Abstract

League of Legends is a multiplayer competitive online game, with two teams of five facing off with the goal of destroying the other team's nexus crystal. League of Legends is a highly popular game with a very passionate community, which enjoys not only playing the game, but also analyzing, watching pro-gamers, and betting on their favorite pro-teams. In fact, in the year 2019, viewership for the "League of Legends" World Championship finals beat numbers for the Superbowl flat, with the esports competition drawing in 100 million viewers across the globe (CNBC). Esports and betting on these competitive gaming teams are a huge business, and it is safe to say that many would benefit from understanding what leads to a team win. From classification modeling, we can hope to understand which tactics/experience lead a team to more wins, and then we can subsequently use this information to bet on teams which have historically had this experience or used such tactics in their gameplay. Pro-gamers can also benefit from this knowledge of gameplay to better their results.

Design

Classification models used for this analysis were kNN, logistic regression, and random forests. The first two were used for their simplicity in explanation to business clientele, especially since the logistic regression model's coefficients can lead pro-gamers and esports betters into inferences about which gameplay features are most influential in a game of League of Legends.

Data

These models were built using a dataset from Kaggle (https://www.kaggle.com/datasets/bobbyscience/league-of-legends-diamond-ranked-games-10-min?resource=download). The dataset has 9879 observations, with each row representing a different game. We see this data from the blue team's perspective, with wins represented by a 1, and losses represented by 0. Features include different gameplay features, like whether or not the blue team had the first kill of another team's champion, how many coins the blue team gathered, whether or not the blue team killed a dragon, etc.

Algorithms

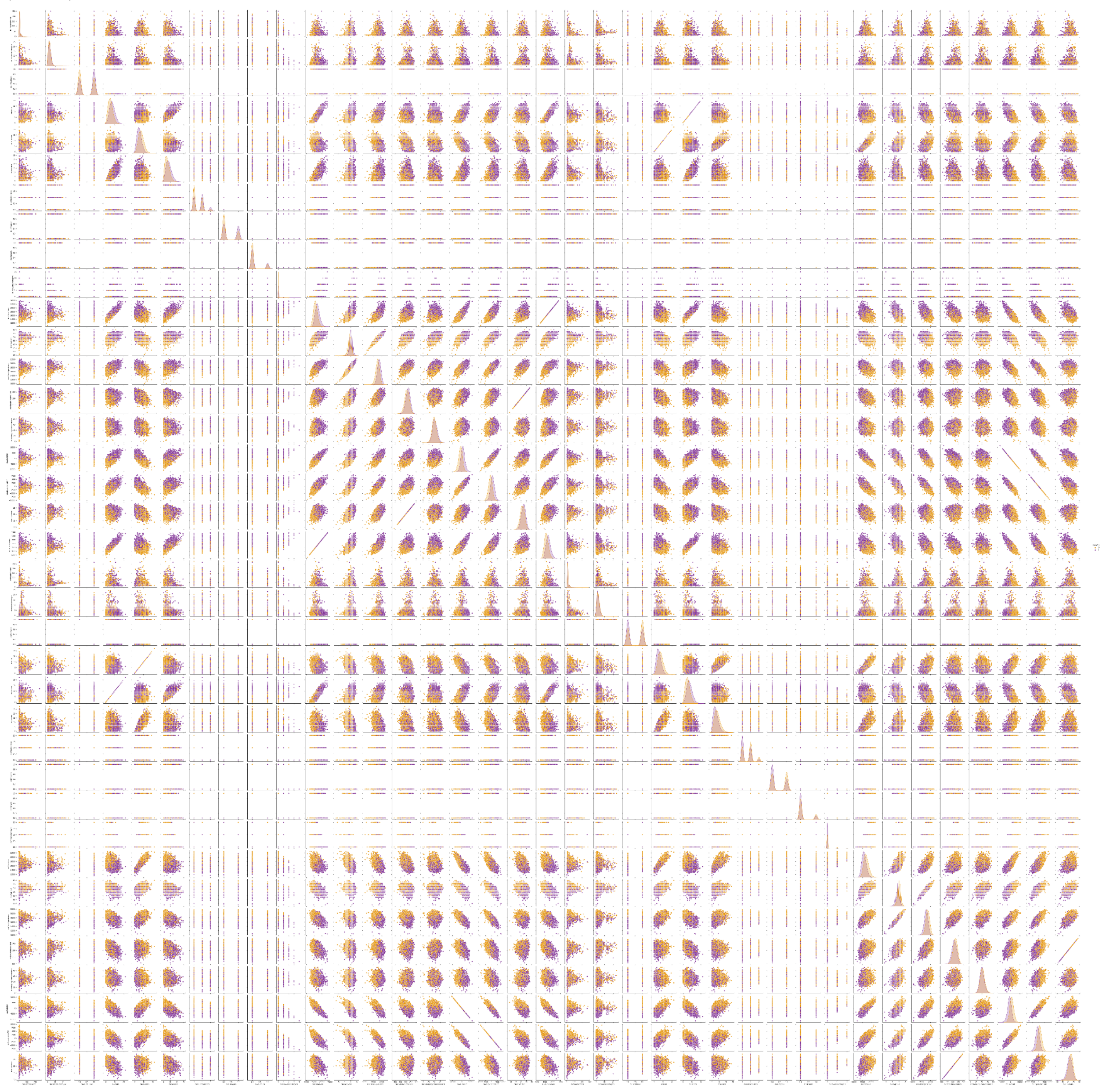
Data Cleaning/EDA:
Data was from Kaggle. No null values were found, there was some scaling of features for purposes of the kNN and logistic regression models

