


```
# 1. Import libraries
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, confusion_matrix
```


```
# 2. Load the data
df = pd.read_csv('/content/Diabetes Dataset(Sheet1).csv')
```

```
# 3. Quick Data Overview
df.head()
df.info()
df.describe()
```

 <class 'pandas.core.frame.DataFrame'>  
RangeIndex: 768 entries, 0 to 767  
Data columns (total 9 columns):  
# Column Non-Null Count Dtype  
--- ---  
0 Pregnancies 768 non-null int64  
1 Glucose 768 non-null int64  
2 BloodPressure 768 non-null int64  
3 SkinThickness 768 non-null int64  
4 Insulin 768 non-null int64  
5 BMI 768 non-null float64  
6 DiabetesPedigreeFunction 768 non-null float64  
7 Age 768 non-null int64  
8 Outcome 768 non-null int64  
dtypes: float64(2), int64(7)  
memory usage: 54.1 KB

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
count	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000
mean	3.845052	120.894531	69.105469	20.536458	79.799479	31.992578	0.471876	33.240885	0.348958
std	3.369578	31.972618	19.355807	15.952218	115.244002	7.884160	0.331329	11.760232	0.476951
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.078000	21.000000	0.000000
25%	1.000000	99.000000	62.000000	0.000000	0.000000	27.300000	0.243750	24.000000	0.000000
50%	3.000000	117.000000	72.000000	23.000000	30.500000	32.000000	0.372500	29.000000	0.000000
75%	6.000000	140.250000	80.000000	32.000000	127.250000	36.600000	0.626250	41.000000	1.000000
max	17.000000	199.000000	122.000000	99.000000	846.000000	67.100000	2.420000	81.000000	1.000000

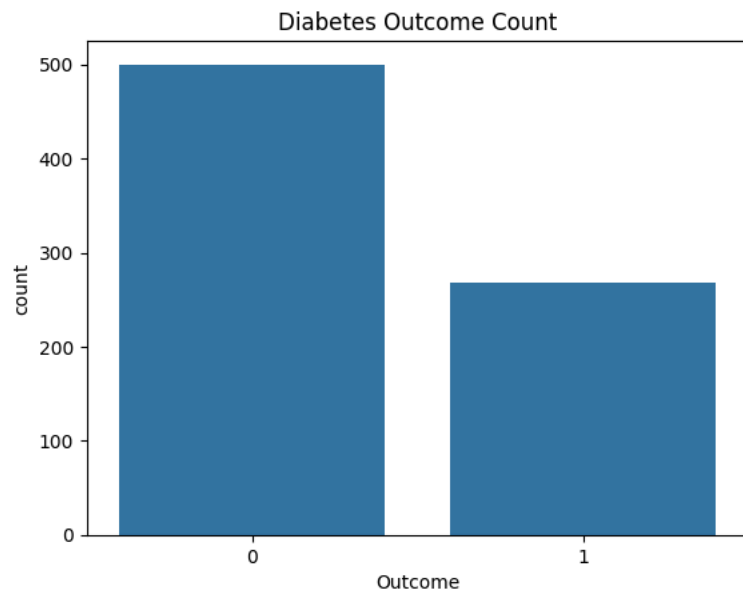
```
# 4. Checking for missing values
(df == 0).sum()
```



