Project - 2

**Uncovering surprising Facts from World Wide Movie Database using Data Cleaning & Data Visualization**

**OVERVIEW**

A project to overlook at the movie’s database and interpret various finding using Data cleaning, Data wrangling and Data Visualization

**Software Requirements**

1. Programming Language: Python

2. Environment: Jupyter Notebooks / Google Collab

3. Database: CSV(export type)

4. Operation System: Windows XP or above

5. Libraries Used: Pandas, Folium, Seaborn, Scikit, SKLEARN, Wordcount

6. Datasets used: TMDB Dataset

1. **Open a New Notebook and import the required libraires and read the csv file**

|  |  |
| --- | --- |
|  | import numpy as np  import pandas as pd  pd.set\_option('max\_columns', None)  import matplotlib.pyplot as plt  import seaborn as sns  %matplotlib inline  plt.style.use('ggplot')  import datetime  from scipy import stats  from wordcloud import WordCloud  from collections import Counter  from nltk.corpus import stopwords  from nltk.util import ngrams  import nltk  nltk.download('stopwords')  stop = set(stopwords.words('english'))  import os  import plotly.offline as py  py.init\_notebook\_mode(connected=True)  import plotly.graph\_objs as go  import plotly.tools as tls  from PIL import Image |

Description:

Import all the required libraries and renaming them with shorter names using as keyword in python.

1. **Loading the training & testing Dataset**

data = pd.read\_csv('/content/data.csv')

Description:

Reading the csv file named ‘data’ with read\_csv( )method whose input parameter is the file path. Here, the data file is stored in content folder which is present by default and storing it a variable called data.

1. **Visualizing the Distribution of Revenue with & without Log**

|  |  |
| --- | --- |
| fig, ax = plt.subplots(figsize=(16,6))  plt.subplot(1, 2, 1)  #kde=kernel density estimate  sns.distplot(data['revenue'], kde=False);  plt.title('Distribution of revenue');  plt.subplot(1, 2, 2)  #calculate the natural logarithmic value of x+1 where x belongs to all the input array elements.  sns.distplot(np.log1p(data['revenue']), kde=False);  plt.title('Distribution of log revenue')  Output:  Description: ­ Subplot( ) is used to plot multiple plots whose input parameters is size of each plot. i.e., height and width of the plot. The next subplot function plt.subplot(1,2,1) indicates there is one row and two columns and the first figure is being plotted. Since the data frame is stored in data variable we are plotting histogram distribution function with the help of seaborn library, by making kde false and plt.title( )is used to mention the title for the plot. Since the number of movies being very high the data is misinterpreted and for this reason logarithmic function is used to represent huge number of values in a comparatively smaller area without loss of information. Hence log values of revenue are plotted.   1. **Finding the Relationship between Movie Revenue & Budget**   data['log\_revenue'] = np.log1p(data['revenue'])  data['log\_budget'] = np.log1p(data['budget'])  plt.figure(figsize=(16, 8))  plt.subplot(1, 2, 1)  sns.scatterplot(data['budget'], data['revenue'])  plt.title('Revenue vs budget');  plt.subplot(1, 2, 2)  sns.scatterplot(data['log\_budget'], data['log\_revenue'])  plt.title('log transfromation of revenue vs budget');  Output: |  |

Description:

Just like distribution plot, another plot called scatter plot is plotted with its x- axis being budget and y- axis being revenue. For more appropriate plot log values are taken but still the plot is not very accurate since for zero budget revenue generated is 20 billion dollars which is impossible and this is called anomaly detection to get rid of this anomaly further statistical operations are required.

1. **Impact of Film’s Revenue with or without Homepage**

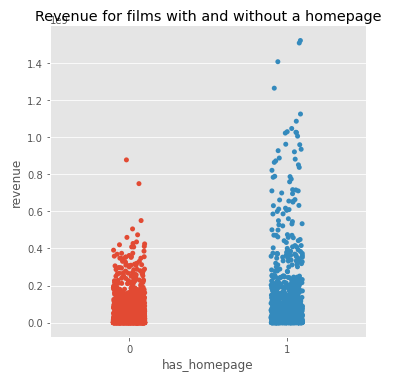
data['has\_homepage'] = 0

data.loc[data['homepage'].isnull() == False, 'has\_homepage'] = 1

sns.catplot(x='has\_homepage', y='revenue', data=data);

plt.title('Revenue for films with and without a homepage');

Output:



Description:

Categorical plot is plotted between revenue and homepage to find the influence by using a function in pandas. If a movie has homepage then we are terming it as 1 else we are terming it as 0, by checking availability of homepage of each movie in the homepage column if it is null we are assigning it 0 else we are assigning 1 to a data frame called has\_homepage. Taking x-axis as having a homepage or not and y axis as revenue. From the above plot it is evident that only in few cases having a homepage is influencing the revenue.

1. **Films Revenue in various Languages**

language\_data = data.loc[data['original\_language'].isin(data['original\_language'].value\_counts().head(10).index)]

plt.figure(figsize=(16,8))

plt.subplot(1, 2, 1)

sns.boxplot(x='original\_language', y = 'revenue', data=language\_data)

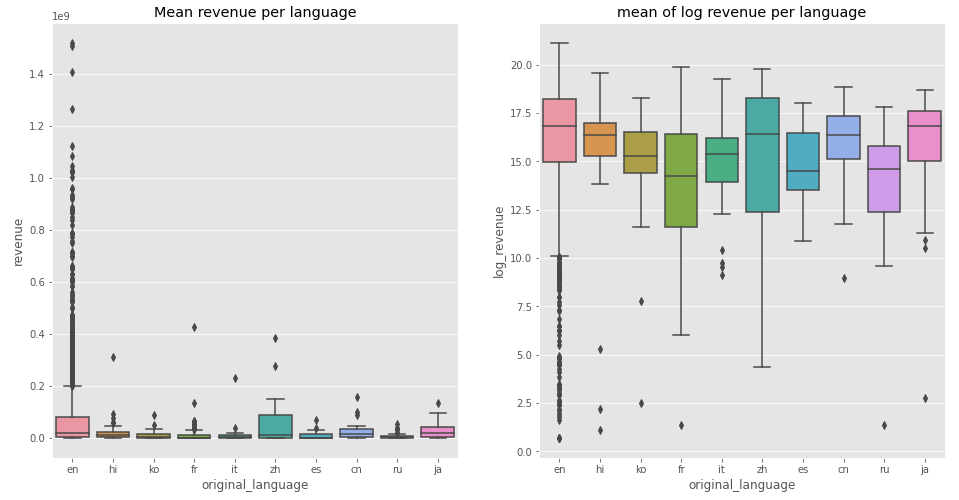
plt.title('Mean revenue per language')

plt.subplot(1, 2, 2)

sns.boxplot(x='original\_language', y = 'log\_revenue', data=language\_data)

plt.title('mean of log revenue per language')

Output:



Description:

When data is in large number for a particular category the box is used in which upper part of the plot is max value and lower part is minimum value and the median is indicated by a horizontal line at the centre of the box. Since the data frame contains many languages we have taken only 10 languages into consideration and taking revenue on y-axis and various languages box plot has been plotted. From graph it indicates the Zurich is able to get more revenue since its currency is greater. Though English language is widely spoken the revenue is not widely distributed. The values which are plotted away from the box are the anomalies. For better visualization log values have been plotted.

1. **Frequent Words in Movie Titles**

plt.figure(figsize=(12, 12))

text =  ' '.join(data['original\_title'].values)

wordcloud = WordCloud(max\_font\_size=None,

                     background\_color ='white',

                     width =1200, height =1000).generate(text)

plt.imshow(wordcloud)

plt.title('Most Frequent Words Used in Movie Titles')

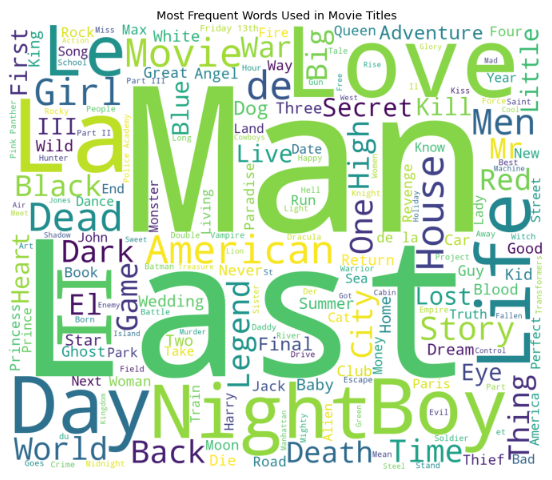
plt.axis('off')

plt.show()

Description:

Word Clouds are one of the most common ways to check for the events which have occurred frequently, since word cloud takes only text as input and not columns so we are joining each title one after the other with a space between each word and storing it in a variable text. Here max font is set to null which can be changed if required. plt.imshow( ) is used to display the word cloud generated. We can specify the title using plt.title( ). plt.axis( ) is set off since word cloud doesn’t have any axis and plt.show( ) is for printing the word cloud.

Output:



1. **Frequent Words in Movie Overviews**

plt.figure(figsize=(12, 12))

text =  ' '.join(train['overview'].fillna('').values)

wordcloud = WordCloud(max\_font\_size=None,

                     background\_color ='white',

                     width =1200, height =1000).generate(text)

plt.imshow(wordcloud)

plt.title('Top word across movie overviews')

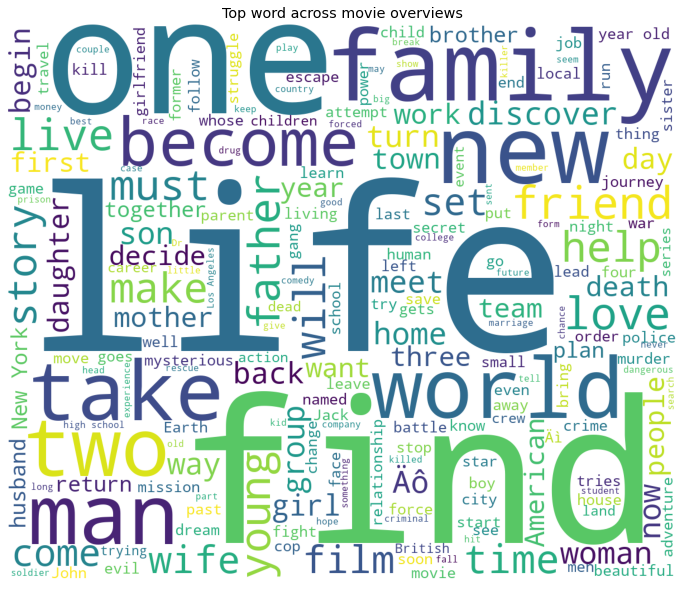
plt.axis('off')

plt.show()

Description:

Word cloud has been generated for overview of a movie.

Output:

****

**9. Most Common Genres**

plt.figure(figsize=(12, 12))

text =  ' '.join(data['all\_genres'].fillna('').values)

wordcloud = WordCloud(max\_font\_size=None,

                     background\_color ='black',

                     width =2000, height =1000).generate(text)

plt.imshow(wordcloud)

plt.title('Top genres across movie overviews')

plt.axis('off')

plt.show()

Description:

Word cloud for most common genres have been generated with black baground

Output:

