Spotify songs Analysis

Bhartendu Dubey

IBM Advanced Data Science Capstone

Contents:

- Data Set
- Use Case
- Data Quality Assessment
- Data Exploration (e.g. correlation between columns)
- Data Visualization (e.g. value distribution of columns)
- Feature Engineering (e.g. imputing missing values)
- Machine Learning Algorithms implemented
- Model Performance Indicator (e.g. F1 score)
- Performance Comparison

Dataset

The dataset used was obtained from Kaggle (https://www.kaggle.com/edalrami/19000-spotify-songs)

The Dataset contains 19,000 songs and had 15 features(as shown below).

	song_name	song_popularity	song_duration_ms	acousticness	danceability	energy	instrumentalness	key	liveness	loudness	audio_mode	speechiness
0	Boulevard of Broken Dreams	73	262333	0.005520	0.496	0.682	0.000029	8	0.0589	-4.095	1	0.0294
1	In The End	66	216933	0.010300	0.542	0.853	0.000000	3	0.1080	-6.407	0	0.0498
2	Seven Nation Army	76	231733	0.008170	0.737	0.463	0.447000	0	0.2550	-7.828	1	0.0792
3	By The Way	74	216933	0.026400	0.451	0.970	0.003550	0	0.1020	-4.938	1	0.1070
4	How You Remind Me	56	223826	0.000954	0.447	0.766	0.000000	10	0.1130	-5.065	1	0.0313

Dataset features:

```
song_name
song_popularity
song_duration_ms
acousticness
danceability
energy
instrumentalness
key
liveness
loudness
audio_mode
speechiness
tempo
time_signature
audio_valence
```

Use Case

To understand the dataset and also perform necessary processing in order to prepare the dataset for training on ML models so that one can predict popularity of various songs present in the dataset.

Identify useful insights.

Performance comparison.

Data Quality Assessment

```
song_data.info() # provides a concise summary of DataFrame
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 18835 entries, 0 to 18834
Data columns (total 15 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   song_name         18835 non-null  object
 1   song_popularity   18835 non-null  int64
 2   song_duration_ms  18835 non-null  int64
 3   acousticness      18835 non-null  float64
 4   danceability      18835 non-null  float64
 5   energy            18835 non-null  float64
 6   instrumentalness  18835 non-null  float64
 7   key               18835 non-null  int64
 8   liveness          18835 non-null  float64
 9   loudness          18835 non-null  float64
 10  audio_mode        18835 non-null  int64
 11  speechiness       18835 non-null  float64
 12  tempo             18835 non-null  float64
 13  time_signature    18835 non-null  int64
 14  audio_valence     18835 non-null  float64
dtypes: float64(9), int64(5), object(1)
memory usage: 2.2+ MB
```

Data Cleaning

```
#Let's have a look at null values in dataset

song_data.columns[song_data.isnull().any()]
```

```
Index([], dtype='object')
```

```
song_data.isnull().sum()
```

```
song_name           0
song_popularity     0
song_duration_ms    0
acousticness        0
danceability        0
energy              0
instrumentalness    0
key                 0
liveness            0
loudness            0
audio_mode          0
speechiness         0
tempo               0
time_signature      0
audio_valence       0
dtype: int64
```

Feature Engineering

Outlier detection was done.

Specifying threshold value based on the mean value of popularity of songs.

```
song_data.describe() #gives statistical details (like percentile, mean, std etc.) of data frame
```

	song_popularity	song_duration_ms	acousticness	danceability	energy	instrumentalness	key	liveness	loudness	audio_mode
count	18835.000000	1.883500e+04	18835.000000	18835.000000	18835.000000	18835.000000	18835.000000	18835.000000	18835.000000	18835.000000
mean	52.991877	2.182116e+05	0.258539	0.633348	0.644995	0.078008	5.289196	0.179650	-7.447435	0.628139
std	21.905654	5.988754e+04	0.288719	0.156723	0.214101	0.221591	3.614595	0.143984	3.827831	0.483314
min	0.000000	1.200000e+04	0.000001	0.000000	0.001070	0.000000	0.000000	0.010900	-38.768000	0.000000
25%	40.000000	1.843395e+05	0.024100	0.533000	0.510000	0.000000	2.000000	0.092900	-9.044000	0.000000
50%	56.000000	2.113060e+05	0.132000	0.645000	0.674000	0.000011	5.000000	0.122000	-6.555000	1.000000
75%	69.000000	2.428440e+05	0.424000	0.748000	0.815000	0.002570	8.000000	0.221000	-4.908000	1.000000
max	100.000000	1.799346e+06	0.996000	0.987000	0.999000	0.997000	11.000000	0.986000	1.585000	1.000000

Feature Engineering

Outlier detection was done.

Specifying threshold value based on the mean value of popularity of songs.

Feature Engineering

```
#Specifying threshold value based on the mean value of popularity of songs

spotify_song_data["popularity"]= [ 1 if i>=52.99 else 0 for i in spotify_song_data.song_popularity ]
spotify_song_data["popularity"].value_counts()
```

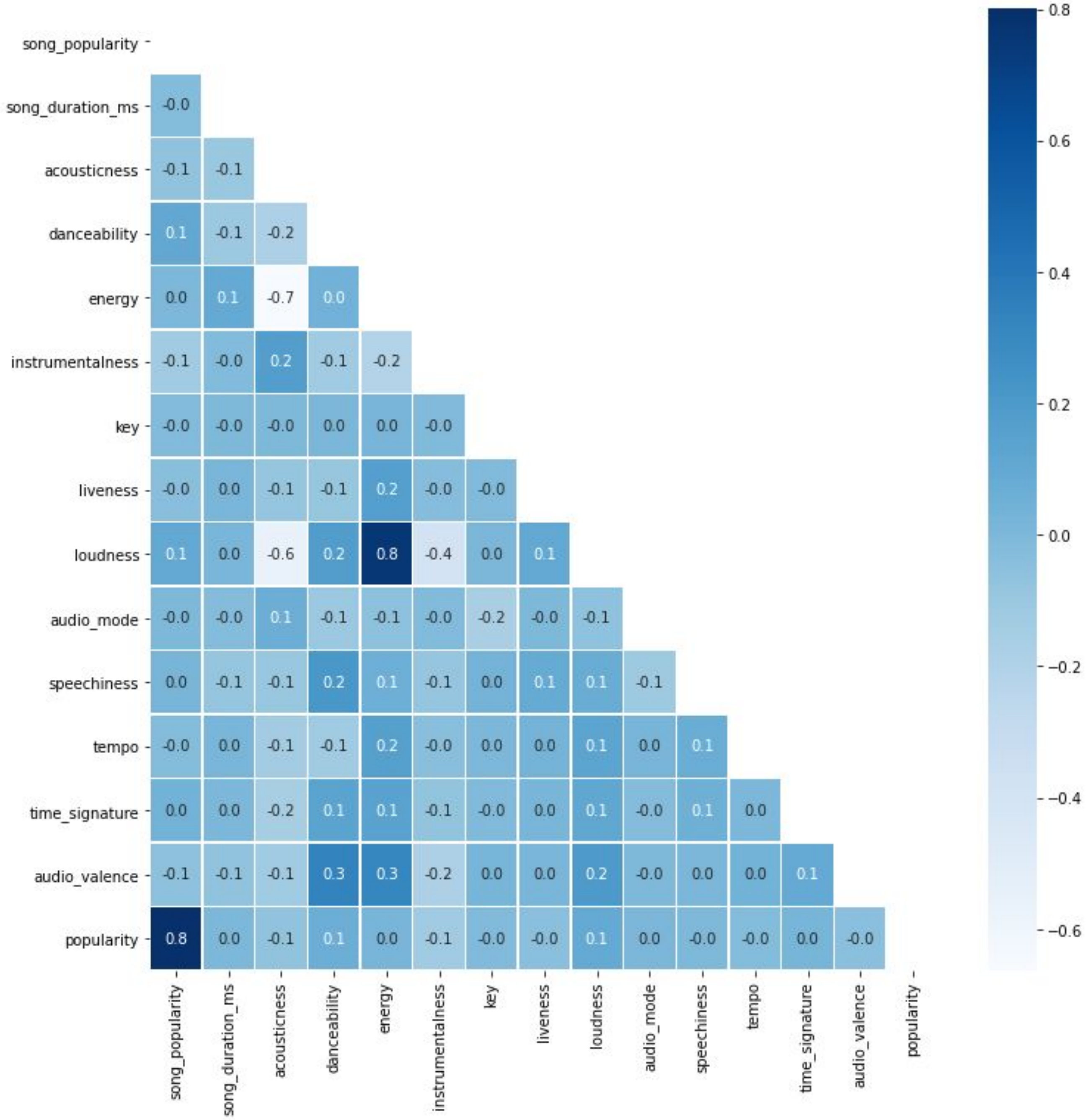
```
1    10516
0     8319
Name: popularity, dtype: int64
```

Data Exploration and Visualization

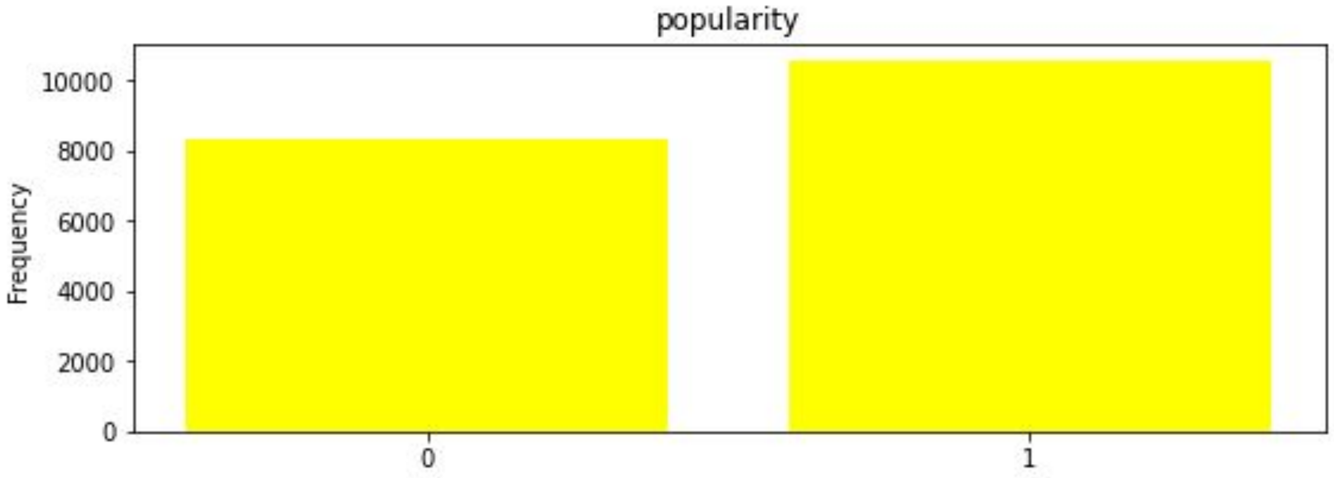
Correlation

The plot for correlation tells that:

- Correlation between loudness and energy is 0.8 (strong)
- All the other correlations are quite low.
- When correlation between song_popularity and all other features is compared, there are no signs of a strong correlation (a linear relationship) that gives us a clear information about popularity.
- danceability and loudness seems to have correlation with popularity feature(0.10).

