Machine Learning

Design of SVM algorithm for classification

Name :

Devarapalli

Nitish Babu

```
import numpy as np 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 classification_report, confusion_matrix, accuracy_score url
    =
    "https://raw.githubusercontent.com/jbrownlee/Datasets/master/pima-
indians- diabetes.data.csv"

columns = ['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin', 'BMI',
'DiabetesPedigreeFunction', 'Age', 'Outcome']
```

```
df = pd.read_csv(url, names=columns)


X = df.drop('Outcome', axis=1) y

= df['Outcome']


scaler = StandardScaler()

X_scaled = scaler.fit_transform(X)


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


svm_clf = SVC(kernel='linear', random_state=42) svm_clf.fit(X_train, y_train)


y_pred = svm_clf.predict(X_test)
print("Classification Report:") print(classification_report(y_test,

y_pred))


conf_matrix = confusion_matrix(y_test, y_pred) print("\nConfusion

Matrix:")

print(conf_matrix)


accuracy = accuracy_score(y_test, y_pred) print(f"\nAccuracy

of the SVM Classifier: {accuracy * 100:.2f}%") def

radar_plot(data, labels, title):

    angles = np.linspace(0, 2 * np.pi, len(labels), endpoint=False).tolist() data

    = np.concatenate((data, [data[0]])) angles

    += angles[:1]
```

```
    fig, ax = plt.subplots(figsize=(6, 6), subplot_kw=dict(polar=True))
    ax.fill(angles, data, color='blue', alpha=0.25) ax.plot(angles, data,
         color='blue',    linewidth=2)    ax.set_yticklabels([])
    ax.set_xticks(angles[:-1]) ax.set_xticklabels(labels, size=10)
    plt.title(title)
    plt.show()

radar_plot(X_train[0], columns[:-1], "Radar Plot for Input Features") plt.figure(figsize=(12,6))
sns.boxplot(data=df.drop('Outcome',   axis=1))   plt.title('Box   Plot   of   Input   Features')
plt.xticks(rotation=90) plt.tight_layout() plt.show()

feature_importance = np.mean(np.abs(X_train), axis=0)
plt.figure(figsize=(10,6))          plt.bar(columns[:-1],
feature_importance) plt.title('Stock Bar Graph of Feature
Importance') plt.xticks(rotation=45)
plt.show()

plt.figure(figsize=(8,6)) plt.scatter(range(len(y_test)), y_test, color='green', label='True Labels')
plt.scatter(range(len(y_pred)), y_pred, color='orange', alpha=0.5, label='Predicted
Labels') plt.title("Scatter Plot - True vs Predicted Labels") plt.legend() plt.show()
plt.figure(figsize=(6,4)) sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues',
xticklabels=['Non-Diabetic', 'Diabetic'], yticklabels=['Non-Diabetic',       'Diabetic'])
         plt.title('Confusion      Matrix')         plt.ylabel('Actual') plt.xlabel('Predicted')
plt.show() plt.figure(figsize=(10,6))      for column in df.columns[:-1]:
    sns.kdeplot(df[column],     label=column)
plt.title("Fair Plot - Distribution of Features")
plt.legend()
plt.show()
```

corr_matrix = df.corr() plt.figure(figsize=(8,6))

sns.heatmap(corr_matrix, annot=True, cmap='coolwarm', square=True) plt.title("Heap

Matrix - Correlation of Features")

