## **Exploring the Dataset with Pandas - Part 2**

Welcome back to using Pandas. In this part, we'll continue our analysis using various Pandas functions and techniques, here we use the Titanic dataset for exploration of the data analysis.

### **Recap**

Before diving in, let's quickly recap what we've done so far. We've loaded the Titanic dataset into a Pandas DataFrame, examined its structure, data types, and performed basic data exploration.

import pandas as pd

df=pd.read\_csv("https://raw.githubusercontent.com/datasciencedojo[/datasets/master/titanic.csv](https://colab.research.google.com/drive/1dZ5Mzhb1zX2OgMKZ_9uF0Z3Z3nmhWjz3?authuser=0#)")

# Displaying the first few rows of the DataFrame

df.head()

# Checking the type of DataFrame

type(df)

# Checking data types of columns

df.dtypes

# Generating descriptive statistics

df.describe()

### Selecting Specific Columns

Often, we're interested in analyzing only specific columns. We can select these columns using indexing or the loc and iloc functions.

# Selecting specific columns

df[['Name','Sex','Ticket','Cabin','Embarked']]

# Using dtypes to filter for object-type columns

df[df.dtypes[df.dtypes == 'object'].index]

# Descriptive statistics for object-type columns

df[df.dtypes[df.dtypes == 'object'].index].describe()

## **Now Start to**

## **Exploring Pandas Series**

In this blog post, we'll delve into the Pandas library and explore Series, one of its fundamental data structures. We'll go through various operations and methods to understand Series better.

### **Series**

A Pandas Series is a one-dimensional labeled array capable of holding data of any type. It consists of two arrays: one containing the data (values) and the other containing the corresponding labels (index).

Let's start by creating a Series from the 'Name' column of the Titanic dataset.

import pandas as pd

# Reading the Titanic dataset into a DataFrame

df = pd.read\_csv("https://raw.githubusercontent.com/datasciencedojo[/datasets/master/titanic.csv](https://colab.research.google.com/drive/1dZ5Mzhb1zX2OgMKZ_9uF0Z3Z3nmhWjz3?authuser=0#)")

# Displaying the column names

df.columns

# Selecting the first 10 elements of the 'Name' column

s = df['Name'][0:10]

# Displaying the Series

### Basic Operations on Series

Let's explore some basic operations on Series:

# Length of the Series

len(s)

# Type of the Series

type(s)

# Accessing elements of the Series

s[0]

# Creating a new Series with custom index

l = ['sudh','b','c','d','e','f','g','h','i','j']

s1 = pd.Series(list(s), index=l)

s1

### Combining Series

We can combine two or more Series using the append() method.

# Combining two Series

s2 = s1.append(s)

s2

### Mathematical Operations on Series

Series support mathematical operations, and they align the data based on the index labels.

# Creating two Series

s4 = pd.Series([3, 4, 5, 6, 6], index=[2, 4, 5, 6, 1])

s5 = pd.Series([34, 345, 45, 45, 454], index=[9, 4, 5, 6, 7])

# Combining two Series

s6 = s4.append(s5)

s6

# Accessing elements of the combined Series

s6[4]

# Slicing the combined Series

s6[0:5]

# Performing mathematical operations

s4 \* s5

s4 + s5

we've explored Pandas Series and learned about their creation, basic operations, combining multiple Series, and performing mathematical operations. Series are a powerful and versatile data structure in Pandas, and they play a crucial role in data manipulation and analysis.

### 

**Pandas provides two primary accessors for selecting subsets of data:**

* **iloc**: Integer-location based indexing.
* **loc**: Label-based indexing.

#### **iloc**

iloc is primarily integer-location based, meaning it is used to select data based on the numerical index of rows and columns.

# Selecting rows 0 to 2 and columns with integer indexes 0, 1, and 2

df.iloc[0:2, [0, 1, 2]]

# Selecting rows 0 to 2 and columns with labels 'PassengerId', 'Survived', and 'Pclass

df.iloc[0:2, df.columns.get\_loc('PassengerId'):df.columns.get\_loc('Pclass')+1]

#### loc

loc, on the other hand, is label-based indexing. It selects data based on the labels of rows and columns.

# Selecting rows 0 to 2 and columns 'PassengerId', 'Survived', and 'Pclass'

df.loc[0:2, ['PassengerId', 'Survived', 'Pclass']]

# Selecting rows with labels 0 to 2 and columns with labels 'PassengerId', 'Survived', and 'Pclass'

df.loc[0:2, 'PassengerId':'Pclass']

In this part of our exploration, we've learned more about using Pandas for data analysis, including iloc and loc for selecting specific subsets of data. These techniques are fundamental for exploring and understanding any dataset using Pandas.

Stay tuned for more insights and analysis!