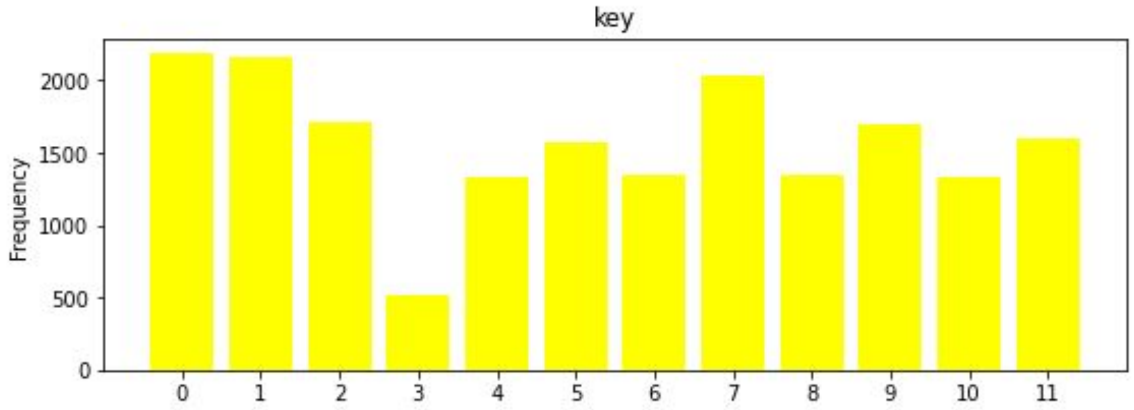


Exploration and Visualization



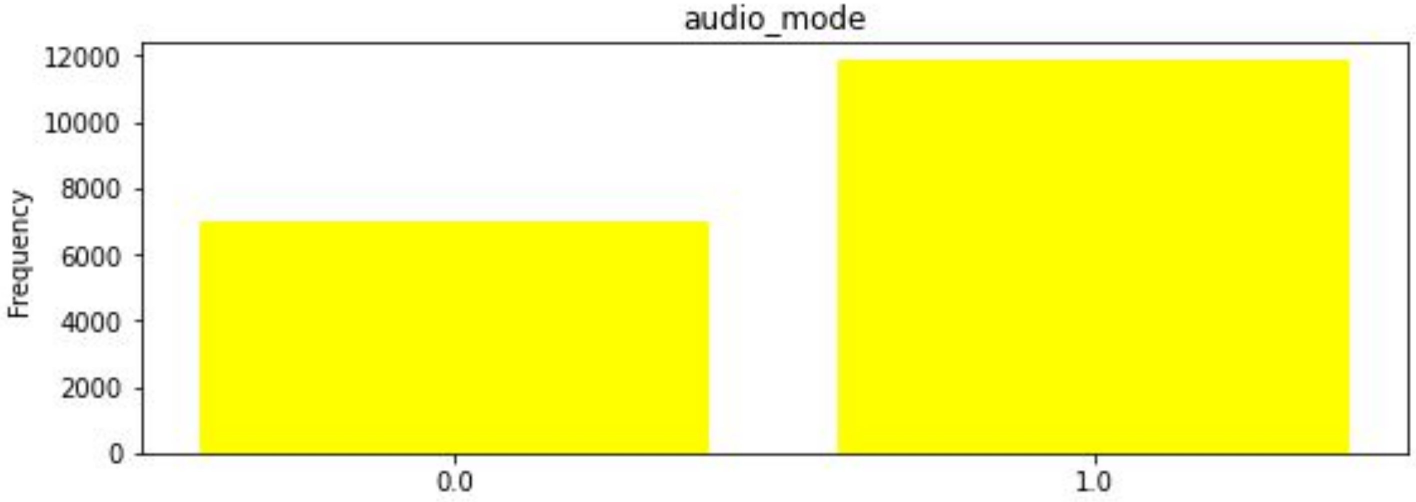

```
popularity:
1    10516
0     8319
Name: popularity, dtype: int64
```

Exploration and Visualization



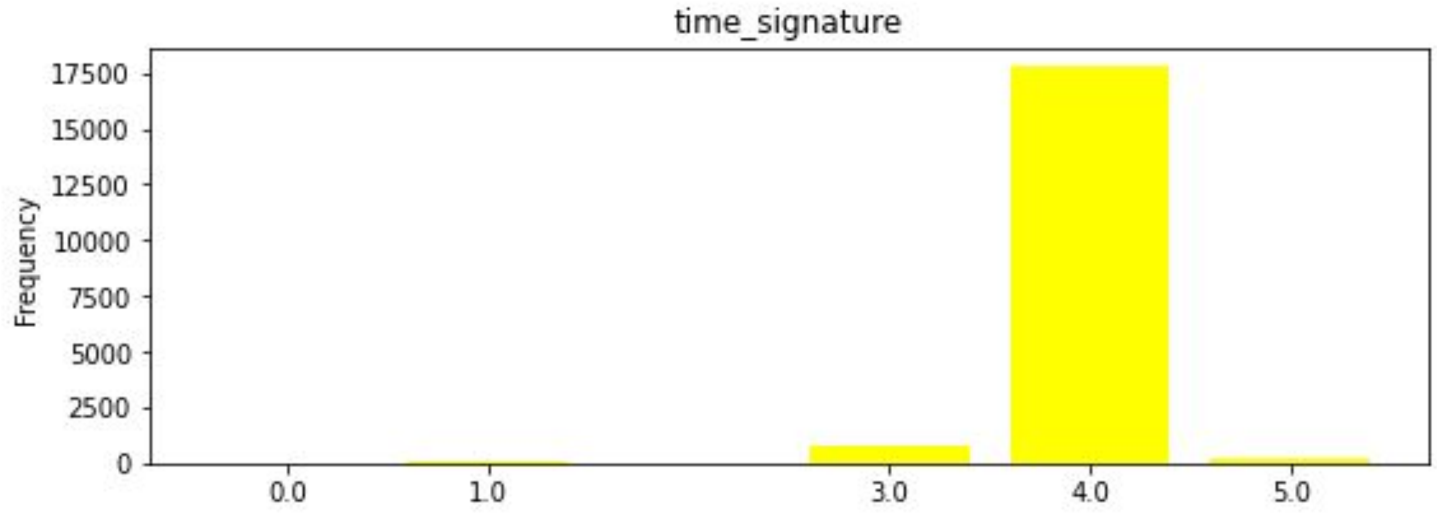

```
key:
0      2182
1      2164
7      2032
2      1715
9      1698
11     1600
5      1574
6      1351
8      1349
10     1331
4      1327
3       512
Name: key, dtype: int64
```

Exploration and Visualization



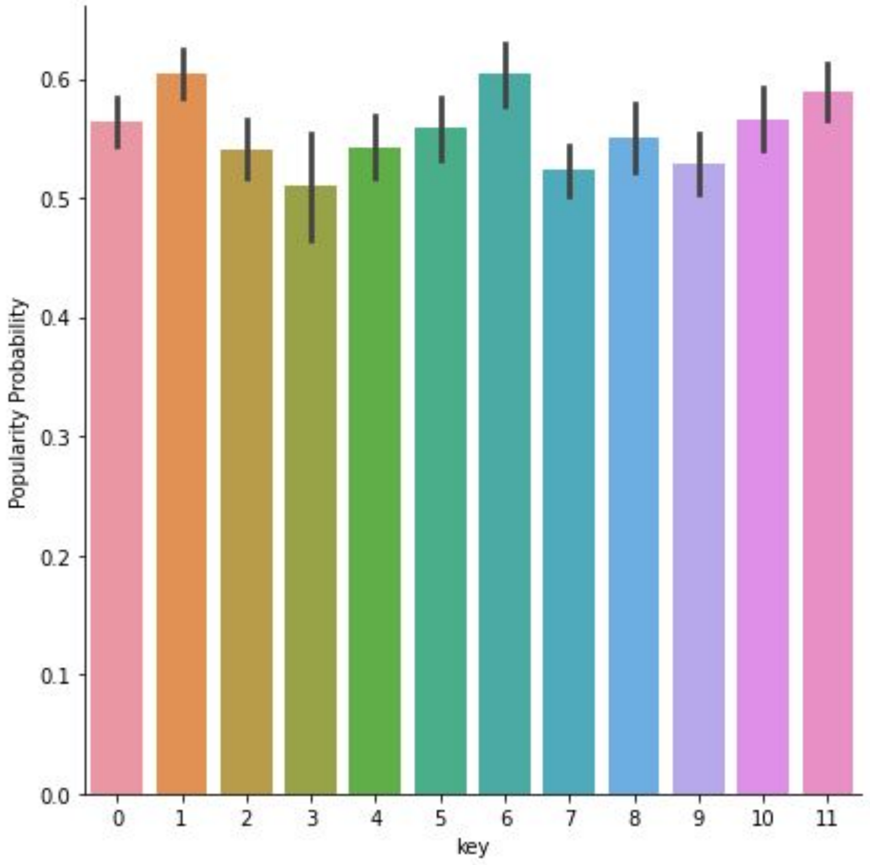

```
audio_mode:
1.0    11831
0.0     7004
Name: audio_mode, dtype: int64
```

