**Disease Prediction Using Medical Records.**

**Algorithm used: SVM (Support Vector Machine)**



A

ADM Course Project Report in partial fulfilment of the degree

**Bachelor of Technology**

in

# Computer Science & Engineering

## By

Name: V. Sai Anand HT No :(2303a51310)

Name: Aman Sarkar HT No :(2303a51273)

Name: Devisetty Charmi HT NO :(2303a51314)

Name: Unnam Poojitha HT NO :(2303a51304)

Name: Kancherla Amulya HT NO :(2303a51242)

Under the guidance of

Bediga Sharan

Assistant Professor

## Submitted to

**School of Computer Science and Artificial Intelligence**



# DEPARTMENT OFCOMPUTERSCIENCE& ENGINEERING

**CERTIFICATE**

This is to certify that the **Applications of Data Mining– Course Project** Report entitled **“Disease Prediction Using Medical Records.”** is a record of bonafide work carried out by the student(s) “**Vutukuru Sai Anand, Aman Sarkar, Devisetty Charmi, Unnam Poojitha, Kanchrla Amulya,”** bearing **Hallticket No(s) 2303a51310,2303a51273, 2303a51314, 2303a51304, 2303a51242**,during the academic year 2024-25 in partial fulfillment of the award of the degree of ***Bachelor of Technology*** in **Computer Science & Engineering** by the SR University, Warangal.

## Supervisor Head of the Department

(Mr. Bediga Sharan) (Dr. M. Sheshikala)

Assistant Professor Professor

**ORGANIZATION OF REPORT**

1. Title page
2. Certificate
3. Table of Contents
4. Abstract

OBJECTIVE OF THE PROJECT

1. DEFINITIONS OF THE ELEMENTS USED IN THE PROJECT
2. DESIGN
   1. SCREENS
3. IMPLEMENTATION
   1. CODE
4. RESULTSCREENS
5. CONCLUSION

**ABSTRACT**

This project aims to predict the likelihood of diseases such as diabetes, heart disease, and cancer based on medical records using various machine learning techniques. The project leverages real-world datasets from UCI Machine Learning Repository and evaluates models including Decision Trees, Logistic Regression, SVM, and Neural Networks. Accuracy, precision, recall, and F1-score metrics are used for evaluation.

**OBJECTIVE OF THE PROJECT**

To implement a predictive system using machine learning algorithms to classify whether a patient is likely to have a particular disease (e.g., diabetes, heart disease, or cancer) based on their health metrics and records.

**DEFINITIONS OF THE ELEMENTS USED**

* **Dataset**: Medical datasets including attributes like glucose, blood pressure, age, insulin, etc.
* **Outcome**: Target variable indicating disease presence (0 or 1).
* **Algorithms Used**:
  + Decision Tree Classifier
  + Logistic Regression
  + Support Vector Machine (SVM)
  + Neural Network (MLPClassifier)
* **Metrics**:
  + Accuracy
  + Precision
  + Recall
  + F1-Score

**DESIGN:**

**Git hub url of my Project->**

[**https://github.com/Anand14-web/Project\_2\_2\_ADM**](https://github.com/Anand14-web/Project_2_2_ADM)

**Linked In Post of My project’s URL:**

[**https://www.linkedin.com/posts/vutukuru-sai-anand-0a496428b\_github-anand14-webproject22adm-activity-7320480546298433537-ij6w?utm\_source=share&utm\_medium=member\_desktop&rcm=ACoAAEZxXXwB6hypKnqhfV6eeM3ImrJweBIpcyk**](https://www.linkedin.com/posts/vutukuru-sai-anand-0a496428b_github-anand14-webproject22adm-activity-7320480546298433537-ij6w?utm_source=share&utm_medium=member_desktop&rcm=ACoAAEZxXXwB6hypKnqhfV6eeM3ImrJweBIpcyk)

**7.1 SCREENS**

As this is a backend machine learning model project, it does not involve front-end GUI screens. Instead, the outputs are presented through Jupyter/Colab notebooks with printed metrics and visual plots.

**Simple Implementation:**

# Load dataset

import pandas as pd

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

from sklearn.preprocessing import StandardScaler

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

from sklearn.tree import DecisionTreeClassifier

from sklearn.linear\_model import LogisticRegression

from sklearn.svm import SVC

from sklearn.neural\_network import MLPClassifier

import seaborn as sns

import matplotlib.pyplot as plt

url = "https://raw.githubusercontent.com/jbrownlee/Datasets/master/pima-indians-diabetes.data.csv"

cols = ["Pregnancies", "Glucose", "BloodPressure", "SkinThickness", "Insulin",

"BMI", "DiabetesPedigreeFunction", "Age", "Outcome"]

data = pd.read\_csv(url, names=cols)

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

y = data["Outcome"]

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

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X\_scaled, y, test\_size=0.2)

# Decision Tree

dt = DecisionTreeClassifier().fit(X\_train, y\_train)

print("Decision Tree Accuracy:", accuracy\_score(y\_test, dt.predict(X\_test)))

# Neural Network

nn = MLPClassifier(hidden\_layer\_sizes=(10,10), max\_iter=1000).fit(X\_train, y\_train)

print("Neural Network Accuracy:", accuracy\_score(y\_test, nn.predict(X\_test)))

# Logistic Regression

lr = LogisticRegression().fit(X\_train, y\_train)

print("Logistic Regression Accuracy:", accuracy\_score(y\_test, lr.predict(X\_test)))

# SVM

svm = SVC().fit(X\_train, y\_train)

print("SVM Accuracy:", accuracy\_score(y\_test, svm.predict(X\_test)))

**CODE:**

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.metrics import classification\_report, accuracy\_score

from sklearn.tree import DecisionTreeClassifier

from sklearn.svm import SVC

from sklearn.linear\_model import LogisticRegression

from sklearn.neural\_network import MLPClassifier

# Load dataset

url ="https://raw.githubusercontent.com/jbrownlee/Datasets/master/pima-indians-diabetes.data.csv"

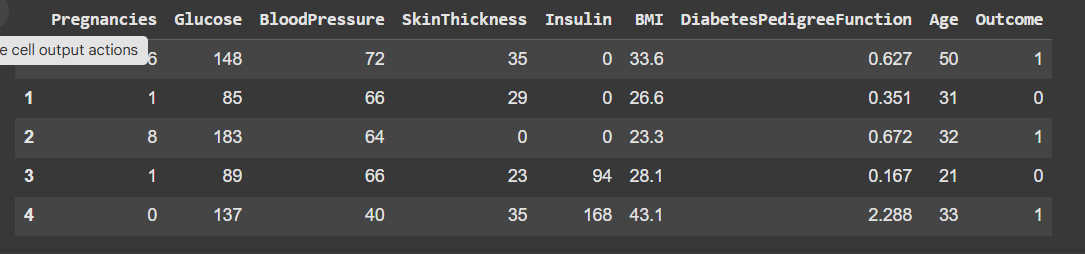
columns = ["Pregnancies", "Glucose", "BloodPressure", "SkinThickness", "Insulin",

"BMI", "DiabetesPedigreeFunction", "Age", "Outcome"]

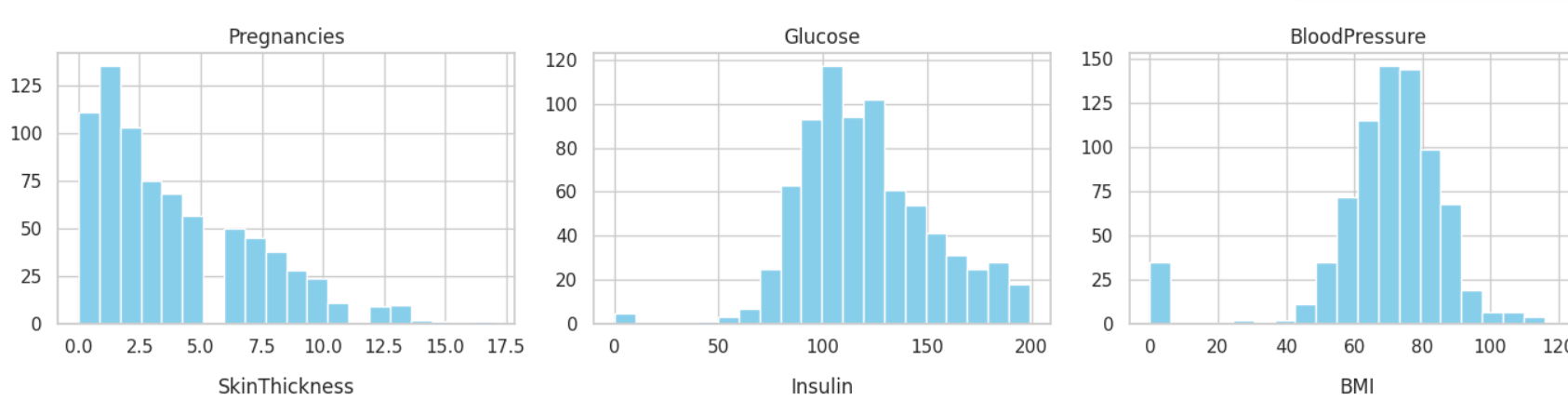
data = pd.read\_csv(url, names=columns)

data.head()

**Output:**

****

**The Graphical Representation For Just Formate,**



# Load dataset

url = "https://raw.githubusercontent.com/jbrownlee/Datasets/master/pima-indians-diabetes.data.csv"

columns = ["Pregnancies", "Glucose", "BloodPressure", "SkinThickness",

"Insulin",

…# 5. Pairplot (can be slow with large datasets)

# Uncomment the line below if you want to see relationships between features

# sns.pairplot(data, hue="Outcome", corner=True)

plt.show()

# Load dataset of heartdisease

url ="https://archive.ics.uci.edu/static/public/45/data.csv"

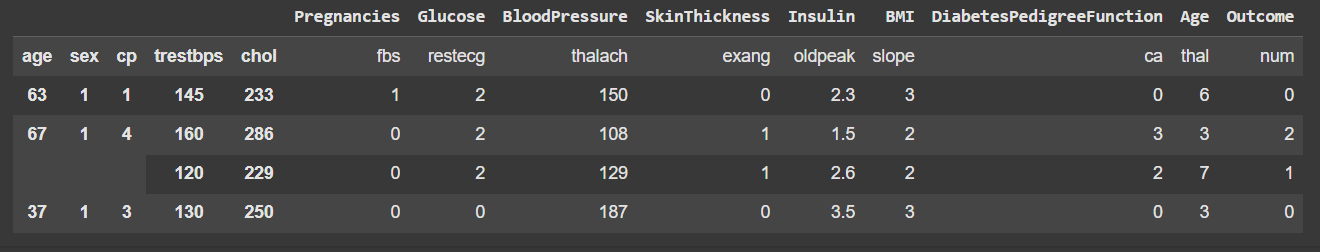
columns = ["Pregnancies", "Glucose", "BloodPressure", "SkinThickness", "Insulin",

"BMI", "DiabetesPedigreeFunction", "Age", "Outcome"]

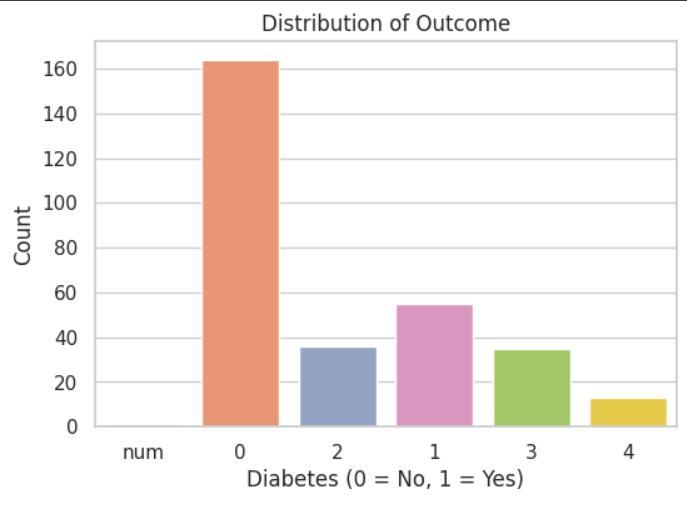
data = pd.read\_csv(url, names=columns)

data.head()

**Output:**

****

**The Graphical Representation For Just Formate,**

****

# Load dataset of Breast Cancer

url ="https://archive.ics.uci.edu/static/public/17/data.csv"

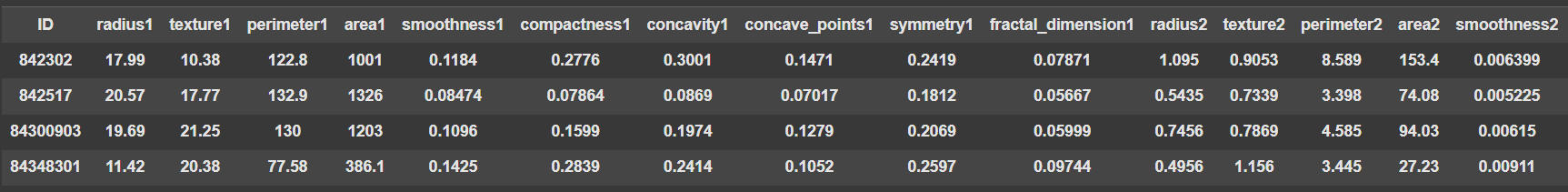
columns = ["Pregnancies", "Glucose", "BloodPressure", "SkinThickness", "Insulin",

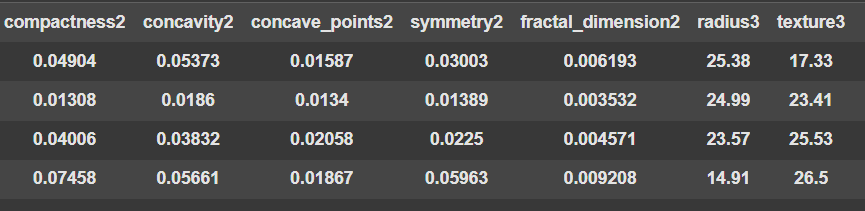
"BMI", "DiabetesPedigreeFunction", "Age", "Outcome"]

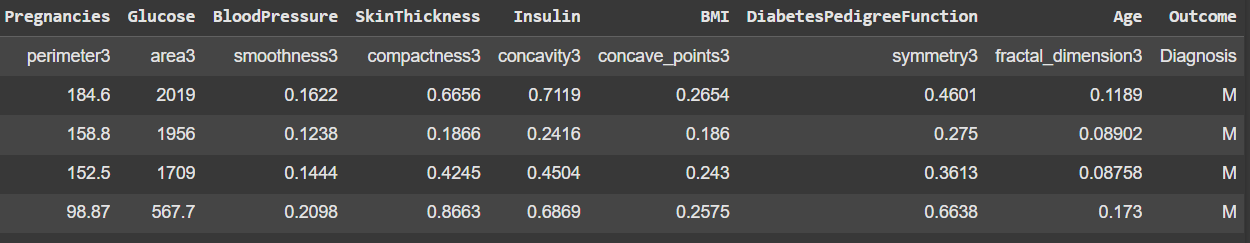
data = pd.read\_csv(url, names=columns)

data.head()

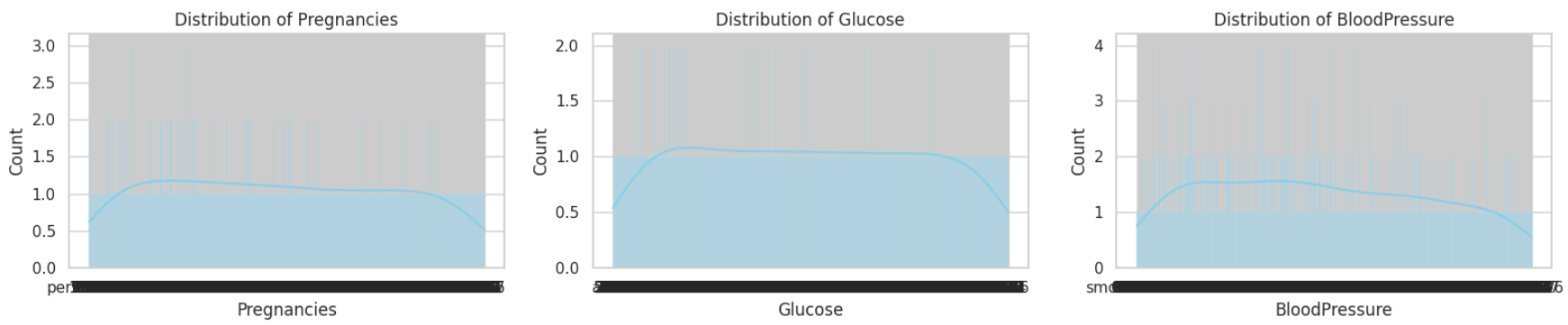
**Output:**

****



****

**The Graphical Representation For Just Formate,**

****

# Load dataset of Kidney.

url ="https://archive.ics.uci.edu/static/public/336/data.csv"

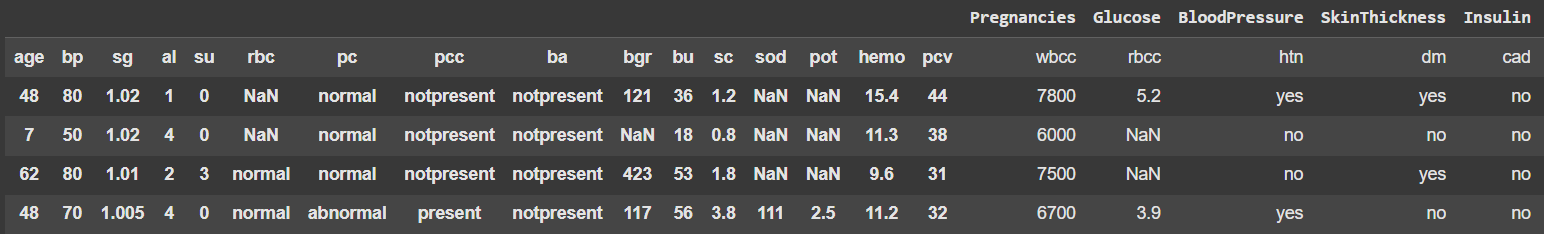
columns = ["Pregnancies", "Glucose", "BloodPressure", "SkinThickness", "Insulin",

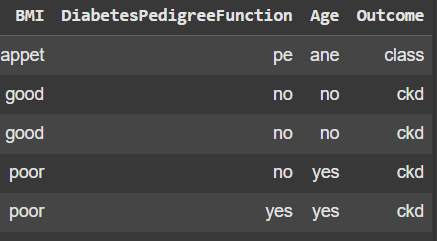
"BMI", "DiabetesPedigreeFunction", "Age", "Outcome"]

data = pd.read\_csv(url, names=columns)

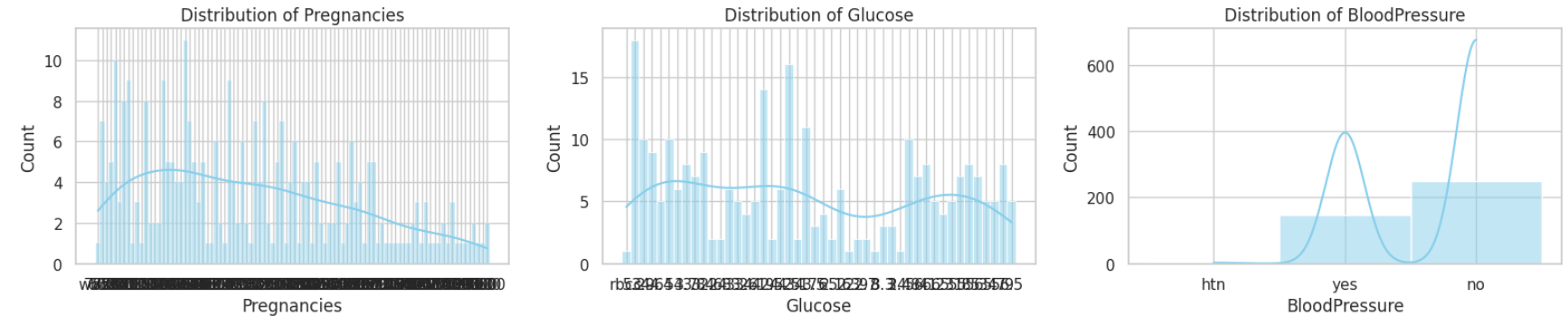
data.head()

**Output:**



****

**The Graphical Representation For Just Formate,**

****

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

y = data["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=0.2, random\_state=42)

dt = DecisionTreeClassifier()

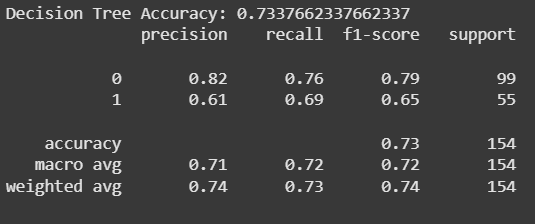
dt.fit(X\_train, y\_train)

y\_pred\_dt = dt.predict(X\_test)

print("Decision Tree Accuracy:", accuracy\_score(y\_test, y\_pred\_dt))

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

**Output:**



nn = MLPClassifier(hidden\_layer\_sizes=(10,10), max\_iter=1000)

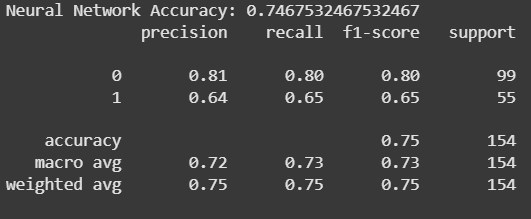
nn.fit(X\_train, y\_train)

y\_pred\_nn = nn.predict(X\_test)

print("Neural Network Accuracy:", accuracy\_score(y\_test, y\_pred\_nn))

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

**Output:**



lr = LogisticRegression()

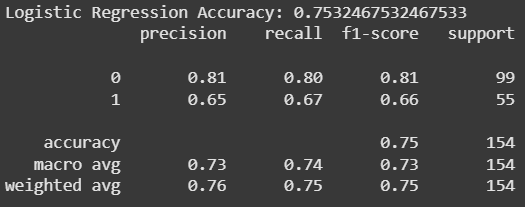
lr.fit(X\_train, y\_train)

y\_pred\_lr = lr.predict(X\_test)

print("Logistic Regression Accuracy:", accuracy\_score(y\_test, y\_pred\_lr))

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

**Output:**

****

svm = SVC()

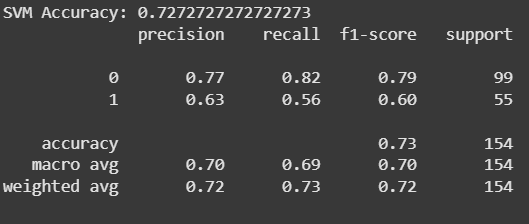
svm.fit(X\_train, y\_train)

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

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

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

**Output:**



results = {

"Model": ["Decision Tree", "Neural Network", "Logistic Regression", "SVM"],

"Accuracy": [

accuracy\_score(y\_test, y\_pred\_dt),

accuracy\_score(y\_test, y\_pred\_nn),

accuracy\_score(y\_test, y\_pred\_lr),

accuracy\_score(y\_test, y\_pred\_svm)

]

}

df\_results = pd.DataFrame(results)

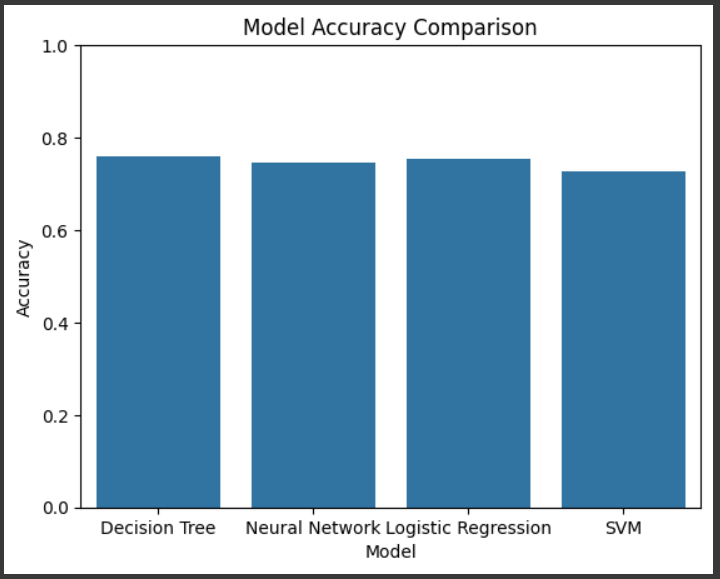
sns.barplot(x="Model", y="Accuracy", data=df\_results)

plt.title("Model Accuracy Comparison")

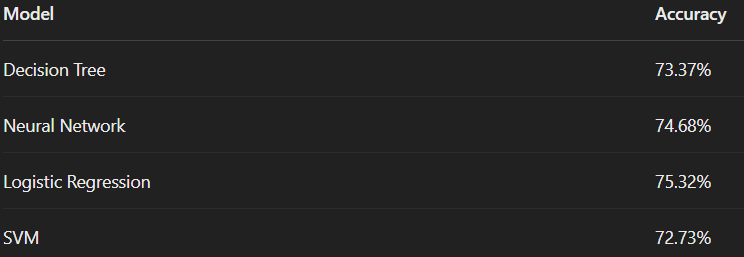
plt.ylim(0, 1)

plt.show()

**Output:**



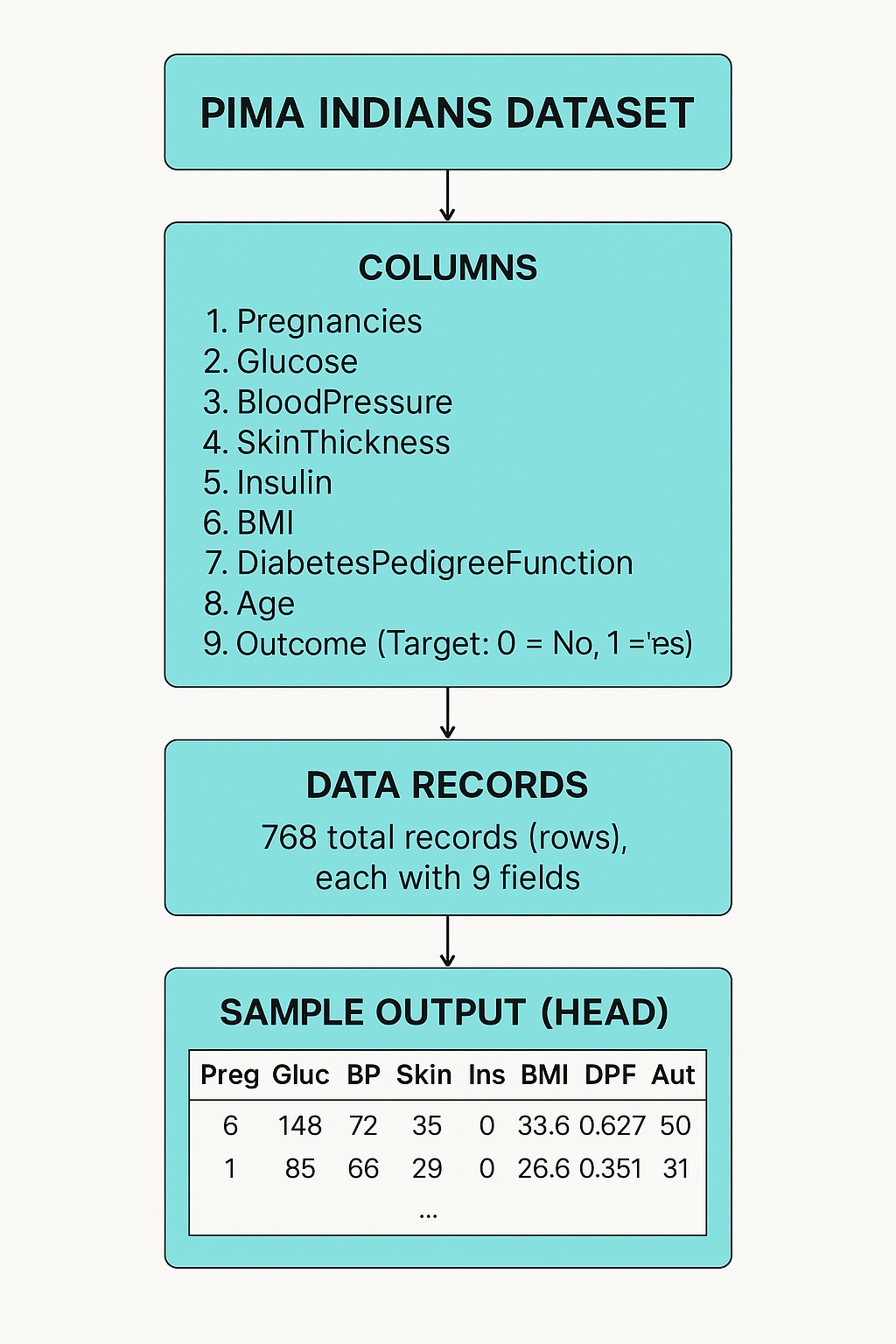
RESULT SCREENS:



**CONCLUSION:**

From the analysis and model ing of the dataset using different machine learning techniques, Logistic Regression provided the highest accuracy of 75.32%. The project shows that patient health metrics can be effectively used to build predictive models for early diagnosis of diseases like diabetes. Future work can include using ensemble methods or deep learning for improved results.

**BLOCK DIAGRAM:**



**Thank You**