

XCSE63 – DATA MINING

**MINI PROJECT REPORT**

ON

**CLASSIFICATION OF DIABETES DISEASE BY USING LOGISTIC REGRESSION , GAUSSIAN NAIVE BAYES , DECISION TREE , SUPPORT VECTOR MACHINE AND NEURAL NETWORKS**

Submitted by

C.NATARAJ (121012012756)

M.SANTHOSH(121012012762)

B.MOKESH (1210120127755)

B .Tech CSE ( III Year)

**Under the Supervision of**

**Dr.S.Appavu alias Balamurugan**

**Professor/CSE**

**Declaration**

We, the undersigned members of Group 1, hereby declare that the Mini–Project Report titled ‘ Classification Of Diabetes Disease By Using Logistic Regression , Gaussian Naive Bayes , Decision Tree , Support Vector Machine And Neural Networks**’** submitted to the Department of Computer Science and Engineering, Periyar Maniammai Institute of Science & Technology (Deemed to be University), Thanjavur, for the course XCSE63 Data Mining, is a record of our original work. We affirm that this report has not been submitted for any other degree or diploma, and the content of this report has not been plagiarized from any other source.

Members of the Group:

1. C.NATARAJ
2. M.SANTHOSH
3. B.MOKESH

**Date: 30/04/2024**

**ABSTRACT :**

The classification of diabetes disease using five distinct machine learning algorithms: logistic regression, Gaussian Naive Bayes, decision tree, support vector machine (SVM), and neural networks. Diabetes is a chronic metabolic disorder with significant public health implications, and early detection plays a critical role in effective management and prevention of complications. The study aims to compare the performance of these algorithms in accurately predicting diabetes status based on patient demographics, medical history, and relevant clinical measurements.

The dataset consists of features such as age, gender, body mass index (BMI), blood pressure, and glucose levels, collected from a cohort of patients. Preprocessing techniques including data cleaning, normalization, and feature selection are applied to prepare the dataset for model training and evaluation. Each machine learning algorithm is implemented and tuned using appropriate hyperparameters to optimize predictive performance.

Evaluation metrics such as accuracy, precision, recall, F1-score, and area under the receiver operating characteristic curve (AUC-ROC) are used to assess the models' performance. Additionally, interpretability, computational complexity, and scalability considerations are taken into account when comparing the algorithms.

The results of the study provide insights into the strengths and limitations of each algorithm in classifying diabetes disease. Such findings can inform healthcare practitioners and policymakers in selecting suitable models for diabetes risk assessment and personalized treatment planning. Furthermore, future research directions may focus on incorporating real-time data, developing user-friendly applications, and integrating predictive models into clinical workflows to enhance diabetes management and patient care.

**INTRODUCTION :**

Machine learning is a subfield of artificial intelligence (AI). The goal of machine learning generally is to understand the structure of data and fit that data into models that can be understood and utilized by people.

Although machine learning is a field within computer science, it differs from traditional computational approaches. In traditional computing, algorithms are sets of explicitly programmed instructions used by computers to calculate or problem solve. Machine learning algorithms instead allow for computers to train on data inputs and use statistical analysis in order to output values that fall within a specific range. Because of this, machine learning facilitates computers in building models from sample data in order to automate decision-making processes based on data inputs.

Any technology user today has benefitted from machine learning. Facial recognition technology allows social media platforms to help users tag and share photos of friends. Optical character recognition (OCR) technology converts images of text into movable type. Recommendation engines, powered by machine learning, suggest what movies or television shows to watch next based on user preferences. Self-driving cars that rely on machine learning to navigate may soon be available to consumers.

Machine learning is a continuously developing field. Because of this, there are some considerations to keep in mind as you work with machine learning methodologies, or analyze the impact of machine learning processes.

In machine learning, tasks are generally classified into broad categories. These categories are based on how learning is received or how feedback on the learning is given to the system developed.

Two of the most widely adopted machine learning methods

are supervised learning which trains algorithms based on example input and output data that is labeled by humans, and unsupervised learning which provides the algorithm with no labeled data in order to allow it to find structure within its input data

# PROBLEM DEFINITION :

Problem Statement:

Diabetes is one of deadliest infections on the planet. It isn’t just an ailment yet additionally a maker of various types of maladies like heart assault, visual deficiency, kidney infections, and so on. The typical recognizing process is that patients need to visit an indicative focus, counsel their specialist, and sit tight for a day or more to get their reports. Also, every time they need to get their conclusion report, they need to squander their cash futile. Be that as it may, with the ascent of Machine Learning approaches we can discover an answer for this issue, we have built up a framework utilizing information mining which can anticipate whether the patient has diabetes or not. Moreover, foreseeing the illness early prompts treating the patients previously it winds up basic. Information mining can remove concealed learning from a colossal measure of diabetes-related information. Therefore, it has a critical part in diabetes examine, now like never before.

Objective

The objective of the project work is to build Five classification framework using Logistic Regression ,Support Vector Machine , Decision Tree, Naïve Bayes, and Neural Networks, which can individually predict whether a patient is having Diabetic or not from the patient data and finally compare the classification accuracy of the three classifiers.

**PROJECT OVERVIEW :**

In this project we have used five different algorithms to find result . The five different algorithms are

* + - DECISION TREE CLASSIFIER
    - GAUSSIAN NAIVE BAYES
    - NEURAL NETWORK
    - SUPPORT VECTOR MACHINE
    - LOGISTIC REGRESSION

**Decision Tree** :

Decision Tree is a Supervised learning technique that can be used for both classification and Regression problems,but mostly it is preferred for solving Classification problems. It is tree-structured classifier,where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome.In a Decision tree, there are two nodes, which are the Decision Node and Leaf Node. Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.The decisions or the test are performed on the basis of features of the given dataset.It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.It is called a decision tree because, similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure.In order to build a tree, we use the CART

algorithm, which stands for Classification and Regression Tree algorithm.A decision tree simply asks a question, and based on the answer (Yes/No), it further split the tree into subtrees.

**Naive Bayes:**

Naïve Bayes algorithm is a supervised learning algorithm, which is based on Bayes theorem and used for solving classification problems.It is mainly used in text classification that includes a high-dimensional training dataset.Naïve Bayes Classifier is one of the simple and most effective Classification algorithms which helps in building the fast machine learning models that can make quick predictions.It is a probabilistic classifier, which means it predicts on the basis of the probability of an object.Some popular examples of Naïve Bayes Algorithm are spam filtration, Sentimental analysis, and classifying articles.

**Neural Network :**

The term "Artificial Neural Network" is derived from Biological neural networks that develop the structure of a human brain. Similar to the human brain that has neurons interconnected to one another, artificial neural networks also have neurons that are interconnected to one another in various layers of the networks. These neurons are known as nodes. To understand the concept of the architecture of an artificial neural network, we have to understand what a neural network consists of. In order to define a neural network that consists of a large number of artificial neurons,

which are termed units arranged in a sequence of layers. Various types of layers available in an artificial neural network.

Artificial Neural Network primarily consists of three layers:

**Input Layer:**

As the name suggests, it accepts inputs in several different formats provided by the programmer.

**Hidden Layer:**

The hidden layer presents in-between input and output layers. It performs all the calculations to find hidden features and patterns.

**Output Layer:**

The input goes through a series of transformations using the hidden layer, which finally results in output that is conveyed using this layer.

The artificial neural network takes input and computes the weighted sum of the inputs and includes a bias. This computation is represented in the form of a transfer function.it determines weighted total is passed as an input to an activation function to produce the output. Activation functions choose whether a node should fire or not. Only those who are fired make it to the output layer.

**Logistic Regression :**

Logistic regression is a statistical method used for binary classification tasks, such as predicting whether an email is spam or not, or whether a patient has a disease. Unlike linear regression, logistic regression predicts the probability of an input belonging to a particular category, using the logistic function to map inputs to probabilities between 0 and 1. It's valued for its simplicity, interpretability, and effectiveness in many practical applications.

**Support Vector Machine :**

Support Vector Machine (SVM) is a powerful algorithm used for classification and regression tasks. It finds the hyperplane that best separates different classes while maximizing the margin between them. SVM works well in high-dimensional spaces and can handle both linearly and non-linearly separable data by using different kernel functions. Its decision function depends only on a subset of the training data, making it memory efficient. SVM is widely used in various fields due to its effectiveness in tasks such as text classification, image recognition, and bioinformatics.

# HARDWARE SPECIFICATION :

The project is designed to run on any device (Desktop or Laptop) with Windows Latest Version.