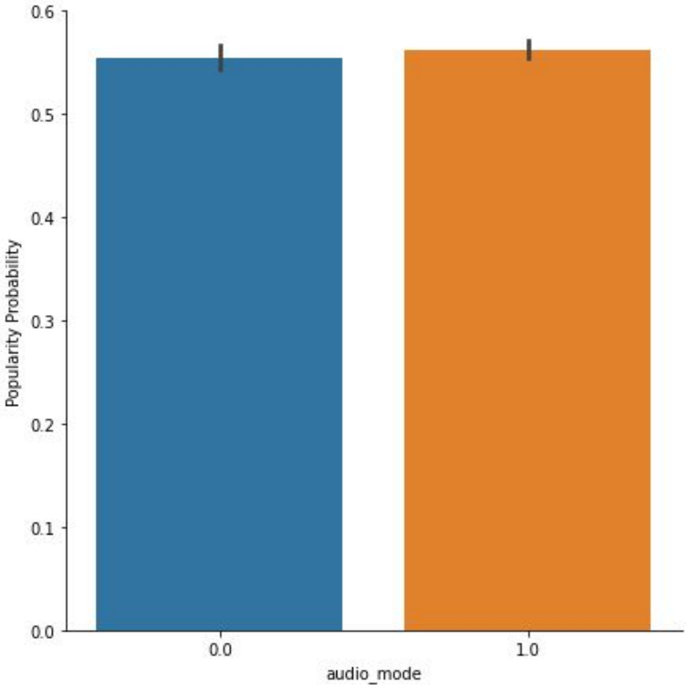
Exploration and Visualization

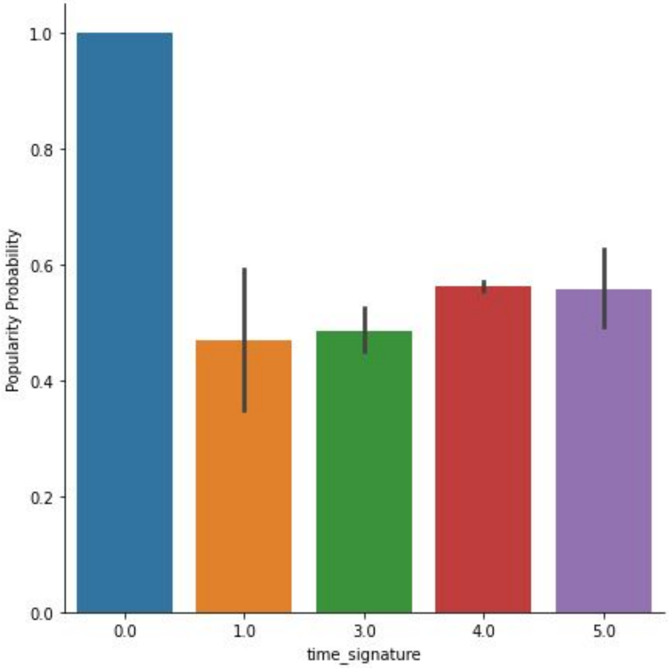



```
time_signature:
4.0    17754
3.0      772
5.0      233
1.0       73
0.0        3
Name: time_signature, dtype: int64
```

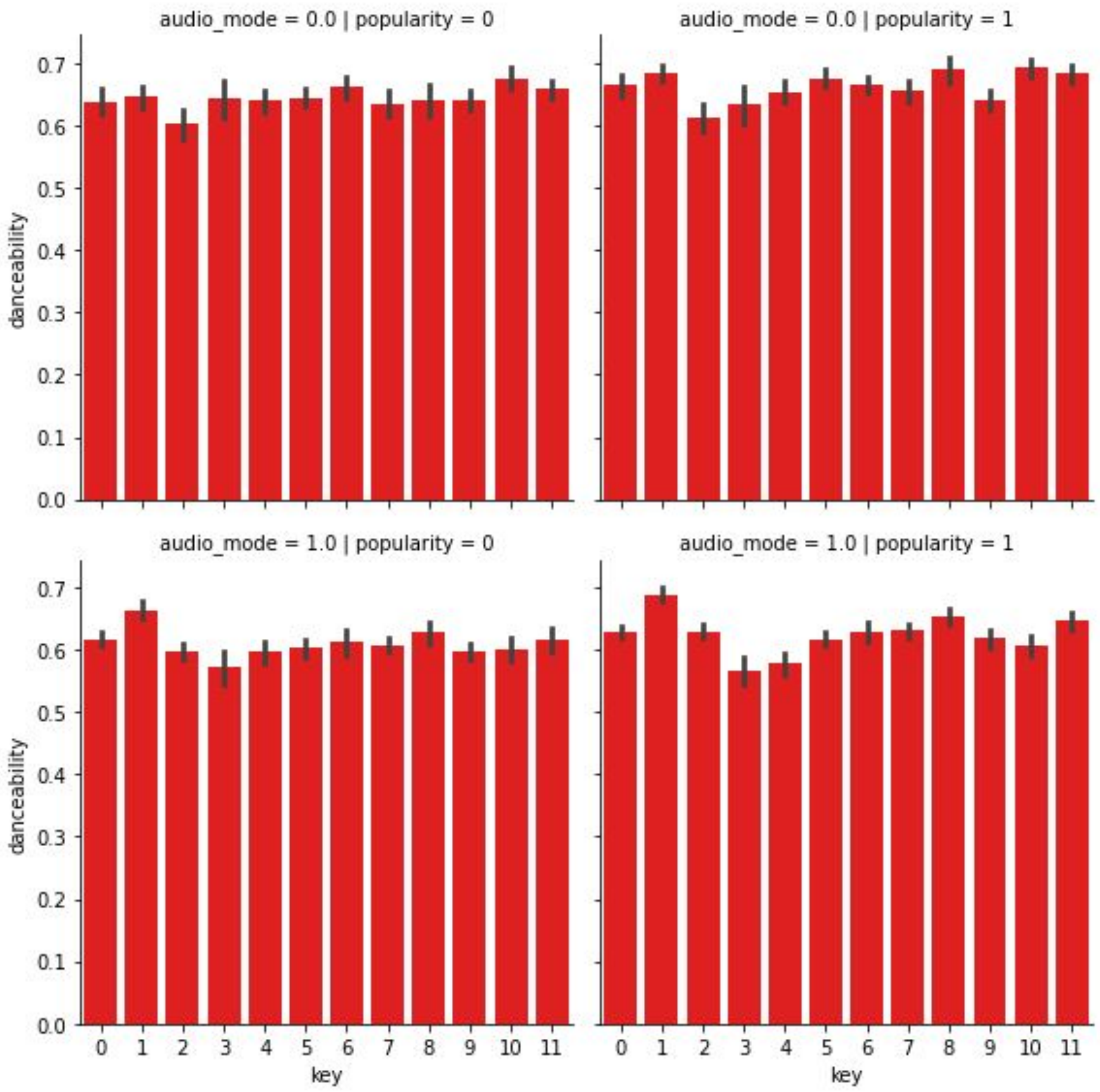
Exploration and Visualization



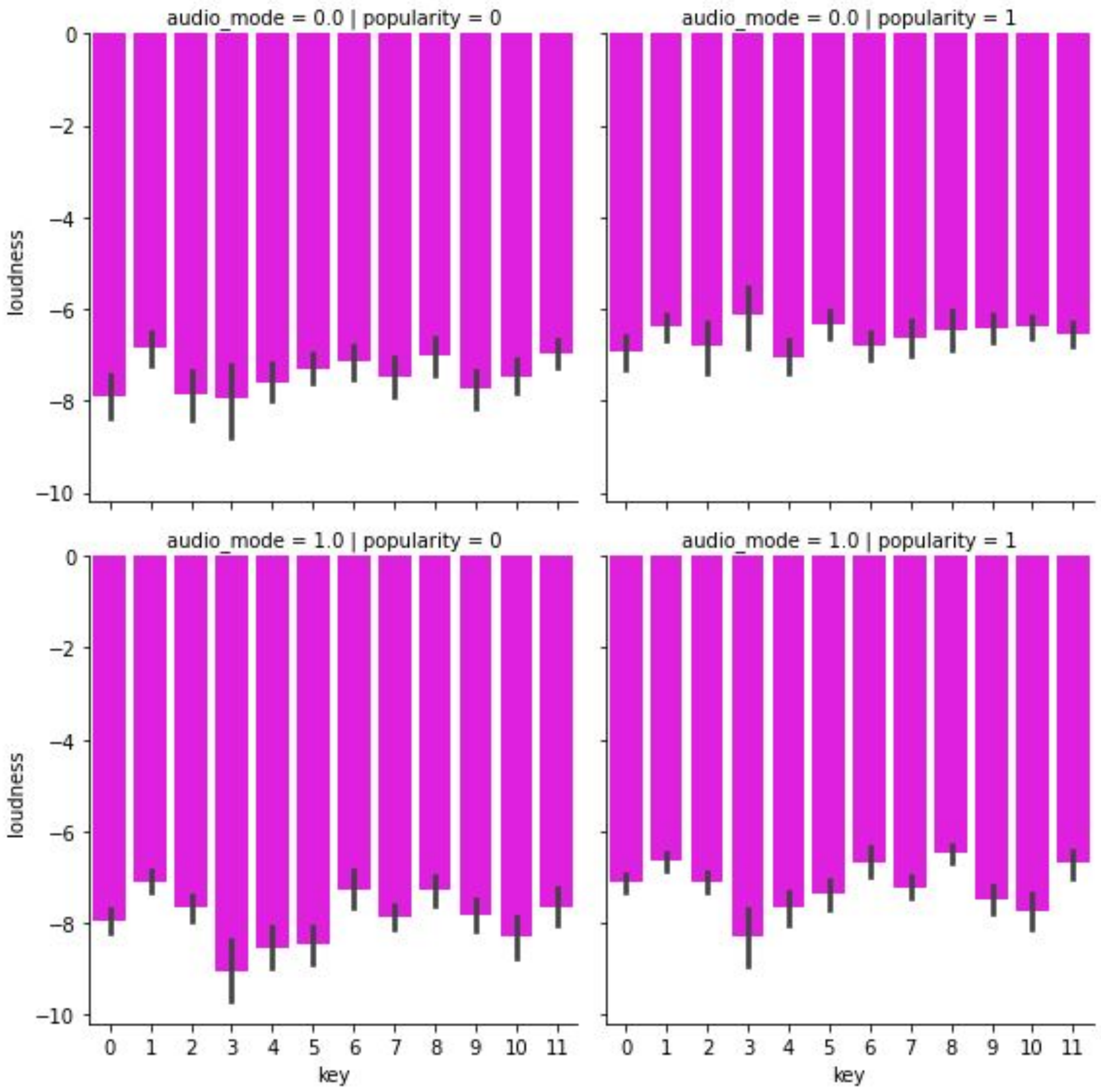




Factor plots

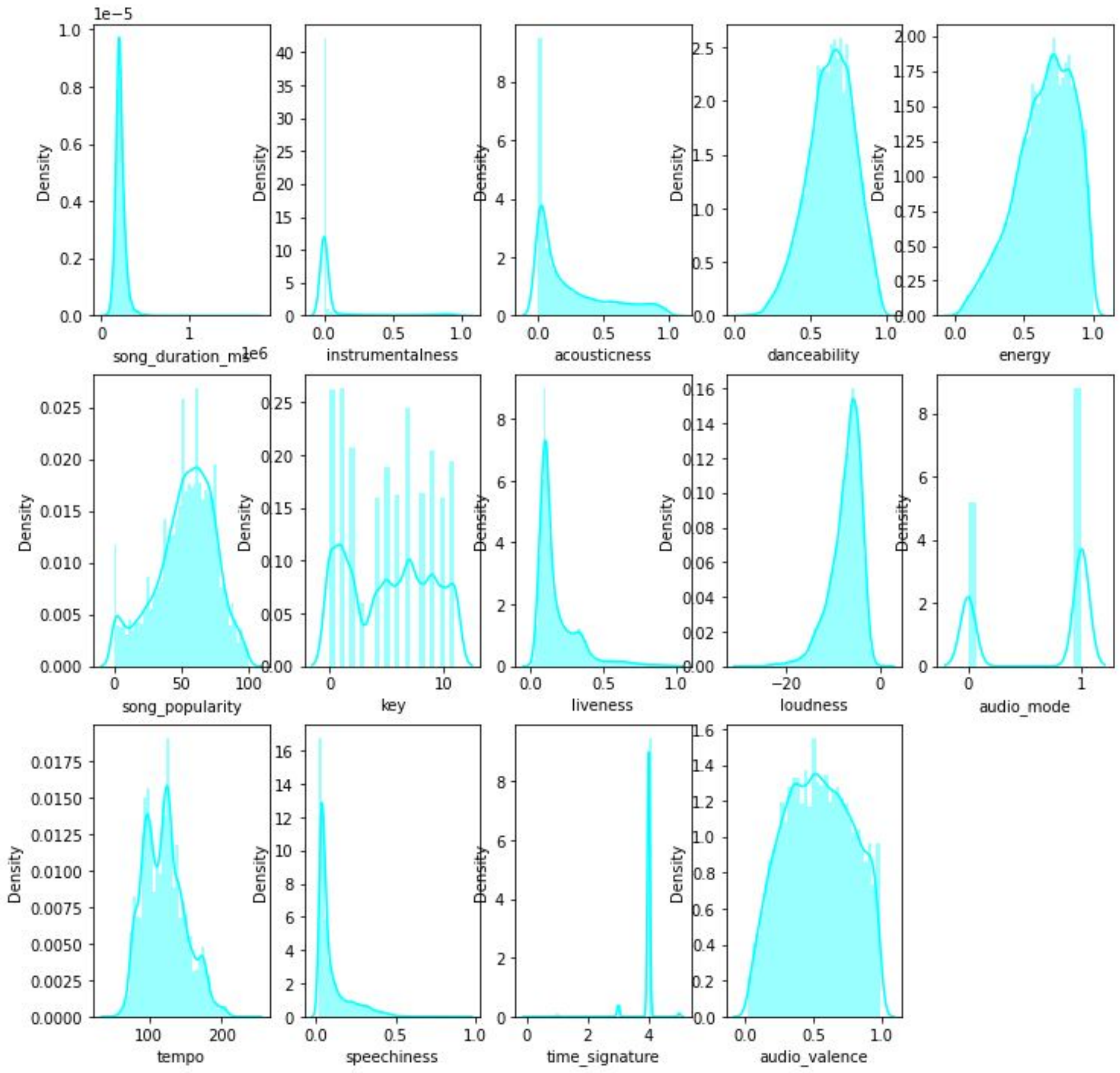


audio_mode = 0.0 | popularity = 0
audio_mode = 0.0 | popularity = 1
audio_mode = 1.0 | popularity = 0
audio_mode = 1.0 | popularity = 1
danceability
