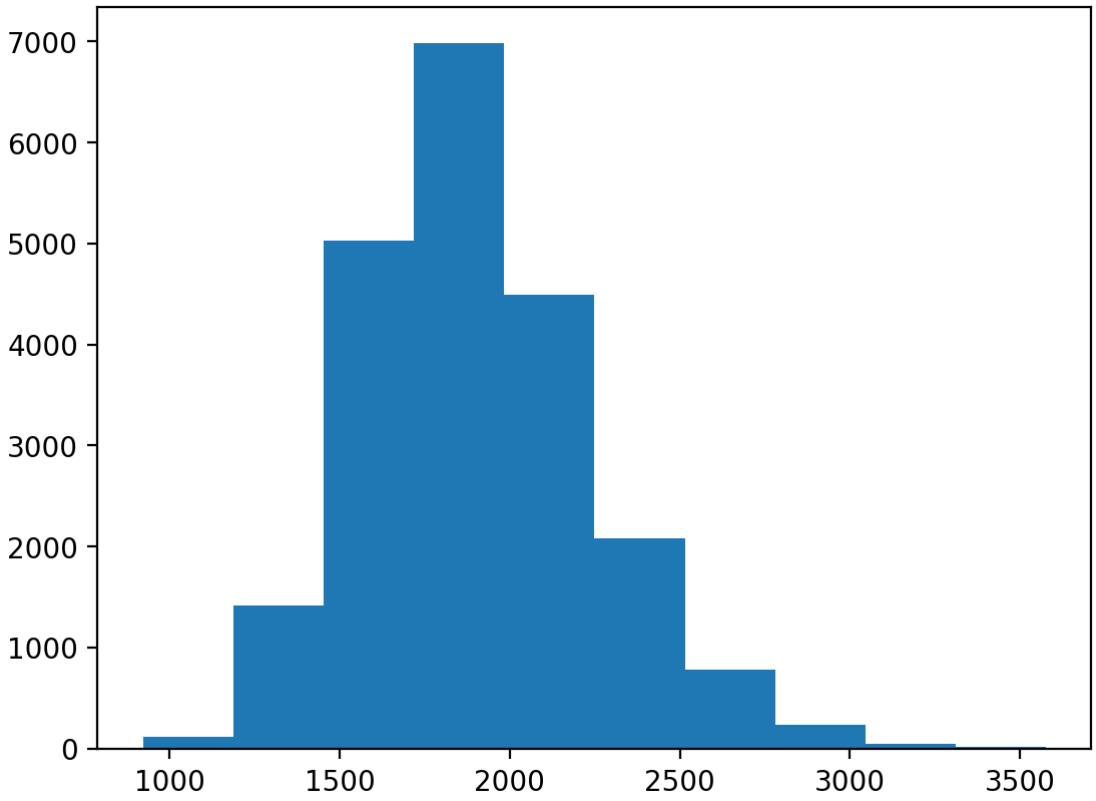
Out[90]:
```
{'1 neighbors': '96.98%',
 '2 neighbors': '97.04%',
 '3 neighbors': '97.43%',
 '4 neighbors': '97.42%',
 '5 neighbors': '97.74%',
 '6 neighbors': '97.77%',
 '7 neighbors': '97.83%',
 '8 neighbors': '97.91%',
 '9 neighbors': '98.0%',
 '10 neighbors': '97.96%',
 '11 neighbors': '97.96%',
 '12 neighbors': '97.98%',
 '13 neighbors': '98.02%',
 '14 neighbors': '98.0%',
 '15 neighbors': '97.98%',
 '16 neighbors': '98.04%',
 '17 neighbors': '98.08%',
 '18 neighbors': '98.0%',
 '19 neighbors': '98.02%',
 '20 neighbors': '98.08%',
 '21 neighbors': '98.04%',
 '22 neighbors': '98.02%',
 '23 neighbors': '98.02%',
 '24 neighbors': '98.02%',
 '25 neighbors': '98.06%',
 '26 neighbors': '98.08%',
 '27 neighbors': '98.0%',
 '28 neighbors': '98.04%',
 '29 neighbors': '98.08%'}
```

Fairness Evaluation

In [137…
```
#Import
import matplotlib.pyplot as plt
from sklearn import metrics
```

In [141…
```
#Used mean of gamelength to calculate threshold since distribution is approx n
plt.hist(team_stats['gamelength'])
```

Out[141]:
```
(array([ 118., 1412., 5030., 6988., 4492., 2084.,  784.,  236.,   48.,
          14.]),
 array([ 921. , 1186.6, 1452.2, 1717.8, 1983.4, 2249. , 2514.6, 2780.2,
        3045.8, 3311.4, 3577. ]),
 <BarContainer object of 10 artists>)
```



