Data Analysis and Visualization Lab

Assignment 01

## TASK 1: Basic DataFrame Operations

import pandas as p

#### I. Download a dataset of your choice (CSV, Excel, or any other format). And load the dataset into a Pandas DataFrame.

data = p.read\_excel('data.xlsx', sheet\_name="Sheet1")

#### II. Display the first 5 rows of the dataset.

data.head()

Movies Director Actor \  
0 Kho Gaye Hum Kahan Arjun Varain Singh Sidhant Chaturvedi   
1 Anyone But You Will Gluck Glen Powell   
2 Cruel Intentions Roger Kumble Ryan Phillippe   
3 It Ends With Us Justin Baldoni Justin Baldoni   
4 The Voyeurs Michael Mohan Justice Smith   
  
 Actress Rating Censor Year Length   
0 Ananya Pandey 8.0 No 2023 135.0   
1 Sydney Sweeney 6.6 No 2024 113.0   
2 Reese Witherspoon 6.8 Yes 1999 97.0   
3 Blake Lively NaN No 2024 NaN   
4 Sydney Sweeney 6.1 Yes 2021 116.0

#### II. Check for missing values and handle them appropriately.

data.isnull().sum()

Movies 0  
Director 0  
Actor 0  
Actress 0  
Rating 1  
Censor 0  
Year 0  
Length 1  
dtype: int64

data['Length'].fillna(data['Length'].mean(), inplace=True)  
data['Rating'].fillna(data['Rating'].mean(), inplace=True)  
data

Movies Director Actor \  
0 Kho Gaye Hum Kahan Arjun Varain Singh Sidhant Chaturvedi   
1 Anyone But You Will Gluck Glen Powell   
2 Cruel Intentions Roger Kumble Ryan Phillippe   
3 It Ends With Us Justin Baldoni Justin Baldoni   
4 The Voyeurs Michael Mohan Justice Smith   
5 Body Heat Lawrence Kasdan William Hurt   
  
 Actress Rating Censor Year Length   
0 Ananya Pandey 8.00 No 2023 135.0   
1 Sydney Sweeney 6.60 No 2024 113.0   
2 Reese Witherspoon 6.80 Yes 1999 97.0   
3 Blake Lively 6.98 No 2024 114.8   
4 Sydney Sweeney 6.10 Yes 2021 116.0   
5 Kathleen Turner 7.40 Yes 1981 113.0

#### II. Get a summary of the dataset using describe().

data.describe()

Rating Year Length  
count 6.000000 6.00 6.000000  
mean 6.980000 2012.00 114.800000  
std 0.658483 18.00 12.106197  
min 6.100000 1981.00 97.000000  
25% 6.650000 2004.50 113.000000  
50% 6.890000 2022.00 113.900000  
75% 7.295000 2023.75 115.700000  
max 8.000000 2024.00 135.000000

#### III. Select a subset of columns from the DataFrame. Use both label-based and position-based indexing.

sub\_data\_by\_index = data[['Actor','Actress']]  
sub\_data\_by\_index

Actor Actress  
0 Sidhant Chaturvedi Ananya Pandey  
1 Glen Powell Sydney Sweeney  
2 Ryan Phillippe Reese Witherspoon  
3 Justin Baldoni Blake Lively  
4 Justice Smith Sydney Sweeney  
5 William Hurt Kathleen Turner

sub\_data\_by\_pos = data.iloc[:,[0,1,2]]  
sub\_data\_by\_pos

Movies Director Actor  
0 Kho Gaye Hum Kahan Arjun Varain Singh Sidhant Chaturvedi  
1 Anyone But You Will Gluck Glen Powell  
2 Cruel Intentions Roger Kumble Ryan Phillippe  
3 It Ends With Us Justin Baldoni Justin Baldoni  
4 The Voyeurs Michael Mohan Justice Smith  
5 Body Heat Lawrence Kasdan William Hurt

#### III. Create a new DataFrame by filtering rows based on a condition.

new\_data = data[data['Censor'] == "Yes"]  
new\_data

Movies Director Actor Actress \  
2 Cruel Intentions Roger Kumble Ryan Phillippe Reese Witherspoon   
4 The Voyeurs Michael Mohan Justice Smith Sydney Sweeney   
5 Body Heat Lawrence Kasdan William Hurt Kathleen Turner   
  