Model: kNN, logistic regression, random forests

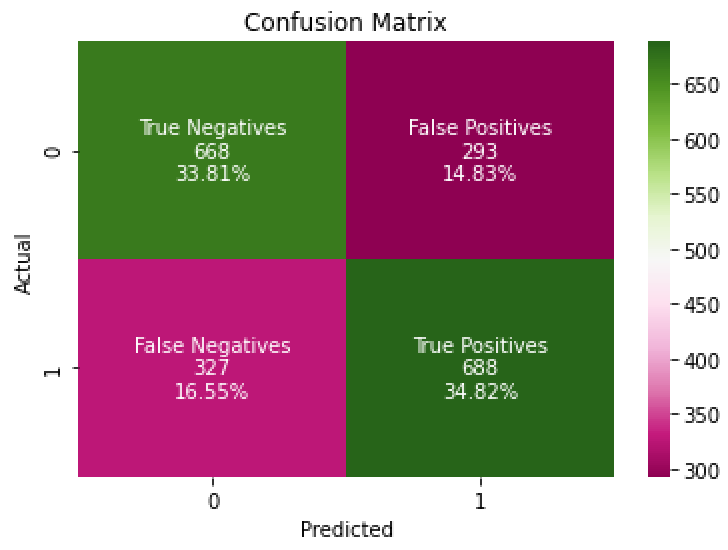
Model Evaluation: The dataset was split into a 80/20 training set and testing set. Used accuracy, confusion matrices, and ROC AUC for evaluation metrics.

Tools
Numpy, Pandas, Matplotlib, Seaborn, Sklearn

Communication
(EDA)



