PlayerUnknown's Battlegrounds

DAVID BUEHLER

Introduction

PlayerUnknown's Battlegrounds (PUBG) was a game released in March of 2017

One of the most popular games on Steam of all time with 3.2 million concurrent players at its peak

Kaggle had a competition to predict final placement from in-game stats and initial player ratings

Question posed by Kaggle: "*What's the best strategy to win in PUBG? Should you sit in one spot and hide your way into victory, or do you need to be the top shot?*"

Who's Interested?

Clientele of this project will be the PUBG Corporation, a subsidiary of Bluehole games

Mainly the developers of the game

This project will potentially help the developers create a better game

Will do this by figuring out if camping or looking for people is the best way to win any given game

- Camping is a playstyle that most of the community finds boring

Approach to the Problem

DATA ACQUISITION AND WRANGLING

DATA STORYTELLING

INFERENTIAL STATISTICS

Data Acquisition and Wrangling

All data pulled from the Kaggle competition website

Only pulled solos data from the entire data set

- The solos game mode gives that “true” battle royale experience of you vs. 99 other people with no one to help you

Pulled data that matched “solo”, “solo-fpp”, or “normal-solo-fpp” into their own DataFrame

Dropped down but not outs (DBNO’s) and revive columns

- Not a relevant statistics in solos game mode

Dropped matchId and groupId since those only gave redundant categorical information

Cheaters

PUBG had a rampant cheating problem early in its life

- I knew I would have to deal with those having firsthand experience of being killed by a few of them

Hto use my experience of what cheaters statistically looked like in order to find them in the dataset

Dopped data points with 50 or more kills in a single game

- Only 9 such entries

Also dropped anyone that had more than 20 kills at a greater than 80% headshot kill to kill ratio

- That is really good accuracy and almost unrealistic for anyone not using some form of help

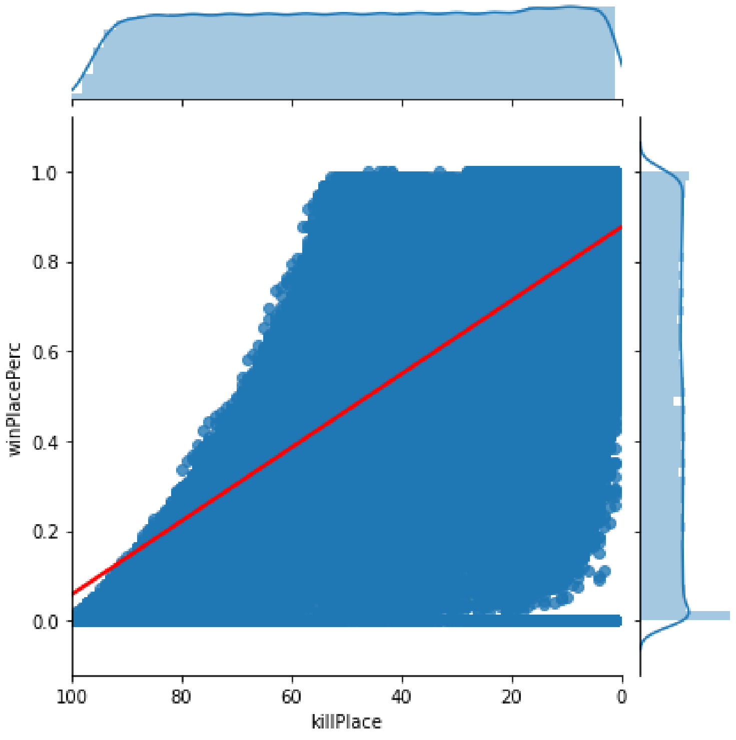
Data Storytelling

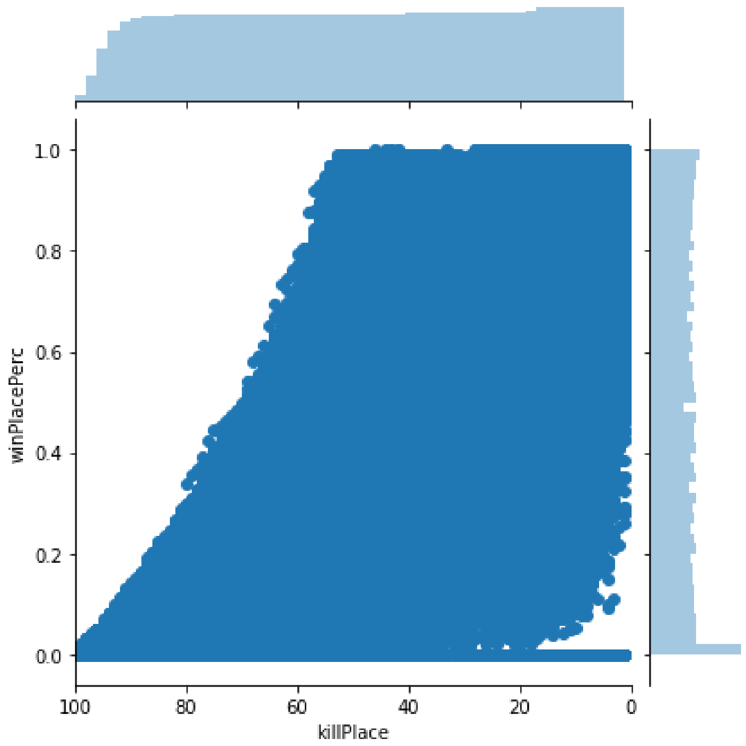
Explored how independent variables affected the target variable win place percentage