danceability
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0 1 2 3 4 5 6 7 8 9 10 11
key
key



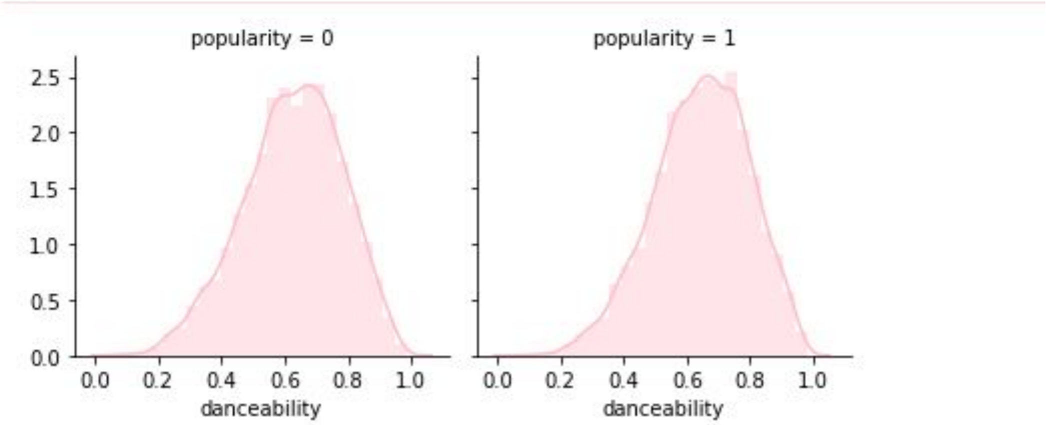
audio_mode = 0.0 | popularity = 0
audio_mode = 0.0 | popularity = 1
audio_mode = 1.0 | popularity = 0
audio_mode = 1.0 | popularity = 1
loudness
key
0
−2
−4
−6
−8
−10
0 1 2 3 4 5 6 7 8 9 10 11

