Handle Missing Values in Dataset

Author: [Muhammad Umar]

Date: [Date]

Section # 1: Data Visualization

HISTROGRAM PLOT OF AGE (DISTRIBUTION CHECK)

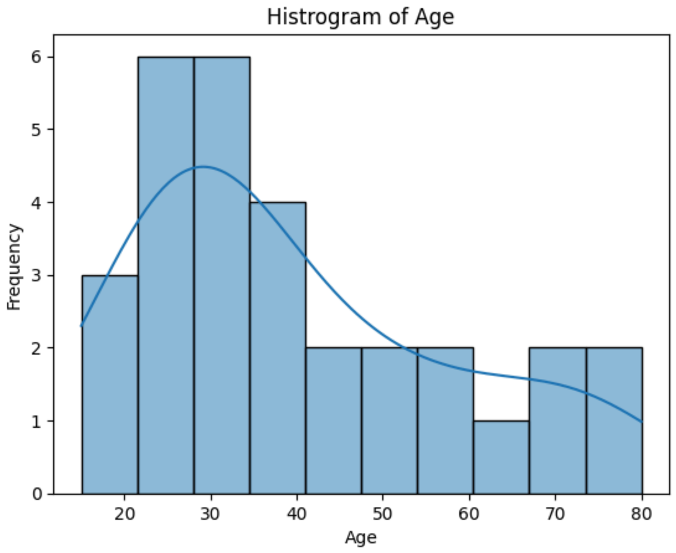
```
# import Libraries
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd

# Dummy data set
ages = [15, 18, 20, 22, 23, 25, 25, 26, 27, 28, 29, 30, 30, 31, 32, 35, 36, 38,
        40, 42, 45, 50, 52, 55, 60, 65, 70, 72, 75, 80 ]

df=pd.DataFrame({"age": ages})

# Histrogram with KDE

sns.histplot(df["age"], kde=True , bins=10)
plt.title("Histrogram of Age")
plt.xlabel("Age")
plt.ylabel("Frequency")
plt.show()
```



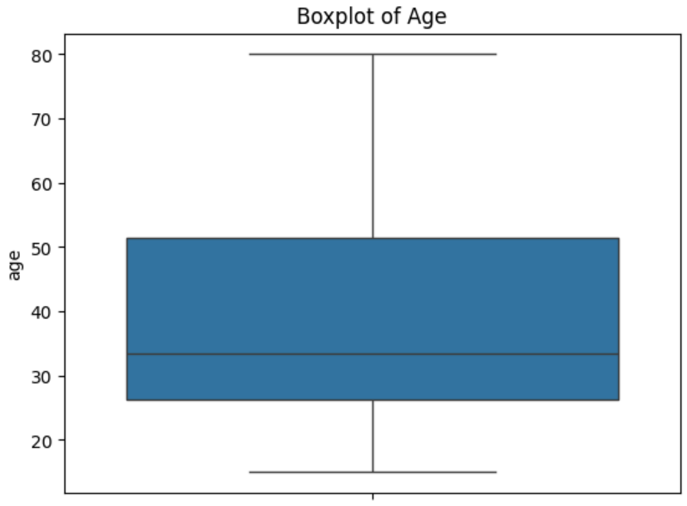
BOX PLOT OF AGE (OUTLIERS DETECTION)

```
# import libraries
import seaborn as sns
import matplotlib.pyplot as plt
import pandas as pd

# Dummy data set
ages = [15, 18, 20, 22, 23, 25, 25, 26, 27, 28, 29, 30, 30, 31, 32, 35, 36, 38,
        40, 42, 45, 50, 52, 55, 60, 65, 70, 72, 75, 80 ]

df=pd.DataFrame({"age": ages})
# boxplot

sns.boxplot(df["age"])
plt.title("Boxplot of Age")
plt.show()
```



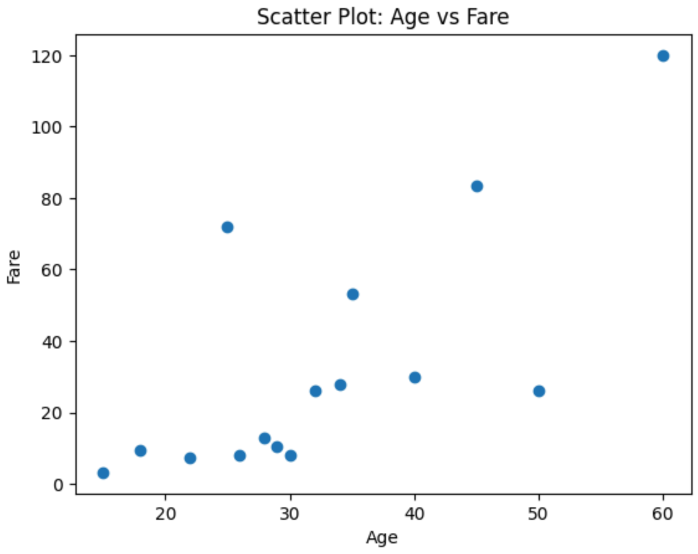
SCATTER PLOT OF AGE AND FARE (OUTLIERS DETECTION)

```
# import libraries
import pandas as pd
import matplotlib.pyplot as plt

# dammy data set
ages = [22, 25, 30, 35, 40, 28, 26, 32, 34, 29, 45, 50, 18, 60, 15]
fares =[7.25, 71.83, 8.05, 53.10, 30.00, 13.00, 7.90, 26.00, 27.72, 10.50,
             83.47, 25.93, 9.35, 120.00, 3.17]

df = pd.DataFrame({
    "Age": ages,
    "Fare": fares
})

# scatter plot with KDE
plt.scatter(df['Age'], df['Fare'])
plt.title("Scatter Plot: Age vs Fare")
plt.xlabel("Age")
plt.ylabel("Fare")
plt.show()
```



HEAT-MAP OF AGE,FARE,INCOME

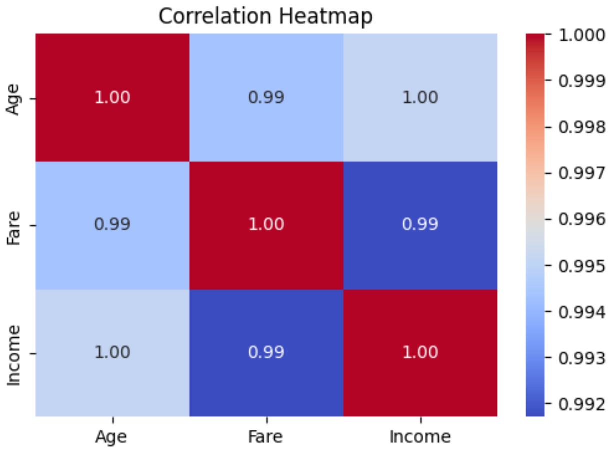
```
# import libraries
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Dummy dataset
data = {
    "Age": [15, 18, 20, 22, 23, 25, 26, 28, 30, 35, 40, 45, 50, 55, 60],
    "Fare": [5, 7, 10, 15, 20, 25, 30, 35, 40, 50, 60, 75, 90, 110, 130],
    "Income": [100, 200, 250, 300, 350, 400, 450, 470, 500, 600, 700, 800, 900, 1000, 1200]
}

df = pd.DataFrame(data)

# Correlation matrix
corr = df.corr()

# Heatmap
plt.figure(figsize=(6,4))
sns.heatmap(corr, annot=True, cmap="coolwarm", fmt=".2f")
plt.title("Correlation Heatmap")
plt.show()
```



SECTION # 2: Handling Missing Values (Mean, Median, Mode)

NOW WE HANDLE MISSING VALUES WITH (MEAN)

```
import pandas as pd
import numpy as np


# dummy Data set:

data=pd.DataFrame({"age" : [10,12,14,13,15,14,19,18,20,25,28,np.nan,29,30,np.nan,33,35]})

# print the data with missing values
print("------------------------------")
print(f"Here is the data with Missing Values:\n {data}")

# Calculate the Mean
mean=data["age"].mean()

# replace the missing values with mean
data["age"]=data["age"].fillna(mean)

# now print the data without missing values
print("------------------------------------")
print(f"Here is the data without missing values:\n {data}")
```

```
------------------------------
Here is the data with Missing Values:
      age
0   10.0
1   12.0
2   14.0
3   13.0
4   15.0
5   14.0
6   19.0
7   18.0
8   20.0
9   25.0
10  28.0
11   NaN
12  29.0
13  30.0
14   NaN
15  33.0
16  35.0
------------------------------------
Here is the data without missing values:
      age
0   10.0
1   12.0
2   14.0
3   13.0
4   15.0
5   14.0
6   19.0
7   18.0
8   20.0
9   25.0
10  28.0
11  21.0
12  29.0
13  30.0
14  21.0
15  33.0
16  35.0
```

NOW WE HANDLE MISSING VALUES WITH (MEDIAN)

```
import pandas as pd
import numpy as np

# Dummy data set:
data=pd.DataFrame({"age" : [10,12,14,13,15,14,19,18,20,25,28,np.nan,29,30,np.nan,33,35]})

# print the data with missing values:
print("------------------------------")
print(f"Here is the data with Missing Values:\n {data}")

# Calculate the medean
median=data["age"].median()

# replace the missing values with median
data["age"]=data["age"].fillna(median)

# now print the data without missing values:
print("------------------------------------")
print(f"Here is the data without missing values:\n {data}")
```

```
------------------------------
Here is the data with Missing Values:
      age
0   10.0
1   12.0
2   14.0
3   13.0
4   15.0
5   14.0
6   19.0
7   18.0
8   20.0
9   25.0
10  28.0
11   NaN
12  29.0
13  30.0
14   NaN
15  33.0
16  35.0
------------------------------------
Here is the data without missing values:
      age
0   10.0
```

```
1   12.0
2   14.0
3   13.0
4   15.0
5   14.0
6   19.0
7   18.0
8   20.0
9   25.0
10  28.0
11  19.0
12  29.0
13  30.0
14  19.0
15  33.0
16  35.0
```

NOW WE HANDLE MISSING VALUES WITH (MODE)

```
# import libraries:
import pandas as pd
import numpy as np

# dummy data set:
data=pd.DataFrame({"fruits" : ["Apple", "Banana", "Orange", "Mango",np.nan, "Pineapple", "Grapes",np.nan, "Strawberry","Apple"]})

# print the data with missing values:
print("----------------------------")
print(f"Here is the data with missing values: \n {data}")

#calculate mode:
mode=data["fruits"].mode()[0]

#replace the missing values with mode:
data["fruits"]=data["fruits"].fillna(mode)

#now print the data without missing values:
print("-----------------------------------")
print(f"Here is the data without missing values: \n {data}")
```

```
----------------------------
Here is the data with missing values: 
        fruits
0       Apple
1      Banana
2      Orange
3       Mango
4         NaN
5   Pineapple
6      Grapes
7         NaN
8  Strawberry
9       Apple
-----------------------------------
Here is the data without missing values: 
        fruits
0       Apple
1      Banana
2      Orange
3       Mango
4       Apple
5   Pineapple
6      Grapes
7       Apple
8  Strawberry
9       Apple
```

CHECK THE MISSING VALUES IN TITANIC DATA:

```
import pandas as pd
import numpy as np
import seaborn as sns

df = sns.load_dataset("titanic")
df.isnull().sum().sort_values(ascending=False)
```

	0
deck	688
age	177
embarked	2
embark_town	2
sex	0
pclass	0
survived	0
fare	0
parch	0
sibsp	0
class	0
adult_male	0
who	0
alive	0
alone	0

dtype: int64

SECTION # 3: Handling Missing Values (KNN & Iterative Imputer)

HANDLE MISSING VALUES WITH KNN IMPUTER (USE FOR ONE BY ONE COLUMN)

```
from sklearn.impute import KNNImputer

imputer= KNNImputer(n_neighbors=4)

df["age"] = imputer.fit_transform(df[["age"]])

df.isnull().sum().sort_values(ascending=False)
```

```
                 0
    survived     0
      pclass     0
         sex     0
         age     0
       sibsp     0
       parch     0
        fare     0
    embarked     0
       class     0
         who     0
  adult_male     0
        deck     0
 embark_town     0
       alive     0
       alone     0

dtype: int64
```

LABEL ENCODING (CONVERT CATEGORICAL TO NUMBER)

```
from sklearn.preprocessing import LabelEncoder

# columns to encode:
columns_to_encode = ["sex","embarked","who","deck","class","embark_town","alive"]

#Dictionary to store labelencoders for each column
label_encoders = {}

#loop to apply labelencoder to each colum for encoding
for col in columns_to_encode:


    # fit and transform the data:
    df[col] = LabelEncoder().fit_transform(df[col])
    # store the encoder in the dictionary
    label_encoders[col] = LabelEncoder()
df.head
```

HANDLE MISSING VALUES WITH ITERATIVE IMPUTER:

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.experimental import enable_iterative_imputer
from sklearn.impute import IterativeImputer

imputer = IterativeImputer(max_iter=10)

#columns to impute
columns_to_impute=["age","pclass","deck"]

# loop to impute each colum

for col in columns_to_impute:
    df[columns_to_impute] = imputer.fit_transform(df[columns_to_impute])


# check the missing values
df.isnull().sum().sort_values(ascending=False)
```

Handle Missing Values – Data Cleaning Project

Overview:

In this project, I explored different techniques to handle missing values in a dataset. This is my first data analysis project, completed in Google Colab and shared on GitHub as part of my learning journey.

Objectives:

Visualize data distribution and detect outliers (Histogram, Boxplot, Scatter Plot, Heatmap)

Handle missing values using:

Mean, Median, Mode

KNN Imputer

Iterative Imputer

Apply Label Encoding for categorical columns

Visualizations:

Histogram ⟶ to check data distribution

Boxplot ⟶ to detect outliers

Scatter Plot ⟶ to understand relationships between variables

Heatmap ⟶ to check correlations

Techniques Used:

Simple Imputation – Mean, Median, Mode

KNN Imputer – fills missing values based on similar rows

Iterative Imputer – advanced, model-based imputation

Encoding – converting categorical data into numeric format

Conclusion:

For small datasets, Mean/Median/Mode are simple and fast solutions.

Median works better when outliers are present.

For complex datasets, KNN and Iterative Imputer provide better results.

Handling missing values is a fundamental step in data cleaning that directly improves machine learning model accuracy.

Files in this Repository:

Handle Missing Values.ipynb ⟶ Colab Notebook

Handle Missing Values.pdf ⟶ Report (PDF format)

Author

Muhammad Umar