Interactive Visualizations of NHL Data Using PySpark and Jupyter Notebooks

William Gary Dawkins
Cheriton School of Computer
Science
University of Waterloo

Analytics in Sport- Project Motivation

The world of professional sports has increasingly embraced the power of data-driven analytics in recent years to optimize performance of teams across leagues all over the world. Most likely, the best-known example of this was in baseball when the 2002 Oakland A's won the world series, detailed in the novel and motion picture 'Money Ball'. In which, a cash strapped Oakland management team was able to field a World Series winning team for the fraction of the cost of what their competitors were spending. They were able to do this by applying a new style of thinking to the game, where more traditional baseball theories were discarded in favor of analytically evaluating players. The idea behind this new style was that traditional baseball minds evaluated players on superficial qualities (appearance, star-power with fans) that were not necessarily representative of a player's ability to help a team win. The new approach used the idea that by studying a player's underlying numbers one could find value in a player that was missed by traditional scouting. Since then, management teams across sports, not just baseball, have integrated such data-based analytics into their scouting departments to evaluate the value of players, and as such, there is a higher demand for dealing with data based on sport.

Interactive Visualizations of NHL Data- Project Goal

The goal of this project is to build interactive visualizations of National Hockey League (NHL) data that could be used as a tool in a larger scale analytics application to the game of hockey.

For this work, a relational database was utilized that contains data collected from the NHL from the year 2010 up until 2019, covering a total of 9 full NHL seasons. Within this data there is information about every game that was played within this timeframe, detailing the events that took place in this game, the type of event (shots, goals, hits etc.), the time of the events and the location on the rink they occurred and that players were involved in these events. It also contains data tables that contain general information about player info (nationality, position played by player) as well as general information about the teams in the league. The data comes from the website Kaggle, on which a user posted this dataset, which can be found at https://www.kaggle.com/martinellis/nhl-game-data. The tables that are present in the data can be related to one another by the schema presented in Figure 1.

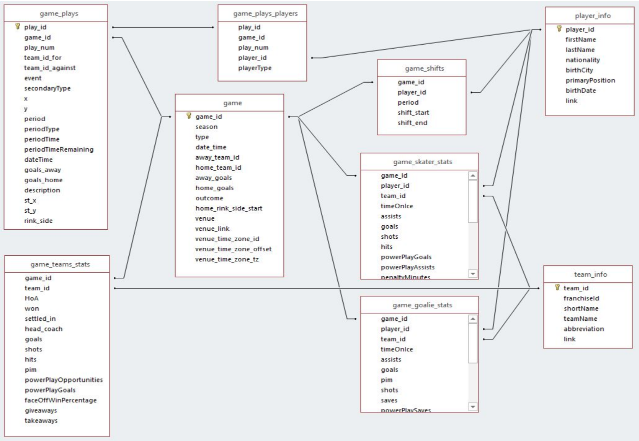


Figure 1: Schema detailing the relationships between tables present in NHL Game Data database

As can be seen in the schema, there is a lot of information that could be of potential interest to study contained within the data. This project has the goal of allowing a user to visually explore the dataset with five main visualizations:

1. A plot that allows the user to explore the leaders of any given game statistic that is tracked by dataset. This visualization will give the user an option to input a given season, a given game statistic (examples are goals, assists, hits etc.) and a given number of players they wish to view. The plot will then update in real time and present the user with a visualization of players ranked by their chosen statistic for their given season, and filter the results to only present the top number of players that the user asked for.

2. A plot that allows the user to visualize the final standings of a given NHL season. Here the user will give a season they wish to view, and a plot will be generated where the teams will be

ranked by points earned that season, and as such ranked by their final standings in that season.

3. A graphic for visualizing the countries of origins of players within the NHL for a given season. Here the user will be able to give a chosen season and be given a world map that is colour coded by country, in accordance to the number of players that come from that country.

4. A plot for visualizing a summary of a player's offensive production in a given season. For this plot the user will give a player that they are interested in and a season. They will be given either a scatter plot or heatmap (designated by the user) of either the goals the player scored during the given season or the shots they took. These plots will allow the user to see where on the ice a player is getting their shots or goals as the plot will be overlayed on a to-scale drawing of an NHL rink, focused on the offensive zone.

