Project Report Form

Date	9th November 2023
Team ID	Team-592607
Project Name	Diabetes Prediction Using Machine Learning

Team Member: AS. Vikram Aditya
Team Member: PVS. Uday Kiran
Team Member: K. Rami Reddy
Team Leader: L. Mohith

1.INTRODUCTION

1.1 Project Overview

A "Diabetes Prediction Using Machine Learning" project leverages data and machine learning to achieve early diabetes detection, risk assessment, and prevention. Data collection involves gathering diverse sources, and data cleaning ensures data quality. Machine learning models, including logistic regression, decision trees, and neural networks, are employed for risk prediction. Model performance is evaluated through metrics and cross-validation.

Deployment integrates the model into healthcare systems or user-friendly interfaces. Privacy and security are essential. Continuous monitoring and updates maintain model accuracy. Interpretability provides understandable predictions. Clinical trials validate real-world effectiveness, and collaboration with healthcare professionals is crucial.

The project ensures scalability and generalization across diverse populations and addresses ethical concerns, promoting bias-free and transparent results. Success yields early interventions, improved patient outcomes, and reduces the diabetes burden.

1.2 Purpose

The primary purpose of "Diabetes Prediction Using Machine Learning" is to leverage data and advanced computational techniques to achieve the following objectives:

1. **Early Detection and Diagnosis**: Machine learning models can help in the early detection and diagnosis of diabetes. By analyzing relevant data, such as patient medical records, lifestyle information, and clinical parameters, these models can identify individuals who may be at risk of developing diabetes, even before they exhibit clinical symptoms. Early diagnosis allows for timely intervention and treatment.
2. **Risk Assessment**: Machine learning models can assess an individual's risk of developing diabetes. By considering various factors such as age, genetics, body mass index (BMI), diet, physical activity, and family history, the models can provide a quantified risk score. This information enables healthcare professionals to identify high-risk individuals and offer personalized preventive strategies.
3. **Preventive Interventions**: Predictive models can recommend preventive interventions and lifestyle modifications for individuals at risk. These recommendations can include dietary changes, exercise regimens, and regular health check-ups to reduce the risk of diabetes or delay its onset.

2. LITERATURE SURVEY

2.1 Existing Problem

Presently the world is facing “Diabetes” to resolve this problem many people are trying to find the cure and some of them are trying to predict whether the patient is going to be affected by diabetes or not, so that they can prevent it by following a proper diet, exercises etc. In most of the surveys everyone proposed some machine algorithms to predict whether the patient is going affected by the diabetes or not at early stages, K.VijiyaKumar et al. [11] proposed random Forest algorithm for the Prediction of diabetes develop a system which can perform early prediction of diabetes for a patient with a higher accuracy by using Random Forest algorithm in machine learning technique. The proposed model gives the best results for diabetic prediction and the result showed that the prediction system is capable of predicting the diabetes disease effectively, efficiently and most importantly, instantly.Yasodhaet al.[1] uses the classification on diverse types of datasets that can be accomplished to decide if a person is diabetic or not. The diabetic patient’s data set is established by gathering data from the hospital warehouse which contains two hundred instances with nine attributes.

These instances of this dataset are referring to two groups i.e. blood tests and urine tests. In this study the implementation can be done by using WEKA to classify the data and the data is assessed by means of a 10-fold cross validation approach, as it performs very well on small datasets, and the outcomes are compared. The naïve Bayes, J48, REP Tree and Random Tree are used. It was concluded that J48 works best showing an accuracy of 60.2% among others. Well in my case Random Forest Classifier has predicted best among the Logistic Regression and XGBClassifier.

2.2 References

1. Diabetes prediction using machine learning techniques
2. Diabetes prediction using machine learning

2.3 Problem Statement Definition

"Diabetes is a widespread chronic health condition with a significant impact on individuals' well-being and healthcare systems. Early diagnosis and timely intervention are crucial for improving patient outcomes and reducing the economic burden associated with diabetes. The objective of this project is to develop an accurate and scalable machine learning model for diabetes prediction that can identify individuals at risk of diabetes or its subtypes.

This project aims to achieve the following goals:

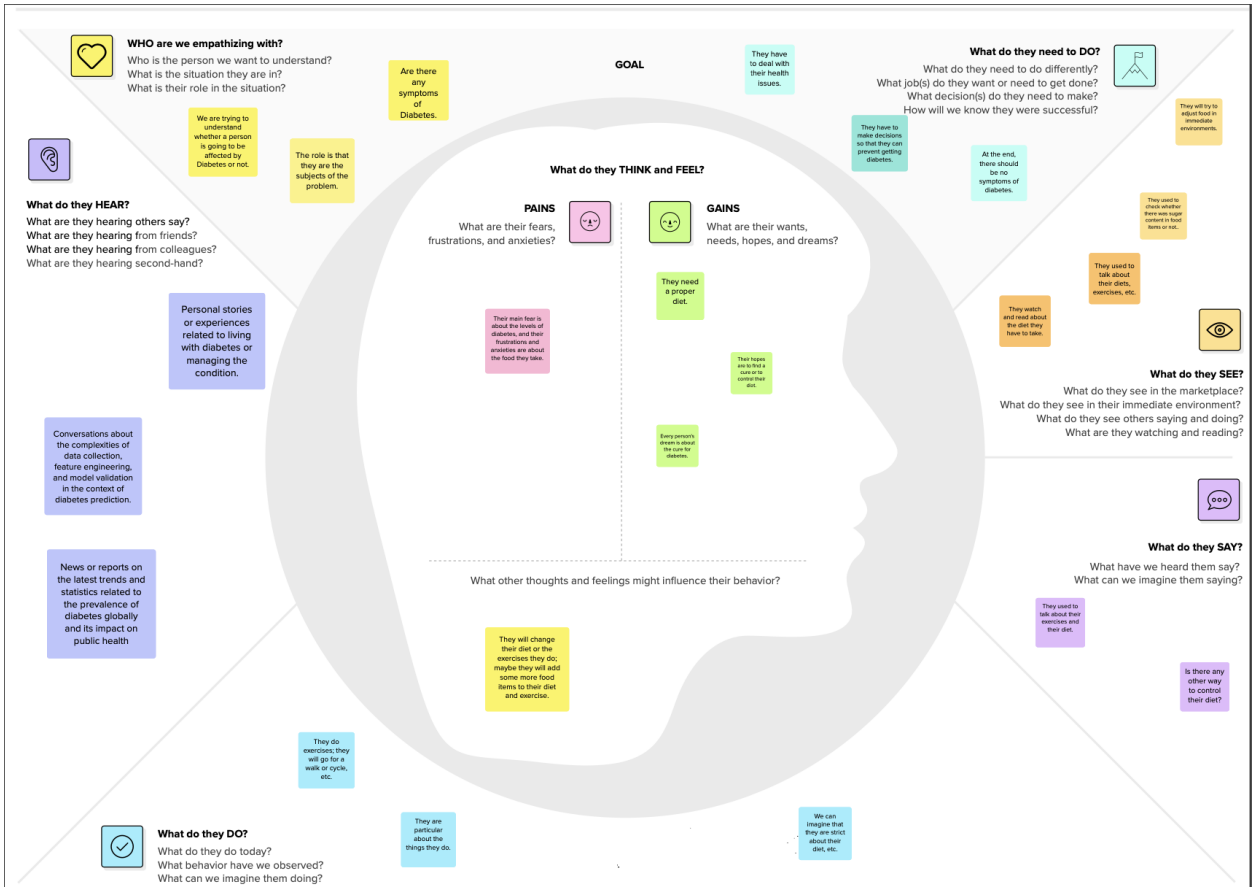
1. Create a predictive model that accurately assesses an individual's risk of developing diabetes.
2. Enable early diagnosis of diabetes or prediabetes, allowing for timely intervention and personalized treatment plans.
3. Develop a user-friendly interface for healthcare professionals and individuals to input relevant data and receive risk assessments.
4. Ensure the model's generalizability across diverse populations, considering factors such as age, ethnicity, and geographic location.
5. Enhance the transparency and interpretability of the model, providing insights into the key risk factors contributing to the predictions.
6. Implement mechanisms for continuous monitoring and updates to keep the model current with evolving medical knowledge and patient data.

