Round Winner Classification

Kenzie Baker + Adam Hobson

Introduction to CS:GO

- **CS:GO** is a **tactical shooter video game** that is made up of **two teams of 5 players each.**
  - **Terrorist team**
  - **Counter-Terrorist team**
- The teams battle in a **best-of 30 rounds** competition, with each round being **1 minute and 55 seconds**. After 15 rounds played, the teams switch roles. **The first team to reach 16 wins is named victorious.**
  - The **Terrorist** team wins a round by either **planting the bomb and making sure it explodes**, or by **eliminating the other team.**
  - The **Counter-Terrorist** team wins a round by either **eliminating the other team**, or by **disarming the bomb**, should it have been planted.

https://www.kaggle.com/datasets/christianlillelund/csgo-round-winner-classification?resource=download

The Dataset

Summary of the Experiment

- **The Dataset:**
  - ***122410*** Observations
  - ***96*** Predictors
    - ***94*** Continuous
    - ***2*** Categorical
  - Response Variable : **Round Winner**
- The dataset consists of **screen snapshots** taken during 700 demos from high level tournament play in 2019 and 2020 and have been **recorded every 20 seconds until the round is decided**.
  - There are ***122410*** total snapshots in the dataset.
  - Warm-up rounds and restarts have been filtered out.
- The **Round Winner** is a two-class categorical response that records if the **Counter-Terrorist** team or the **Terrorist** team won the match.

https://www.kaggle.com/datasets/christianlillelund/csgo-round-winner-classification?resource=download

Research Questions

- Based on the dataset, we have decided to explore the following questions through our model fitting and model selection processes:
  - ***What types of machine learning models perform best on this dataset?***
    - Throughout this experiment we will attempt to fit many different classification models in an attempt to find the model that that performs best on the data.
  - ***Which features are most indicative of which teams wins the round?***
    - After finding and validating the best model for the data, we will use the Most Important Predictor functionality to determine which predictors had the largest impact on winning.

https://www.kaggle.com/datasets/christianlillelund/csgo-round-winner-classification?resource=download

Continuous Predictors

The *94 Continuous* predictors in this dataset can be summarized into 4 main categories:

1. Team Scores
    - Indicates the current team scores at the time the snapshot was taken.
    - This category represents 2 predictors.
2. Health and Shield
    - Indicates different levels of total team health, armor, money, helmets, defuse kits, and players still alive for each team at the time the snapshot was taken.
    - This category represents 11 predictors.
3. Weapon Counts
    - Indicates the count of 40 different weapons that each team has at the time the snapshot was taken.
    - This category represents 80 predictors.
4. Time
    - Indicates the time left in the round at the time the snapshot was taken.
    - This category represents 1 predictor.

Categorical Predictors

- ***Bomb Planted*** : Indicates whether bomb has been planted or not.
- ***Map*** : Indicates the current map used during the game.

Categorical Dummy Variables:

- ***Bomb Planted*** : 1 Dummy Variables created : Not Planted = 0 Planted = 1
  - *bomb_planted*
- ***Map*** : 8 Dummy Variables created : Map Not in Use = 0 Map in Use = 1
  - map_de_cache, map_de_dust2, map_de_inferno, map_de_mirage, map_de_nuke, map_de_overpass, map_de_train, map_de_vertigo

Missing Data

- **No missing data in the dataset**
- No need to remove any predictors
- Imputation will not be needed

```
[ ]  # Find NaN values in the DataFrame
     missing_values = data.isna().sum()
     # Print columns with missing values
     print("Columns with missing values:")
     print(missing_values[missing_values > 0])
```

```
Columns with missing values:
Series([], dtype: int64)
```

