Sid Nagaich

**LA Clippers Internship Data Challenge**

To see a complete walkthrough of each solution, please view the accompanying Jupyter Notebook, which contains code and annotations. I included some small screenshots of code when useful, but for larger data manipulations I just explained what we were trying to achieve.

**1) [SQL] Do weekend games (Friday, Saturday, Sunday), on average, have a higher attendance than weekday games (Monday through Thursday)?**

To solve this question, we first use SQL to count all scans when grouped by the event name in order to get a total number of scans per game. In another query, we selected only the home games and split the event name on commas in order to isolate the day of the week of each game. We then separated weekends and weekdays into their own tables, and used an inner join to calculate the average number of scans per weekend game and per weekday game.

Text

Description automatically generated with low confidence

We found that weekend games were weakly associated with higher attendance. We further looked at difference between each day of the week:

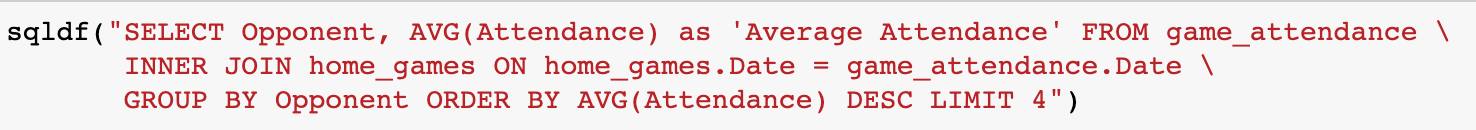
|  |  |
| --- | --- |
| Table  Description automatically generated | Table  Description automatically generated |

Average attendance per day does not differ by more than two standard deviations from the mean for a given day, so we cannot conclude that any differences between the days are statistically significant (assuming an alpha = 0.05).

**2) [SQL] Identify and rank the top 4 opponents with the highest average number of attendances.**

Since we know the attendance of each game and who the games were played against, we can then calculate the average attendance per opponent.

To do this, we can join our tables from queries in Question 1 after standardizing each of the [‘Date’] columns to a DateTime object. We join the game\_attendance table with our home\_games table on the date which the games were played.



Table

Description automatically generated

We sort our result by the average attendance per opponent in descending order and limited the result to 4 so that we could easily view the top 4 opponents with the highest average number of attendances.

**3) Identify and rank the top 10 sections that are, on average, the most filled to their capacity.**

To answer this question we join two SQL SELECT statements which gather the required information. We first sum the available seats in each section, and next calculate the average number of tickets that scanned into each section. We will use these values to calculate the proportion of the section that is full, on average. Finally, we will sort by how full the section is on average and limit our results to the top 10 sections.

Text

Description automatically generated with low confidence

Table

Description automatically generated

**4) Is there a correlation between the opponent team having a higher Vegas Odds Score (indicating higher probability of winning a championship) and higher attendance?**

To answer this question we use our previous statement from Q2 and join it on the team table by matching the Team Full Name with the Opponent names we used before. From the team table we can get the Vegas Odds per opponent and use it alongside the weekly attendance to visualize and analyze our data to see if there is a correlation between Vegas Odds and attendance.

A picture containing text

Description automatically generated

Table

Description automatically generated

We then visualize the data using PyPlot and calculated R and R2 values:

Chart, scatter chart

Description automatically generated

R: 0.216079

R-squared: 0.046690

We observe a weak positive correlation between the opponent's Vegas odds and game attendance such that higher Vegas odds are associated with higher attendance.

**5) On average, which 15-minute period before or after the start of a game has the highest number of people scanning. (Example 30 – 16 minutes prior to the start of a game)**

To answer this question we subtract the game's start time from each ticket scan time (and adjust for the time difference). Negative results will indicate that the spectator scanned after the game had started. We then define 15-minute intervals in which we will count the ticket scans. We can then compare which time interval is the most popular for scanning (See Jupyter Notebook for SQL code).

|  |  |
| --- | --- |
| 1. Getting the minutes away from start time for each scan: | 1. Defining Buckets: |
| A screenshot of a computer  Description automatically generated with medium confidence | Table  Description automatically generated |
| 1. Counting scans into the buckets: | 1. Diving by Number of Home Games for Average Scans per Interval |
| Table  Description automatically generated | Table  Description automatically generated |

***Graph of results on next page***

Chart, bar chart

Description automatically generated

*Note: Ticket Scans are average ticket scans per game, not season totals*