5. A scatter plot for visualizing the shots, goals, hits, takeaways and giveaways by both teams in a given game. Here the user will enter a specific game and event they are interested in viewing and will be given a scatter plot of where on the ice each teams' events took place, for use of comparing the two teams' performance in a given game.

It is the hope that these five visualizations will serve as an introduction to some qualitative studies of the NHL dataset, and could set the stage for further analysis.

Achieving Interactive Visualizations – Methodology

The first step of achieving the desired visualizations as described above is to make use of a programming environment that is capable of relational processing that can be applied to our dataset. For this purpose, this project will make use of the PySpark.sql module. This module integrates the SQL capabilities of working with tabular data with sparks ability to work with distributed data. The SQL capabilities will be crucial in manipulating the tabular data and extracting the specific information needed from various tables in order to generate our final visualizations. On the PySpark side, while this particular dataset is not large enough to necessitate the power of distributed computing as the whole dataset takes up roughly 1.2 GB, one could imagine an expanded version of this data set that could take up much more memory.

As stated above, this data set is limited to NHL data from the 2010-2011 season to the 2018-2019 season, however, as a league the NHL is over 100 years old, so such a dataset could be built to cover to entire history of the NHL, already increasing the size of the dataset by a factor of 100. Additionally, the dataset could also be expanded to cover other hockey leagues around the world. In total, there are almost 500 hockey leagues distributed over the professional, junior, college and minor levels, not to mention the ability to expand the data set to include data from international tournaments. This is also only considering men's hockey; the dataset could easily also be expanded to cover women's leagues as well. Such a dataset could be of much use to management teams over all these various leagues for investigating their team's/league's performance, but also for scouting players across leagues, such as NHL management scouting junior players for drafting purposes, or countries evaluating players when selecting their teams for international tournaments, such as the Olympics or the World Cup of Hockey, two very high stakes tournaments with international prestige.

Given these facts, we can understand that what we have in this dataset is a mere snapshot of the data that could be collected across the hockey world, and that the potential to increase our data size significantly exists. It is not unreasonable to assume our dataset could increase by a factor of $10^{\wedge 3}$, and as such necessitating distributed data systems. Therefore, to allow for future expansion, PySpark will be used so that the work in this project would be able to handle an expanded hockey dataset that would need distributed analytics.

Once the proper manipulations have been performed to extract the specific information of interest, the power of Python visualization libraries such as Matplotlib and Seaborn will be used to generate plots. The main process will be to perform the needed manipulations on the Spark dataframes and pass them to Python Pandas dataframes only once they contain the exact information to be visualized, which should be manageable to load into the local memory of the program. The Pandas dataframes will then be fed into the plotting functions.

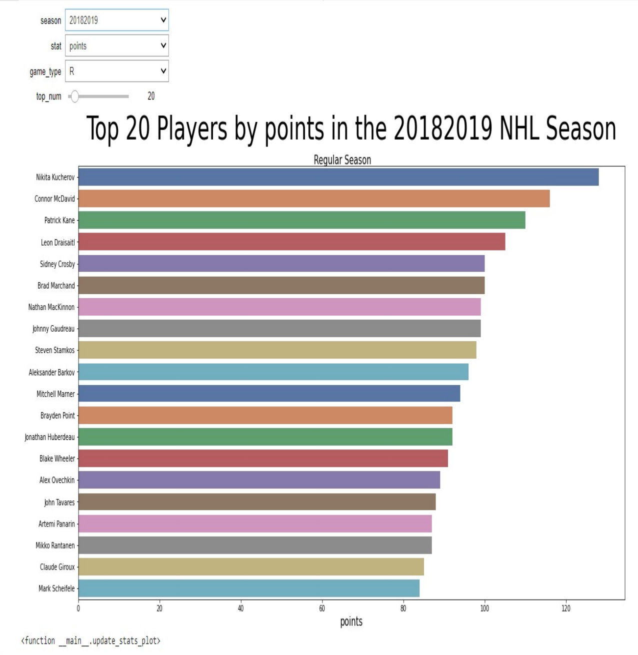
For ease of use, this project will allow the user to interact with and update generated plots in real time, without having to re-execute any code for every update they wish to view. In order to achieve this, this project will be done in a Jupyter notebook and make use of the ipywidgets library. The ipywidgets library allows for use of the 'interact' function in which it is possible to connect plots to interactive widgets such as dropdown menus or text boxes. This creates a visual UI for the user in which they can change the settings for a plot.