3 IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas

An empathy map is a simple, easy-to-digest visual that captures knowledge about a user’s behaviors and attitudes. It is a useful tool to help teams better

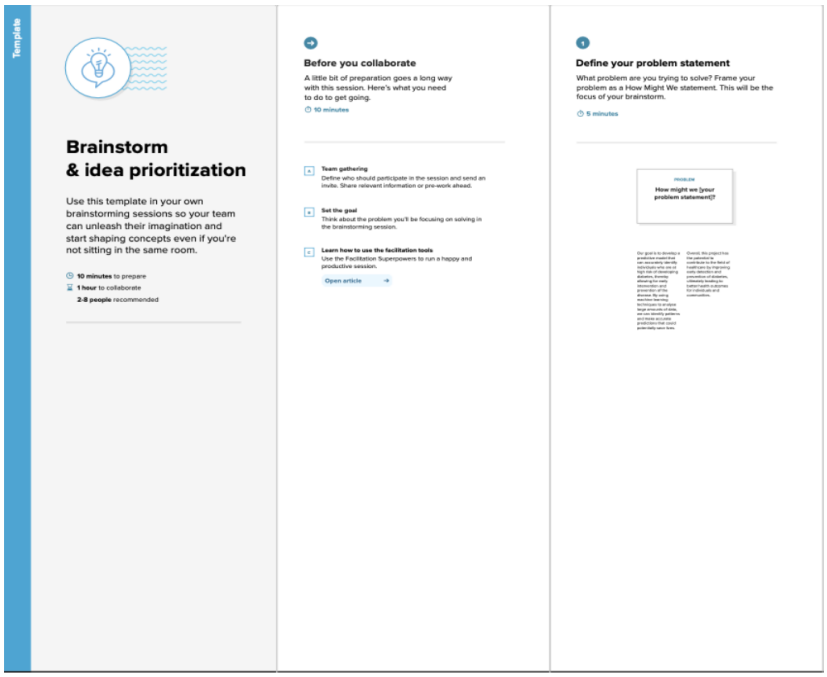
understand their users. Creating an effective solution requires understanding the true problem and the person who is experiencing it. The exercise of creating the map helps participants consider things from the user's perspective along with his or her goals and challenges.



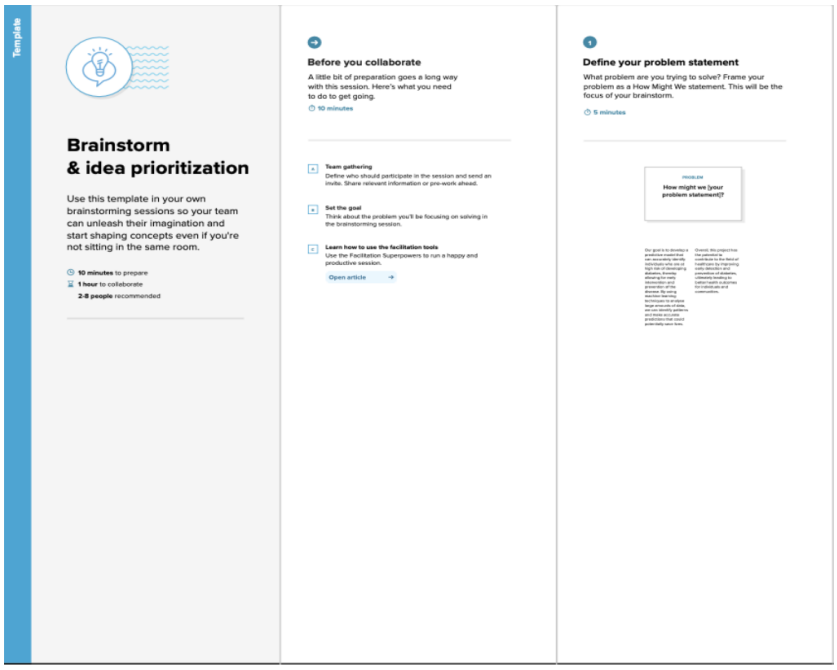
3.2 Ideation & Brainstorming

Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem solving. Prioritizing volume over value, out-of-the-box ideas are welcome and built upon, and all participants are encouraged to collaborate, helping each other develop a rich amount of creative solutions. Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

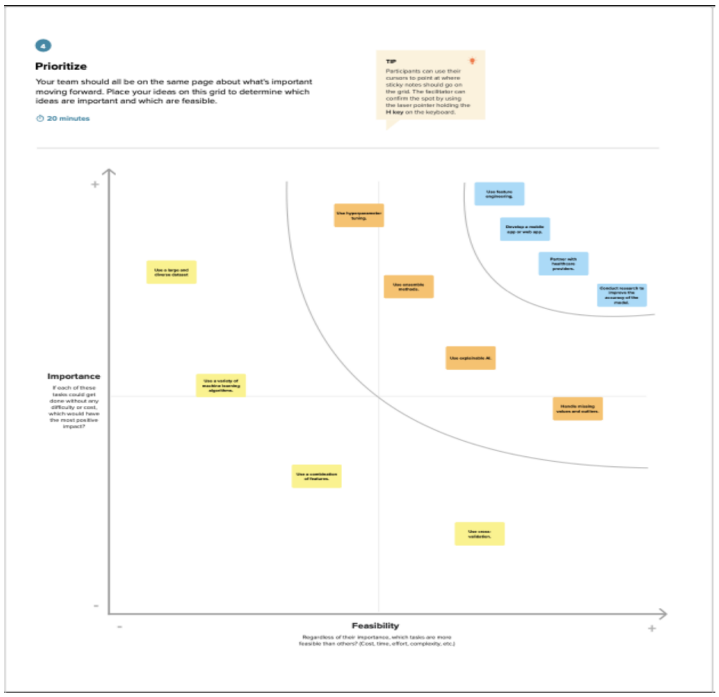
Step:1 Team Gathering, Collaboration and Select the Problem Statement



Step-2: Brainstorm, Idea Listing and Grouping



Step-3: Idea Prioritization



4. REQUIREMENT ANALYSIS

4.1 Functional Requirements

Functional Requirements:

Data Collection and Integration:

- The system must be able to collect and integrate data from various sources, including electronic health records, wearable devices, and patient surveys.

