Rocket League 3v3 Win/Loss Prediction on Tertiary Stats

By Nick Hazard

Obtaining the Data

- Taught myself how to write code to query API's
- Pulled data from Ballchasing.com
  - 3rd-party site that stores millions of games of RL and all stats from those games
  - Robust documentation for their API
- Took many attempts and 5+ hour long periods of running queries
- Got 50,000 RL games
  - created a 300,000 row dataframe (6 players per game, each with their own row)
- All 88 or so stats for each player is too long to list here

Predicting Wins

- Can we predict RL game outcomes via only tertiary stats
- Examples of NON-tertiary stats:
  - Goals
  - Shots
  - Saves
  - Assists
- Example of tertiary stats:
  - Boost Used per Minute (BPM)
  - Boost Collected per Minute (BCPM)
  - Percent in Defensive/Neutral/Offensive Third
  - Percent Behind Ball

```
feature_cols = ['bpm', 'bcpm',
       'amount_collected', 'amount_stolen', 'percent_zero_boost',
       'percent_full_boost', 'avg_speed', 'time_powerslide',
       'count_powerslide',
       'percent_slow_speed', 'percent_boost_speed', 'percent_supersonic_speed',
       'percent_ground', 'percent_low_air', 'percent_high_air', 'avg_distance_to_ball_possession',
       'avg_distance_to_ball_no_possession', 'avg_distance_to_mates',
       'percent_defensive_third', 'percent_offensive_third',
       'percent_neutral_third', 'percent_behind_ball', 'percent_infront_ball']
```

Pre-Processing

- Since the data was already numeric with very few nulls, most preprocessing was feature engineering
- Non-feature engineering steps:
  - Loading, flattening and looping through the data's json format to obtain the correct stats
  - Creating our y variable we want to predict as a new 'win' column
    - Via looping through data and applying a function that looked at 'team_color' and 'goals_against' to calculate the new binary 'win' column
  - Very few (<100) nulls present, replaced with median values
  - Converting all numeric columns to float
  - Applying StandardScaler
  - Concatenating the 3 players on each team's stats into a 1D array for input into model
- Some of my EDA Insights:
  - Using the version of stats with percentages instead of straight numeric values is valuable due to the possibility of forfeits in RL games
  - Lots of highly UNcorrelated stats, such as boost stolen, boost overfill, count collected big/small, etc.

Solution

- NN was not the correct framework for this type and size of data
- Logistic Regression worked much better
  - Provided an initial accuracy of 75%
  - Improved to 81% with more feature engineering
  - Here we can see the classification report and confusion matrix:

	precision	recall	f1-score	support
0.0	0.81	0.80	0.80	5011
1.0	0.80	0.81	0.80	4988
accuracy			0.80	9999
macro avg	0.80	0.80	0.80	9999
weighted avg	0.80	0.80	0.80	9999

```
[[4027  984]
 [ 971 4017]]
```

SVM & XGBoost Models

SVM Classification Report:

	precision	recall	f1-score	support
0.0	0.80	0.81	0.81	5011
1.0	0.81	0.80	0.80	4988
accuracy			0.80	9999
macro avg	0.80	0.80	0.80	9999
weighted avg	0.80	0.80	0.80	9999

SVM Confusion Matrix:

```
[[4051  960]
 [1001 3987]]
```

XGBoost Classification Report:

	precision	recall	f1-score	support
0.0	0.80	0.79	0.79	5011
1.0	0.79	0.80	0.79	4988
accuracy			0.79	9999
macro avg	0.79	0.79	0.79	9999
weighted avg	0.79	0.79	0.79	9999

XGBoost Confusion Matrix:

```
[[3938 1073]
 [ 992 3996]]
```

Why Do This?

- I find it fun and interesting
- I’m not solving cancer diagnoses or other topics that may more directly impact many people’s lives
- Not everything has to have a serious and impactful reason behind it
- Still many uses that benefit eSports in general:
  - eSports betting
  - Team fairness in competitions (such as the one my Org, 9Moons, runs every week)
  - Talking points for Casters/Competitors

Next Steps

- Obtaining more data (another 50,000 games or so)
- Re-exploring NN's, as with more data, this may be a viable option
  - I like the idea of NN's because they may be able to take into account more patterns, such as one team having more defensive players vs the other team having more offensive players and how that would affect the model's predictive power)
- Even more in-depth feature engineering
- Real world implementation and productizing, including:
  - Using the model in 9Moons Events
  - Make some money on eSports betting websites >:)
  - Wrap the final model in a basic web/mobile app for ease of use and potential selling