**Predicting NBA Regular Season Outcomes-**

**The Problem**

A few months ago, I came across this [article](http://harvardsportsanalysis.org/2017/04/nba-playoffs-yes-regular-season-matchups-matter/) from Harvard Sports Analytics, breaking down the correlation between an NBA team’s regular season matchup outcomes, and their playoff outcomes.

Considering this, I wondered if I could create a predictive model for regular season matchups, seeing the team winner for regular season matchups to predict the outcomes of playoff series.

**Data Collection:**

I scraped all the relevant statistics from BBallReference.com, using Python Selenium to do so. I scraped all the team’s head to head regular season matchup records, as well as their basic stats.

**Data Formatting:**

This might have been the toughest part of this project; for each year the standings were extracted as seen in Table 1. This table needed to be [unstacked](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.unstack.html#pandas.DataFrame.unstack) to match the teams and their opponents together along with their series matchup. I was then able to [join](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.join.html) this data frame with the offense and defense stats for each team.

Unfortunately, the data is not quite yet ready for modelling. As explained in [this article](https://medium.com/@urvashilluniya/why-data-normalization-is-necessary-for-machine-learning-models-681b65a05029) by Urvashi Jaitley, if you don’t normalize the data, features with larger values and ranges will much more heavily influence the model than features with smaller values and ranges.

In order to counteract this, we can use [Sklearns MinMaxScaler function](https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html), which transforms all the data in the data set such that it is in the given range (0-1). The mathematical formula for this is given below.



Unfortunately, this still does not account for changes in pace over the years. As seen in the figure below, the pace of the league has drastically changed over the years.

Chart, line chart

Description automatically generated

In order to work around this, we must group together the data by year, and standardize it within each year. This can easily be done by grouping the data together using Pandas [groupby](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.groupby.html), and then Pandas [transform](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.core.groupby.DataFrameGroupBy.transform.html) to standardize the data by year.

After normalization and standardization, we can see that the new feature values will not change by year, so we can now more accurately compare data across features and across years.

Chart

Description automatically generated

**Feature Selection:**

We can now move on to feature selection. Below we plot a heatmap of the autocorrelations between the different features, for both the offense and the defense.

A picture containing text, different

Description automatically generated

We can see that the features are largely uncorrelated, with the exception of makes and attempts for field goals, three pointers, and free throws. Since we have the percentages, we can delete the attempts from the data. There also seems to be a high autocorrelation between the 2-point field goals and the overall field goals. It would also be safe to remove the 2-point stats and keep the overall field goals.

**Modelling:**

Now for the part we’ve all been waiting for.