Feature Selection and Preprocessing:

- It should perform feature selection and data preprocessing to handle missing values, outliers, and standardize data for modeling.

Machine Learning Models:

- The system should implement various machine learning models, such as logistic regression, decision trees, random forests, and neural networks, to predict diabetes risk.

Model Training and Testing:

- It must be able to split the dataset into training and testing sets, train the models on the training data, and evaluate their performance on the testing data.

Model Deployment:

- The system should allow for the deployment of trained models, making them accessible to healthcare professionals and patients.

User Interface:

- A user-friendly interface should be developed for inputting relevant data and receiving risk assessments, along with explanations of predictions.

Continuous Monitoring:

- Implement mechanisms for continuous monitoring of patient data and provide real-time alerts and recommendations for managing diabetes risk.

Privacy and Security:

- Ensure that the system complies with data privacy and security regulations, including encryption and access control.

Interpretability:

- Provide explanations for model predictions to make them understandable and actionable for healthcare professionals and patients.

Scalability:

- Ensure that the system is scalable to handle large volumes of data and can accommodate a growing user base.

4.2 Non-Functional Requirements

Accuracy:

- The system should achieve a high level of prediction accuracy to ensure reliable risk assessments.

Performance:

- The response time for predicting diabetes risk should be fast and efficient to provide real-time recommendations.

Scalability:

- The system should be able to scale and handle a large volume of user requests and data.

Reliability:

- The system must be reliable and available for users, with minimal downtime or disruptions.

Security:

- Data security and privacy should be a top priority to protect sensitive healthcare information.

Interpretability:

- Ensure that the models are interpretable, providing clear explanations for predictions.

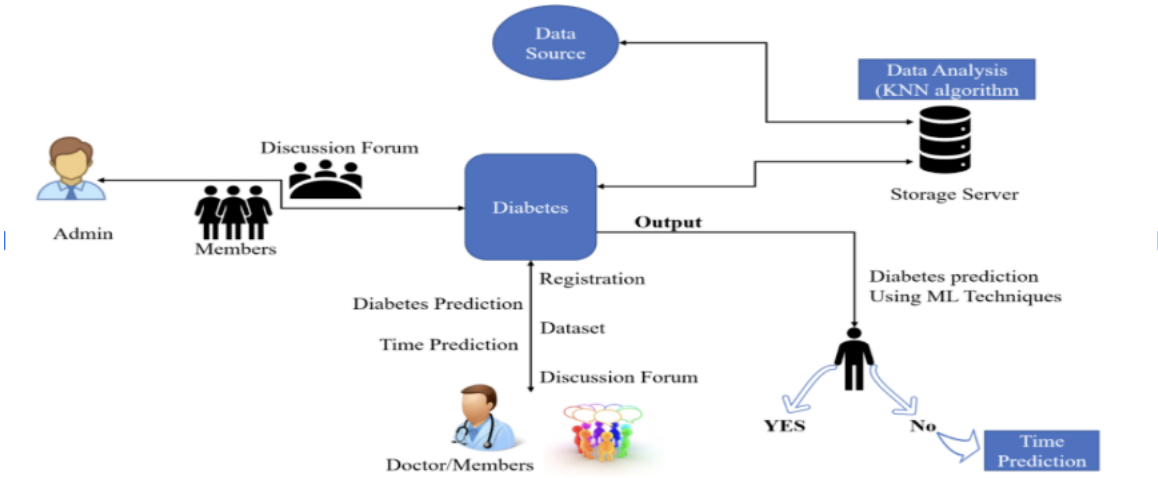
5. PROJECT DESIGN

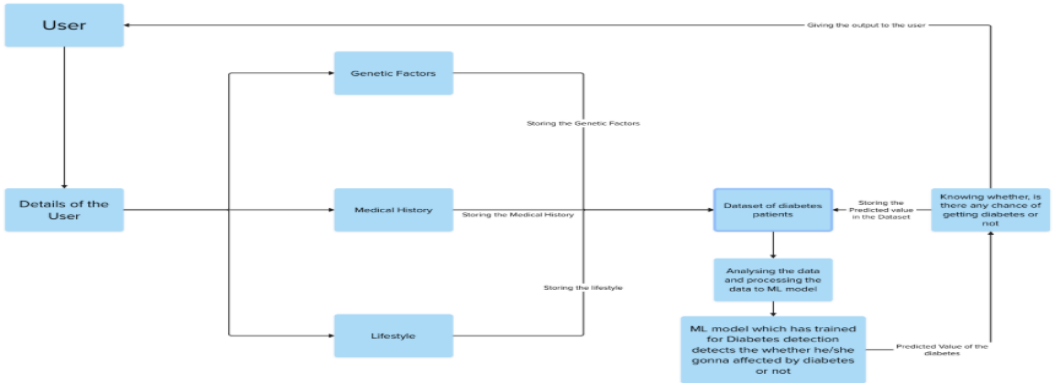
5.1 Data Flow Diagram & User Stories

Data Flow Diagrams:

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is store

Example:





User Stories

Use the below template to list all the user stories for the product.