Kills, kill place, damage dealt, heals, weapons acquired

OLS regression from statsmodels to get R^2 and coefficient values

Variable	R^2	Coefficient
Kills	.233	.0916
Damage Dealt	.237	.0009
Kill Place	.576	-.0082
Heals	.161	.0498
Weapons Acquired	.369	.0703





Kill Place Graphs

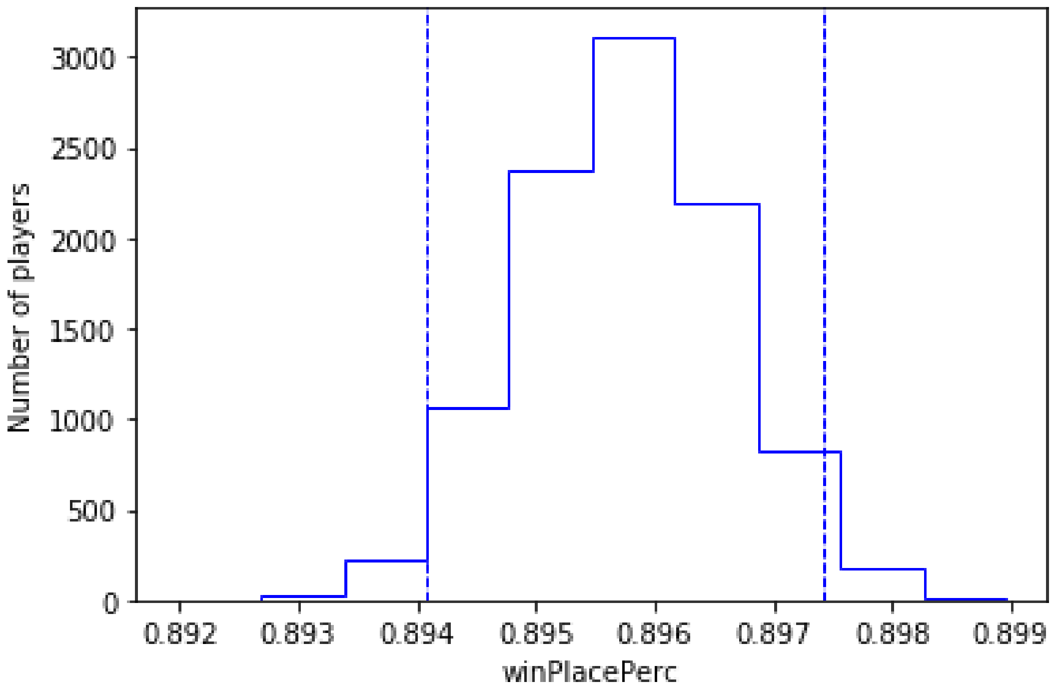
Inferential Statistics

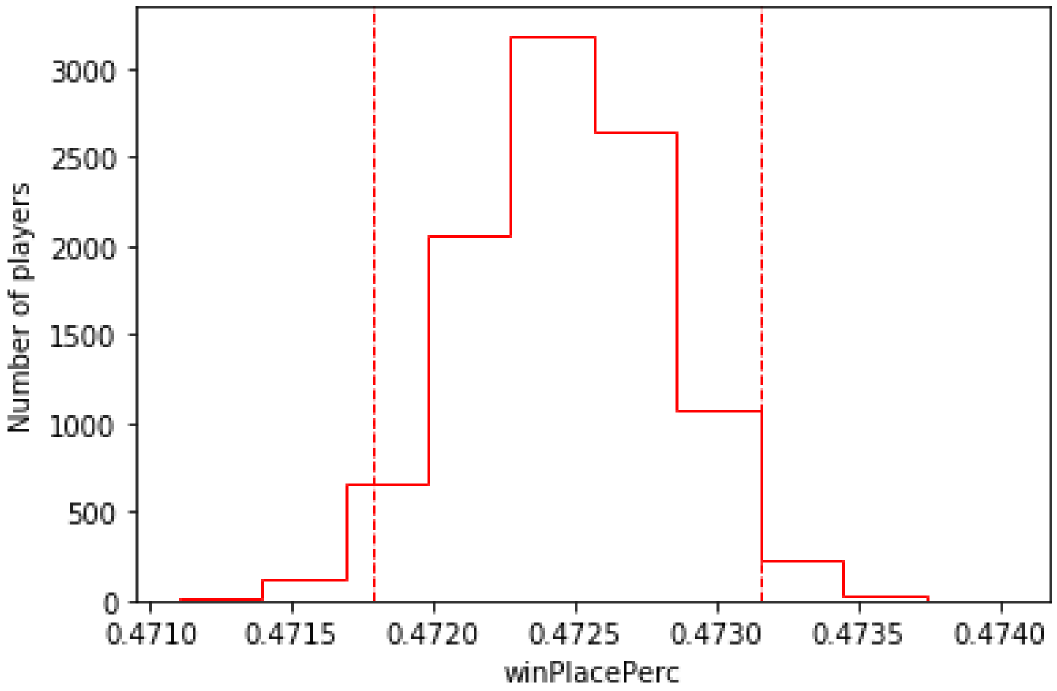
Null Hypothesis: On average, players with 5 or more kills win games as often as players with less than 5 kills.

Performed bootstrap test on the two groups

Found 95% confidence interval of the two groups

	Mean	95% Min Interval	95% Max Interval
>= 5 kills	.8958	.8941	.8974
< 5 kills	.4725	.4718	.4732





Bootstrap Test Graphs

Modeling

BASELINE MODELING

EXTENDED MODELING

Linear Regression

Started off with a base linear regression model

- No hyperparameters

No overfitting

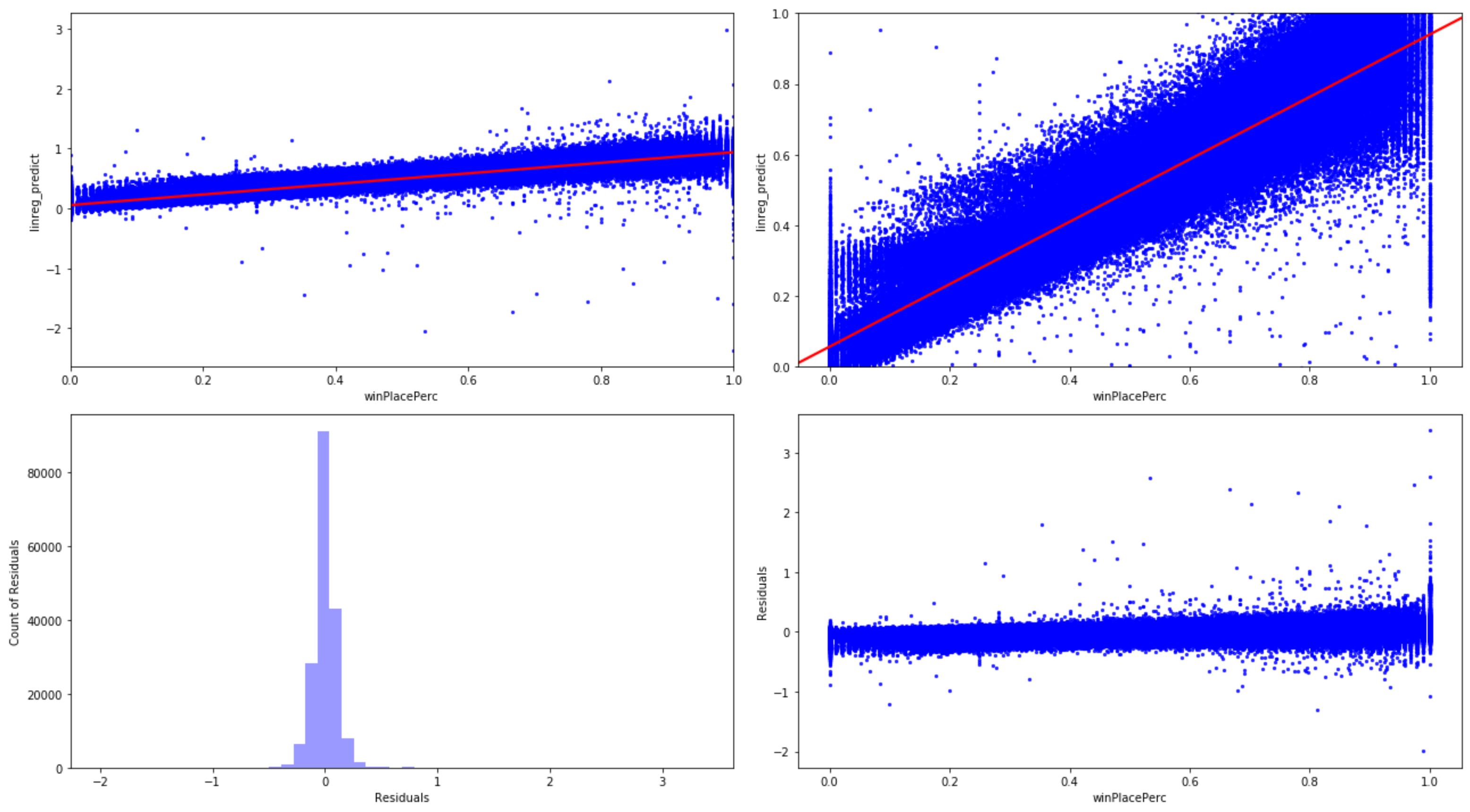
- Model performed similarly on new data as test data