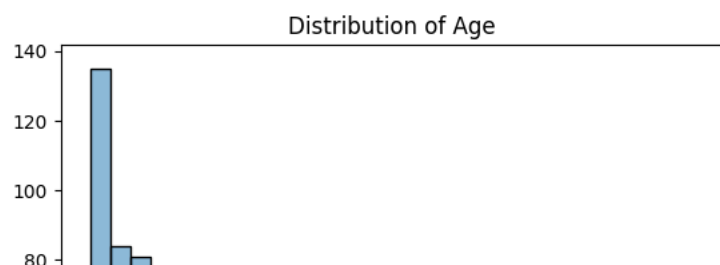
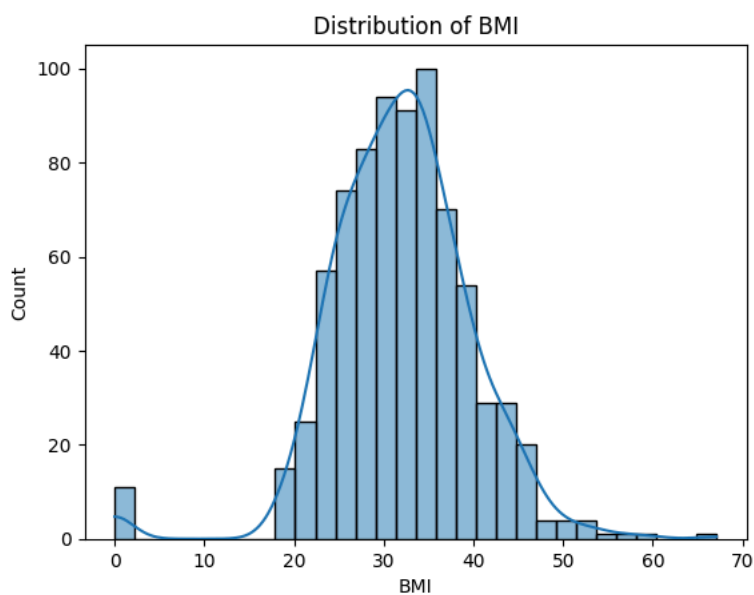
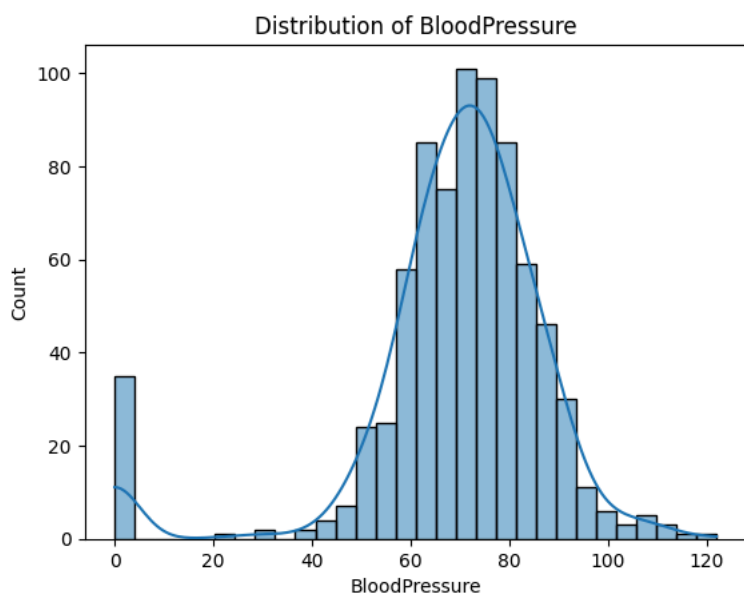
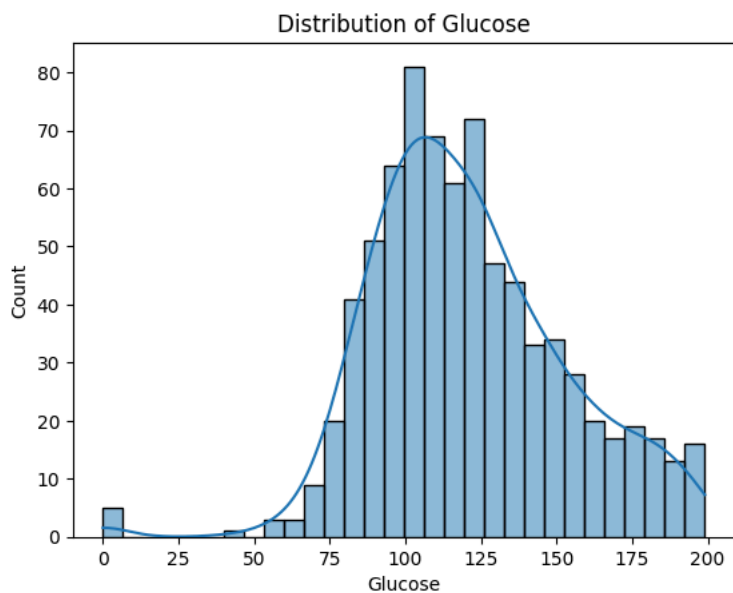
	0
Pregnancies	111
Glucose	5
BloodPressure	35
SkinThickness	227
Insulin	374
BMI	11
DiabetesPedigreeFunction	0
Age	0
Outcome	500

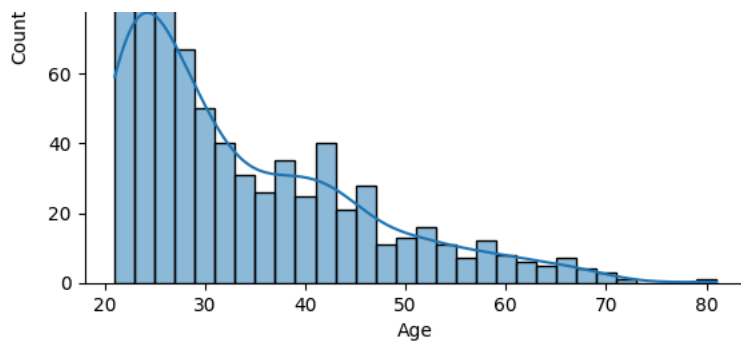
dtype: int64

```
# 5. Outcome Distribution Using Countplot
sns.countplot(x='Outcome', data=df)
plt.title('Diabetes Outcome Count')
plt.show()
```

