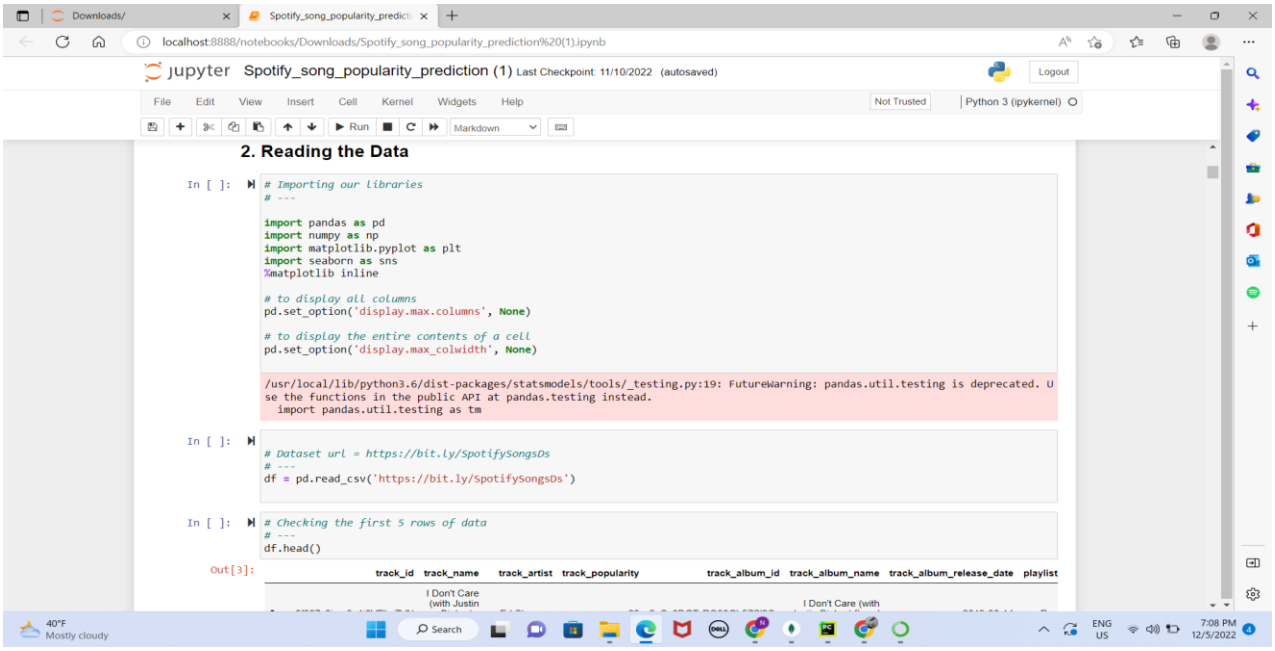
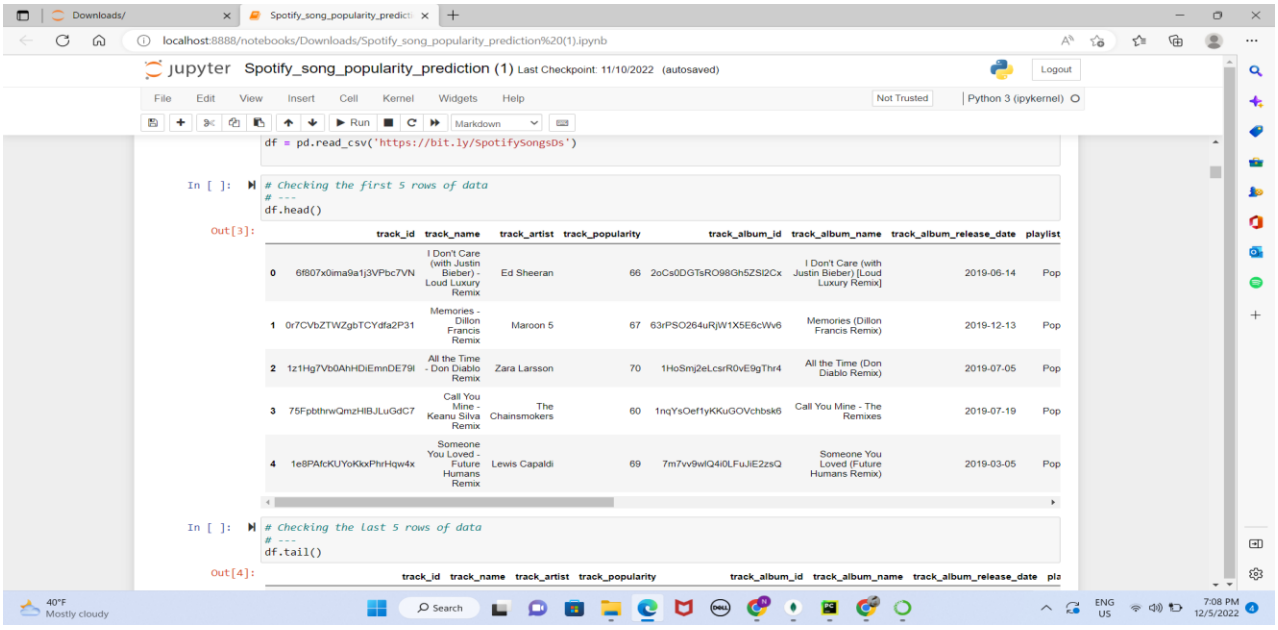
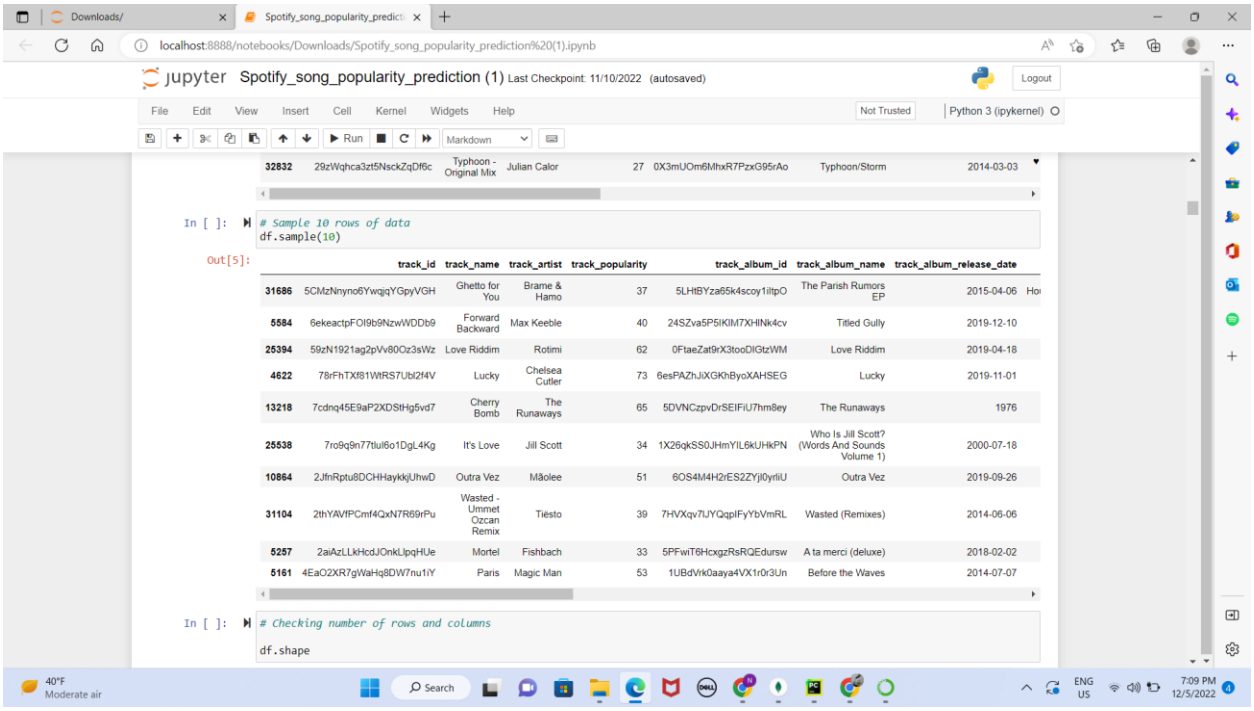
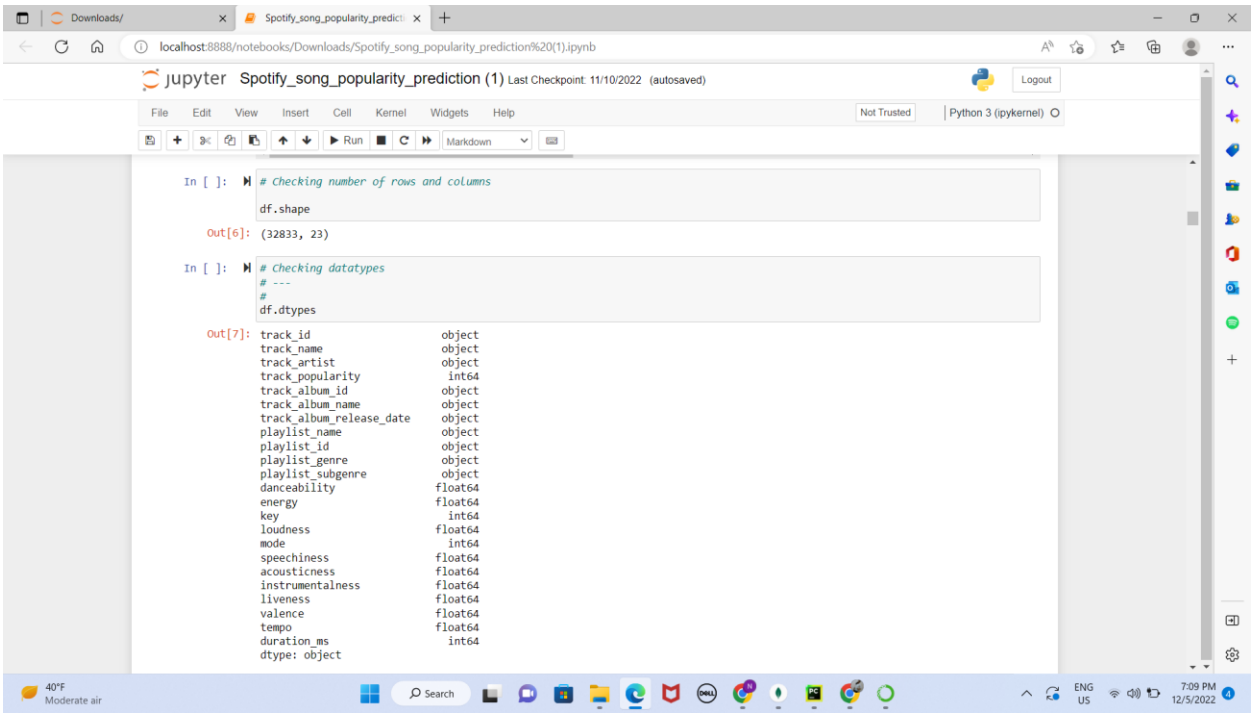
MACHINE LEARNING - SCREENSHOTS OF RESULTS

Student ID: 700734361
Student Name: Kiran Kumar Kongari
Subject Code: CS5710
CRN: 12675

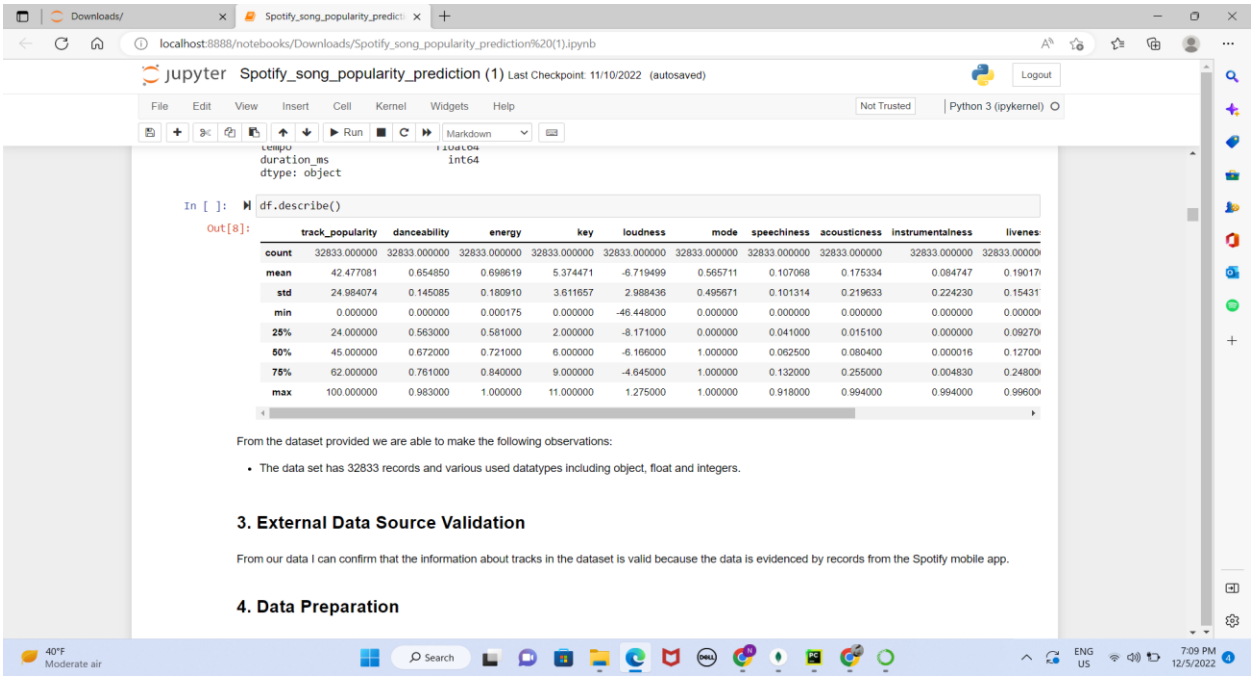





Downloads/
Spotify_song_popularity_predicti
localhost:8888/notebooks/Downloads/Spotify_song_popularity_prediction%20(1).ipynb
jupyter Spotify_song_popularity_prediction (1) Last Checkpoint: 11/10/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Markdown
32832 29zWqhca3zt5NsckZqDf6c Typhoon - Original Mix Julian Calor 27 0X3mUOm6MhxR7PzxG95rAo Typhoon/Storm 2014-03-03
In [ ]:
# Sample 10 rows of data
df.sample(10)
Out[5]:
track_id track_name track_artist track_popularity track_album_id track_album_name track_album_release_date
31686 5CMzNnyno6YwqjqYGpyVGH Ghetto for You Brame & Hamo 37 5LHtBYza65k4scoy1iltpO The Parish Rumors EP 2015-04-06
5584 6ekeactpFOI9b9NzwWDDb9 Forward Backward Max Keeble 40 24SZva5P5IKIM7XHINk4cv Titled Gully 2019-12-10
25394 59zN1921ag2pVv80Oz3sWz Love Riddim Rotimi 62 0FtaeZat9rX3tooDlGtzWM Love Riddim 2019-04-18
4622 78rFhTXf81WtRS7Ubl2f4V Lucky Chelsea Cutler 73 6esPAZhJiXGKhByoXAHSEG Lucky 2019-11-01
13218 7cdnq45E9aP2XDStHg5vd7 Cherry Bomb The Runaways 65 5DVNCzpvDrSEIFiU7hm8ey The Runaways 1976
25538 7ro9q9n77tlul6o1DgL4Kg It's Love Jill Scott 34 1X26qkSS0JHmYlL6kUHkPN Who Is Jill Scott? (Words And Sounds Volume 1) 2000-07-18
10864 2JfnRptu8DCHHaykkjUhwD Outra Vez Mãolee 51 6OS4M4H2rES2ZYjl0yrliU Outra Vez 2019-09-26
31104 2thYAVfPCmf4QxN7R69rPu Wasted - Ummet Ozcan Remix Tiësto 39 7HVXqv7lJYQqpIFyYbVmRL Wasted (Remixes) 2014-06-06
5257 2aiAzLLkHcdJOnkLlpqHUe Mortel Fishbach 33 5PFwiT6HcxgzRsRQEdursw A ta merci (deluxe) 2018-02-02
5161 4EaO2XR7gWaHq8DW7nu1iY Paris Magic Man 53 1UBdVrk0aaya4VX1r0r3Un Before the Waves 2014-07-07
In [ ]:
Checking number of rows and columns
df.shape
40°F Moderate air
Search
ENG US
7:09 PM 12/5/2022