Coefficient analysis

- Largest positive coefficient: Road Kills, .0341
- Largest negative coefficient: Kill Streaks, -.1895

Did not stay within bounds of target variable

- Percentage value, needs to stay between 0 and 1
- Max value: 2.99



Linear Regression Graph

Log Linear Regression

Performed to restrict bounds on the model

Dropped 0.0 winPlacePerc values out of dataset

- Took log of winPlacePerc then fit and predicted

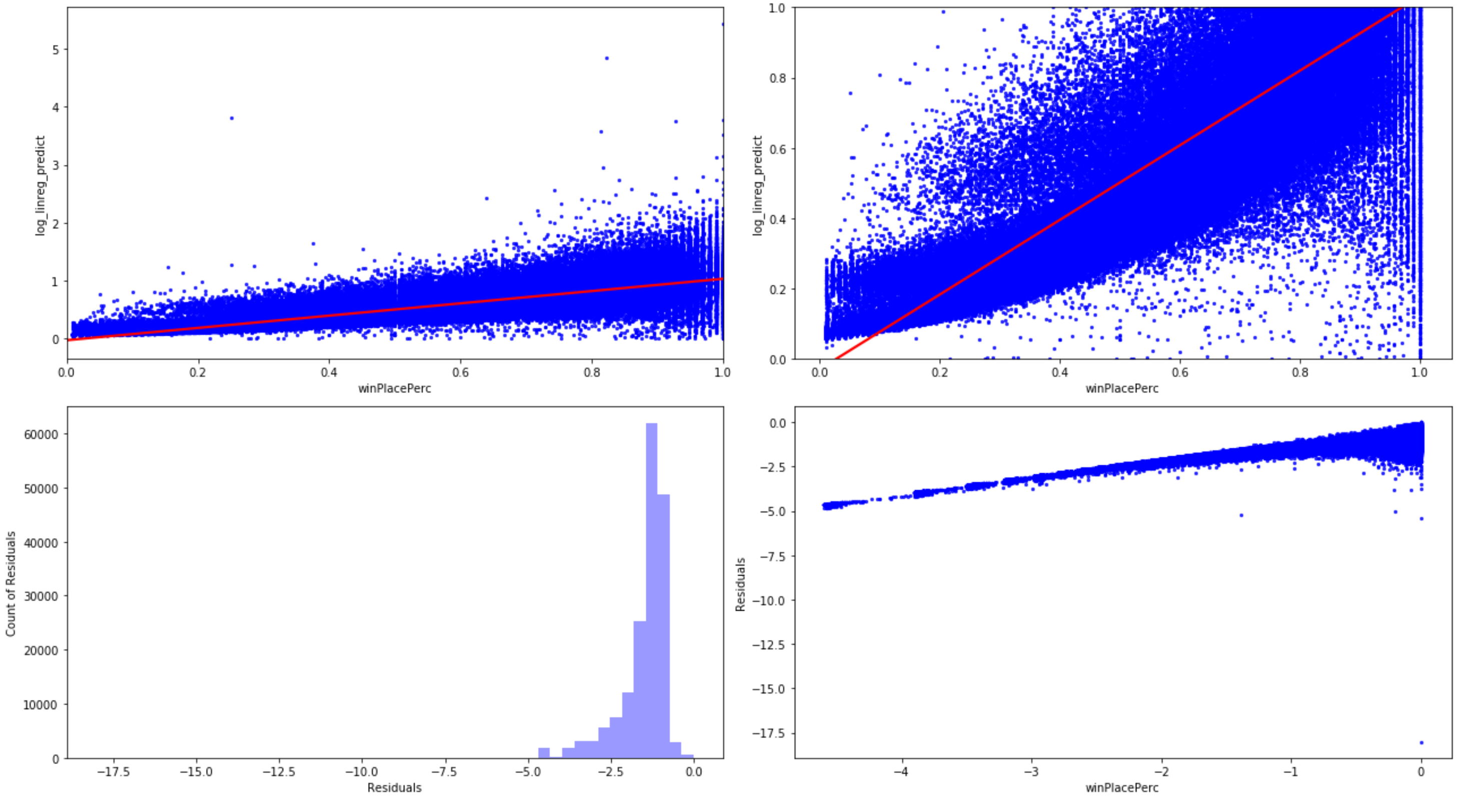
Exponentiated predicted values to get back to original

