Name: Sehar NAWAZ

RollNo: 13

Assignment:no:02

Task # 01

Step:01

Generate a database having 10 features and you have generate the random values for about 10,000 instances using random values variable.Write a python program that delete some other values.Write the python program to check the missing values.Fill the missing values using the average value of that feature?

```python
import numpy as np
import pandas as pd

np.random.seed(42)

data = np.random.rand(10000, 10)

columns = [f'Feature_{i+1}' for i in range(10)]
df = pd.DataFrame(data, columns=columns)

for col in df.columns:
    df.loc[df.sample(frac=0.05).index, col] = np.nan

print("Missing values before filling:")
print(df.isnull().sum())

df_filled = df.fillna(df.mean())

print("\nMissing values after filling:")
print(df_filled.isnull().sum())

```

```
Missing values before filling:
Feature_1     500
Feature_2     500
Feature_3     500
Feature_4     500
Feature_5     500
Feature_6     500
Feature_7     500
Feature_8     500
Feature_9     500
Feature_10    500
dtype: int64

Missing values after filling:
Feature_1     0
Feature_2     0
Feature_3     0
Feature_4     0
Feature_5     0
Feature_6     0
Feature_7     0
Feature_8     0
Feature_9     0
Feature_10    0
dtype: int64
```

step:02

Write a program that delete some other values

```python
import numpy as np
import pandas as pd

np.random.seed(1)

data = np.random.rand(10000, 10)
df = pd.DataFrame(data)

rows = np.random.randint(0, 10000, 500)
```

```
cols = np.random.randint(0, 10, 500)

for i in range(500):
    df.iloc[rows[i], cols[i]] = np.nan

print(df.isnull().sum())

```

```
0    53
1    50
2    40
3    47
4    50
5    50
6    43
7    58
8    48
9    60
dtype: int64
```

Step:03

Write the program to check the missing values

```
import numpy as np
import pandas as pd

data = np.random.rand(10000, 10)
df = pd.DataFrame(data)

df.iloc[0, 1] = np.nan
df.iloc[5, 3] = np.nan
df.iloc[20, 7] = np.nan

print("Missing values in each feature:")
print(df.isnull().sum())

print("\nTotal missing values in dataset:")
print(df.isnull().sum().sum())

```

```
Missing values in each feature:
0    0
1    1
2    0
3    1
4    0
5    0
6    0
7    1
8    0
9    0
dtype: int64

Total missing values in dataset:
3
```

Step:04

Fill the missing values using the average value of that feature

```
import numpy as np
import pandas as pd

data = np.random.rand(10000, 10)
df = pd.DataFrame(data)

df.iloc[2, 1] = np.nan
df.iloc[10, 4] = np.nan
df.iloc[25, 7] = np.nan

print("Missing values before filling:")
print(df.isnull().sum())

df = df.fillna(df.mean())

print("\nMissing values after filling:")
print(df.isnull().sum())

```

```
Missing values before filling:
0    0
1    1
2    0
3    0
4    1
5    0
6    0
7    1
8    0
9    0
dtype: int64

Missing values after filling:
0    0
1    0
2    0
3    0
4    0
5    0
6    0
7    0
8    0
9    0
dtype: int64
```

Task # 02

To calculate precision, recall, f1 score, specificity, sensitivity

```python
import numpy as np
from sklearn.metrics import precision_score, recall_score, f1_score, confusion_matrix

y_true = np.array([1, 0, 1, 1, 0, 1, 0, 0, 1, 0])
y_pred = np.array([1, 0, 1, 0, 0, 1, 1, 0, 1, 0])

precision = precision_score(y_true, y_pred)
recall = recall_score(y_true, y_pred)
f1 = f1_score(y_true, y_pred)

cm = confusion_matrix(y_true, y_pred)
tn, fp, fn, tp = cm.ravel()

specificity = tn / (tn + fp)
sensitivity = tp / (tp + fn)

print("Precision:", precision)
print("Recall:", recall)
print("F1 Score:", f1)
print("Specificity:", specificity)
print("Sensitivity:", sensitivity)

```

```
Precision: 0.8
Recall: 0.8
F1 Score: 0.8
Specificity: 0.8
Sensitivity: 0.8
```

Naive Bayes formula

```python
import numpy as np
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import precision_score, recall_score, f1_score, confusion_matrix

X = np.array([
    [2.5, 1.5],
    [3.0, 1.0],
    [2.0, 1.0],
    [1.0, 3.5],
    [1.5, 3.0],
    [1.0, 2.5]
])

y = np.array([1, 1, 1, 0, 0, 0])

model = GaussianNB()
model.fit(X, y)

y_pred = model.predict(X)

```

```python
precision = precision_score(y, y_pred)
recall = recall_score(y, y_pred)
f1 = f1_score(y, y_pred)

cm = confusion_matrix(y, y_pred)
tn, fp, fn, tp = cm.ravel()

specificity = tn / (tn + fp)
sensitivity = tp / (tp + fn)

print("Precision:", precision)
print("Recall:", recall)
print("F1 Score:", f1)
print("Specificity:", specificity)
print("Sensitivity:", sensitivity)

```

```
Precision: 1.0
Recall: 1.0
F1 Score: 1.0
Specificity: 1.0
Sensitivity: 1.0
```