 Rating Censor Year Length   
2 6.8 Yes 1999 97.0   
4 6.1 Yes 2021 116.0   
5 7.4 Yes 1981 113.0

## TASK 2: Data Cleaning and Preprocessing

#### II. Create a new column by applying a mathematical operation on existing columns. Convert a categorical variable into numerical representation (e.g., using one-hot encoding).

data = p.get\_dummies(data, columns=['Censor'], drop\_first=True)  
data

Movies Director Actor \  
0 Kho Gaye Hum Kahan Arjun Varain Singh Sidhant Chaturvedi   
1 Anyone But You Will Gluck Glen Powell   
2 Cruel Intentions Roger Kumble Ryan Phillippe   
3 It Ends With Us Justin Baldoni Justin Baldoni   
4 The Voyeurs Michael Mohan Justice Smith   
5 Body Heat Lawrence Kasdan William Hurt   
  
 Actress Rating Year Length Censor\_Yes   
0 Ananya Pandey 8.00 2023 135.0 0   
1 Sydney Sweeney 6.60 2024 113.0 0   
2 Reese Witherspoon 6.80 1999 97.0 1   
3 Blake Lively 6.98 2024 114.8 0   
4 Sydney Sweeney 6.10 2021 116.0 1   
5 Kathleen Turner 7.40 1981 113.0 1

#### III. Group the data by a specific column. Apply aggregation functions (sum, mean, count) to the grouped data. Present the results in a meaningful way.

data.groupby("Year")["Movies"].agg(list).reset\_index().mean()

C:\Users\HP\AppData\Local\Temp\ipykernel\_5904\1789202964.py:1: FutureWarning: The default value of numeric\_only in DataFrame.mean is deprecated. In a future version, it will default to False. In addition, specifying 'numeric\_only=None' is deprecated. Select only valid columns or specify the value of numeric\_only to silence this warning.  
 data.groupby("Year")["Movies"].agg(list).reset\_index().mean()

Year 2009.6  
dtype: float64

data.groupby("Length")["Movies"].agg(list).reset\_index().mean()

C:\Users\HP\AppData\Local\Temp\ipykernel\_5904\387590397.py:1: FutureWarning: The default value of numeric\_only in DataFrame.mean is deprecated. In a future version, it will default to False. In addition, specifying 'numeric\_only=None' is deprecated. Select only valid columns or specify the value of numeric\_only to silence this warning.  
 data.groupby("Length")["Movies"].agg(list).reset\_index().mean()

Length 115.16  
dtype: float64

data.groupby("Rating")["Movies"].agg(list).reset\_index().mean()

C:\Users\HP\AppData\Local\Temp\ipykernel\_5904\2275640151.py:1: FutureWarning: The default value of numeric\_only in DataFrame.mean is deprecated. In a future version, it will default to False. In addition, specifying 'numeric\_only=None' is deprecated. Select only valid columns or specify the value of numeric\_only to silence this warning.  
 data.groupby("Rating")["Movies"].agg(list).reset\_index().mean()

Rating 6.98  
dtype: float64

## TASK 3: Load two different datasets. Merge them using different types of joins (inner, outer, left, right). Analyze the impact of each type of join on the merged dataset.

movies = p.read\_excel("data.xlsx", sheet\_name='Sheet1')  
genre = p.read\_excel("data.xlsx", sheet\_name='Genres')

inner\_join\_result = p.merge(movies, genre, on='Movies', how='inner')  
print("\nInner Join Result:")  
print(inner\_join\_result)

Inner Join Result:  
 Movies Director Actor \  
0 Kho Gaye Hum Kahan Arjun Varain Singh Sidhant Chaturvedi   
1 Anyone But You Will Gluck Glen Powell   
2 Cruel Intentions Roger Kumble Ryan Phillippe   
3 It Ends With Us Justin Baldoni Justin Baldoni   
4 The Voyeurs Michael Mohan Justice Smith   
5 Body Heat Lawrence Kasdan William Hurt   
  