# SOFTWARE SPECIFICATION :

* Google Colaboratory
* Windows 10 10.0 • IDE – Pandas
* Language-Python

# BACKGROUND STUDY:

Analysing the train and test data , we need to build five different models

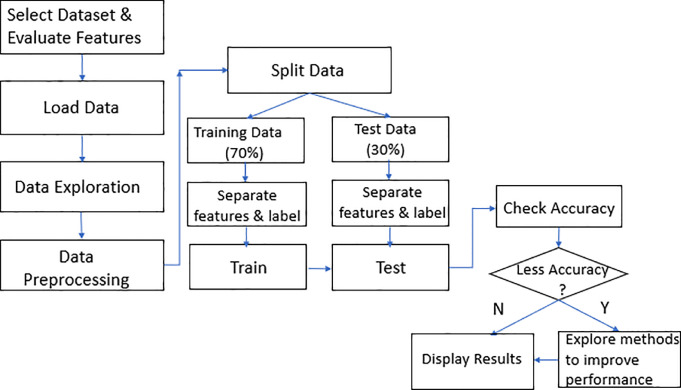
For training the data and then the best model is used to get the results of the testing data .

**Medical Background on Diabetes**:

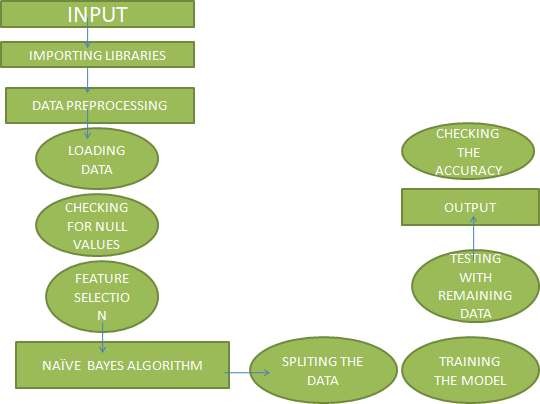
* Diabetes is a chronic metabolic disorder characterized by high blood sugar levels over a prolonged period.
* It has various types, including Type 1, Type 2, and gestational diabetes, each with its own causes, symptoms, and risk factors.
* Diagnosis often involves measuring blood glucose levels and considering symptoms such as increased thirst, frequent urination, and unexplained weight loss.
* Early detection and management of diabetes are crucial to prevent complications such as cardiovascular diseases, kidney failure, and vision problems

**PROPOSED MODEL :**

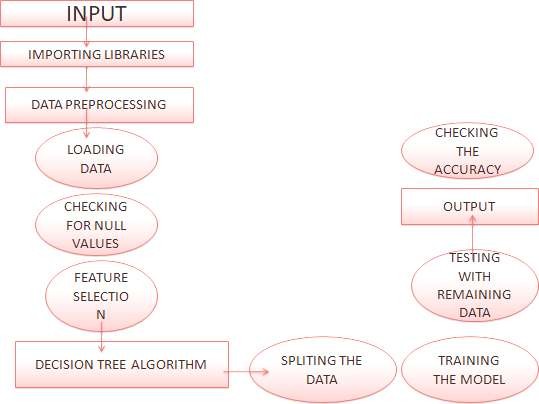
**LOGISTIC REGRESSION :**



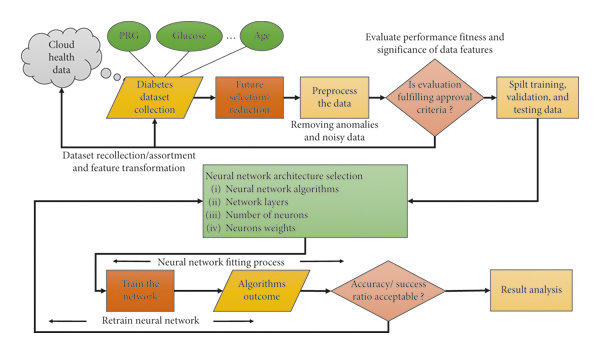
**GAUSSIAN NAIVE BAYES :**



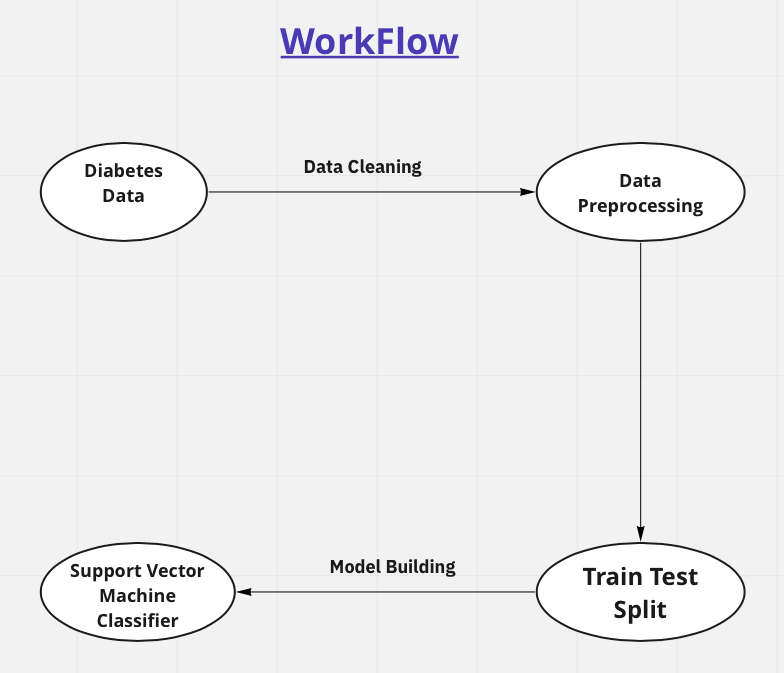
**DECISION TREE CLASSIFIER :**



**NEURAL NETWORKS :**



**SUPPORT VECTOR MACHINE :**



**Implementation of Classification Of Diabetes Disease :**

#Install required libraries

import os

import sys

from tempfile import NamedTemporaryFile

from urllib.request import urlopen

from urllib.parse import unquote, urlparse

from urllib.error import HTTPError

from zipfile import ZipFile

import tarfile

import shutil

CHUNK\_SIZE = 40960

DATA\_SOURCE\_MAPPING = 'diabetes-prediction-dataset:https%3A%2F%2Fstorage.googleapis.com%2Fkaggle-data '

KAGGLE\_INPUT\_PATH='/kaggle/input'

KAGGLE\_WORKING\_PATH='/kaggle/working'

KAGGLE\_SYMLINK='kaggle'

!umount /kaggle/input/ 2> /dev/null

shutil.rmtree('/kaggle/input', ignore\_errors=True)

os.makedirs(KAGGLE\_INPUT\_PATH, 0o777, exist\_ok=True)

os.makedirs(KAGGLE\_WORKING\_PATH, 0o777, exist\_ok=True)

try:

  os.symlink(KAGGLE\_INPUT\_PATH, os.path.join("..", 'input'), target\_is\_directory=True)

except FileExistsError:

  pass

try:

  os.symlink(KAGGLE\_WORKING\_PATH, os.path.join("..", 'working'), target\_is\_directory=True)

except FileExistsError:

  pass

for data\_source\_mapping in DATA\_SOURCE\_MAPPING.split(','):

    directory, download\_url\_encoded = data\_source\_mapping.split(':')

    download\_url = unquote(download\_url\_encoded)

    filename = urlparse(download\_url).path

    destination\_path = os.path.join(KAGGLE\_INPUT\_PATH, directory)

    try:

        with urlopen(download\_url) as fileres, NamedTemporaryFile() as tfile:

            total\_length = fileres.headers['content-length']

            print(f'Downloading {directory}, {total\_length} bytes compressed')

            dl = 0

            data = fileres.read(CHUNK\_SIZE)

            while len(data) > 0:

                dl += len(data)

                tfile.write(data)

                done = int(50 \* dl / int(total\_length))

                sys.stdout.write(f"\r[{'=' \* done}{' ' \* (50-done)}] {dl} bytes downloaded")

                sys.stdout.flush()

                data = fileres.read(CHUNK\_SIZE)

            if filename.endswith('.zip'):

              with ZipFile(tfile) as zfile:

                zfile.extractall(destination\_path)

            else:

              with tarfile.open(tfile.name) as tarfile:

                tarfile.extractall(destination\_path)

            print(f'\nDownloaded and uncompressed: {directory}')

    except HTTPError as e:

        print(f'Failed to load (likely expired) {download\_url} to path {destination\_path}')

        continue

    except OSError as e:

        print(f'Failed to load {download\_url} to path {destination\_path}')

        continue

print('Data source import complete.')

