Prediction League of Legends Promotion Using Classification Algorithms

Jeff Williams
Washburn University
1700 SW College Ave
Topeka, KS 66621

Abstract

Many E-sports teams use data analysis to develop in game and out of game strategies. Many would argue that the data available is not consistent or large enough to apply machine learning algorithms to help making these decisions. I believe that with proper data preprocessing, I can develop a model to accurately predict player promotion from minor region to major region League of Legends teams. Most resources I have seen claim an F1 score of .5 as the acceptable minimum for a model, so if I can reach that number with any of the models, I would consider that a success.

Data Collection

I began collecting data by exploring the various data set that are publicly available on OraclesElixir.com. Oracles Elixir is a free website that compiles and organises match data from all League of Legends professional and amateur leagues and tournaments. Exploratory data analysis was done using the raw game data provided in the "Downloads" tab on the site. I realized that the data from this data set contained far too many features, and would require a significant amount of manipulation to even consider it usable. Moving over to the players tab, I found data that was much more manageable, and provided more in-depth data about each player.

Preparing the Data Set

I started by collecting data set for the Summer 2022 minor region players and the Spring 2023 major region players. From there, I combined the minor regions into on data set, and the major regions into another. To determine whether the minor region players were promoted to a major region, I found the intersection of the two data sets, and created a new column, "Promoted", in the minor region table and added values, either "1" for promoted, or "0", for not promoted.

Before I started training the models, I noticed two major issues with the data set.

- The data set was about 85 percent skewed towards the "not promoted" class.
- The data contained too many features for most models: around 25 features in total.

Copyright © 2023, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

Methods

When conducting research, I found a paper titled "Combination of PCA with SMOTE Oversampling for Classification of High-Dimensional Imbalanced Data" [1]. The data used in that paper had the same two issues as the data I was conducting my research on. Based on the success of their models, I decided to follow the same methods they did when preparing my data and training my models. Doing this would allow me to compare the results of my tests to the results stated in the paper.

The three algorithms will all be tested without preprocessing, and then with SMOTE, SMOTE plus PCA together. In my testing I found that with the size of my data set, PCA has little effect on the predictions without using SMOTE first. The models will be evaluated at each stage using accuracy, F-Measure, and ROC area.

Model Selection

The paper previously mentioned used three different classification algorithms, K-Nearest Neighbor (KNN), Logistic Regression (LR), and Decision Trees (DT). These models have the advantage of being relatively easy to tune their hyperparameters, as well as generally having a high accuracy rate [1].

Techniques for Re-balancing Classes

One issues that can arise when training a classification model occurs when the classes are unbalanced. This means that one or more of the classes are under-represented in the training data. This can cause issues when training because the model can result in good accuracy with the test data just by predicting all samples as belonging to the majority class. There are two ways to solve this issue, under-sampling and over-sampling. Under-sampling re-balances the classes by selecting a certain proportion of samples from the majority class, and dropping the rest. This technique works best with large data set [2]. For smaller data sets, over-sampling is preferred. Over-sampling works by copying the data points from the minority class and creating new samples from that data.

The technique used in the paper, as well as the technique that I will be using is a type of over-sampling called "Synthetic Minority Over-sampling Technique" (SMOTE) [2].

SMOTE works by not just copying the data from the existing samples, but it applies specific operations, like rotating and skewing the data [2]. The synthetic samples are then joined by its k-nearest neighbors, depending on the amount of over-sampling required [2].

Dimensionality Reduction

In addition to rebalancing, I also wanted to test dimensionality reduction techniques to reduce the number of features being tested. Both the paper and I are using Principal Component Analysis (PCA) to reduce the number of features in the data set. PCA has many uses in addition to dimensionality reduction, including exploratory data analysis [3]. PCA works by finding relationships between features, combining them, and removing features that have little to no impact on the model predictions [3]. This leads to an overall reduction in features for the model to be trained on.