Downloads/
Spotify_song_popularity_predicti
localhost:8888/notebooks/Downloads/Spotify_song_popularity_prediction%20(1).ipynb
jupyter Spotify_song_popularity_prediction (1) Last Checkpoint: 11/10/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Markdown
In [ ]:
Checking number of rows and columns
df.shape
Out[6]: (32833, 23)
In [ ]:
Checking datatypes

#
df.dtypes
Out[7]:
track_id object
track_name object
track_artist object
track_popularity int64
track_album_id object
track_album_name object
track_album_release_date object
playlist_name object
playlist_id object
playlist_genre object
playlist_subgenre object
danceability float64
energy float64
key int64
loudness float64
mode int64
speechiness float64
acousticness float64
instrumentalness float64
liveness float64
valence float64
tempo float64
duration_ms int64
dtype: object
40°F Moderate air
Search
ENG US
7:09 PM 12/5/2022



```
duration_ms                int64
dtype: object
```

```
In [ ]: df.describe()
```

Out[8]:

	track_popularity	danceability	energy	key	loudness	mode	speechiness	acousticness	instrumentalness	liveness
count	32833.000000	32833.000000	32833.000000	32833.000000	32833.000000	32833.000000	32833.000000	32833.000000	32833.000000	32833.00000
mean	42.477081	0.654850	0.698619	5.374471	-6.719499	0.565711	0.107068	0.175334	0.084747	0.19017
std	24.984074	0.145085	0.180910	3.611657	2.988436	0.495671	0.101314	0.219633	0.224230	0.15431
min	0.000000	0.000000	0.000175	0.000000	-46.448000	0.000000	0.000000	0.000000	0.000000	0.00000
25%	24.000000	0.563000	0.581000	2.000000	-8.171000	0.000000	0.041000	0.015100	0.000000	0.09270
50%	45.000000	0.672000	0.721000	6.000000	-6.166000	1.000000	0.062500	0.080400	0.000016	0.12700
75%	62.000000	0.761000	0.840000	9.000000	-4.645000	1.000000	0.132000	0.255000	0.004830	0.24800
max	100.000000	0.983000	1.000000	11.000000	1.275000	1.000000	0.918000	0.994000	0.994000	0.99600

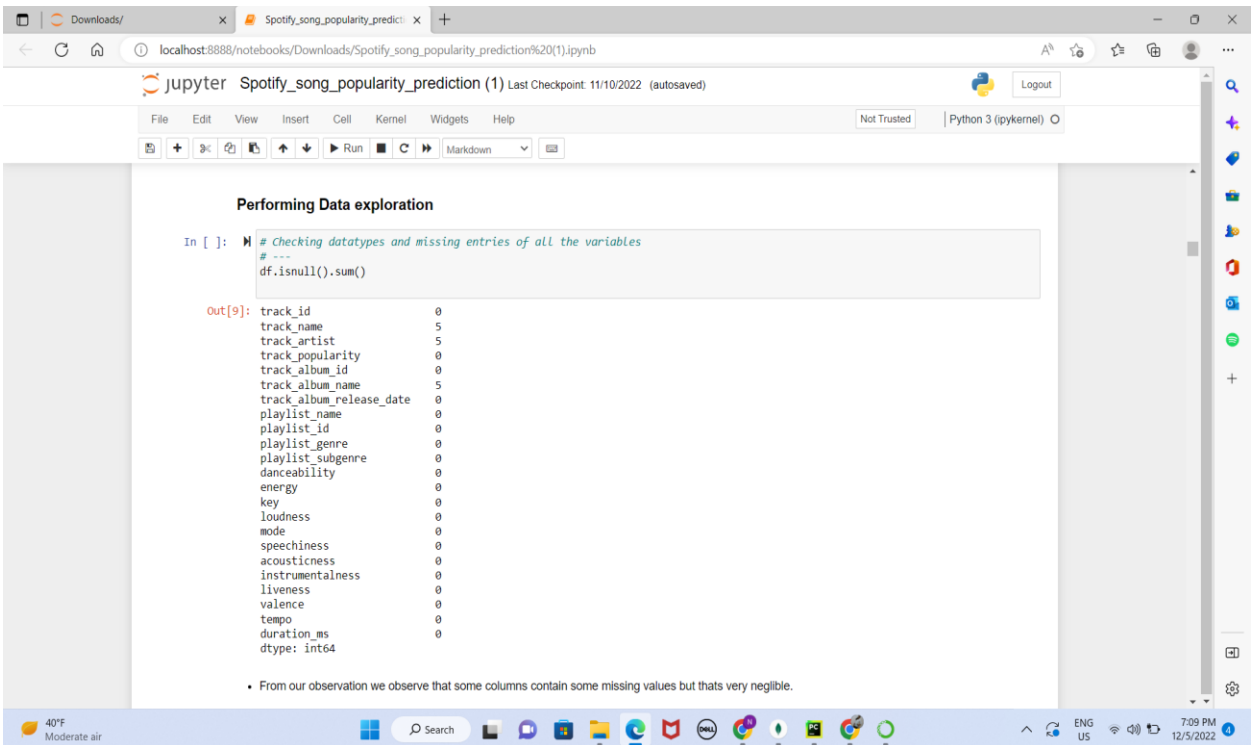
From the dataset provided we are able to make the following observations:

- The data set has 32833 records and various used datatypes including object, float and integers.

3. External Data Source Validation

From our data I can confirm that the information about tracks in the dataset is valid because the data is evidenced by records from the Spotify mobile app.

4. Data Preparation

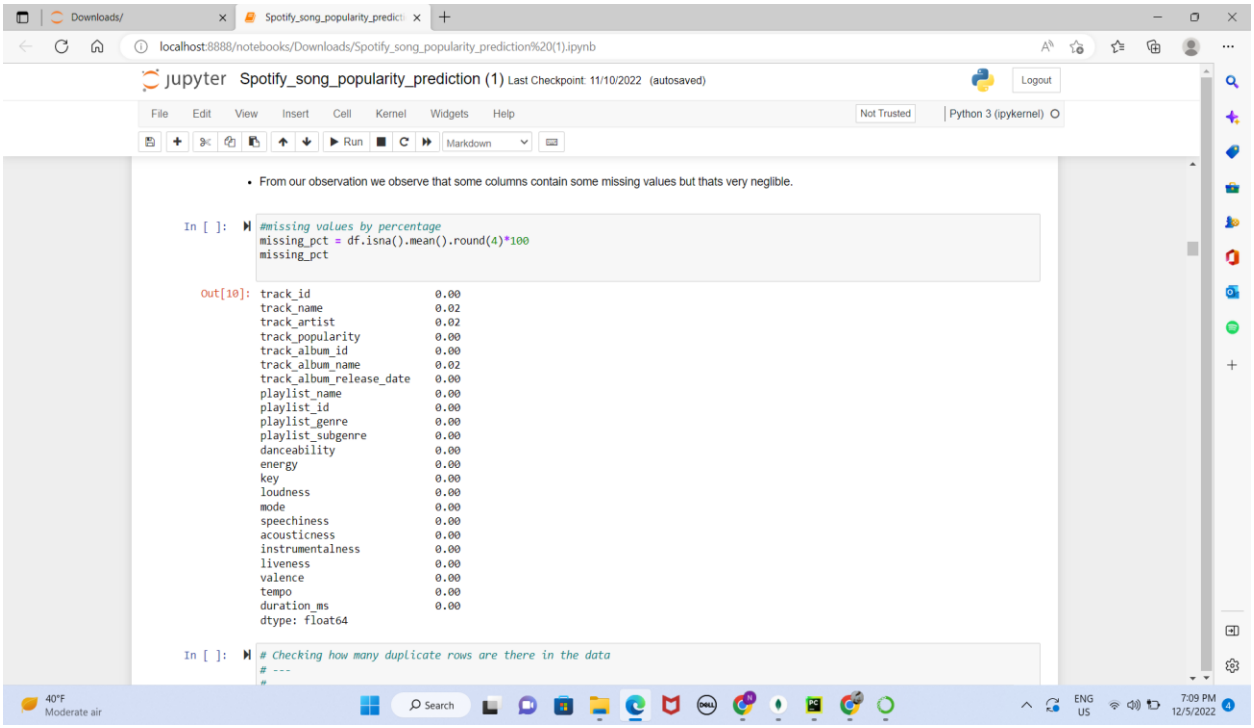


Performing Data exploration

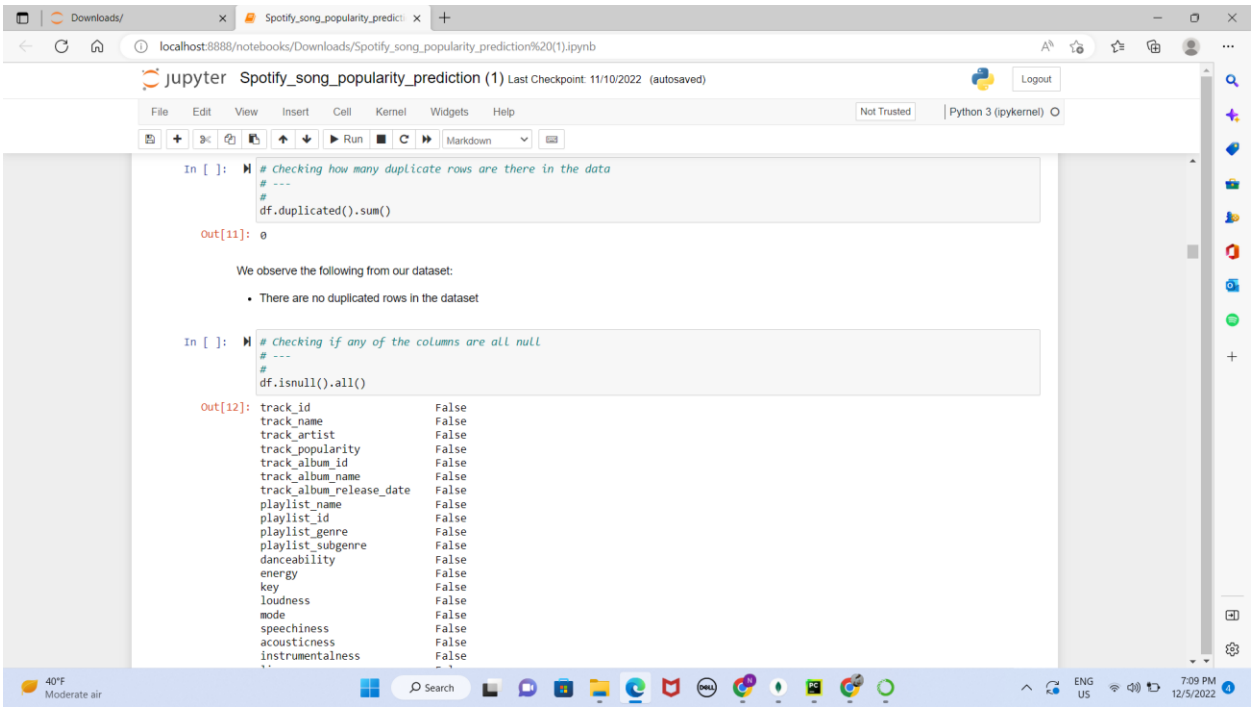
```
In [ ]: # Checking datatypes and missing entries of all the variables
        # ---
        df.isnull().sum()
```

```
Out[9]: track_id                  0
        track_name                5
        track_artist              5
        track_popularity          0
        track_album_id            0
        track_album_name          5
        track_album_release_date  0
        playlist_name             0
        playlist_id               0
        playlist_genre            0
        playlist_subgenre         0
        danceability              0
        energy                    0
        key                       0
        loudness                  0
        mode                      0
        speechiness               0
        acousticness              0
        instrumentalness          0
        liveness                  0
        valence                   0
        tempo                     0
        duration_ms               0
        dtype: int64
```

- From our observation we observe that some columns contain some missing values but thats very neglible.

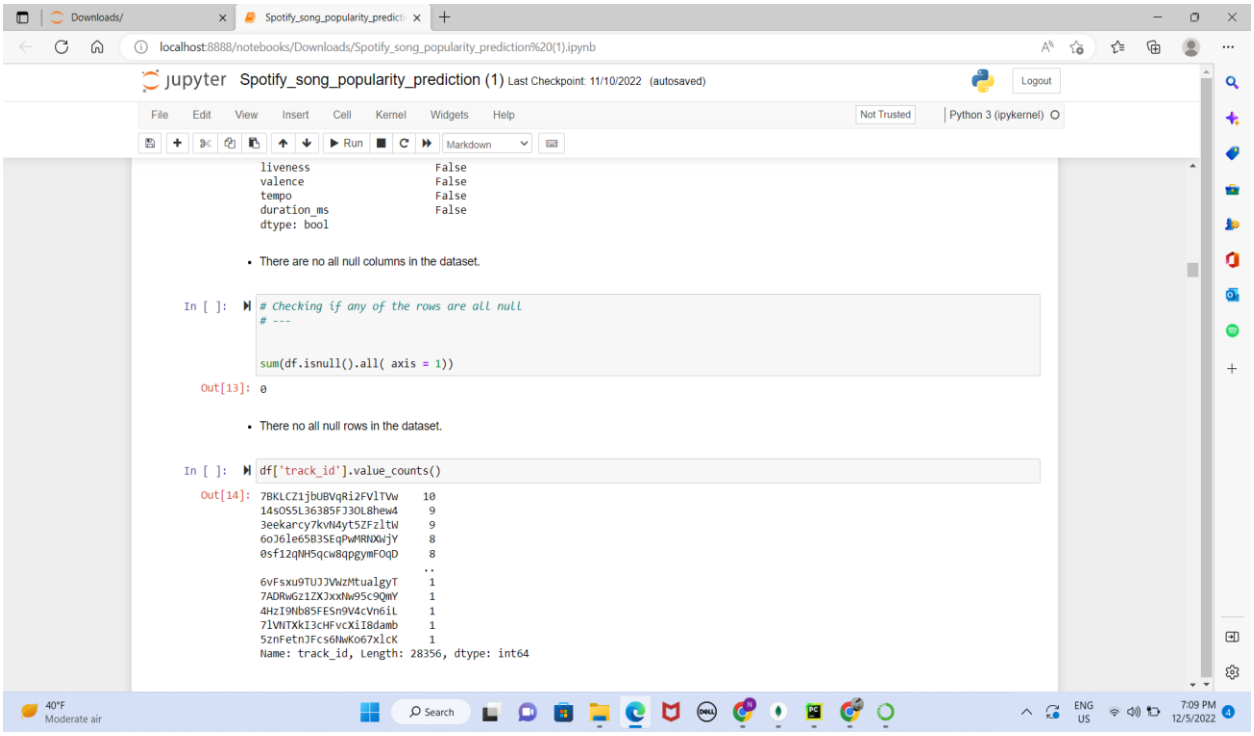

Downloads/
Spotify_song_popularity_predict
localhost:8888/notebooks/Downloads/Spotify_song_popularity_prediction%20(1).ipynb
jupyter Spotify_song_popularity_prediction (1) Last Checkpoint: 11/10/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Markdown
• From our observation we observe that some columns contain some missing values but thats very neglible.
In [ ]:
#missing values by percentage
missing_pct = df.isna().mean().round(4)*100
missing_pct
Out[10]:
track_id 0.00
track_name 0.02
track_artist 0.02
track_popularity 0.00
track_album_id 0.00
track_album_name 0.02
track_album_release_date 0.00
playlist_name 0.00
playlist_id 0.00
playlist_genre 0.00
playlist_subgenre 0.00
danceability 0.00
energy 0.00
key 0.00
loudness 0.00
mode 0.00
speechiness 0.00
acousticness 0.00
instrumentalness 0.00
liveness 0.00
valence 0.00
tempo 0.00
duration_ms 0.00
dtype: float64
In [ ]:
Checking how many duplicate rows are there in the data

