Analysis of the Classic Rock Dataset

This report deals with the 'classic_rock.db' dataset that was provided in an earlier lab of this course. The dataset in this file is comprised of two tables, namely, "rock_plays" and "rock_songs". The "rock_plays" table gives us information about the songs that were played on specific radio stations in the week of June 16, 2014 (Monday) to June 22, 2014 (Sunday). Some brief information regarding the columns of the table is shown below:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 37673 entries, 0 to 37672
Data columns (total 11 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   SONG RAW      37673 non-null  object
 1   Song Clean    37647 non-null  object
 2   ARTIST RAW    37668 non-null  object
 3   ARTIST CLEAN  37665 non-null  object
 4   CALLSIGN      37673 non-null  object
 5   TIME          37673 non-null  int64
 6   UNIQUE_ID     37673 non-null  object
 7   COMBINED      37673 non-null  object
 8   First?        37673 non-null  int64
 9   date_time     37673 non-null  object
 10  day_of_week   37673 non-null  int64
dtypes: int64(3), object(8)
memory usage: 3.2+ MB
```

The table has eleven columns. The first column, "SONG RAW", contains the name of the song, sometimes in poor format. The second column, "Song Clean", takes care of this and has the songs in a standard format with the first letter of each word in capital letters. Similarly, there are two columns for the name of the artist in a similar way. The fifth column called "CALLSIGN" contains the name of the radio station on which this song was played. The next column "TIME" tells us the time, in seconds, at which this song was played with the starting point being the year 1970. As expected, the value of the cells in this column is huge, approximately 1.4e9. This column will not be useful in our analysis since the last two columns provide the same information in a much better format. The "date_time" column gives the date in the yyyy-mm-dd format and the time in hh:mm:ss format. Furthermore, the "day_of_week" column tells us the specific day of the week the song was played with 0 for Monday (June 16, 2014) and 6 for Sunday (June 22, 2014). The column "UNIQUE_ID", as its name suggests has a unique string to identify the particular play of a song at a given radio station. The "COMBINED" column contains the name of the song and the artist name in the format "Song Clean" + "by" + "ARTIST CLEAN". Finally, the "First?" column contains either a 1 or a 0 depending on whether this is the first time the song was played during the week.

Using the data of the above table, we are given another table titled “rock_songs”. This table contains the songs from the previous table clubbed together in the format given below:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1650 entries, 0 to 1649
Data columns (total 4 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   Song          1650 non-null   object
 1   Artist        1650 non-null   object
 2   Release_Year  1650 non-null   float64
 3   PlayCount     1650 non-null   int64
dtypes: float64(1), int64(1), object(2)
memory usage: 51.7+ KB
```

This table tells us the number of times a particular song was played during the week of June 16, 2014 on all the radio stations that occur in the “CALLSIGN” column of the previous table. This table only contains four columns. The first two columns directly come from the previous table. The “Release_Year” column tells us about the year in which the song was released in the form of a float, e.g., 1970.0, and contains new information which was absent in the previous table. Lastly, the “PlayCount” column contains the number of times the song is cumulatively played.

Initial Plan for Data Exploration

We will begin by cleaning the data. The first step will be to suitably address all the missing elements in this data. Subsequently, we will look at the duplicate elements in the “rock_songs” table. This will tellus if there are any songs that appear in the table multiple times because of small differences in the artist name, which are undesirable. After correcting that, we will go to the “rock_plays” table and correct the corresponding artist names there, since that table is the source for the artist names that appear in the “rock_songs” table.

Next, we will turn to feature engineering for the “rock_plays” table. In this step, we will drop the columns which have unique elements in all cells and change the format of the “date_time” column to make it more useful. We will then use one-hot encoding for the radio stations that appear in the “CALLSIGN” column since there are only 25 of them.

These two steps will allow us to look at a wide variety of patterns in our data. For example, using the “First?” column that tells us if a particular entry is the first time the song is played in that week, we can look at the days of the week when most songs are played for the first time. Similarly, the “rock_songs” table will allow us to plot the release years of the songs and verify whether the “golden age of rock” is indeed the supposed period of the late 1960s and 1970s. We will formulate three such hypotheses regarding this dataset after finishing feature engineering and looking at some broad aspects of the data.

