Pandas Basic Assignment 19

October 25, 2023

Q1. Create a Pandas Series that contains the following data: 4, 8, 15, 16, 23, and 42. Then, print the series.

[1]:
```
pip install pandas
```

```
Requirement already satisfied: pandas in /opt/conda/lib/python3.10/site-packages
(1.5.2)
Requirement already satisfied: python-dateutil>=2.8.1 in
/opt/conda/lib/python3.10/site-packages (from pandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /opt/conda/lib/python3.10/site-
packages (from pandas) (2022.6)
Requirement already satisfied: numpy>=1.21.0 in /opt/conda/lib/python3.10/site-
packages (from pandas) (1.23.5)
Requirement already satisfied: six>=1.5 in /opt/conda/lib/python3.10/site-
packages (from python-dateutil>=2.8.1->pandas) (1.16.0)
Note: you may need to restart the kernel to use updated packages.
```

[1]:
```
import pandas as pd
```

[2]:
```
data = [4, 8, 15, 16, 23, 42]
my_series = pd.Series(data)
```

[3]:
```
print(my_series)
```

```
0     4
1     8
2    15
3    16
4    23
5    42
dtype: int64
```

[ ]:
```

```

Q2. Create a variable of list type containing 10 elements in it, and apply pandas.Series function on the variable print it.

[10]:
```
import pandas as pd
my_list=[2,4,6,885,67,56,446,856,4,6]
my_series=pd.Series(my_list)
print(my_series)
```

```
0      2
1      4
2      6
3    885
4     67
5     56
6    446
7    856
8      4
9      6
dtype: int64
```

[ ]:

Q3. Create a Pandas DataFrame that contains the following data:

[22]:
```
import pandas as pd
df=pd.DataFrame({'Name':['Alice','Bob','Claire'],'Age':
  ↪['25','30','27'],'Gender':['Female','Male','Female']})
df
```

[22]:

	Name	Age	Gender
0	Alice	25	Female
1	Bob	30	Male
2	Claire	27	Female

[ ]:

Q4. What is 'DataFrame' in pandas and how is it different from pandas.series? Explain with an example.

ANS:

0.1 Pandas DataFrame vs. Series

Pandas Series: - A Pandas Series is a one-dimensional labeled array that can hold data of any data type. - It's similar to a single column in a spreadsheet and is equipped with an index. - Useful for representing a single variable or one-dimensional data. - Example: A series of temperatures for a week.

Pandas DataFrame: - A Pandas DataFrame is a two-dimensional data structure, like a table or spreadsheet. - It consists of multiple columns, each of which is a Pandas Series. - Suitable

for structured, tabular data with multiple variables or attributes. - Example: A table containing information about people, with columns for 'Name,' 'Age,' and 'City.'

In summary, Pandas Series are for one-dimensional data, while Pandas DataFrames are for structured, tabular data with multiple attributes. DataFrames are widely used for data manipulation and analysis in data science.

[27]:
```
import pandas as pd

# Create a Pandas Series for daily temperatures
temperatures = pd.Series([72, 74, 75, 73, 70, 72, 76], name="Temperature")

# Print the Series
print("Pandas Series:\n",temperatures)
```

```
Pandas Series:
 0    72
1    74
2    75
3    73
4    70
5    72
6    76
Name: Temperature, dtype: int64
```

[29]:
```
import pandas as pd

# Create a Pandas DataFrame
data = {
    'Name': ['Alice', 'Bob', 'Charlie', 'David'],
    'Age': [25, 30, 35, 28],
    'City': ['New York', 'Los Angeles', 'Chicago', 'San Francisco']
}

df = pd.DataFrame(data)

# Print the DataFrame
print('DataFrame:\n',df)
```

```
DataFrame:
       Name  Age           City
0    Alice   25       New York
1      Bob   30    Los Angeles
2  Charlie   35        Chicago
3    David   28  San Francisco
```

[ ]:
```

```

Q5. What are some common functions you can use to manipulate data in a Pandas DataFrame?

Can you give an example of when you might use one of these functions?

ANS:

Pandas provides a wide range of functions that can be used to manipulate data in a DataFrame. Some common functions include:

- head() and tail(): to view the first or last n rows of a DataFrame
- describe(): to view the statistical summary of the DataFrame
- shape: to view the number of rows and columns in the DataFrame
- drop(): to remove rows or columns from the DataFrame
- groupby(): to group rows based on a column and apply a function to each group
- sort_values(): to sort the DataFrame by one or more columns
- fillna(): to fill missing values in the DataFrame with a specified value or method
- apply(): to apply a function to each element of a DataFrame or a Series
- merge(): to join two or more DataFrames based on a common column or index

Here's an example of when you might use one of these functions. Suppose you have a DataFrame containing information about employees in a company, and you want to view the statistical summary of their salaries:

[31]:
```
import pandas as pd

employee_data = {
    'Name': ['John', 'Jane', 'Bob', 'Alice', 'Mike'],
    'Age': [25, 30, 35, 40, 45],
    'Salary': [50000, 60000, 70000, 80000, 90000]
}
df = pd.DataFrame(employee_data)

# Using the describe() function to view the statistical summary of the Salary␣
 ↪column
print(df['Salary'].describe())
```

```
count        5.000000
mean     70000.000000
std      15811.388301
min      50000.000000
25%      60000.000000
50%      70000.000000
75%      80000.000000
max      90000.000000
Name: Salary, dtype: float64
```

[ ]:
```

```

Q6. Which of the following is mutable in nature Series, DataFrame, Panel?

ANS:

Among the three data structures provided by Pandas, only the DataFrame and Panel are mutable

in nature.

- A DataFrame is mutable because you can add, remove or modify columns and rows. Similarly, a Panel is mutable because you can add or remove items along the axis.
- On the other hand, a Series is immutable because it represents a single column of data with an index. Once created, you cannot add or remove elements from a Series. However, you can modify the values of existing elements in a Series.

[ ]:

Q7. Create a DataFrame using multiple Series. Explain with an example.

[32]:
```
import pandas as pd

Name=pd.Series(['Varun','Alia','Mandeep','Suman'])
Age=pd.Series(['35','29','32','25'])
Gender=pd.Series(['Male','Female','Male','Female'])

df=pd.DataFrame({'Name':Name,'Age':Age,'Gender':Gender})

df
```

[32]:

	Name	Age	Gender
0	Varun	35	Male
1	Alia	29	Female
2	Mandeep	32	Male
3	Suman	25	Female

[ ]: