Recommendations On Spotify

By Adnan Kpodo | Mentor: Dr. Nan Wang

Abstract

Jupyter Notebook is a web-based interactive development environment for notebooks, code, and data and has been running since 2014. It combines live code, equations, narrative text, visualizations, interactive dashboards and other media.

Kaggle is a subsidiary of Google LLC and functions as an online community of data scientists and machine learning practitioners. It contains many dataset for available use for people like me.

Introduction

- Using a dataset called “Spotify Analysis” I am trying to find a correlation between certain columns that will lead to a preferable result
- That result being the likeliness that the listener will like a song based on its attributes
- Attributes being, acousticness, danceability, duration_ms, energy, key, liveness, loudness, and mode.
- Mode being whether they liked the song or not and is represented by 0 and 1, 1 being they liked the song and 0 being they disliked the song

Methodology

- Using python libraries like Pandas, Numpy, Seaborn and etc. I am able to manipulate data in many different ways
- And by using Algorithms like KNN, Random Forest Classifier, Logistic Regression etc. we can test how accurate the algorithms are in predicting if someone liked a song or not
- So based off of this, the idea was to pair different columns up together to see if they shared anything in common

Pulling Data From A DataSet

- By using specific lines of code I am able to pull information from the dataset on any of the columns that I choose
- With these 3 examples I am selecting a name from the “artist” column and requesting info from the “song_title” column which displays the artist songs in the given dataset

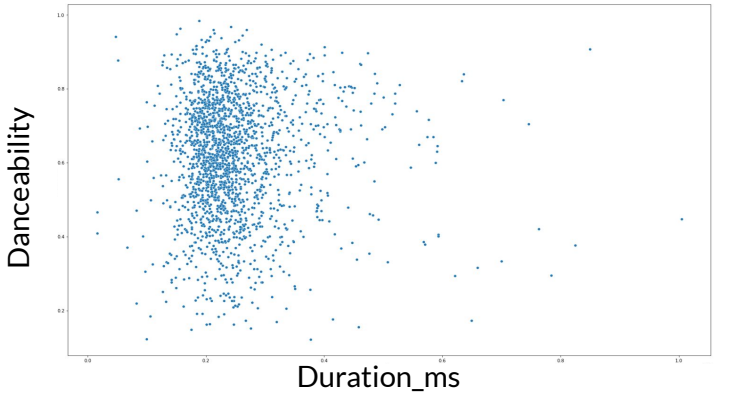
```
In [11]: df[df['artist']== 'Kendrick Lamar']['song_title']

Out[11]: 157                                    Alright
         421     Swimming Pools (Drank) - Extended Version
         526                                 Money Trees
         662                                m.A.A.d city
         1162                                    HUMBLE.
         1226    Swimming Pools (Drank) - Extended Version
         Name: song_title, dtype: object
```

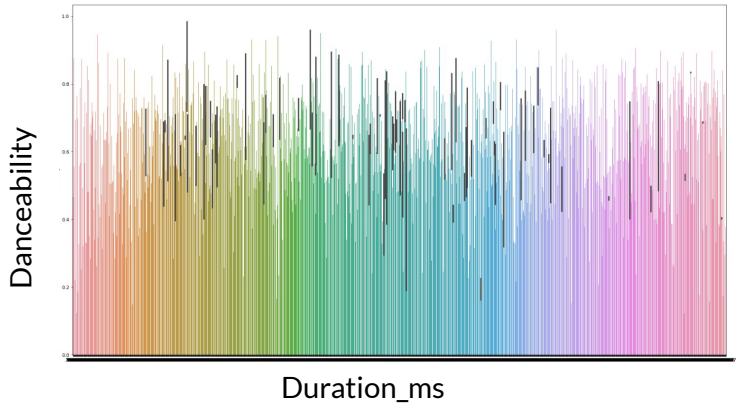
```
In [12]: df[df['artist']== 'Drake']['song_title']

Out[12]: 5                          Sneakin’
         6                        Childs Play
         37                         Jumpman
         117                   Hotline Bling
         154                       Big Rings
         158                   Know Yourself
         168                  Wu-Tang Forever
         440                       Headlines
         589       Started From the Bottom
         676     Hold On, We're Going Home
         717     Hold On, We're Going Home
         779       0 To 100 / The Catch Up
         1016              Skepta Interlude
         1183                  Passionfruit
         1218               Best I Ever Had
         1223                    Take Care
         Name: song_title, dtype: object
```

```
In [10]: df[df['artist']== 'Rain Man']['song_title']

Out[10]: 2014    Habit - Dack Janiels & Wenzday Remix
         Name: song_title, dtype: object
```

