MIT Academy of Engineering

python

EDS PROJECT ON:

Exploring the Movie Data

Presented by:

Guide by: Madhavi Minkar

650 - Vedant Swami
647 - Manoj Pandit
650 - Vedant Pawar
658 - Om Suryawanshi

INTRODUCTION

- Data analytics is the process of examining vast volumes of data to extract meaningful patterns, trends, and correlations.

- In the context of the movie_ dataset, data analysis becomes a window through which we can do various analysis such as data manipulation, data visualization ,etc.

- By applying robust data analysis techniques, we aim to uncover the factors that influenced survival rates, understand the demographics of the passengers, and reveal intriguing correlations within the dataset

MOTIVATION

Do you like movies? We do too! When working with our data science & analysis students, we like to use datasets that everyone can relate to – because it makes learning more fun! In this data analysis example, you will analyze a dataset of movie ratings to draw various conclusions. You will learn how to:

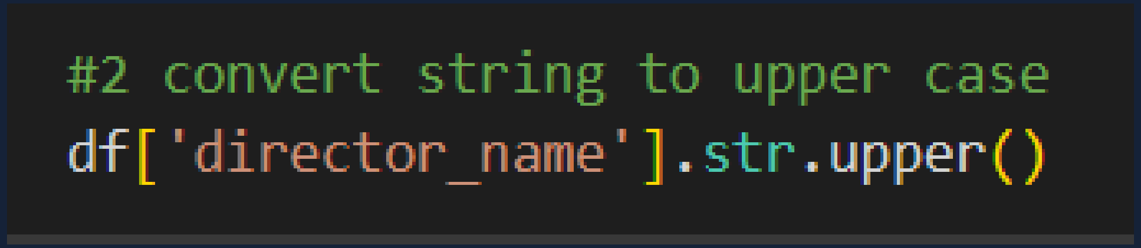
- Get and Clean the data
- Get the overall figures and basic statistics with their interpretation
- Join datasets, aggregate and filter your data by conditions
- Discover hidden patterns and insights
- Create summary tables

DETAIL OF DATASET

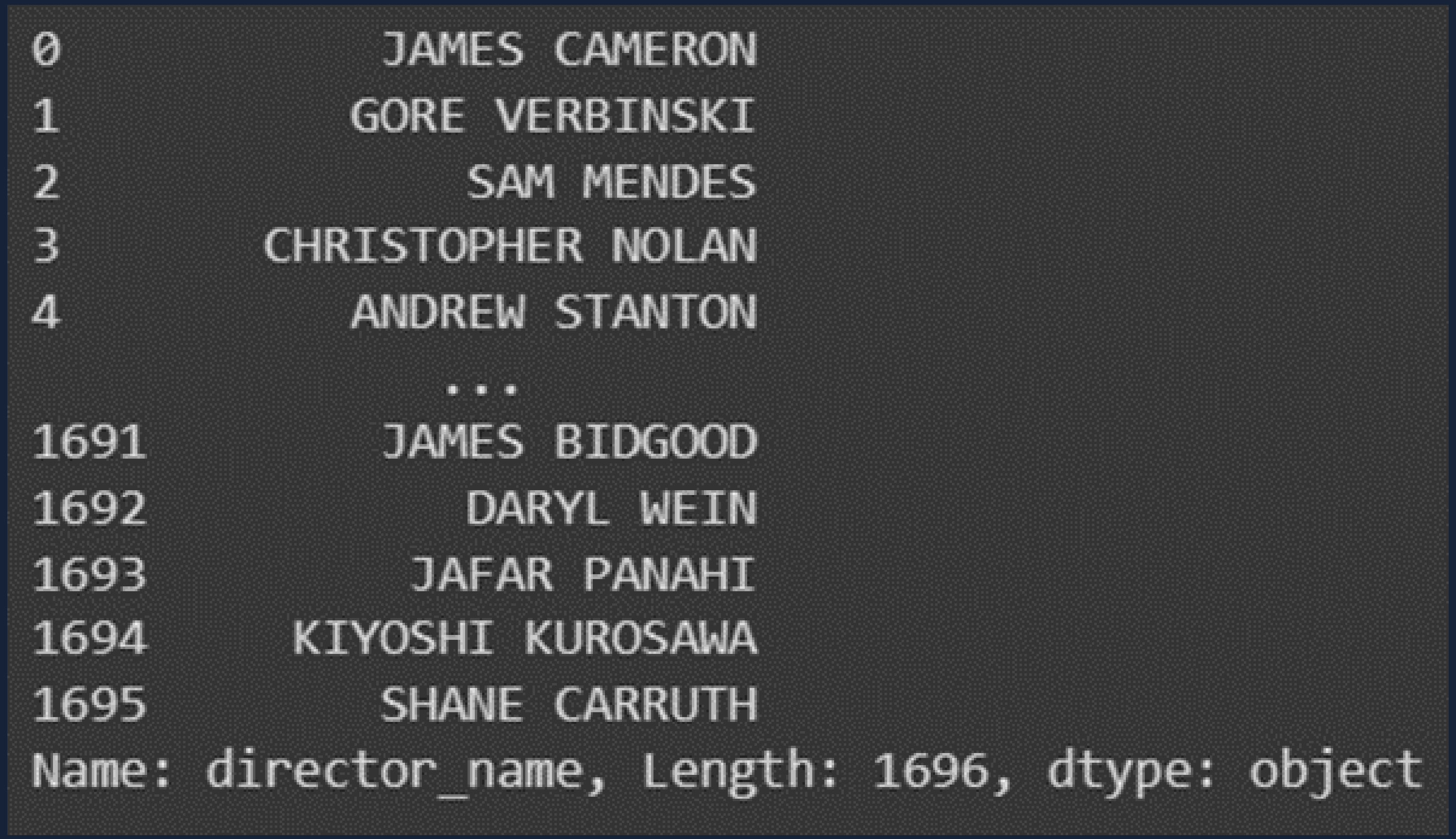
- Name: Movie Dataset
- Number of features: 16
- Number of records: 5000

DATA MANIPULATION

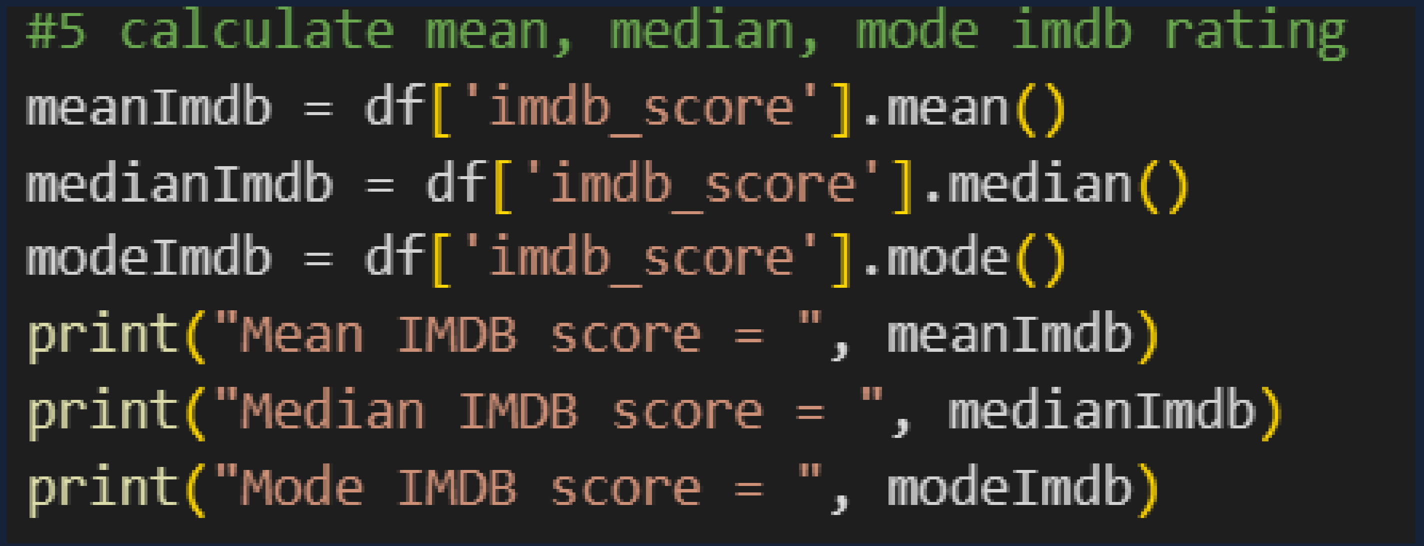
Data manipulation is a fundamental process in data analysis that involves transforming and preparing raw data to make it suitable for further exploration and analysis. It encompasses a range of operations aimed at ensuring data quality, consistency, and usability. Missing values can be imputed or removed, while outliers can be addressed through various methods such as transformation



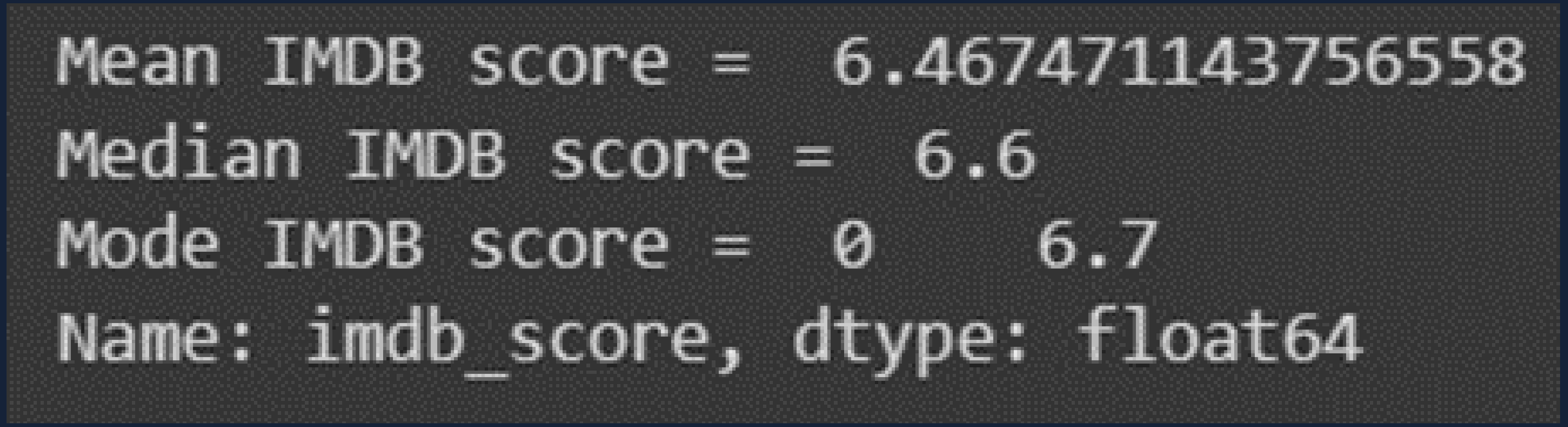
```
#2 convert string to upper case
df['director_name'].str.upper()
```



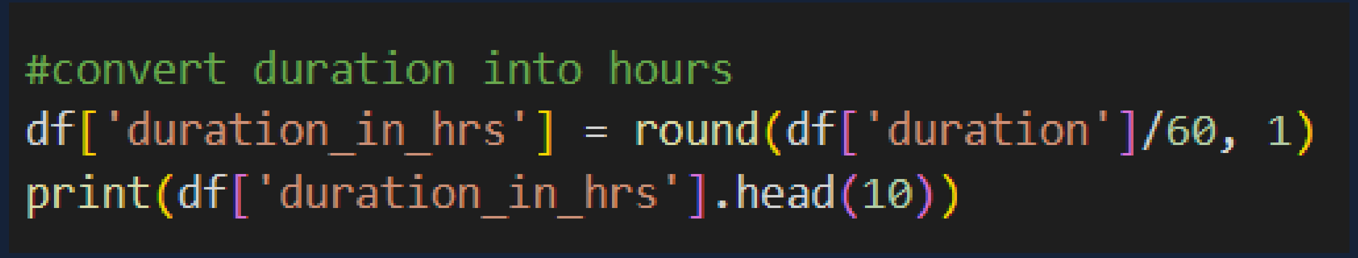
```
0            JAMES CAMERON
1           GORE VERBINSKI
2               SAM MENDES
3        CHRISTOPHER NOLAN
4           ANDREW STANTON
                ...
1691         JAMES BIDGOOD
1692            DARYL WEIN
1693          JAFAR PANAHI
1694      KIYOSHI KUROSAWA
1695          SHANE CARRUTH
Name: director_name, Length: 1696, dtype: object
```



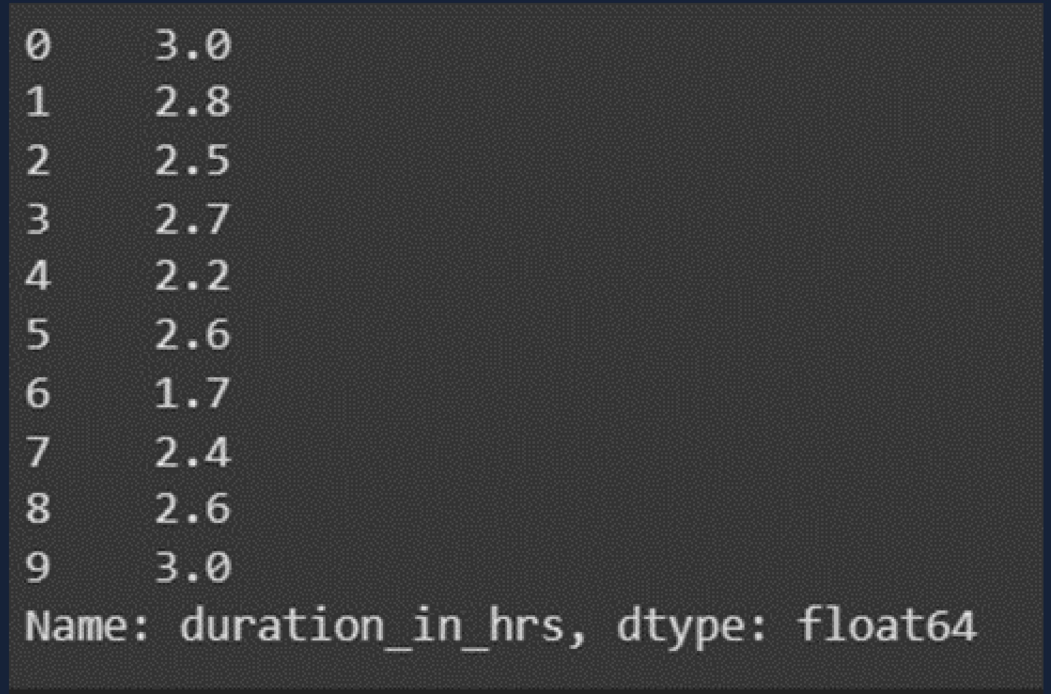
```
#5 calculate mean, median, mode imdb rating
meanImdb = df['imdb_score'].mean()
medianImdb = df['imdb_score'].median()
modeImdb = df['imdb_score'].mode()
print("Mean IMDB score = ", meanImdb)
print("Median IMDB score = ", medianImdb)
print("Mode IMDB score = ", modeImdb)
```



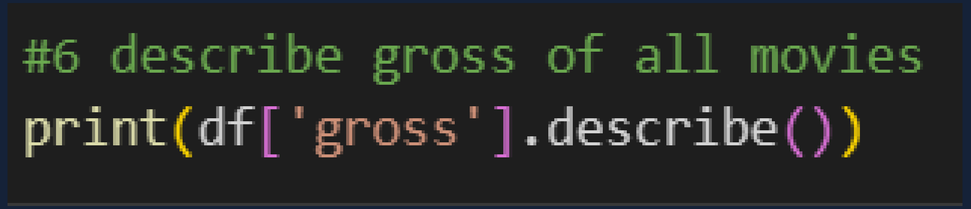
```
Mean IMDB score =  6.467471143756558
Median IMDB score =  6.6
Mode IMDB score =  0    6.7
Name: imdb_score, dtype: float64
```



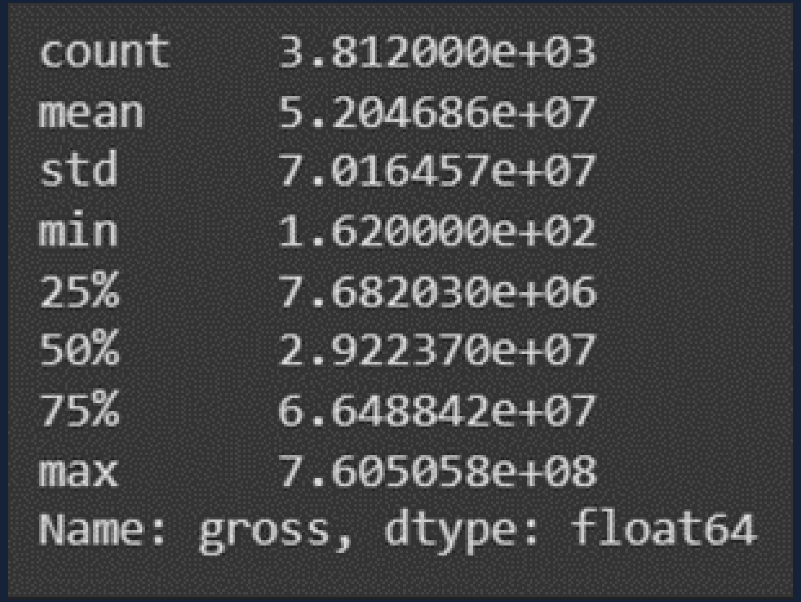
```
#convert duration into hours
df['duration_in_hrs'] = round(df['duration']/60, 1)
print(df['duration_in_hrs'].head(10))
```



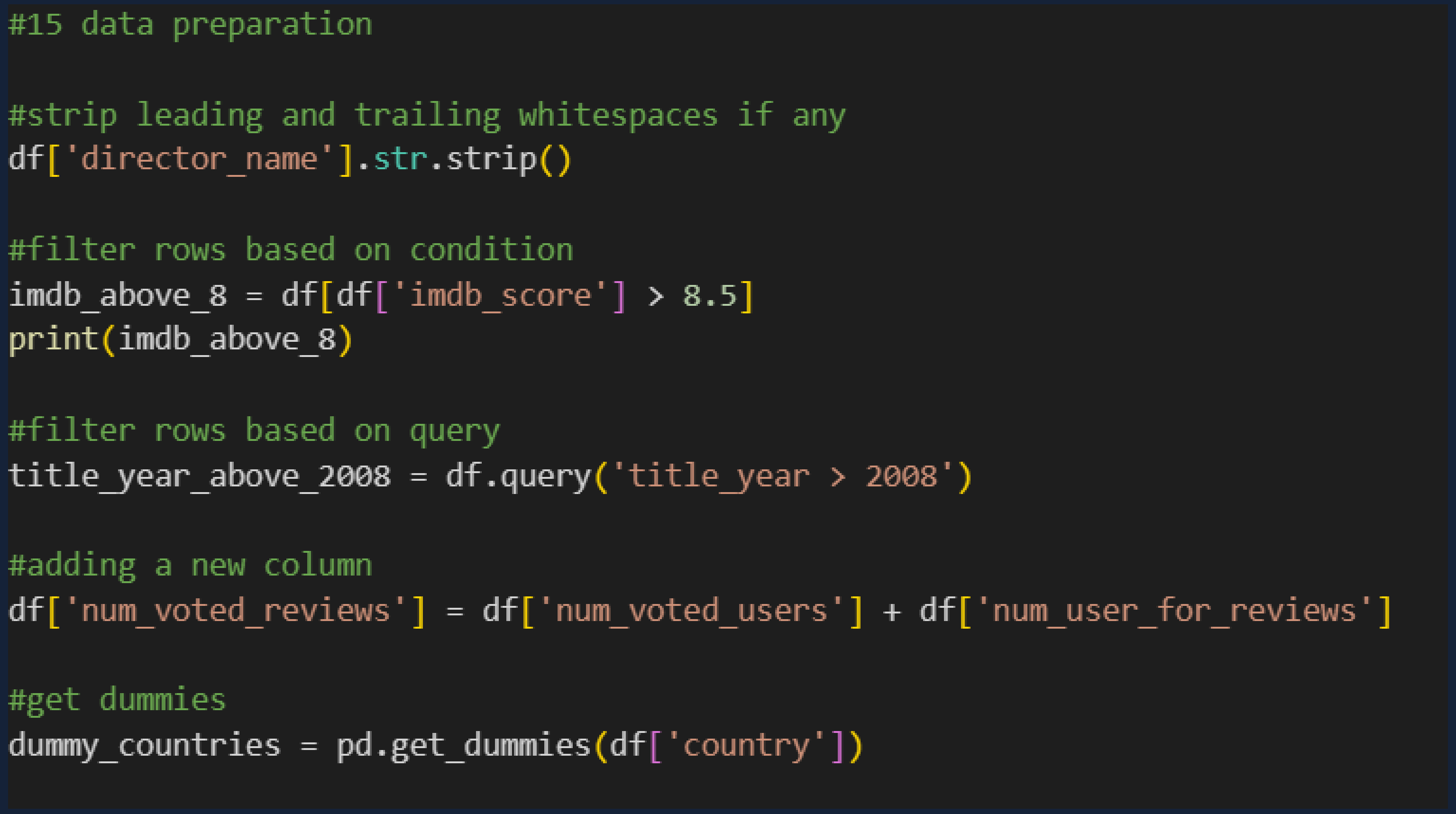
```
0    3.0
1    2.8
2    2.5
3    2.7
4    2.2
5    2.6
6    1.7
7    2.4
8    2.6
9    3.0
Name: duration_in_hrs, dtype: float64
```