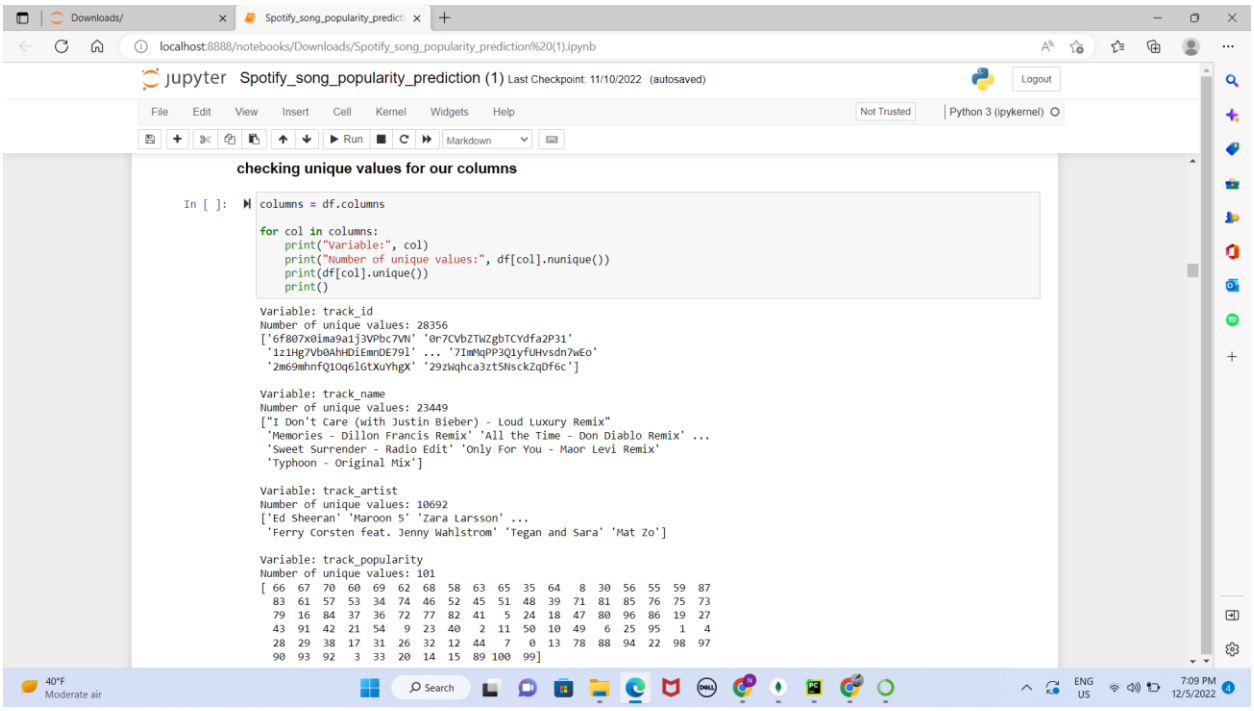
#
40°F
Moderate air
Search
ENG
US
7:09 PM
12/5/2022

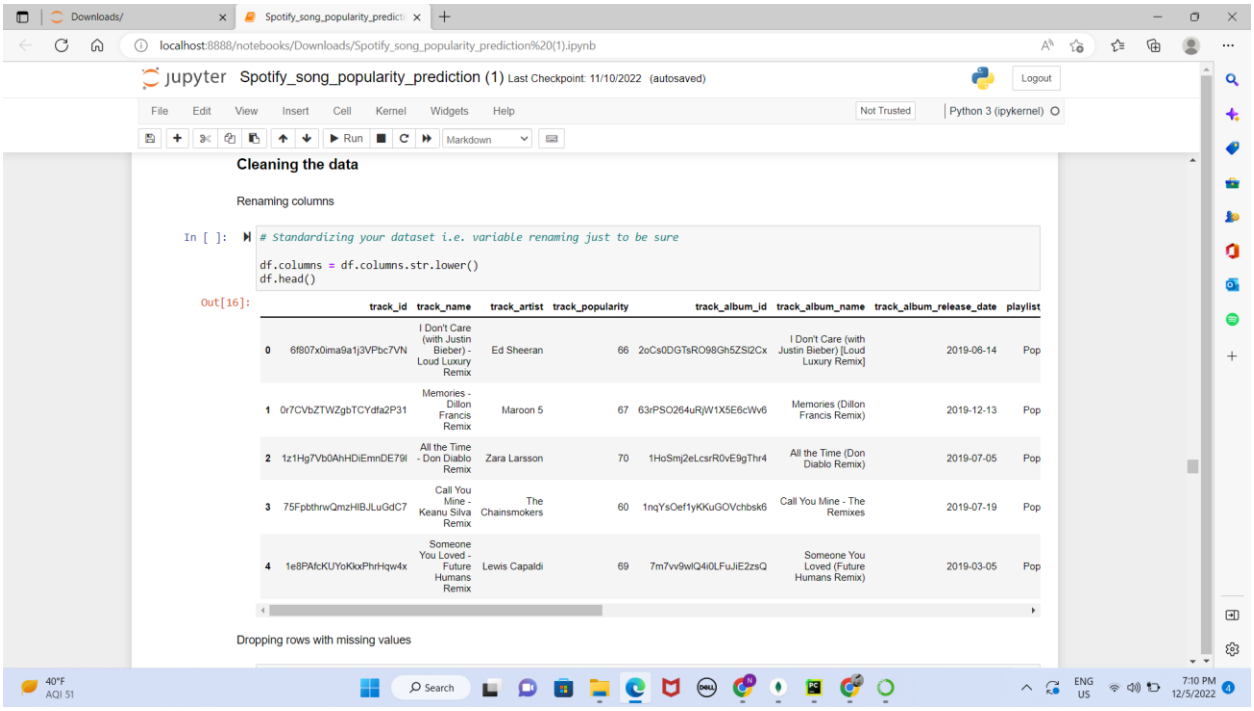

Downloads/
Spotify_song_popularity_predict
localhost:8888/notebooks/Downloads/Spotify_song_popularity_prediction%20(1).ipynb
jupyter Spotify_song_popularity_prediction (1) Last Checkpoint: 11/10/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Markdown
In [ ]:
Checking how many duplicate rows are there in the data

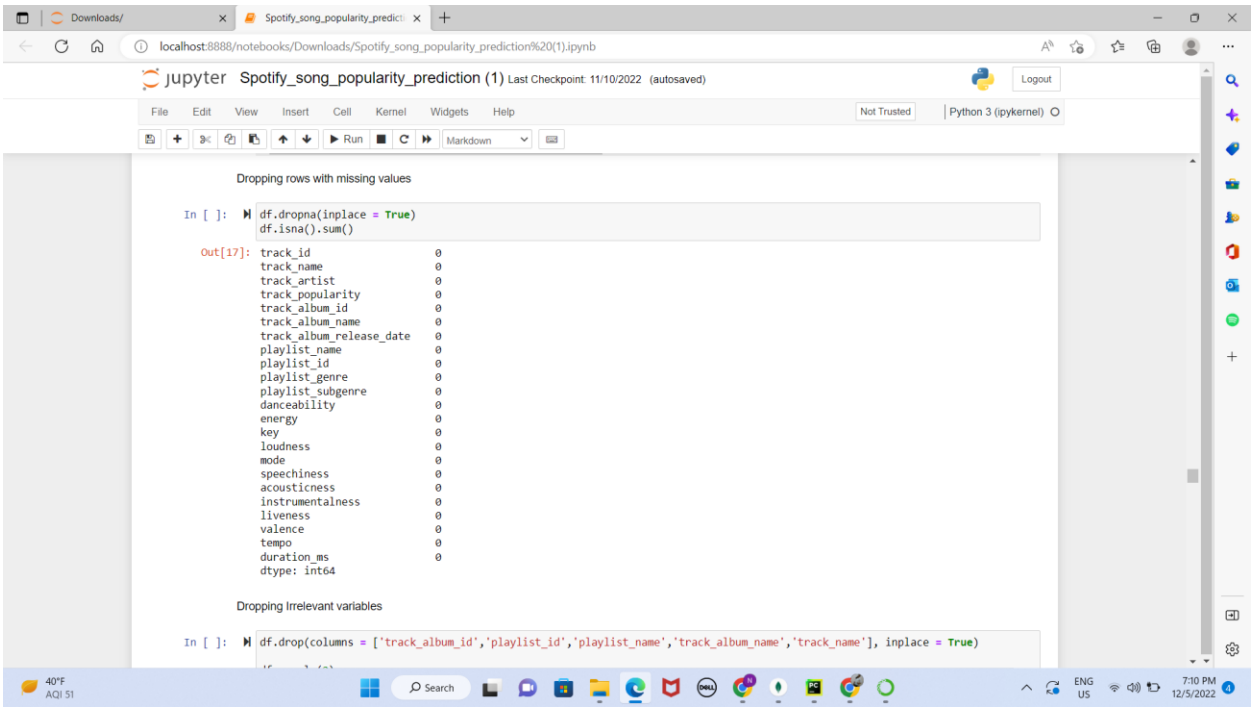
#
df.duplicated().sum()
Out[11]:
0
We observe the following from our dataset:
• There are no duplicated rows in the dataset
In [ ]:
Checking if any of the columns are all null

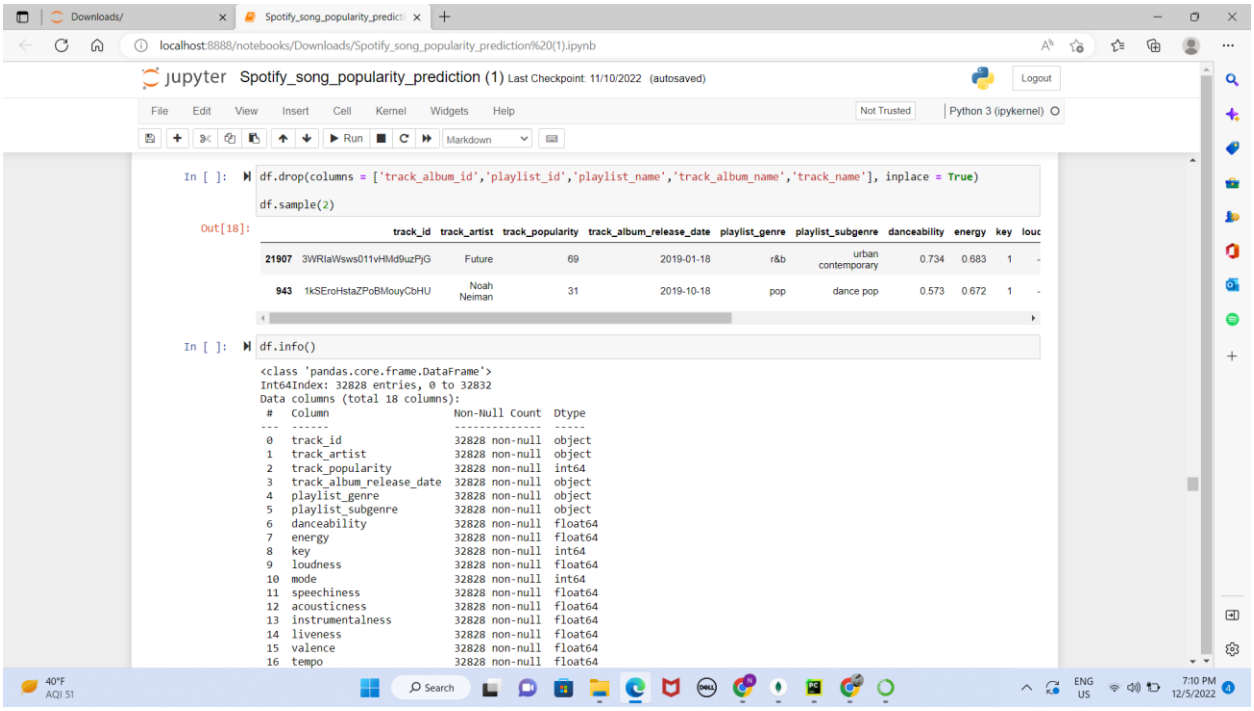
#
df.isnull().all()
Out[12]:
track_id False
track_name False
track_artist False
track_popularity False
track_album_id False
track_album_name False
track_album_release_date False
playlist_name False
playlist_id False
playlist_genre False
playlist_subgenre False
danceability False
energy False
key False
loudness False
mode False
speechiness False
acousticness False
instrumentalness False
40°F
Moderate air
Search
ENG
US
7:09 PM
12/5/2022

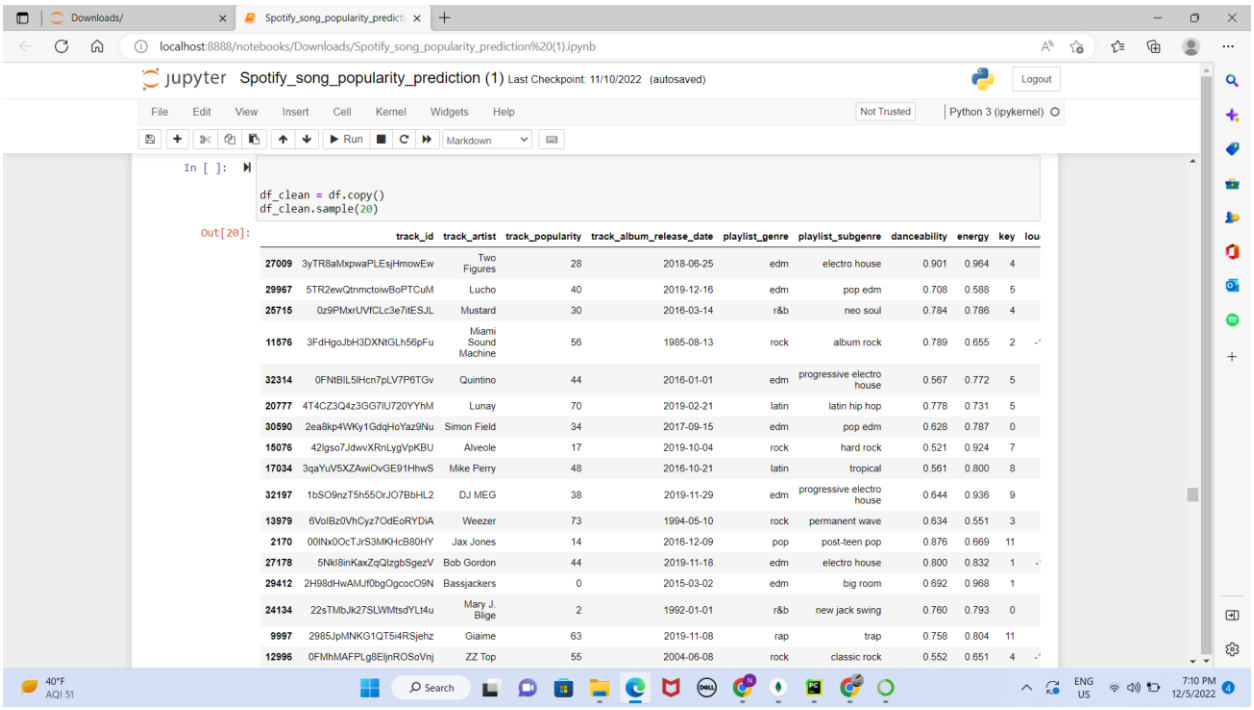

Downloads/
Spotify_song_popularity_predicti
localhost:8888/notebooks/Downloads/Spotify_song_popularity_prediction%20(1).ipynb
jupyter Spotify_song_popularity_prediction (1) Last Checkpoint: 11/10/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Markdown
liveness False
valence False
tempo False
duration_ms False
dtype: bool
• There are no all null columns in the dataset.
In [ ]:
# Checking if any of the rows are all null
# ---
sum(df.isnull().all(axis = 1))
Out[13]: 0
• There no all null rows in the dataset.
In [ ]:
df['track_id'].value_counts()
Out[14]:
7BKLCZ1jbUBVqRi2FVlTVw 10
14sOS5L36385FJ3OL8hew4 9
3eekarcy7kvN4yt5ZFzltW 9
6oJ6le65B3SEqPwMRNXWjY 8
0sf12qNH5qcw8qpgymFOqD 8
..
6vFsxu9TUJJVWzMtualgyT 1
7ADRwGz1ZXJxxNw95c9QmY 1
4HzI9Nb85FESn9V4cVn6iL 1
7lVNTXkI3cHFvcXiI8damb 1
5znFetnJFcs6NwKo67xlcK 1
Name: track_id, Length: 28356, dtype: int64
40°F
Moderate air
Search
ENG
US
7:09 PM
12/5/2022

