

ST JOSEPH COLLEGE OF ENGINEERING

**TITLE : AI-BASED DIABETICS
PREDICTION MODEL
PHASE-4**

PROJ_227128_TEAM_1

Coding for the diabetics prediction system

C
O
D
E

Step 1: Import necessary libraries

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore", category=UserWarning)
```

Step 2: Load the dataset df =

```
pd.read_csv('/kaggle/input/diabetesdata-set/diabetes.csv')
```


C O D E

Step 3: Data Cleaning Check for Missing Values

code:

```
missing_values = df.isnull().sum()
print("Missing Values:")
print(missing_values) Handle
missing values (if any) mean_fill =
df.mean() df.fillna(mean_fill,
inplace=True) Check for Duplicate
Rows duplicate_rows =
df[df.duplicated()]
print("\nDuplicate Rows:")
print(duplicate_rows) Handle
duplicate rows (if any)
```

```
df.drop_duplicates(inplace=True)
```

Step 4: Data Analysis Summary Statistics

C

code:

```
summary_stats = df.describe()
```

```
print("\nSummary Statistics:")
```

```
print(summary_stats) Class
```

```
Distribution
```

O

```
class_distribution = df['Outcome'].value_counts()
```

```
print("\nClass Distribution:")
```

D

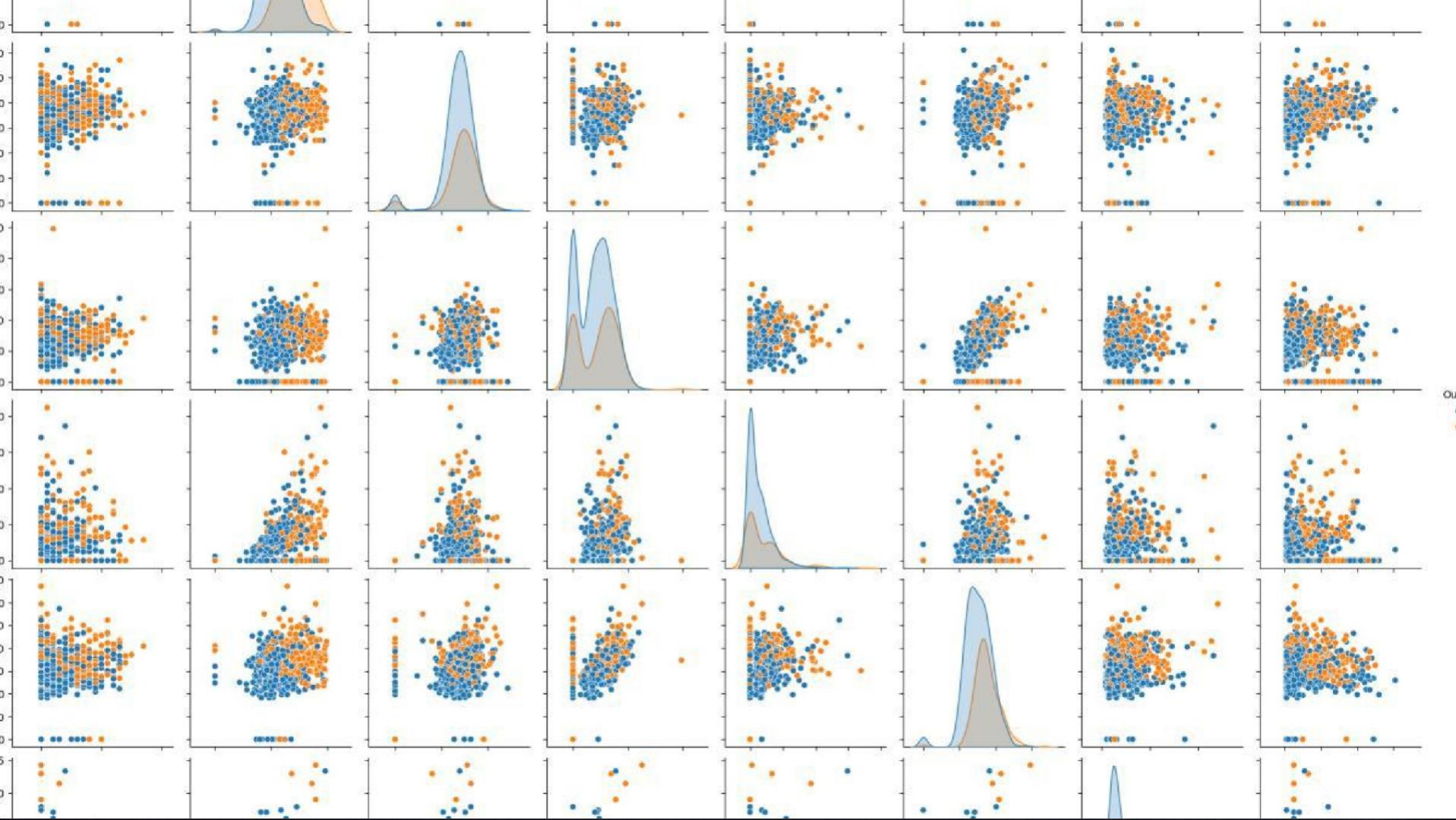
```
print(class_distribution)
```

E

Step 5: Data
Visualization

code:

```
sns.pairplot(df, hue='Outcome')  
plt.show()
```

The previous slide provide the visualization of the requires data given in the dataset

code:

```
sns.pairplot(df, hue='Outcome')  
plt.show()
```

this code is the reason to show each and every section of ther diabetics result as it is easy to predict by the user

this code is used for data set cleaning and provide the solution for the dataset give

C O D E

```
import pandas as pd import numpy
as np import matplotlib.pyplot as plt
import seaborn as sns from
sklearn.model_selection import
train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn import svm from sklearn.metrics
import classification_report from sklearn.metrics
import confusion_matrix
from sklearn.metrics import
ConfusionMatrixDisplay
RED = "\033[91m"
GREEN = "\033[92m"
YELLOW = "\033[93m"
BLUE = "\033[94m"
RESET = "\033[0m"
df = pd.read_csv("/kaggle/input/diabetes-
dataset/diabetes.csv")
```

DATA CLEANING

C O D E

```
print(BLUE + "\nDATA CLEANING" +
RESET) Check for missing values
missing_values = df.isnull().sum()
print(GREEN + "Missing Values : " + RESET)
print(missing_values) Handle missing values
mean_fill = df.fillna(df.mean())
df.fillna(mean_fill, inplace=True) Check for
duplicate values duplicate_values =
df.duplicated().sum() print(GREEN +
"Duplicate Values : " + RESET)
print(duplicate_values)      Drop
duplicate                    values
df.drop_duplicates(inplace=True
) DATA ANALYSIS
print(BLUE + "\nDATA ANALYSIS" + RESET)
Summary Statistics summary_stats =
df.describe() print(GREEN + "Summary
Statistics : " + RESET)
print(summary_stats) Class Distribution
class_distribution =
```

C O D E

```
df["Outcome"].value_counts() print(GREEN +  
"Class Distribution : " + RESET) Support Vector  
Machine Modelling print(BLUE + "\nMODELLING"  
+ RESET) X = df.drop("Outcome", axis=1) y =  
df["Outcome"]  
Splitting the data into training and testing sets  
X_train, X_test, y_train, y_test = train_test_split(  
X, y, test_size=0.2, random_state=42  
)  
Standardize Features scaler =  
StandardScaler() X_train =  
scaler.fit_transform(X_train) X_test =  
scaler.transform(X_test) init and train SVM  
model model = svm.SVC(kernel="linear")  
model.fit(X_train, y_train) Predict on test  
data y_pred = model.predict(X_test)  
Evaluate model performance accuracy =  
model.score(X_test, y_test) print(GREEN +  
"Model Accuracy : " + RESET)  
print(accuracy)
```


C O D E

Classification Report and Confusion Matrix

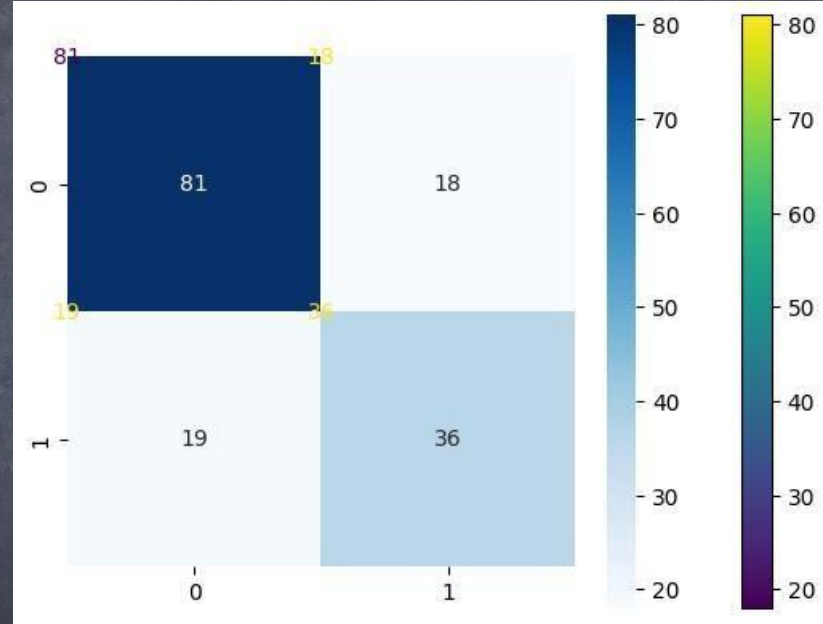
```
print(GREEN + "Classification Report : " + RESET)
print(classification_report(y_test, y_pred))
print(GREEN + "Confusion Matrix : " + RESET) cm
=
```

```
ConfusionMatrixDisplay.from_predictions(y_test,
y_pred)
sns.heatmap(cm.confusion_matrix, annot=True,
cmap="Blues")
plt.show()
print("Displayed")
```

SAVING THE FILE

```
df.to_csv("/kaggle/working/cleaned_diabetes.csv",
index=False) print(BLUE + "\nDATA SAVING" +
RESET) print(GREEN + "Data Cleaned and Saved !" +
RESET)
print("\n")
```

THIS REPRESENTATION REPRESENT THE DATA HANDLED BY
THE CODE WE PROVIDES AND IT GIVE THE
REPRESENTATION THROUGH GRAPHICAL WAY





THANK YOU !

REPORTED BY SANJAY.A
212921104044

PROJ_227128_TEAM_1