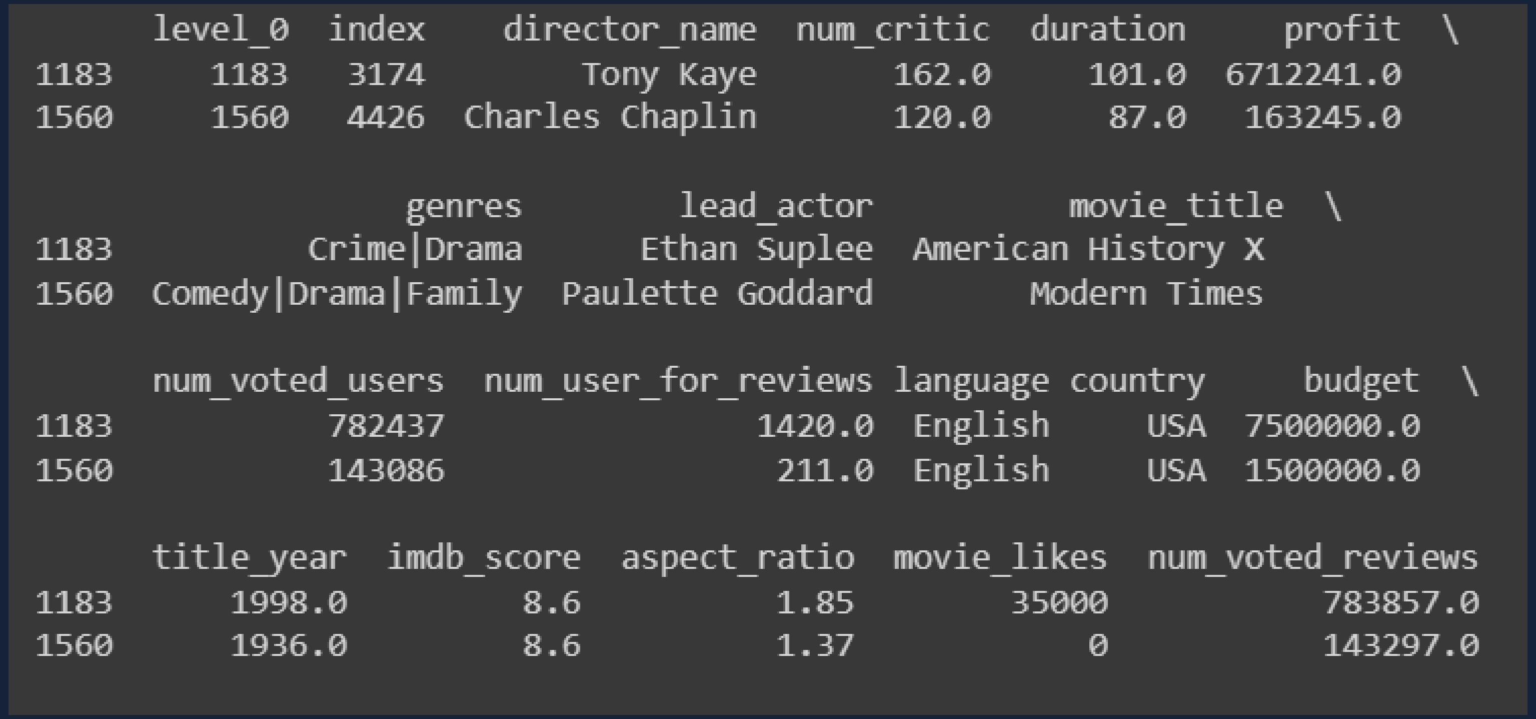
```
#6 describe gross of all movies
print(df['gross'].describe())
```



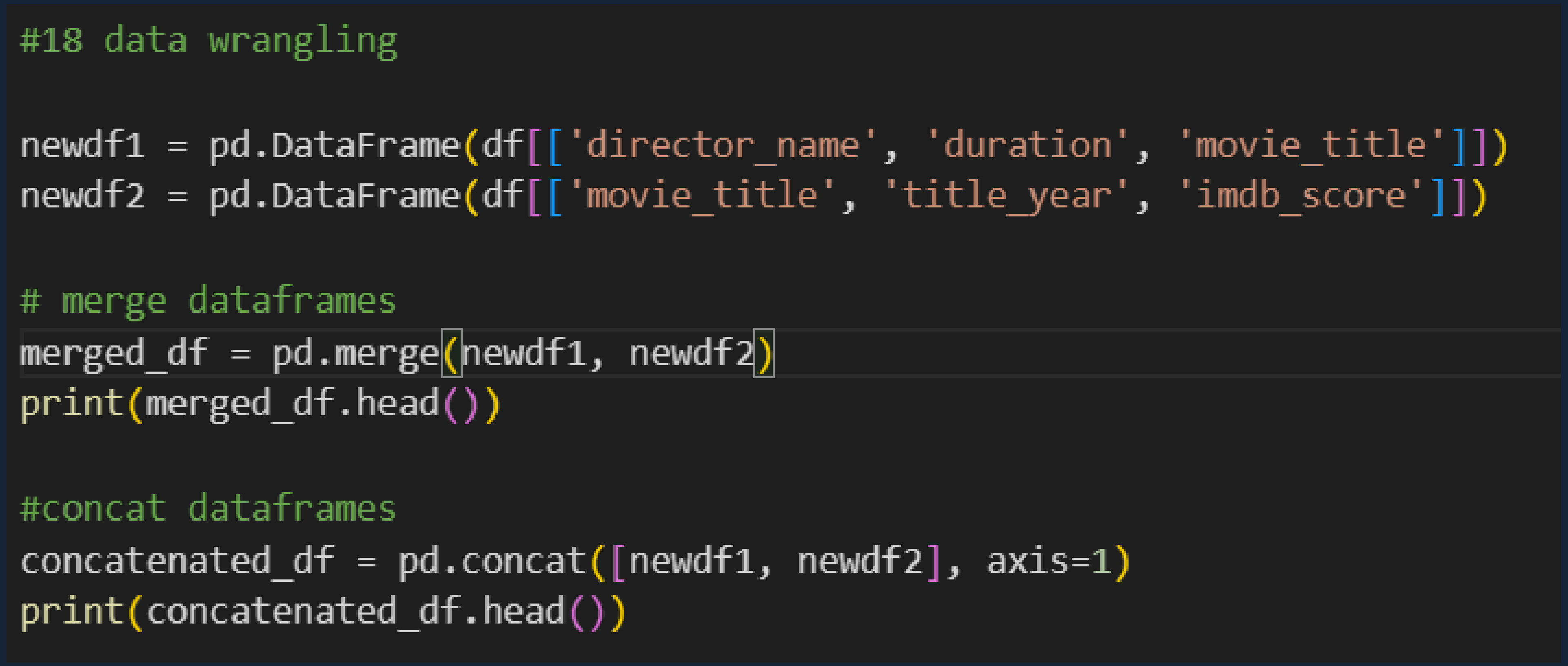
```
count    3.812000e+03
mean     5.204686e+07
std      7.016457e+07
min      1.620000e+02
25%      7.682030e+06
50%      2.922370e+07
75%      6.648842e+07
max      7.605058e+08
Name: gross, dtype: float64
```



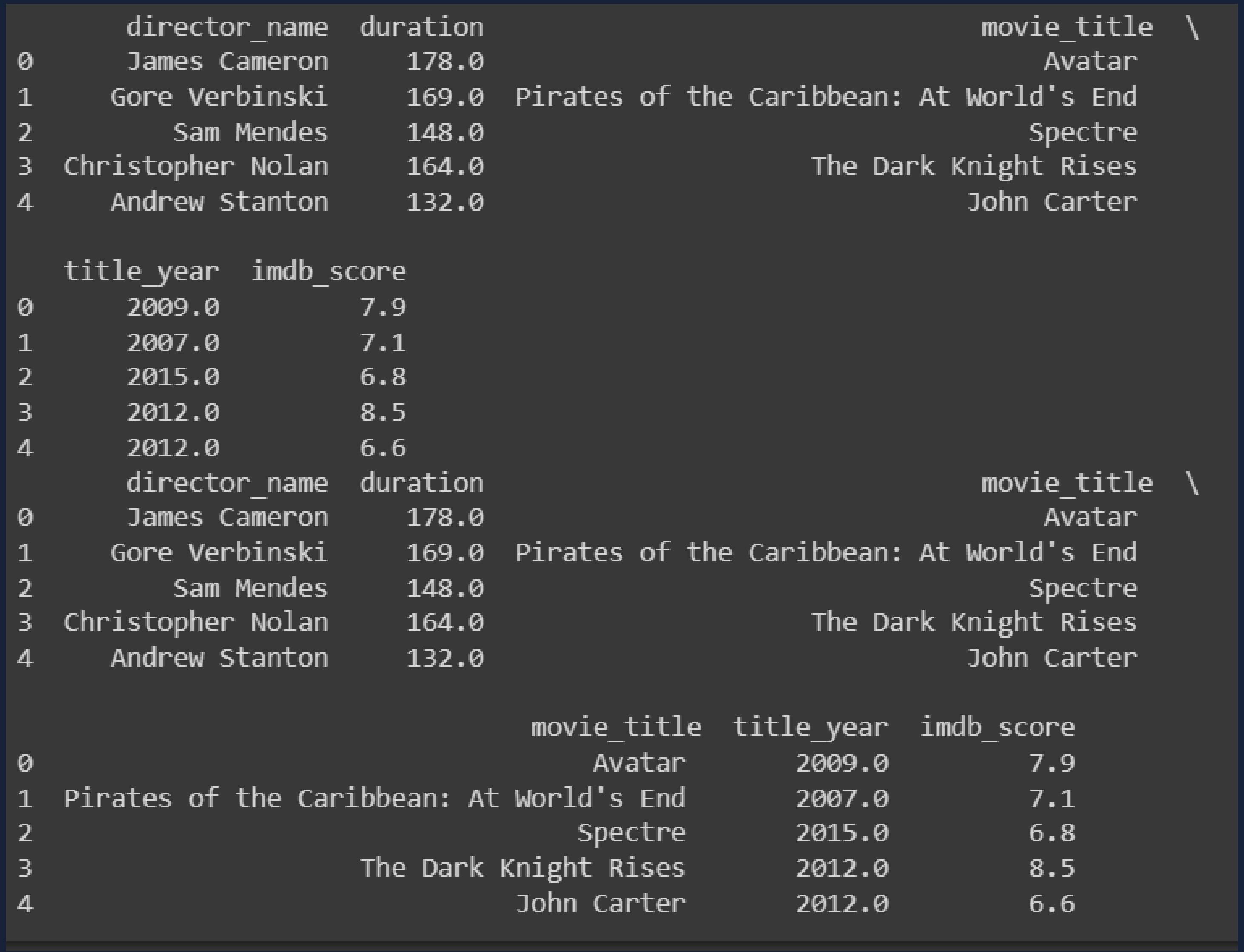
```
#15 data preparation

#strip leading and trailing whitespaces if any
df['director_name'].str.strip()

#filter rows based on condition
imdb_above_8 = df[df['imdb_score'] > 8.5]
print(imdb_above_8)

#filter rows based on query
title_year_above_2008 = df.query('title_year > 2008')

#adding a new column
df['num_voted_reviews'] = df['num_voted_users'] + df['num_user_for_reviews']

#get dummies
dummy_countries = pd.get_dummies(df['country'])
```



```
      level_0  index    director_name  num_critic  duration     profit  \
1183     1183   3174        Tony Kaye       162.0     101.0  6712241.0
1560     1560   4426  Charles Chaplin       120.0      87.0   163245.0

                   genres        lead_actor         movie_title  \
1183          Crime|Drama      Ethan Suplee  American History X
1560  Comedy|Drama|Family  Paulette Goddard        Modern Times

      num_voted_users  num_user_for_reviews language country     budget  \
1183           782437                1420.0  English     USA  7500000.0
1560           143086                 211.0  English     USA  1500000.0

      title_year  imdb_score  aspect_ratio  movie_likes  num_voted_reviews
1183      1998.0         8.6          1.85        35000           783857.0
1560      1936.0         8.6          1.37            0           143297.0
```



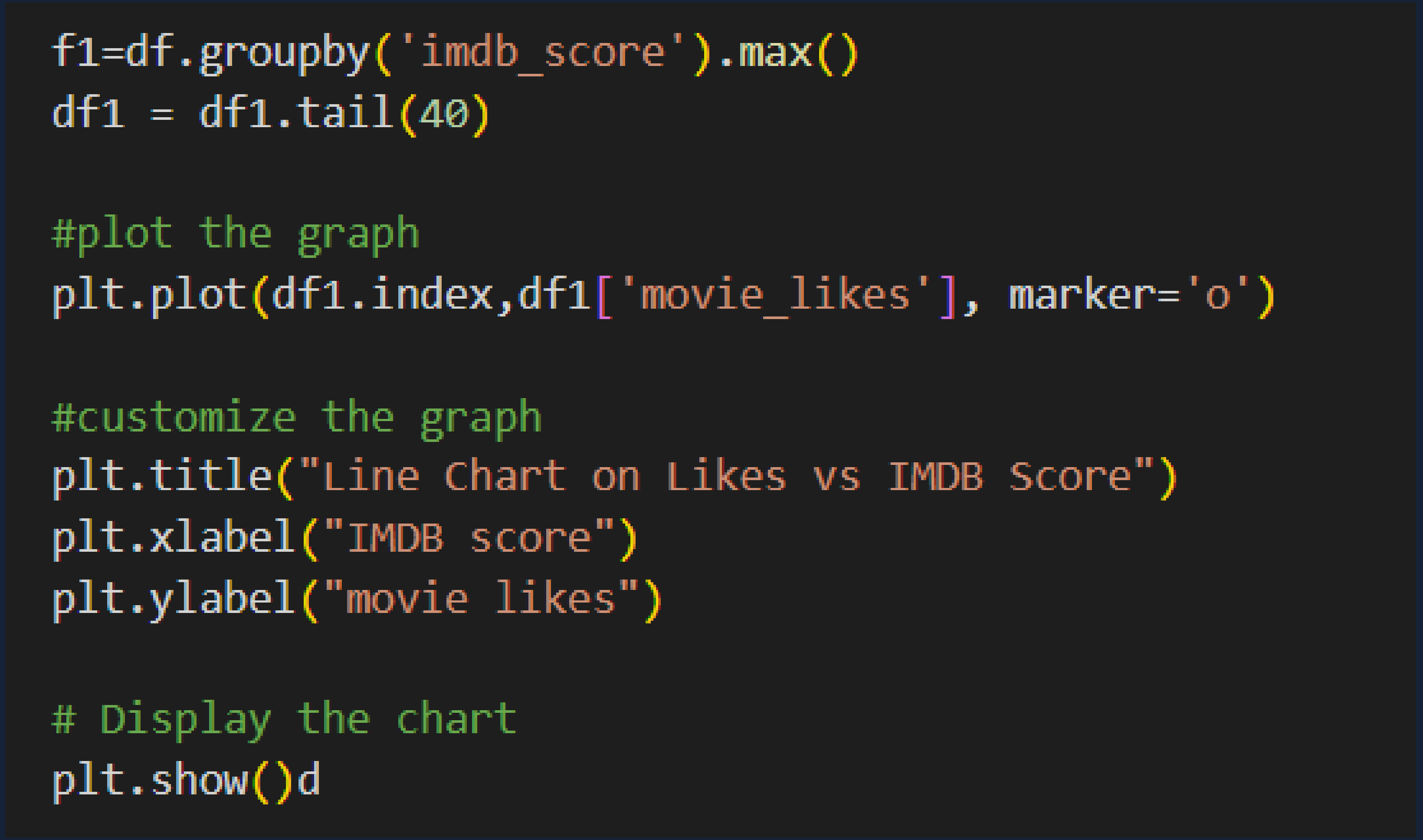
```
#18 data wrangling

newdf1 = pd.DataFrame(df[['director_name', 'duration', 'movie_title']])
newdf2 = pd.DataFrame(df[['movie_title', 'title_year', 'imdb_score']])

# merge dataframes
merged_df = pd.merge(newdf1, newdf2)
print(merged_df.head())

#concat dataframes
concatenated_df = pd.concat([newdf1, newdf2], axis=1)
print(concatenated_df.head())
```



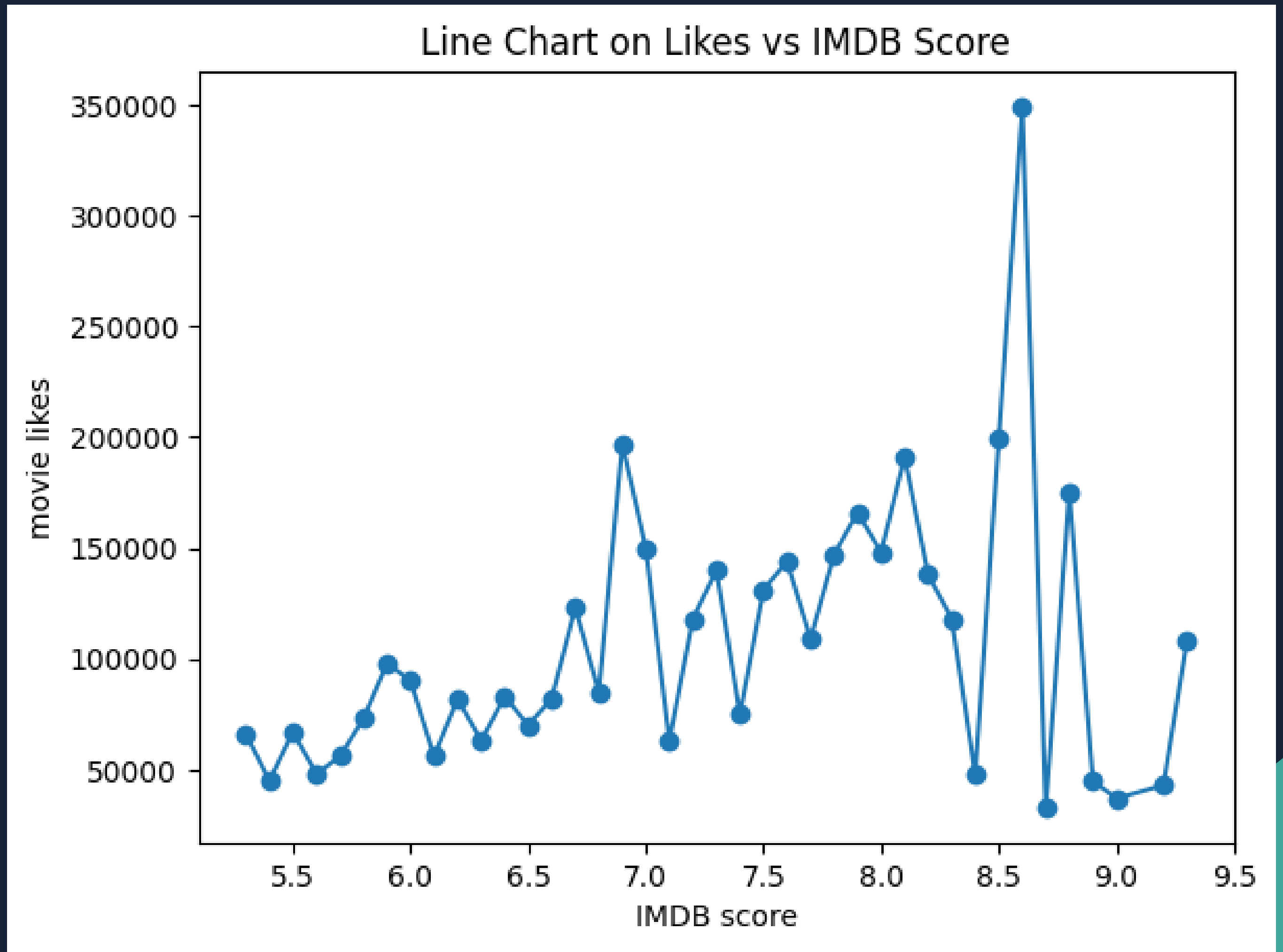
```
       director_name  duration                               movie_title  \
0      James Cameron     178.0                                   Avatar
1     Gore Verbinski     169.0  Pirates of the Caribbean: At World's End
2         Sam Mendes     148.0                                  Spectre
3  Christopher Nolan     164.0                    The Dark Knight Rises
4     Andrew Stanton     132.0                              John Carter

   title_year  imdb_score
0      2009.0         7.9
1      2007.0         7.1
2      2015.0         6.8
3      2012.0         8.5
4      2012.0         6.6
       director_name  duration                               movie_title  \
0      James Cameron     178.0                                   Avatar
1     Gore Verbinski     169.0  Pirates of the Caribbean: At World's End
2         Sam Mendes     148.0                                  Spectre
3  Christopher Nolan     164.0                    The Dark Knight Rises
4     Andrew Stanton     132.0                              John Carter

                                movie_title  title_year  imdb_score
0                                   Avatar      2009.0         7.9
1  Pirates of the Caribbean: At World's End      2007.0         7.1
2                                  Spectre      2015.0         6.8
3                    The Dark Knight Rises      2012.0         8.5
4                              John Carter      2012.0         6.6
```

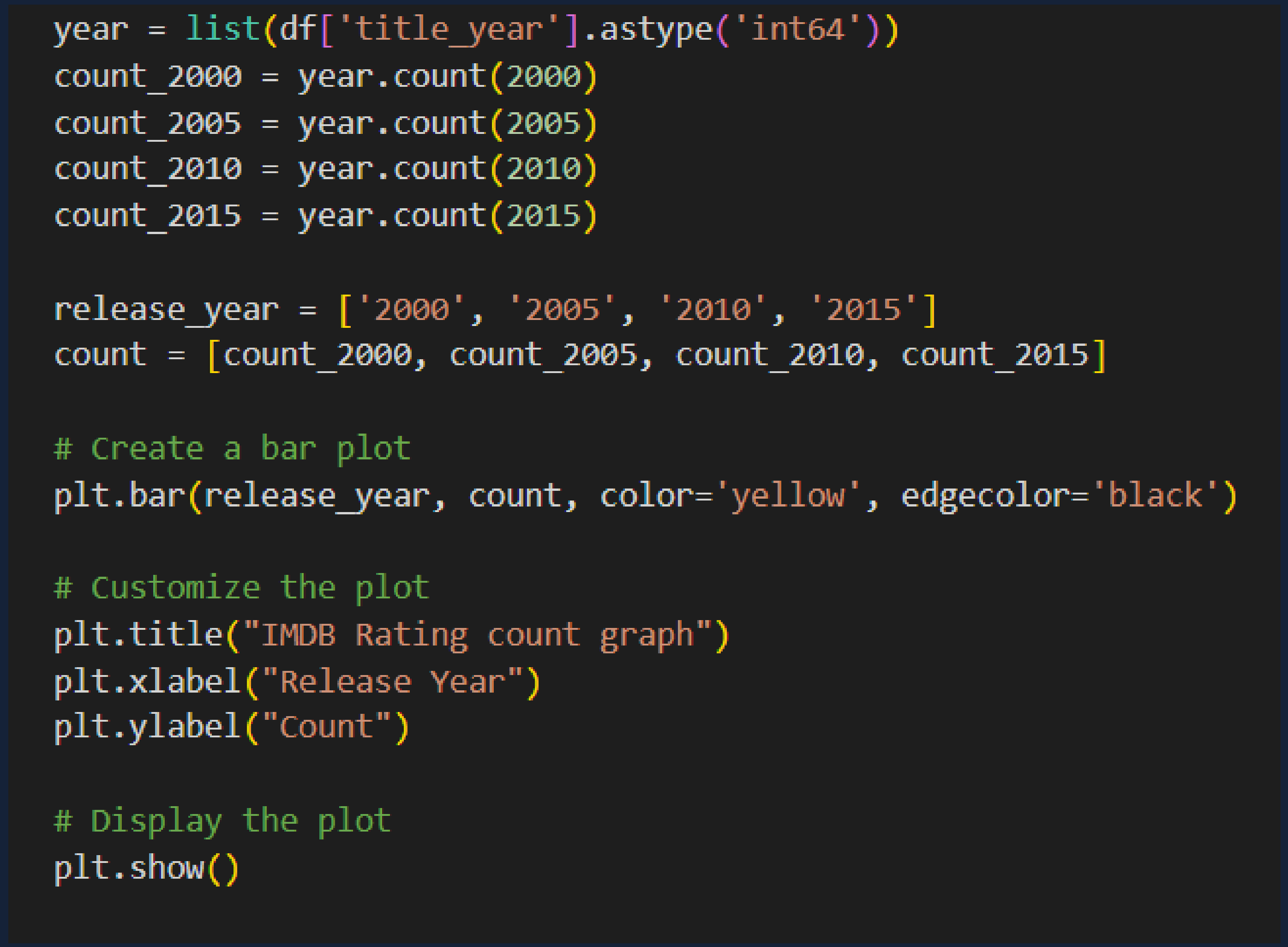
DATA VISUALIZATION

Data visualization is the process of representing data and information visually through charts, graphs, maps, and other graphical elements. It is a powerful technique that allows us to effectively communicate complex concepts, patterns, and trends in a visual format. Data visualization transforms complex data into visual representations that enhance understanding, reveal patterns, and support decision-making

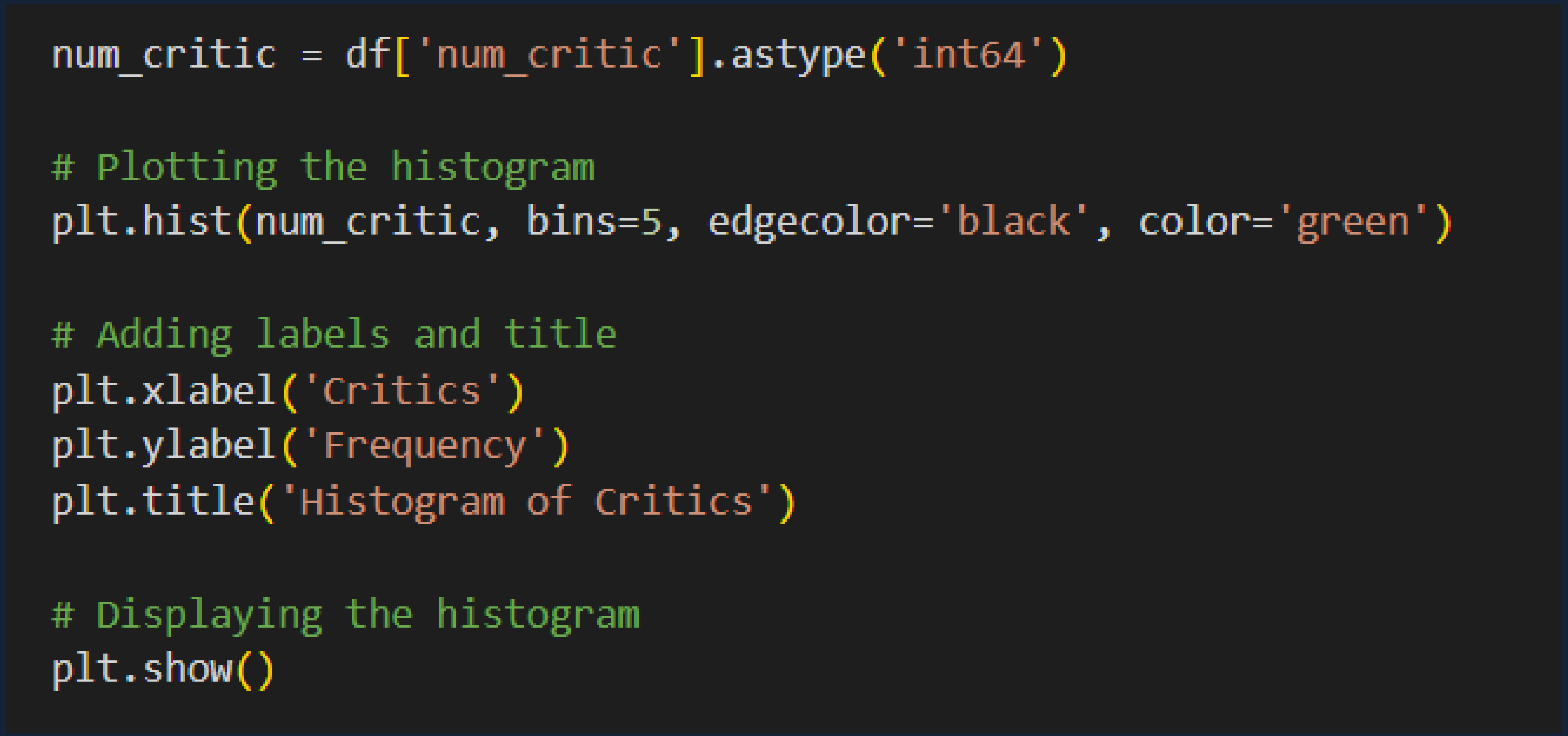


```
f1=df.groupby('imdb_score').max()
df1 = df1.tail(40)

#plot the graph
plt.plot(df1.index,df1['movie_likes'], marker='o')

#customize the graph
plt.title("Line Chart on Likes vs IMDB Score")
plt.xlabel("IMDB score")
plt.ylabel("movie likes")

# Display the chart
plt.show()d
```