#Install Required Libraries

!pip install scikit-optimize

import time

import pandas as pd

import numpy as np

import scipy as sp

import scipy.stats as stats

import matplotlib.pyplot as plt

import seaborn as sns

import sklearn

import sklearn.model\_selection

from sklearn.manifold import TSNE

from sklearn.metrics import f1\_score

from sklearn.metrics import accuracy\_score

from sklearn.metrics import precision\_score

from sklearn.metrics import recall\_score

from sklearn.metrics import confusion\_matrix

from sklearn.metrics import classification\_report

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

from sklearn.neighbors import KNeighborsClassifier

from sklearn.linear\_model import LogisticRegression

from sklearn.tree import DecisionTreeClassifier

from sklearn.ensemble import RandomForestClassifier

from sklearn.svm import SVC

from sklearn.neural\_network import MLPClassifier

from sklearn.ensemble import RandomForestClassifier

from sklearn.naive\_bayes import GaussianNB

from sklearn.linear\_model import SGDClassifier

from skopt import gp\_minimize

from skopt.space import Real, Integer

from skopt.utils import use\_named\_args

from skopt.plots import plot\_convergence

import seaborn as sns

sns.set\_style("darkgrid")

import warnings

warnings.filterwarnings("ignore")

df = pd.read\_csv('/kaggle/input/diabetes-prediction-dataset/diabetes\_prediction\_dataset.csv')

df.head()

df.shape

df.dtypes

df.isna().sum()

df['age'] = df['age'].astype(int)

df['smoking\_history'].value\_counts()

df['gender'].value\_counts()

df = df.drop(columns = 'smoking\_history')

df.describe()

df['blood\_glucose\_level'] = df['blood\_glucose\_level'].astype(float)

df.dtypes

df.describe()

sns.catplot(data = df, x = 'diabetes', y = 'blood\_glucose\_level', kind = 'box', hue = 'gender')

plt.show()

sns.catplot(data = df, x = 'diabetes', y = 'age', kind = 'box', hue = 'gender')

plt.show()

sns.catplot(data = df, x = 'hypertension', y = 'age', kind = 'box', hue = 'gender')

plt.show()

sns.relplot(data = df, x = 'HbA1c\_level', y = 'blood\_glucose\_level', kind = 'line', hue = 'gender', ci = None, markers = True)

plt.show()

sns.relplot(data = df, x = 'blood\_glucose\_level', y = 'bmi', kind = 'line', hue = 'gender', ci = None)

plt.show()

design\_matrix = df.iloc[:,:-1]

design\_matrix = pd.get\_dummies(design\_matrix, columns= ['gender'], drop\_first=True)

response\_y = df.iloc[:,-1]

design\_matrix

response\_y

X\_train, X\_test, y\_train, y\_test = train\_test\_split(design\_matrix, response\_y, test\_size = 0.2, random\_state = 110)

print(X\_train.shape)

print(y\_train.shape)

print(X\_test.shape)

print(y\_test.shape)

# Define a dictionary to store the results

results = {}

start\_time = time.time()

logistic = LogisticRegression()

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

y\_pred = logistic.predict(X\_test)

end\_time = time.time()

training\_time = end\_time - start\_time

results['Logistic Regression'] = [accuracy\_score(y\_test, y\_pred),

precision\_score(y\_test, y\_pred, average='weighted'),

recall\_score(y\_test, y\_pred, average='weighted'),

f1\_score(y\_test, y\_pred, average='weighted'),

training\_time]

start\_time = time.time()

gnb = GaussianNB()

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

y\_pred = gnb.predict(X\_test)

end\_time = time.time()

training\_time = end\_time - start\_time

results['GaussianNB'] = [accuracy\_score(y\_test, y\_pred),

precision\_score(y\_test, y\_pred, average='weighted'),

recall\_score(y\_test, y\_pred, average='weighted'),

f1\_score(y\_test, y\_pred, average='weighted'),

training\_time]

start\_time = time.time()

dt = DecisionTreeClassifier()

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

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

end\_time = time.time()

training\_time = end\_time - start\_time

results['Decision Trees'] = [accuracy\_score(y\_test, y\_pred),

precision\_score(y\_test, y\_pred, average='weighted'),

recall\_score(y\_test, y\_pred, average='weighted'),

f1\_score(y\_test, y\_pred, average='weighted'),

training\_time]

start\_time = time.time()

svm = SVC()

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

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

end\_time = time.time()

training\_time = end\_time - start\_time

results['Support Vector Machines'] = [accuracy\_score(y\_test, y\_pred),

precision\_score(y\_test, y\_pred, average='weighted'),

recall\_score(y\_test, y\_pred, average='weighted'),

f1\_score(y\_test, y\_pred, average='weighted'),

training\_time]

start\_time = time.time()

mlp = MLPClassifier()

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

y\_pred = mlp.predict(X\_test)

end\_time = time.time()

training\_time = end\_time - start\_time

results['Neural Networks (Multi-layer Perceptron)'] = [accuracy\_score(y\_test, y\_pred),

precision\_score(y\_test, y\_pred, average='weighted'),

recall\_score(y\_test, y\_pred, average='weighted'),

f1\_score(y\_test, y\_pred, average='weighted'),

training\_time]

# Create a DataFrame from the results

df\_results = pd.DataFrame.from\_dict(results, orient='index', columns=['Accuracy', 'Precision', 'Recall', 'F1-Score', 'Training Time'])