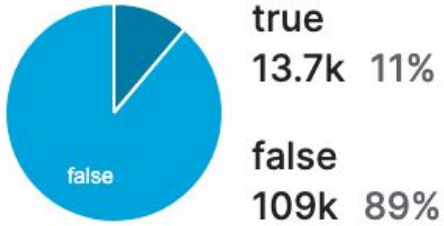
Pre-Processing

Distribution of Continuous Data

- Without being able to show all 94 predictor's distributions, we will summarize the distribution trends based on the 4 categories previously described:
  - **Team Scores:**
    - Both predictors in this category are **single peaked, fairly right skewed data** as both teams begin each round at 0.
  - **Health and Shield:**
    - A **mixture of single peaked and multi-peaked data**, but all 11 predictors in this category are **skewed to one direction or the other**. Multi-peaked data is representative of the fact that if one or more players are eliminated from the round, the team's maximum total health automatically drops.
  - **Weapon Counts:**
    - All 80 predictors in this category are **single peaked**. Many of the distributions have **limited variability**, but some have **strong skew**. This is representative of the fact that players are limited to the number of weapons they can carry.
  - **Time:**
    - The ***time_left*** predictor is **single peaked** and **strongly left skewed** as not all rounds last the full 1 minute and 55 seconds.

Distribution of Categorical Data

- **Bomb Planted:**
  - Boolean predictor with unbalanced responses.
  - Unbalanced distribution displays the difficulty that exists for the Terrorist team to successfully plant the bomb during each round.



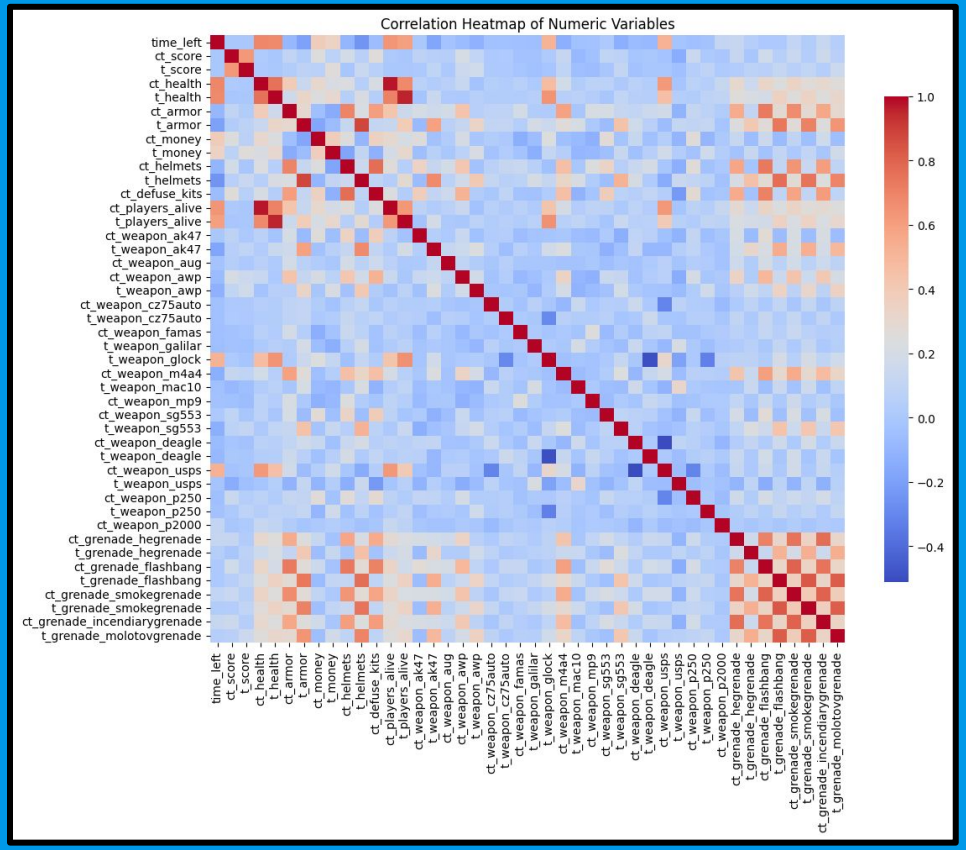
- **Map:**
  - Fairly evenly distributed use of each of the 8 maps in the dataset.
    - The 2 most used maps are *de_inferno* map (19%) and the *de_dust2* map (18%).