We will then spend some time discussing the results of a formal significance test performed on the hypothesis that we can make most comfortably.

Data Cleaning

We start off by cleaning the column names in the table “rock_plays”. One example is the change in name of the column “SONG RAW” to “Song Raw” and “CALLSIGN” to “Call Sign”. Next, we look at the missing elements in the table.

The table shown on page 1 tells us that for some songs, even though their raw names are present in the table, the cleaned name is absent. We use the “pd.isnull()” method to find the location of the null elements and find that all the missing elements in the “Clean Song” column are present in two distinct locations of the table: the first is between indices (25325, 25342) and the second is between indices (37665, 37672), where the above indices are included. This gives a total of 18 + 8 = 26 missing elements, which when added to 37647 (the non-null elements of the column “Clean Song”), gives us 37673, which is the number of rows in the table, thus accounting for all the missing elements in this table. These missing elements are easily added since the raw name of the song is available.

Similarly, there are some rows that have an entry for “ARTIST RAW” but no entry for “ARTIST CLEAN”. We add these artist names to the “ARTIST CLEAN” column in the correct format. Next, there are some empty cells in the ‘ARTIST RAW’ column. For all but one of these songs, we find that in other places in the table the same song occurs and has a corresponding artist name. We fill in the “ARTIST RAW” column for these songs and suitably add in the entries for the “ARTIST CLEAN” column. For just one row, with index number 37672, we find that the raw artist name is absent, and this song doesn’t appear anywhere else in the table. We drop this row. This takes care of all the missing elements in our dataset.

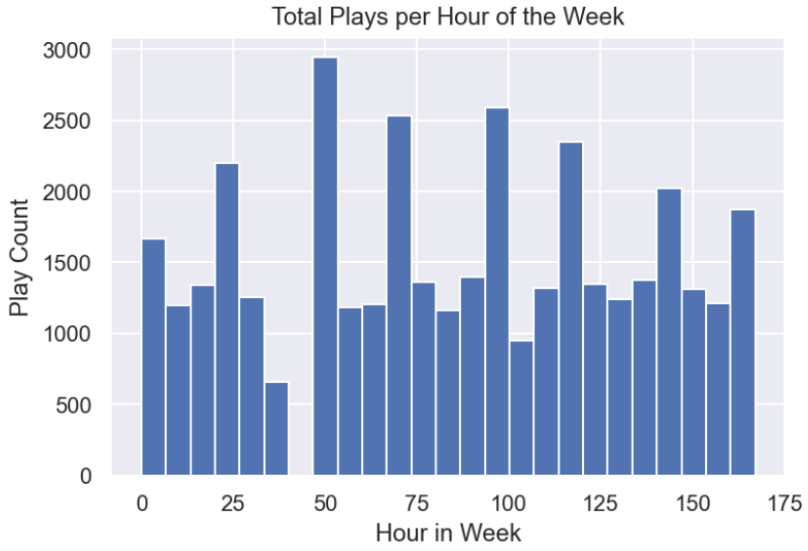
Next, we look at the duplicate elements in the “Song” column of the “rock_songs” table. We find that even though the table has 1650 rows, it only has 1621 unique elements. On further inspection, we notice that some songs, even though they have the same name, are written by different artists with different release years. We ignore these duplicates since they should be in different rows. However, some songs with the same names have trivial differences in the artists’ names and have the same release years. For example, the song “Funk #49” appears twice with the artists’ names “James Gang” and “The James Gang”. There are nine such, duplicates and we address them by adding their “PlayCount” entry to the row that contains the artist name that appears multiple times in the table and deleting the row corresponding to the duplicate entry. We then change the artist names of these corresponding songs in the “rock_plays” table since that table is the source for the “rock_songs” table.

Feature Engineering and Exploratory Data Analysis

The number of unique elements in the “TIME” and “UNIQUE_ID” columns is equal to the number of rows in the table. Thus, these two columns are useless for making any prediction. We will, henceforth,

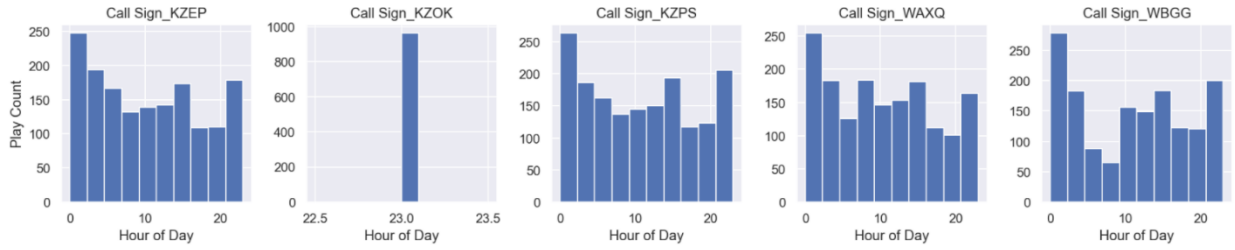
drop these two columns. Next, we change the format of the ”date_time” column. This column contains, both, the date when the song was played as well as the exact time. Since we already know that the entire table contains songs only played during the week of June 16, 2014, the information provided in the ”day_of_week” column makes the date redundant. Thus, we modify the “date_time” column to remove the dates and only keep the time of the play. We also relabel this column to “Time in Hours”

Furthermore, we remove the minutes and seconds from the time since it makes our data more complicated without giving much more detail. Thus, we only retain the hour of the day the song was played having a number between 0 and 23. We add another column titled “Hour of Week” that takes in values between 0 and 167 (=24*7 -1) and tells us the hour of the week when the song was played. This column will allow us to look at the distribution of song plays across the week, while considering daily fluctuations and tell us if there is any pattern that the radio stations follow. We plot a histogram that shows us the total songs played per hour of the week:



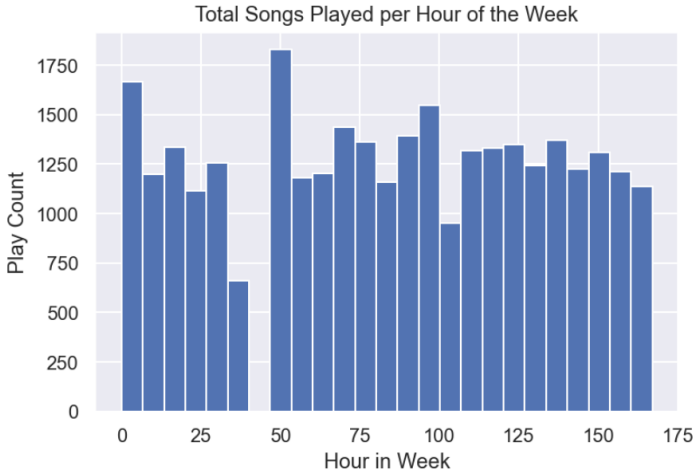
This shows that there are some hours of the week for which there is no song played. Additionally, every 24 hours we see that there is a spike which suggests that there is a particular time during the day when the number of songs played significantly increases. On further inspection, we find that for hours 38 to 46, there is no song played. This appears to be a systematic issue since no radio station has any song played during these 8 hours.

Next, we convert the entries in the “CALLSIGN” column that contains the name of the radio stations to numbers using one-hot encoding. There are 25 different entries for this column and after we are done with one-hot encoding, we shall erase this column and produce 25 new columns in its place. Then, we look at the histogram plots for play count versus the hour of the day for each of the radio stations. While 20 histograms look normal, 5 have all their plays at a particular time of the day. The radio stations are - KZOK, WCSX, WMGK, WNCX and WZLX. A sample of some of the histograms is shown below:



We see that out of the five radio stations whose data is plotted, four have their plays distributed throughout the day whereas one, namely the station KZOK, has all the plays reported at 11pm. The same is true of the stations WCSX, WMGK, WNCX and WZLX.

These five radio stations were biasing our data in the previous histogram giving the impression that more songs played at a particular time of the day when in reality these stations don't have their data arranged by the time of the day and thus bias the data for "23" hours (11pm). Removing the data from these stations only for our present purpose, we get the following table in which there are no discernible peaks or troughs, and the plays are much more evenly distributed.