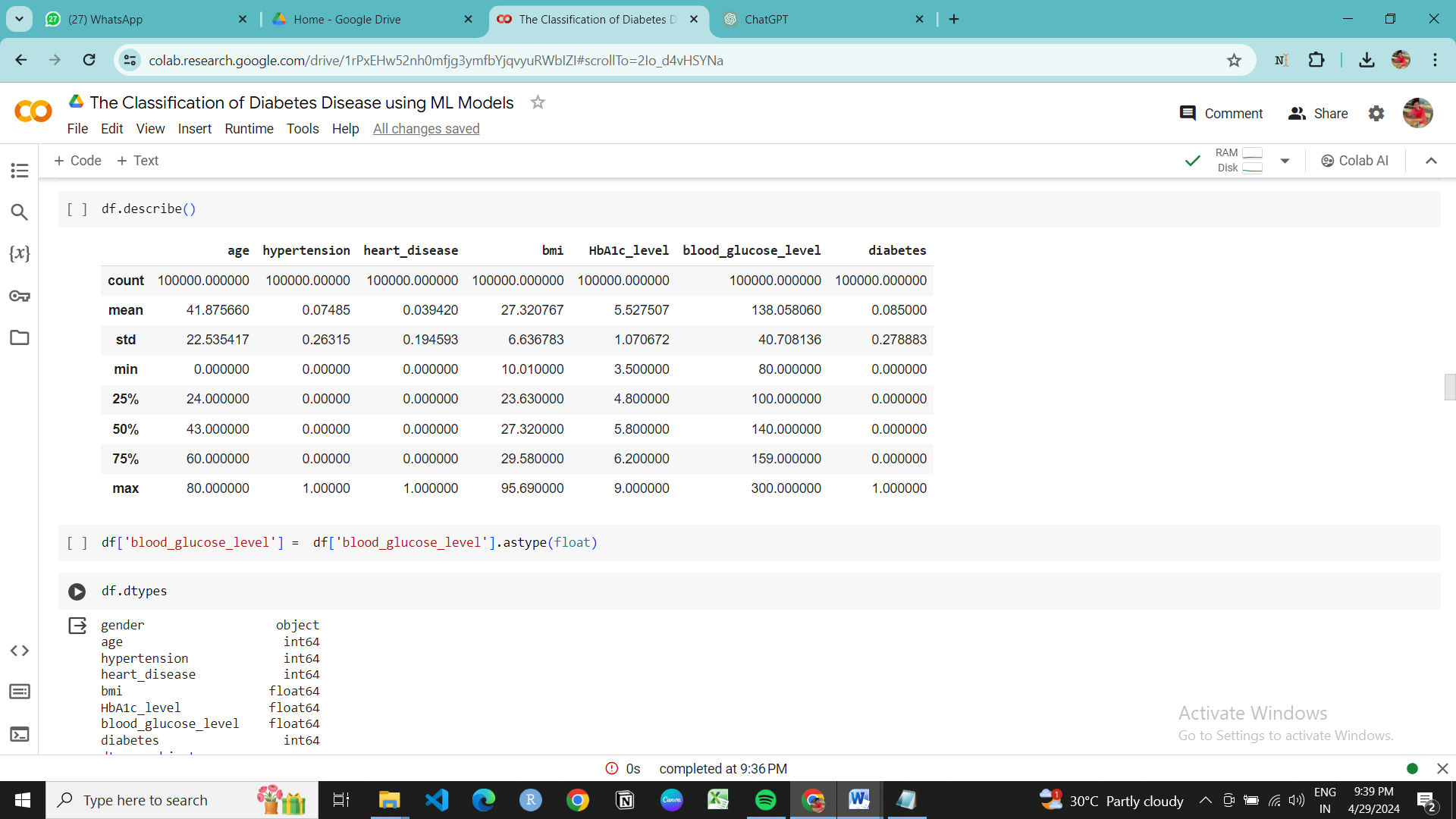
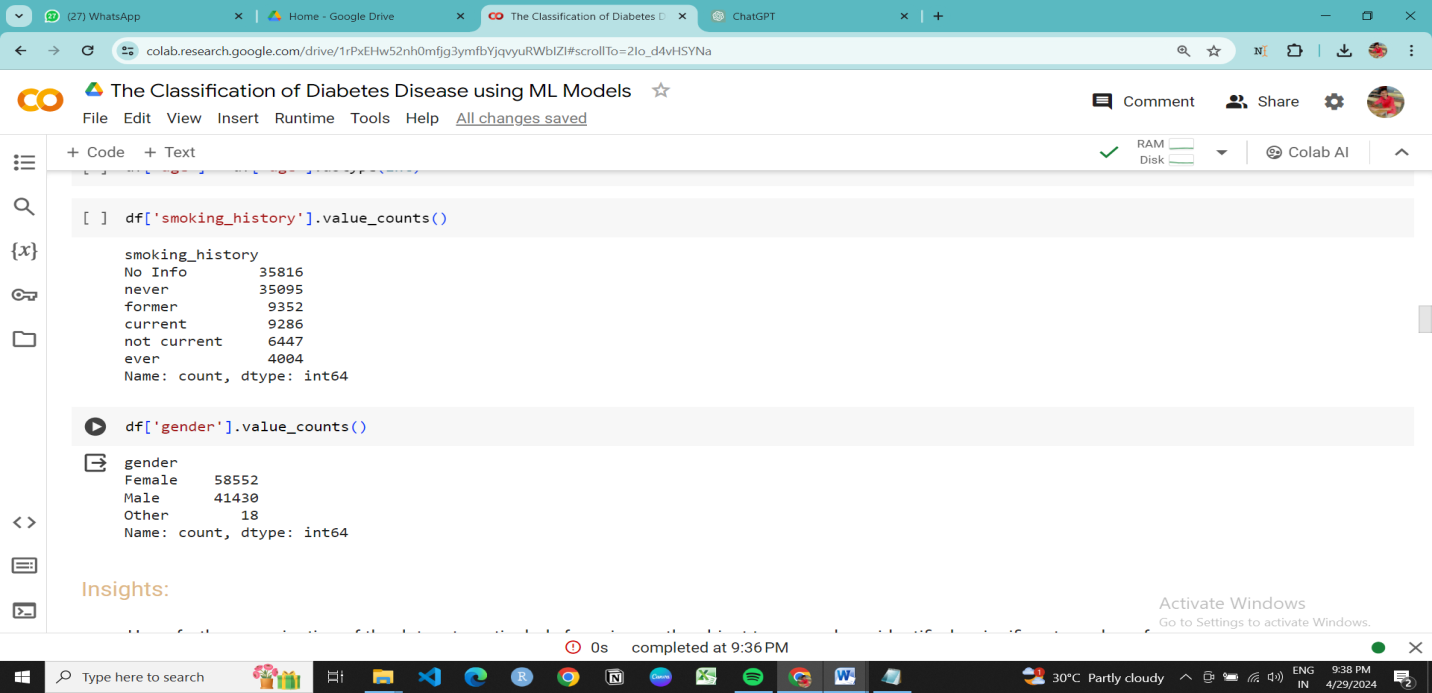
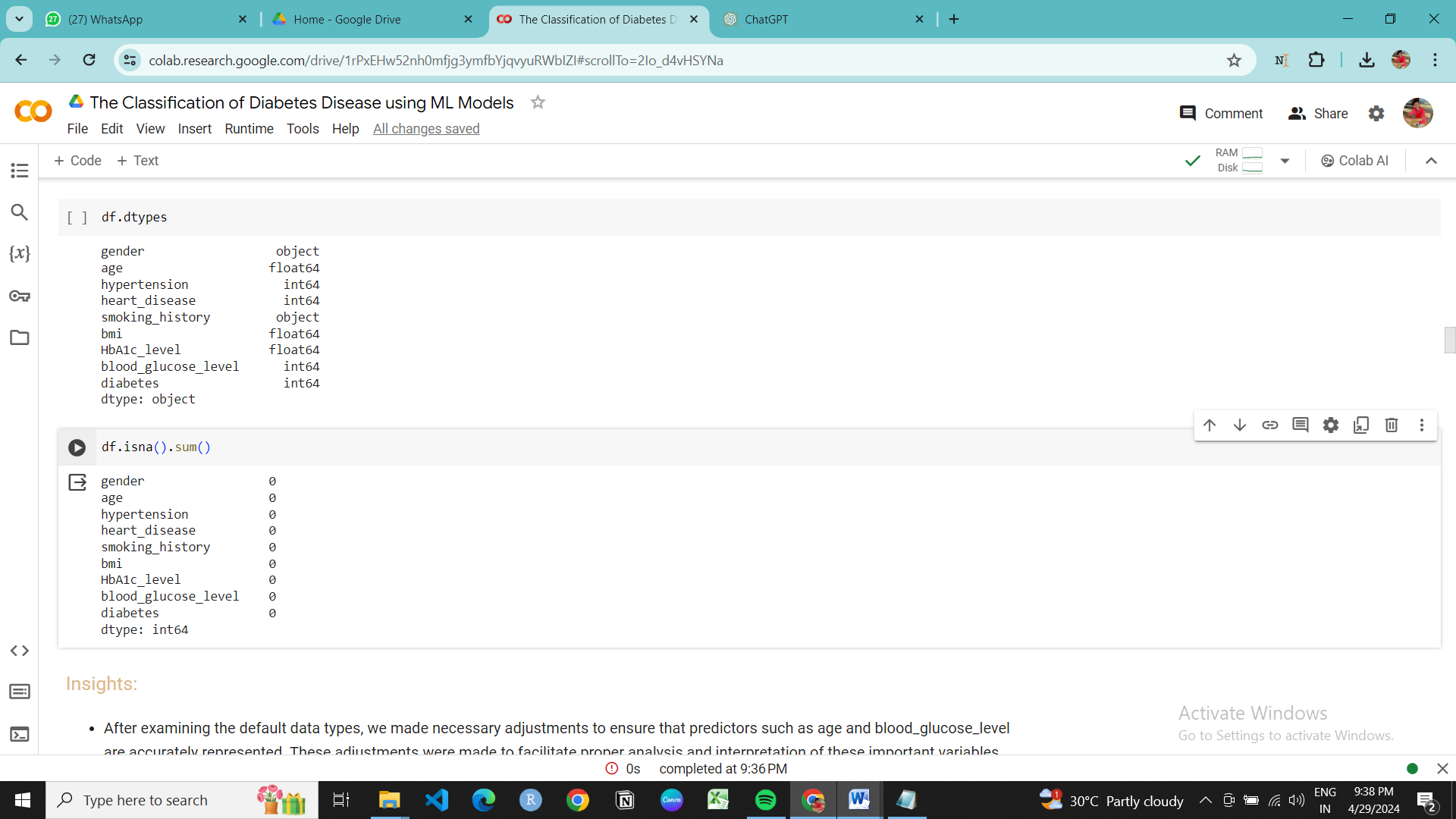
df\_results

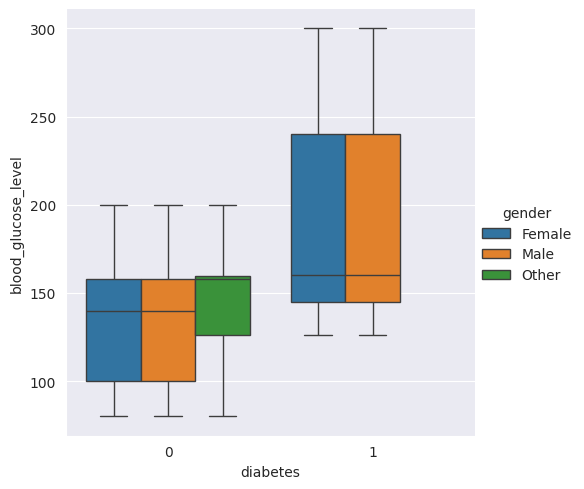
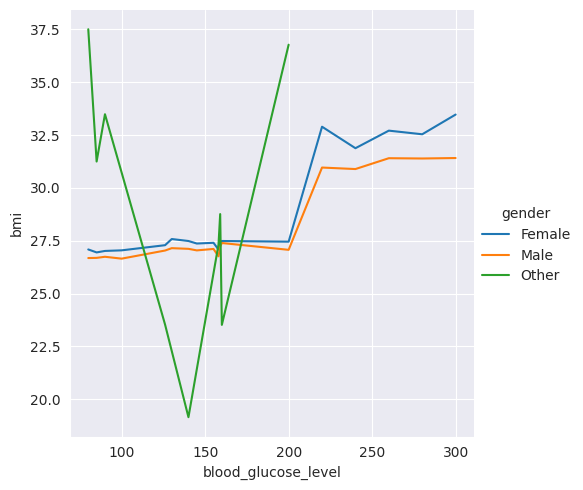
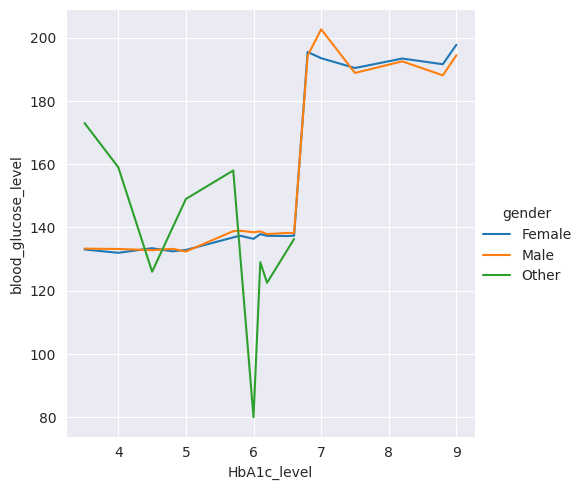
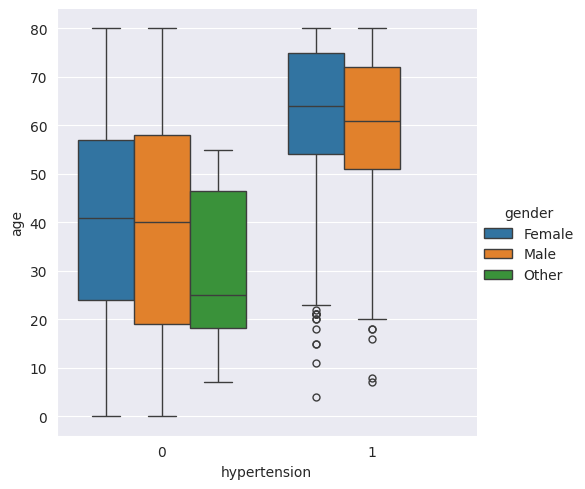
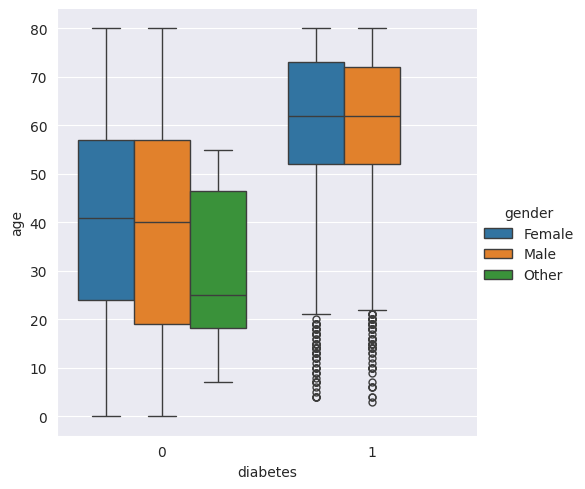
#Model Result Comparison