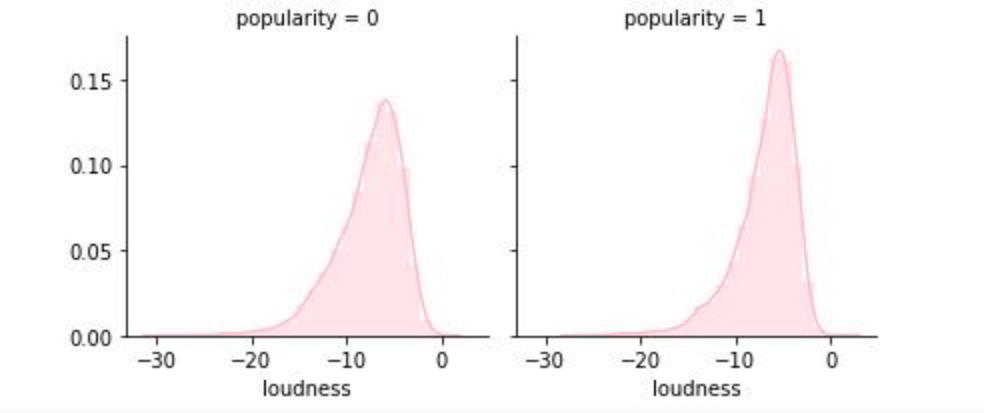
Distplots



song_duration_ms
instrumentalness
acousticness
danceability
energy
song_popularity
key
liveness
loudness
audio_mode
tempo
speechiness
time_signature
audio_valence
Density

Exploration and Visualization



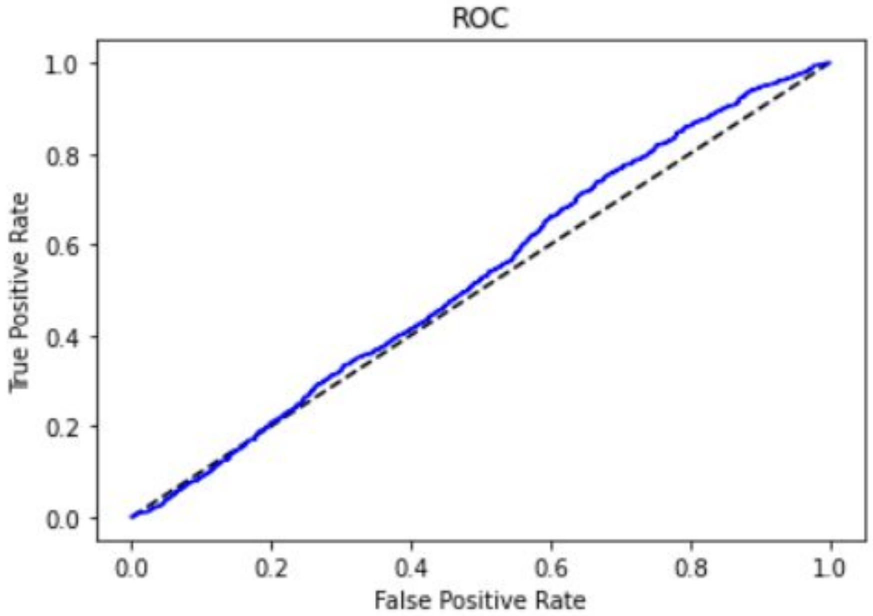


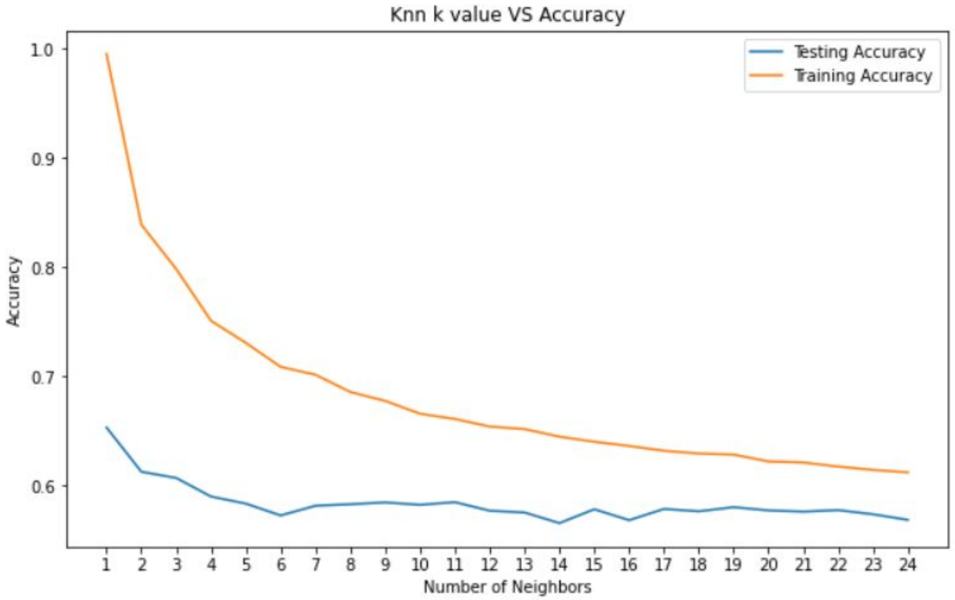
ML Algorithms Implemented:

Logistic Regression

K-Nearest Neighbors

Support Vector Machine






```
Best accuracy is 0.653160453808752 with K = 1
```

Results

Model Performance Indicator: accuracy score, ROC curve

	Model	Accuracy
1	K-NearestNeighbors	0.653160
2	LogisticRegession	0.560508
0	SVM	0.559157

Hence, out of the all applied ML algorithms K-Nearest Neighbors(KNN) gave the best results for predicting popularity (Popular/Unpopular) of a song with an accuracy of 65.31%

Conclusion

Dataset was explored and visualized

Feature engineering

Correlation of various features was found

Performance comparison of ML Models.

THANK YOU!