Robert Ballard and Ameya Bahirat
CSCI 4831 - Sabermetrics

How-To:
Clone Repo
Install following dependencies: tKinter, CSV, Matplotlib, Numpy for python 3
Run command python3 frontend.py

SWP: Simulated Win Percentage

Simulated Win Percentage (SWP) is a stat generated through a simulation of numerous baseball games and comparing their results. In short, a large number of baseball games are simulated off of the starting lineups, pitchers, and stadium provided. The end result is a percentage of how often the home team won. It is worth noting that the SWP for the away team would be one minus the home team's SWP.

The simulation takes into account basic stats provided by the Lahman database, and all common events that could occur during a game. That is to say, each game has 9 innings (or more in the case of ties), each team has 3 outs during each of those innings, and whenever a batter is thrown a ball, a random variable decides how it plays out -- whether they gain a strike, a ball, hit into a single, etc. This random variable is decided by using the probabilities obtained from the batter's stats, e.g. how many singles, doubles, balls, did they get in their at-bats -- after having a modifier applied to take into account the pitcher's stats (a batter is likely to better against a weak pitcher than a strong one). This modifier serves as a second, intermediate stat that this application needs to produce, and is described in greater detail in the section below. Irregardless, each play is simulated, and the game is played to completion. The results are stored, and another game is simulated with the same teams and situation. This continues until a certain number of games have been simulated, and then the final calculation is produced. This simulation, due to limitations in the data being used as well as time constraints, does not account for injuries, variations in fielding (e.g. how does the game change if there are better or worse fielders), situational maneuvers (e.g. sacrifice fly with someone on third base), and player replacement beyond relief pitchers (e.g. Nolan Arenado charging the pitcher).

SWP is a useful stat because it seeks to answer a question most stats fail to: very simply, who is favored to win. Most other statistics either focus in on certain types of plays (e.g. batting average (BA), which looks for the probability of a player batting successfully or not), or on the specific contribution of a player (e.g. wins above replacement (WAR), which seeks the expected value of a player in theoretical runs). SWP provides a simple answer to the question “who will win?” whilst providing an inherent degree of confidence through the probability it outputs (for instance, a SWP of .55 indicates that the home team is slightly favored to win the matchup, while a SWP of .05 indicates that the away team is extremely favored to win). The closest stat to SWP currently in practice would be win expectancy (WE). They both exist to provide an answer to the same question, but WE is different in that it relies directly on past scenarios to decide (e.g. how often has the home team won when the score is 2-4 and the bases are loaded), instead of probability and simulation. Furthermore, the purposes of the two statistics are different: WE is usually used as a “story” statistic, which means that it serves to show what events in a game impacted it the most (in this case, what plays caused WE to change the most). SWP is intended to be more predictive, as more of a guess as to how a game will work out than a retrospective explanation of what plays mattered.

In order to generate SWP, we will be taking into consideration only the statistics for the batter and the pitcher. This is done primarily to simplify the complexity involved in predicting the outcome of a baseball game. A much more comprehensive (and perhaps more accurate) might take into account all players, weather conditions and other information, but due to the time constraints of this project we will be ignoring these. The way the simulation will take place involves finding the probability of a hit or no hit based on the stats of the batter and the stats of the pitcher, and then finding the probability of the event that precedes it. To calculate whether a hit occurred or not, a “true” batting average is calculated taking into account the statistics of the pitcher. The batting average (BA) will be considered as the probability of a hit, while 1-BA will be considered as the probability of a non-hit. The pitcher statistic we use will be multiplied by the batting average, giving us the true batting average of the player. This means that pitchers with better stats will make the batting average (probability of a hit) of the batter lower, while lower than average pitchers will make the batting average higher. To calculate what event precedes a hit, the batters statistics over all games is taken into account. A percentage is generated for each event, and one is randomly chosen using the percentages as weights. There is some difficulty in finding pitcher statistics that could provide a fair measure into calculating whether a hit was made or not, also since information about ball speed and type of throw are not given in the Lahman database, there is some limitation with regards to finding what makes a pitcher good.

Final Update

The statistical measure that was used to evaluate the true batting average of players was Difference from Average Strikeout Probability (DASP). This is a calculated by taking a pitchers strikeouts divided by the batters faced by pitcher. We believe this to be a good measure of a pitchers skill, and provide us a number that can reasonably impact a batter’s batting average. A constant multiplier of 0.5 was added to this calculation to provide us with a reasonable number of runs in each game while running the simulation.

When running the program. The user will have the option to simulation one game (with a team of pre-defined players) 100 times, and will be provided with a win expectancy of the home team. This is done by hitting the “calculate” button. The “calculate bulk WP” button will generate a random set of batters and pitchers for each team selected and simulate 100 games, each with a random sets of pitchers and batters. It will provide the user with the simulated win percentage, alongside the actual win percentage extracted from the Lahman database. Both buttons provide the user with graphs of the run expectancy for the home team.

Generally we found our statistic to be accurate, any discrepancies to real world data might be caused by retrosheet providing us with win-loss data for multiple years. Another reason for some lack of correlation between the data might be caused from our statistic not taking into account ground outs.