In [142…
```
#Calculate threshold
threshold = team_stats['gamelength'].mean()
threshold
```

Out[142]: 1901.3369800999717

In [143…
```
#Create column saying whether game was short or long
results = X_test
results['prediction'] = test_pred
results['result'] = y_test
results['is_gamelong'] = (results['gamelength'] < threshold).replace({True: 'sh
results
```

Out[143]:

	gamelength	side	dragons	heralds	barons	towers	vspm	earned gpm	killdiff	totalg
0	1972	Blue	4.0	1.0	1.0	11.0	8.0325	1258.7830	9	
1	1972	Red	1.0	1.0	0.0	2.0	7.2718	933.4990	-9	
2	2488	Blue	4.0	1.0	0.0	5.0	7.1624	1040.2814	3	
3	2488	Red	2.0	1.0	2.0	9.0	9.1640	1252.5965	-3	
4	2910	Blue	5.0	2.0	2.0	8.0	7.6701	1133.0722	-3	
...	...	...	...	...	...	...	...	...	...	
5297	2224	Red	2.0	1.0	2.0	9.0	6.5288	1084.1007	-8	
5298	1501	Blue	3.0	1.0	1.0	10.0	5.5163	1466.3424	23	
5299	1501	Red	0.0	1.0	0.0	0.0	5.9161	723.4777	-23	
5300	1388	Blue	2.0	0.0	1.0	8.0	4.5821	1367.8963	15	
5301	1388	Red	1.0	2.0	0.0	2.0	4.7983	937.3487	-15	

5302 rows × 13 columns

In [106…
```
results['is_gamelong'].value_counts()
```

Out[106]:
```
short    2934
long     2368
Name: is_gamelong, dtype: int64
```

In [148…
```
#Demographic parity
results.groupby('is_gamelong')['prediction'].mean().to_frame()
```

Out[148]:

is_gamelong	prediction
long	0.499578
short	0.500000

In [150…
```
#Accuracy parity
(
    results
    .groupby('is_gamelong')
    .apply(lambda x: metrics.accuracy_score(x['result'], x['prediction']))
    .rename('accuracy')
    .to_frame()
)
```

Out[150]:

is_gamelong	accuracy
long	0.952280
short	0.997614

In [144…
```
#Observed test statistic (Difference in accuracy of long games and short games)
obs = -(results.groupby('is_gamelong').apply(lambda x: metrics.accuracy_score(x
obs
```

Out[144]: -0.045333773190368354

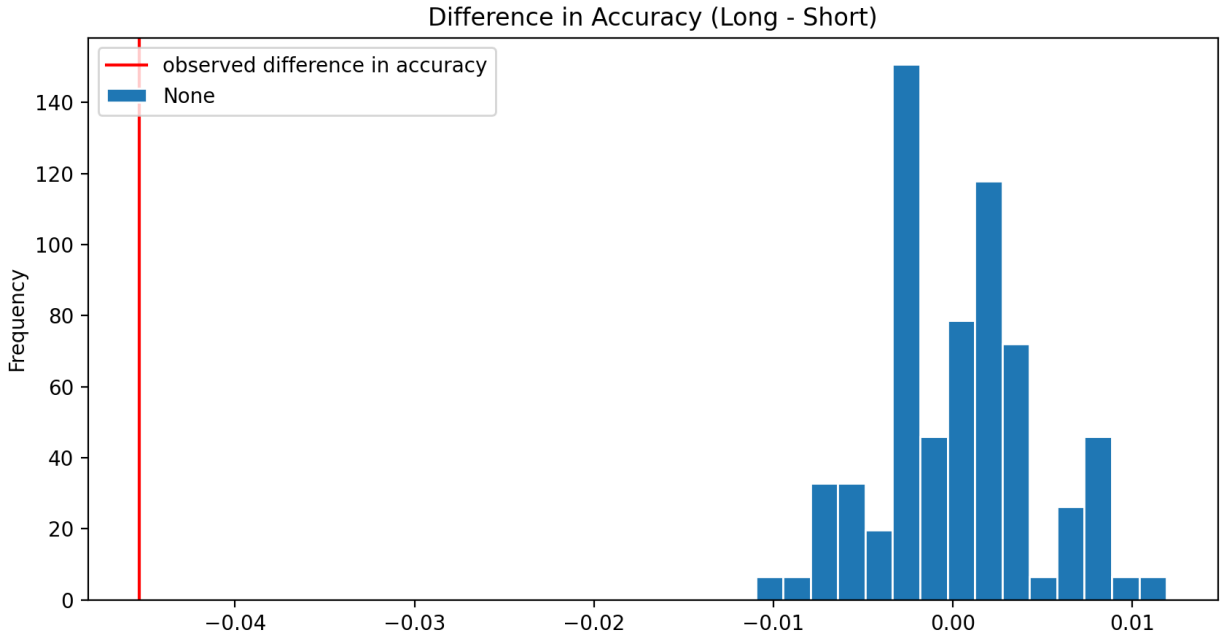
In [145…
```
#Perform permutation test
diff_in_acc = []
for _ in range(100):
    s = (
        results[['is_gamelong', 'prediction', 'result']]
        .assign(is_gamelong=results.is_gamelong.sample(frac=1.0, replace=False)
        .groupby('is_gamelong')
        .apply(lambda x: metrics.accuracy_score(x['result'], x['prediction']))
        .diff()
        .iloc[-1]
    )
    diff_in_acc.append(-s)
diff_in_acc
```