Using PCA is relatively simple, the data must first be scaled, and then when PCA is applied, it returns an order list of variances for the data set. Then you can run PCA again on the scaled data set, and select the number of features (components) for the output data.

Testing Without Pre-Processing

The results from the models using raw data were unsurprising. KNN appears to perform well, having high accuracy and ROC scores. However, KNN managed to make 0 correct guesses, as shown by having an F1 score of 0. The LR and DT models had similar results. Below are the metrics for each.

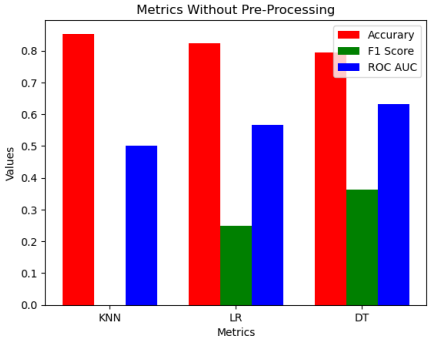

Figure 1: Test Results with Raw Data

Testing After Applying Smote

After applying SMOTE to the training set, I retrained all three models on the new data set. Results from this set of tests were more promising. LR was just shy of hitting a .5 F1 score, which is about where I wanted to be at minimum. Accuracy dropped a bit for all three models, though were still within acceptable ranges. Below are the results from testing after applying SMOTE.

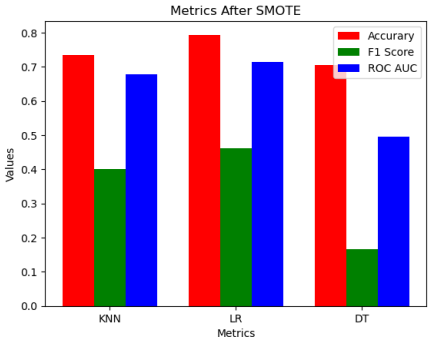

Figure 2: Test Results After Applying SMOTE

Testing After Applying PCA

After applying PCA to the training set, and then applying SMOTE, I again retrained the models on the new data. The results were somewhat expected. Most of the models took a slight hit to their accuracy score. The most notable improvement came from LR, which was able to get an F1 score of .5, which is what the minimum I was looking for in my hypothesis.

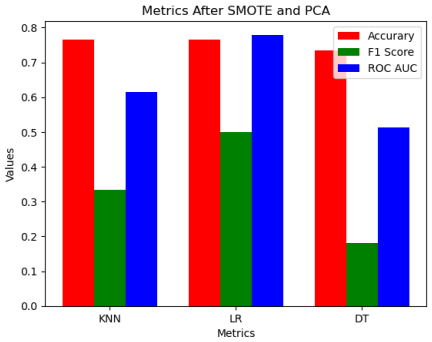

Figure 3: Test Results After Applying SMOTE and PCA

Conclusion

Though most of the test metrics would not be considered acceptable is most classification allocations, with the data that was available and the biases within the data itself, I was able to build a model that can make correct guesses regarding player promotion between professional League of Legends teams. Using SMOTE, PCA, and LR, I was able to meet the minimum acceptable F1 score of 0.5 that I defined in my hypothesis.

References

Link to notebook with data and results
https://github.com/Jefferw/PromotionClassifier/

MULLA, Guhdar Demir, Yıldırım Hassan, Masoud. (2021). Combination of PCA with SMOTE Oversampling for Classification of High-Dimensional Imbal-

anced Data. Bitlis Eren Üniversitesi Fen Bilimleri Dergisi. 10.17798/bitlisfen.939733.

Chawla, N. V., Bowyer, K. W., Hall, L. O., Kegelmeyer, W. P. (2011). SMOTE: Synthetic Minority Over-sampling Technique. arXiv. https://doi.org/10.48550/ARXIV.1106.1813

Sevenhuysen, T. (n.d.). Oracle's Exiler. Oracle's Elixer. oracleselixer.com