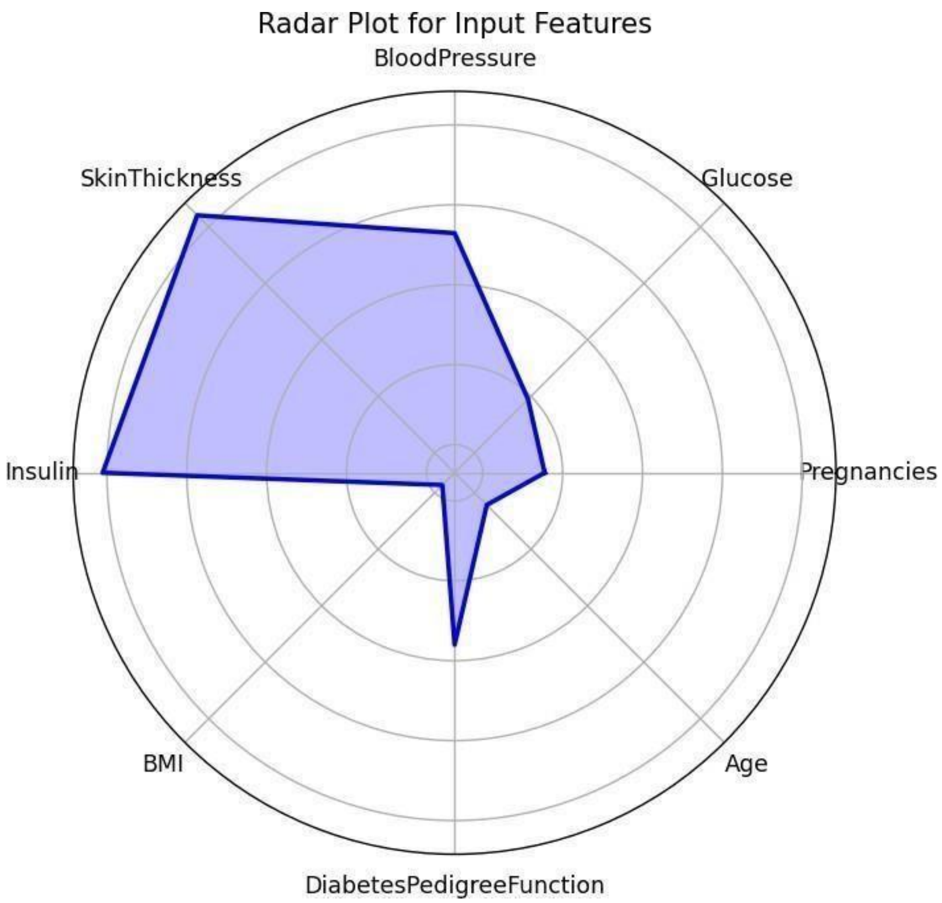
plt.show()

Output :

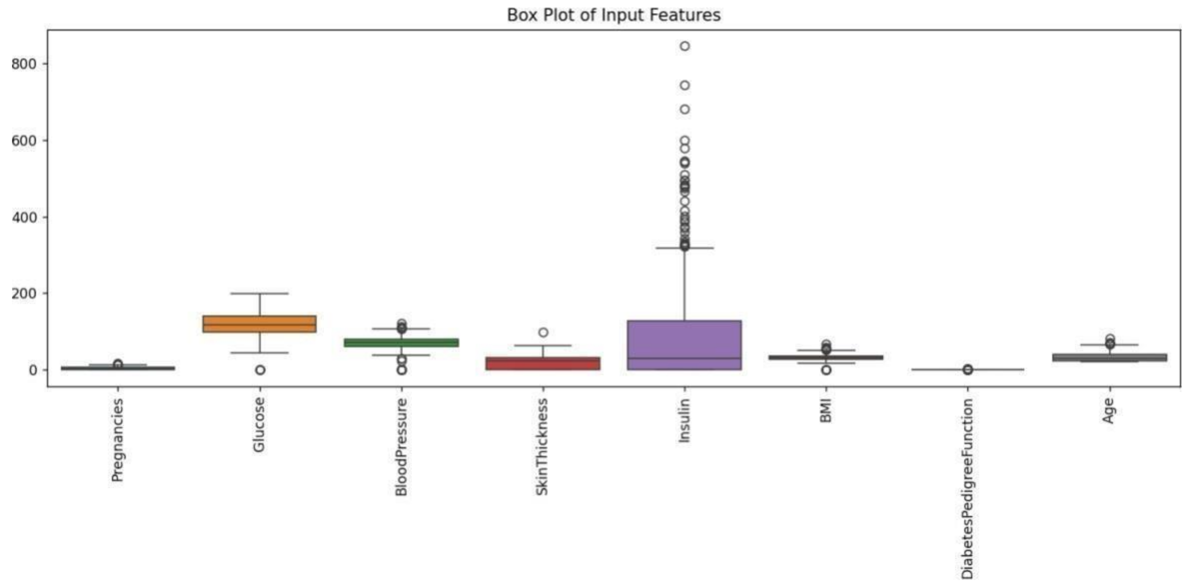
```
Classification Report:
              precision    recall  f1-score   support

           0       0.80      0.81      0.81       151
           1       0.64      0.62      0.63        80

    accuracy                           0.75       231
   macro avg       0.72      0.72      0.72       231
weighted avg       0.75      0.75      0.75       231


Confusion Matrix:
[[123  28]
 [ 30  50]]

Accuracy of the SVM Classifier: 74.89%
```