Visualizations – Results

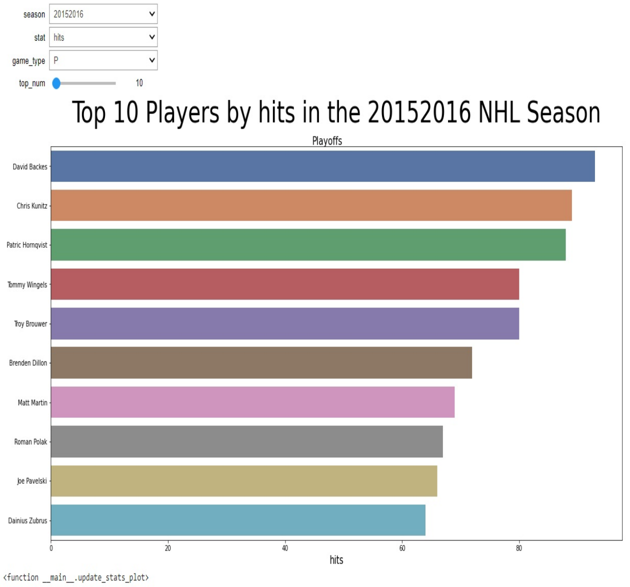
Interactive Player Rankings by Statistic

In order to generate this plot it is necessary to perform .join() with the game_skaters_stats and game tables, and then perform another join with the player_info table. This produces a table that lists games played by every player in the time span of the data collection, their stats for each game and the additional information that will be needed such as what season the game took place in, if it was in the regular season or playoffs and player names. Some filters are then applied. The given season is filtered on, only leaving stats for the season that is of interest, as well as a filter applied to the ‘stats’ column, so that I am only left with the statistic I am interested in. The .groupBy() is then used on the dataframe, where the player_id is given as an argument, and the sum operation is then called on the stat column. I am then left with a dataframe containing the total for the given statistic over the given season for each player in the league. The .orderBy() and .limit() operations are then used to rank the players based on their totals and returns the given number of players the user is interested in. The additional option was implemented to give the user the option between regular season games (i.e. the usual 82 games all teams in the league play in), or the playoffs (the final tournament to win the season championship, after the regular season, where only the top 16 teams qualify).

The season, stat and game type are fed to the spark operations via dropdown menus containing all options for each parameter, and a slidebar is used for the number of players to be listed. The ipywidgets function ‘interact’ is used to generate these widgets, and connect them to a Seaborn barplot, which takes in the dataframe after it has been converted to Pandas. Figure 2 shows example outputs for this visualization.


season 20182019
stat points
game_type R
top_num 20

```
<function __main__.update_stats_plot>
```

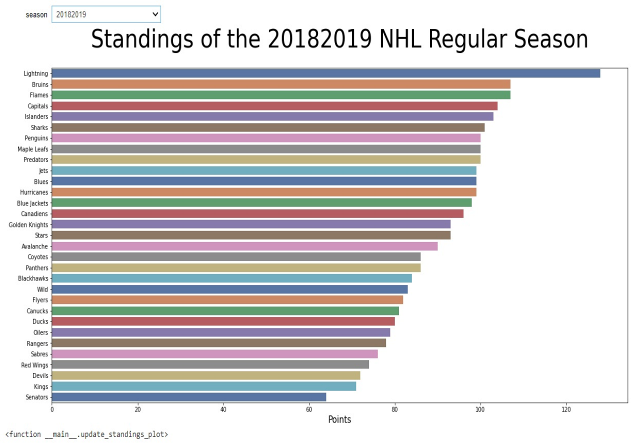

season 20152016
stat hits
game_type P
top_num 10

```
<function __main__.update_stats_plot>
```

Figure 2: A bar plot of the ranked top 20 players by points in the 2018-2019 NHL season (top) and a bar plot of the top 10 players by hits in the 2015-2016 NHL playoffs (bottom).

Team Standings for a Given Season

For this visualization the game_teams_stats, team_info and games tables are joined. This gives a dataframe that contains every game that every team has played in the time over which the data was collected, and what the outcome of that game was. The season is then filtered by the season inputed the user, and also a filter is applied to make sure only regular season games are present in the dataframe. A new column is then defined to calculate the number of points a team was awarded for each game they played, the NHL awards two points for a win, one points for an overtime or shoot out loss and zero points for a lose of any kind, for this calculation I use the coloumn that gives the outcome of the game. A groupBy is then performed by team and ordered in ascending order, the dataframe is then passed to Pandas and plotted in a Seaborn barplot. Again, an ipywidget dropdown menu is used so that the user can select which season they are interested in viewing. Figure 3 show an example output for this visualization.



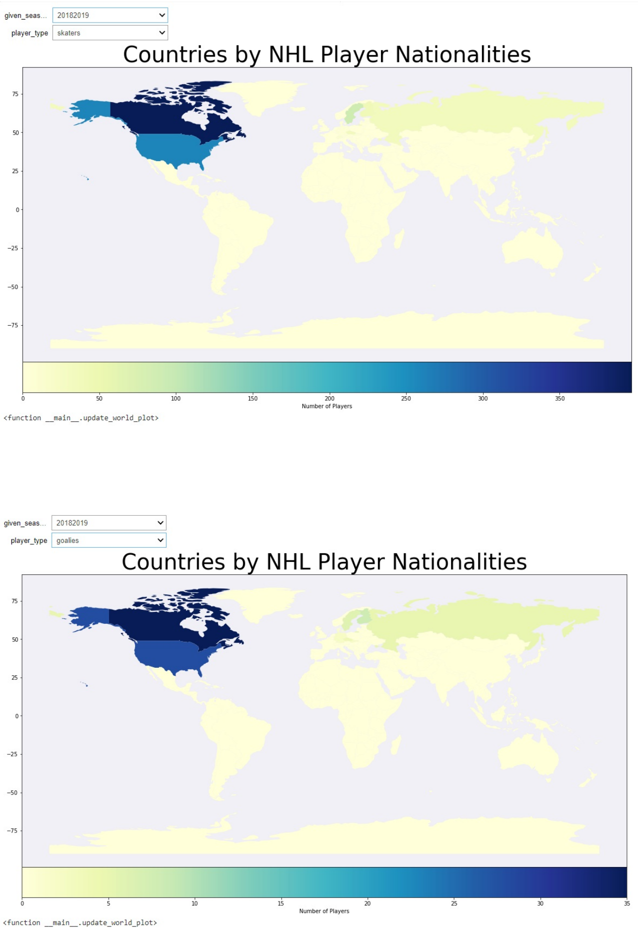
season 20182019

```
<function __main__.update_standings_plot>
```

Figure 3: Bar plot of regular season standings for the 2018-2019 NHL season

World Map of NHL Player's Nationalities

In this section it is possible to map NHL player nationalities for both 'skater' type players (this consists of forwards and defensemen, essentially non-goalies) and players who play the goalie position. Here two separate dataframes are formed, one where game_skater_stats, games and player_info is joined, and another where game_goalie_stats, games and player_info is joined. For both duplicates are dropped based on player_id and season, so that players aren't counted towards totals twice, groupBy and count is then used, applied to nationality and season so that I am left with a dataframe containing the nationality counts for all seasons. The season is then filtered based on the users input. For this visualization the GeoPandas library is used. This library contains dataframes that list countries and their geometry for creating plots. By joining our produced dataframe with pre-existing geopandas dataframes, I am able to plot out nationalities counts in the style of a world map using matplotlib. Again, ipywidgets are used to create dropdown menus where the user inputs the season and player type they are interested in viewing. Figure 4 shows example outputs of this visualization.



given_seas... 20182019
player_type skaters

```
<function __main__.update_world_plot>
```

