NumPy and Pandas

Agenda

- NumPy and Pandas Quiz
- NumPy and NumPy Functions
- Pandas and Pandas Functions
- Merge and Join in Pandas
- Loading Datasets in Pandas

Let's begin the discussion by answering a few questions on NumPy and Pandas

NumPy and Pandas Quiz

What does the following code snippet do?

```
np.arange(1, 10, 2)
```

A. Return an array of integers from 1 to 10 (included) with step size 2.

B. Return an array of integers from 1 to 9 (included) with step size 2.

C. Return an array of integers from 1 to 9 (excluded) with step size 2.

D. Return an array of integers from 2 to 10 (included) with step size 1.

NumPy and Pandas Quiz

What does the following code snippet do?

```
np.arange(1, 10, 2)
```

A. Return an array of integers from 1 to 10 (included) with step size 2.

B. Return an array of integers from 1 to 9 (included) with step size 2.

C. Return an array of integers from 1 to 9 (excluded) with step size 2.

D. Return an array of integers from 2 to 10 (included) with step size 1.

NumPy

NumPy stands for **Numerical Python** - one of the fundamental packages for **mathematical**, **logical**, and **statistical operations** with Python

Provides powerful **n-dimensional array** object, called **ndarray**

Provides a large set of functions for creating, manipulating, and transforming ndarrays

Function	Syntax	Example	Description
np.array()	`np.array(object, dtype=None)`	`np.array([1, 2, 3])`	To create an array
np.arange()	`np.arange(start, stop, step)`	`np.arange(0, 10, 2)`	To create an array of evenly spaced values within a given interval

NumPy and Pandas Quiz

What does the following code snippet do?

```
np.random.randint(10, 20, 1000)
```

A. Return an array of 1000 integers between 10 and 20 (excluded)

B. Return an array of 10 integers between 20 and 1000 (included)

C. Return an array of 10 integers between 20 and 1000 (excluded)

D. Return an array of 1000 integers between 10 and 20 (included)

NumPy and Pandas Quiz

What does the following code snippet do?

```
np.random.randint(10, 20, 1000)
```

A. Return an array of 1000 integers between 10 and 20 (excluded)

B. Return an array of 10 integers between 20 and 1000 (included)

C. Return an array of 10 integers between 20 and 1000 (excluded)

D. Return an array of 1000 integers between 10 and 20 (included)

NumPy Functions

Function	Syntax	Example	Description
np.random.rand()	`np.random.rand(d0, d1, ..., dn)` • **d0, d1, ..., dn:** The dimensions of the returned array.	`np.random.rand(3, 2)`	To create an array of specified shape filled with random values from the uniform distribution
np.random.randint()	`np.random.randint(low, high, size)`	`np.random.randint(1, 10, size=(2, 3))`	To create an array of specified shape filled with random integers from low (inclusive) to high (exclusive)
np.random.randn()	`np.random.randn(d0, d1, ..., dn)` • **d0, d1, ..., dn:** The dimensions of the returned array.	`np.random.randn(2, 3)`	To create an array of specified shape filled with random values from the standard normal distribution

NumPy and Pandas Quiz

Great Learning
POWER AHEAD

Consider a dataframe cust_data having information on the following attributes of 200 customers (in the same order) - ID, Name, Age, Annual Income, Job Category
Which of the following can be used to fetch the Age and Annual Income of the first 100 customers?

A
```
cust_data.iloc[:100, 2:3]
```

B
```
cust_data.iloc[:100, 2:4]
```

C
```
cust_data.loc[:100, 'Age':'Annual Income']
```

D
```
cust_data.loc[:100, 'Age':'Job Category']
```

NumPy and Pandas Quiz

Great Learning
POWER AHEAD

Consider a dataframe cust_data having information on the following attributes of 200 customers (in the same order) - ID, Name, Age, Annual Income, Job Category
Which of the following can be used to fetch the Age and Annual Income of the first 100 customers?

A
```
cust_data.iloc[:100, 2:3]
```

B
```
cust_data.iloc[:100, 2:4]
```

C
```
cust_data.loc[:100, 'Age':'Annual Income']
```

D
```
cust_data.loc[:100, 'Age':'Job Category']
```

Pandas

Pandas is primarily used for **analysis and manipulation of tabular data**

Offers two major data structures - **Series & Dataframe**

One can think of a **pandas dataframe like an excel spreadsheet** - data stored in rows and columns.

