Source for Predicting heart disease using machine learning¶

This notebook looks into using various Python-based machine learning and data science libraries in an attempt to build a machine learning model capable of predicting whether or not someone has heart disease based on their medical attributes.

I am going to take the following approach:

1. Problem definition
2. Data
3. Evaluation
4. Features
5. Modelling
6. Experimentation

Preparing the Tools

```
import numpy as np

 import pandas as pd

 import matplotlib.pyplot as plt

import seaborn as sns

%matplotlib inline

from sklearn.linear_model import LogisticRegression

from sklearn.neighbors import KNeighborsClassifier

from sklearn.ensemble import RandomForestClassifier

from sklearn.model_selection import train_test_split, cross_val_score

from sklearn.model_selection import RandomizedSearchCV, GridSearchCV

from sklearn.metrics import confusion_matrix, classification_report

from sklearn.metrics import precision_score, recall_score, f1_score, RocCurveDisplay
```

Load Data

PROGRAM:

df = pd.read_csv("Downloads/healthcare_dataset.csv.zip") df.shape

df.shape

OUTPUT:

(55500, 15)

Data Exploration (exploratory data analysis or EDA)

PROGRAM:

df.head()

OUTPUT:

	Name	Age	Gender	Blood Type	Medical Condition	Date of Admission	Doctor	Hospital	Insurance Provider	Billing Amount	Room Number	Admission Type	Discharge Date	Medication	Test Results
0	Bobby JacksOn	30	Male	B-	Cancer	2024-01-31	Matthew Smith	Sons and Miller	Blue Cross	18856.281306	328	Urgent	2024-02-02	Paracetamol	Normal
1	LesLie TErRy	62	Male	A+	Obesity	2019-08-20	Samantha Davies	Kim Inc	Medicare	33643.327287	265	Emergency	2019-08-26	Ibuprofen	Inconclusive
2	DaNnY sMitH	76	Female	A-	Obesity	2022-09-22	Tiffany Mitchell	Cook PLC	Aetna	27955.096079	205	Emergency	2022-10-07	Aspirin	Normal
3	andrEw waTtS	28	Female	O+	Diabetes	2020-11-18	Kevin Wells	Hernandez Rogers and Vang,	Medicare	37909.782410	450	Elective	2020-12-18	Ibuprofen	Abnormal
4	adrIENNE bEll	43	Female	AB+	Cancer	2022-09-19	Kathleen Hanna	White-White	Aetna	14238.317814	458	Urgent	2022-10-09	Penicillin	Abnormal

PROGRAM:

df.tail()

OUTPUT:

	Name	Age	Gender	Blood Type	Medical Condition	Date of Admission	Doctor	Hospital	Insurance Provider	Billing Amount	Room Number	Admission Type	Discharge Date	Medication	R
55495	eLIZABeTH jaCkSOn	42	Female	O+	Asthma	2020-08-16	Joshua Jarvis	Jones-Thompson	Blue Cross	2650.714952	417	Elective	2020-09-15	Penicillin	Abn
55496	KYle pEREz	61	Female	AB-	Obesity	2020-01-23	Taylor Sullivan	Tucker-Moyer	Cigna	31457.797307	316	Elective	2020-02-01	Aspirin	N
55497	HEATher WaNG	38	Female	B+	Hypertension	2020-07-13	Joe Jacobs DVM	and Mahoney Johnson Vasquez,	UnitedHealthcare	27620.764717	347	Urgent	2020-08-10	Ibuprofen	Abn
55498	JENniFER JOneS	43	Male	O-	Arthritis	2019-05-25	Kimberly Curry	Jackson Todd and Castro,	Medicare	32451.092358	321	Elective	2019-05-31	Ibuprofen	Abn
55499	jAMES GARCiA	53	Female	O+	Arthritis	2024-04-02	Dennis Warren	Henry Sons and	Aetna	4010.134172	448	Urgent	2024-04-29	Ibuprofen	Abn

PROGRAM:

df["Gender"].value_counts()

OUTPUT:

Gender
Male 27774
Female 27726
Name: count, dtype: int64

PROGRAM:

```
df["Blood Type"].value_counts()
```

OUTPUT:

Blood Type
A- 6969
A+ 6956
AB+ 6947
AB- 6945
B+ 6945
B- 6944
O+ 6917
O- 6877
Name: count, dtype: int64

PROGRAM:

```
df["Test Results"].value_counts()
```

OUTPUT:

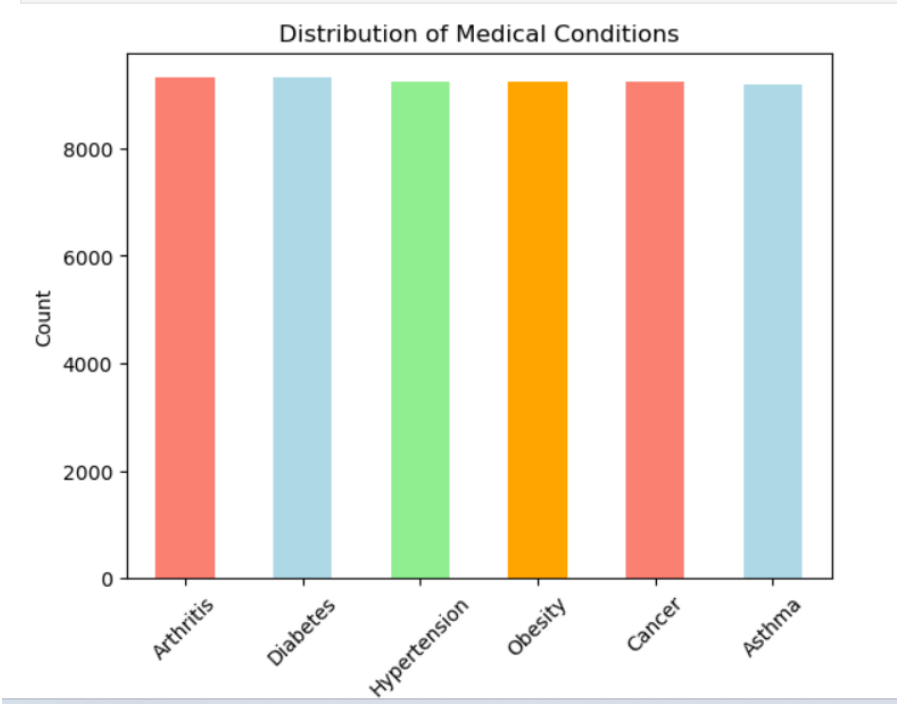
Test Results
Abnormal 18627
Normal 18517
Inconclusive 18356
Name: count, dtype: int64

PROGRAM:

```
df["Medical Condition"].value_counts().plot(kind='bar', color=["salmon", "lightblue", "lightgreen", "orange"])

plt.title("Distribution of Medical Conditions")

plt.xlabel("Condition")

plt.ylabel("Count")

plt.xticks(rotation=45)
```

plt.show()

OUTPUT:



PROGRAM:

df.info()