 Actress Rating Censor Year Length Genre Sub-Genre   
0 Ananya Pandey 8.0 No 2023 135.0 Drama Rom-Com   
1 Sydney Sweeney 6.6 No 2024 113.0 Comdey Romantic   
2 Reese Witherspoon 6.8 Yes 1999 97.0 Teen Drama Thriller   
3 Blake Lively NaN No 2024 NaN Drama Romantic   
4 Sydney Sweeney 6.1 Yes 2021 116.0 Erotic Thriller   
5 Kathleen Turner 7.4 Yes 1981 113.0 Erotic Thriller

outer\_join\_result = p.merge(movies, genre, on='Movies', how='outer')  
print("\nOuter Join Result:")  
print(outer\_join\_result)

Outer Join Result:  
 Movies Director Actor \  
0 Kho Gaye Hum Kahan Arjun Varain Singh Sidhant Chaturvedi   
1 Anyone But You Will Gluck Glen Powell   
2 Cruel Intentions Roger Kumble Ryan Phillippe   
3 It Ends With Us Justin Baldoni Justin Baldoni   
4 The Voyeurs Michael Mohan Justice Smith   
5 Body Heat Lawrence Kasdan William Hurt   
  
 Actress Rating Censor Year Length Genre Sub-Genre   
0 Ananya Pandey 8.0 No 2023 135.0 Drama Rom-Com   
1 Sydney Sweeney 6.6 No 2024 113.0 Comdey Romantic   
2 Reese Witherspoon 6.8 Yes 1999 97.0 Teen Drama Thriller   
3 Blake Lively NaN No 2024 NaN Drama Romantic   
4 Sydney Sweeney 6.1 Yes 2021 116.0 Erotic Thriller   
5 Kathleen Turner 7.4 Yes 1981 113.0 Erotic Thriller

left\_join\_result = p.merge(movies, genre, on='Movies', how='left')  
print("\nLeft Join Result:")  
print(left\_join\_result)

Left Join Result:  
 Movies Director Actor \  
0 Kho Gaye Hum Kahan Arjun Varain Singh Sidhant Chaturvedi   
1 Anyone But You Will Gluck Glen Powell   
2 Cruel Intentions Roger Kumble Ryan Phillippe   
3 It Ends With Us Justin Baldoni Justin Baldoni   
4 The Voyeurs Michael Mohan Justice Smith   
5 Body Heat Lawrence Kasdan William Hurt   
  
 Actress Rating Censor Year Length Genre Sub-Genre   
0 Ananya Pandey 8.0 No 2023 135.0 Drama Rom-Com   
1 Sydney Sweeney 6.6 No 2024 113.0 Comdey Romantic   
2 Reese Witherspoon 6.8 Yes 1999 97.0 Teen Drama Thriller   
3 Blake Lively NaN No 2024 NaN Drama Romantic   
4 Sydney Sweeney 6.1 Yes 2021 116.0 Erotic Thriller   
5 Kathleen Turner 7.4 Yes 1981 113.0 Erotic Thriller

right\_join\_result = p.merge(movies, genre, on='Movies', how='right')  
print("\nRight Join Result:")  
print(right\_join\_result)

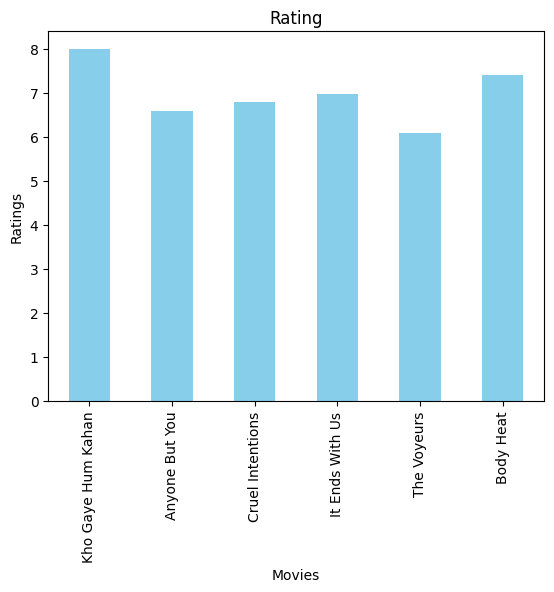
Right Join Result:  
 Movies Director Actor \  
0 Kho Gaye Hum Kahan Arjun Varain Singh Sidhant Chaturvedi   
1 Anyone But You Will Gluck Glen Powell   
2 Cruel Intentions Roger Kumble Ryan Phillippe   
3 It Ends With Us Justin Baldoni Justin Baldoni   
4 The Voyeurs Michael Mohan Justice Smith   
5 Body Heat Lawrence Kasdan William Hurt   
  
 Actress Rating Censor Year Length Genre Sub-Genre   
0 Ananya Pandey 8.0 No 2023 135.0 Drama Rom-Com   
1 Sydney Sweeney 6.6 No 2024 113.0 Comdey Romantic   
2 Reese Witherspoon 6.8 Yes 1999 97.0 Teen Drama Thriller   
3 Blake Lively NaN No 2024 NaN Drama Romantic   
4 Sydney Sweeney 6.1 Yes 2021 116.0 Erotic Thriller   
5 Kathleen Turner 7.4 Yes 1981 113.0 Erotic Thriller

## TASK 4: Visualization

#### I. Create a bar plot, line plot, and scatter plot using Pandas plotting functions. Customize the plots to make them more informative.

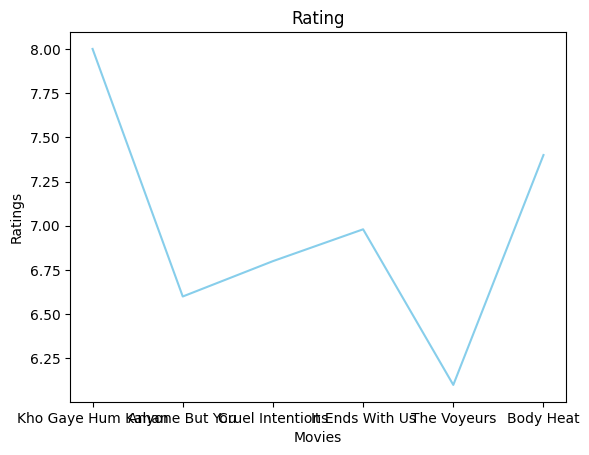
data.plot(kind='bar', x='Movies', y='Rating', color='skyblue', legend=False, title='Rating', ylabel="Ratings", xlabel="Movies")

<Axes: title={'center': 'Rating'}, xlabel='Movies', ylabel='Ratings'>



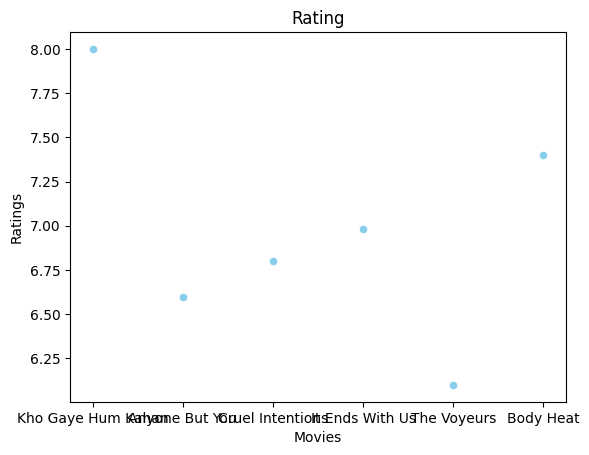
data.plot(kind='line', x='Movies', y='Rating', color='skyblue', legend=False, title='Rating', ylabel="Ratings", xlabel="Movies")

<Axes: title={'center': 'Rating'}, xlabel='Movies', ylabel='Ratings'>



data.plot(kind='scatter', x='Movies', y='Rating', color='skyblue', legend=False, title='Rating', ylabel="Ratings", xlabel="Movies")