Similarly, we plot a few other graphs, which are discussed in the next section, to get a deeper insight into the quality and behavior of our data:

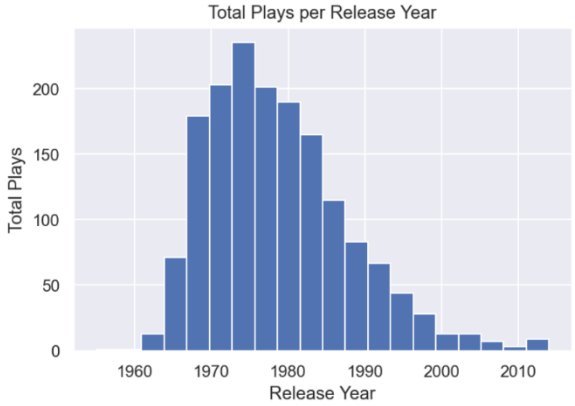
- The total songs played per release year to get information about the best years for rock music.
- The total number of songs played every day of the week to find any patterns regarding the volume of music played at different times of the week.
- The number of times different songs are played. This will tell us the degree to which the most famous songs overshadow the others.
- Number of new songs played every day. We get this by plotting the number of entries in the "First?" column per day of the week. This shows us the time of the week when most new music is played and other times when a smaller pool of songs dominates the radio stations.

There are many other plots that one could look at, e.g., we could restrict ourselves to some of the top artists by only looking at those who had more than say 5 songs in the list and look at their pairplots between the number of songs, average plays, and release years.

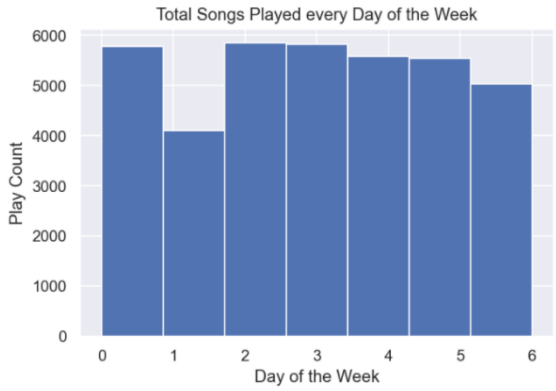
Key Insights of Exploratory Data Analysis

Our analysis of this dataset so far has given us a good insight into the nature of this dataset. We find that this data can be readily used to make various assessments about the patterns that rock radio stations follow regarding the songs they play. Our inspection of the data showed us that even though at first it appeared that a significantly greater number of songs were played at 11pm every night, that was in fact not true and instead there was a bias in our dataset with five stations reporting all their songs at exactly this time of the day. Some other key insights we got about the dataset are as follows:

- We find that the most played rock songs were released between the late 1960s and the early 1980s. This agrees with the expectation we had prior to examining this dataset since this period is broadly considered to be the best time for rock music.

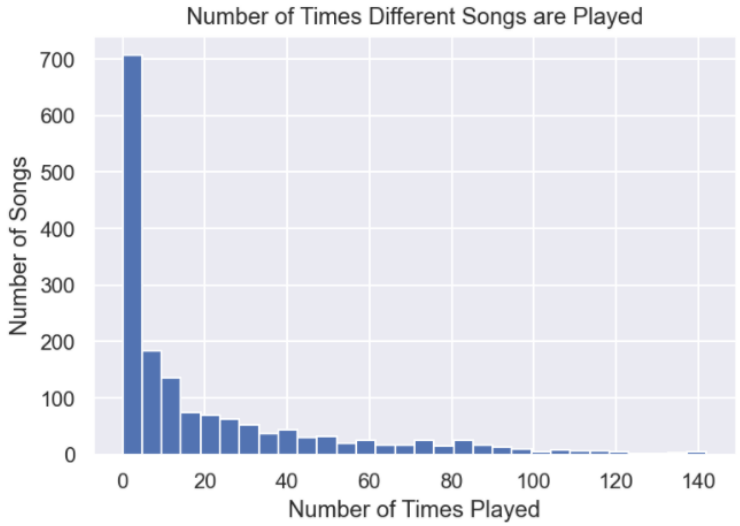


- We find that roughly the same number of songs are played every day of the week as shown in the figure below. While there is a considerable drop for the second day of the week (Tuesday), this is completely explained by the fact that there was no data available for 8 hours on Tuesday, between hour 38 and hour 46 of the week, which corresponds to missing data between 2pm and 10 pm. Since this period is a third of the day, a dip by roughly a third is to be expected

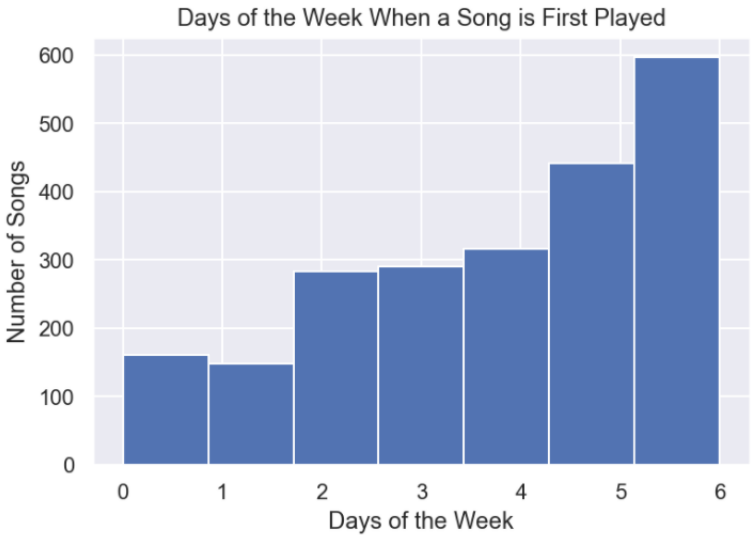


compared to the other days of the week, which is exactly what we observe. This also tells us that the distribution of the total songs played throughout the week is likely uniform.

- Some songs are played many times more than most other songs. While most songs are only played a handful of times, a total of 36 songs are played more than 100 times in the week. The top 3 most played songs in descending order are Dream On by Aerosmith, Sweet Emotion by Aerosmith, and All Along the Watchtower by Jimi Hendrix.



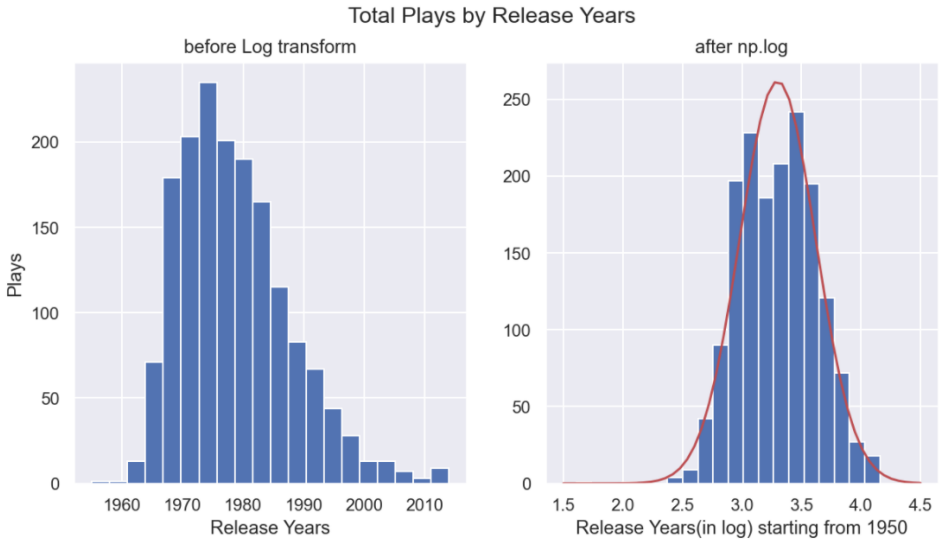
- We find that almost 50% of the songs are played for the first time over the weekend and the fewest number of unique songs are played early in the week. This is a bit counter-intuitive since we would have thought that as the week progresses there would be more repetition. However, it appears that radio stations play new music on the weekends to maintain the interest of their listeners.



