

Box office Data Analysis & Interpretation

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 libraries and read the csv file**

```

import numpy as np
import pandas as pd
import seaborn as s
import matplotlib.pyplot as plt
pd.set_option('max_columns', None)
%matplotlib inline
plt.style.use('ggplot')
plt.style.use('tableau-colorblind10')
import datetime
import pandas.util.testing as tm
from scipy import stats
from scipy.sparse import hstack, csr_matrix
from sklearn.model_selection import train_test_split, KFold
from wordcloud import WordCloud
from collections import Counter
import nltk
from nltk.corpus import stopwords
from nltk.util import ngrams
nltk.download('stopwords')
stop = set(stopwords.words('english'))
import os
import plotly.tools as tls
from sklearn.preprocessing import StandardScaler
from sklearn.feature_extraction.text import TfidfVectorizer, CountVectorizer
import json, ast
from urllib.request import urlopen
from PIL import Image

```

- ↵ NumPy is the fundamental package for scientific computing in Python. It is a Python library that provides a multidimensional array object, various derived objects and an assortment of routines for fast operations on arrays.
- ↵ pandas is a Python package providing fast, flexible, and expressive data structures designed to make working with relational or labeled data both easy and intuitive. pandas is well suited for many different kinds of data.
- ↵ Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.
- ↵ matplotlib. pyplot is a collection of functions that make matplotlib work like MATLAB. Each pyplot function makes some change to a figure: e.g., creates a figure, creates a plotting area in a figure.
- ↵ ggplot is a Python implementation of the grammar of graphics.
- ↵ %matplotlib inline sets the backend of matplotlib to the 'inline' backend: With this backend, the output of plotting commands is displayed inline within frontends like the Jupyter notebook, directly below the code cell that produced it.
- ↵ Datetime module supplies classes to work with date and time. These classes provide a number of functions to deal with dates, times and time intervals
- ↵ SciPy is a scientific computation library that uses NumPy underneath. SciPy stands for Scientific Python. It provides more utility functions for optimization, stats and signal processing.

- ↯ Sklearn and Model_selection is a Python library that offers various features for data processing that can be used for classification, clustering, and model selection.
- ↯ train_test_split is a function in Sklearn model selection for splitting data arrays into two subsets: for training data and for testing data.
- ↯ Wordcloud is a visual representation of text data. It displays a list of words, the importance of each being shown with font size or color.

The Natural Language Toolkit, or more commonly NLTK, is a suite of libraries and programs for symbolic and statistical natural language processing (NLP) for English written in the Python programming language

- ↯ plotly is an interactive, open-source plotting library that supports over 40 unique chart types covering a wide range of statistical, financial, geographic, scientific
- ↯ JavaScript Object Notation (JSON) is a standardized format commonly used to transfer data as text that can be sent over a network. JSON represents objects as name/value pairs, just like a Python dictionary.

2. Loading the training & testing Dataset

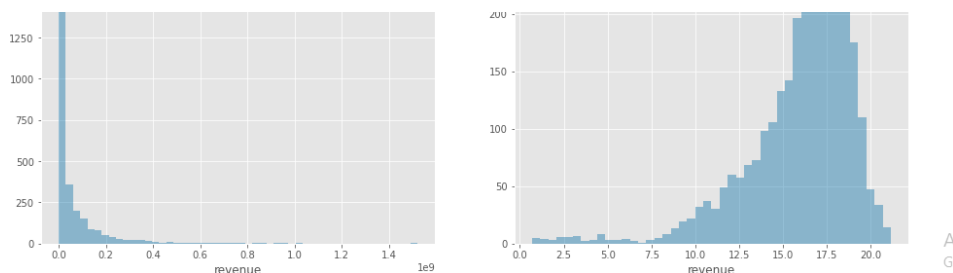
```
# Importing the dataset
import io
train = pd.read_csv('train.csv')
test = pd.read_csv('test.csv')
```

In this step we upload the train and test data set and read its data.

3. Visualizing the Distribution of Revenue with & without Log

```
fig, ax = plt.subplots(figsize=(16,6))
plt.subplot(1,2,1)
s.distplot(train['revenue'], kde= False);
plt.title('Distribution of Revenue');
plt.subplot(1,2,2)
s.distplot(np.log1p(train['revenue']), kde= False);
plt.title('Distribution of log transformed Revenue');
```

Output:



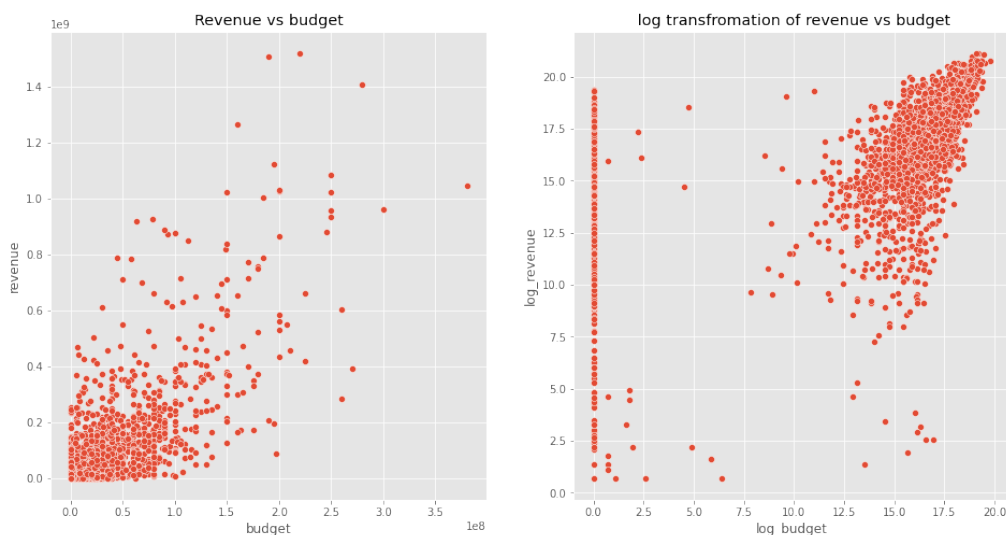
- ↳ subplots method provides a way to plot multiple plots on a single figure.
- ↳ Seaborn distplot shows a histogram with a line on it. We use seaborn in combination with matplotlib, the Python plotting module. A distplot plots a univariate distribution of observations.
- ↳ plt.title title() method in matplotlib module is used to specify title of the visualization depicted and displays the title using various attributes.

4. Finding the Relationship between Movie Revenue & Budget

```
train['log_revenue'] = np.log1p(train['revenue'])
train['log_budget'] = np.log1p(train['budget'])

plt.figure(figsize=(16, 8))
plt.subplot(1, 2, 1)
sns.scatterplot(train['budget'], train['revenue'])
plt.title('Revenue vs budget');
plt.subplot(1, 2, 2)
sns.scatterplot(train['log_budget'], train['log_revenue'])
plt.title('log transformation of revenue vs budget');
```

Output:



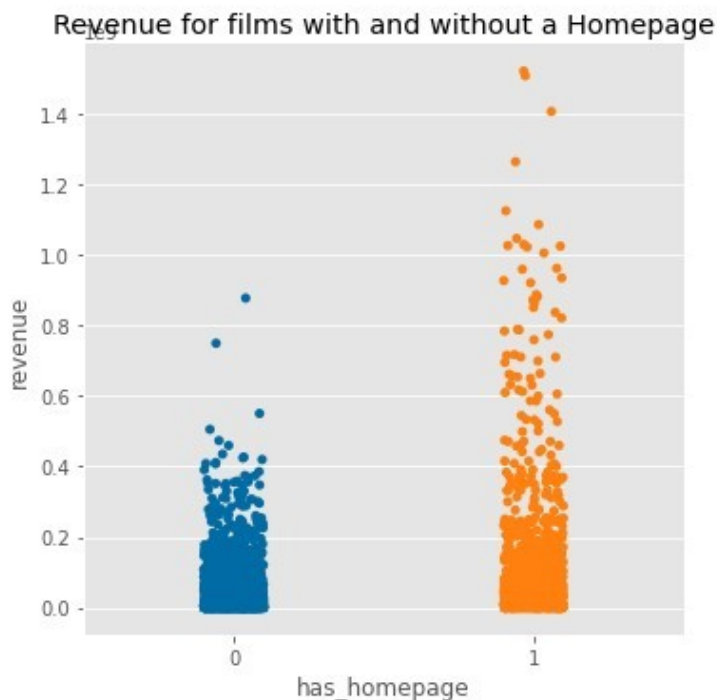
- `subplots()` without arguments returns a Figure and a single Axes
- `plt.scatterplot` will create a scatter plot for the budget and revenue from training dataset
- `plt.title` is used to set the title of the plot to budget v/s revenue
- `s.scatterplot(train['log_budget'], train['log_revenue'])`
- this will create a scatter plot for log budget and log revenue
- `plt.title('Log Budget v/s Log Revenue');`
- this will set the title as Log Budget v/s Log Revenue

5. Impact of Film's Revenue with or without Homepage

- for training dataset has_homepage value is equal to 0
- loc method to check and return a boolean value I.e. true or false if train['homepage'] is null.
- catplot shows frequencies (or optionally fractions or percents) of the categories of one, two or three categorical variables. it takes x and y axis and dataset as input
- plt.title is used to set title as Revenue for films with and without a homepage

```
test['has_homepage'] = 0  
test.loc[test['homepage'].isnull()== False, 'has_homepage'] = 1
```