given_seas... 20182019
player_type goalies

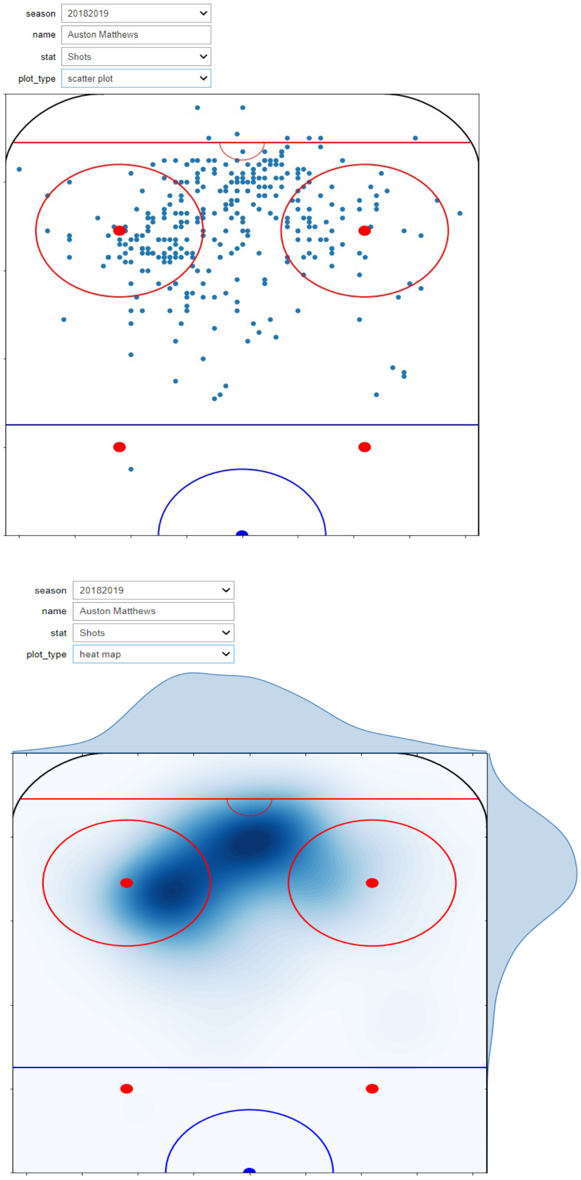
```
<function __main__.update_world_plot>
```

Figure 4: World map coloured by NHL skater nationalities for the 2018-2019 season (top), World map coloured by NHL goalie nationalities for the 2018-2019 season (bottom).

Individual Player Offensive Scatter plots/Heat Maps

For these plots the game_plays, games, game_plays_players and player_info tables are joined. Since the game_plays table is where every event of every game contained in the database is stored, this is by far the largest data table. As such, I prefilter these joined tables by event types shots and goals, and cache each of these tables. I then filter on the given player provided by the user and return the dataframe the user is interested in (either shots or goals). The 'st_x' and 'st_y' columns are selected out; these are the standardized coordinates of where on the ice the players have taken their shots or scored their goals. Standardized here means all events are transformed so that the attacking player is always attacking left to right on the rink, this is included in the data for ease of use. I then filter out the specific season the user has specified and now have a data frame that contains the shots or goals for the given player over that entire season.

Once I have this information, the user is allowed to view it as either a scatter plot or a heat map overlayed onto a geometry that is representative of an NHL ice surface. These rinks are drawn using matplotlib's 'patches' elements, which allow for shapes such as rectangles, circles and arcs to be drawn. If the user has selected the scatter plot option then a matplotlib scatter plot is placed over the rink drawing. If the heat map option is picked then a Seaborn joint plot is used. This is a plot type that is used for visualizing 2D data, and has a contour option, which along with colouring settings can be utilized to make a plot that is colour coded based on where there is a higher concentration of shots or goals. This plot type also introduces curves that are 2D projections of the contour maps on both the x and y coordinates, this allows the user to see where in both x and y the player is obtaining most of their shots and goals. Examples of these plots can be seen in Figure 5.


season 20182019
name Auston Matthews
stat Shots
plot_type scatter plot

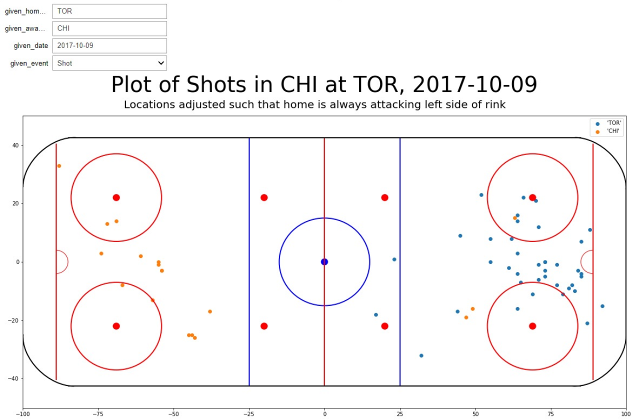
season 20182019
name Auston Matthews
stat Shots
plot_type heat map

Figure 5: Scatter plot of shot locations for Auston Matthews over the 2018-2019 NHL season (top), Heat map of shot locations for Auston Matthews over the 2018-2019 NHL season (bottom).