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Clinics	Assessment	USN-1	Clinicians use diabetes prediction models to identify patients at high risk of developing diabetes, and to develop personalized prevention and management plans for them.	The system should accurately generate personalized risk assessments and recommendations for patients.	High	Sprint-1
Patients	Prevention	USN-2	Patients can use diabetes prediction models to assess their own risk of developing diabetes, and to make lifestyle changes and get screened for diabetes regularly.	The system should provide personalized information and advice to help patients prevent or manage diabetes.	High	Sprint-1
Researchers	Discovery	USN-3	Researchers use diabetes prediction models to develop new insights into the disease, identify new biomarkers for early detection, and develop new treatments and prevention strategies.	The system should produce reproducible and generalizable results that can be used to advance the understanding and prevention of diabetes.	High	Sprint-1
Public and Health Officials	Intervention	USN-4	Public health officials use diabetes prediction models to develop strategies for preventing diabetes at a population level, identify communities at high risk, and develop targeted prevention programs.	The system should be scalable and effective in identifying and intervening with populations at high risk of diabetes.	Medium	Sprint-2
Insurance companies	Risk	USN-5	Insurance companies use diabetes prediction models to set premiums and identify individuals who are eligible for special	The system should fairly and accurately assess risk to help set	Medium	Sprint-2

			programs or services.	premiums and offer targeted programs and services.		
Employers	Wellness	USN-6	Employers use diabetes prediction models to develop workplace wellness programs and identify employees who may be at risk of developing diabetes-related complications.	The system should be engaging and effective in helping employers develop and implement workplace wellness programs.	Low	Sprint-2
Pharmaceutical Companies	Treatment	USN-7	Pharmaceutical companies use diabetes prediction models to develop new drugs and treatments for diabetes.	The system should effectively identify new targets for drug development and help design and conduct successful clinical trials.	Low	Sprint-3

5.2 Solution Architecture

Data ingestion and processing: This component is responsible for collecting and processing data from a variety of sources, such as electronic health records (EHRs), wearable devices, and patient surveys. The data may need to be cleaned, transformed, and enriched before it can be used for prediction.

Feature engineering: This component is responsible for creating features from the raw data that are informative for predicting diabetes risk. For example, features could be created for age, sex, family history, lifestyle habits, and clinical measurements.

Machine learning model: This component is responsible for training and deploying a machine learning model to predict diabetes risk. A variety of machine learning algorithms can be used for diabetes prediction, such as logistic regression, decision trees, support vector machines, and random forests.

Prediction service: This component is responsible for exposing the machine learning model to users and generating predictions of diabetes risk. The prediction service can be implemented as a web service, mobile app, or other type of application.

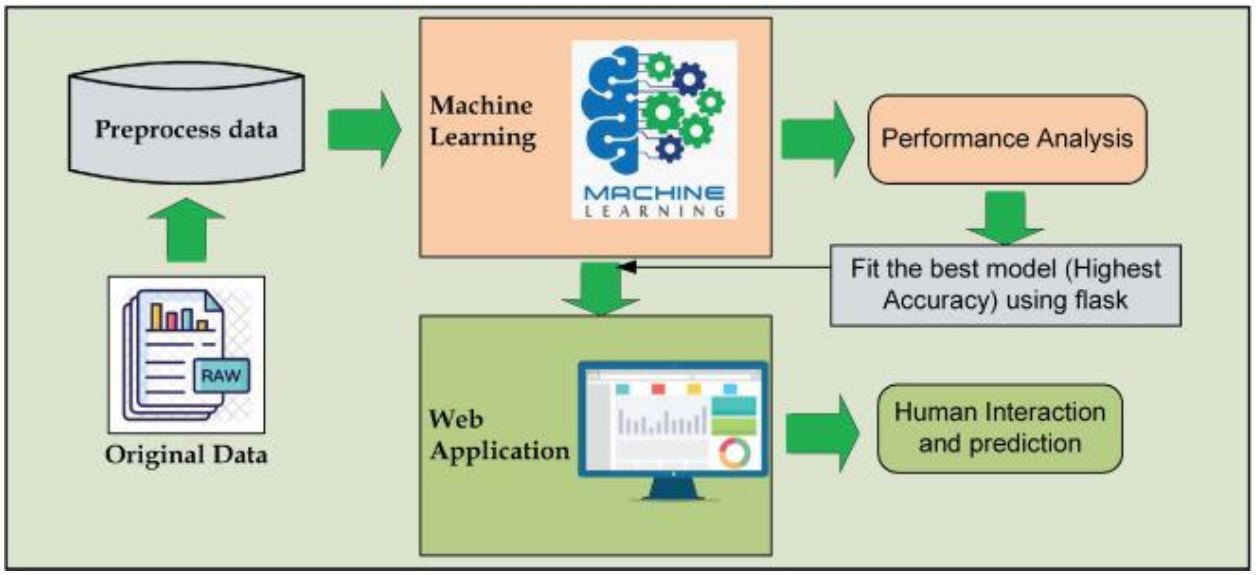
Model monitoring and retraining: This component is responsible for monitoring the performance of the machine learning model and retraining it as needed. This is important to ensure that the model remains accurate over time as new data becomes available.

Scalability: Scalability is the ability of a system to handle increasing amounts of data and users without sacrificing performance. Diabetes prediction systems need to be scalable because they may need to process large volumes of data from a large number of users.

Reliability: Reliability is the ability of a system to function consistently and correctly over a period of time. Diabetes prediction systems need to be reliable because they are used to make important decisions about people's health.

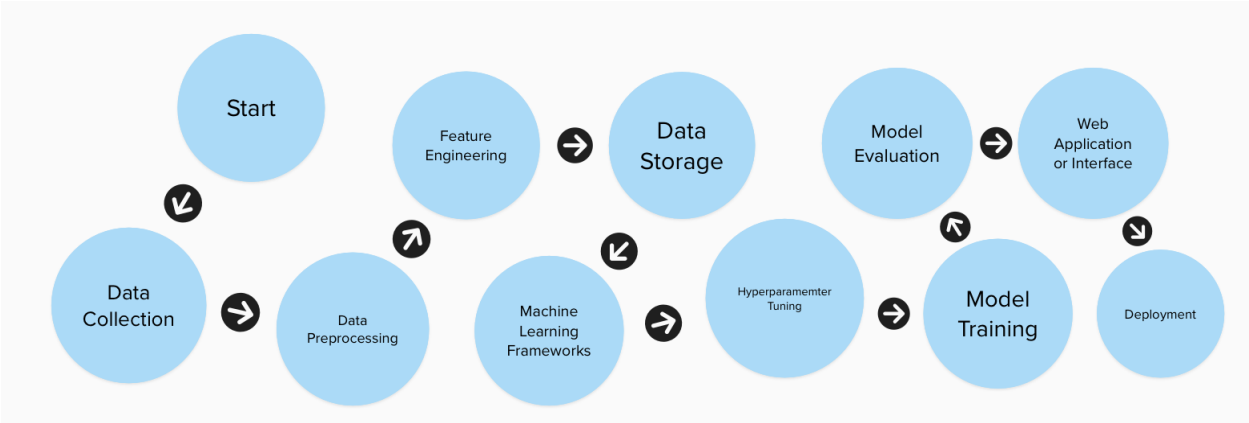
Security: Security is the ability of a system to protect data from unauthorized access, use, disclosure, disruption, modification, or destruction. Diabetes prediction systems need to be secure because they handle sensitive personal health information.

