UDACITY

Return to Classroom

Investigate a Dataset

REVIEW

HISTORY

Meets Specifications

Excellent!

This is a great report, you state interesting questions that address important aspects of the data and your analysis allows you to produce excellent insights from the data. As you continue with the program forward, I encourage you to post more questions in the knowledge forum, which will help you and other students.

Please see my comments inside the review. If you have any further questions, please do not hesitate to post them in the knowledge forum.

Code Functionality

All code is functional and produces no errors when run. The code given is sufficient to reproduce the results described.

Code Functionality & Readability

The code is well-formatted and appropriately commented. That makes it easy to follow the analysis steps and identify a specific functional operation. A well-documented and easy-to-follow code will save you a lot of time when you need to debug it. If you like you can examine the python style document.
https://www.python.org/dev/peps/pep-0008/

Rules for Python variables Names,

I would like to encourage you to look into this link below, which discusses Rules and conventions for Python variable Names. This is important because that will allow other programmers, to better understand and follow your code. In many cases, you will be part of a team that will appreciate the clarity that these Rules and conventions provide.
https://www.w3schools.com/python/gloss_python_variable_names.asp

Python Comments

You can also look into this link that includes a discussion about python convention for code comments
https://www.w3schools.com/python/python_comments.asp

The project uses NumPy arrays and Pandas Series and DataFrames where appropriate rather than Python lists and dictionaries. Where possible, vectorized operations and built-in functions are used instead of loops.

Pandas and Numpy Operators

The analysis makes use of both single and multiple variable explorations to investigate different features and the relations between these features in the dataset. It is awesome that you demonstrate the use of pandas and NumPy, these libraries are widely used for data analysis and data manipulation.

Built In functions

It is awesome that you make use of the functions `.info()` and `.describe()` to examine the structure of the entire data, identify missing values, and the summary statistics for the numerical features.

- `DataFrame.groupby`: Allows you to aggregate the data according to specific categories: http://pandas.pydata.org/pandas-docs/stable/groupby.html
  For example, here I calculate different statistics for each category

```
df.groupby(['Sex' ])['Age'].median()
```

```
Sex
female    27.0
male      29.0
Name: Age, dtype: float64
```

```
df.groupby(['Sex' ])['Age'].mean()
```

```
Sex
female    27.915709
male      30.726645
Name: Age, dtype: float64
```

```
df.groupby(['Sex' ])['Age'].std()
```

```
Sex
female    14.110146
male      14.678201
Name: Age, dtype: float64
```

- `DataFrame.value_counts`: Return a Series containing counts of unique rows in the DataFrame
  https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.value_counts.html
- `pandas.cut`: This allows you to easily cut continuous variables into segments. https://pandas.pydata.org/docs/reference/api/pandas.cut.html
  Here I cut the age to several selected ranges and add a new column with this new information

```
df['Age_range' ] = pd.cut(x=df['Age' ], bins=[0, 20, 40, 60, 80, 100])
df['Age_range' ].sample(4)
```

```
592    (40, 60]
562    (20, 40]
599    (40, 60]
724    (20, 40]
Name: Age_range, dtype: category
Categories (5, interval[int64, right]): [(0, 20] < (20, 40] < (40, 60] < (60, 80] < (80, 100]]
```

The code makes use of at least 1 function to avoid repetitive code. The code contains good variable names that have meaning. Comments and docstrings are used as needed to document code functionality making it easy to read.

It is awesome that you created a custom function that reduces repetitions and simplifies the code.
Usually, the documentation should be inside the function, which allows you to use help and look into the docstring.
https://www.programiz.com/python-programming/docstrings
https://pythonprogramminglanguage.com/functions/

```
#Defines a function to get the averages of a column (col), returns the mean and median.
def averages(col):
    avgs_mean = df_movies_clean[col].mean()
    avgs_median = df_movies_clean[col].median()


    return avgs_mean, avgs_median
```

```
#These visualizations use the Seaborn package. These are called pairplots, which show the distribution of each variable and their relationships.

def popbudgrev_func(col):
    pop_budg_rev_list = df_movies_clean.columns[1:4]
    sns.pairplot(df_movies_clean[pop_budg_rev_list], hue=col, height=3.5);
```

Quality of Analysis

The project clearly states one or more questions, then addresses those questions in the rest of the analysis.

Project Introduction

The report states clear and relevant questions that are being addressed by the following analysis.

It will be very useful for your readers if you expand the introduction to discuss the analysis that you intend to implement.

> The questions we're going to explore:
>
> - Are movies with higher budgets more popular?
> - Do more popular movies have higher revenue?
> - Do movies with higher budgets have higher revenue?
> - Do certain directors have higher budgets?
> - Do certain directors influence revenue?
> - Is there a correlation between directors and movie popularity?
> - Are runtime and popularity correlated?

Exploration Phase

The project investigates the stated question(s) from multiple angles. At least three variables are investigated using both single-variable (1d) and multiple-variable (2d) explorations.

Single and Multiple Variable Explorations - Be Comprehensive

The analysis makes use of both single and multiple variable explorations to investigate different features and the relations between these features in the dataset.

The project's visualizations are varied and show multiple comparisons and trends. Relevant statistics are computed throughout the analysis when an inference is made about the data.
At least two kinds of plots should be created as part of the explorations.

Diversified visualizations

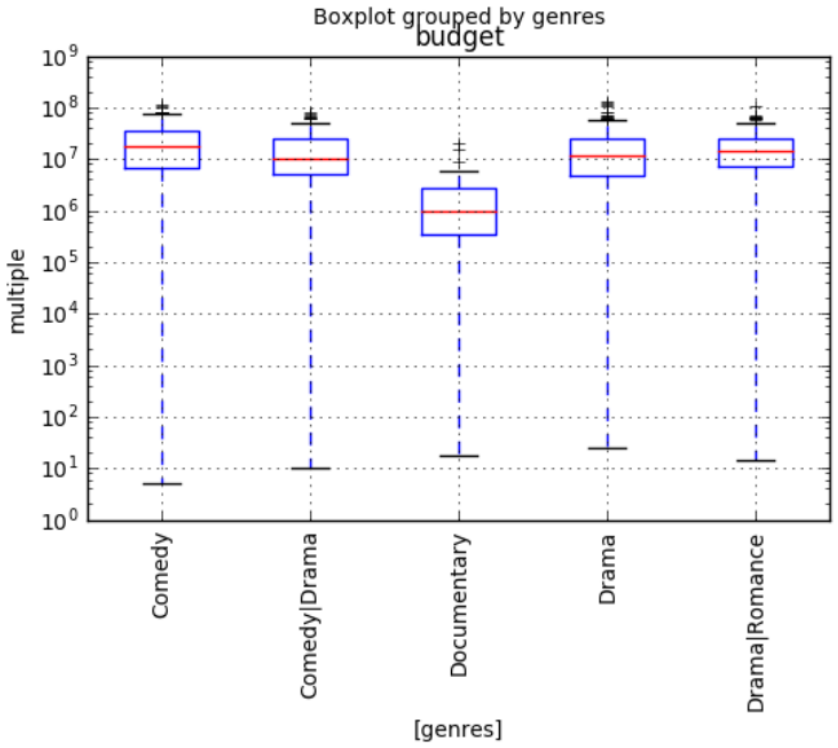
The report uses different chart type to explore and depict the insights and the results of the analysis. I strongly encourage you to include the relevant statistics next to each figure. Below I show a few examples of different chart types and the relevant descriptive statistics.

A simple box plot allows you to depict the distribution of a continuous feature for different categories,

```
# Sort the data acording to the sample numbers , sellecting the top 5
df_1=df.groupby(['genres' ])[['id']].count().sort_values(by=['id'], ascending=False)[0:5]
df_new = df[df['genres'].isin(df_1.index.values.tolist())]
# remove budget = 0
df_new1= df_new[df_new['budget']>0]

df_new1.boxplot(column=['budget'],by = ['genres'], rot=90).set_yscale('log')
plt.ylabel("multiple")
pd.DataFrame(df_new1.groupby( ['genres'])['budget'].describe().loc[:,['mean','std']])
```

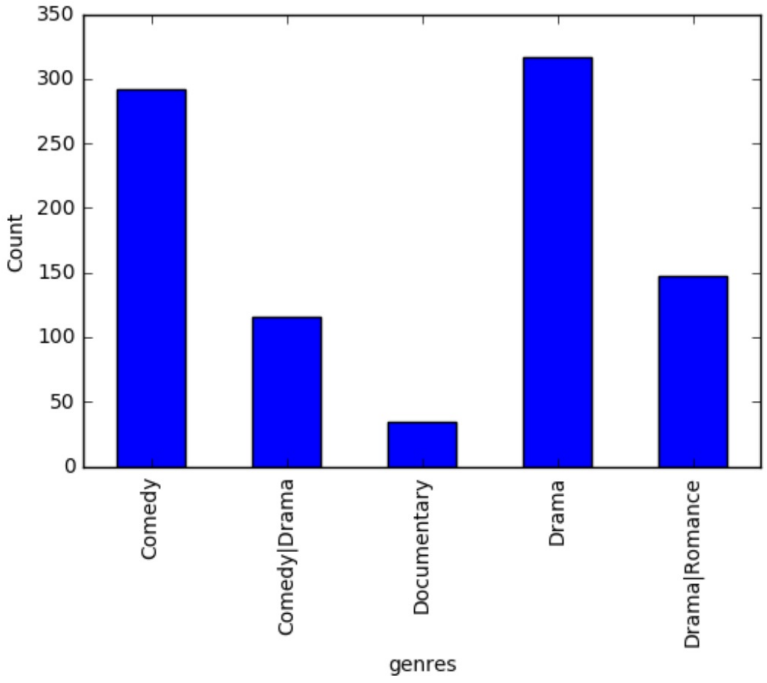
		budget
genres		
Comedy	**mean**	2.318910e+07
	std	2.145831e+07
Comedy|Drama	**mean**	1.731101e+07
	std	1.798798e+07
Documentary	**mean**	2.592782e+06
	std	4.251575e+06
Drama	**mean**	1.825644e+07
	std	1.986337e+07
Drama|Romance	**mean**	1.839379e+07
	std	1.740442e+07



A single variable bar plot depict the count distribution for categorical variable or Bivariate bar plot allow you to depict the ratio of one feature in different categories

```
df_new1.groupby([ 'genres'])['id'].count().plot(kind='bar').set_ylabel('Count')
df_new1.groupby(['genres' ])[['id']].count()
```

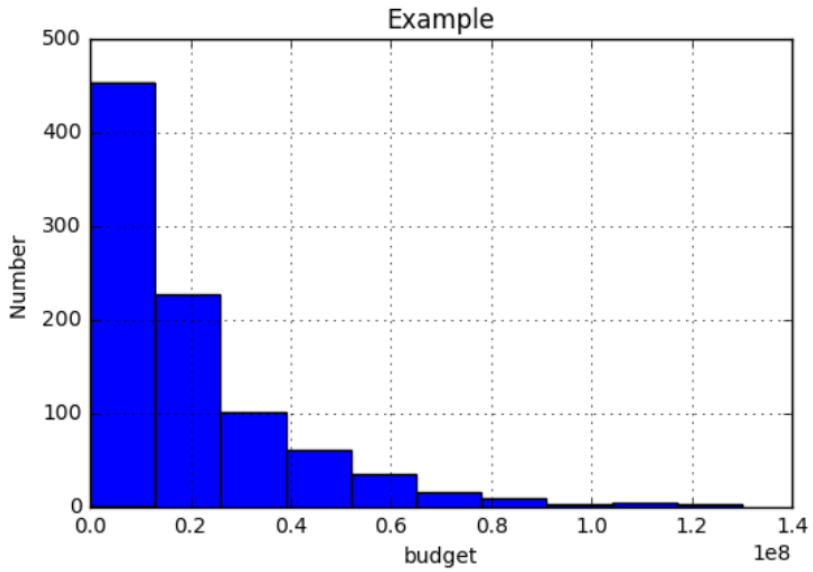
	id
genres	
Comedy	292
Comedy|Drama	116
Documentary	35
Drama	317
Drama|Romance	147



Histograms depict the distribution of continuous features.

```
ax = df_new1['budget'].hist()
ax.set_ylabel('Number ')
ax.set_xlabel('budget')
ax.set_title('Example')
pd.DataFrame(df_new1['budget'].describe())
```