<Axes: title={'center': 'Rating'}, xlabel='Movies', ylabel='Ratings'>

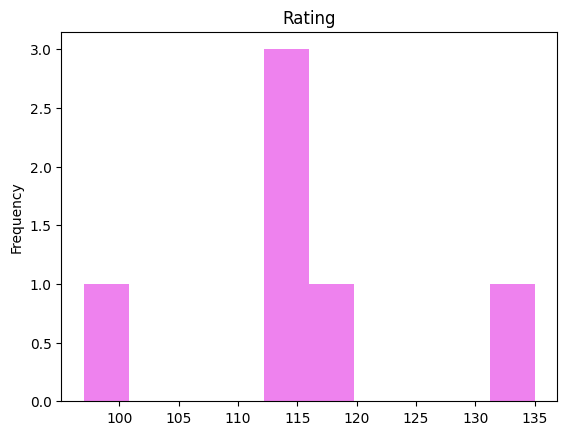


#### II. Visualize the correlation matrix of numerical columns. Highlight highly correlated features.

#### III. Create histograms and box plots for numerical columns. Analyze the distribution and presence of outliers

data.plot(kind='hist', x='Movies', y='Length', color='violet', legend=False, title='Rating')

<Axes: title={'center': 'Rating'}, ylabel='Frequency'>



## TASK 5: Basic NumPy Operations

import numpy as np

#### 1. Create a NumPy array 'arr' with values from 1 to 10.

arr = np.arange(1,11, dtype="int32")  
arr

array([ 1, 2, 3, 4, 5, 6, 7, 8, 9, 10])

#### 2. Create another NumPy array 'arr2' with values from 11 to 20.

arr2 = np.arange(11,21, dtype="int32")  
arr2

array([11, 12, 13, 14, 15, 16, 17, 18, 19, 20])

#### 3. Add, subtract, multiply, and divide 'arr' and 'arr2'. Print the results.

print(f"The sum of the arrays is: {arr+arr2}")  
print(f"The difference of the arrays is: {arr-arr2}")  
print(f"The multiplication of the arrays is: {arr\*arr2}")  
print(f"The division of the arrays is: {arr/arr2}")

The sum of the arrays is: [12 14 16 18 20 22 24 26 28 30]  
The difference of the arrays is: [-10 -10 -10 -10 -10 -10 -10 -10 -10 -10]  
The multiplication of the arrays is: [ 11 24 39 56 75 96 119 144 171 200]  
The division of the arrays is: [0.09090909 0.16666667 0.23076923 0.28571429 0.33333333 0.375  
 0.41176471 0.44444444 0.47368421 0.5 ]

## TASK 6: Array Manipulation

#### 1. Reshape 'arr' into a 2x5 matrix.

new\_arr = arr.reshape(2,5)  
new\_arr

array([[ 1, 2, 3, 4, 5],  
 [ 6, 7, 8, 9, 10]])

#### 2. Transpose the matrix obtained in the previous step.

new\_arr.transpose()

array([[ 1, 6],  
 [ 2, 7],  
 [ 3, 8],  
 [ 4, 9],  
 [ 5, 10]])

#### 3. Flatten the transposed matrix into a 1D array.

new\_arr.reshape(1,10)

array([[ 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]])

#### 4. Stack 'arr' and 'arr2' vertically. Print the result.

np.vstack((arr, arr2))

array([[ 1, 2, 3, 4, 5, 6, 7, 8, 9, 10],  
 [11, 12, 13, 14, 15, 16, 17, 18, 19, 20]])

## TASK 7: Statistical Operations

#### 1. Calculate the mean, median, and standard deviation of 'arr'.

std = np.std(arr)  
med = np.median(arr)  
mean = np.mean(arr)  
print(f"The median is: {med}")  
print(f"The mean is: {mean}")  
print(f"The standard deviation is: {std}")

The median is: 5.5  
The mean is: 5.5  
The standard deviation is: 2.8722813232690143

#### 2. Find the maximum and minimum values in 'arr'.

print(f"The max is is: {max(arr)}")  
print(f"The min is: {min(arr)}")

The max is is: 10  
The min is: 1

#### 3. Normalize 'arr' (subtract the mean and divide by the standard deviation).

normal\_array = []  
for data in arr:  
 normal\_array.append((data-mean)/std)  
print(f"Normalized Array: {normal\_array}")

Normalized Array: [-1.5666989036012806, -1.2185435916898848, -0.8703882797784892, -0.5222329678670935, -0.17407765595569785, 0.17407765595569785, 0.5222329678670935, 0.8703882797784892, 1.2185435916898848, 1.5666989036012806]

