Experiment Number 6

Aim: Performance of exploratory data analysis on DataFrame

Theory:

Exploratory data analysis (EDA) is used by data scientists to analyze and investigate data sets and summarize their main characteristics, often employing data visualization methods.

EDA helps determine how best to manipulate data sources to get the answers you need, making it easier for data scientists to discover patterns, spot anomalies, test a hypothesis, or check assumptions.

EDA is primarily used to see what data can reveal beyond the formal modeling or hypothesis testing task and provides a provides a better understanding of data set variables and the relationships between them. It can also help determine if the statistical techniques you are considering for data analysis are appropriate.

Why is EDA important in data science?

The main purpose of EDA is to help look at data before making any assumptions. It can help identify obvious errors, as well as better understand patterns within the data, detect outliers or anomalous events, find interesting relations among the variables.

Data scientists can use exploratory analysis to ensure the results they produce are valid and applicable to any desired business outcomes and goals. EDA also helps stakeholders by confirming they are asking the right questions. EDA can help answer questions about standard deviations, categorical variables, and confidence intervals. Once EDA is complete and insights are drawn, its features can then be used for more sophisticated data analysis or modeling, including machine learning.

Types of EDA

There are four primary types of EDA:

- **Univariate non-graphical**
- **Univariate graphical**
- **Multivariate non-graphical**
- **Multivariate graphical**

Univariate non-graphical

This is simplest form of data analysis, where the data being analyzed consists of just one variable. Since it's a single variable, it doesn't deal with causes or relationships. The main purpose of univariate analysis is to describe the data and find patterns that exist within it.

Univariate graphical

Non-graphical methods don't provide a full picture of the data. Graphical methods are therefore required. Common types of univariate graphics include:

- Stem-and-leaf plots, which show all data values and the shape of the distribution.

- Histograms, a bar plot in which each bar represents the frequency (count) or proportion (count/total count) of cases for a range of values.
- Box plots, which graphically depict the five-number summary of minimum, first quartile, median, third quartile, and maximum.

Multivariate nongraphical

Multivariate data arises from more than one variable. Multivariate non-graphical EDA techniques generally show the relationship between two or more variables of the data through cross-tabulation or statistics.

Multivariate graphical

Multivariate data uses graphics to display relationships between two or more sets of data. The most used graphic is a grouped bar plot or bar chart with each group representing one level of one of the variables and each bar within a group representing the levels of the other variable.

Other common types of multivariate graphics include:

- Scatter plot, which is used to plot data points on a horizontal and a vertical axis to show how much one variable is affected by another.
- Multivariate chart, which is a graphical representation of the relationships between factors and a response.
- Run chart, which is a line graph of data plotted over time.
- Bubble chart, which is a data visualization that displays multiple circles (bubbles) in a two-dimensional plot.
- Heat map, which is a graphical representation of data where values are depicted by color.

Basically EDA Consist of Following steps:

1. Understand the Data

- **Load the Dataset**: Use Pandas to load the titanic.csv file.
- **Inspect the Data**: Use head() to view the first few rows and info() or describe() to understand the columns, data types, and summary statistics.
- **Identify Variables**: Understand what each column represents (e.g., PassengerId, Survived, Pclass, Name, Sex, Age, Fare, Cabin, Embarked).

Code Example Python:

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
# Load the datasets
df = pd.read_csv('titanic.csv')
# Display the first few rows of the training data
```

```
print(df.head())

# Get information about the dataset
print(df.info())
```

2. Handle Missing Values

- **Identify Missing Data**: Use isnull().sum() to find columns with missing values.
- **Impute Missing Values**: Decide how to fill missing values (e.g., using the mean or median for numerical columns like Age, or the mode for categorical columns like Embarked).

Code Example Python

```
# Check for missing values
print(df.isnull().sum())
# Fill missing Age values with the median age
df['Age'].fillna(df['Age'].median(), inplace=True)
# Fill missing Embarked values with the most frequent port
df['Embarked'].fillna(df['Embarked'].mode()[0], inplace=True)
```

3. Analyze Categorical Variables

- **Bar Plots for Survival**:

Use Seaborn's countplot to visualize the count of survivors and non-survivors.

- **Compare Categories**:

Analyze the survival rates across different categories like gender (Sex) and passenger class (Pclass).

Code Example Python

```
# Survival by Sex
sns.countplot(x='Sex', hue='Survived', data=df)
plt.title('Survival by Sex')
plt.show()
# Survival by Pclass
sns.countplot(x='Pclass', hue='Survived', data=df)
plt.title('Survival by Pclass')
plt.show()
```

4. Analyze Numerical Variables

- **Histograms and Distributions**: Use histograms to see the distribution of Age and Fare.
- **Violin Plots**: Create violin plots to understand the distribution of Age across different Pclass and Sex.

Code Example Python

```
# Distribution of Age
sns.histplot(df['Age'], kde=True)
plt.title('Distribution of Age')
plt.show()
# Age distribution by Sex and Pclass
sns.violinplot(x='Pclass', y='Age', hue='Sex', data=df)
plt.title('Age Distribution by Pclass and Sex')
plt.show()
```

5. Find Relationships Between Features

- **Correlation Heatmap**:

Create a correlation heatmap to visualize the relationships between numerical features.

- **Pair Plots**:

Use pairplot to visualize pairwise relationships between numerical variables.

Code Example Python

```
# Correlation heatmap of numerical features
sns.heatmap(df.corr(numeric_only=True), annot=True, cmap='coolwarm')
plt.title('Correlation Heatmap')
plt.show()
```

Diagram of each code if possible

Conclusion: In this way EDA is performed on Titanic Dataset