# Sports Statistics Index

Website

Will Michaels

December 2021

## Overview

I created Sports Statistics Index in high school when I wanted to learn how to code. I knew coding was a powerful tool, but I didn't know how to get started. I knew I learned most easily through experience, so I decided to create a project that would teach me how to code. The project, Sports Statistics Index, is a website that hosts my predictions about sporting events generated using statistical modeling. I am no longer actively making predictions, but this project allowed me to improve my skills in programming and statistics.

The entire prediction model was created using Python. I used NumPy and SciPy extensively, as well as SQLite3 and Pandas. Data was pulled from websites like Basketball Reference and NBA.com.

## Prediction

### Multiple Regression

The model comes up with a probability of victory by considering many pieces of data about a team going into a particular game. By using historical data, we can identify the chance a team will win a game based on the value of one of their stats. This is done by creating a plot like this:

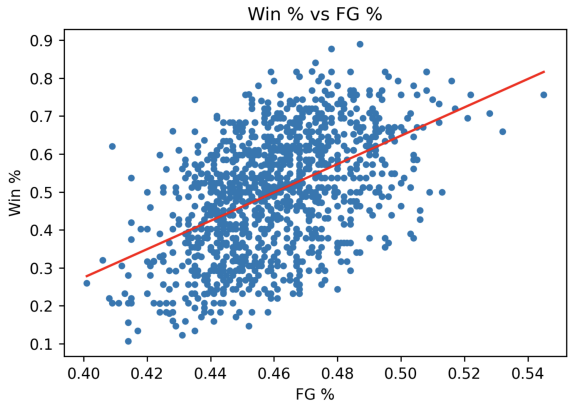


Figure 1: Win probability vs. FG percentage

This plot shows the historical probability of a team winning a game based on their average scoring, as well as a linear fit to the data. We can plug the team's value for a stat into a polynomial model similar to this one and predict the chance of them winning the game. These relationships can be linear or nonlinear. Both scenarios can

be handled by `np.polyfit()`. Once this is done for many stats, the probabilities can be combined in a weighted average to form the first part of the prediction.

### Elo Ratings

The data used to make a prediction is not limited to in game statistics. Factors such as amount of rest, travel distance, home court advantage, and injuries are considered when formulating a final prediction. These things can be factored into the model via Elo ratings. An Elo rating is a numerical measure of a team's strength relative to its competitors. It is increased or decreased after each game based on whether a team wins, loses, or draws. A team's chance of winning a game is determined by its Elo rating and the Elo rating of its opponent. In particular the probability of team A beating team B is,

$$P(A) = \frac{1}{1 + 10^{(R_B - R_A)/400}}$$

Where $R_A$ and $R_B$ are the ratings of team A and B, respectively. This means that a team's probability of victory as a function of the Elo rating difference $R_B - R_A$ looks like this,

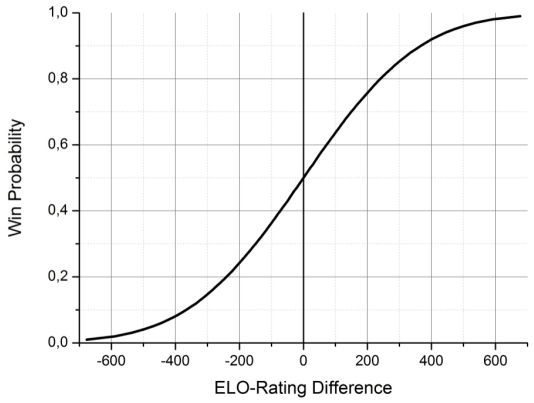


Figure 2: Win probability vs. Elo rating difference (Source)

The nonstatistical parameters of the model are used to tweak the Elo rating of a team before each game. For example, if a team's best player is unable to play, their Elo rating will be decreased accordingly. The specific value of the adjustment is straightforward to compute for factors such as home court, but more complicated for injuries. In the latter case we must determine each player's contribution to the team Elo rating and subtract off the contribution from the injured player.

## Data Manipulation

The other main challenge of creating a prediction model is the acquisition and storage of data. Essentially all of the data used in the predictions was scraped from the web. The Python package BeautifulSoup4 is invaluable for this purpose. For storage, I experimented with multiple different tools and eventually settled on a combination of CSV files and SQLite databases. The CSV files are used primarily for historical data, while the SQLite databases are for actively updated data such as Elo ratings.