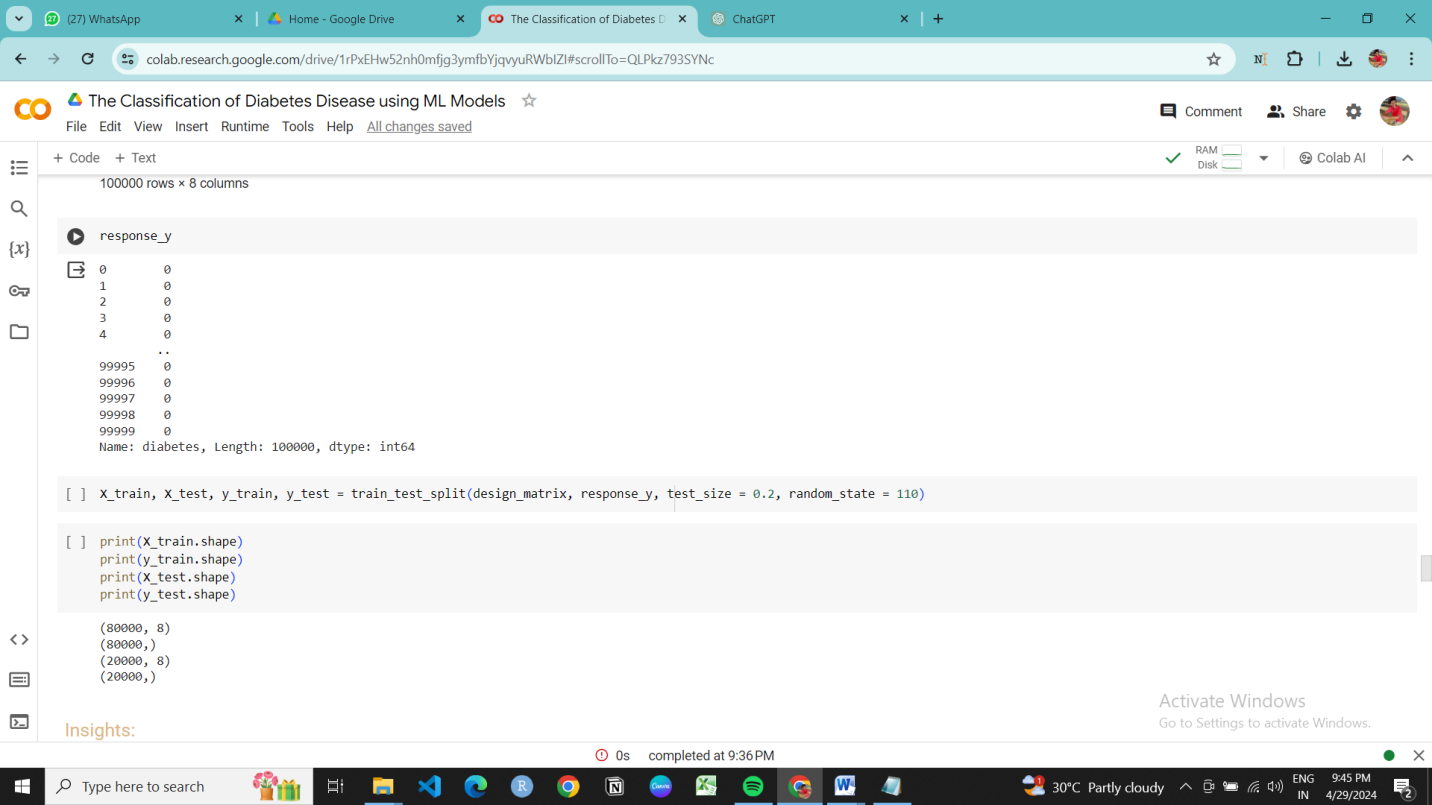
from matplotlib import pyplot as plt

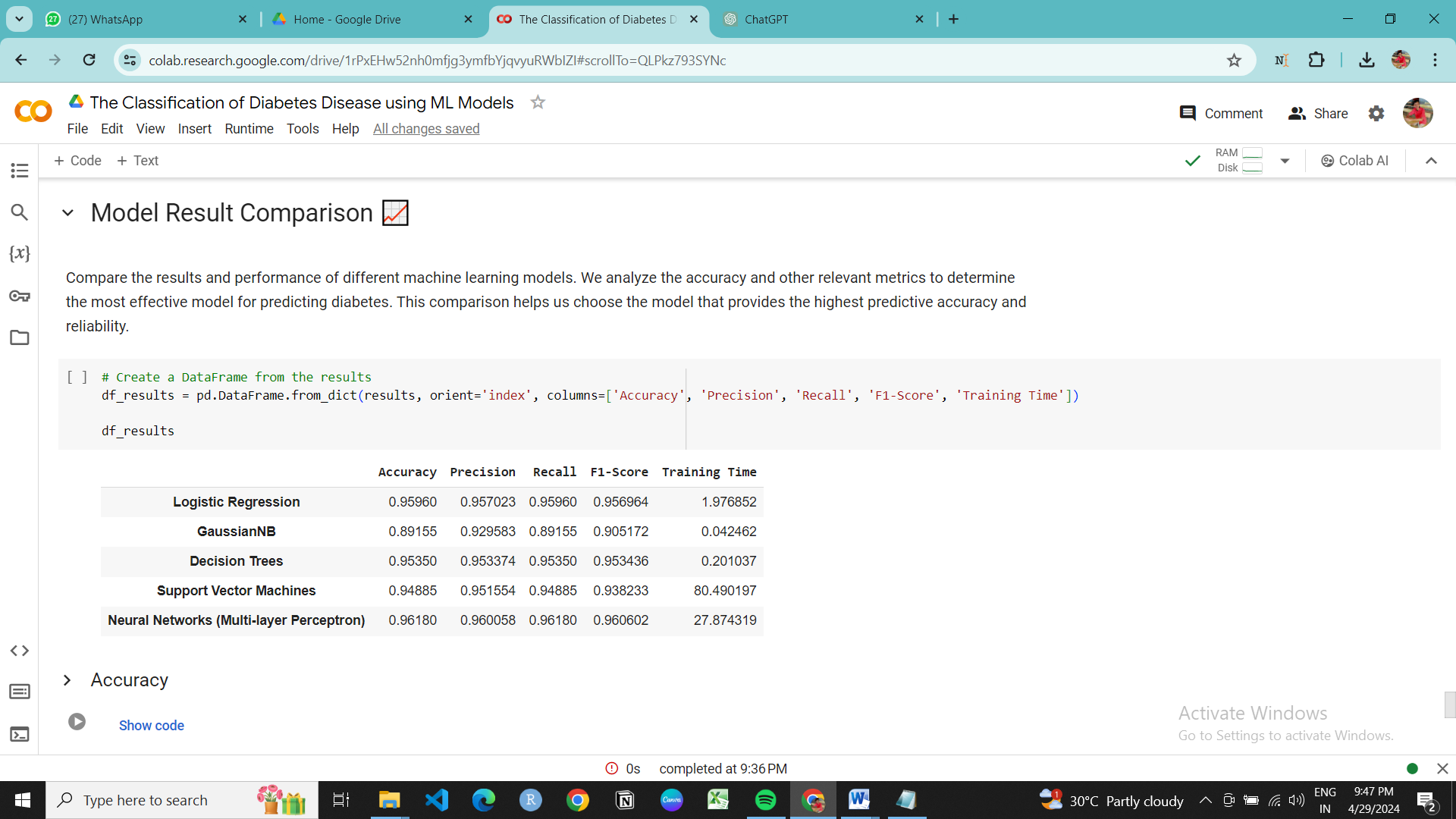
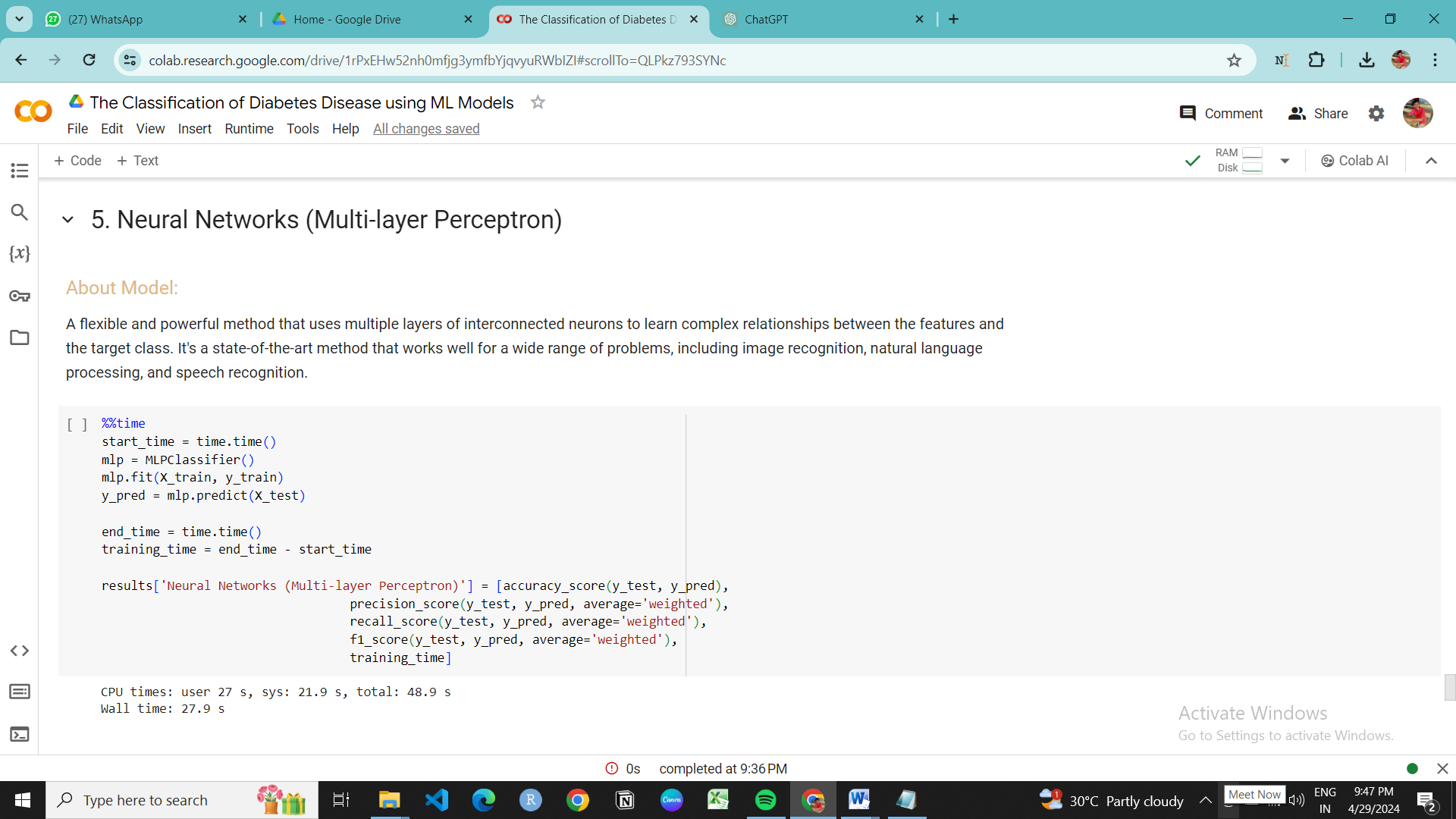
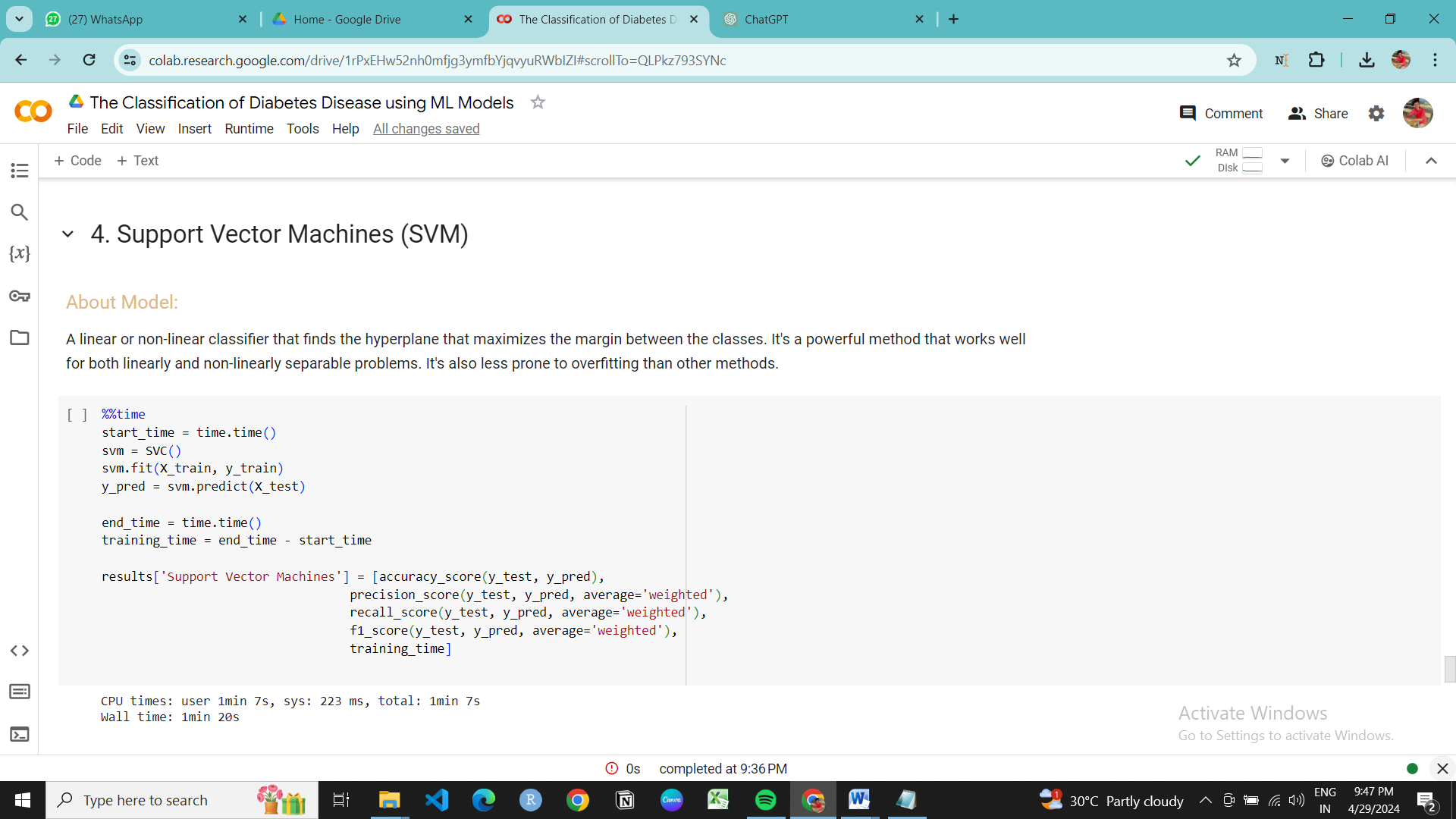