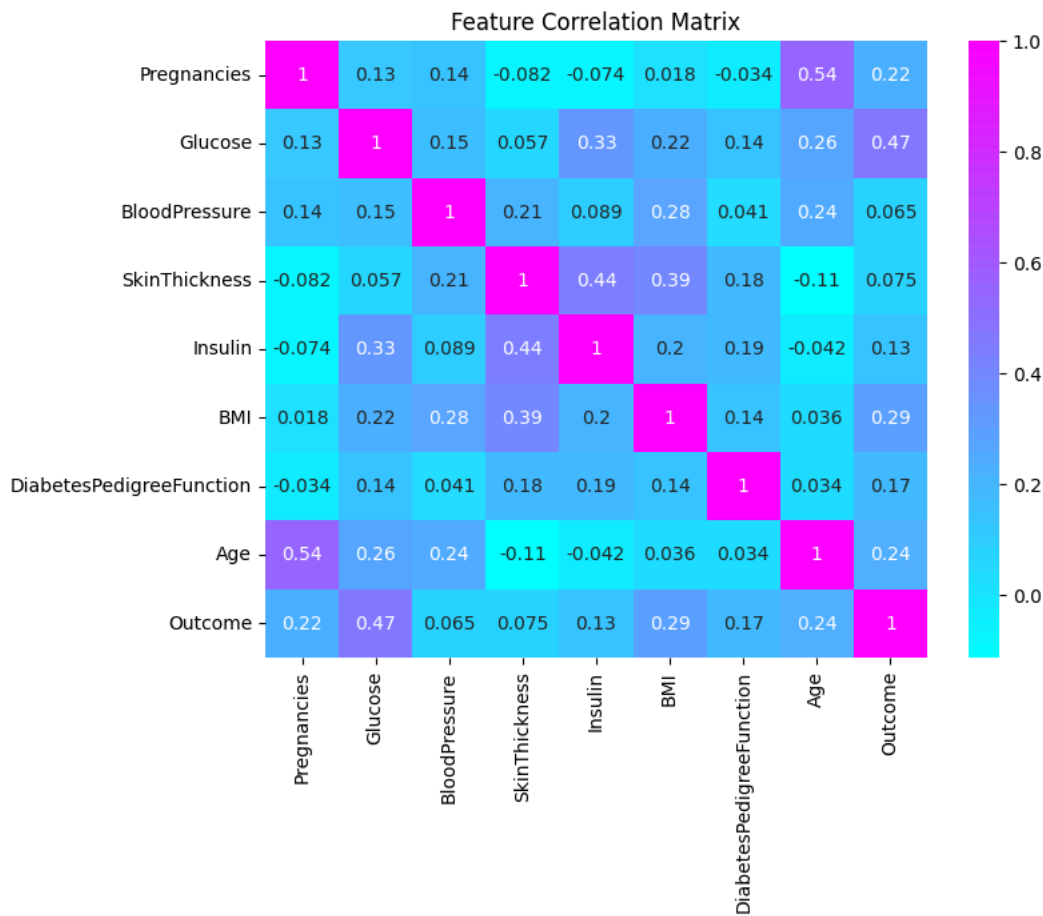


```
# 6. Visualize Feature Distributions Using Histogram
features = ['Glucose', 'BloodPressure', 'BMI', 'Age']
for col in features:
    plt.figure()
    sns.histplot(df[col], bins=30, kde=True)
    plt.title(f'Distribution of {col}')
    plt.show()
```





```
#Finding relationships between variables Using Heatmap
plt.figure(figsize=(8, 6))
sns.heatmap(df.corr(), annot=True, cmap='cool')
plt.title('Feature Correlation Matrix')
plt.show()
```



```
# The data for machine learning
X = df.drop('Outcome', axis=1)
y = df['Outcome']
```

```
# Scale the features for SVM
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

```
# Splitting the dataset into training and test sets
X_train, X_test, y_train, y_test = train_test_split(
    X_scaled, y, test_size=0.2, random_state=0, stratify=y
)
```

```
# Fitting a Support Vector Classifier (SVM)
clf = SVC(kernel='rbf', probability=True, random_state=0)
clf.fit(X_train, y_train)
```



▼ SVC ⓘ ?  
SVC(probability=True, random\_state=0)

```
# Evaluating the test set
y_pred = clf.predict(X_test)
acc = accuracy_score(y_test, y_pred)
cm = confusion_matrix(y_test, y_pred)
```

```
acc
cm
```



```
array([[89, 11],
       [23, 31]])
```

```
# Showing confusion matrix
```