Solution Architecture Diagram:



6. PROJECT PLANNING & SCHEDULING

6.1 Technical Architecture



6.2 Sprinting & Planning Estimation

Use the below template to create product backlog and sprint schedule

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Assessment	USN-1	Clinicians use diabetes prediction models to identify patients at high risk of developing diabetes, and to develop personalized prevention and management plans for them.	10	High	PVS. Uday Kiran
Sprint-1	Prevention	USN-2	Patients can use diabetes prediction models to assess their own risk of developing diabetes, and to make lifestyle changes and get screened for diabetes regularly.	5	High	L. Mohith
Sprint-1	Discovery	USN-3	Researchers use diabetes prediction models to develop	5	High	K. Rami Reddy

			new insights into the disease, identify new biomarkers for early detection, and develop new treatments and prevention strategies.			
Sprint-2	Intervention	USN-4	Public health officials use diabetes prediction models to develop strategies for preventing diabetes at a population level, identify communities at high risk, and develop targeted prevention programs.	8	Medium	PVS. Uday Kiran
Sprint-2	Risk	USN-5	Insurance companies use diabetes prediction models to set premiums and identify individuals who are eligible for special programs or services.	6	Medium	K. Rami Reddy
Sprint-2	Wellness	USN-6	Employers use diabetes prediction models to develop workplace wellness programs and identify employees who may be at risk of developing diabetes-related complications.	6	Low	AS. Vikram Aditya
Sprint-3	Treatment	USN-7	Pharmaceutical companies use diabetes prediction models to develop new drugs and treatments for diabetes.	10	Low	AS. Vikram Aditya, L.Mohith

6.3 Sprint Delivery Schedule

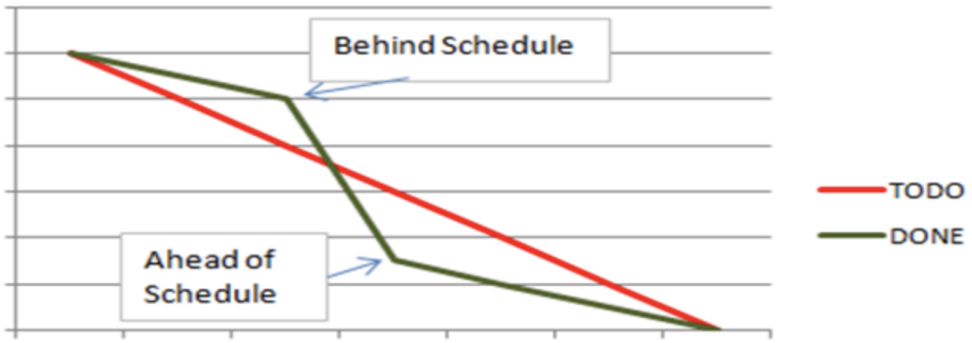
Project Tracker, Velocity & Burndown Chart: (4 Marks)

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	8 Days	12 Oct 2023	19 Oct 2023	15	19 Oct 2023
Sprint-2	20	8 Days	20 Oct 2023	25 Oct 2023	25	25 Oct 2023
Sprint-3	10	4Days	26 Oct 2023	27 Oct 2023	10	27 Oct 2023

Velocity:
We have a 20-days sprint duration, and the velocity of the team is 15(points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day)

$$AV = \frac{Sprint\ Duration}{Velocity} = \frac{20}{15} = 1.33$$

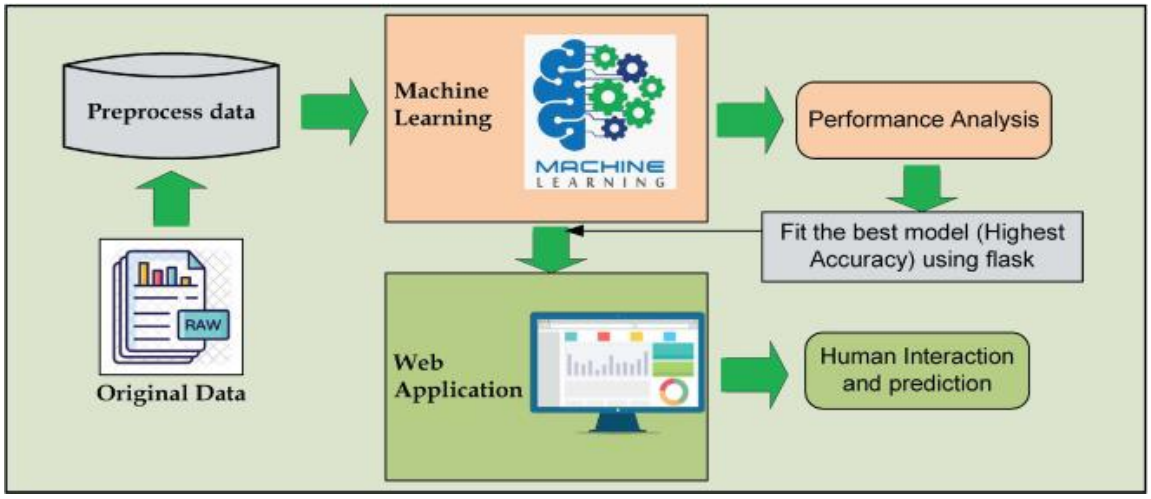
Burndown Chart:



7. CODING & SOLUTIONING

Diabetes Prediction Using Machine Learning

In this project, we aim to use machine learning algorithms to predict the onset of diabetes in individuals based on their health records and other relevant factors such as age, BMI, family history, and lifestyle habits. The dataset used in this project will include information on various clinical parameters such as blood pressure, BMI, Heart diseases and cholesterol levels.
Our goal is to develop a predictive model that can accurately identify individuals who are at high risk of developing diabetes, thereby allowing for early intervention and prevention of the disease. By using machine learning techniques to analyze large amounts of data, we can identify patterns and make accurate predictions that could potentially save lives.
Overall, this project has the potential to contribute to the field of healthcare by improving early detection and prevention of diabetes, ultimately leading to better health outcomes for individuals and communities.



Project Flow:

- User interacts with the UI to enter the input.
- Entered input is analyzed by the model which is integrated.
- Once model analyzes the input the prediction is showcased on the UI

To accomplish this, we have to complete all the activities listed below,

- Define Problem / Problem Understanding
  - Specify the business problem
  - Business requirements
  - Literature Survey
  - Social or Business Impact.