Radar Plot :

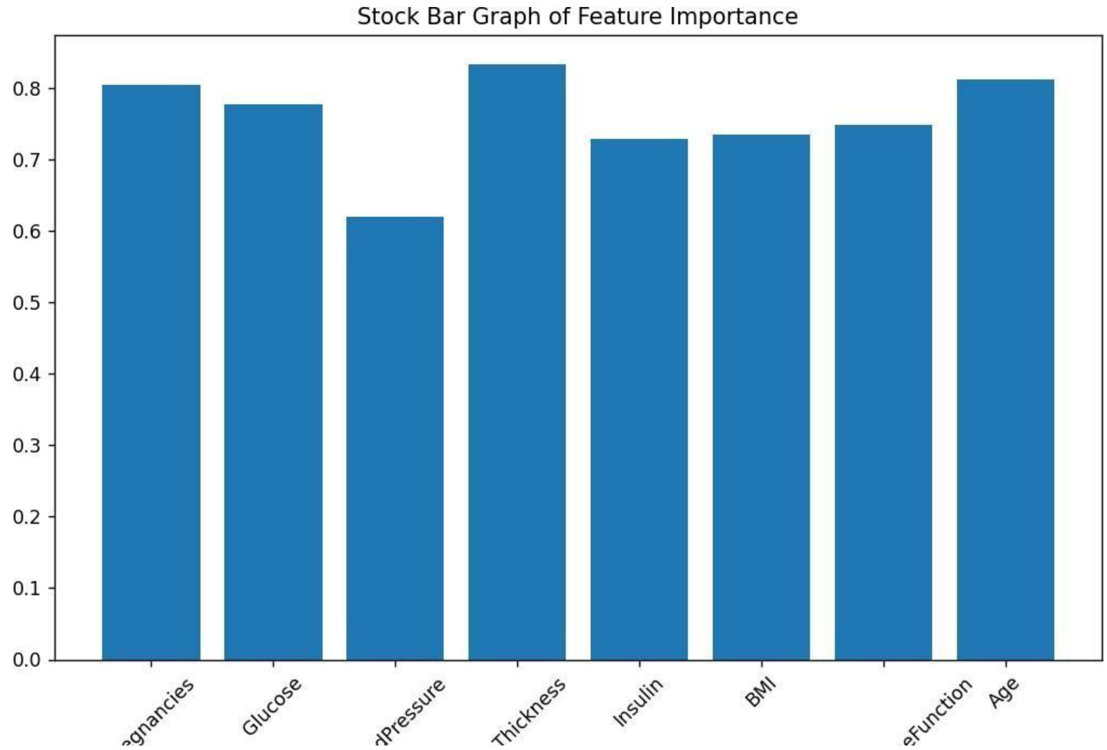
Bar Graph :



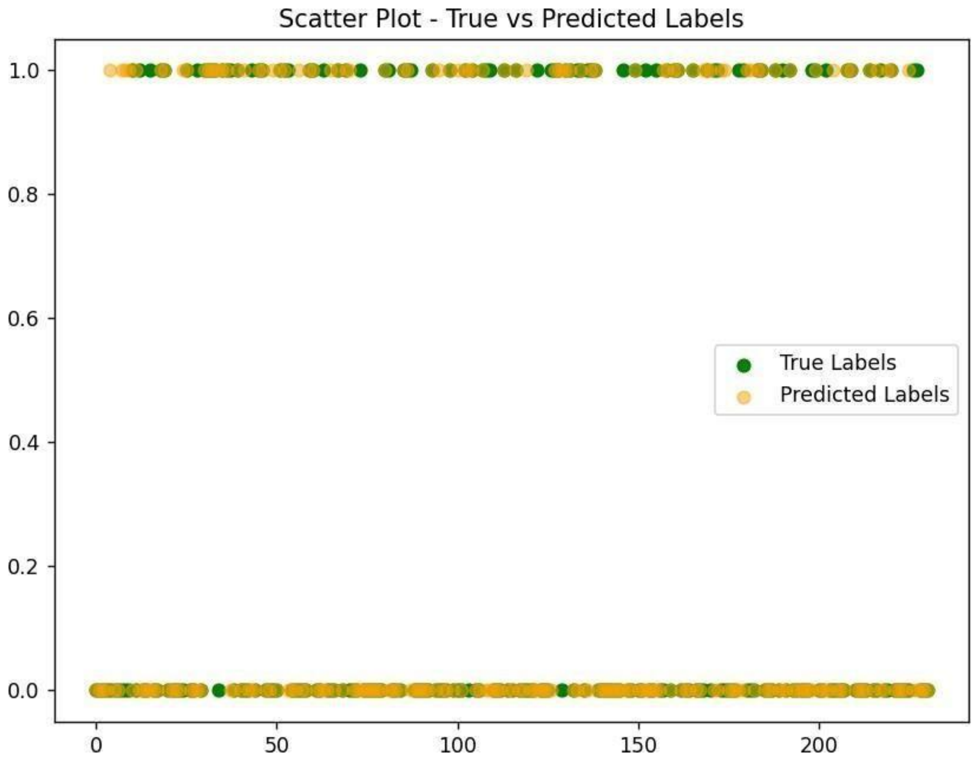
Box Plot :