Using Graphs To Find Relations

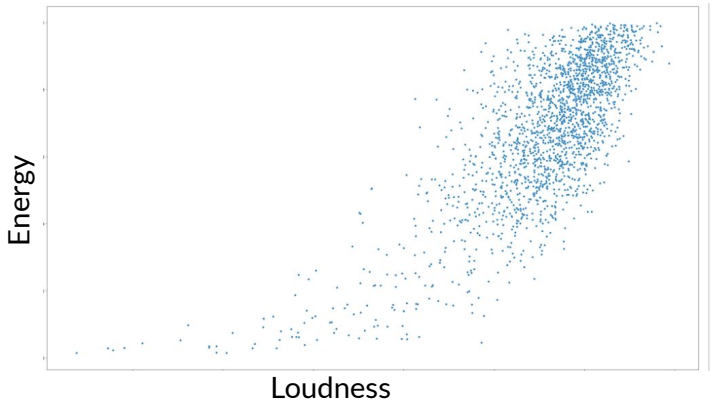
- By making Graphs based on my dataset on “Spotify Analysis” I am able to help better determine based on the listener what song should be recommended to them. Increasing the probability that they’ll like it.
- An example would be these two graphs that show the relation ‘between the duration of songs’(x) and ‘the danceability’(y)
- The first graph being scatter plot is shown to be more reliable than the barplot in this case as the dataset seems to be too large to be readable using a barplot



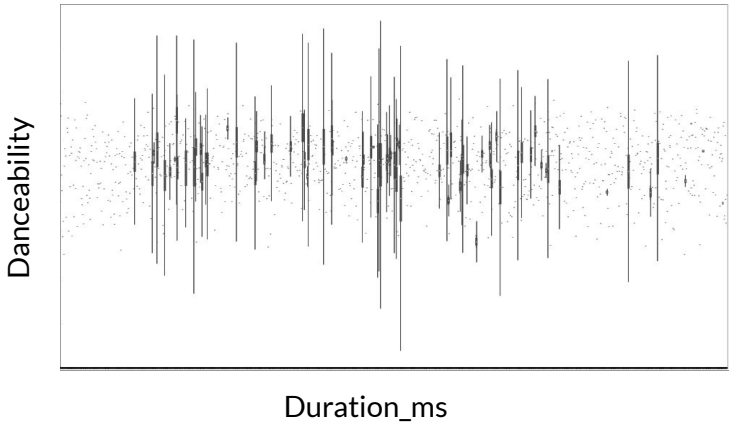


More Graphs

Scatterplot

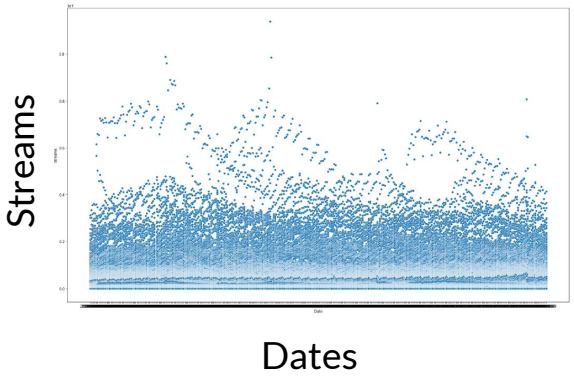


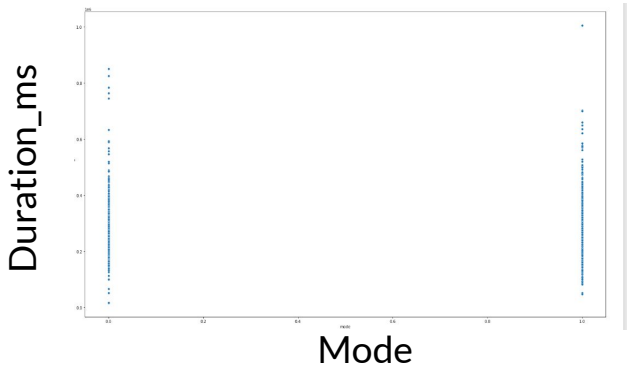
Violinplot



Defective/Unusable Graphs

- Both of these graphs have proven to be unusable as the the one on the left hand side has too much data to even comprehend what is going on
- And the one on the right hand side column and row values do not work well together rendering it defective





Algorithm Accuracy

- Using predictive algorithms I am able to decide which method best comes close to the actual number of likes and dislike to a song.

1. **Logistic Regression** was 56% accurate with dislikes and 63% accurate with likes
2. **Random Forest Classifier** was 58% accurate with dislikes and 66% accurate with likes
3. **Supporting Vector Machine** had 0% accuracy with dislikes and 60% accuracy with likes
4. **KNeighborsClassifier** was 45% accurate with dislikes and 62 percent accurate with likes

- In conclusion **Random Forest Classifier** is the most accurate

Conclusion

- Using jupyter and the many libraries at my disposal I was able to find many connections between the different columns to come to a conclusion regarding patterns
- Using those patterns I am able to narrow down the chances that a user will like a specific song
- Without a doubt the actual recommendation process is far more complex, but the way I manipulated data helps to give us a gist of what the process is like
- Future Work: an addition I would make in the future is adding a column called Artist. With that in mind I could more accurately recommend songs based on the artist the user most frequently listens to