- Data Collection & Preparation
  - Collect the dataset
  - Data Preparation

- Exploratory Data Analysis
  - Descriptive statistical
  - Visual Analysis

- Model Building
  - Training the model in multiple algorithms
  - Testing the model

- Performance Testing
  - Testing model with multiple evaluation metrics

- Model Deployment
  - Save the best model

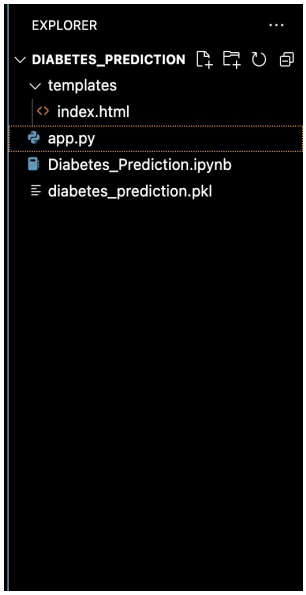
- Integrate with Web Framework

Prior Knowledge:

You must have prior knowledge of following topics to complete this project.

● Random Forest
● Logistic Regression
● XGBClassifier

Project Structure:



Create the Project folder which contains files as shown below

● We are building a flask application which needs HTML pages stored in the templates folder and a python script app.py for scripting.
● diabetes_prediction.pkl is our saved model. Further we will use this model for flask integration.
● Training folder contains a model training file.

Milestone 1: Define Problem / Problem Understanding

Activity 1: Specify the business problem

The business problem addressed in this project is the early detection and prediction of diabetes using machine learning algorithms. The goal is to develop a predictive model that can accurately identify individuals at high risk of developing diabetes based on their health records and other relevant factors. Early detection and management of diabetes can improve healthcare outcomes, reduce costs, and benefit healthcare providers and insurance companies. Therefore, developing an accurate and reliable predictive model for diabetes detection can have a significant impact on healthcare outcomes and costs.

Activity 2: Business requirements

Business requirements are the specific needs and expectations of the business stakeholders regarding the desired outcome of the project. In the case of the diabetes prediction project, the following are the key business requirements:

- Accurate prediction: The predictive model should be accurate in identifying individuals who are at high risk of developing diabetes based on their health records and other relevant factors.
- Efficiency: The model should be efficient and fast in analyzing large amounts of data to provide timely predictions.
- Scalability: The model should be scalable to handle large datasets and accommodate future growth in data volume.
- Flexibility: The model should be flexible and adaptable to accommodate changes in data sources or input parameters.
- User-friendliness: The model should be user-friendly, easy to use, and understand by healthcare providers and insurance companies.
- Integration: The model should be easily integrated with existing healthcare systems and processes.
- Security: The model should be secure and protect patient data privacy. ● Compliance: The model should comply with relevant healthcare regulations and standards.

Activity 3: Literature Survey

A literature survey is an essential step in any diabetes prediction using machine learning:

- Gauri D. Kalyankar, Shivananda R. Poojara and Nagaraj V. Dharwadkar," Predictive Analysis of Diabetic Patient Data Using Machine Learning and Hadoop", International Conference On I-SMAC, 978-1-5090-3243-3, 2017.

- Ayush Anand and Divya Shakti," Prediction of Diabetes Based on Personal Lifestyle Indicators", 1st International Conference on Next Generation Computing Technologies, 978-1-4673-6809-4, September 2015.

- B. Nithya and Dr. V. Ilango," Predictive Analytics in Health Care Using Machine Learning Tools and Techniques", International Conference on Intelligent Computing and Control Systems, 978-1-5386-2745-7, 2017.

Activity 4: Social or Business Impact

Accurate diabetes prediction using machine learning can have a significant social and business impact. It can help identify individuals at high risk of developing diabetes, leading to earlier intervention and prevention efforts. From a business perspective, accurate prediction can help healthcare providers and insurers manage healthcare costs and resources better. Additionally, it can lead to more personalized healthcare, improving patient outcomes and adherence to treatment plans. In conclusion, accurate diabetes prediction using machine learning can improve patient outcomes, reduce healthcare costs, and lead to more personalized healthcare.

Milestone 2: Data Collection & Preparation

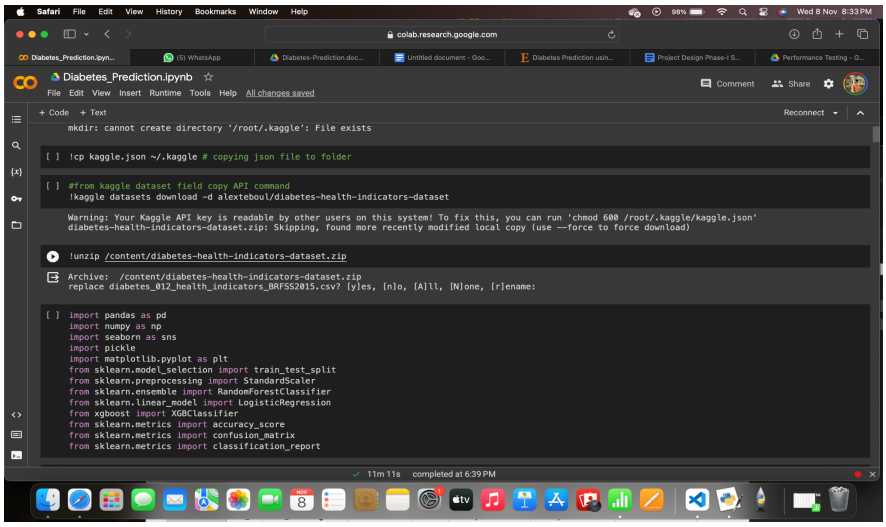
ML depends heavily on data. It is the most crucial aspect that makes algorithm training possible. So, this section allows you to download the required dataset.

Activity 1: Collect the dataset

There are many popular open sources for collecting the data. Eg: kaggle.com, UCI repository, etc.In this project we have used .csv data. This data is downloaded from kaggle.com. Please refer to the link given below to download the dataset.
Link: Diabetes Health Indicators Dataset | Kaggle
As the dataset is downloaded. Let us read and understand the data properly with the help of some visualization techniques and some analyzing techniques.
Note: There are a number of techniques for understanding the data. But here we have used some of it. In an additional way, you can use multiple techniques.

Activity 1.1: Importing the libraries

Import the necessary libraries as shown in the image.



Activity 1.2: Read the Dataset

Our dataset format might be in .csv, excel files, .txt, .json, etc. We can read the dataset with the help of pandas. In pandas we have a function called read_csv() to read the dataset. As a parameter we have to give the directory of the csv file.

```
[ ] df = pd.read_csv('/content/diabetes_012_health_indicators_BRFSS2015.csv')
    df.head()
```

Activity 2: Data Preparation

As we have understood how the data is, let's pre-process the collected data.
The download data set is not suitable for training the machine learning model as it might have so much randomness so we need to clean the dataset properly in order to fetch good results.
This activity includes the following steps.

- Handling missing values
- Handling categorical data
- Handling Outliers

Note: These are the general steps of pre-processing the data before using it for machine learning. Depending on the condition of your dataset, you may or may not have to go through all these steps.