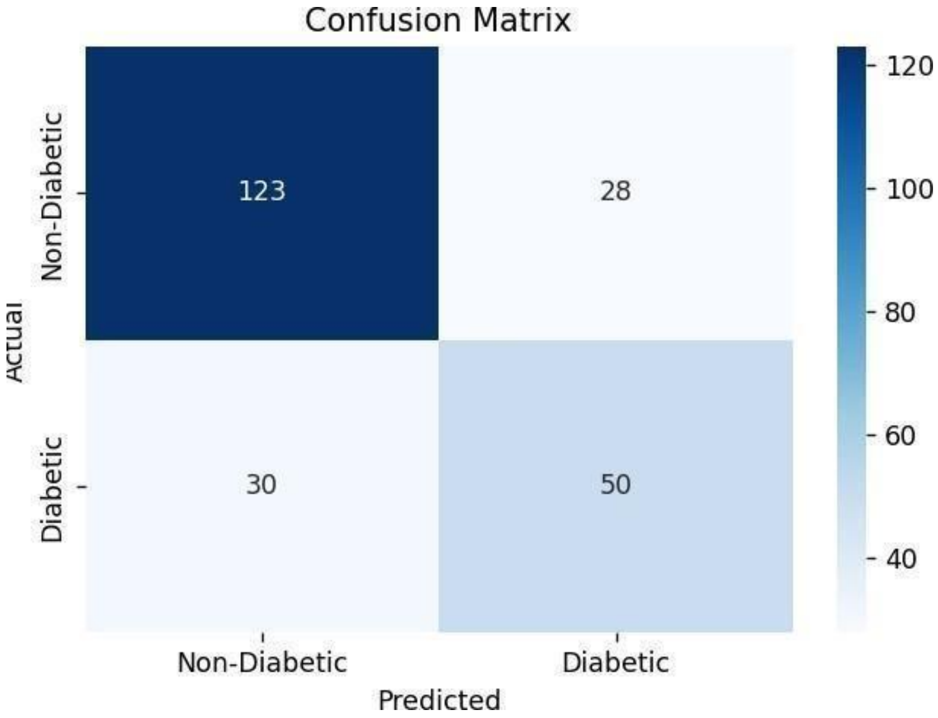
Bar Graph :



