# Song Data Analytics

## Table of Contents

1. Objective
2. Result
3. Future Development
4. Repository
5. Data Source
6. Deliverables
   1. Single Bash Script
   2. Exploratory Data Analysis (EDA)
   3. Preprocessing, Encoding, and Data Manipulation
   4. Song Analytics Data Challenge Pipeline
7. Conclusion and Final Thoughts

## Objective

Set up reproduceable and semi-automated sets of extraction, transformation, and loading operations to answer questions specified by the instruction utilizing commonly used scripting languages

## Result

All questions are provided with answers and some discussion, as well as adequate explanations about the code and thought process used to navigate the questions. Some alternative algorithms are also provided. Python 3.0 is selected for this task.

## Future Development

The items highlighted as follows can be completed as needed.

1. Data Ingestion Automation from the data source:

Dataflow between the source and the designated data warehouse (the destination system) needs to be designed for this project. Data can be extracted, downloaded and loaded in different manner including continuous, asynchronous, real-time or batched. Depending on the source format and protocol, the process may require some additional type of transformation or conversion. Incremental batch loading with a given frequency, or bulk batch with truncating and reloading are recommended based on the data’s size, source, and latency.

1. Integration with the Front-end and Deployment of the model server:

For the client’s front-end a server/instance is needed to be dedicated/shared to the task/schedule.

A Single Page Application (SPA) is a common choice for building front-end applications for this task. It can be built through using AngularJS or React frameworks. For a faster turnaround a micro web framework for Python, such as Flask, is recommended for building the RESTful APIs. Using the Flask and Server-side sessions and serializing with Pickle library is aligned with the standard and best practice of containerization in the production environment. This enables the model to be readily available if/when production phase starts.

## Repository

A public GitHub repository located at the link below contains all component deliverables. Your team should be able to clone this repository and reproduce the analysis.

All configurations and requirements can be found in the repository along with this document.

The link to the git-hub Repository: <https://github.com/BijanVafaei1992/Song_Analytics>

## Data Source

The available data source was provided in csv format. It can be accessed through the project repository at the link below;

<https://github.com/BijanVafaei1992/Song_Analytics/blob/master/data/chart2000-songmonth-0-3-0054.csv>

The dataset stores Music Billboard Chart data across multiple countries from Jan 2000 to June 2019. The Dataset creates an aggregated scoring system to assess how popular music was performing worldwide in each month. Each row of data in the chart represents a song’s placement on the top music charts in one given month, including the artist, its aggregated score, and its position on the charts in various countries around the world. Each datapoint may provide the position of the song in one given month in six countries of US, Germany, France, Canada, and Australia. The countries’ specific position may be blank which is indicated by “-”. There were 11,700 datapoints available in this dataset with 11 features. A quick QA/QC and cross validation was conducted and the datapoint provided in the dataset was assumed accurate.

TABLE 1 – The Snapshot of the Music Chart Dataset (Jan 2000 to June 2019)

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **index** | **month** | **position** | **artist** | **song** | **score** | **us** | **uk** | **de** | **fr** | **ca** | **au** |
| 0 | Jan-00 | 25 | Will Smith | Will 2K | 1750.797 | 57 | 21 | 62 | 32 | - | 3 |
| 1 | Jan-00 | 11 | Whitney Houston | My Love Is Your Love | 2772.888 | 4 | - | - | 97 | - | - |
| 2 | Jan-00 | 23 | Train | Meet Virginia | 1838.804 | 20 | - | - | - | - | - |
| 3 | Jan-00 | 28 | TLC | Unpretty | 1697.405 | 25 | - | - | - | - | 41 |
| 4 | Jan-00 | 43 | The Goo Goo Dolls | Black Balloon | 1478.309 | 33 | - | - | - | - | - |
| 5 | Jan-00 | 22 | The Foo Fighters | Learn To Fly | 1883.219 | 19 | - | - | - | - | 48 |
| 6 | Jan-00 | 46 | The Dixie Chicks | Cowboy Take Me Away | 1428.473 | 27 | - | - | - | - | - |
| 7 | Jan-00 | 44 | The Backstreet Boys | Show Me The Meaning Of Being Lonely | 1457.522 | 16 | 97 | - | - | - | - |
| 8 | Jan-00 | 39 | Sugar Ray | Someday | 1557.48 | 27 | - | - | - | - | - |
| 9 | Jan-00 | 16 | Smash Mouth | Then The Morning Comes | 2161.426 | 11 | - | - | - | - | - |
| … | … | … | … | … | … | … | … | … | … | … | … |
| 11695 | Jun-19 | 8 | Ava Max | Sweet But Psycho | 1661.839 | 10 | 45 | 39 | 69 | 12 | 32 |
| 11696 | Jun-19 | 14 | Ariana Grande | 7 Rings | 1346.183 | 12 | 80 | 99 | 27 | 9 | 30 |
| 11697 | Jun-19 | 35 | Ariana Grande | Break Up With Your Girlfriend, I'm | 875.253 | 20 | 96 | - | - | 33 | - |
| 11698 | Jun-19 | 38 | A Boogie Wit da Hoodie | Look Back At It | 849.373 | 28 | - | - | 45 | - | - |
| 11699 | Jun-19 | 43 | 5 Seconds of Summer | Easier | 781.932 | 48 | 27 | - | - | 37 | 12 |

A summary of descriptive statistics that quantitatively describes the features of the dataset is provided in the Table-2 as shown below.

TABLE 2- Summary of Descriptive Statistics of the features

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Features** | **Data Type** | **No. of Value** | **No. Null Value** | **No. of Unique Value** | **Mode** | **Frequency** | **mean** | **std.** | **min** | **50%** | **75%** | **max** |
| month | Datetime | 11700 | 0 | 234 | - | 50 | - | - | Jan 2000 | - | - | June 2019 |
| position | Integer | 11700 | 0 | 50 | - | - | 25.5 | 14.43 | 1 | 25.5 | 38 | 50 |
| artist | String | 11700 | 0 | 1722 | Rihanna | 123 | - | - | - | - | - | - |
| song | String | 11700 | 0 | 2964 | Perfect | 23 | - | - | - | - | - | - |
| score | Float | 11700 | 0 | - | - | - | 1366.7 | 710.3 | 343.3 | 1225.7 | 1688.2 | 5524.5 |
| us | Float | 10813 | 887 | 101 | - | - | 20.96 | 17.15 | 1 | 17 | 29 | 100 |
| uk | Float | 6667 | 5033 | 101 | - | - | 29.37 | 25.31 | 1 | 22 | 44 | 100 |
| de | Float | 6149 | 5551 | 101 | - | - | 28.45 | 25.6 | 1 | 21 | 44 | 100 |
| fr | Float | 5062 | 6638 | 201 | - | - | 44.4 | 44.76 | 1 | 29 | 63 | 200 |
| ca | Float | 5708 | 5992 | 101 | - | - | 19.18 | 14.85 | 1 | 16 | 28 | 100 |
| au | Float | 5771 | 5929 | 51 | - | - | 18.16 | 14.03 | 1 | 15 | 29 | 50 |

Data can be loaded through the standard procedure as described below using the JSON format and Pandas framework’s ability to read csv formatted files.



## Deliverables

The main deliverables of this submittals are a single bash script, ETL pipeline, Exploratory Data Analysis (EDA), Preprocessing, Encoding, and Data Manipulation (as needed), and the Song Analytics Data Challenge Pipeline.

The model also provided some alternative solutions as well as more in-depth details on the answers and also expanded on data exploratory/explanatory.

### Single Bash Script

a single bash script named "run.sh" that will allow you to run the entire pipeline:

run.sh can be found in the root directory of the repository. The following script is simply used in the bash script.

### Exploratory Data Analysis (EDA)

The repository contains the following Jupyter notebook(s) for exploratory / explanatory components. Comments are provided in the notebook(s): Song\_Analytics\_EDA.py.

Some explanatory and analytical components are provided which lead to the preprocessing and better understanding the dataset.

Any additional instructions necessary to ensure that the team can reproduce the result or to direct their attention to the right places can be provided upon request for expansion.

### **Song Analytics** Challenge Pipeline

The goal was defined to set up a reproduceable set of extraction, transformation, and loading operations to answers the questions given in the instruction document.

The class of “Song Analytics Data Challenge” from “Song Analytics” containing five generic, eight question-specific, and one plotting functions was prepared. The “Song Analytics Data Challenge” Class, and its underlying functions were imported, and utilized in the Model\_run.py to provide the output for each question.

Five generic functions were defined to conduct the following tasks;

1. Reading and loading the data to the Pandas dataframe: read\_data
2. Showing the unique values of a given feature (column): unique\_value
3. Filtering the dataframe to its desired feature(s): filter\_col
4. Filtering the dataframe to the desired value(s) in each feature(s): filter\_value
5. Aggregating the dataframe through group by and count: agg



In addition to the generic functions, eight question-specific functions, and one plotting function were designed to provide the answers for each question.

Questions and answers are as follows,

Question 1:

a) How many unique artists are represented in the charts?

b) How many unique songs are represented?

Answer 1:

Function q1 is designed to pass as many as arguments desired to the unique\_value function and return the number of unique values for those arguments.

By Passing the name of ‘artist’, and ‘song’ features to function q1 it returns the



There are 1722 artists represented in the charts from January 2000 to June 2019, and there are 2964 unique songs in the charts for the same period.

Question 2:

a) What song has the highest one-month score of all time?

b) What month was this?

c) Who was the artist?

Answer 2:

Function q2 is defined to find the maximum score exist in the dataframe and filter the dataset through using the filter\_value function with the maximum score as the argument. Function q2 returns the required features for the selected song.



The song named ‘Yeah!’ by Usher, Lil’ Jon & Ludacris has the highest one-month score of all time approximately about 5524.5 point. This song was No. one in April 2004.

Question 3:

As I am writing this, I am listening to the song “Safe and Sound” by Capital Cities.

a) How many months was this song on the charts in the United States?

b) What was its peak US chart position?

c) In what month did it achieve this peak US chart position?

Answer 3:

Function q3 receives as many as desired key value pairs (\*\*kwargs) and pass those through the filter\_value function to identify the related datapoints to the given attributes (in this questions song and artist). The specified song/artist is called the favorite song in this function. The function q3 finds the number of months that the favorite song was on chart in US by using to\_list () method of the series of fav\_song[‘month]. Then the function identifies the best position that the favorite song hits in US through utilizing the .idxmin() method and in the end, it finds out what month was the best month for this song that it achieve the peak position in the US chart.

****

‘Safe And Sound’ song by ‘Capital Cities’ was on US chart for 10 month and it secure the 8th position on the US chart as the best position in September 2013.

Question 4:

a) Which artist has spent the most time on the charts as defined by chart-months?

b) How many chart-months have they scored?

Answer 4:

In order to answer to this question, function q4 is designed to aggregate the dataframe on the ‘artist’ feature and count the number of months that the artist has been on the chart. After aggregating the dataset, the function finds the artist with most number of months on chart through using .iloc() and .idxmax() methods. Function q4 returns the name of the artist with the highest number of months on chart and the total number of month he/she stayed on the chart.

Per the definition of the chart\_month from the instruction, a “chart-month” is a row in the data that represents one song spending one month on the charts. If a song is represented on the chart for multiple months, it collects multiple chart\_months as many months as it is stays on the chart. If an artist has multiple songs on the chart, the artist will receive the same number of chart\_months as many songs as they have on the chart.



Rihanna was present on the charts for 123 months as the artist with the most time spent on the charts.

Rihanna’s record is followed by Tylor Swift, Pink, Katy Perry and the Black Eyed Peas. The top five artists with the most time spent on the charts and the number of months they stayed on the charts are shown in Table 3.

TABLE 3 – Top Five Artist Who Spent the most time on the Charts

|  |  |  |
| --- | --- | --- |
| **Rank** | **Artist** | **No. of Month on Charts** |
| 1 | Rihanna | 123 |
| 2 | Taylor Swift | 109 |
| 3 | Pink | 107 |
| 4 | Katy Perry | 93 |
| 5 | The Black Eyed Peas | 87 |

Question 5:

Answer 5:

Missing Values:

In this machine learning projects the corrupted data of parking citations need to be preprocessed in order to be in the ideal format for producing the best performing model.

Some missing values were observed with the following statistics. For handling the missing values, the following items need to be mentioned. Removal, replacement, and imputing are used as follows.

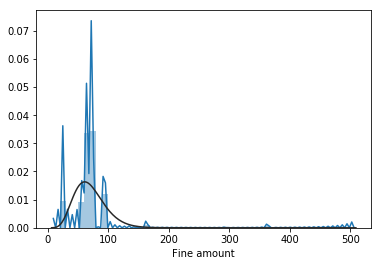
Encoding Categorical Variables:

There are often several transformational steps such as encoding categorical variables, feature scaling and normalization that need to be performed. Scikit-learn has built in functions for most of these commonly used transformations in the preprocessing package.

From sklearn.preprocessing, LabelEncoder is chosen for the ease of use and quick turnaround.

Normalization:

Fine Amount variable seems to fit the Gamma distribution Ranging from $10 to $505 with the mean of $70 and standard deviation of $32.



Normalization was performed on the Fine amount to transforms the feature by scaling each to a given range between Min and Max amount.

This transformation scales and translates the feature individually such that it is in the given range on the training set, e.g. between zero and one.

### Validation

Cross Validation (CV)

To evaluate the performance of the machine learning model, cross validation was performed on data. With that we can say weather our model is Under-fitting/Over-fitting/Well generalized.

Cross validation (CV) is one of the techniques used to test the effectiveness of a machine learning models, it is also a re-sampling procedure used to evaluate a model if we have a limited data.

Based on previous experience, K-fold Cross Validation, the most popular method of cross validation was introduced to avoid over fitting and general biases. The K-fold Cross Validation approach was also checked instead of train-test\_split method. Cross validation avoided over fitting and yielded higher accuracy on average.