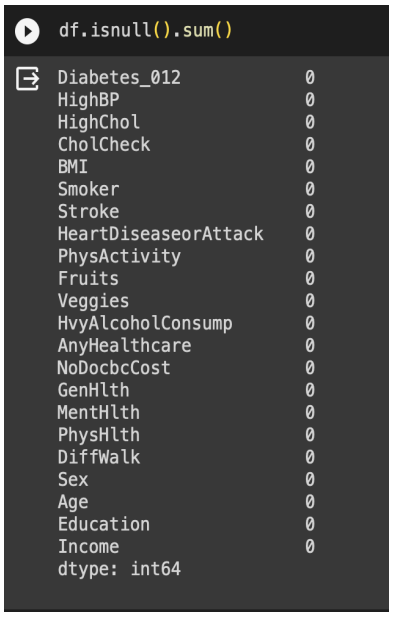
Activity 2.1: Handling missing values

- Let's know the info and describe our dataset first. To find the shape of our data, the df.shape method is used. To find the data type, df.info() function is used

```
df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 253680 entries, 0 to 253679
Data columns (total 22 columns):
 #   Column                Non-Null Count   Dtype
---  ------                --------------   -----
 0   Diabetes_012          253680 non-null  float64
 1   HighBP                253680 non-null  float64
 2   HighChol              253680 non-null  float64
 3   CholCheck             253680 non-null  float64
 4   BMI                   253680 non-null  float64
 5   Smoker                253680 non-null  float64
 6   Stroke                253680 non-null  float64
 7   HeartDiseaseorAttack  253680 non-null  float64
 8   PhysActivity          253680 non-null  float64
 9   Fruits                253680 non-null  float64
 10  Veggies               253680 non-null  float64
 11  HvyAlcoholConsump     253680 non-null  float64
 12  AnyHealthcare         253680 non-null  float64
 13  NoDocbcCost           253680 non-null  float64
 14  GenHlth               253680 non-null  float64
 15  MentHlth              253680 non-null  float64
 16  PhysHlth              253680 non-null  float64
 17  DiffWalk              253680 non-null  float64
 18  Sex                   253680 non-null  float64
 19  Age                   253680 non-null  float64
 20  Education             253680 non-null  float64
 21  Income                253680 non-null  float64
dtypes: float64(22)
memory usage: 42.6 MB
```

● For checking the null values, df.isnull() function is used. To sum those null values we use .sum() function. From the below image we found that there are no null values present in our dataset. So we can skip handling the missing values step.

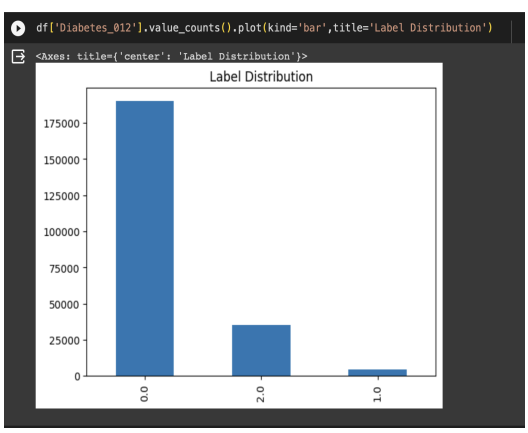


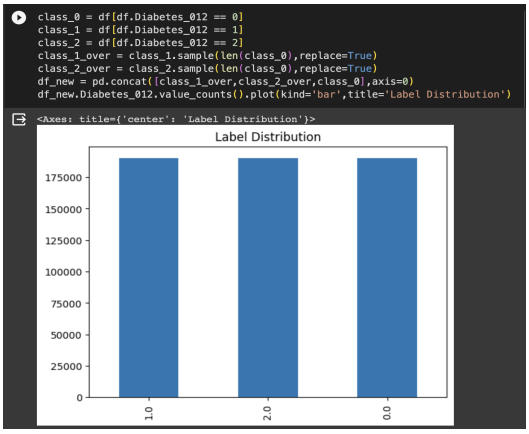
Activity 2.2: Handling Categorical Values

As we can see our dataset has categorical data, we must convert the categorical data to integer encoding or binary encoding.
To convert the categorical features into numerical features we use encoding techniques. There are several techniques but in our project we are using Label encoding with the help of list comprehension.
• In our project, there are no categorical features therefore no encoding has to be done.

Activity 2.3: Handling Imbalance Data

With the help of boxplot, outliers are visualized. And here we are going to find upper bound and lower bound of some columns feature with some mathematical formula ● From the below diagram, we could visualize that some of the features have outliers Boxplot from seaborn library is used here.





Milestone 3: Exploratory Data Analysis

Activity 1: Descriptive statistical

Descriptive analysis is to study the basic features of data with the statistical process. Here pandas has a worthy function called describe. With this describe function we can understand the unique, top and frequent values of categorical features. And we can find mean, std, min, max and percentile values of continuous features

.

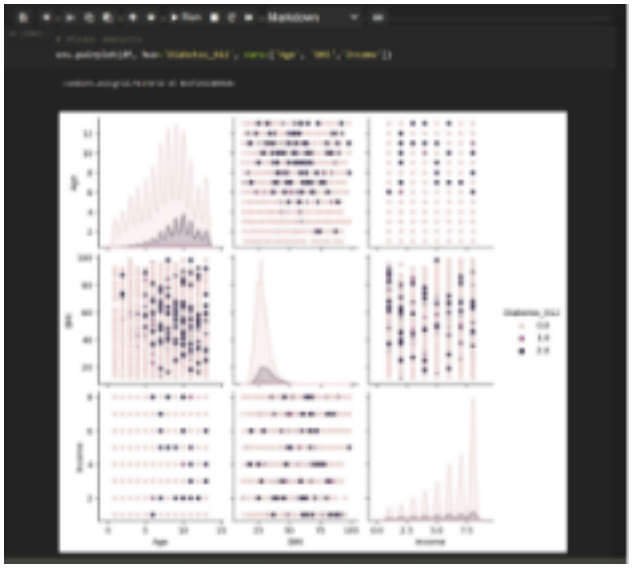
```
df.describe()
```

	Diabetes_012	HighBP	HighChol	CholCheck	BMI	Smoker	Stroke	HeartDiseaseorAttack	PhysActivity	Fruits	...	Any
count	253680.000000	253680.000000	253680.000000	253680.000000	253680.000000	253680.000000	253680.000000	253680.000000	253680.000000	253680.000000	...	25
mean	0.296921	0.429001	0.424121	0.962670	28.382364	0.443169	0.040571	0.094186	0.756544	0.634256	...	
std	0.698160	0.494934	0.494210	0.189571	6.608694	0.496761	0.197294	0.292087	0.429169	0.481639	...	
min	0.000000	0.000000	0.000000	0.000000	12.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	
25%	0.000000	0.000000	0.000000	1.000000	24.000000	0.000000	0.000000	0.000000	1.000000	0.000000	...	
50%	0.000000	0.000000	0.000000	1.000000	27.000000	0.000000	0.000000	0.000000	1.000000	1.000000	...	
75%	0.000000	1.000000	1.000000	1.000000	31.000000	1.000000	0.000000	0.000000	1.000000	1.000000	...	
max	2.000000	1.000000	1.000000	1.000000	98.000000	1.000000	1.000000	1.000000	1.000000	1.000000	...	