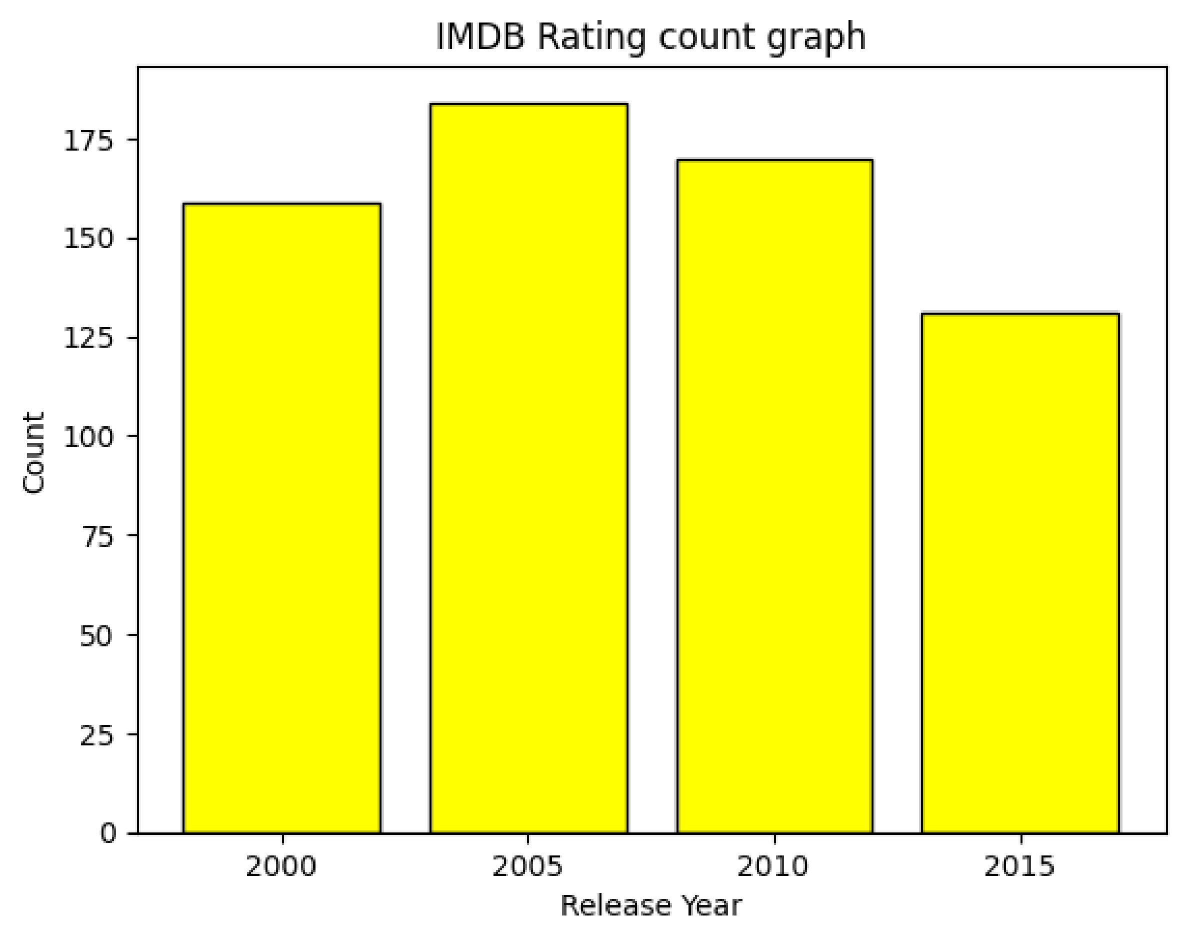


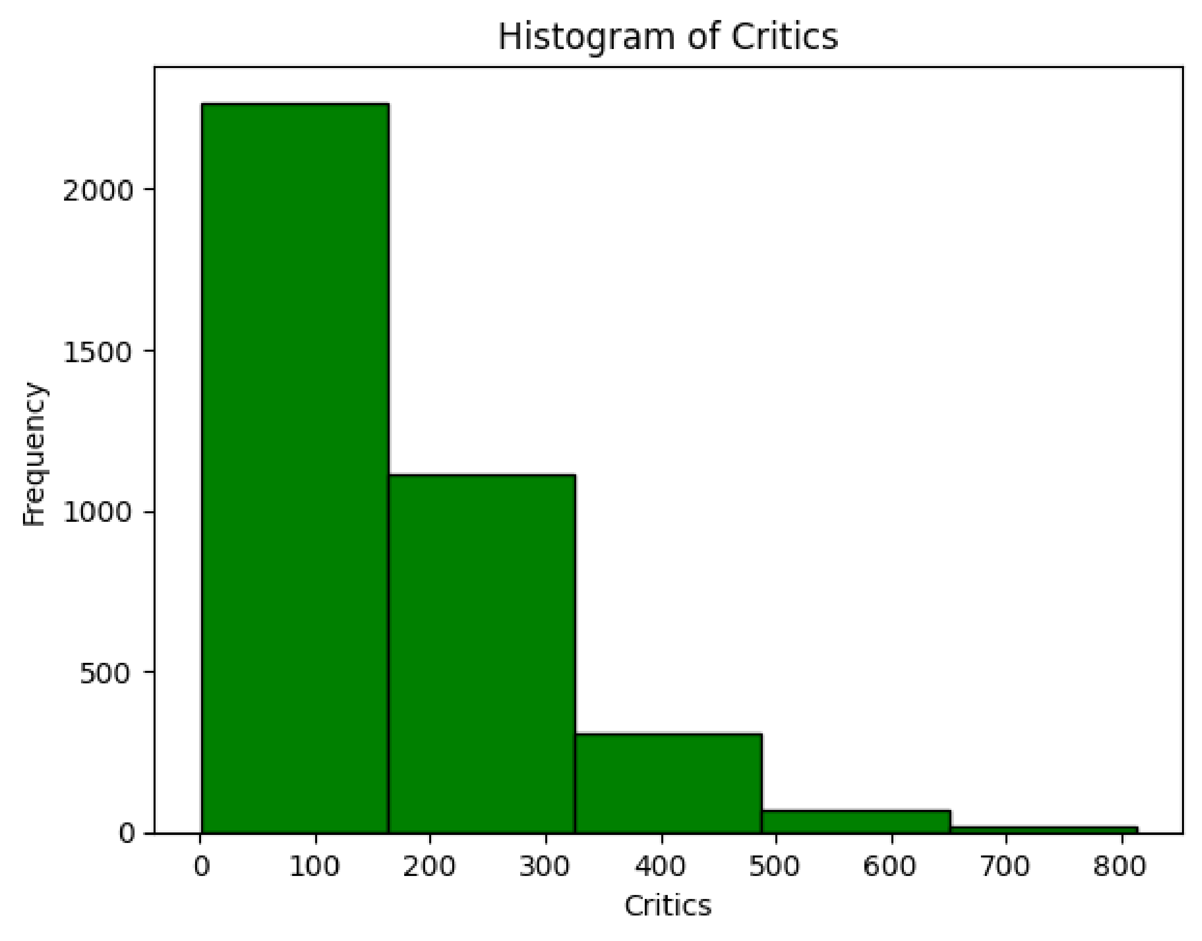


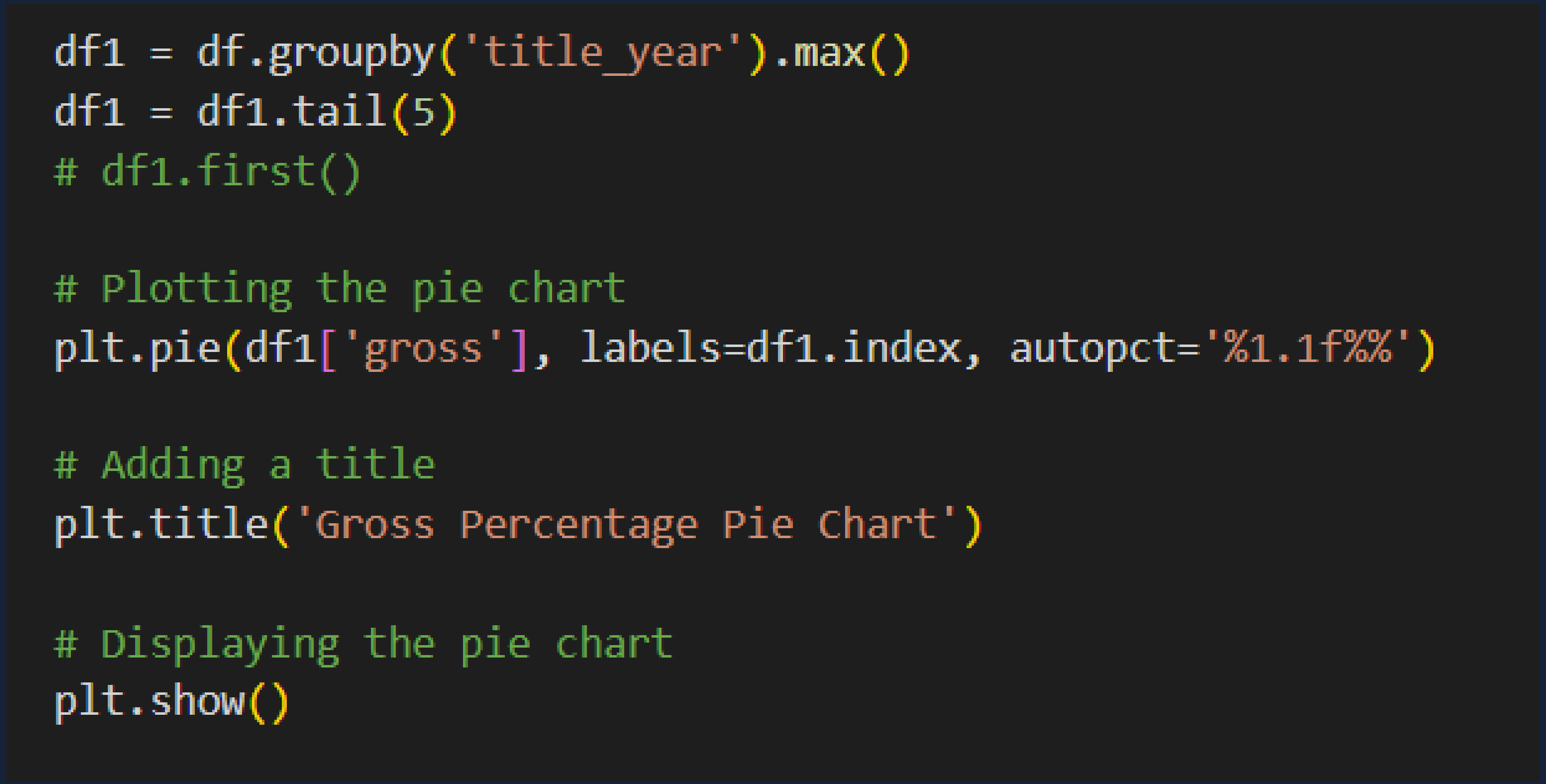
```
year = list(df['title_year'].astype('int64'))
count_2000 = year.count(2000)
count_2005 = year.count(2005)
count_2010 = year.count(2010)
count_2015 = year.count(2015)

release_year = ['2000', '2005', '2010', '2015']
count = [count_2000, count_2005, count_2010, count_2015]

# Create a bar plot
plt.bar(release_year, count, color='yellow', edgecolor='black')

# Customize the plot
plt.title("IMDB Rating count graph")
plt.xlabel("Release Year")
plt.ylabel("Count")

# Display the plot
plt.show()
```



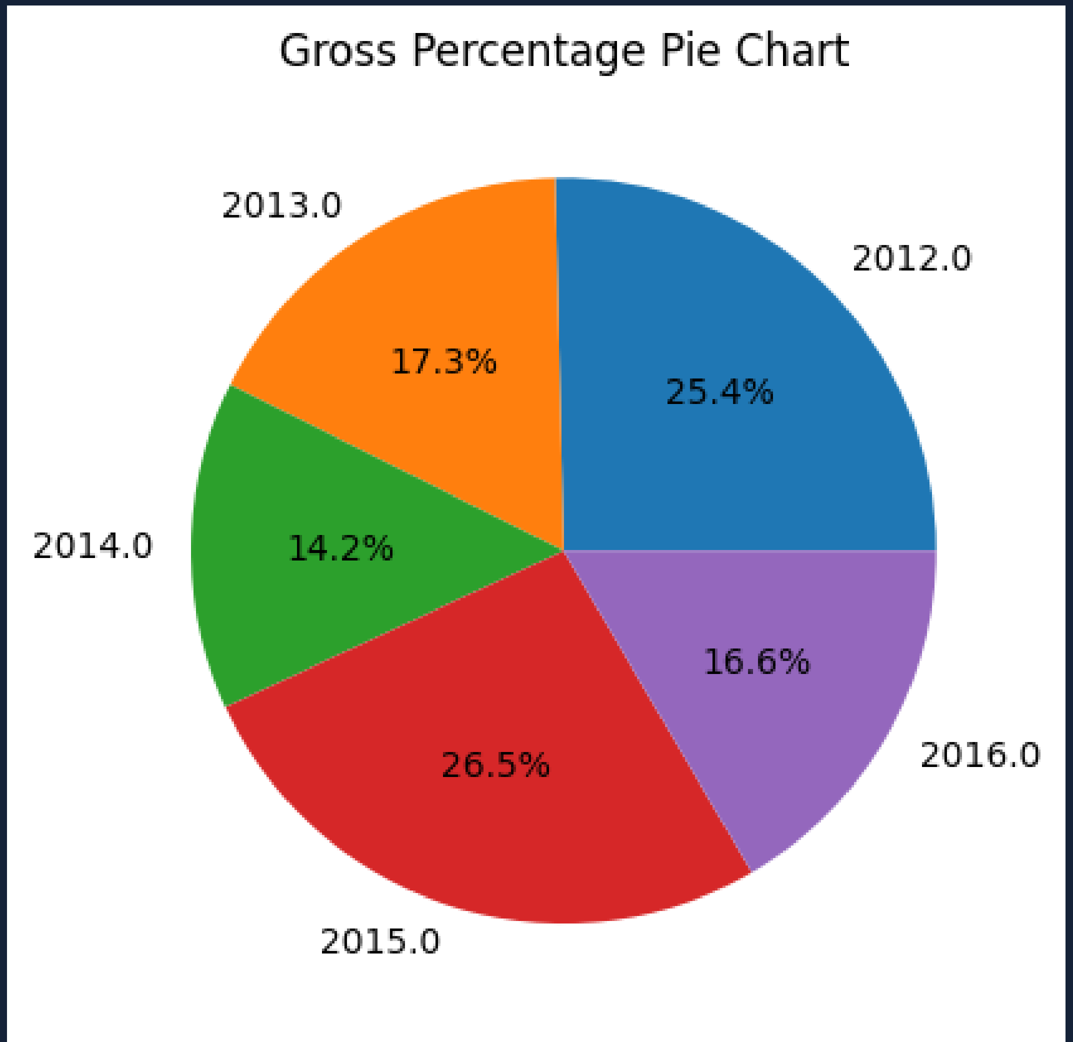
```
num_critic = df['num_critic'].astype('int64')

# Plotting the histogram
plt.hist(num_critic, bins=5, edgecolor='black', color='green')

# Adding labels and title
plt.xlabel('Critics')
plt.ylabel('Frequency')
plt.title('Histogram of Critics')

# Displaying the histogram
plt.show()
```



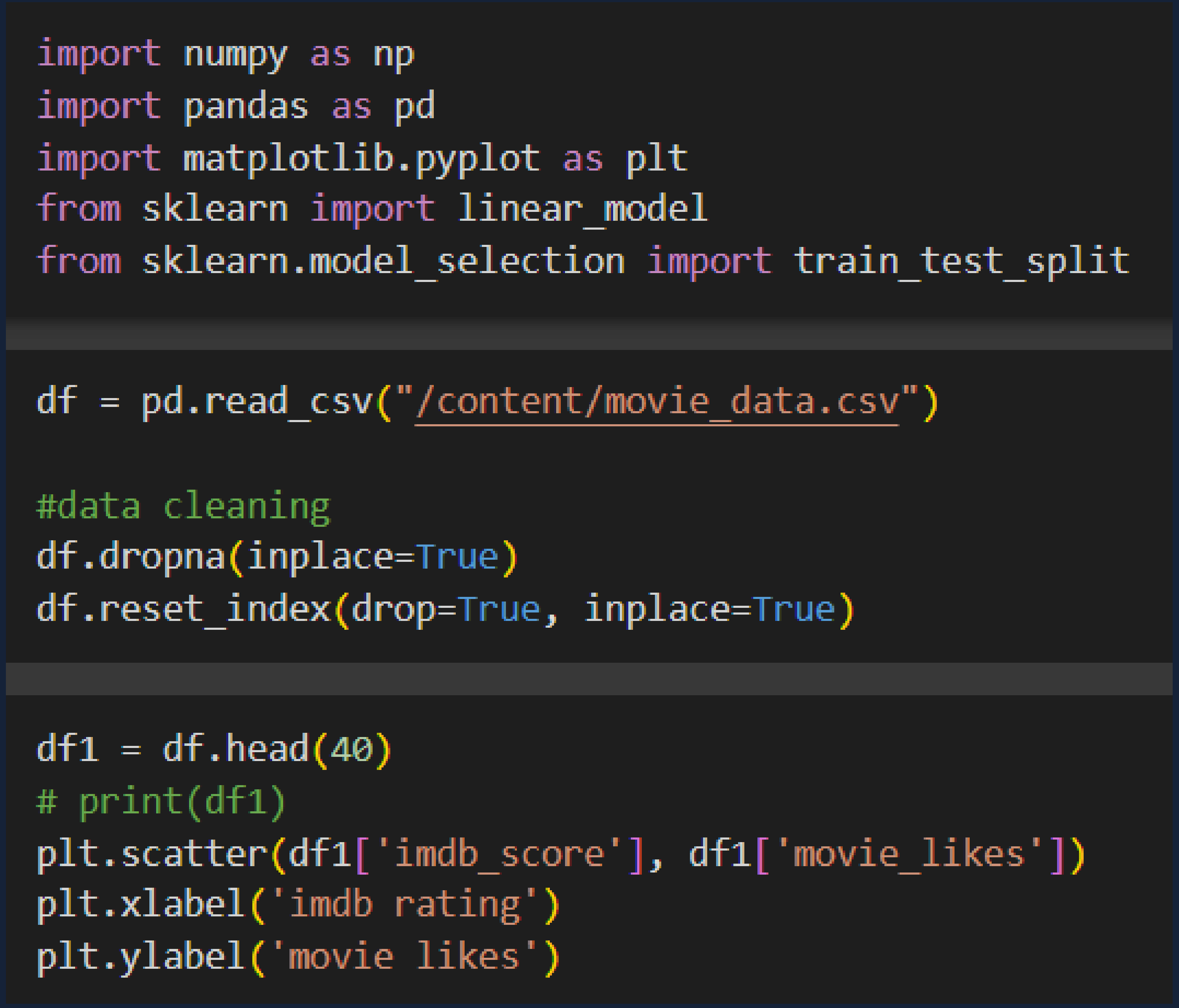




```
df1 = df.groupby('title_year').max()
df1 = df1.tail(5)
# df1.first()

# Plotting the pie chart
plt.pie(df1['gross'], labels=df1.index, autopct='%1.1f%%')

# Adding a title
plt.title('Gross Percentage Pie Chart')

# Displaying the pie chart
plt.show()
```



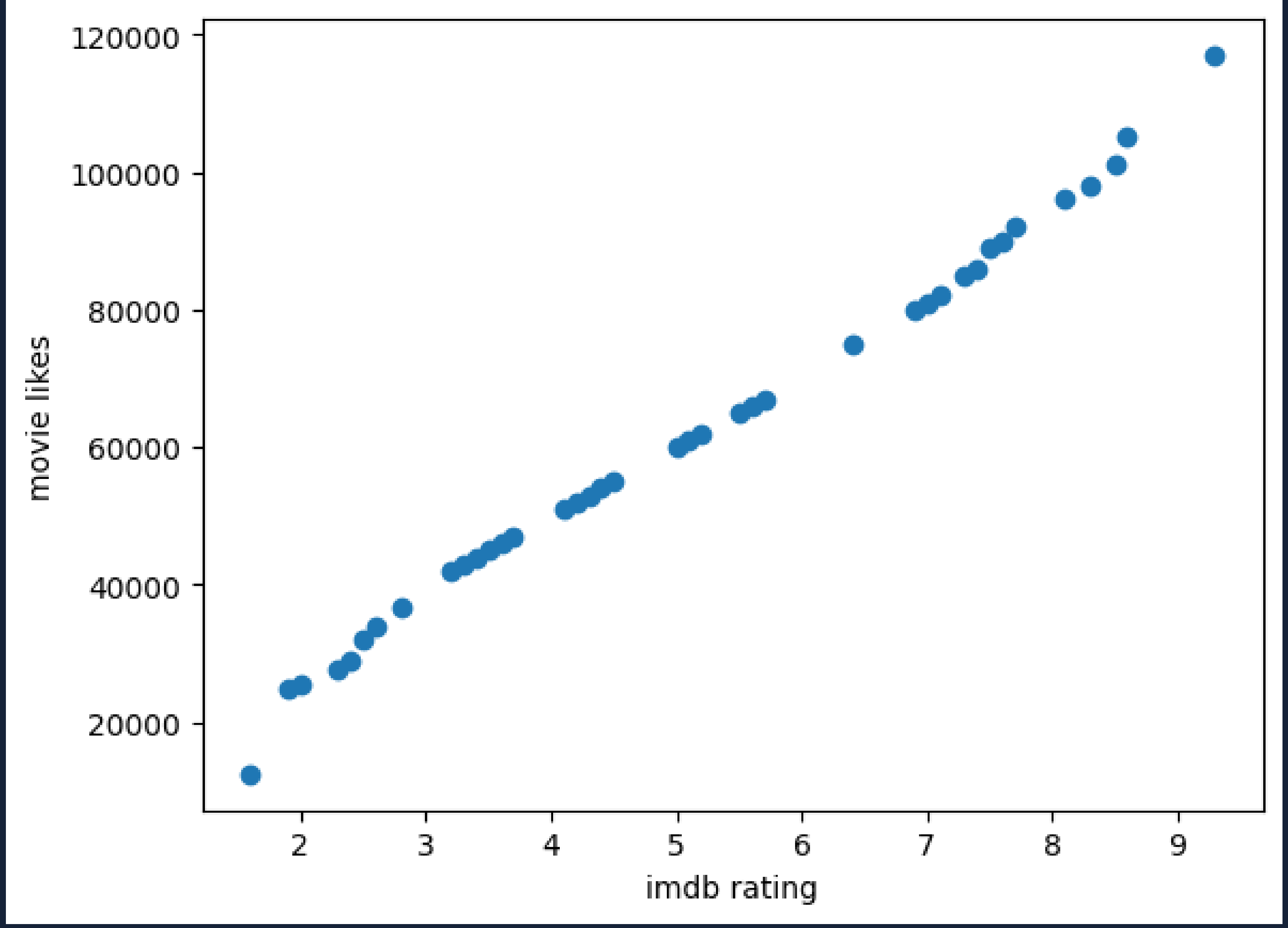
PREDICTIVE TECHNIQUE
(LINEAR RIGRESSION)

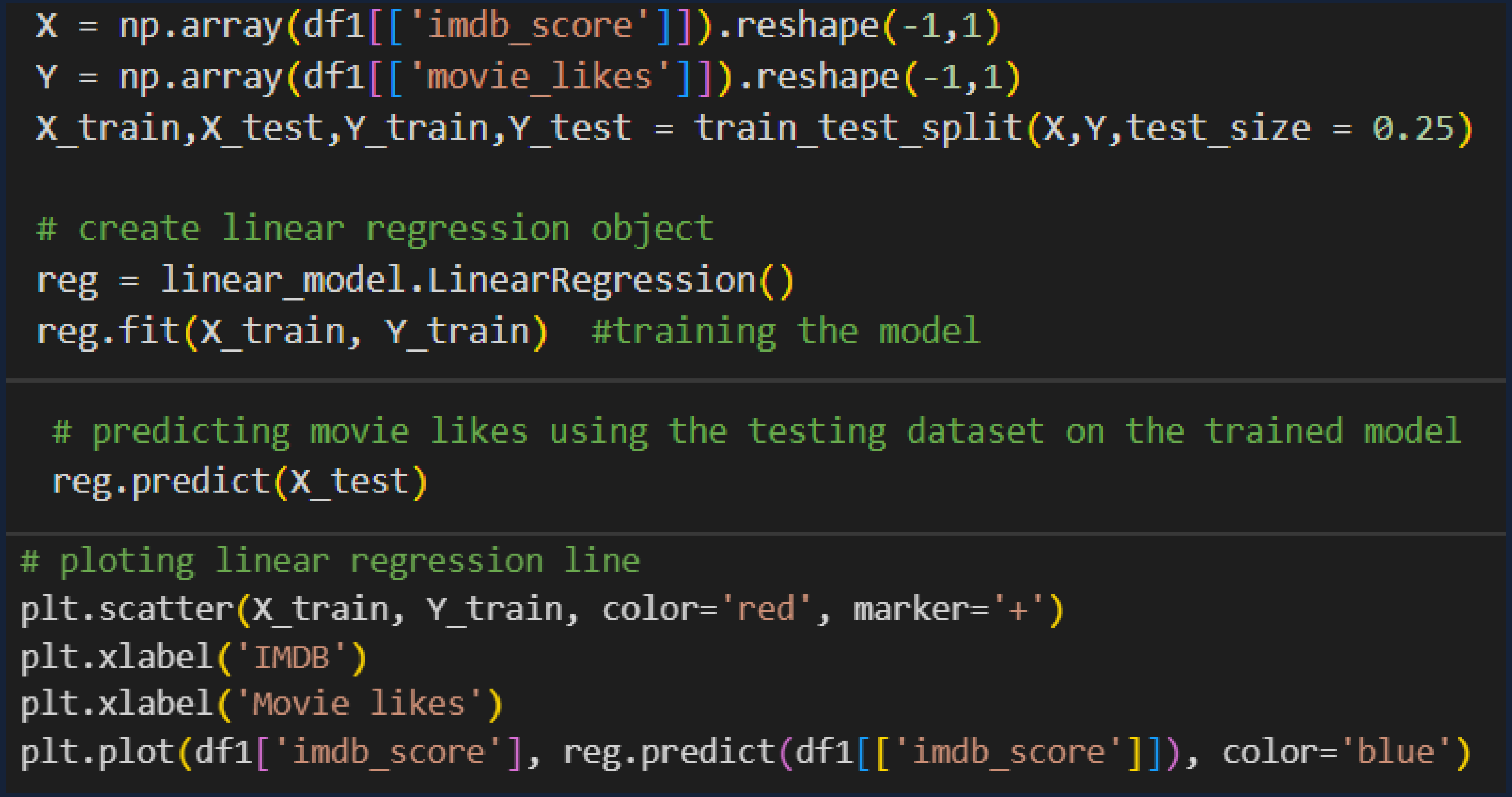


```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn import linear_model
from sklearn.model_selection import train_test_split
```

```
df = pd.read_csv("/content/movie_data.csv")

#data cleaning
df.dropna(inplace=True)
df.reset_index(drop=True, inplace=True)
```

```
df1 = df.head(40)
# print(df1)
plt.scatter(df1['imdb_score'], df1['movie_likes'])
plt.xlabel('imdb rating')
plt.ylabel('movie likes')
```

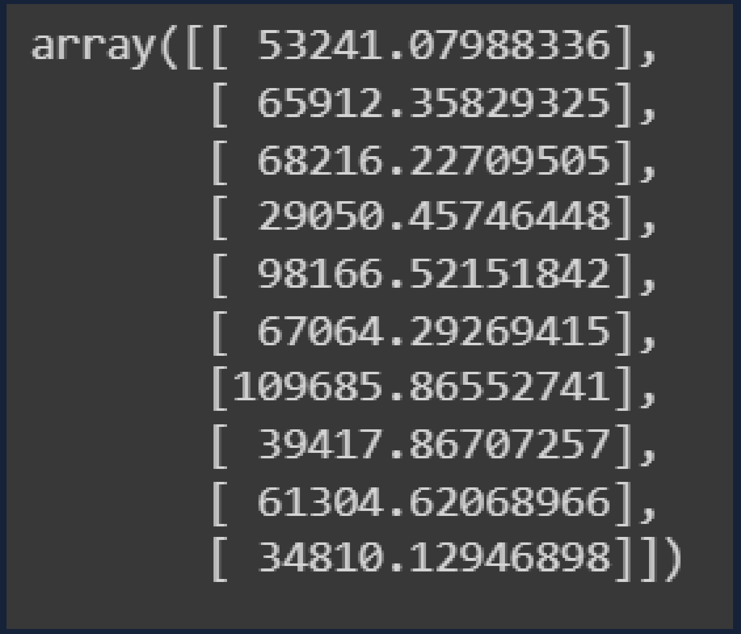




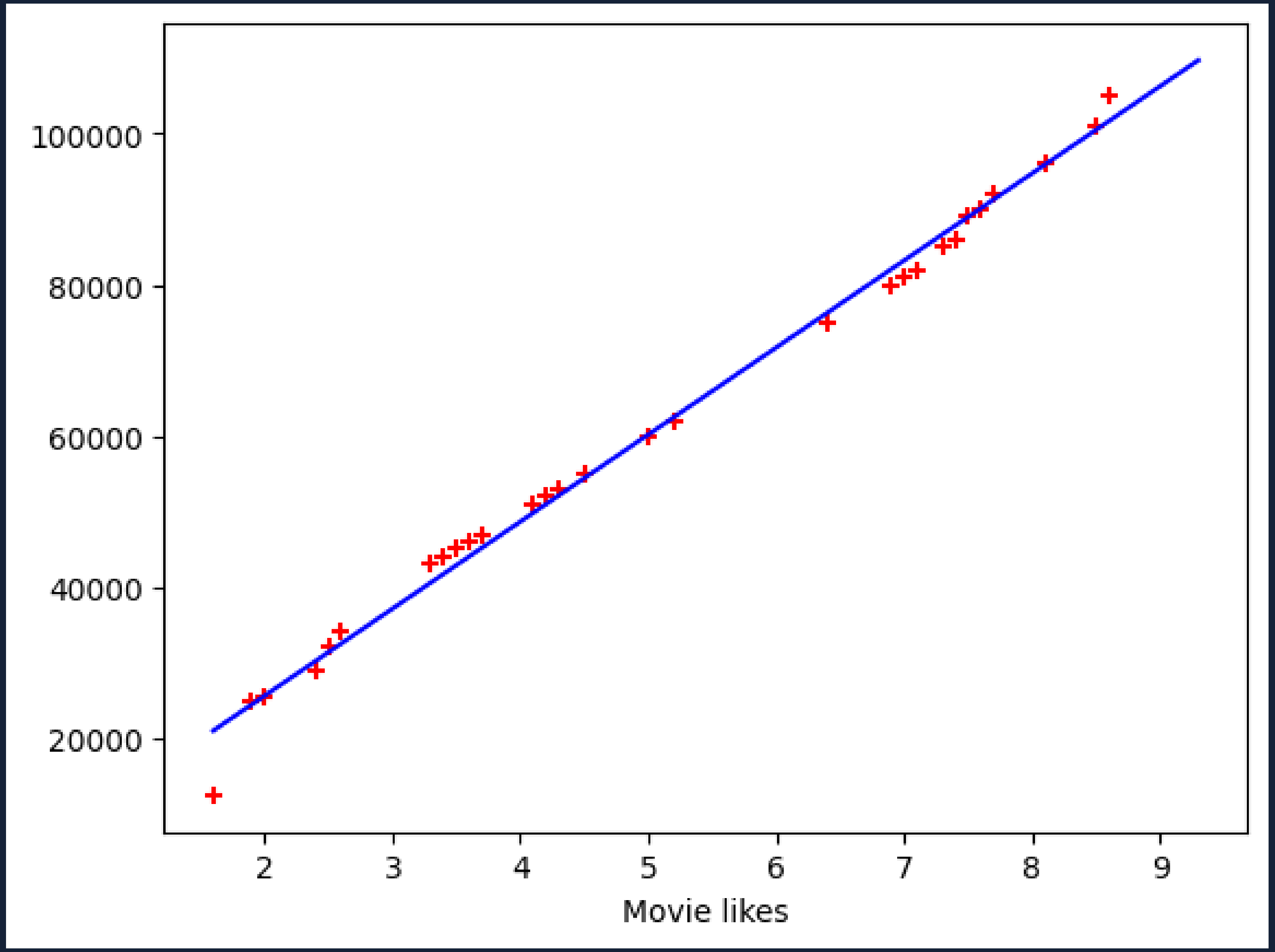
```python
X = np.array(df1[['imdb_score']]).reshape(-1,1)
Y = np.array(df1[['movie_likes']]).reshape(-1,1)
X_train,X_test,Y_train,Y_test = train_test_split(X,Y,test_size = 0.25)

# create linear regression object
reg = linear_model.LinearRegression()
reg.fit(X_train, Y_train)  #training the model
```

```python
# predicting movie likes using the testing dataset on the trained model
reg.predict(X_test)
```

```python
# ploting linear regression line
plt.scatter(X_train, Y_train, color='red', marker='+')
plt.xlabel('IMDB')
plt.xlabel('Movie likes')
plt.plot(df1['imdb_score'], reg.predict(df1[['imdb_score']]), color='blue')
```



```
array([[ 53241.07988336],
       [ 65912.35829325],
       [ 68216.22709505],
       [ 29050.45746448],
       [ 98166.52151842],
       [ 67064.29269415],
       [109685.86552741],
       [ 39417.86707257],
       [ 61304.62068966],
       [ 34810.12946898]])
```



APPLICATION

- By performing data manipulation techniques such as cleaning, filtering, and transforming the dataset, you can gain a deeper understanding of the data.
- Exploring summary statistics, distributions, and correlations between variables can provide insights into the characteristics and relationships within the dataset.
- Visualizing the Titanic dataset can help uncover patterns, trends, and relationships between variables. Plots such as histograms, scatter plots and bar charts can provide visual representations.
- After performing data manipulation, visualizing the data, and clustering using K means, the resulting clusters can serve as new features for predictive modeling.
- The cluster labels can be used as input features to build a classification model to predict survival or any other relevant outcome

CONCLUSION

- In conclusion, our analysis of the Titanic dataset has provided valuable insights into the passengers and the factors influencing their survival.
- We discovered significant correlations between survival and variables such as age, gender, passenger class, and family size.
- The analysis highlighted the importance of preparedness, class disparities, and gender biases during this tragic event.
- Through data cleaning, preprocessing, visualization, and modeling, we were able to extract meaningful information

THANK YOU