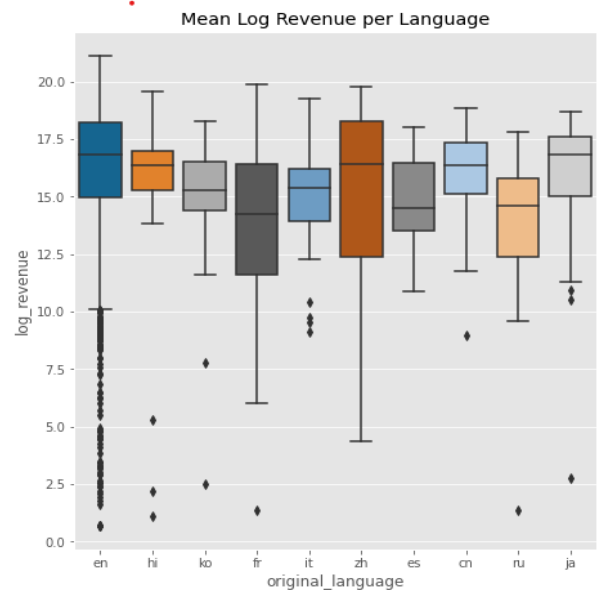
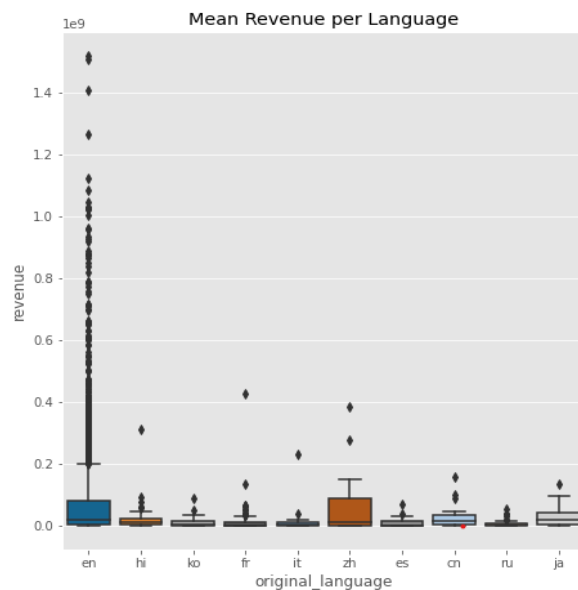
```
s.catplot(x= 'has_homepage' , y= 'revenue' , data= train)  
plt.title('Revenue for films with and without a Homepage');
```



6. Films Revenue in various Languages

```
language_data=train.loc[train['original_language'].isin(train['original_language'].value_counts().head(10).index)]
```

```
plt.figure(figsize=(16,8))
plt.subplot(1,2,1)
s.boxplot(x='original_language', y='revenue', data= language_data)
plt.title('Mean Revenue per Language');
plt.subplot(1,2,2)
s.boxplot(x='original_language', y='log_revenue', data= language_data)
plt.title('Mean Log Revenue per Language');
```




- plt.figure is used to get the figure of desired size.
- .We have to call plt.figure() explicitly.
- plt.subplot(1, 2, 1)
- The first two optional arguments of plt.subplots define the number of rows and columns of the subplot grid.
- s.boxplot(x='original_language', y = 'revenue', data=language_data)
- we find the boxplot of the data using s.boxplot and feed it In the data set.
- plt.title('Mean revenue per language')
- plt. title is used to set title of the plot
- s.boxplot(x='original_language', y='log_revenue', data= language_data)
- .boxplot(x='original_language', y = 'revenue', data=language_data)
- we find the boxplot of the data using s.boxplot and giving it x, y axis

names and feed it data set.

7. Frequent Words in Movie Titles

```
plt.figure(figsize= (12,12))
text= ' '.join(train['original_title'].values)
wordcloud= WordCloud(max_font_size= None,
                      background_color= 'white',
                      width= 1200, height=1000).generate(text)
plt.imshow(wordcloud)
plt.title('Top words across Movie Titles')
plt.axis('off')
plt.show()
```

- 
 - `plt.figure()` is used to mention the size of the plot.
 - `text = ' '.join(train['original_title'].values)`
 - we use `""`.join takes all the iterable and joins them into one here `original_title`.
 - `max_font` describes the maximum size of the wordcloud, `background_color` describes the background color and width and height describe the width and height of the wordcloud.
 - `plt.imshow()` displays the grayscale image in a figure.
 - `plt.title()` displays the title of the wordcloud.
 - `plt.show()` displays the wordcloud.

Conclusion:

We have successfully interpreted and analyzed box office data.
We also interpreted box graphs and analyzed clusters and found out relations between various attributes of the data.