Based on these insights, we are now ready to formulate various hypotheses regarding this dataset.

Hypothesis Formulation

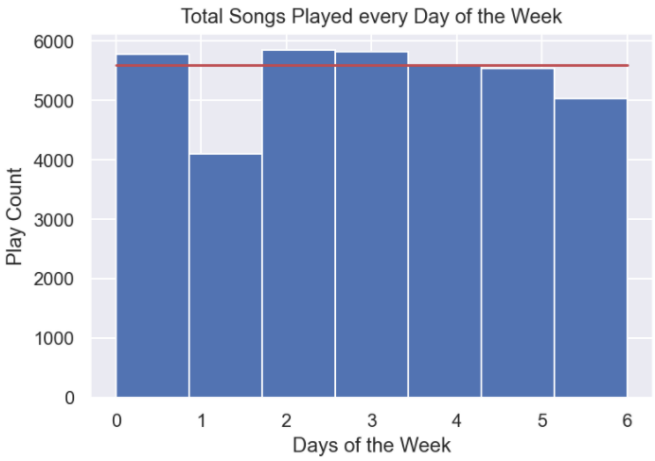
1. The first hypothesis is based on the histogram showing the release years of the various songs played. This curve has a positive skew ensuring that it cannot have a normal distribution. However, that is to be expected since the first rock songs only come from around 1950 and there are no songs before that. While for a normal distribution, the tail extends to infinity in both directions. A natural way to take this into account is to consider a log-normal distribution for the release years. This is because, the log only outputs positive terms, so if we re-orient our x-axis such that the year 1950 is set to zero, then the log-normal distribution will be zero for all years prior to 1950. A log transformation of this histogram changes it in the following way.



   The red curve is a normal distribution with the mean and standard deviation of the histogram obtained after the log transformation. Based on a superficial examination, this hypothesis could be reasonable, so we state this as our first hypothesis, namely **– The total plays as a function of the release year follow a log-normal distribution.**

2. Our second hypothesis is based on the histogram showing the total songs played on the different days of the week. Except for Tuesday (marked with index 1), every other day has a very similar number of songs played. This is to be expected because every day a rock radio station

will spend the bulk of the time playing rock songs with probably a small reduction during the weekend when they would spend slightly more time on their advertisements.



Moreover, as previously explained, Tuesday has a bunch of missing data, corresponding to a third of the day, because of which the numbers for that day are off by a factor of about a third. The red line in the histogram shows the best-fit uniform distribution for this data, ignoring the plays on Tuesday. As we can see, the errors are not huge and the calculated standard deviation of this fit with the actual data is only 136 while the mean is 5600. **Thus, our second hypothesis is that the number of songs played every day of the week follows a uniform distribution.**

3. Our third hypothesis is on the histogram showing the number of different times a song is played. As is evident, the most famous songs are played much more than the others. In fact, a popular science understanding of this phenomenon is the so-called 80-20 rule, namely the top 20% of the songs receive 80% of all airtimes. This is a special case of the Pareto distribution. **Hence, our third hypothesis is that the songs played on the radio follow a Pareto distribution.**

Formal Significance Test for the Third Hypothesis

Our hypothesis is that the songs played on the radio follow a Pareto distribution. This distribution has the form

$$f(x) = \frac{\mathrm{A}}{x^{1.16}},$$

where A is depends on the minimum possible value of x. While for a general Pareto distribution, the exponent in the denominator can take any positive value, we restrict ourselves to the form that gives the 80-20 rule.

To conduct a formal significance test on this hypothesis, we postulate a null and an alternative hypothesis as follows:

- Null Hypothesis (H_0): The songs played on the radio don't follow a Pareto distribution.
- Alternate Hypothesis (H_1): The songs played on the radio follow a Pareto distribution.

We now conduct an Ordinary Least Squares (OLS) regression test to see if the null hypothesis should be accepted or rejected. We set the p-value for our regression test to be 0.05 and will only reject the null hypothesis if the p-value is smaller than 0.05. We set our equation for regression as follows

$$y = \beta \frac{A}{x^{1.16}}.$$

We don't allow a constant term to be added in our regression since that changes the nature of the distribution and the resultant distribution is no longer a Pareto distribution. Below is the result of our regression test.

OLS Regression Results

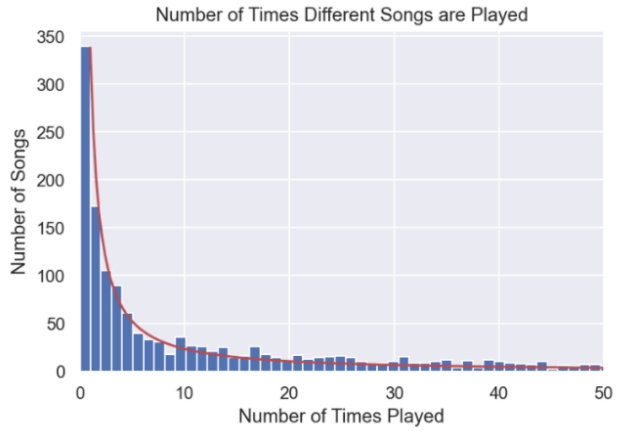
Dep. Variable:	y	R-squared (uncentered):	0.816
Model:	OLS	Adj. R-squared (uncentered):	0.814
Method:	Least Squares	F-statistic:	523.3
Date:	Tue, 08 Mar 2022	Prob (F-statistic):	3.42e-45
Time:	12:25:08	Log-Likelihood:	-503.90
No. Observations:	119	AIC:	1010.
Df Residuals:	118	BIC:	1013.
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
x1	1.1008	0.048	22.875	0.000	1.006	1.196

Omnibus:	105.794	Durbin-Watson:	0.540
Prob(Omnibus):	0.000	Jarque-Bera (JB):	1107.478
Skew:	3.049	Prob(JB):	3.27e-241
Kurtosis:	16.644	Cond. No.	1.00

Notes:
[1] R^2 is computed without centering (uncentered) since the model does not contain a constant.
[2] Standard Errors assume that the covariance matrix of the errors is correctly specified.

We see that β, our regression coefficient is set equal to 1.108 with a very small standard error and an incredibly small p-value. The p-value is smaller than 0.0005, which is why it is rounded off to 0.000 in our regression analysis. **Given, such a low p-value, we reject the null hypothesis and accept the alternative hypothesis that the songs played on the radio follow a Pareto distribution.** The actual data is shown in the histogram below with the Pareto distribution shown in red.



Suggestions for the Next Steps

The dataset offers us the possibility of conducting a wide variety of statistical tests out of which we have chosen to conduct a particular one in-depth. Some other possibilities of statistical tests we have for this dataset include conducting OLS regression tests on the other two hypotheses stated in a previous section. Furthermore, we can formulate a few other hypotheses as well and conduct in-depth statistical tests on them. Some possibilities include the distribution of the number of songs played as a function of the number of artists. We can also look at the distribution of the times when a song is played for the first time on the radio. Lastly, we could also look at the distribution of the release years of the songs as a function of the time of day when they were played to see if the songs from the 1970s (the golden age of rock) are played more during the day or at night.

Quality of the Data

There were very few missing values and they too occurred in some patterns, making it very easy to clean our data. The columns were self-explanatory and allowed us to perform feature engineering without any major issues. There were a couple of redundant features which were easy to identify, and we got rid of them in the early steps of data analysis. The timestamps from 5 radio stations out of 25 were not included in the data properly. All the songs played on these stations were only reported as being played at 11 pm. Even though it was easy to identify these radio stations and get rid of their data for some relevant steps, this was the one place where the quality of the data could have been better. Overall, the quality of data was good, and it allowed us to perform a variety of statistical tests without any major difficulty and get a good insight about the rock radio stations. As additional data, if we could get similar datasets about the songs played by rock stations during other weeks, then we could perform tests to see if our understanding of this dataset continues to hold or if there was something special during the particular week of the dataset that gave rise to our insights but breaks down on larger time scales. Finally, data regarding the time duration of songs could also be useful in identifying interesting patterns.

As an addendum, I have attached the Jupyter notebook on which all the analysis has been conducted. It is not necessary to look at that notebook to follow this report. However, if the reader chooses to verify anything in this report, then the python code can be readily used.