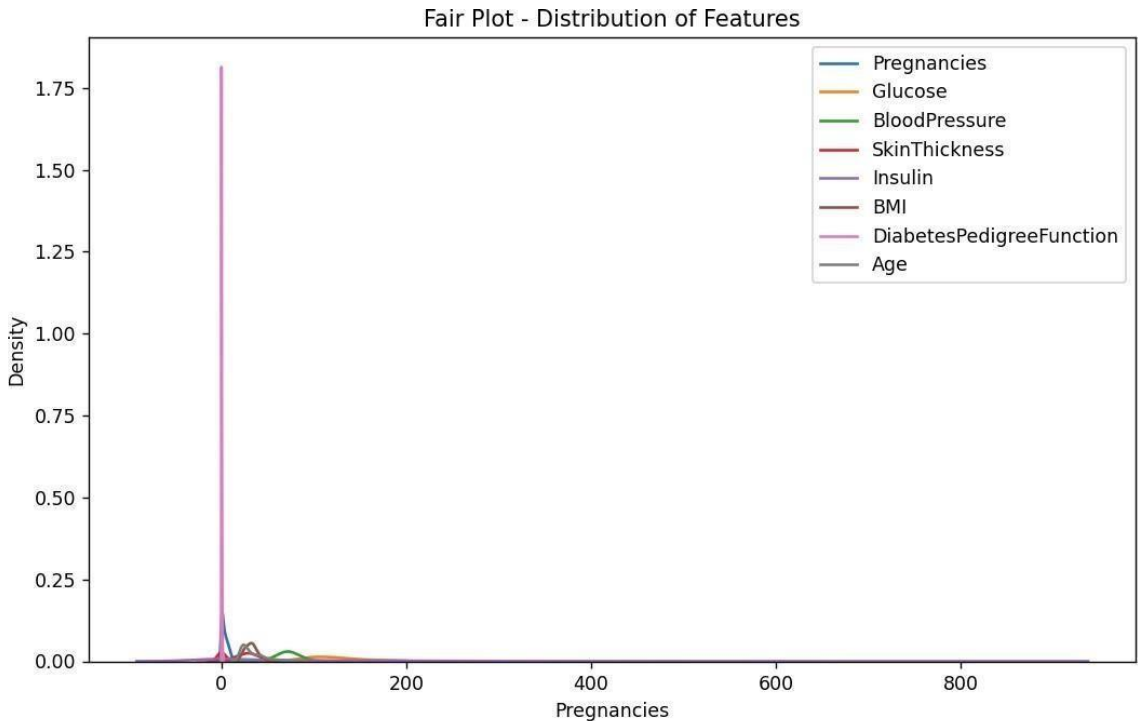
Scatter Plot :



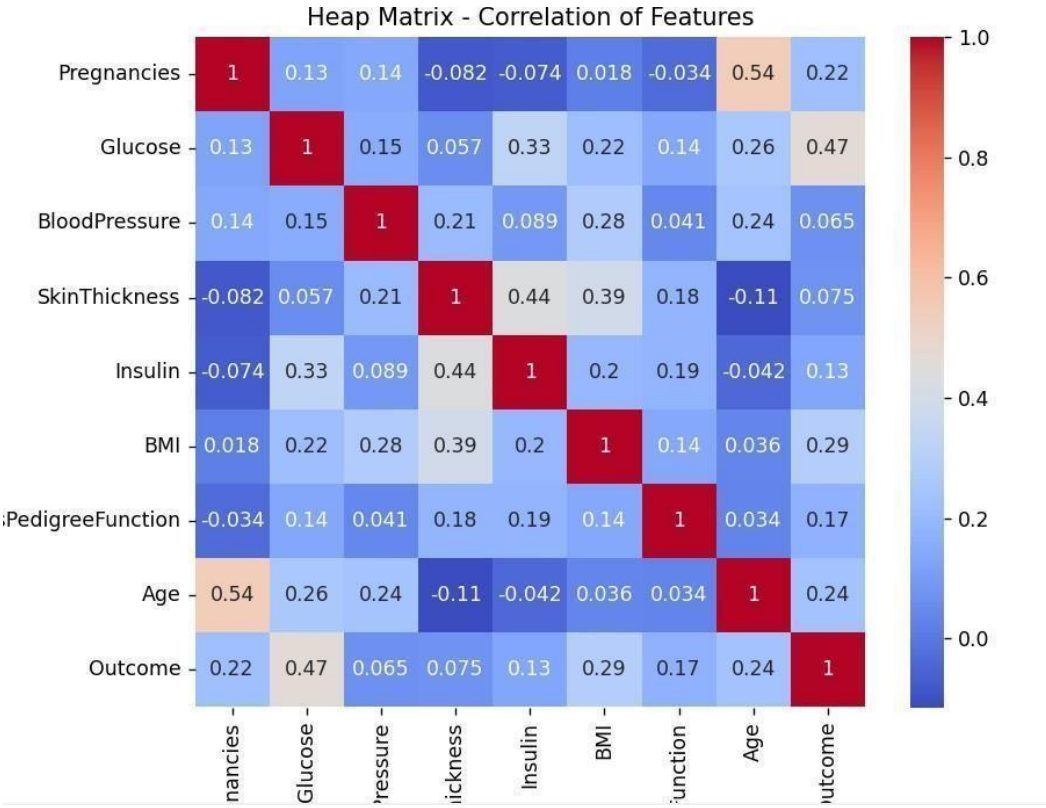
Confusion Matrix :



Fair Plot :



Heap Matrix :



Heap Matrix - Correlation of Features
Pregnancies
Glucose
BloodPressure
SkinThickness
Insulin
BMI
PedigreeFunction
Age
Outcome
1.0
0.8
0.6
0.4
0.2
0.0