de_inferno	19.451842
de_dust2	18.090025
de_nuke	15.542031
de_mirage	15.175231
de_overpass	11.503145
de_train	11.021158
de_vertigo	9.098113
de_cache	0.118454

Near-Zero Variance Predictors

- **Near-Zero Variance predictors are predictors that have only a handful of unique values that occur with very low frequencies**
- **Predictors are removed if:**
  - 1) The fraction of unique values over the sample size is low (10 % or less).
  - 2) The ratio of the frequency of the most prevalent value to the frequency of the second most prevalent value is large (say around 20:1).

- **50 Near-Zero Variance predictors were removed from the dataset.**
  - This removes the predictors that recorded the number of specified weapons that are barely used amongst snapshots in the dataset.
  - All variables that were removed were continuous predictors.



Correlation Matrix

- Appears to have highly correlated predictors based on the matrix.
- We removed 3 highly correlated predictors using a threshold of 0.85 and -0.85.

Skewness Values

- If a skewness value is less than −1 or greater than +1, the distribution is highly skewed.
- **Based on this metric, 23 out of 41 remaining continuous predictors are highly skewed.**
  - There is a skewness issue in the data.
  - May need to perform a Box-Cox transformation to fix this issue.

Outliers

- If a datapoint is 1.5 * IQR above the third quartile or below the first quartile of a predictor's distribution, the datapoint is an outlier.
- **Outliers are naturally present due to the skewness issue.**
- **Based on this metric, 18 out of 41 remaining continuous predictors contain outliers.**
- **There are 105,014 total outliers.**
  - There is an outlier issue in the data.
  - May need to perform Centering, Scaling, and Winsorization transformations to fix this issue.

Transformations

- **Center + Scale** : Skewness + Outliers
- **Box-Cox** : Skewness
- **Winsorization** : Outliers

After Transformations:

Skewness Values

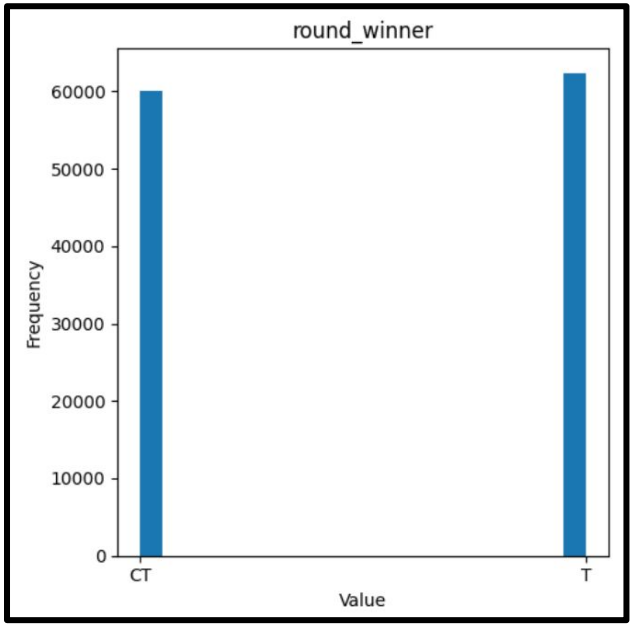
- Before Transformations:
  - 23 out of 41 highly skewed continuous predictors.
- After Transformations:
  - 14 out of 41 highly skewed continuous predictors.
  - The transformations improved the skewness issue!

Outliers

- Before Transformations:
  - 18 out of 41 predictors contain outliers.
  - 105,014 total outliers.
- After Transformations:
  - 9 out of 41 predictors contain outliers.
  - 98,284 total outliers.
  - The transformation improved the outlier issue.

