Python Data Analyst (Code Script) – JuypterNotebook

**Code Shell 1**

# This project aims to analyse the correlation between revenue and each of the respective columns. The hypothesis:

# Budget, Company will have hogh correlations as well as date

import numpy as np

import pandas as pd

import seaborn as sns

import matplotlib

import matplotlib.pyplot as plt

plt.style.use('ggplot')

from matplotlib.pyplot import figure

%matplotlib inline

matplotlib.rcParams['figure.figsize'] = (12, 8) # This adjusts the configuration of the plots I'm going to create

#This line of code will allow me to read in the data

df = pd.read\_csv(r'\\Client.barclayscorp.com\Homeshare\HomeUK0055\G01317496\My Documents\movies.csv')

df.head()

**Code Shell #2**

# Identifying missing data in dataset

for col in df.columns:

pct\_missing = np.mean(df[col].isnull())

print('{} - {}'.format(col, pct\_missing))

**Code Shell #3**

# Examining the datatypes for each column

df.dtypes

**Code Shell #4**

# changed originally to 'Int64' but changes back to float for regression line functionality

df['budget'] = df['budget'].astype('float')

df['gross'] = df['gross'].astype('float')

df.head()

**Code Shell #5**

# Year column inconsistent with releases column. Will take year from released column and make a new column.

# Had to edit data in excel to remove date monnth

df['year\_correction'] = df['released2'].astype(str).str[:5]

df.head()

**Code Shell #6**

# Sort columns by gross revenue

df.sort\_values(by=['gross'], inplace=True, ascending=False)

df.head()

**Code Shell #7**

# Code to remove built in limiter on df rows

pd.set\_option('display.max\_rows', 30)

df

**Code Shell #8**

# Drop duplicates

df['company'].drop\_duplicates().sort\_values(ascending=False)

**Code Shell #9**

# Had error as null values were being passed through float. code to convert null to 0 for for numeric columns

df['gross']=df['gross'].fillna(0)

df['budget']=df['budget'].fillna(0)

**Code Shell #10**

#Build scatterplot to compare budget and revenue, this will show the correlation between budget and revenue

plt.scatter(x=df['budget'], y=df['gross'])

plt.title('Budget vs Revenue')

plt.xlabel('Budget')

plt.ylabel('Revenue')

plt.show()

**Code Shell #11**

# This line shows that there is positive regression between budget and gross. I have edited the colour of the line to make

# the line more clear

sns.regplot(x='budget', y='gross', data=df, scatter\_kws={"color":"black"},line\_kws={"color":"purple"})

**Code Shell #12**

# Line of code to establish what the correlation is between budget, revenue and other columns however cannot include strings

# in calculation

df.corr() #Pearson correlation

df.corr(method = 'kendall') #kendal correlation

df.corr(method = 'spearman') #spearman

**Code Shell #13**

#Placing correlation matrix into visualtion, heatmap visualisation

correlation\_matrix = df.corr(method = 'pearson')

sns.heatmap(correlation\_matrix, annot = True)

plt.title('Correlation Matrix for Numeric Features')

plt.xlabel('Movie Features')

plt.ylabel('Movie Features')

plt.show

**Code Shell #14**

#Analysing correlations betwee non-numenrical values, by 'numerizing' the data values

df\_numerized = df

for col\_name in df\_numerized.columns:

if (df\_numerized[col\_name].dtype == 'object'): #Checks the datatype of the value

df\_numerized[col\_name] = df\_numerized[col\_name].astype('category') #will convert datavalue to categore if object

df\_numerized[col\_name] = df\_numerized[col\_name].cat.codes #this part of the code will give the non-numeric values random values

df\_numerized

**Code Shell #15**

# Creating correlation matrix for each column

correlation\_matrix = df\_numerized.corr(method = 'pearson')

sns.heatmap(correlation\_matrix, annot = True)

plt.title('Correlation Matrix for Numeric Features')

plt.xlabel('Movie Features')

plt.ylabel('Movie Features')

plt.show

**Code Shell #16**

# Code line to see most correlated columns more visibly in correlation matrix using unstacking, also increasing number of rows

pd.set\_option('display.max\_rows', None)

df

correlation\_mat = df\_numerized.corr()

corr\_pairs = correlation\_mat.unstack()

corr\_pairs

**Code Shell #17**

# Further sorting the unstacked datapoints

sorted\_pairs = corr\_pairs.sort\_values()

sorted\_pairs

**Code Shell #18**

# Looking at pairs with high correlation

high\_corr = sorted\_pairs[(sorted\_pairs) > 0.5]

high\_corr

**Code Shell #18**

**#** In conclusion, the votes had the highest correlation to budget and gross revenue showing our initial assumption was inaccurate