Function	Syntax	Example	Description
df.loc[]	df.loc[row label_start:row label end, column label_start:column _label_end]	df.loc[10:100, 'Age':'Annual Income']	Access elements via label-based indexing (includes the end label)
df.iloc[]	df.iloc[row_index_start:row_index_end, column_index_start:column _index_end]	df.iloc[10:20, 2:4]	Access elements via integer-based indexing (excludes the end index)

NumPy and Pandas Quiz

Which of the following can be used to drop the Job Category column and ensures modification is directly made to the dataframe?

A
```
cust_data.drop('Job Category', axis=0, inplace=True)
```

B
```
cust_data.drop('Job Category', axis=1, inplace=False)
```

C
```
cust_data.drop('Job Category', axis=0, inplace=False)
```

D
```
cust_data.drop('Job Category', axis=1, inplace=True)
```

NumPy and Pandas Quiz

Great Learning
POWER AHEAD

Which of the following can be used to drop the Job Category column and ensures modification is directly made to the dataframe?

A
```
cust_data.drop('Job Category', axis=0, inplace=True)
```

B
```
cust_data.drop('Job Category', axis=1, inplace=False)
```

C
```
cust_data.drop('Job Category', axis=0, inplace=False)
```

D
```
cust_data.drop('Job Category', axis=1, inplace=True)
```

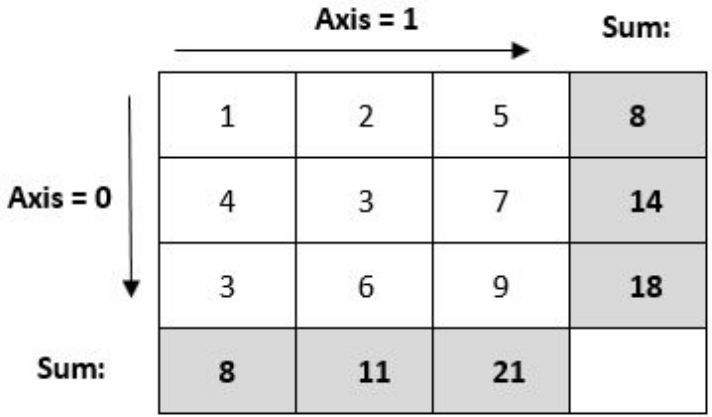
Pandas Functions

Function	Syntax	Example	Description
df.drop()	`df.drop(labels, axis, inplace)`	`df.drop('Job Category', axis=1, inplace=True)`	Drop specified labels from rows or columns

inplace=True: Modifies a dataframe directly, avoids creating a copy of the original dataframe

axis=0: Performs operations row-wise

axis=1: Performs operations column-wise



NumPy and Pandas Quiz

Consider a dataframe df having the columns Gender and Height. Which of the following can be used to get the average height by different categories of gender?

A
```
df.groupby(['Height'])['Gender'].mean()
```

B
```
df.groupby(['Gender']).Height.mean()
```

C
```
df.groupby(['Gender'])['Height'].mean
```

D
```
df.groupby(['Gender'])['Height'].mean()
```

NumPy and Pandas Quiz

Consider a dataframe df having the columns Gender and Height. Which of the following can be used to get the average height by different categories of gender?