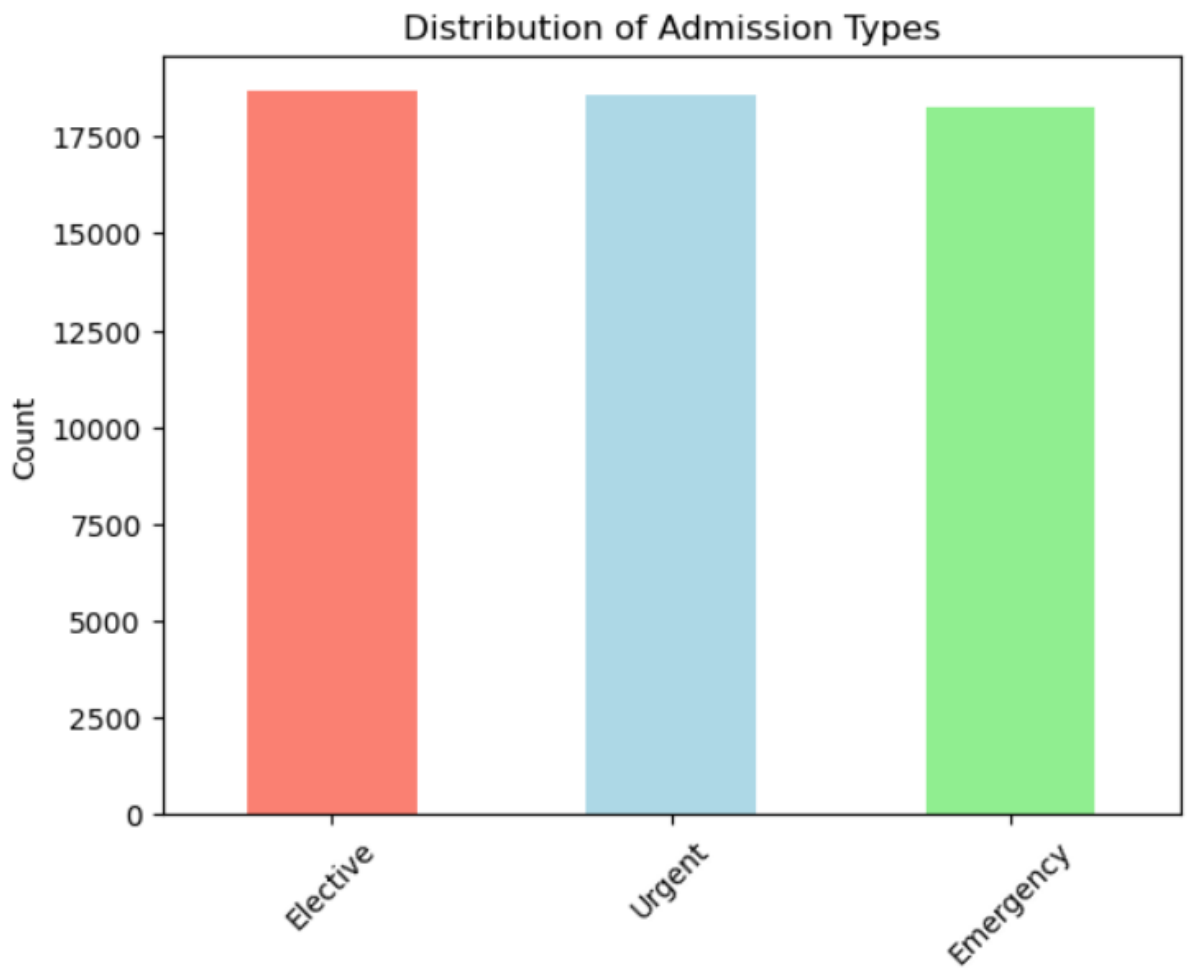
OUTPUT:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 55500 entries, 0 to 55499
Data columns (total 15 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   Name                55500 non-null  object
 1   Age                 55500 non-null  int64
 2   Gender              55500 non-null  object
 3   Blood Type          55500 non-null  object
 4   Medical Condition   55500 non-null  object
 5   Date of Admission   55500 non-null  object
 6   Doctor              55500 non-null  object
 7   Hospital            55500 non-null  object
 8   Insurance Provider  55500 non-null  object
 9   Billing Amount      55500 non-null  float64
 10  Room Number         55500 non-null  int64
 11  Admission Type      55500 non-null  object
 12  Discharge Date      55500 non-null  object
 13  Medication          55500 non-null  object
 14  Test Results        55500 non-null  object
dtypes: float64(1), int64(2), object(12)
memory usage: 6.4+ MB
```

PROGRAM:

```
df["Admission Type"].value_counts().plot(kind='bar', color=["salmon", "lightblue", "lightgreen"])

plt.title("Distribution of Admission Types")

plt.xlabel("Type")

plt.ylabel("Count")

plt.xticks(rotation=45)

plt.show()
```

OUTPUT:



PROGRAM:

df.isna().sum()

OUTPUT:

```
Name                  0
Age                   0
Gender                0
Blood Type            0
Medical Condition     0
Date of Admission     0
Doctor                0
Hospital              0
Insurance Provider    0
Billing Amount        0
Room Number           0
Admission Type        0
Discharge Date        0
Medication            0
Test Results          0
dtype: int64
```

PROGRAM:

df.describe()

OUTPUT:

	Age	Billing Amount	Room Number
count	55500.000000	55500.000000	55500.000000
mean	51.539459	25539.316097	301.134829
std	19.602454	14211.454431	115.243069
min	13.000000	-2008.492140	101.000000
25%	35.000000	13241.224652	202.000000
50%	52.000000	25538.069376	302.000000
75%	68.000000	37820.508436	401.000000
max	89.000000	52764.276736	500.000000

PROGRAM:

df.Gender.value_counts()

OUTPUT:

Gender
Male 27774
Female 27726
Name: count, dtype: int64

PROGRAM:

pd.crosstab(df["Medical Condition"], df["Gender"])

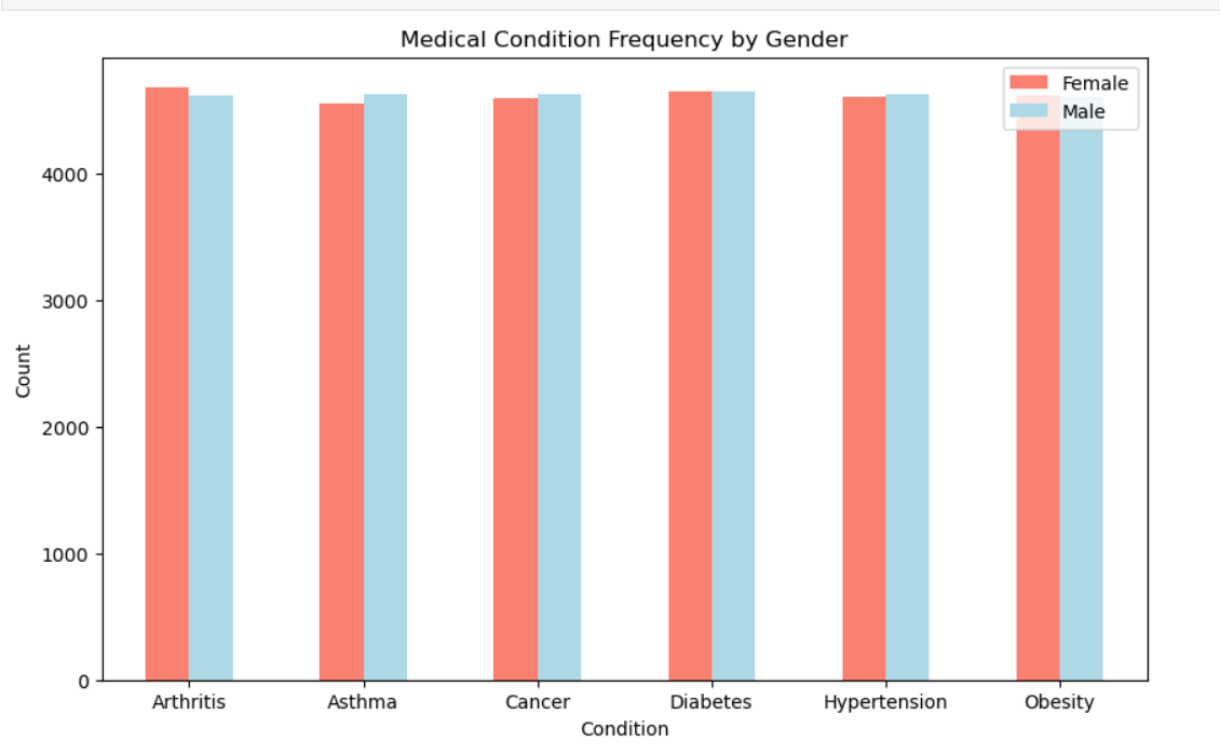
OUTPUT:

Gender	Female	Male
Medical Condition		
Arthritis	4686	4622
Asthma	4553	4632
Cancer	4602	4625
Diabetes	4651	4653
Hypertension	4612	4633
Obesity	4622	4609

PROGRAM:

```
pd.crosstab(df["Medical Condition"], df["Gender"]).plot(kind="bar", figsize=(10, 6), color=["salmon", "lightblue"])

plt.title("Medical Condition Frequency by Gender")

plt.xlabel("Condition")

plt.ylabel("Count")

plt.legend(["Female", "Male"])

plt.xticks(rotation=0)

plt.show()
```

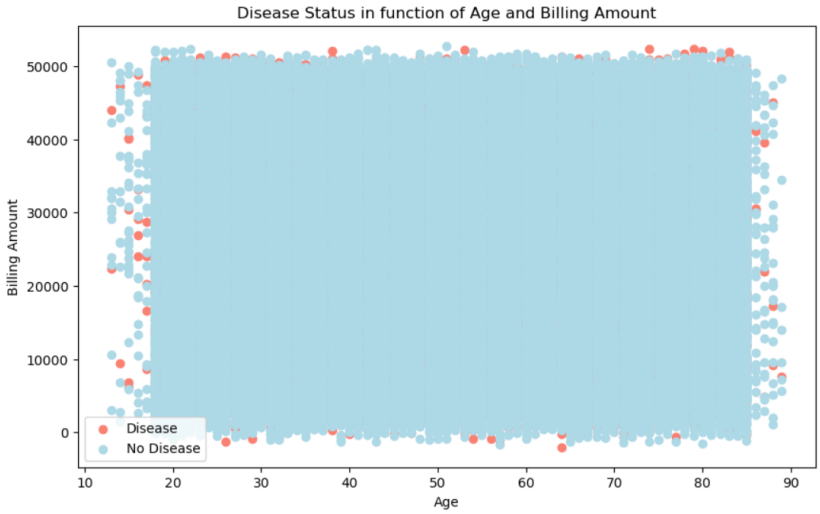
OUTPUT:



PROGRAM:

```
plt.figure(figsize=(10,6))
plt.scatter(df["Age"][df["Medical Condition"] == "Cancer"],
            df["Billing Amount"][df["Medical Condition"] == "Cancer"],
            c="salmon")
plt.scatter(df["Age"][df["Medical Condition"] != "Cancer"],
           df["Billing Amount"][df["Medical Condition"] != "Cancer"],
            c="lightblue")
plt.title("Disease Status in function of Age and Billing Amount")
plt.xlabel("Age")
plt.ylabel("Billing Amount")
plt.legend(["Disease", "No Disease"])
plt.show()
```

OUTPUT:

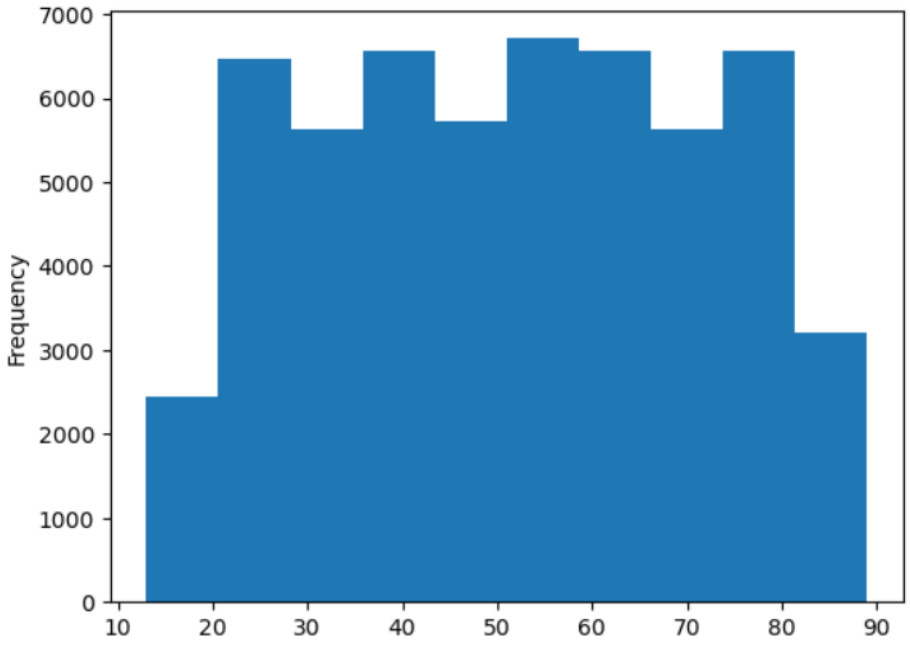


PROGRAM:

df.Age.plot.hist()

OUTPUT:

```
: <Axes: ylabel='Frequency'>
```

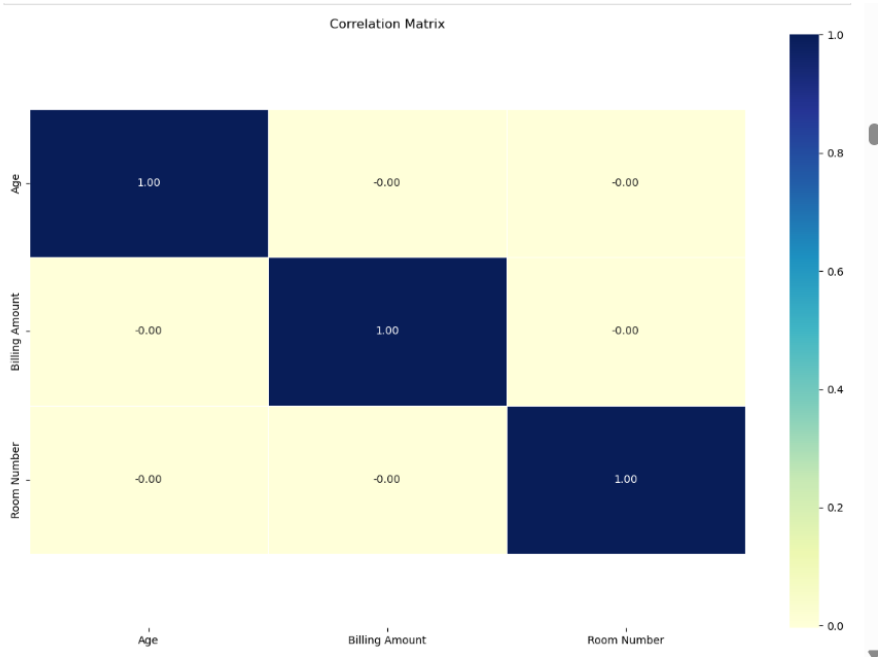


correlation matrix

PROGRAM:

```
numeric_df = df.select_dtypes(include=[np.number])
corr_matrix = numeric_df.corr()
fig, ax = plt.subplots(figsize=(15, 10))
sns.heatmap(corr_matrix, annot=True, linewidths=0.5, fmt=".2f", cmap="YlGnBu", ax=ax)
bottom, top = ax.get_ylim()
 ax.set_ylim(bottom + 0.5, top - 0.5)
plt.title("Correlation Matrix")
 plt.show()
```

OUTPUT:

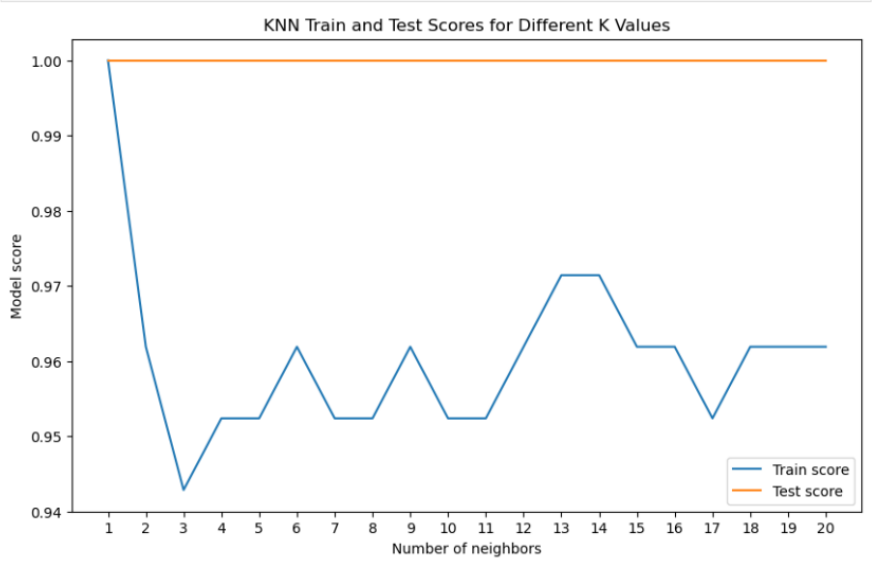


PROGRAM:

```
import numpy as np
 import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
```

```
from sklearn.neighbors import KNeighborsClassifier

from sklearn.datasets import load_iris

from sklearn.metrics import accuracy_score

data = load_iris()

X = data.data

y = data.target

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

train_scores = [] test_scores = [] neighbors = range(1, 21)

for k in neighbors:

 knn = KNeighborsClassifier(n_neighbors=k)
knn.fit(X_train, y_train)
train_accuracy = accuracy_score(y_train, knn.predict(X_train))
test_accuracy = accuracy_score(y_test, knn.predict(X_test))
train_scores.append(train_accuracy)
test_scores.append(test_accuracy)

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

plt.plot(neighbors, train_scores, label="Train score")

 plt.plot(neighbors, test_scores, label="Test score")

 plt.xticks(np.arange(1, 21, 1))

plt.xlabel("Number of neighbors")

 plt.ylabel("Model score")

 plt.title("KNN Train and Test Scores for Different K Values")

 plt.legend()

plt.show()

print(f"Maximum KNN score on the test data: {max(test_scores)*100:.2f}%")
```

OUTPUT:

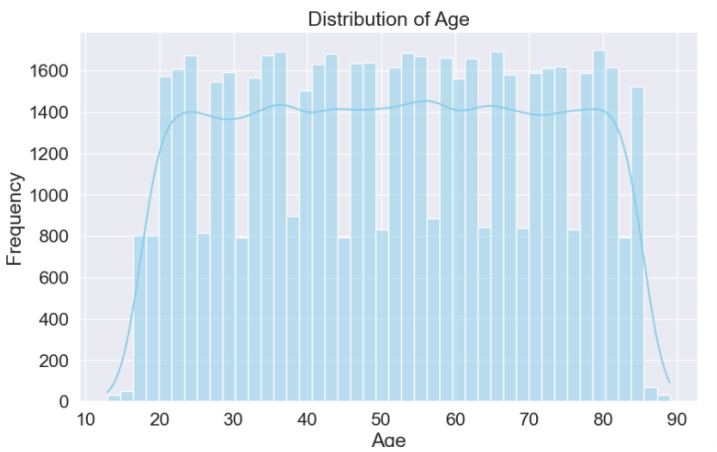


Maximum KNN score on the test data: 100.00%

PROGRAM:

```
import seaborn as sns

import matplotlib.pyplot as plt


# Visualize distribution of Age

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

sns.histplot(df['Age'], kde=True, color='skyblue')

plt.title('Distribution of Age')

plt.xlabel('Age')

plt.ylabel('Frequency')

plt.show()
```

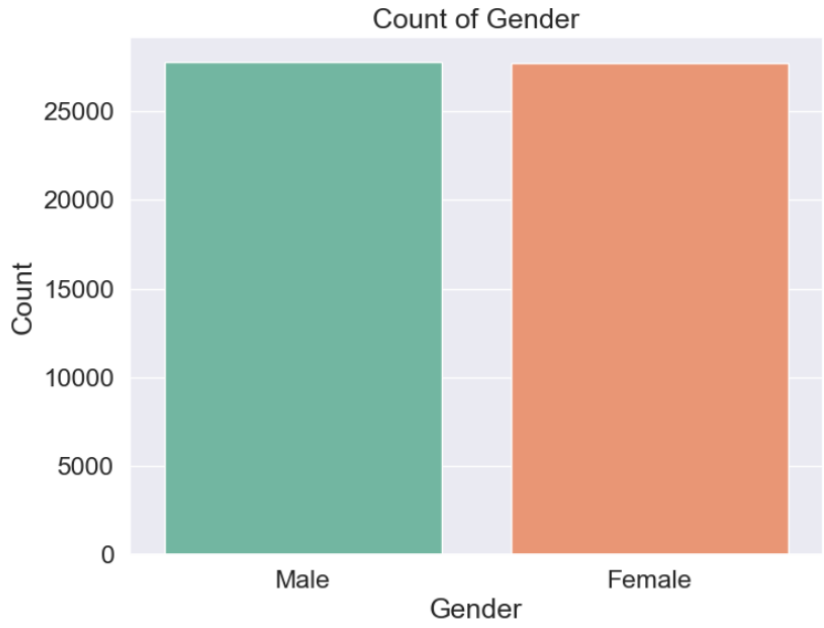
OUTPUT:



PROGRAM:

```
plt.figure(figsize=(8,6))

sns.countplot(x='Gender', data=df, palette='Set2')

plt.title('Count of Gender')

plt.xlabel('Gender')

plt.ylabel('Count')

plt.show()
```

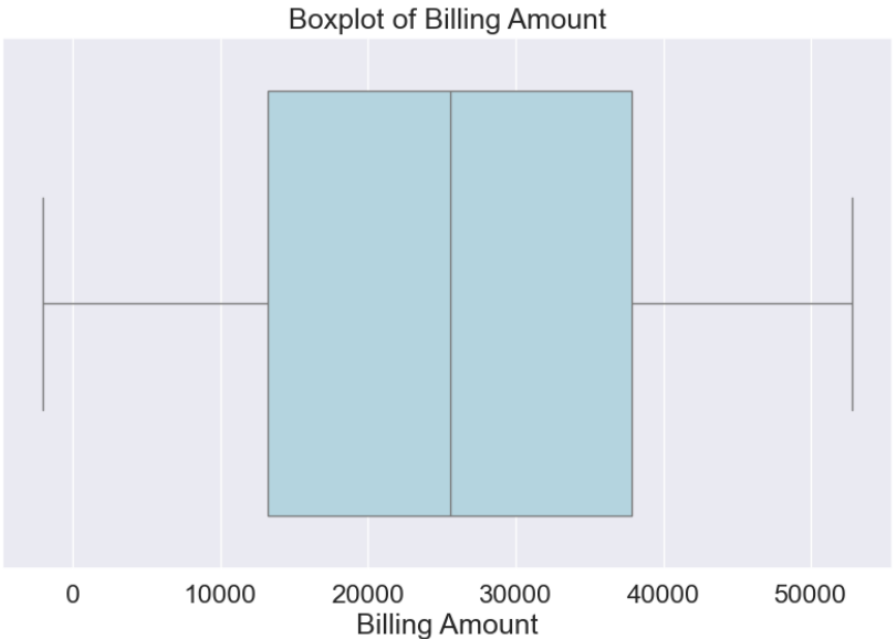
OUTPUT:



PROGRAM:

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

sns.boxplot(x='Billing Amount', data=df, color='lightblue')

plt.title('Boxplot of Billing Amount')

plt.xlabel('Billing Amount')

plt.show()
```

OUTPUT:



Outliers using IQR (Interquartile Range)

PROGRAM:

```
Q1 = df[['Age', 'Billing Amount']].quantile(0.25)

 Q3 = df[['Age', 'Billing Amount']].quantile(0.75)

 IQR = Q3 - Q1

outliers = ((df[['Age', 'Billing Amount']] < (Q1 - 1.5 * IQR)) | (df[['Age', 'Billing Amount']] > (Q3 + 1.5 * IQR)))
```

```
outlier_count = outliers.sum()

 print("\nOutliers per column:")

 print(outlier_count)

df_cleaned_no_outliers = df[~outliers.any(axis=1)]
```

OUTPUT:

```
Outliers per column:
Age              0
Billing Amount   0
dtype: int64
```

duplicate rows

PROGRAM:

```
duplicates = df.duplicated().sum()

print(f"Number of duplicate rows: {duplicates}")

df_cleaned_no_duplicates = df.drop_duplicates()
```

OUTPUT:

```
Number of duplicate rows: 534
```