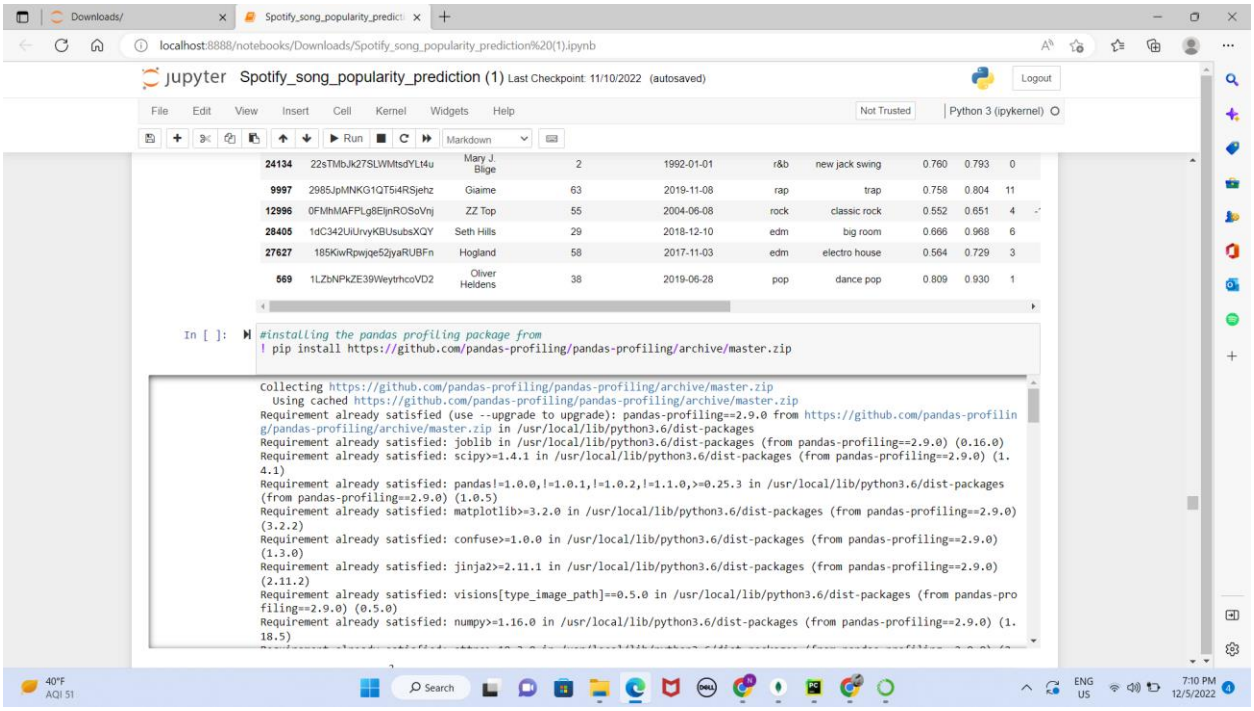

Downloads/
Spotify_song_popularity_predicti
localhost:8888/notebooks/Downloads/Spotify_song_popularity_prediction%20(1).ipynb
jupyter Spotify_song_popularity_prediction (1) Last Checkpoint: 11/10/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Markdown
checking unique values for our columns
In [ ]:
columns = df.columns
for col in columns:
print("Variable:", col)
print("Number of unique values:", df[col].nunique())
print(df[col].unique())
print()
Variable: track_id
Number of unique values: 28356
['6f807x0ima9a1j3VPbc7VN' '0r7CVbZTWZgbTCYdfa2P31'
'1z1Hg7Vb0AhHDiEmnDE79l' ... '7ImMqPP3Q1yfUHvsdn7wEo'
'2m69mhnfQ1Oq6lGtXuYhgX' '29zWqhca3zt5NsckZqDf6c']
Variable: track_name
Number of unique values: 23449
["I Don't Care (with Justin Bieber) - Loud Luxury Remix"
'Memories - Dillon Francis Remix' 'All the Time - Don Diablo Remix' ...
'Sweet Surrender - Radio Edit' 'Only For You - Maor Levi Remix'
'Typhoon - Original Mix']
Variable: track_artist
Number of unique values: 10692
['Ed Sheeran' 'Maroon 5' 'Zara Larsson' ...
'Ferry Corsten feat. Jenny Wahlstrom' 'Tegan and Sara' 'Mat Zo']
Variable: track_popularity
Number of unique values: 101
[66 67 70 60 69 62 68 58 63 65 35 64 8 30 56 55 59 87
83 61 57 53 34 74 46 52 45 51 48 39 71 81 85 76 75 73
79 16 84 37 36 72 77 82 41 5 24 18 47 80 96 86 19 27
43 91 42 21 54 9 23 40 2 11 50 10 49 6 25 95 1 4
28 29 38 17 31 26 32 12 44 7 0 13 78 88 94 22 98 97
90 93 92 3 33 20 14 15 89 100 99]
40°F
Moderate air
Search
ENG
US
7:09 PM
12/5/2022

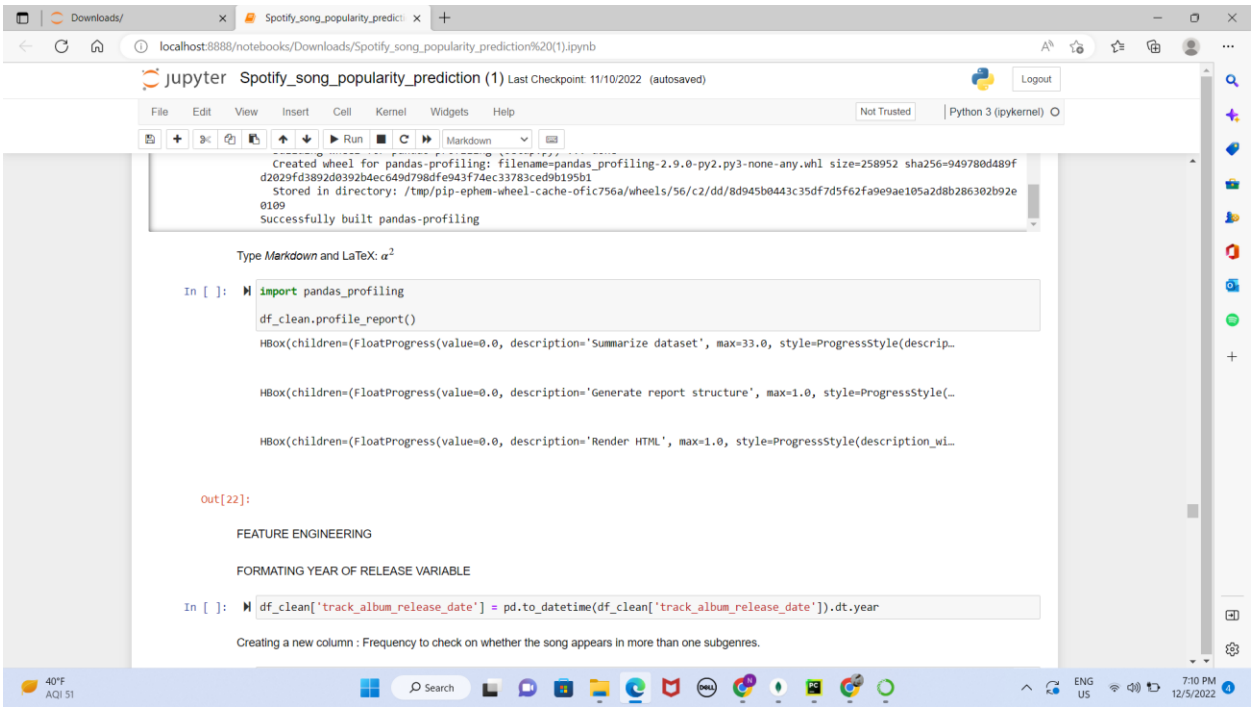

Downloads/
Spotify_song_popularity_predicti
localhost:8888/notebooks/Downloads/Spotify_song_popularity_prediction%20(1).ipynb
jupyter Spotify_song_popularity_prediction (1) Last Checkpoint: 11/10/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Markdown
Cleaning the data
Renaming columns
In [ ]:
# Standardizing your dataset i.e. variable renaming just to be sure
df.columns = df.columns.str.lower()
df.head()
Out[16]:
track_id track_name track_artist track_popularity track_album_id track_album_name track_album_release_date playlist
0 6f807x0ima9a1j3VPbc7VN I Don't Care (with Justin Bieber) - Loud Luxury Remix Ed Sheeran 66 2oCs0DGTsRO98Gh5ZSl2Cx I Don't Care (with Justin Bieber) [Loud Luxury Remix] 2019-06-14 Pop
1 0r7CVbZTWZgbTCYdfa2P31 Memories - Dillon Francis Remix Maroon 5 67 63rPSO264uRjW1X5E6cWv6 Memories (Dillon Francis Remix) 2019-12-13 Pop
2 1z1Hg7Vb0AhHDiEmnDE79l All the Time - Don Diablo Remix Zara Larsson 70 1HoSmj2eLcsrR0vE9gThr4 All the Time (Don Diablo Remix) 2019-07-05 Pop
3 75FpbthrwQmzHlBJLuGdC7 Call You Mine - Keanu Silva Remix The Chainsmokers 60 1nqYsOef1yKKuGOVchbsk6 Call You Mine - The Remixes 2019-07-19 Pop
4 1e8PAfcKUYoKkxPhrHqw4x Someone You Loved - Future Humans Remix Lewis Capaldi 69 7m7vv9wlQ4i0LFuJiE2zsQ Someone You Loved (Future Humans Remix) 2019-03-05 Pop
Dropping rows with missing values
40°F AQI 51
Search
ENG US
7:10 PM 12/5/2022

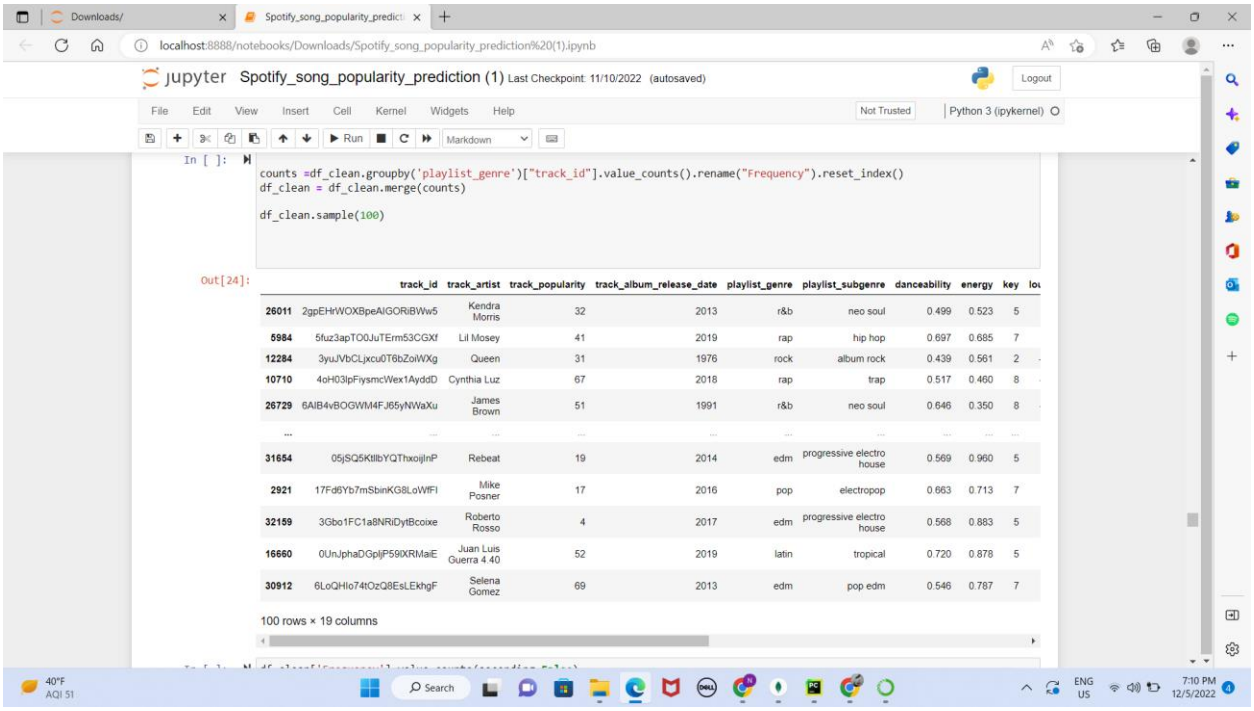

Downloads/
Spotify_song_popularity_predicti
localhost:8888/notebooks/Downloads/Spotify_song_popularity_prediction%20(1).ipynb
jupyter Spotify_song_popularity_prediction (1) Last Checkpoint: 11/10/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Markdown
Dropping rows with missing values
In [ ]:
df.dropna(inplace = True)
df.isna().sum()
Out[17]:
track_id 0
track_name 0
track_artist 0
track_popularity 0
track_album_id 0
track_album_name 0
track_album_release_date 0
playlist_name 0
playlist_id 0
playlist_genre 0
playlist_subgenre 0
danceability 0
energy 0
key 0
loudness 0
mode 0
speechiness 0
acousticness 0
instrumentalness 0
liveness 0
valence 0
tempo 0
duration_ms 0
dtype: int64
Dropping Irrelevant variables
In [ ]:
df.drop(columns = ['track_album_id','playlist_id','playlist_name','track_album_name','track_name'], inplace = True)
40°F AQI 51
Search
ENG US
7:10 PM 12/5/2022

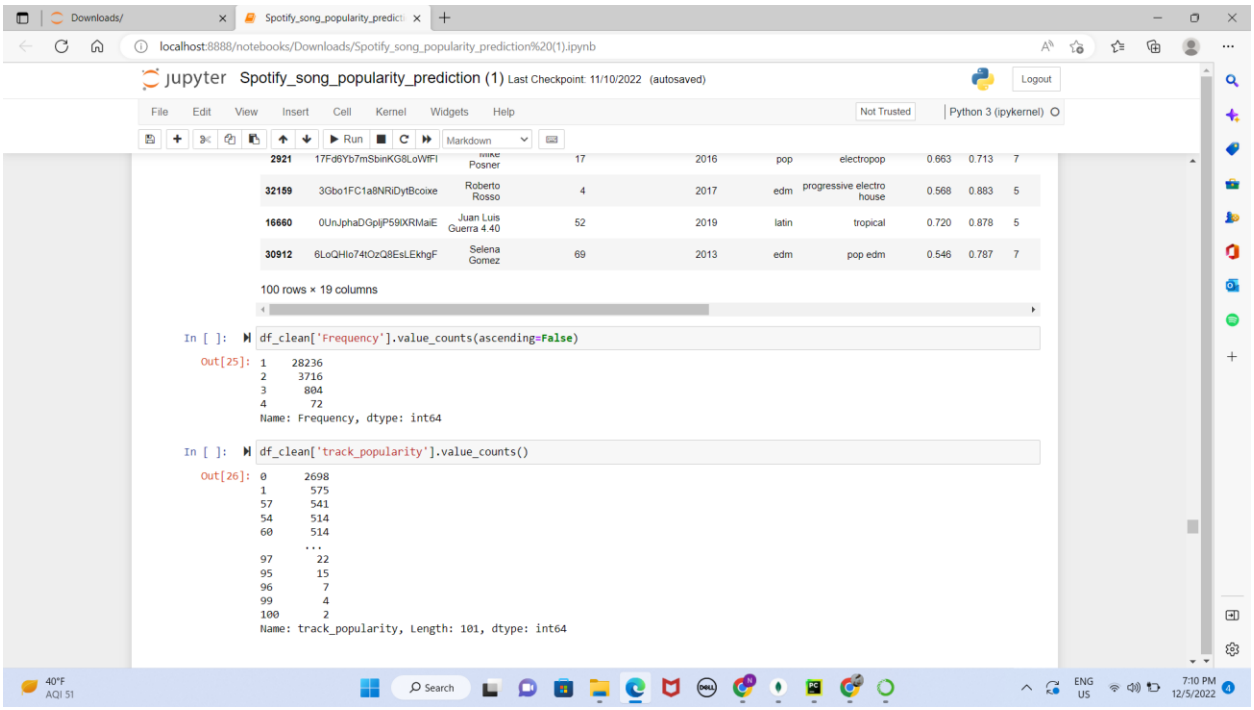

Downloads/
Spotify_song_popularity_predicti
localhost:8888/notebooks/Downloads/Spotify_song_popularity_prediction%20(1).ipynb
jupyter Spotify_song_popularity_prediction (1) Last Checkpoint: 11/10/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Markdown
In [ ]:
df.drop(columns = ['track_album_id','playlist_id','playlist_name','track_album_name','track_name'], inplace = True)
df.sample(2)
Out[18]:
track_id track_artist track_popularity track_album_release_date playlist_genre playlist_subgenre danceability energy key loud
21907 3WRIaWsws011vHMd9uzPjG Future 69 2019-01-18 r&b urban contemporary 0.734 0.683 1
943 1kSEroHstaZPoBMouyCbHU Noah Neiman 31 2019-10-18 pop dance pop 0.573 0.672 1
In [ ]:
df.info()
<class 'pandas.core.frame.DataFrame'>
Int64Index: 32828 entries, 0 to 32832
Data columns (total 18 columns):
Column Non-Null Count Dtype
0 track_id 32828 non-null object
1 track_artist 32828 non-null object
2 track_popularity 32828 non-null int64
3 track_album_release_date 32828 non-null object
4 playlist_genre 32828 non-null object
5 playlist_subgenre 32828 non-null object
6 danceability 32828 non-null float64
7 energy 32828 non-null float64
8 key 32828 non-null int64
9 loudness 32828 non-null float64
10 mode 32828 non-null int64
11 speechiness 32828 non-null float64
12 acousticness 32828 non-null float64
13 instrumentalness 32828 non-null float64
14 liveness 32828 non-null float64
15 valence 32828 non-null float64
16 tempo 32828 non-null float64
40°F AQI 51
Search
ENG US
7:10 PM 12/5/2022