A
```
df.groupby(['Height'])['Gender'].mean()
```

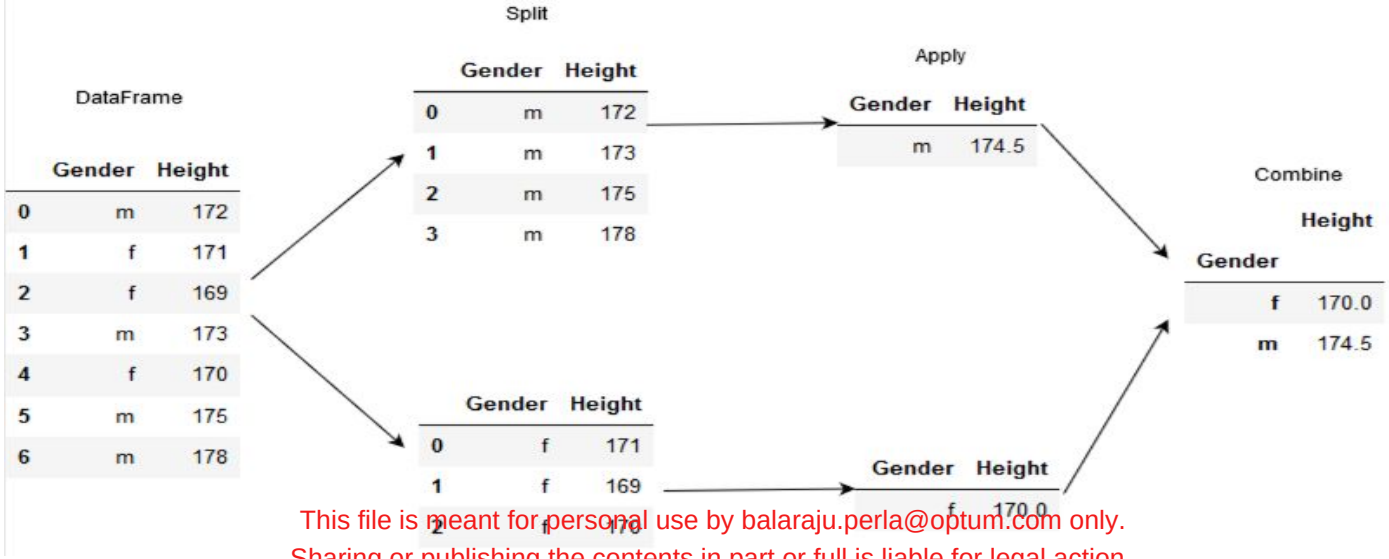
B
```
df.groupby(['Gender']).Height.mean()
```

C
```
df.groupby(['Gender'])['Height'].mean
```

D
```
df.groupby(['Gender'])['Height'].mean()
```

Pandas Functions

Function	Syntax	Example	Description
df.groupby()	`df.groupby(['column_name'])[aggregate_column].agg_func()`	`df.groupby(['Gender'])['Height'].mean()`	To split, apply and combine the data structures to get aggregated values wrt attribute(s)



NumPy and Pandas Quiz

Great Learning
POWER AHEAD

Consider two dataframes df1 and df2 containing a common column Cust_ID. Which of the following code snippets will merge these two dataframes?

A
```
pd.merge(df1, df2, on='Cust_ID', how='inner')
```

B
```
pd.merge(df1, on='Cust_ID', how='inner')
```

C
```
df1.merge(df2, on='Cust_ID', how='inner')
```

D
```
pd.merge(df1, df2, on='Cust_ID')
```

NumPy and Pandas Quiz

Great Learning
POWER AHEAD

Consider two dataframes df1 and df2 containing a common column Cust_ID. Which of the following code snippets will merge these two dataframes?

A
```
pd.merge(df1, df2, on='Cust_ID', how='inner')
```

B
```
pd.merge(df1, on='Cust_ID', how='inner')
```

C
```
df1.merge(df2, on='Cust_ID', how='inner')
```

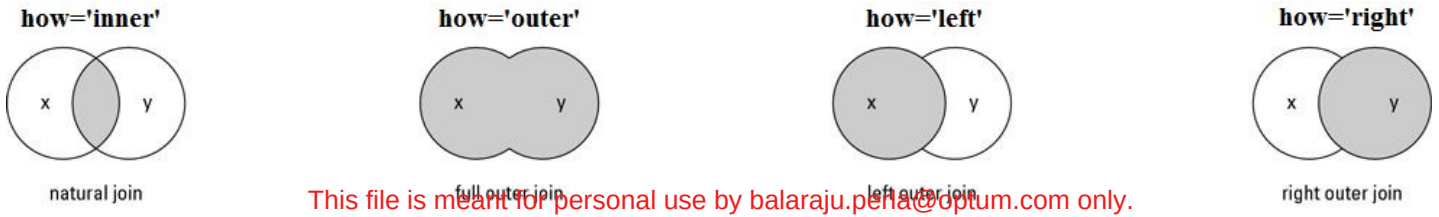
D
```
pd.merge(df1, df2, on='Cust_ID')
```

