Diabetes Prediction in Machine Learning using Python

Dataset Link :--https://www.kaggle.com/datasets/uciml/pima-indians-diabetes-database

pip install pandas numpy scikit-learn matplotlib seaborn

import pandas as pd

import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import StandardScaler

from sklearn.linear\_model import LogisticRegression

from sklearn.svm import SVC

from sklearn.metrics import classification\_report, confusion\_matrix, accuracy\_score

# Load dataset

df = pd.read\_csv("diabetes.csv") # Ensure this file is in your working directory

# Explore the data

print("Dataset shape:", df.shape)

print(df.head())

# Check for missing values

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

# Replace zeroes in important columns with NaN for correction

cols\_with\_zeroes = ["Glucose", "BloodPressure", "SkinThickness", "Insulin", "BMI"]

df[cols\_with\_zeroes] = df[cols\_with\_zeroes].replace(0, np.NaN)

# Fill NaNs with column mean

df.fillna(df.mean(), inplace=True)

# Correlation heatmap

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

sns.heatmap(df.corr(), annot=True, cmap="coolwarm")

plt.title("Feature Correlation")

plt.show()

# Split features and target

X = df.drop("Outcome", axis=1)

y = df["Outcome"]

# Normalize features

scaler = StandardScaler()

X\_scaled = scaler.fit\_transform(X)

# Train/test split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X\_scaled, y, test\_size=0.25, random\_state=42)

# Logistic Regression Model

log\_model = LogisticRegression()

log\_model.fit(X\_train, y\_train)

y\_pred\_log = log\_model.predict(X\_test)

print("---- Logistic Regression Results ----")

print("Accuracy:", accuracy\_score(y\_test, y\_pred\_log))

print(confusion\_matrix(y\_test, y\_pred\_log))

print(classification\_report(y\_test, y\_pred\_log))

# SVM Model

svm\_model = SVC(kernel='linear') # You can try 'rbf' or 'poly'

svm\_model.fit(X\_train, y\_train)

y\_pred\_svm = svm\_model.predict(X\_test)

print("---- SVM Results ----")

print("Accuracy:", accuracy\_score(y\_test, y\_pred\_svm))

print(confusion\_matrix(y\_test, y\_pred\_svm))

print(classification\_report(y\_test, y\_pred\_svm))