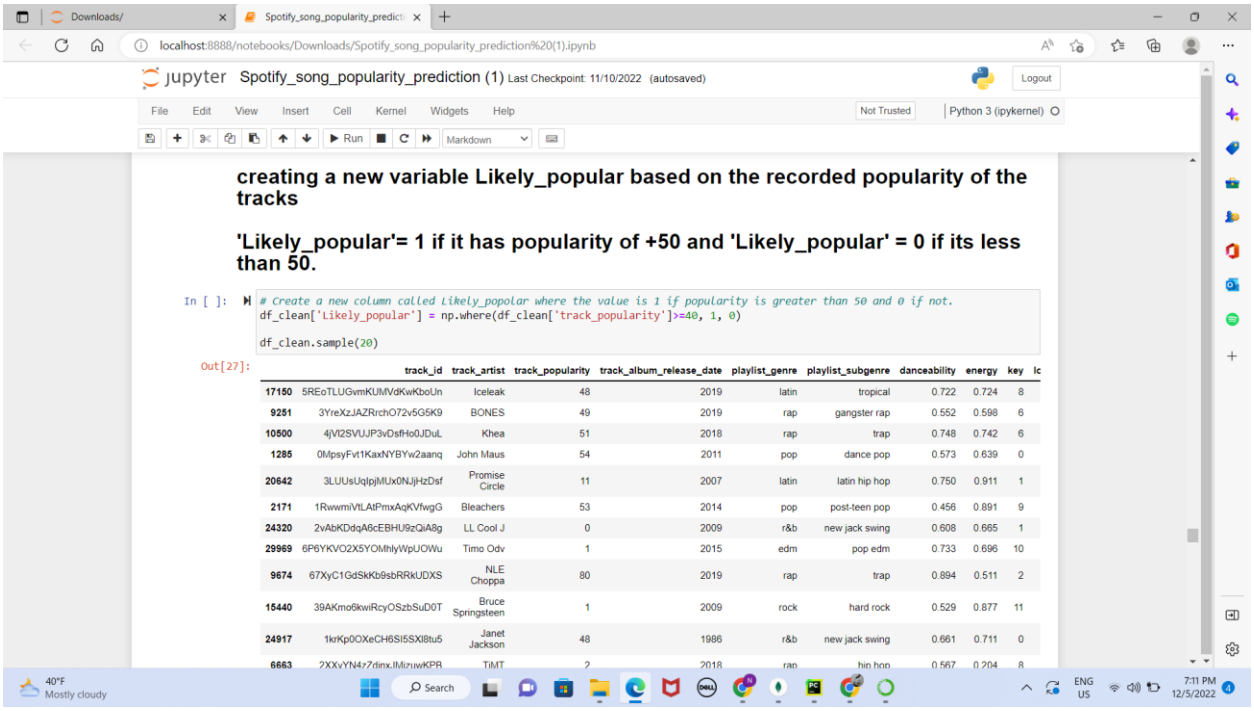

Downloads/
Spotify_song_popularity_predicti
localhost:8888/notebooks/Downloads/Spotify_song_popularity_prediction%20(1).ipynb
jupyter Spotify_song_popularity_prediction (1) Last Checkpoint: 11/10/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Markdown
In [ ]:
df_clean = df.copy()
df_clean.sample(20)
Out[20]:
track_id track_artist track_popularity track_album_release_date playlist_genre playlist_subgenre danceability energy key lou
27009 3yTR8aMxpwaPLEsjHmowEw Two Figures 28 2018-06-25 edm electro house 0.901 0.964 4
29967 5TR2ewQtnmctoiwBoPTCuM Lucho 40 2019-12-16 edm pop edm 0.708 0.588 5
25715 0z9PMxrUVfCLc3e7itESJL Mustard 30 2016-03-14 r&b neo soul 0.784 0.786 4
11676 3FdHgoJbH3DXNtGLh56pFu Miami Sound Machine 56 1985-08-13 rock album rock 0.789 0.655 2
32314 0FNtBIL5lHcn7pLV7P6TGv Quintino 44 2016-01-01 edm progressive electro house 0.567 0.772 5
20777 4T4CZ3Q4z3GG7lU720YYhM Lunay 70 2019-02-21 latin latin hip hop 0.778 0.731 5
30590 2ea8kp4WKy1GdqHoYaz9Nu Simon Field 34 2017-09-15 edm pop edm 0.628 0.787 0
15076 42lgso7JdwvXRnLygVpKBU Alveole 17 2019-10-04 rock hard rock 0.521 0.924 7
17034 3qaYuV5XZAwiOvGE91HhwS Mike Perry 48 2016-10-21 latin tropical 0.561 0.800 8
32197 1bSO9nzT5h55OrJO7BbHL2 DJ MEG 38 2019-11-29 edm progressive electro house 0.644 0.936 9
13979 6VolBz0VhCyz7OdEoRYDiA Weezer 73 1994-05-10 rock permanent wave 0.634 0.551 3
2170 00INx0OcTJrS3MKHcB80HY Jax Jones 14 2016-12-09 pop post-teen pop 0.876 0.669 11
27178 5Nkl8inKaxZqQlzgbSgezV Bob Gordon 44 2019-11-18 edm electro house 0.800 0.832 1
29412 2H98dHwAMJf0bgOgcocO9N Bassjackers 0 2015-03-02 edm big room 0.692 0.968 1
24134 22sTMbJk27SLWMtsdYLt4u Mary J. Blige 2 1992-01-01 r&b new jack swing 0.760 0.793 0
9997 2985JpMNKG1QT5i4RSjehz Giaime 63 2019-11-08 rap trap 0.758 0.804 11
12996 0FMhMAFPLg8EljnROSoVnj ZZ Top 55 2004-06-08 rock classic rock 0.552 0.651 4
40°F AQI 51
Search
ENG US
7:10 PM 12/5/2022

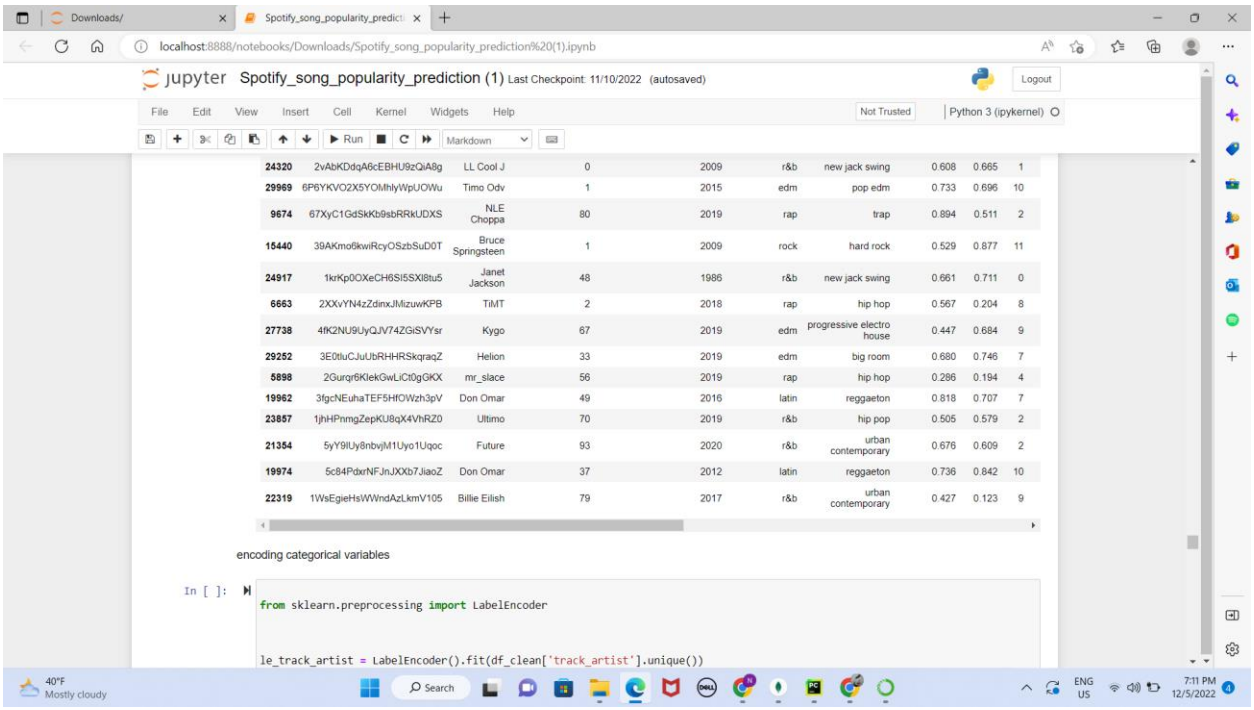

Downloads/
Spotify_song_popularity_predict
localhost:8888/notebooks/Downloads/Spotify_song_popularity_prediction%20(1).ipynb
jupyter Spotify_song_popularity_prediction (1) Last Checkpoint: 11/10/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Markdown
24134 22sTMbJk27SLWMtsdYLt4u Mary J. Blige 2 1992-01-01 r&b new jack swing 0.760 0.793 0
9997 2985JpMNKG1QT5i4RSjehz Giaime 63 2019-11-08 rap trap 0.758 0.804 11
12996 0FMhMAFPLg8EljnROSoVnj ZZ Top 55 2004-06-08 rock classic rock 0.552 0.651 4
28405 1dC342UiUrvyKBUsubsXQY Seth Hills 29 2018-12-10 edm big room 0.666 0.968 6
27627 185KiwRpwjqe52jyaRUBFn Hogland 58 2017-11-03 edm electro house 0.564 0.729 3
569 1LZbNPkZE39WeytrhcoVD2 Oliver Heldens 38 2019-06-28 pop dance pop 0.809 0.930 1
In [ ]:
#installing the pandas profiling package from
! pip install https://github.com/pandas-profiling/pandas-profiling/archive/master.zip
Collecting https://github.com/pandas-profiling/pandas-profiling/archive/master.zip
Using cached https://github.com/pandas-profiling/pandas-profiling/archive/master.zip
Requirement already satisfied (use --upgrade to upgrade): pandas-profiling==2.9.0 from https://github.com/pandas-profiling/pandas-profiling/archive/master.zip in /usr/local/lib/python3.6/dist-packages
Requirement already satisfied: joblib in /usr/local/lib/python3.6/dist-packages (from pandas-profiling==2.9.0) (0.16.0)
Requirement already satisfied: scipy>=1.4.1 in /usr/local/lib/python3.6/dist-packages (from pandas-profiling==2.9.0) (1.4.1)
Requirement already satisfied: pandas!=1.0.0,!=1.0.1,!=1.0.2,!=1.1.0,>=0.25.3 in /usr/local/lib/python3.6/dist-packages (from pandas-profiling==2.9.0) (1.0.5)
Requirement already satisfied: matplotlib>=3.2.0 in /usr/local/lib/python3.6/dist-packages (from pandas-profiling==2.9.0) (3.2.2)
Requirement already satisfied: confuse>=1.0.0 in /usr/local/lib/python3.6/dist-packages (from pandas-profiling==2.9.0) (1.3.0)
Requirement already satisfied: jinja2>=2.11.1 in /usr/local/lib/python3.6/dist-packages (from pandas-profiling==2.9.0) (2.11.2)
Requirement already satisfied: visions[type_image_path]==0.5.0 in /usr/local/lib/python3.6/dist-packages (from pandas-profiling==2.9.0) (0.5.0)
Requirement already satisfied: numpy>=1.16.0 in /usr/local/lib/python3.6/dist-packages (from pandas-profiling==2.9.0) (1.18.5)
40°F AQI 51
Search
ENG US
7:10 PM 12/5/2022

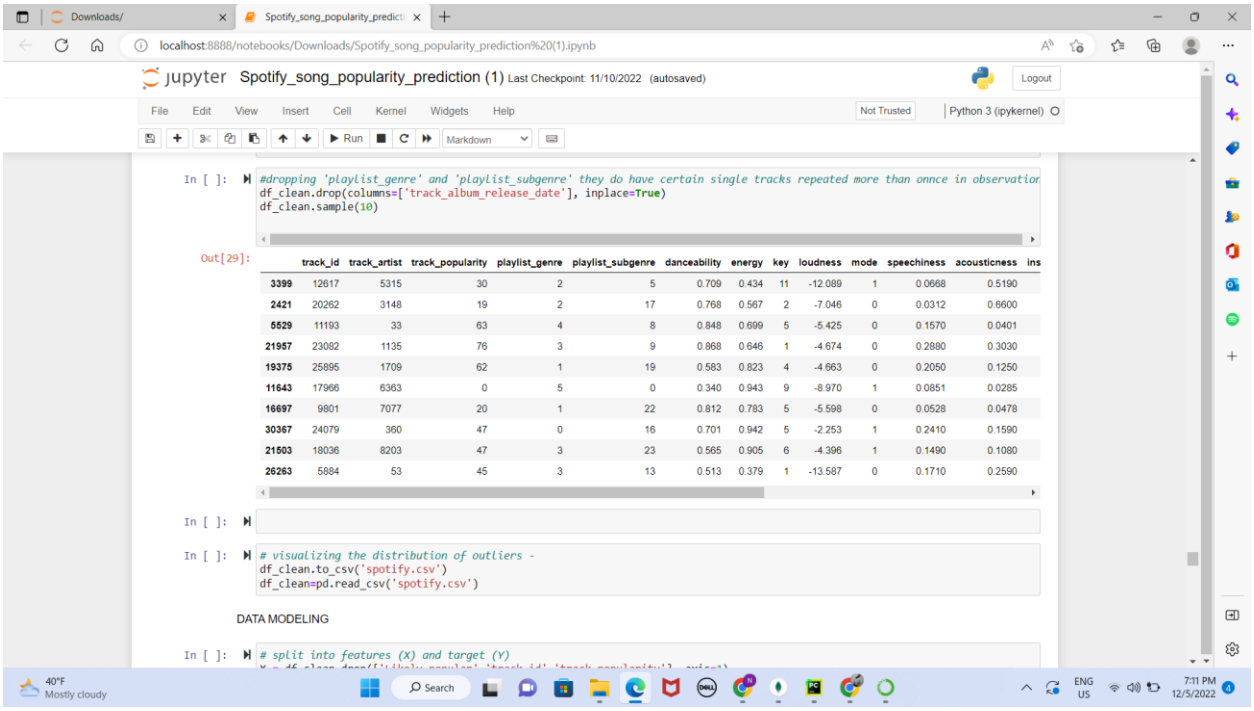

Downloads/
Spotify_song_popularity_predict
localhost:8888/notebooks/Downloads/Spotify_song_popularity_prediction%20(1).ipynb
jupyter Spotify_song_popularity_prediction (1) Last Checkpoint: 11/10/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Markdown
Created wheel for pandas-profiling: filename=pandas_profiling-2.9.0-py2.py3-none-any.whl size=258952 sha256=949780d489fd2029fd3892d0392b4ec649d798dfe943f74ec33783ced9b195b1
Stored in directory: /tmp/pip-ephem-wheel-cache-ofic756a/wheels/56/c2/dd/8d945b0443c35df7d5f62fa9e9ae105a2d8b286302b92e0109
Successfully built pandas-profiling
Type Markdown and LaTeX: α^2
In [ ]:
import pandas_profiling
df_clean.profile_report()
HBox(children=(FloatProgress(value=0.0, description='Summarize dataset', max=33.0, style=ProgressStyle(descrip…
HBox(children=(FloatProgress(value=0.0, description='Generate report structure', max=1.0, style=ProgressStyle(…
HBox(children=(FloatProgress(value=0.0, description='Render HTML', max=1.0, style=ProgressStyle(description_wi…
Out[22]:
FEATURE ENGINEERING
FORMATING YEAR OF RELEASE VARIABLE
In [ ]:
df_clean['track_album_release_date'] = pd.to_datetime(df_clean['track_album_release_date']).dt.year
Creating a new column : Frequency to check on whether the song appears in more than one subgenres.
40°F AQI 51
Search
ENG US
7:10 PM 12/5/2022

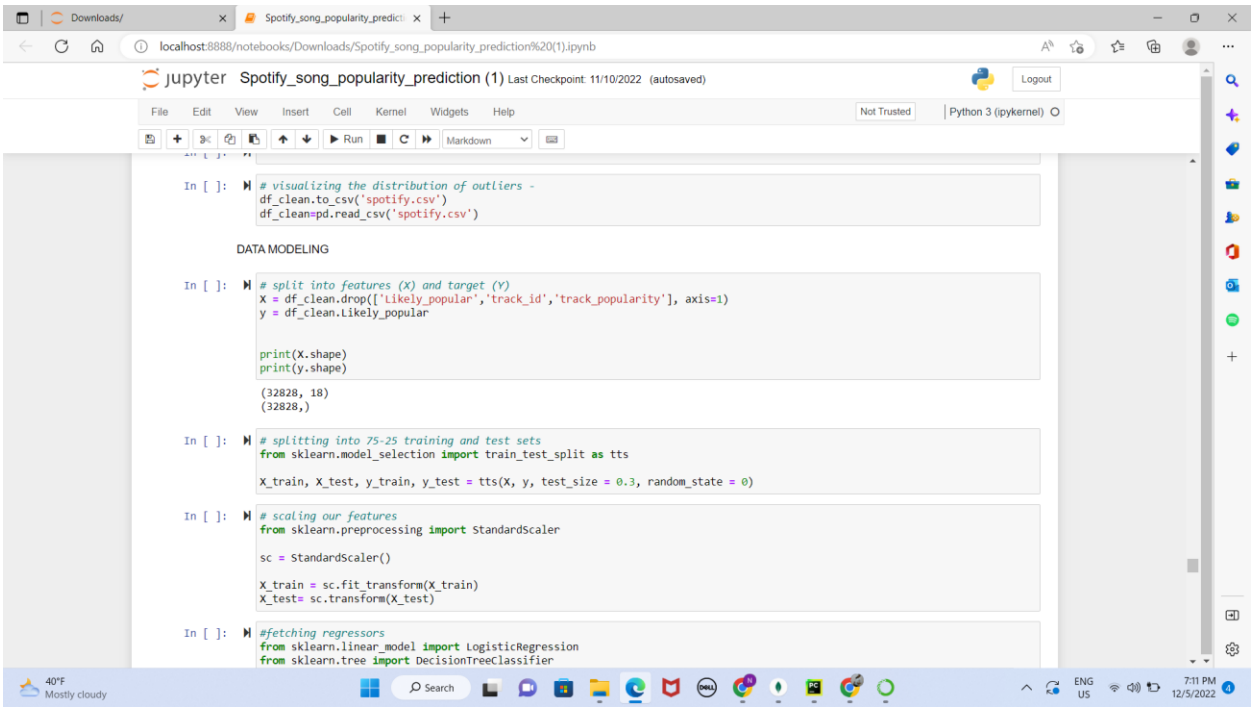

Downloads/
Spotify_song_popularity_predict
localhost:8888/notebooks/Downloads/Spotify_song_popularity_prediction%20(1).ipynb
jupyter Spotify_song_popularity_prediction (1) Last Checkpoint: 11/10/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Markdown
In [ ]:
counts =df_clean.groupby('playlist_genre')["track_id"].value_counts().rename("Frequency").reset_index()
df_clean = df_clean.merge(counts)
df_clean.sample(100)
Out[24]:
track_id track_artist track_popularity track_album_release_date playlist_genre playlist_subgenre danceability energy key
26011 2gpEHrWOXBpeAlGORiBWw5 Kendra Morris 32 2013 r&b neo soul 0.499 0.523 5
5984 5fuz3apTO0JuTErm53CGXf Lil Mosey 41 2019 rap hip hop 0.697 0.685 7
12284 3yuJVbCLjxcu0T6bZoiWXg Queen 31 1976 rock album rock 0.439 0.561 2
10710 4oH03lpFiysmcWex1AyddD Cynthia Luz 67 2018 rap trap 0.517 0.460 8
26729 6AIB4vBOGWM4FJ65yNWaXu James Brown 51 1991 r&b neo soul 0.646 0.350 8
31654 05jSQ5KtllbYQThxoijInP Rebeat 19 2014 edm progressive electro house 0.569 0.960 5
2921 17Fd6Yb7mSbinKG8LoWfFl Mike Posner 17 2016 pop electropop 0.663 0.713 7
32159 3Gbo1FC1a8NRiDytBcoixe Roberto Rosso 4 2017 edm progressive electro house 0.568 0.883 5
16660 0UnJphaDGpljP59lXRMaiE Juan Luis Guerra 4.40 52 2019 latin tropical 0.720 0.878 5
30912 6LoQHlo74tOzQ8EsLEkhgF Selena Gomez 69 2013 edm pop edm 0.546 0.787 7
100 rows × 19 columns
40°F AQI 51
Search
ENG US
7:10 PM 12/5/2022

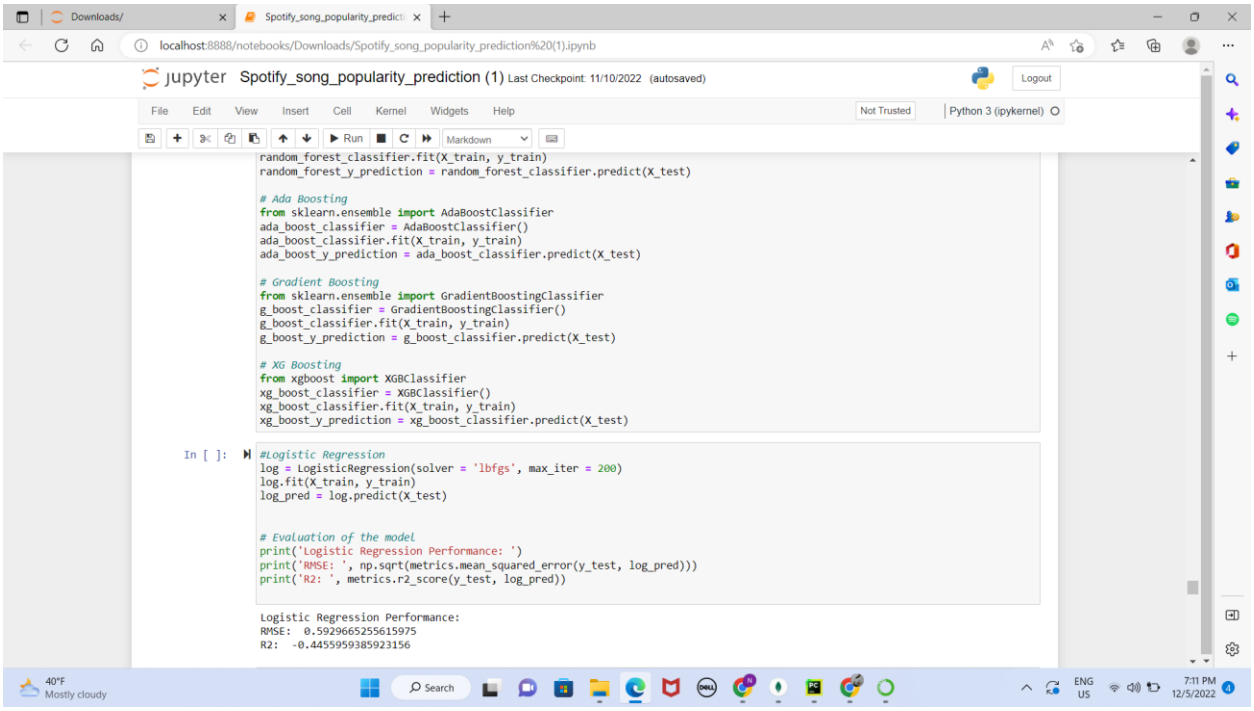

Downloads/
Spotify_song_popularity_predict
localhost:8888/notebooks/Downloads/Spotify_song_popularity_prediction%20(1).ipynb
jupyter Spotify_song_popularity_prediction (1) Last Checkpoint: 11/10/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Markdown
2921 17Fd6Yb7mSbinKG8LoWfFl Mike Posner 17 2016 pop electropop 0.663 0.713 7
32159 3Gbo1FC1a8NRiDytBcoixe Roberto Rosso 4 2017 edm progressive electro house 0.568 0.883 5
16660 0UnJphaDGpljP59lXRMaiE Juan Luis Guerra 4.40 52 2019 latin tropical 0.720 0.878 5
30912 6LoQHlo74tOzQ8EsLEkhgF Selena Gomez 69 2013 edm pop edm 0.546 0.787 7
100 rows × 19 columns
In [ ]:
df_clean['Frequency'].value_counts(ascending=False)
Out[25]:
1 28236
2 3716
3 804
4 72
Name: Frequency, dtype: int64
In [ ]:
df_clean['track_popularity'].value_counts()
Out[26]:
0 2698
1 575
57 541
54 514
60 514
...
97 22
95 15
96 7
99 4
100 2
Name: track_popularity, Length: 101, dtype: int64
40°F AQI 51
Search
ENG US
7:10 PM 12/5/2022

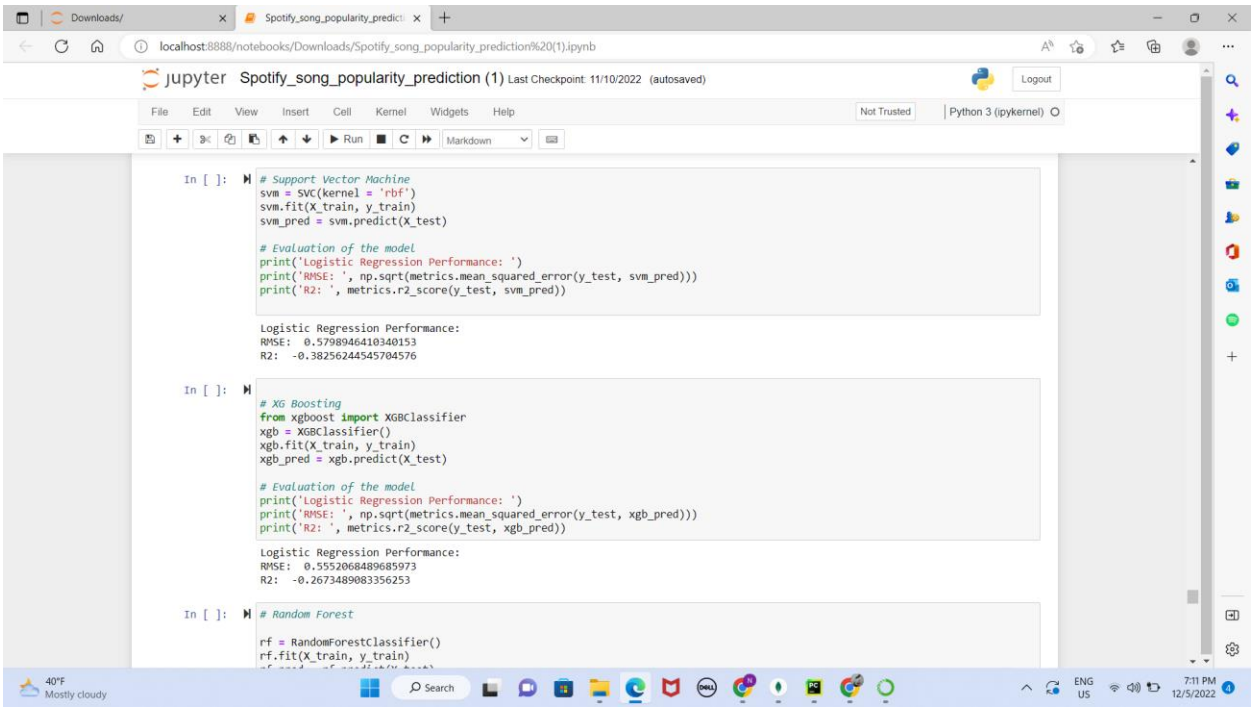

Downloads/
Spotify_song_popularity_predict
localhost:8888/notebooks/Downloads/Spotify_song_popularity_prediction%20(1).ipynb
jupyter Spotify_song_popularity_prediction (1) Last Checkpoint: 11/10/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Markdown
creating a new variable Likely_popular based on the recorded popularity of the tracks
'Likely_popular'= 1 if it has popularity of +50 and 'Likely_popular' = 0 if its less than 50.
In [ ]:
Create a new column called Likely_popular where the value is 1 if popularity is greater than 50 and 0 if not.
df_clean['Likely_popular'] = np.where(df_clean['track_popularity']>=40, 1, 0)
df_clean.sample(20)
Out[27]:
track_id track_artist track_popularity track_album_release_date playlist_genre playlist_subgenre danceability energy key lo
17150 5REoTLUGvmKUMVdKwKboUn Iceleak 48 2019 latin tropical 0.722 0.724 8
9251 3YreXzJAZRrchO72v5G5K9 BONES 49 2019 rap gangster rap 0.552 0.598 6
10500 4jVl2SVUJP3vDsfHo0JDuL Khea 51 2018 rap trap 0.748 0.742 6
1285 0MpsyFvt1KaxNYBYw2aanq John Maus 54 2011 pop dance pop 0.573 0.639 0
20642 3LUUsUqIpjMUx0NJjHzDsf Promise Circle 11 2007 latin latin hip hop 0.750 0.911 1
2171 1RwwmiVtLAtPmxAqKVfwgG Bleachers 53 2014 pop post-teen pop 0.456 0.891 9
24320 2vAbKDdqA6cEBHU9zQiA8g LL Cool J 0 2009 r&b new jack swing 0.608 0.665 1
29969 6P6YKVO2X5YOMhlyWpUOWu Timo Odv 1 2015 edm pop edm 0.733 0.696 10
9674 67XyC1GdSkKb9sbRRkUDXS NLE Choppa 80 2019 rap trap 0.894 0.511 2
15440 39AKmo6kwiRcyOSzbSuD0T Bruce Springsteen 1 2009 rock hard rock 0.529 0.877 11
24917 1krKp0OXeCH6Si5SXl8tu5 Janet Jackson 48 1986 r&b new jack swing 0.661 0.711 0
6663 2XXvYN4zZdinxJMizuwKPB TiMT 2 2018 rap hip hop 0.567 0.204 8
40°F Mostly cloudy
Search
ENG US
7:11 PM 12/5/2022

