EDA

1. Display data structure – head()
2. Descriptive statistic – describe()
   1. Mean
   2. Standard deviation
   3. Median
   4. Mode
   5. Quantiles
   6. Percentiles
3. Univariate analysis – visualization(histograms, box plots and bar charts)
4. Multivate analysis – relationships between variables(scatterplot, heatmap)
5. Correlation – measure that describe strength of relationship between variables
6. Correlation matrix – correlation coefficients between variables
7. Pairwise relationship – pairplot()

**Statistics on variables** ->

import pandas as pd

import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt

df = pd.read\_csv("train.csv")

df.head()

df.info()

df.describe()

**missing variables**

df.isnull().sum()

**heatmap for missing variables**

import seaborn as sns

import matplotlib.pyplot as plt

plt.figure(figsize=(10, 6))

sns.heatmap(df.isnull(), cmap="viridis", cbar=False)

plt.show()

**distribution of target variable**

1. **Check unique values in target variable**

df["Survived"].value\_counts()

1. **Visualizing target variable distribution**

import seaborn as sns

import matplotlib.pyplot as plt

sns.countplot(x="Survived", data=df, palette="coolwarm")

plt.title("Distribution of Survival (Target Variable)")

plt.xlabel("Survived (0 = No, 1 = Yes)")

plt.ylabel("Count")

plt.show()

1. **Pie chart**

df["Survived"].value\_counts().plot.pie(autopct="%1.1f%%", colors=["red", "green"])

plt.title("Survival Distribution")

plt.ylabel("")

plt.show()

1. **Histogram**

sns.histplot(df["Fare"], bins=30, kde=True)

plt.title("Fare Distribution")

plt.show()

**correlation between the variables**

1. **Computer correlation matrix**

import pandas as pd

correlation\_matrix = df.corr()

print(correlation\_matrix)

1. **Visualize correlation with heatmap**

import seaborn as sns

import matplotlib.pyplot as plt

plt.figure(figsize=(10, 6))

sns.heatmap(correlation\_matrix, annot=True, cmap="coolwarm", fmt=".2f", linewidths=0.5)

plt.title("Correlation Heatmap")

plt.show()

1. **Identify correlated features**

high\_corr\_pairs = correlation\_matrix.unstack().sort\_values(ascending=False)

high\_corr\_pairs = high\_corr\_pairs[high\_corr\_pairs < 1]

print(high\_corr\_pairs.head(10)) # Show top 10 correlated pairs