Performed worse than Linear regression

- .77 training/testing R^2
- Linear regression had .88 training/testing R^2

Did not restrict bounds

- Max of 18.01 on predictions



Log Linear Regression Graph

Lasso Regression

Not necessarily needed, linear regression did not show overfitting

- Performed as practice

Had to find alpha value

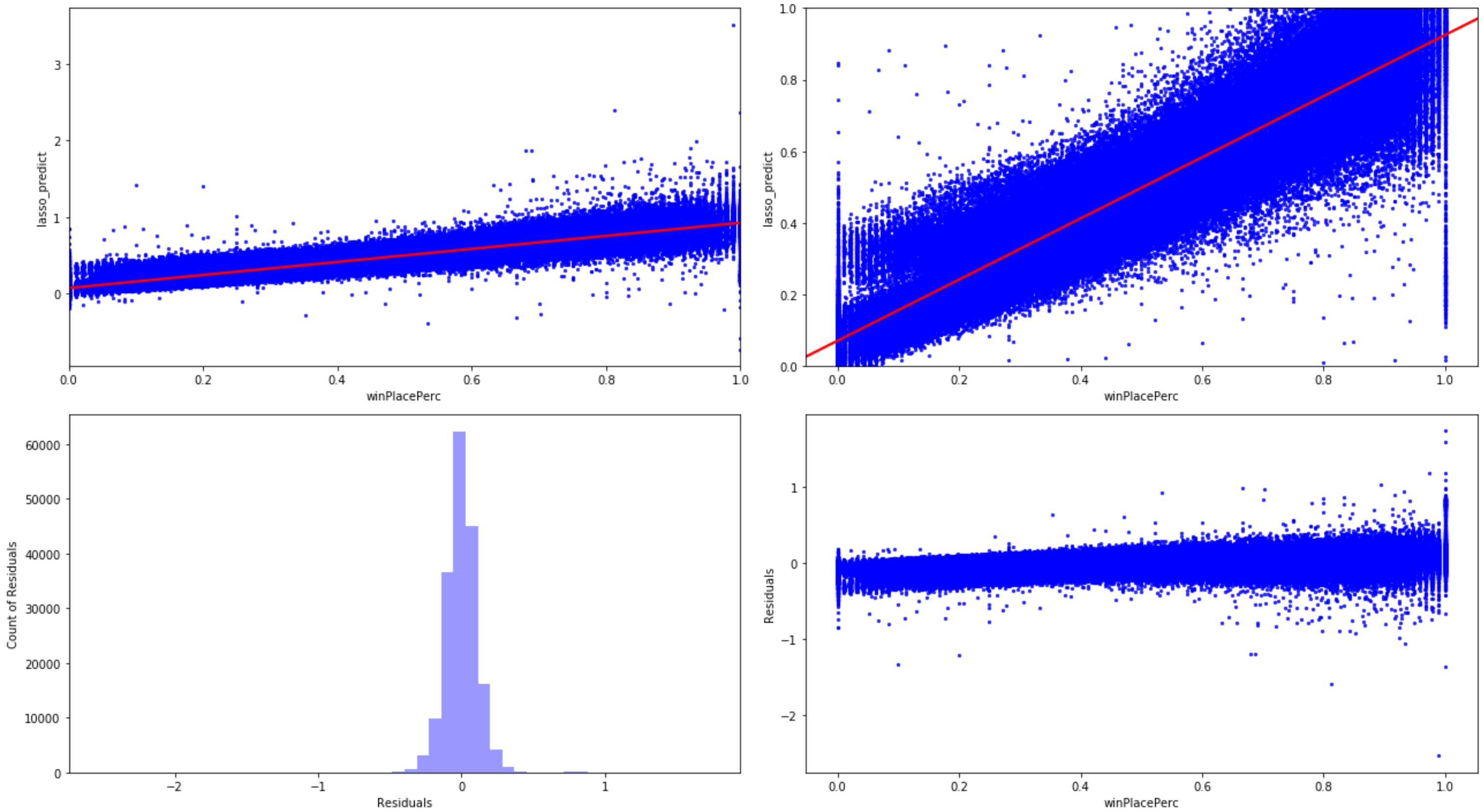
- GridSearchCV gave .01 as best alpha

Coefficient analysis

- Largest positive coefficient: Weapons Acquired, .0161
- Largest negative coefficient: Kill streaks, -.0659

Did not stay within bounds of target variable

- Max value of 3.52



Lasso Regression Graph

Ridge Regression

Not necessarily needed, linear regression did not show overfitting

- Performed as practice

Had to find alpha value

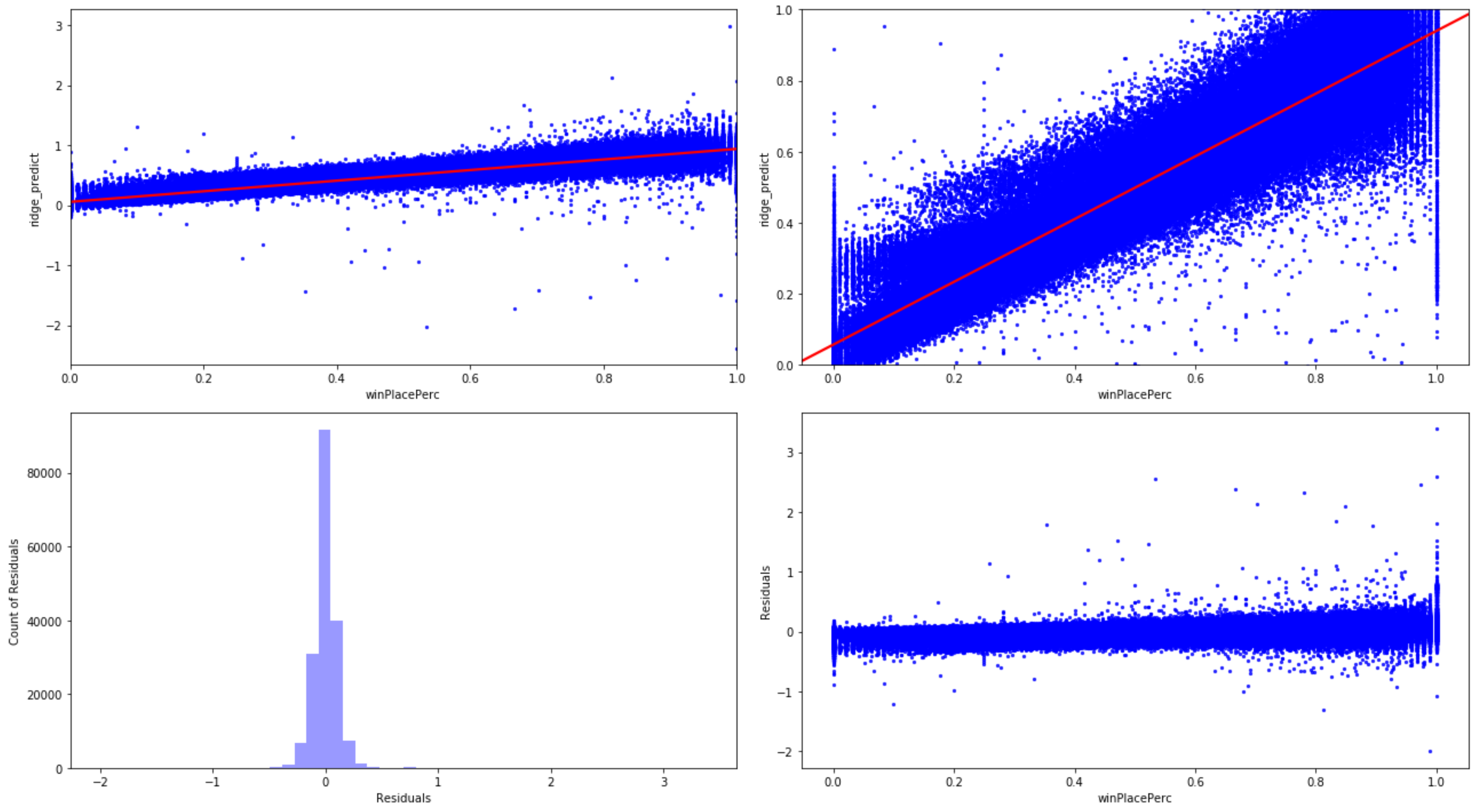
- GridSearchCV gave 190 as best alpha

Coefficient analysis

- Largest positive coefficient: Road Kills, .0323
- Largest negative coefficient: Kill streaks, -.1886

Did not stay within bounds of target variable

- Max value of 2.99



Ridge Regression Graph

Random Forest Regression

Attempt to improve accuracy on the model

Needed to find hyperparameters first

Used GridSearchCV to find best values for number of estimators and maximum depth

- Best n_estimators: 100
- Best max_depth: 20

Better than any linear model, performing at .95 R^2 on testing data

Only model to stay within the 0 to 1 bounds

Random Forest Regression (cont.)

Feature importance

- Walk distance was the most important variable at .774
- Kill Place in second place at .157

Key findings

- Walk distance was a significant factor for this model
- Kill place and kills were less important than originally predicted

Random Forest Regression (cont.)

