Classification Project Write-Up

Abstract

The goal of this project is to predict whether an NBA rookie player would play beyond their rookie year contract (4 years) based on their first-year stats. After iterating through different models, Random Forest gave the best performance in terms of F1 score. Thus, I will be utilizing a Random Forest with optimized hyperparameters to predict this classification problem.

Design

Using a tuned Random Forest model, we'll be able to predict the longevity of an NBA rookie player's career. More specifically, the model will predict whether the rookie player will play beyond their rookie contract, i.e. more than 4 years, based on their first-year stats. This can help NBA teams scout out rookies with great longevity potential and draw up contracts and negotiations appropriately.

Data

Data was webscraped off the NBA official website using Selenium, going as far back as 1996-97 season to present. There were over 12,000 data points with 20+ features. After compressing down to allow only first-year stats, we got just shy of 2,500 data points to use in our model.

Algorithms

For baseline models, I iterated through kNN, logistic regression with 1-, 3-, and all-features, Gaussian Naïve Bayes, a single Decision Tree, Random Forest, and XGBoost. I used cross-validation and stratification to evaluate the models due to the small size of my dataset (and imbalance of class distribution). I decided to score all models on the F1 score to find a balance between precision and recall.

To address class distribution imbalance, I used random oversampling on the minority class (target class) to balance out with the majority class.

To avoid violating normality assumption in Gaussian, I did a log transformation of the top 5 features, plucked from the list of feature importances. Note that Gaussian wasn't the final model chosen, so log transformation of features became unnecessary.

In the end, Random Forest was the top performing model. To tune the hyperparameters for this model, I used a wide range of parameters and narrowed it down using RandomizedSearchCV. With the narrower range of numbers, I used GridSearchCV to nail down the optimal set of hyperparameters for Random Forest.

Ultimately, I decided to use feature selection to only include the most contributory features. I was able to trim down my list of 20+ features to 11 at the end of feature reduction.

Tools

- **Data collection:** Selenium, BeautifulSoup, Python
- **Data storage:** Pickle library
- **Data cleaning/EDA:** Pandas, NumPy
- **Data Visualizations:** Matplotlib, Seaborn, Canva
- **Classification Models:** kNN, Logistic Regression, Gaussian Naïve Bayes, Decision Tree, Random Forest, XGBoost