Pandas - Merge and Join

join - works best when joining dataframes on their indices (though you can specify another column to join on)

merge - more versatile and allows to specify columns (besides the index) to join on

how='inner'	how='outer'	how='left'	how='right'
Retains only the rows that are common between the dataframes	Retains all the rows from both the dataframes	Retains all the rows from the first dataframe and only the matching ones from the second	Retains all the rows from the second dataframe and only the matching ones from the first



natural join

full outer join

left outer join

right outer join

Pandas - Merge Example

LEFT JOIN

Syntax: `merged = pd.merge (left, right, on = 'Customer id', how = 'left')`

Index	Customer_id	Product
0	1	Oven
1	2	Oven
2	3	Oven
3	4	Television
4	5	Television
5	6	Television

Left Table

Index	Customer_id	State
0	2	California
1	4	California
2	6	Texas
3	7	Las Vegas
4	8	Las Vegas

Right Table

Index	Customer_id	Product	State
0	1	Oven	NaN
1	2	Oven	California
2	3	Oven	NaN
3	4	Television	California
4	5	Television	NaN
5	6	Television	Texas

After left join on Customer_id

RIGHT JOIN

Syntax: `merged = pd.merge (left, right, on = 'Customer id', how = 'right')`

Index	Customer_id	Product
0	1	Oven
1	2	Oven
2	3	Oven
3	4	Television
4	5	Television
5	6	Television

Left Table

Index	Customer_id	State
0	2	California
1	4	California
2	6	Texas
3	7	Las Vegas
4	8	Las Vegas

Right Table

Index	Customer_id	Product	State
0	2	Oven	California
1	4	Television	California
2	6	Television	Texas
3	7	NaN	Las Vegas
4	8	NaN	Las Vegas

After right join on Customer_id

Pandas - Merge Example

INNER JOIN

Syntax: `merged = pd.merge (left, right, on = 'Customer id')`

Index	Customer_id	Product
0	1	Oven
1	2	Oven
2	3	Oven
3	4	Television
4	5	Television
5	6	Television

Left Table

Index	Customer_id	State
0	2	California
1	4	California
2	6	Texas
3	7	Las Vegas
4	8	Las Vegas

Right Table

Index	Customer_id	Product	State
0	2	Oven	California
1	4	Television	California
2	6	Television	Texas

After inner join on Customer_id

OUTER JOIN

Syntax: `merged = pd.merge (left, right, on = 'Customer id', how = 'outer')`

Index	Customer_id	Product
0	1	Oven
1	2	Oven
2	3	Oven
3	4	Television
4	5	Television
5	6	Television

Left Table

Index	Customer_id	State
0	2	California
1	4	California
2	6	Texas
3	7	Las Vegas
4	8	Las Vegas

Right Table

Index	Customer_id	Product	State
0	1	Oven	NaN
1	2	Oven	California
2	3	Oven	NaN
3	4	Television	California
4	5	Television	NaN
5	6	Television	Texas
6	7	NaN	Las Vegas
7	8	NaN	Las Vegas

After outer join on Customer_id

NumPy and Pandas Quiz

Great Learning
POWER AHEAD

Consider a file Customer_Data.csv that contains multiple attributes of 100 customers. Which of the following can be used to load the file into a pandas dataframe?

A
```
pd.read_csv(Customer_Data.csv)
```

B
```
pd.read_csv("Customer_Data.csv")
```

C
```
pd.read_csv('Customer_Data.csv')
```

D
```
pd.read_csv(Customer_Data)
```

NumPy and Pandas Quiz

Great Learning
POWER AHEAD

Consider a file Customer_Data.csv that contains multiple attributes of 100 customers. Which of the following can be used to load the file into a pandas dataframe?

A
```
pd.read_csv(Customer_Data.csv)
```

B
```
pd.read_csv("Customer_Data.csv")
```

C
```
pd.read_csv('Customer_Data.csv')
```

D
```
pd.read_csv(Customer_Data)
```

Loading Datasets in Pandas

read_csv - pandas function used to load datasets in CSV format into a pandas dataframe

Syntax: `df = pd.read_csv("file_name.csv")`

Pandas has to be imported with alias pd - `import pandas as pd`

The file name has to be enclosed in quotation marks (single or double)

Above syntax works when the file (dataset) is in the same working directory as the Python notebook

When the file (dataset) and the Python notebook are not in the same working directory, the path to the file has to be specified

Great Learning

Happy Learning !