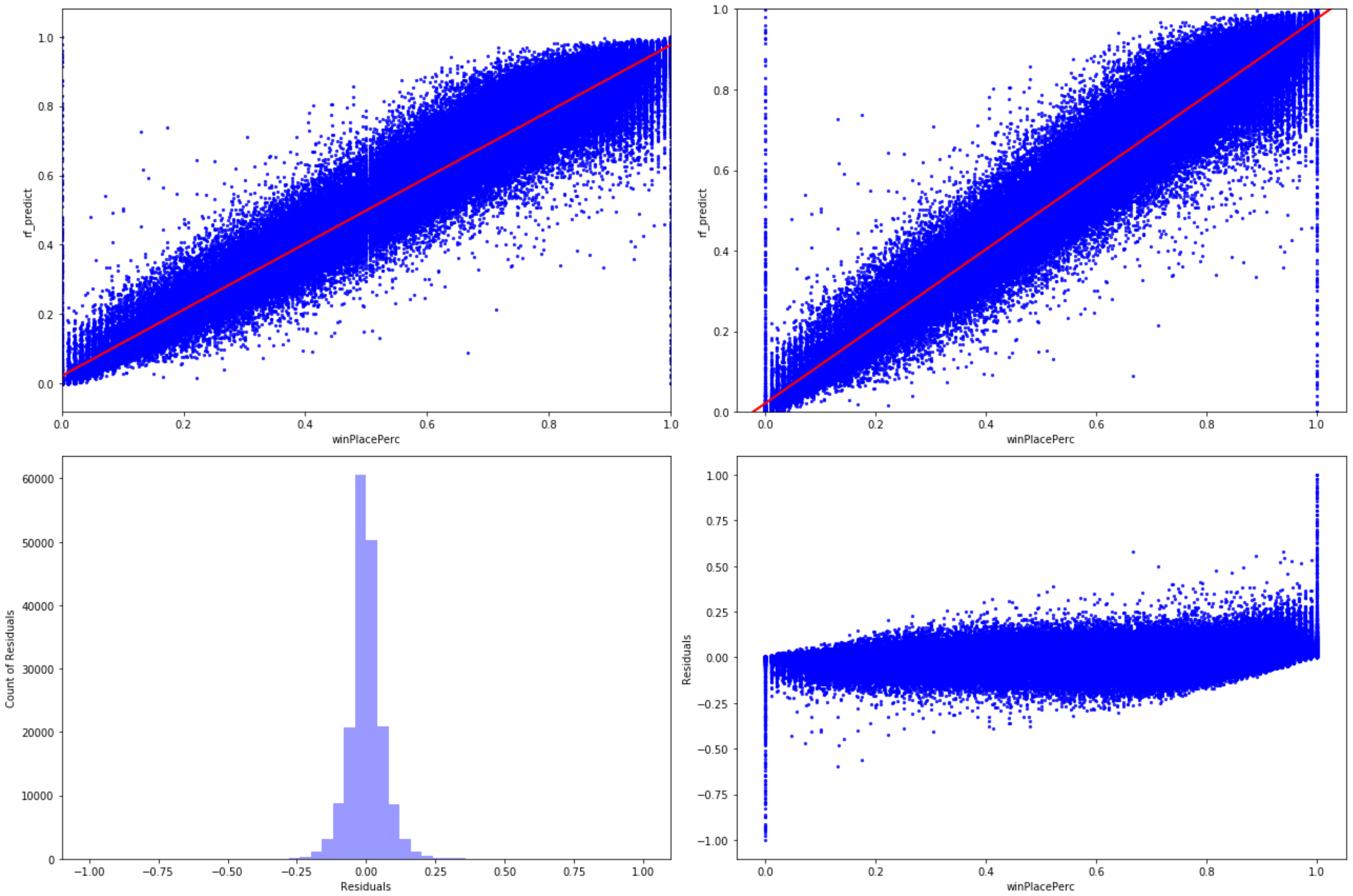
Examined maximum features parameter of the Random Forest regressor

Compared auto, sqrt, log2 max_features

- R^2 values:
  - Auto: .96
  - Sqrt: .95
  - Log2: .95
- Auto is the best max feature

Feature importance

- Looked at feature importance for the different max features anyway
- Walk distance is still the most important variable, no matter the feature