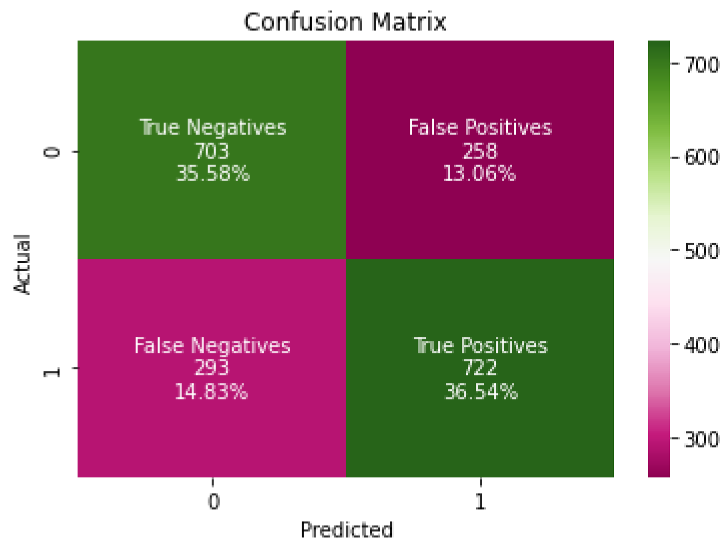
kNN confusion matrix



Model: kNN

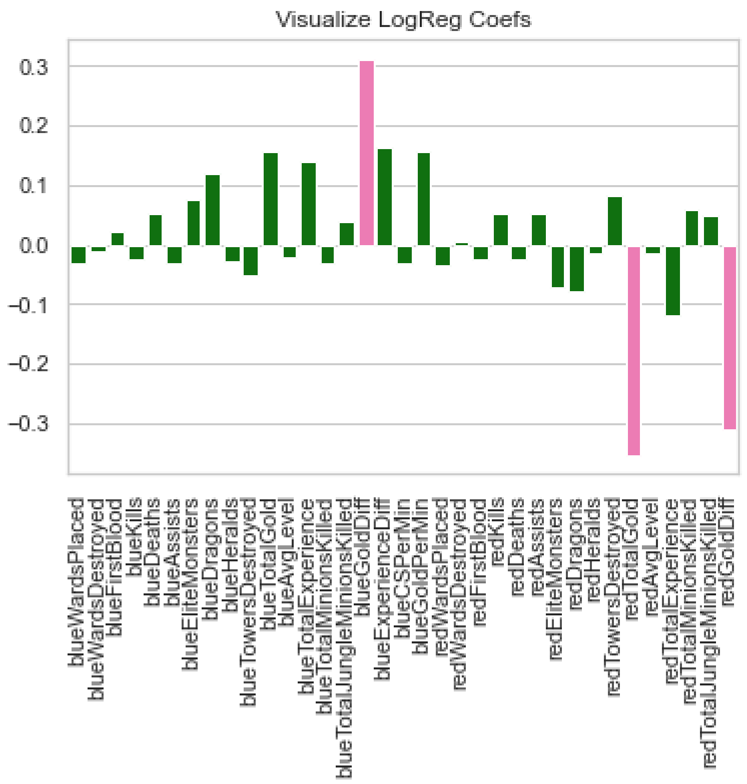
Test set accuracy: 0.6862348178137652
Train set accuracy: 0.7699607743894723
accuracy score: 0.6862348178137652

Logistic regression confusion matrix

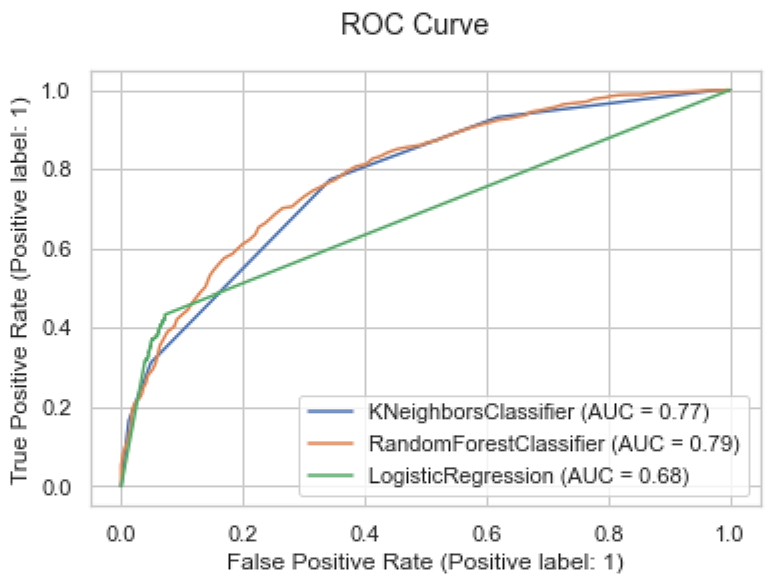


Model: LogReg

Test set accuracy: 0.7211538461538461
Train set accuracy: 0.7368088067822346
accuracy score: 0.7211538461538461
precision score: 0.736734693877551



Visualize LogReg Coefs
0.3
0.2
0.1
0.0
−0.1
−0.2
−0.3
blueWardsPlaced
blueWardsDestroyed
blueFirstBlood
blueKills
blueDeaths
blueAssists
blueEliteMonsters
blueDragons
blueHeralds
blueTowersDestroyed
blueTotalGold
blueAvgLevel
blueTotalExperience
blueTotalMinionsKilled
blueTotalJungleMinionsKilled
blueGoldDiff
blueExperienceDiff
blueCSPerMin
blueGoldPerMin
redWardsPlaced
redWardsDestroyed
redFirstBlood
redKills
redDeaths
redAssists
redEliteMonsters
redDragons
redHeralds
redTowersDestroyed
redTotalGold
redAvgLevel
redTotalExperience
redTotalMinionsKilled
redTotalJungleMinionsKilled
redGoldDiff



ROC Curve
True Positive Rate (Positive label: 1)
False Positive Rate (Positive label: 1)
0.0
0.2
0.4
0.6
0.8
1.0
KNeighborsClassifier (AUC = 0.77)
RandomForestClassifier (AUC = 0.79)
LogisticRegression (AUC = 0.68)