Classification Project Write-Up

By: Colin Gallagher

Abstract:

Classification models were created to predict if an MLB team won their game or not based off their team total performance statistics in the game played. Data was scraped from the baseball-reference.com website and each game's boxscore webpage was grabbed to obtain each teams total performance statistics in a game. This data was organized by year and then each year was saved as a separate table in a SQL database. Using Python, the data was pulled in, analyzed, and various classification models were designed to predict what kind of performance in the game of baseball leads to a win. Clients that would benefit from using these classification models would be players, managers, owners, fans, gamblers, and sports analysts. The models reflect which statistics are most important when trying to win games.

Design:

The purpose of this project is create a classification model that solves a problem. The problem that this classification model solves is a way for either players, managers, or team owners to see what parts of the game are most important when it comes to getting the win which is the ultimate goal in professional sports. Fans, gamblers, and sports analysts would also enjoy using the model to gain more knowledge of the sport as well. The models created highlight the team total statistics that give the best chance at coming up with a win.

Data:

The raw data that was used for this project was scraped from the baseball-reference.com website. A total of 5,785 games' boxscore webpages were scraped. Each page was a game and had four main tables of data: batting and pitching statistics for Team 1, and the batting and pitching statistics for Team 2. Each teams' total row of each table was combined to form one row per team, so each page resulted in two rows of data totaling up to 11,570 rows of data. Each year or season was separated into its own SQL table in the SQL database. Each row or table contained 58 columns or features. Many of these features ended up being useless, but every column was scraped from the webpages just in case.

Algorithms:

To scrape the data, first a list of links was acquired and built for the games of interest. This list was simple to create because of how the website organized its links' formats. For this project, this list was just composed of three links for the three seasons of interest: 2020-2022. Using those links, the links for every game could be scraped in that season. A list for the actual scraping of the data was then built. The final list ended up containing 5,785 links, one for each game, divided into the three separate years. These links were scraped and the rows were continuously concatenated into a data frame. Once the last game of the year was scraped, the data frame was saved as a table to the SQL database with the name being the season/year. After the creation of the database, the data was loaded in with Python and classification models were started to be trained. Four different types of classification models were trained: KNN, Logistic Regression, Decision Tree and Random Forest. Starting with KNN as a baseline, features were selected that had decent separateability and then continuously refined to make more sense of certain feature combinations. After trying different combinations three candidate models were finalized: batting only, pitching only, and combination of the two. These three feature combinations were then used as the starting point for logistic regression model creation. During the training of logistic regression models, it was apparent that a huge feature for win classification was Run Expectancy Based on 24 Base Outs. Creating models with just this one features on its own performed well, but although that statistic is a great measure of team performance, the answer to the initial question would simply be: score runs when you have the opportunity to. Now the question is, how so? A new set of feature combination was found to answer this question and it was: On Base Plus Slugging, Runs Batted In, Batting Average with Runners in Scoring Position, Earned Run Average, Walks, and Errors. After validating this model it performed with an accuracy of 96.7%, precision of 96.2%, and recall of 97.1%, but accuracy was the main focus for this problem. The coefficients for the features were: 1.25, 5.31, 0.62, -6.98, -0.35, -1.00. This shows that when on offense, if the team has a higher than average OPS, RBI, and BA w/RISP, the probability of classification of a win increases with RBI being the most important. It also shows if a team gives up less earned runs (most important), walks, and errors than average, they benefit more which makes sense. This same feature set was used to train a classification model using Random Forest, but this was more for a verification. Logistic Regression is more interpretable, so logistic regression ended up being the final model chosen. With the Random Forest method feature importance values of 0.15, 0.26, 0.07, 0.46, 0.03, 0.02 which lines up with the coefficients of the logistic regression model.

Tools:

Tools used for building the data storage and processing pipeline:

- **Python**: Coding language used to perform actions on the data with its various libraries.
- **Pandas**: Python library used to clean and place data into organized data frames.
- **BeautifulSoup**: Python library used to scrape and parse through HTML to get the data.
- **Selenium**: Python library used to also scrape the webpages, specifically in this case it had to be used on the boxscore pages.
- **SQL Alchemy**: Python library used to create and load in the database.
- **Sklearn**: Python library used to train and test the classification models.
- **Seaborn**: Python library used to create a pair plot for initial feature selection.
- **Numpy**: Python library used to create and manipulate arrays to make predictions and score models.

Communication:

A five minute slide presentation will be given to explain the classification, how it was created, the results, conclusions, any future work that can be done to improve the project, and answer any questions.