Random Forest Graph

Gradient Boosting

Last model used was XGBoost gradient boosting

Used 2000 as the number of estimators for this model

- To save time instead of using GridSearchCV to find a best n_estimators

Used “reg:squarederror” objective to make sure it was a regression model

R^2 value: .96

- Best model tested

Did not stay within 0 to 1 bounds

- Max of 1.18

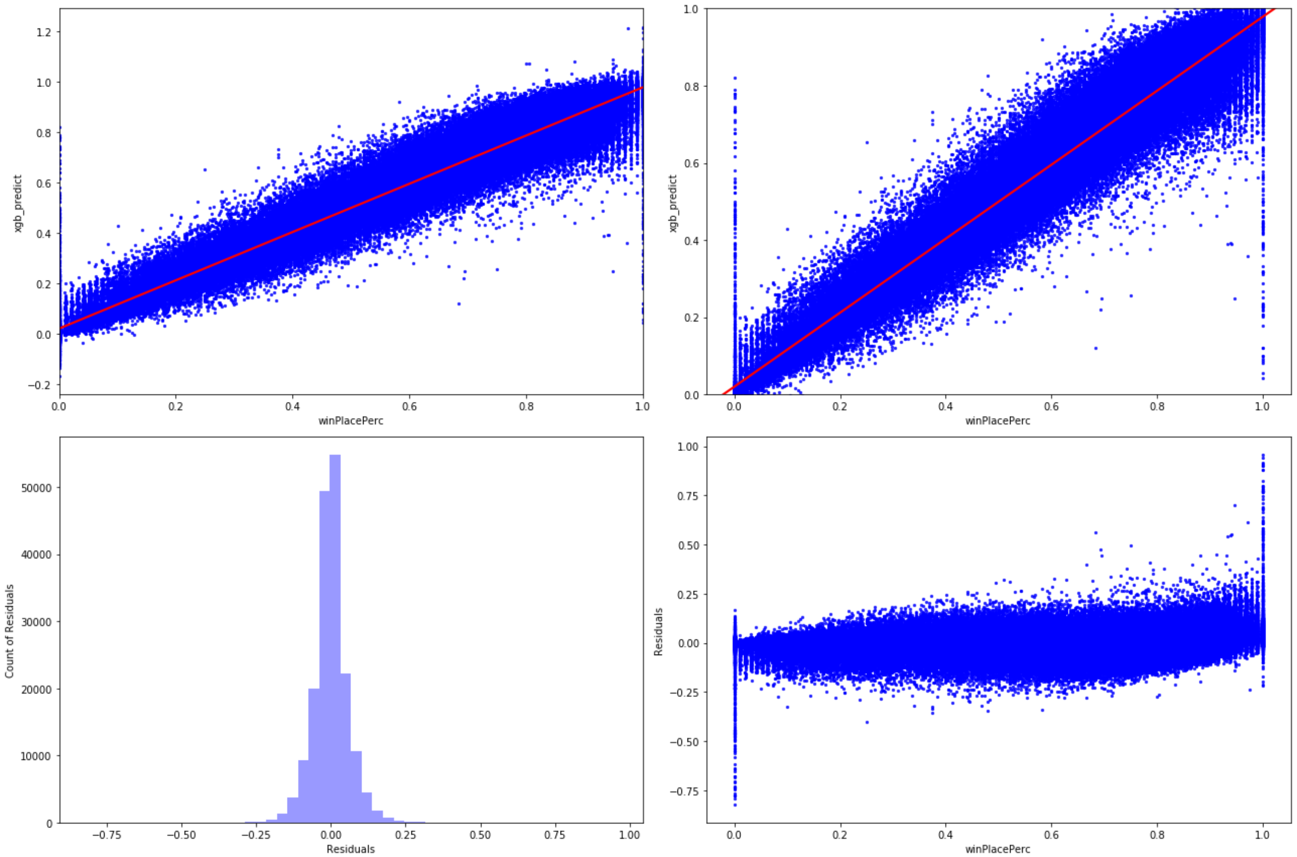
Gradient Boosting (cont.)

Feature importance

- Walk distance still most important at .731
- Boosts now in second place at .084
- Kill place drops to third at .079

Key findings

- Walk distance is king in making predictions in random forest models



XGBoost Graph

Results

Model	R^2 (test)	RMSE (test)	95% Min Resid	95% Max Resid
Linear	0.88	0.1033	-0.2056	0.2011
Lasso	0.86	0.1098	-0.2611	0.2727
Ridge	0.88	.1033	-0.2056	0.2011
Random Forest	0.95	0.6233	-0.1226	0.1267
XGBoost	0.96	0.6011	-0.1189	0.1245

Model	95% Min Residual	95% Max Residual	Residual Range
Linear	-0.2056	0.2011	0.4067
Lasso	-0.2611	0.2727	0.5338
Ridge	-0.2056	0.2011	0.4067
Random Forest	-0.1226	0.1267	0.2493
XGBoost	-0.1189	0.1245	0.2434

Conclusion

Best way to win in PUBG is to keep moving

- Linear models top coefficients
  - Road Kills and Weapons Acquired
- Random Forest most important features
  - Walk Distance, kill place, and boosts

These being the top variables would suggest that walking around and looking for people is the best way to win a given game

- Don’t camp and stay in one spot

Future Work

Explore why walk distance is important in the Random Forest model

- This could provide key insight into keeping players engaged in the game
- If they stay in the game longer, they are more likely to finish the game with a higher placement
- More distance walked equals more time spent in a game (for the most part)

Further investigation should be done on the tails on the ends of the actual vs. predicted scatter plots.

- At winPlacePerc values of 0 and 1, it appears that large tails where predictions are either far above 0 or below 1.
- It is likely that this is a result of how the models are built from bagging
- This would give additional insight to the impact when a good player has a bad day, or a bad player has an amazing game

Recommendations for the Client

Incentivise player movement

- Players have more of a chance to win the more they find each other
- Ways to incentivise movement:
  - Increase blue circle closing speed
  - Increase car spawn rates
  - Increase red zone frequency and lethality