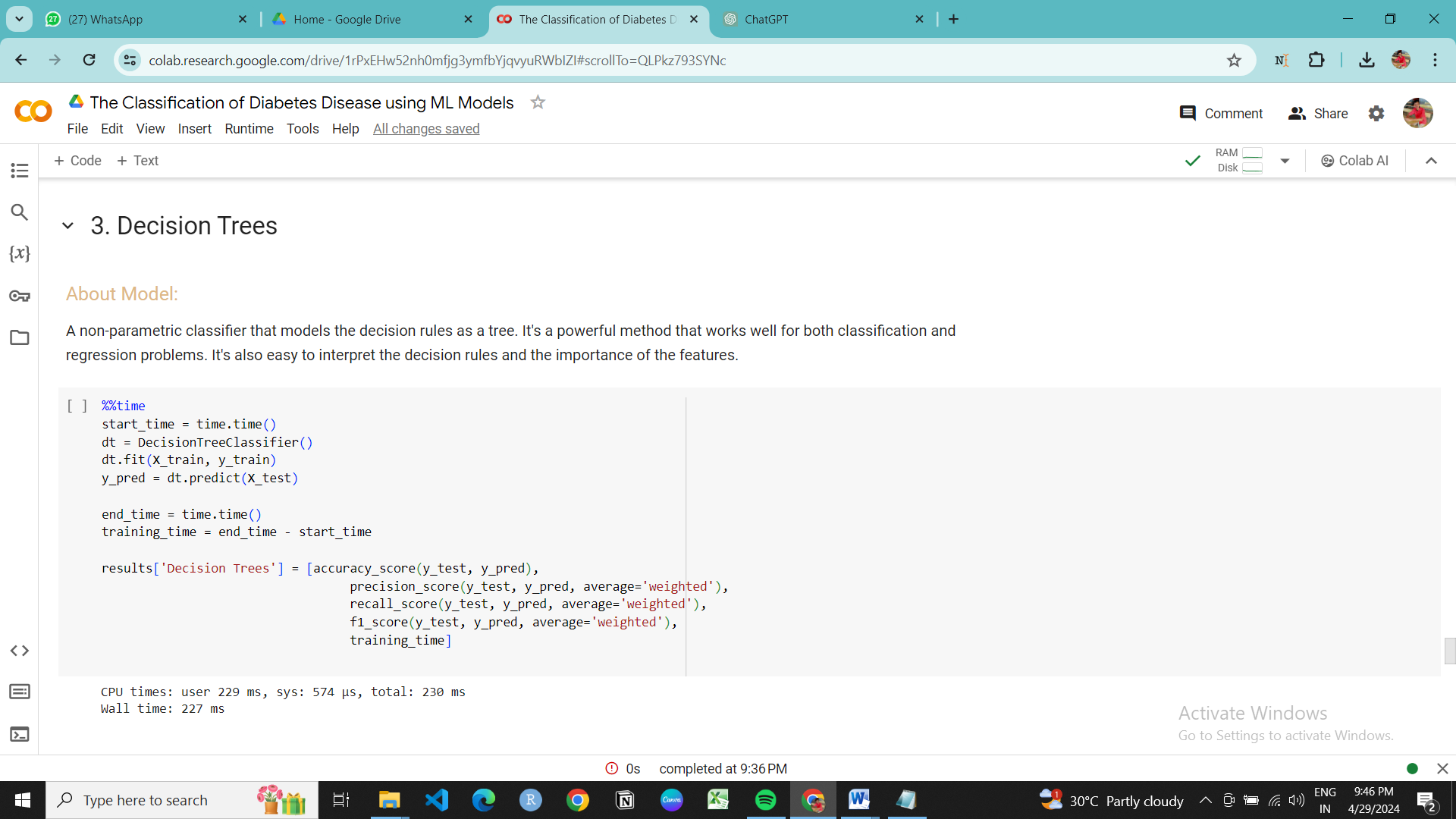
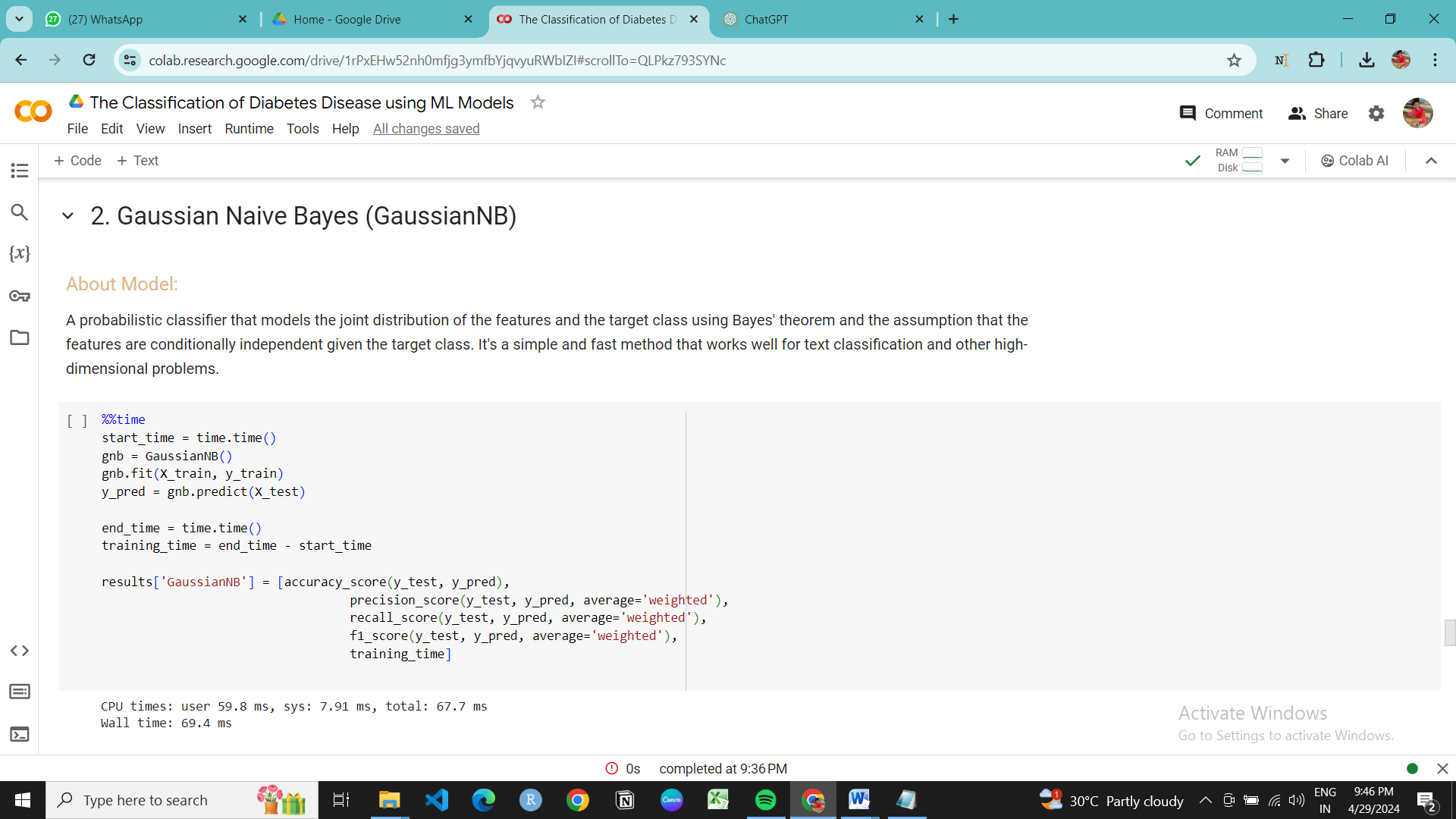
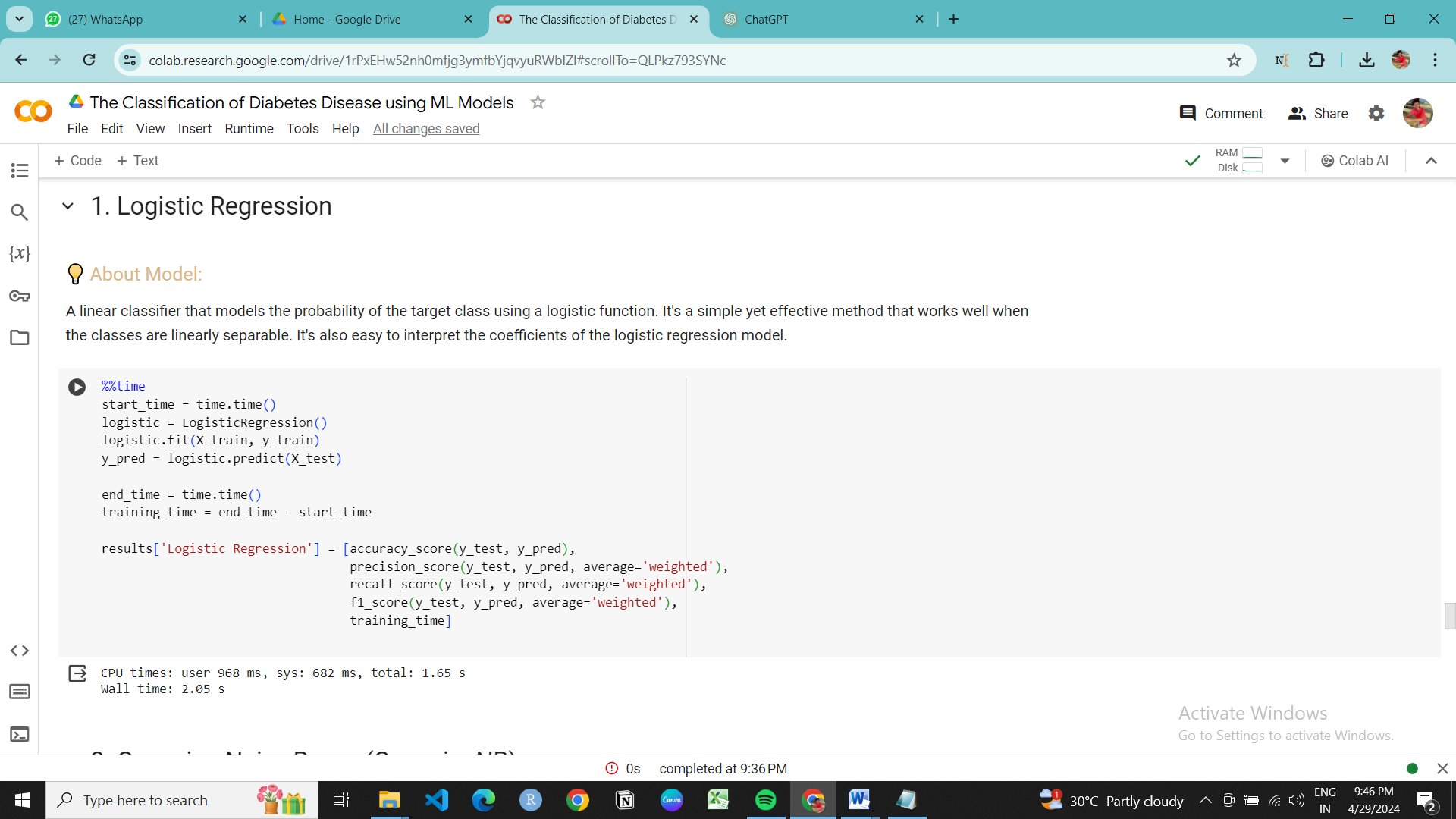
df\_results['Accuracy'].plot(kind='hist', bins=20, title='Accuracy')