Game-By-Game Scatter Plots

For this section the games, game_teams_stats and team_info tables were joined. Then, two separate dataframes were formed, one for away teams in every game and the other for home teams. When a given season and given home and away teams are entered into ipywidgets text boxes, the given stat entered into a drop-down menu is presented as a scatter plot, again overlayed onto ice-rink geometry.

Some basic transformations were applied to the x- and y-coordinates of the events, such that the home team is always attacking left to right, and the away team the other way. This is down such that the home teams' shots and goals are concentrated on the right side of the plot, and the away the other side, unless either team shot the ice down the puck from their own side, in this special case a shot would be registered from the opposite end (examples of this can be seen in the given plots). Figure 6 shows two examples of scatter plots generated with this visualization.


given_hom... TOR
given_awa... CHI
given_date 2017-10-09
given_event Shot

Plot of Shots in CHI at TOR, 2017-10-09
Locations adjusted such that home is always attacking left side of rink
'TOR'
'CHI'
40
20
0
−20
−40
−100
−75
−50
−25
0
25
50
75
100

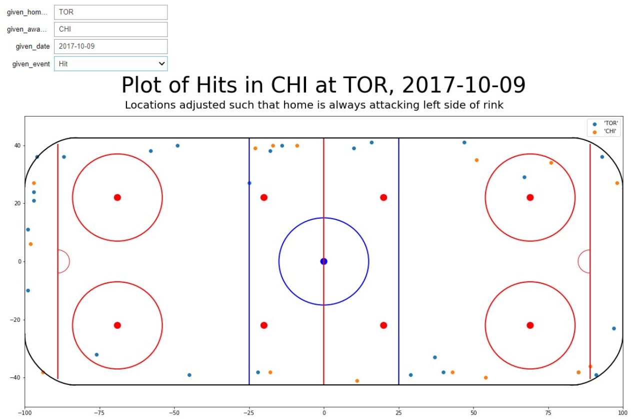

given_hom... TOR
given_awa... CHI
given_date 2017-10-09
given_event Hit
Plot of Hits in CHI at TOR, 2017-10-09
Locations adjusted such that home is always attacking left side of rink
'TOR'
'CHI'
40
20
0
−20
−40
−100
−75
−50
−25
0
25
50
75
100

Figure 6: Scatter plot of shots in Toronto Vs. Chicago game played on 2017-10-09 (top), Scatter plot of hits in Toronto Vs. Chicago game played on 2017-10-09 (bottom).

Evaluation of Results and Future Work

Overall, I am pleased with the outcome of the visualizations I initially set out to produce. I believe all five allow the user to easily and intuitively work their way through the NHL dataset and explore some summary statistics in a visual way to gain some insights. However, there is one instance where I believe improvements could be made and that is to the last visualization, the scatter plots of stats from specific games. A main goal of this project was ease of use for a user working with the notebook. This was the motivation behind making the visualizations interactive, and not requiring code to be run every time a new plot was to be generated. A challenge with these plots was devising a method for the user to choose a game to view. In the other visualizations most options for the user are presented in dropdown menus. Because of this, it is possible to explore the data without having a predetermined goal. For example, in the players ranked by stats visualization, the user can see all seasons and stats available to them, and pick whichever they are interested in, it is not required for them to specifically enter ‘goals’ or ‘20182019’ season for example. The gives a more open-ended experience for the user. With the scatter plots for any game’s stats, the user must explicitly type in the home team, away team and date of the game they wish to see. Meaning they must already have a game in mind, and know the date. It is not completely unreasonable to expect the user to have this information, perhaps they are studying a specific team, or season, and have a game schedule in front of them with all this information, but this still reduces the ‘explorability’ that I was hoping to achieve. I ran into this issue due to the sheer number of games; it would be unreasonable to have them all in a drop down menu, but with more work a more elegant solution could probably be found then manually entering the date.

Aside from this, I believe this project could act as a first step in a wider statistical analysis of hockey-based data. More high-level analytics could be explored, like quantifying in a more numerical sense where goals are scored from on the ice, and translating that to evaluating player based on where they get their shots. A deeper dive could be done into the takeaway and giveaway stats, in order to evaluate how responsible a player is with the puck, or how defensive they are. All of these calculations would most likely require similar SQL transformations to what I did for my visualizations, and I believe my visualizations would be a good way to qualitatively supplement more quantitative analysis.