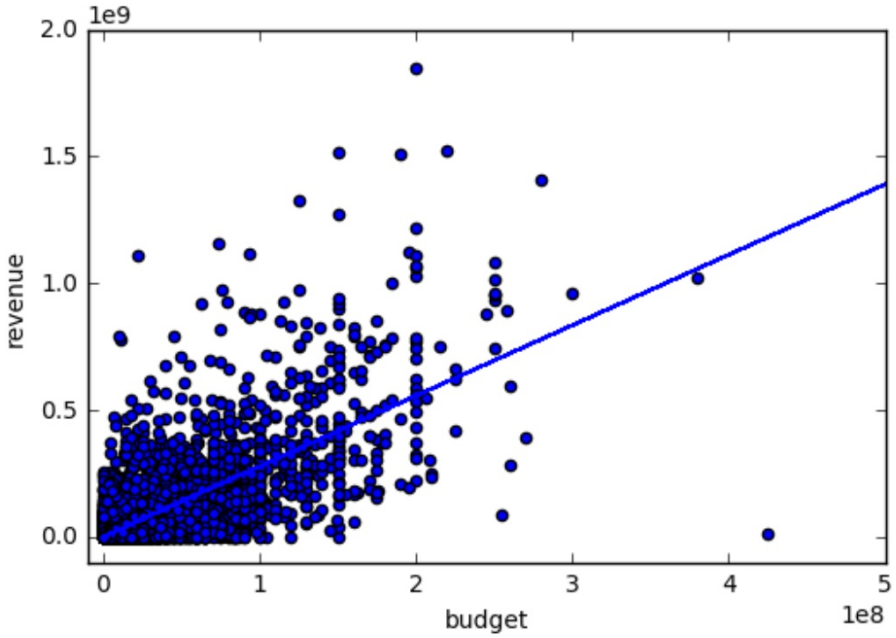
	budget
count	9.070000e+02
mean	1.914137e+07
std	1.981735e+07
min	5.000000e+00
25%	5.000000e+06
50%	1.350000e+07
75%	2.700000e+07
max	1.300000e+08



Scatter plots depict the relations between two continuous features.

```
import statsmodels.api as sm
import scipy
# regress "expression" onto "motifScore" (plus an intercept)
model = sm.OLS(df.revenue, sm.add_constant(df.budget))
p = model.fit().params
# generate x-values for your regression line (two is sufficient)
x = df.revenue
# scatter-plot data
ax = df.plot(x='budget', y='revenue', kind='scatter')
# plot regression line on the same axes, set x-axis limits
ax.plot(x, p.const + p.budget* x)
ax.set_xlim([-10000000,500000000])
ax.set_ylim([-100000000,2000000000])
print ("correlation :",scipy.stats.pearsonr(df.budget, df.revenue) )
```

```
correlation : (0.73490068190761171, 0.0)
```



Data Wrangling Phase

The project documents any changes that were made to clean the data, such as merging multiple files, handling missing values, etc.

Documenting Data Preparation - Be detailed and descriptive!

Well Done for reporting the missing values in the dataset and documenting the changes made in the dataset. This is important because it makes it possible for the readers to repeat your analysis if needed. Please note that for some of the columns, a major portion of the data is missing. That might affect the result of the analysis. Think about other ways to handle missing values.

Conclusions Phase

The results of the analysis are presented such that any limitations are clear. The analysis does not state or imply that one change causes another based solely on a correlation.

Analysis Shortcoming & Data Limitations

Excellent! The report includes a discussion about the limitations and shortcomings of the analysis and the dataset.

Communication

The reasoning is provided for each analysis decision, plot, and statistical summary.

Analysis Description

The analysis follows a logical flow, the discussion includes reasonings, explanations about the analysis, and relevant statistics to quantify the results and insights.

For the scatter plots, please choose a smaller dot size and lower alpha to lower the overplotting.

Only for Project Reviewers (No student work needed)

This rubric is ungraded. The reviewer will provide the student a code review.

This rubric will be ungraded. The reviewer will brief the students about the concepts learned in this section of the Nanodegree program.

I would just like to mention the data analysis process, usually, we start by obtaining the data and reading that into the computer.
The first step is the cleaning of the data, I will use the function info to identify the missing values. And then next I will use a histogram or bar plot to examine the structure of the data.

After we are familiar with the data we will start looking for relations in the data, which should be pursued first by visualization and next using statistical tests to verify if the changes we see in the visualizations are indeed significant.

As we go over these steps it is important to document and explain each step. don't assume that your reader knows python, instead explain in plain English each step of the analysis the methods that you are using and the results that you obtained.

This rubric is ungraded. If the learner has asked a question pertaining to the implementation of the project, the reviewer will provide an answer along with links to any helpful resources.