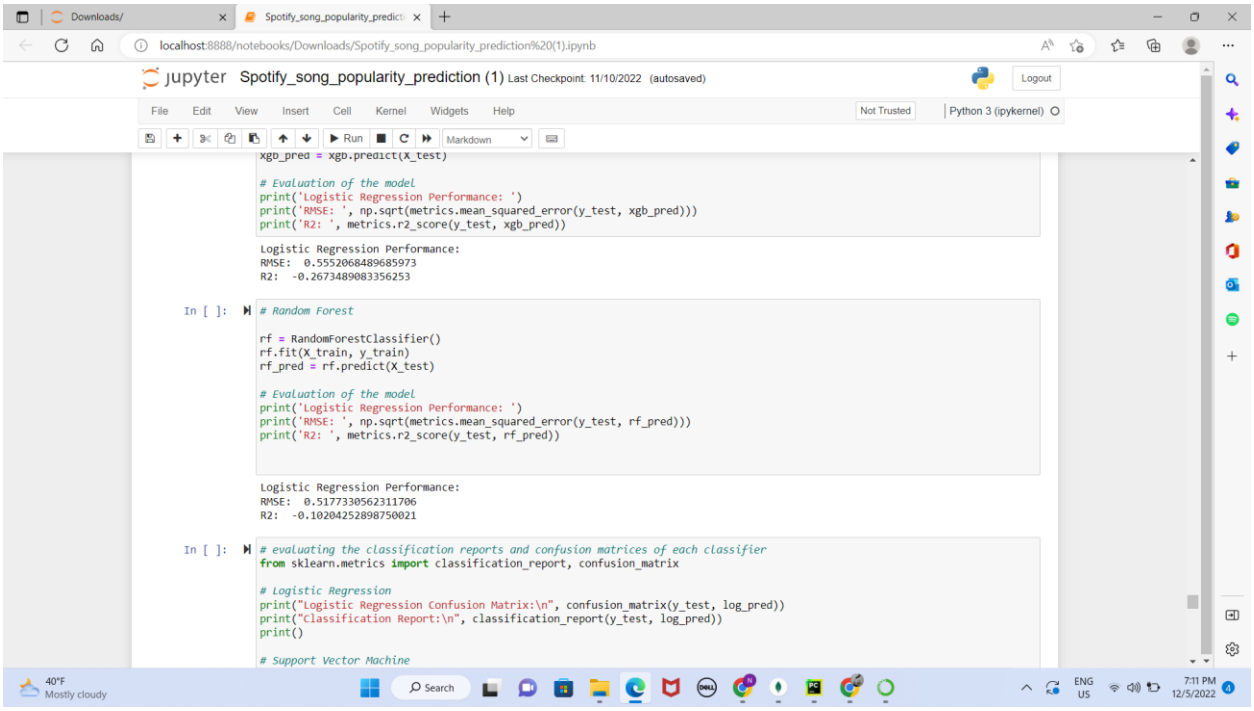

Downloads/
Spotify_song_popularity_predict
localhost:8888/notebooks/Downloads/Spotify_song_popularity_prediction%20(1).ipynb
jupyter Spotify_song_popularity_prediction (1) Last Checkpoint: 11/10/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Markdown
24320 2vAbKDdqA6cEBHU9zQiA8g LL Cool J 0 2009 r&b new jack swing 0.608 0.665 1
29969 6P6YKVO2X5YOMhlyWpUOWu Timo Odv 1 2015 edm pop edm 0.733 0.696 10
9674 67XyC1GdSkKb9sbRRkUDXS NLE Choppa 80 2019 rap trap 0.894 0.511 2
15440 39AKmo6kwiRcyOSzbSuD0T Bruce Springsteen 1 2009 rock hard rock 0.529 0.877 11
24917 1krKp0OXeCH6Si5SXl8tu5 Janet Jackson 48 1986 r&b new jack swing 0.661 0.711 0
6663 2XXvYN4zZdinxJMizuwKPB TiMT 2 2018 rap hip hop 0.567 0.204 8
27738 4fK2NU9UyQJV74ZGiSVYsr Kygo 67 2019 edm progressive electro house 0.447 0.684 9
29252 3E0tluCJuUbRHHRSkqraqZ Helion 33 2019 edm big room 0.680 0.746 7
5898 2Gurqr6KlekGwLiCt0gGKX mr_slace 56 2019 rap hip hop 0.286 0.194 4
19962 3fgcNEuhaTEF5HfOWzh3pV Don Omar 49 2016 latin reggaeton 0.818 0.707 7
23857 1jhHPnmgZepKU8qX4VhRZ0 Ultimo 70 2019 r&b hip pop 0.505 0.579 2
21354 5yY9lUy8nbvjM1Uyo1Uqoc Future 93 2020 r&b urban contemporary 0.676 0.609 2
19974 5c84PdxrNFJnJXXb7JiaoZ Don Omar 37 2012 latin reggaeton 0.736 0.842 10
22319 1WsEgieHsWWndAzLkmV105 Billie Eilish 79 2017 r&b urban contemporary 0.427 0.123 9
encoding categorical variables
In [ ]:
from sklearn.preprocessing import LabelEncoder
le_track_artist = LabelEncoder().fit(df_clean['track_artist'].unique())
40°F Mostly cloudy
Search
ENG US
7:11 PM 12/5/2022

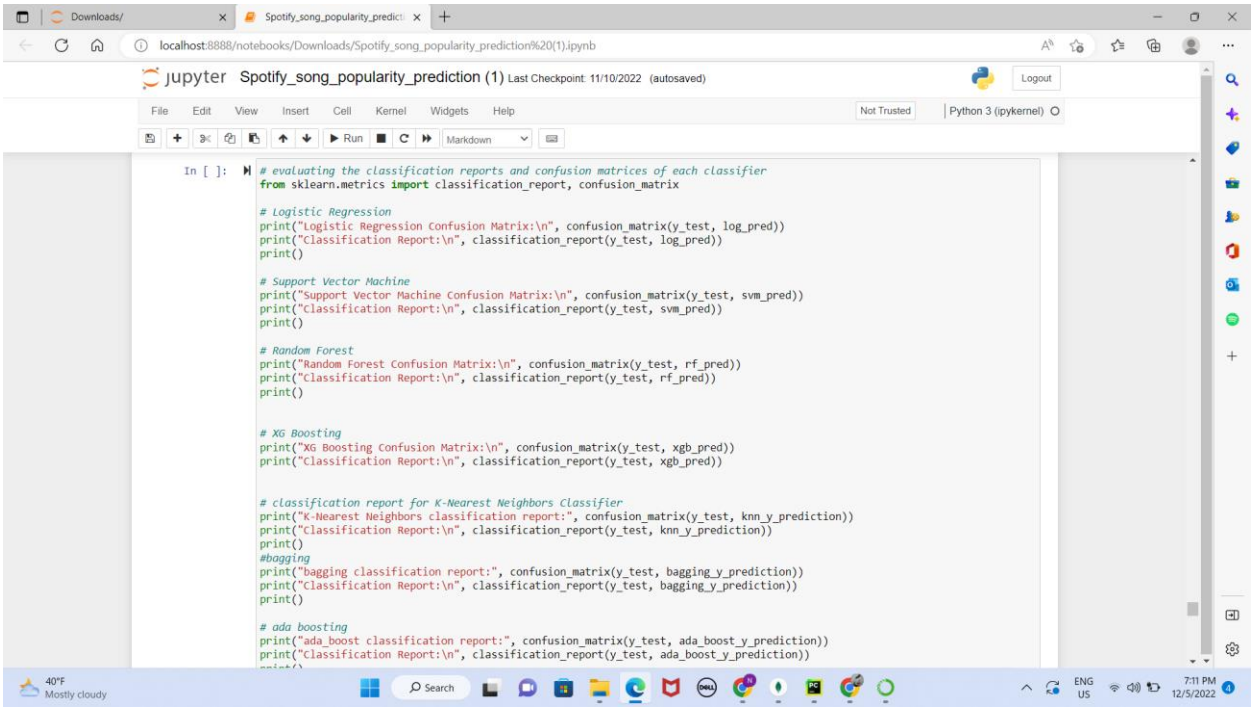

Downloads/
Spotify_song_popularity_predicti
localhost:8888/notebooks/Downloads/Spotify_song_popularity_prediction%20(1).ipynb
jupyter Spotify_song_popularity_prediction (1) Last Checkpoint: 11/10/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Markdown
In [ ]:
#dropping 'playlist_genre' and 'playlist_subgenre' they do have certain single tracks repeated more than onnce in observation
df_clean.drop(columns=['track_album_release_date'], inplace=True)
df_clean.sample(10)
Out[29]:
track_id track_artist track_popularity playlist_genre playlist_subgenre danceability energy key loudness mode speechiness acousticness ins
3399 12617 5315 30 2 5 0.709 0.434 11 -12.089 1 0.0668 0.5190
2421 20262 3148 19 2 17 0.768 0.567 2 -7.046 0 0.0312 0.6600
5529 11193 33 63 4 8 0.848 0.699 5 -5.425 0 0.1570 0.0401
21957 23082 1135 76 3 9 0.868 0.646 1 -4.674 0 0.2880 0.3030
19375 25895 1709 62 1 19 0.583 0.823 4 -4.663 0 0.2050 0.1250
11643 17966 6363 0 5 0 0.340 0.943 9 -8.970 1 0.0851 0.0285
16697 9801 7077 20 1 22 0.812 0.783 5 -5.598 0 0.0528 0.0478
30367 24079 360 47 0 16 0.701 0.942 5 -2.253 1 0.2410 0.1590
21503 18036 8203 47 3 23 0.565 0.905 6 -4.396 1 0.1490 0.1080
26263 5884 53 45 3 13 0.513 0.379 1 -13.587 0 0.1710 0.2590
In [ ]:
In [ ]:
# visualizing the distribution of outliers -
df_clean.to_csv('spotify.csv')
df_clean=pd.read_csv('spotify.csv')
DATA MODELING
In [ ]:
split into features (X) and target (Y)
40°F
Mostly cloudy
Search
ENG
US
7:11 PM
12/5/2022

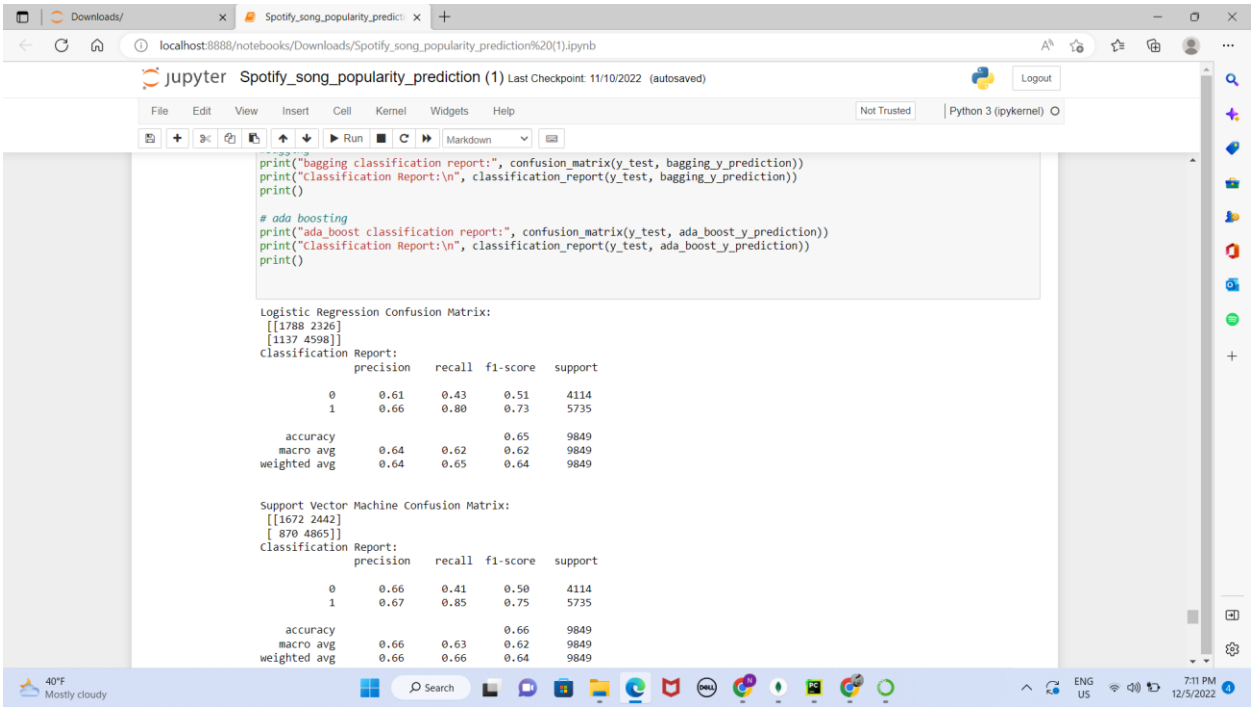

Downloads/
Spotify_song_popularity_predicti
localhost:8888/notebooks/Downloads/Spotify_song_popularity_prediction%20(1).ipynb
jupyter Spotify_song_popularity_prediction (1) Last Checkpoint: 11/10/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Markdown
In [ ]:
# visualizing the distribution of outliers -
df_clean.to_csv('spotify.csv')
df_clean=pd.read_csv('spotify.csv')
DATA MODELING
In [ ]:
# split into features (X) and target (Y)
X = df_clean.drop(['Likely_popular','track_id','track_popularity'], axis=1)
y = df_clean.Likely_popular
print(X.shape)
print(y.shape)
(32828, 18)
(32828,)
In [ ]:
# splitting into 75-25 training and test sets
from sklearn.model_selection import train_test_split as tts
X_train, X_test, y_train, y_test = tts(X, y, test_size = 0.3, random_state = 0)
In [ ]:
# scaling our features
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test= sc.transform(X_test)
In [ ]:
#fetching regressors
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
40°F
Mostly cloudy
Search
ENG
US
7:11 PM
12/5/2022

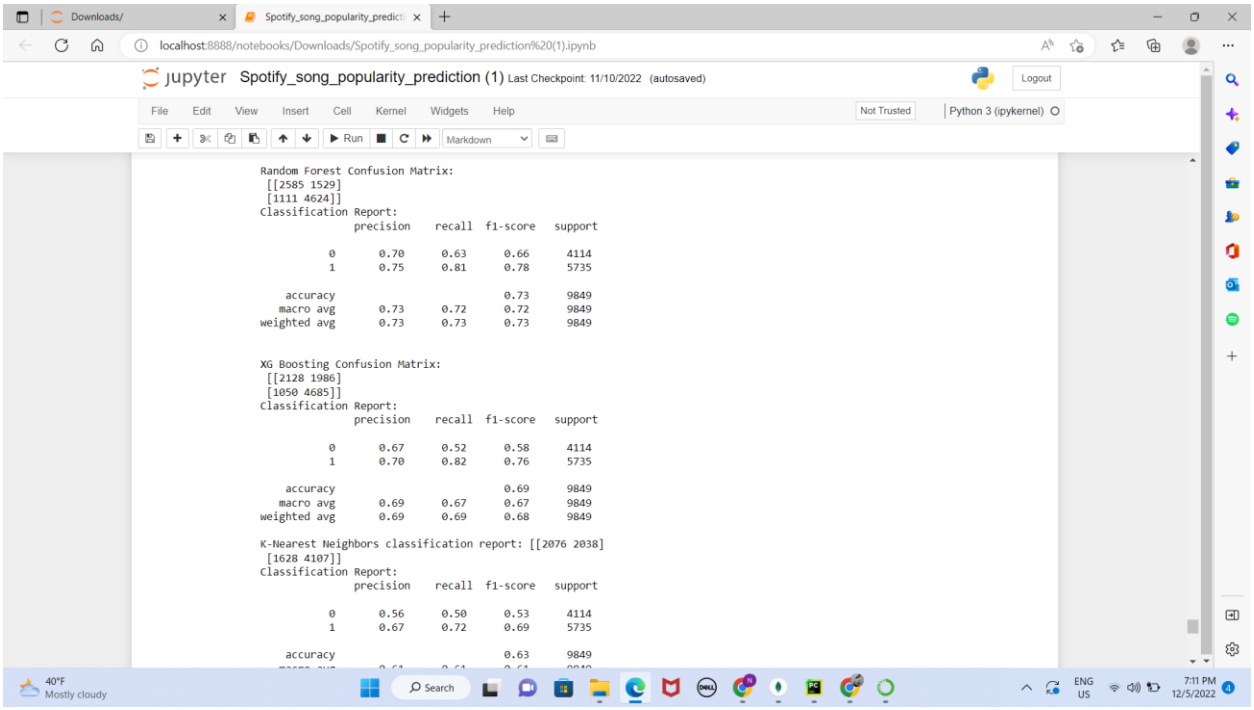

Downloads/
Spotify_song_popularity_predict
localhost:8888/notebooks/Downloads/Spotify_song_popularity_prediction%20(1).ipynb
jupyter Spotify_song_popularity_prediction (1) Last Checkpoint: 11/10/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Markdown
random_forest_classifier.fit(X_train, y_train)
random_forest_y_prediction = random_forest_classifier.predict(X_test)
# Ada Boosting
from sklearn.ensemble import AdaBoostClassifier
ada_boost_classifier = AdaBoostClassifier()
ada_boost_classifier.fit(X_train, y_train)
ada_boost_y_prediction = ada_boost_classifier.predict(X_test)
# Gradient Boosting
from sklearn.ensemble import GradientBoostingClassifier
g_boost_classifier = GradientBoostingClassifier()
g_boost_classifier.fit(X_train, y_train)
g_boost_y_prediction = g_boost_classifier.predict(X_test)
# XG Boosting
from xgboost import XGBClassifier
xg_boost_classifier = XGBClassifier()
xg_boost_classifier.fit(X_train, y_train)
xg_boost_y_prediction = xg_boost_classifier.predict(X_test)
In [ ]:
#Logistic Regression
log = LogisticRegression(solver = 'lbfgs', max_iter = 200)
log.fit(X_train, y_train)
log_pred = log.predict(X_test)
# Evaluation of the model
print('Logistic Regression Performance: ')
print('RMSE: ', np.sqrt(metrics.mean_squared_error(y_test, log_pred)))
print('R2: ', metrics.r2_score(y_test, log_pred))
Logistic Regression Performance:
RMSE: 0.5929665255615975
R2: -0.4455959385923156
40°F Mostly cloudy
Search
ENG US
7:11 PM 12/5/2022