Out[145]:
```
[-0.003361682234381602,
 0.003506478103870725,
 -0.004124811160854169,
 -0.0010722954549641228,
 0.003506478103870725,
 0.0004539623979807894,
 -0.0010722954549641228,
 -0.00183542438143669,
 0.003506478103870725,
 -0.004124811160854169,
 0.0004539623979807894,
 -0.003361682234381602,
 -0.00870358471968901 7,
 -0.0003091665284916667,
 0.0027433491773981578,
 -0.004124811160854169,
 -0.0010722954549641228,
 0.003506478103870725,
 0.009611509515650707,
 0.008085251662705684,
 -0.003361682234381602,
 0.004269607030343181,
 -0.003361682234381602,
 0.0027433491773981578,
 -0.0003091665284916667,
 -0.00870358471968901 7,
 0.005032735956815637,
 0.0027433491773981578,
 0.0027433491773981578,
 -0.0048879400873266254,
 0.0019802202509258127,
 0.005032735956815637,
 -0.003361682234381602,
 -0.00183542438143669,
 -0.0003091665284916667,
 0.0012170913244533565,
 0.004269607030343181,
 0.0019802202509258127,
 -0.00183542438143669,
 -0.0010722954549641228,
 0.0027433491773981578,
 0.0012170913244533565,
 0.0004539623979807894,
 -0.002598553307909146,
 -0.002598553307909146,
 -0.0003091665284916667,
 0.0004539623979807894,
 -0.0056510690137990816,
 -0.003361682234381602,
 -0.004124811160854169,
 0.008085251662705684,
 -0.003361682234381602,
 0.012664025221540531,
 -0.0003091665284916667,
 -0.00794045579321645,
 -0.007177326866743994,
 0.003506478103870725,
```

```
 -0.0056510690137990816,
 0.0004539623979807894,
 0.005795864883288204,
 -0.0056510690137990816,
 0.0027433491773981578,
 0.0019802202509258127,
 -0.003361682234381602,
 -0.002598553307909146,
 0.0019802202509258127,
 0.0027433491773981578,
 -0.002598553307909146,
 0.0073221227362332275,
 -0.0048879400873266254,
 0.0027433491773981578,
 -0.0003091665284916667,
 -0.011756100425578953,
 0.005032735956815637,
 0.0004539623979807894,
 -0.0010722954549641228,
 0.003506478103870725,
 0.003506478103870725,
 0.0027433491773981578,
 0.004269607030343181,
 -0.0003091665284916667,
 -0.002598553307909146,
 -0.002598553307909146,
 -0.0048879400873266254,
 0.0027433491773981578,
 0.00884838058917814,
 0.005795864883288204,
 -0.0056510690137990816,
 -0.004124811160854169,
 0.005795864883288204,
 -0.0048879400873266254,
 0.0012170913244533565,
 0.0019802202509258127,
 0.004269607030343181,
 -0.0056510690137990816,
 -0.0003091665284916667,
 0.0012170913244533565,
 0.0004539623979807894,
 -0.003361682234381602,
 -0.00183542438143669]
```

In [136…

```
plt.figure(figsize=(10, 5))
pd.Series(diff_in_acc).plot(kind='hist', ec='w', density=True, bins=15, title=
plt.axvline(x=obs, color='red', label='observed difference in accuracy')
plt.legend(loc='upper left');
```


Difference in Accuracy (Long - Short)
observed difference in accuracy
None
Frequency
0
20
40
60
80
100
120
140
−0.04
−0.03
−0.02
−0.01
0.00
0.01