**DEVELOPMENT OF AI BASED DIABETES PREDICTION SYSTEM PHASE-4**

**Introduction**

The development of AI-based diabetes prediction systems represents an important advancement in healthcare technology. Diabetes is a chronic condition affecting millions of people worldwide, and early detection and management can significantly improve patient outcomes. AI-driven prediction systems hold the potential to assist healthcare professionals and patients in identifying diabetes risk factors and enabling early intervention. Here's an introduction to the development of AI-based diabetes prediction systems:

**Data Preprocessing and Exploration**

Data preprocessing is a critical step in the machine learning and data analysis pipeline. It involves cleaning, transforming, and organising raw data into a format suitable for training machine learning models.

Data preprocessing can be a time-consuming and iterative process, but it is crucial for ensuring that your machine learning model can effectively learn from the data. The specific preprocessing steps you need to perform depend on the nature of your data, the problem you're trying to solve, and the algorithms you plan to use.

Here we use some machine learning algorithms to develop AI based diabetes prediction systems.Before building a machine learning model,we need to prepare the data.

Import the necessary libraries like numpy,pandas,scikit-learn and so on. And handle some missing data,if any.Split the dataset into training and testing sets.

#Importing the given dataset

from google.colab import files

uploaded=files.upload()

import pandas as pd

import io

df=pd.read\_csv(io.BytesIO(uploaded['diabetes.csv']))

# Import necessary libraries

import numpy as np

import pandas as pd

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

from sklearn.preprocessing import StandardScaler

from sklearn.linear\_model import LogisticRegression

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

from sklearn.naive\_bayes import GaussianNB

from sklearn.ensemble import RandomForestClassifier

import matplotlib.pyplot as plt

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

from sklearn.linear\_model import LinearRegression

from sklearn.metrics import mean\_squared\_error, r2\_score

**Feature Selection**

Feature selection is the process of isolating the most consistent, non-redundant, and relevant features to use in model construction.

X = df[['BloodPressure' ]]

y = df[['Age']]

**Model Selection**

Artificial Intelligence (AI) and machine learning encompass a wide range of algorithms and techniques. These algorithms can be categorized into several common types based on their functionality and application. Here are some popular AI and machine learning algorithms:

**Linear Regression**

Used for regression tasks, linear regression models the relationship between a dependent variable and one or more independent variables by fitting a linear equation.

**Logistic Regression:**

Used for classification tasks, logistic regression models the probability of a binary outcome.

**Decision Trees:**

A tree-like structure that makes decisions by splitting data into branches based on features. Decision trees can be used for both classification and regression.

**Random Forest:**

An ensemble learning method that combines multiple decision trees to improve accuracy and reduce overfitting.

**Support Vector Machines:**

Used for classification and regression, SVM finds a hyperplane that best separates data into distinct classes.

**K-Nearest Neighbors (K-NN):**

A simple instance-based learning algorithm for classification and regression that classifies data points based on the majority class of their k-nearest neighbours.

**Naive Bayes:**

A probabilistic algorithm based on Bayes' theorem used for classification tasks, particularly in text and document classification.

**Model Training**

Model training is a crucial step in the development of machine learning and AI models. It involves using a dataset to teach a model to make predictions or decisions. The trained model learns patterns, relationships, and characteristics within the data, allowing it to generalise and make predictions on new, unseen data.

# Train the model on the training data

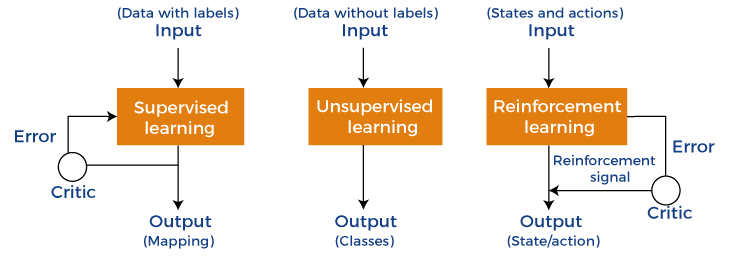
model.fit(X\_train, y\_train)

The process of training a machine learning model is iterative and may involve multiple cycles of data preprocessing, model selection, and tuning. The goal is to develop a model that generalises well to new, unseen data and provides valuable insights or predictions for your specific application.

### **Classification of Model Training**

### Based on different business goals and data sets, there are three learning models for algorithms. Each machine learning algorithm settles into one of the three models:

* **Supervised Learning**
* **Unsupervised Learning**
* **Reinforcement Learning**

****

**Model Evaluation**

Model evaluation is the process of using different evaluation metrics to understand a machine learning model's performance, as well as its strengths and weaknesses.

**# Make predictions on the test data**

**y\_pred = model.predict(X\_test)**

**# Calculate accuracy and other classification metrics**

**accuracy = accuracy\_score(y\_test, y\_pred)**

**report = classification\_report(y\_test, y\_pred)**

**# Print the results**

**print(f'Accuracy: {accuracy}')**

**print('Classification Report:\n', report)**

**Visualisation**

Data visualisation is a crucial aspect of machine learning that enables analysts to understand and make sense of data patterns, relationships, and trends. Through data visualisation, insights and patterns in data can be easily interpreted and communicated to a wider audience, making it a critical component of machine learning. In this article, we will discuss the significance of data visualisation in machine learning, its various types, and how it is used in the field.

# Plot the training data

plt.scatter(X\_train, y\_train, label='Training Data')

plt.plot(X\_test, y\_pred, color='red', linewidth=3, label='Regression Line')

plt.xlabel('X')

plt.ylabel('y')

plt.legend()

plt.show()

**Example Code**

**Let’s see the above steps, we’ve selected the linear regression model for machine learning algorithm. First we’ve imported the given dataset and necessary libraries .And then split the data into training and testing sets, and create linear regression model and then train model on training data. Make the prediction on the test data and calculate the mean squared error and print the result.Finally plot the training data and regression line and visualise the result with help of matplotlib.**

**#LINEAR REGRESSION**

**X = df[['BloodPressure' ]]**

**y = df[['Age']]**

**# Split 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)**

**# Create a linear regression model**

**model = LinearRegression()**

**# Train the model on the training data**

**model.fit(X\_train, y\_train)**

**# Make predictions on the test data**

**y\_pred = model.predict(X\_test)**

**# Calculate the mean squared error and R-squared (coefficient of determination)**

**mse = mean\_squared\_error(y\_test, y\_pred)**

**r2 = r2\_score(y\_test, y\_pred)**

**# Print the results**

**print(f'Mean Squared Error: {mse}')**

**print(f'R-squared: {r2}')**

**# Plot the training data and the regression line**

**plt.scatter(X\_train, y\_train, label='Training Data')**

**plt.plot(X\_test, y\_pred, color='red', linewidth=3, label='Regression Line')**

**plt.xlabel('X')**

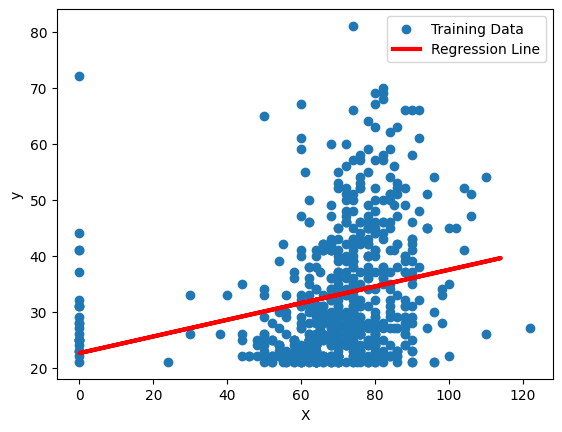
**plt.ylabel('y')**

**plt.legend()**

**plt.show()**

**Mean Squared Error: 153.53893981827423**

**R-squared: 0.039590936856064185**

****

**#NAIVE BAYES CLASSIFIER**

**X = df[['Glucose' ]]**

**y = df[['Age']]**

**# Split 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)**

**# Create a Gaussian Naive Bayes model**

**model = GaussianNB()**

**# Train the model on the training data**

**model.fit(X\_train, y\_train)**

**# Make predictions on the test data**

**y\_pred = model.predict(X\_test)**

**# Calculate accuracy and other classification metrics**

**accuracy = accuracy\_score(y\_test, y\_pred)**

**report = classification\_report(y\_test, y\_pred)**

**# Print the results**

**print(f'Accuracy: {accuracy}')**

**print('Classification Report:\n', report)**

**Accuracy: 0.07792207792207792**

**Classification Report:**

**precision recall f1-score support**

**21 0.09 0.50 0.15 12**

**22 0.08 0.38 0.13 13**

**23 0.00 0.00 0.00 7**

**24 0.00 0.00 0.00 10**

**25 0.00 0.00 0.00 8**

**26 0.00 0.00 0.00 3**

**27 0.00 0.00 0.00 5**

**28 0.00 0.00 0.00 9**

**29 0.00 0.00 0.00 10**

**30 0.00 0.00 0.00 4**

**31 0.00 0.00 0.00 3**

**32 0.00 0.00 0.00 4**

**33 0.00 0.00 0.00 2**

**34 0.00 0.00 0.00 2**

**36 0.00 0.00 0.00 4**

**37 0.00 0.00 0.00 3**

**38 0.00 0.00 0.00 6**

**39 0.00 0.00 0.00 4**

**40 0.00 0.00 0.00 2**

**41 0.11 0.33 0.17 3**

**42 0.00 0.00 0.00 4**

**43 0.00 0.00 0.00 4**

**44 0.00 0.00 0.00 3**

**45 0.00 0.00 0.00 2**

**48 0.00 0.00 0.00 1**

**49 0.00 0.00 0.00 1**

**50 0.00 0.00 0.00 2**

**51 0.00 0.00 0.00 1**

**53 0.00 0.00 0.00 2**

**54 0.00 0.00 0.00 2**

**55 0.00 0.00 0.00 1**

**56 0.00 0.00 0.00 1**

**57 0.00 0.00 0.00 1**

**58 0.00 0.00 0.00 4**

**60 0.00 0.00 0.00 3**

**61 0.00 0.00 0.00 0**

**62 0.00 0.00 0.00 3**

**63 0.00 0.00 0.00 2**

**64 0.00 0.00 0.00 0**

**65 0.00 0.00 0.00 2**

**67 0.00 0.00 0.00 1**

**68 0.00 0.00 0.00 0**

**70 0.00 0.00 0.00 0**

**72 0.00 0.00 0.00 0**

**81 0.00 0.00 0.00 0**

**accuracy 0.08 154**

**macro avg 0.01 0.03 0.01 154**

**weighted avg 0.02 0.08 0.03 154**

**Conclusion**

The development of AI-based diabetes prediction systems represents a significant advancement in the field of healthcare and data science. Such systems have the potential to revolutionise the management of diabetes and improve patient outcomes. In conclusion, the development of AI-based diabetes prediction systems offers several key benefits and implications:

In summary, the development of AI-based diabetes prediction systems holds great promise in improving diabetes care, reducing healthcare costs, and ultimately enhancing the quality of life for individuals with or at risk of diabetes. However, responsible development, ethical considerations, validation, and continuous improvement are essential aspects of this promising technology.