Data Splitting

Random Sampling



- Due to the **balanced data structure** of the two-class Round Winner response variable, we chose to perform the Random Sampling technique to split the data.
  - **80% Training Set**
  - **20% Testing Set**

After Data Splitting & Preprocessing:

Number of Predictors

50

Sample Size

122,410

Training Set Size

97,928

Testing Set Size

24,482

Model Fitting

We chose to fit the linear and nonlinear classification models because our response variable is a categorical measure of two classes. We have chosen to use the Accuracy and F1-Score metrics for evaluating the performance of each model. We used 5-fold cross validation due to the amount of data we had and lack of computational power.

Model Performance Summary:

Models fit on the Training Set:

- Logistic Regression : Accuracy: 0.748 - F1-Score: 0.748
- K-Nearest Neighbor : Accuracy: 0.886 - F1-Score: 0.886
- Neural Networks : Accuracy: 0.794 - F1-Score: 0.741
- Naive Bayes : Accuracy: 0.707 - F1-Score: 0.706
- Decision Tree : Accuracy: 0.790 - F1-Score: 0.790

KNN Prediction on Testing Set

Model Summary

```
Test Accuracy: 0.9028265664569888
Test F1: 0.9028265664569888
Best Hyperparameters: {'metric': 'manhattan', 'n_neighbors': 1, 'weights': 'uniform'}
```

Testing Accuracy

0.902827

Testing F1-Score

0.902827

Decision Tree Prediction on Testing Set

Model Summary

```
Accuracy on the test set: 0.8038150477902132
Best Parameters:  {'max_depth': None, 'min_samples_leaf': 2}
f1 score:  [0.80585311 0.80173375]
f1 score:  0.8038150477902132
```

Testing Accuracy

0.803815

Testing F1-Score

0.803815

Final Model Selection:

K-Nearest Neighbor

Testing Accuracy

0.903

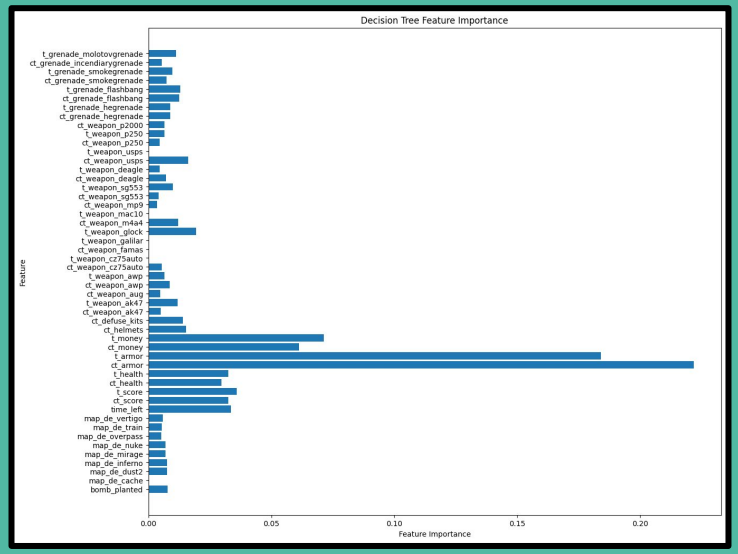
Testing F1-Score

0.903

Feature Importance

Our final model is uniformly weighted which achieves high accuracy but lacks the capability to calculate feature importance due to its uniform weighting. To identify which attributes are most important to win a round, we decided to calculate feature importance using the Decision Tree model which also displayed high performance.

Top 2 Most Important Predictors:



1. ct_armor

2. t_armor

- *ct_armor* : The total armor of all Counter-Terrorist players.
- *t_armor* : The total armor of all Terrorist players.

Summary

We selected the KNN model for our final model because it showed the best performance on the Testing Set for the Accuracy (0.903) and F1-Score (0.903) metrics. Additionally, with the capabilities of the feature importance function, we were able to use the Decision Tree model to describe which predictors (*ct_armor* and *t_armor*) had the largest influence on winning any given CS:GO round. With these results, players can be more strategic and effective with their gameplay and have a better chance to lead their teams to victory.

Thank You