## TASK 8: Boolean Indexing

#### 1. Create a boolean array 'bool\_arr' for elements in 'arr' greater than 5.

bool\_arr = list(map((lambda x: x>5), arr))  
bool\_arr

[False, False, False, False, False, True, True, True, True, True]

#### 2. Use 'bool\_arr' to extract the elements from 'arr' that are greater than 5.

bool\_arr2 = arr[bool\_arr]  
bool\_arr2

array([ 6, 7, 8, 9, 10])

## TASK 9: Random Module

#### 1. Generate a 3x3 matrix with random values between 0 and 1.

rand\_arr = np.random.randint(0,1, (3,3))  
rand\_arr

array([[0, 0, 0],  
 [0, 0, 0],  
 [0, 0, 0]])

#### 2. Create an array of 10 random integers between 1 and 100.

rand\_arr = np.random.randint(1,100, (1,10))  
rand\_arr

array([[72, 77, 34, 91, 67, 59, 36, 48, 78, 16]])

#### 3. Shuffle the elements of 'arr' randomly.

np.random.shuffle(arr)  
arr

array([ 9, 5, 3, 7, 6, 8, 2, 4, 1, 10])

## TASK 10: Random Module

#### 1. Apply the square root function to all elements in 'arr'.

list(map(lambda x: np.sqrt(x),arr))

[3.0,  
 2.23606797749979,  
 1.7320508075688772,  
 2.6457513110645907,  
 2.449489742783178,  
 2.8284271247461903,  
 1.4142135623730951,  
 2.0,  
 1.0,  
 3.1622776601683795]

### OR

square\_root = np.sqrt(arr)  
square\_root

array([3. , 2.23606798, 1.73205081, 2.64575131, 2.44948974,  
 2.82842712, 1.41421356, 2. , 1. , 3.16227766])

#### 2. Use the exponential function to calculate exex for each element in 'arr'.

exponential\_array = np.exp(arr)  
print(exponential\_array)

[8.10308393e+03 1.48413159e+02 2.00855369e+01 1.09663316e+03  
 4.03428793e+02 2.98095799e+03 7.38905610e+00 5.45981500e+01  
 2.71828183e+00 2.20264658e+04]

## TASK 11: Linear Algebra Operations

#### 1. Create a 3x3 matrix 'mat\_a' with random values.

mat\_a = np.matrix(np.random.randint(0,100, (3,3)))  
mat\_a

matrix([[82, 26, 45],  
 [34, 92, 0],  
 [25, 19, 26]])

#### 2. Create a 3x1 matrix 'vec\_b' with random values.

vec\_b = np.matrix(np.random.randint(0,100, (3,1)))  
vec\_b

matrix([[32],  
 [98],  
 [60]])

#### 3. Multiply 'mat\_a' and 'vec\_b' using the dot product.

mat\_a.dot(vec\_b)

matrix([[ 7872],  
 [10104],  
 [ 4222]])

## TASK 12: Broadcasting

#### 1. Create a 2D array 'matrix' with values from 1 to 9.

matrix = np.arange(1,9)  
matrix

array([1, 2, 3, 4, 5, 6, 7, 8])

#### 2. Subtract the mean of each row from each element in that row.

matrix = [[1, 2, 3],  
 [4, 5, 6],  
 [7, 8, 9]]

mean\_row = []  
for x in range(len(matrix)):  
 for y in range(len(matrix)):  
 mean\_row.append(matrix[x][y]-np.mean(matrix[x]))  
mean\_row

[-1.0, 0.0, 1.0, -1.0, 0.0, 1.0, -1.0, 0.0, 1.0]

mean\_row = []  
for x in range(len(matrix)):  
 for y in range(len(matrix)):  
 matrix[x][y]=matrix[x][y]-np.mean(matrix[x])  
matrix

[[-1.0, 0.6666666666666667, 2.111111111111111],  
 [-1.0, 1.6666666666666665, 3.777777777777778],  
 [-1.0, 2.666666666666667, 5.444444444444445]]

vertopal convert assignment.ipynb --to docx

Cell In[113], line 1  
 vertopal convert assignment.ipynb --to docx  
 ^  
SyntaxError: invalid syntax