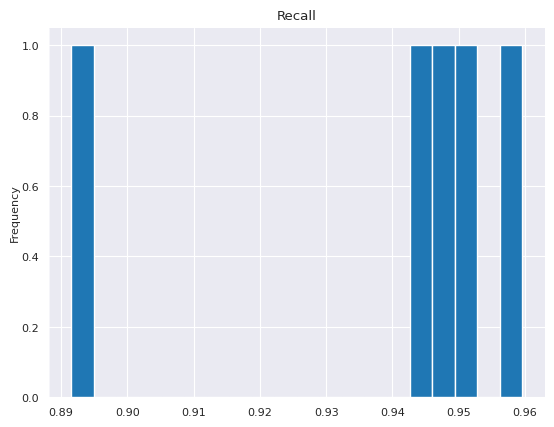
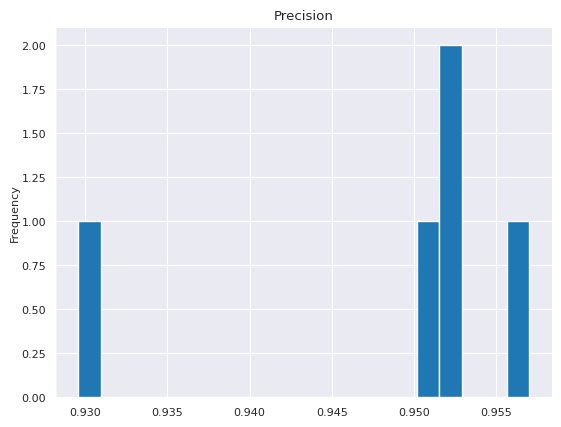
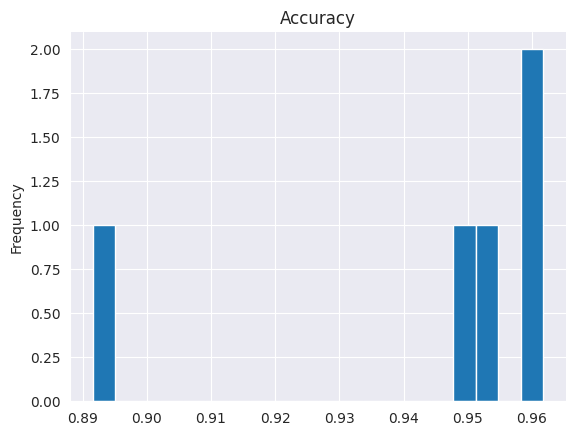
plt.gca().spines[['top', 'right',]].set\_visible(False)

**RESULTS AND DISCUSSION:**







****

**FUTURE DEVELOPMENT** :

1. **Real-time Data Integration**:
   * Develop systems capable of integrating real-time patient data from wearable devices, electronic health records (EHRs), and other sources.
   * Implement data streaming and processing techniques to handle continuous streams of data efficiently.
   * Utilize advanced machine learning models that can adapt and update in real-time as new data becomes available, allowing for timely and accurate predictions.
2. **Integration with Streamlit or Similar Platforms**:
   * Build user-friendly interfaces using platforms like Streamlit to visualize model outputs, input new patient data, and receive instant predictions.
   * Design interactive dashboards that provide clinicians with insights into patient risk factors, diagnostic probabilities, and recommended interventions.
   * Allow for easy customization of the application interface to accommodate different hospital workflows and user preferences.
3. **Development of Hospital Applications**:
   * Create specialized applications tailored to the needs of hospitals and healthcare providers.
   * Include features such as patient management tools, risk assessment modules, and decision support systems for diabetes diagnosis and treatment planning.
   * Ensure compliance with healthcare regulations and data privacy standards, such as HIPAA, to maintain patient confidentiality and security.
4. **Continuous Model Improvement**:
   * Implement mechanisms for continuous model monitoring and evaluation to ensure ongoing performance optimization.
   * Incorporate feedback from clinicians and healthcare professionals to refine and enhance the predictive accuracy and clinical relevance of the models.
   * Explore techniques such as transfer learning and ensemble methods to leverage knowledge from related domains and improve model generalization.
5. **Integration with Clinical Workflows**:
   * Integrate classification models seamlessly into existing clinical workflows to streamline decision-making processes.
   * Enable automatic generation of alerts or notifications for high-risk patients based on model predictions, facilitating proactive intervention and management.
   * Collaborate with healthcare IT specialists and clinical stakeholders to design interoperable systems that can exchange data with other hospital systems and electronic health records.

**CONCLUSION :**

In conclusion, the classification of diabetes disease using various machine learning algorithms such as logistic regression, Gaussian Naive Bayes, decision tree, support vector machine (SVM), and neural networks offers a multifaceted approach to identifying individuals at risk or already affected by diabetes.

* **Logistic Regression**: Provides interpretability and simplicity, making it suitable for understanding the relationship between predictors and diabetes likelihood.
* **Gaussian Naive Bayes**: Despite its assumption of feature independence, it can perform well, especially with small datasets, and offers computational efficiency.
* **Decision Tree**: Offers interpretability and can capture non-linear relationships, making it useful for understanding the decision-making process behind diabetes classification.
* **Support Vector Machine (SVM)**: Effective in high-dimensional spaces, capable of handling non-linear relationships, and provides robust performance with appropriate kernel functions.
* **Neural Networks**: Offers powerful capabilities for learning complex patterns in data, though at the cost of increased computational complexity and potential interpretability challenges.

The Result Dataframe provides scores, time taken, precision, recall, and F1 values for each tested model. The scores indicate the accuracy achieved on the test dataset. The time taken represents the duration of one iteration for each model, with more complex models typically requiring more time.The F1 score serves as a combined measure of precision and recall. A higher F1 score implies better performance in terms of both metrics, making it an optimal criterion for model evaluation.

**REFERENCES :**

1. Agarwal, P. (2015). Predictive modeling of type 2 diabetes mellitus using machine learning algorithms. *Biomedical Engineering and Computational Biology*, 6, 1-8.
2. Kavakiotis, I., Tsave, O., Salifoglou, A., Maglaveras, N., Vlahavas, I., & Chouvarda, I. (2017). Machine learning and data mining methods in diabetes research. *Computational and Structural Biotechnology Journal*, 15, 104-116.
3. Sarker, I. H., Khan, M. A. H., & Colman, A. (2018). Application of machine learning techniques in diabetes prediction: A systematic review. *Journal of Healthcare Engineering*, 2018, 9851067.
4. American Diabetes Association. (2019). Classification and diagnosis of diabetes: Standards of medical care in diabetes—2019. *Diabetes Care*, 42(Supplement 1), S13-S28.
5. Neural Networks to predict Diabetes | Kaggle