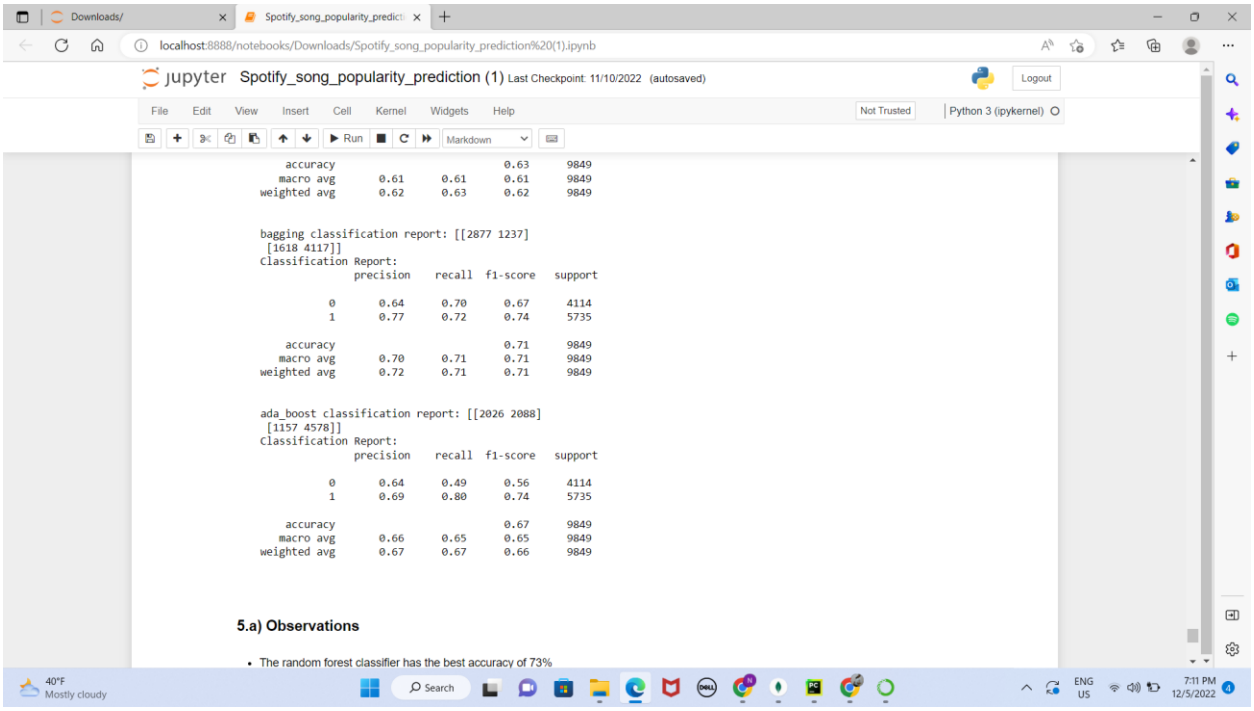

Downloads/
Spotify_song_popularity_predict
localhost:8888/notebooks/Downloads/Spotify_song_popularity_prediction%20(1).ipynb
jupyter Spotify_song_popularity_prediction (1) Last Checkpoint: 11/10/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Markdown
In [ ]:
# Support Vector Machine
svm = SVC(kernel = 'rbf')
svm.fit(X_train, y_train)
svm_pred = svm.predict(X_test)
# Evaluation of the model
print('Logistic Regression Performance: ')
print('RMSE: ', np.sqrt(metrics.mean_squared_error(y_test, svm_pred)))
print('R2: ', metrics.r2_score(y_test, svm_pred))
Logistic Regression Performance:
RMSE: 0.5798946410340153
R2: -0.38256244545704576
In [ ]:
# XG Boosting
from xgboost import XGBClassifier
xgb = XGBClassifier()
xgb.fit(X_train, y_train)
xgb_pred = xgb.predict(X_test)
# Evaluation of the model
print('Logistic Regression Performance: ')
print('RMSE: ', np.sqrt(metrics.mean_squared_error(y_test, xgb_pred)))
print('R2: ', metrics.r2_score(y_test, xgb_pred))
Logistic Regression Performance:
RMSE: 0.5552068489685973
R2: -0.2673489083356253
In [ ]:
# Random Forest
rf = RandomForestClassifier()
rf.fit(X_train, y_train)
40°F Mostly cloudy
Search
ENG US
7:11 PM 12/5/2022


Downloads/
Spotify_song_popularity_predicti
localhost:8888/notebooks/Downloads/Spotify_song_popularity_prediction%20(1).ipynb
jupyter Spotify_song_popularity_prediction (1) Last Checkpoint: 11/10/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Markdown
xgb_pred = xgb.predict(X_test)
# Evaluation of the model
print('Logistic Regression Performance: ')
print('RMSE: ', np.sqrt(metrics.mean_squared_error(y_test, xgb_pred)))
print('R2: ', metrics.r2_score(y_test, xgb_pred))
Logistic Regression Performance:
RMSE: 0.5552068489685973
R2: -0.2673489083356253
In [ ]:
# Random Forest
rf = RandomForestClassifier()
rf.fit(X_train, y_train)
rf_pred = rf.predict(X_test)
# Evaluation of the model
print('Logistic Regression Performance: ')
print('RMSE: ', np.sqrt(metrics.mean_squared_error(y_test, rf_pred)))
print('R2: ', metrics.r2_score(y_test, rf_pred))
Logistic Regression Performance:
RMSE: 0.5177330562311706
R2: -0.10204252898750021
In [ ]:
# evaluating the classification reports and confusion matrices of each classifier
from sklearn.metrics import classification_report, confusion_matrix
# Logistic Regression
print("Logistic Regression Confusion Matrix:\n", confusion_matrix(y_test, log_pred))
print("Classification Report:\n", classification_report(y_test, log_pred))
print()
# Support Vector Machine
40°F
Mostly cloudy
Search
ENG
US
7:11 PM
12/5/2022


Downloads/
Spotify_song_popularity_predicti
localhost:8888/notebooks/Downloads/Spotify_song_popularity_prediction%20(1).ipynb
jupyter Spotify_song_popularity_prediction (1) Last Checkpoint: 11/10/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Markdown
In [ ]:
# evaluating the classification reports and confusion matrices of each classifier
from sklearn.metrics import classification_report, confusion_matrix
# Logistic Regression
print("Logistic Regression Confusion Matrix:\n", confusion_matrix(y_test, log_pred))
print("Classification Report:\n", classification_report(y_test, log_pred))
print()
# Support Vector Machine
print("Support Vector Machine Confusion Matrix:\n", confusion_matrix(y_test, svm_pred))
print("Classification Report:\n", classification_report(y_test, svm_pred))
print()
# Random Forest
print("Random Forest Confusion Matrix:\n", confusion_matrix(y_test, rf_pred))
print("Classification Report:\n", classification_report(y_test, rf_pred))
print()
# XG Boosting
print("XG Boosting Confusion Matrix:\n", confusion_matrix(y_test, xgb_pred))
print("Classification Report:\n", classification_report(y_test, xgb_pred))
# classification report for K-Nearest Neighbors Classifier
print("K-Nearest Neighbors classification report:", confusion_matrix(y_test, knn_y_prediction))
print("Classification Report:\n", classification_report(y_test, knn_y_prediction))
print()
#bagging
print("bagging classification report:", confusion_matrix(y_test, bagging_y_prediction))
print("Classification Report:\n", classification_report(y_test, bagging_y_prediction))
print()
# ada boosting
print("ada_boost classification report:", confusion_matrix(y_test, ada_boost_y_prediction))
print("Classification Report:\n", classification_report(y_test, ada_boost_y_prediction))
40°F
Mostly cloudy
Search
ENG
US
7:11 PM
12/5/2022


Downloads/
Spotify_song_popularity_predicti
localhost:8888/notebooks/Downloads/Spotify_song_popularity_prediction%20(1).ipynb
jupyter Spotify_song_popularity_prediction (1) Last Checkpoint: 11/10/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Markdown
print("bagging classification report:", confusion_matrix(y_test, bagging_y_prediction))
print("Classification Report:\n", classification_report(y_test, bagging_y_prediction))
print()
# ada boosting
print("ada_boost classification report:", confusion_matrix(y_test, ada_boost_y_prediction))
print("Classification Report:\n", classification_report(y_test, ada_boost_y_prediction))
print()
Logistic Regression Confusion Matrix:
[[1788 2326]
[1137 4598]]
Classification Report:
precision recall f1-score support
0 0.61 0.43 0.51 4114
1 0.66 0.80 0.73 5735
accuracy 0.65 9849
macro avg 0.64 0.62 0.62 9849
weighted avg 0.64 0.65 0.64 9849
Support Vector Machine Confusion Matrix:
[[1672 2442]
[870 4865]]
Classification Report:
precision recall f1-score support
0 0.66 0.41 0.50 4114
1 0.67 0.85 0.75 5735
accuracy 0.66 9849
macro avg 0.66 0.63 0.62 9849
weighted avg 0.66 0.66 0.64 9849
40°F
Mostly cloudy
Search
ENG
US
7:11 PM
12/5/2022

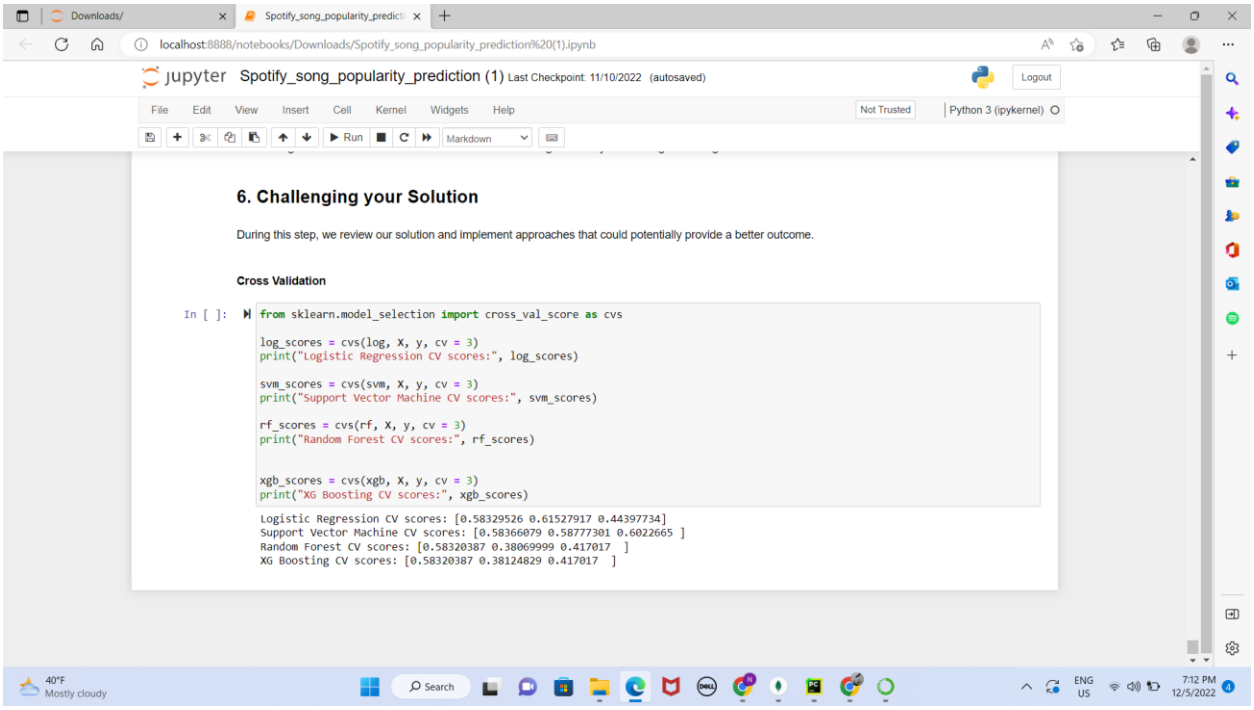

Downloads/
Spotify_song_popularity_predicti
localhost:8888/notebooks/Downloads/Spotify_song_popularity_prediction%20(1).ipynb
jupyter Spotify_song_popularity_prediction (1) Last Checkpoint: 11/10/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Markdown
Random Forest Confusion Matrix:
[[2585 1529]
[1111 4624]]
Classification Report:
precision recall f1-score support
0 0.70 0.63 0.66 4114
1 0.75 0.81 0.78 5735
accuracy 0.73 9849
macro avg 0.73 0.72 0.72 9849
weighted avg 0.73 0.73 0.73 9849
XG Boosting Confusion Matrix:
[[2128 1986]
[1050 4685]]
Classification Report:
precision recall f1-score support
0 0.67 0.52 0.58 4114
1 0.70 0.82 0.76 5735
accuracy 0.69 9849
macro avg 0.69 0.67 0.67 9849
weighted avg 0.69 0.69 0.68 9849
K-Nearest Neighbors classification report: [[2076 2038]
[1628 4107]]
Classification Report:
precision recall f1-score support
0 0.56 0.50 0.53 4114
1 0.67 0.72 0.69 5735
accuracy 0.63 9849
40°F
Mostly cloudy
Search
ENG
US
7:11 PM
12/5/2022



```
                accuracy                           0.63      9849
               macro avg       0.61      0.61      0.61      9849
            weighted avg       0.62      0.63      0.62      9849


bagging classification report: [[2877 1237]
 [1618 4117]]
Classification Report:
              precision    recall  f1-score   support

           0       0.64      0.70      0.67      4114
           1       0.77      0.72      0.74      5735

    accuracy                           0.71      9849
   macro avg       0.70      0.71      0.71      9849
weighted avg       0.72      0.71      0.71      9849


ada_boost classification report: [[2026 2088]
 [1157 4578]]
Classification Report:
              precision    recall  f1-score   support

           0       0.64      0.49      0.56      4114
           1       0.69      0.80      0.74      5735

    accuracy                           0.67      9849
   macro avg       0.66      0.65      0.65      9849
weighted avg       0.67      0.67      0.66      9849
```

5.a) Observations

- The random forest classifier has the best accuracy of 73%



6. Challenging your Solution

During this step, we review our solution and implement approaches that could potentially provide a better outcome.

Cross Validation

In [ ]:

```
from sklearn.model_selection import cross_val_score as cvs

log_scores = cvs(log, X, y, cv = 3)
print("Logistic Regression CV scores:", log_scores)

svm_scores = cvs(svm, X, y, cv = 3)
print("Support Vector Machine CV scores:", svm_scores)

rf_scores = cvs(rf, X, y, cv = 3)
print("Random Forest CV scores:", rf_scores)


xgb_scores = cvs(xgb, X, y, cv = 3)
print("XG Boosting CV scores:", xgb_scores)
```

```
Logistic Regression CV scores: [0.58329526 0.61527917 0.44397734]
Support Vector Machine CV scores: [0.58366079 0.58777301 0.6022665 ]
Random Forest CV scores: [0.58320387 0.38069999 0.417017  ]
XG Boosting CV scores: [0.58320387 0.38124829 0.417017  ]
```

GITHUB Link:
https://github.com/KiranKumarKongari/ML-Final_Project