8 rows × 22 columns

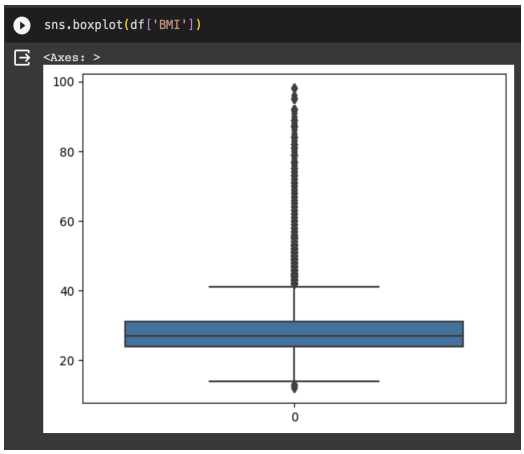
Activity 2: Visual analysis

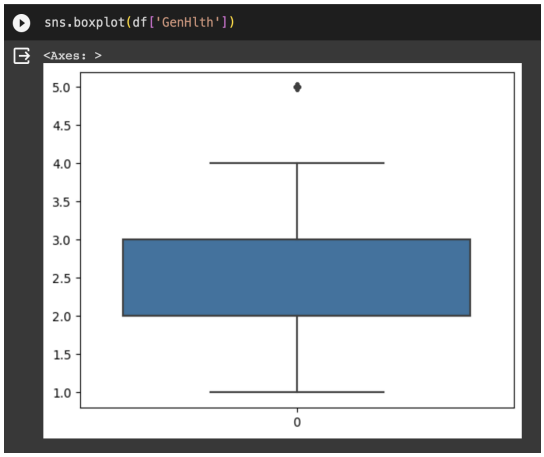
Visual analysis is the process of using visual representations, such as charts, plots, and graphs, to explore and understand data. It is a way to quickly identify patterns, trends, and outliers in the data, which can help to gain insights and make informed decisions.



Activity 2.1: Univariate analysis

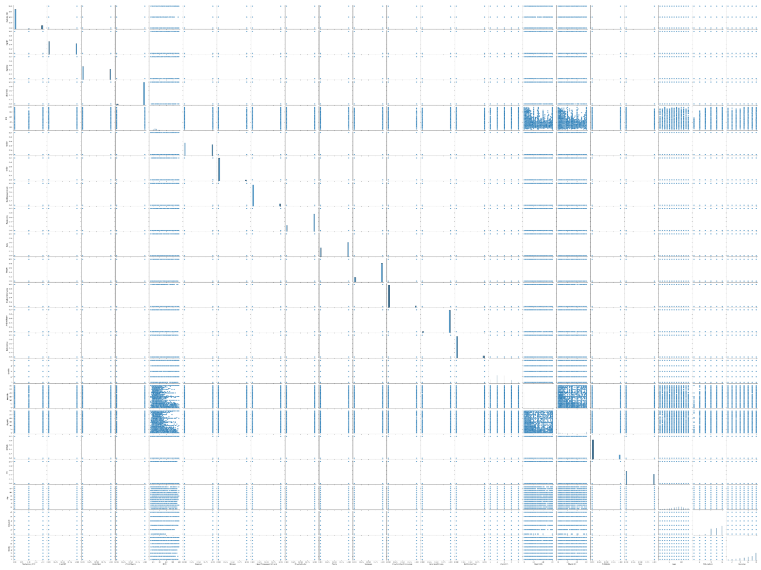
In simple words, univariate analysis is understanding the data with single feature. Here we have displayed two different graphs such as distplot and count plot.
In our dataset we have some categorical features. With the count plot function, we are going to count the unique category in those features.





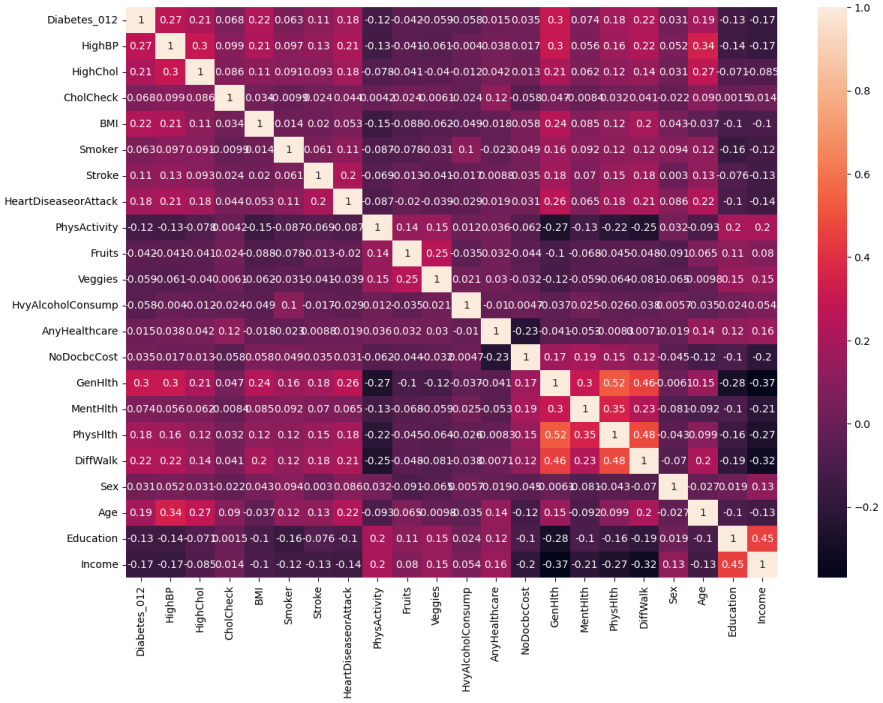
Activity 2.2: Bivariate analysis

To find the relation between two features we use bivariate analysis. Here we are visualizing. Countplot is used here. As a 1st parameter we are passing x value and as a 2nd parameter we are passing hue value.



Activity 2.3: Multivariate analysis

In simple words, multivariate analysis is to find the relation between multiple features. Here we have used a heatmap from the seaborn package.



Splitting data into train and test:

Now let's split the Dataset into train and test sets. First split the dataset into x and y and then split the data set

Here x and y variables are created. On x variable, df is passed with dropping the target variable. And my target variable is passed. For splitting training and testing data we are using the train_test_split() function from sklearn. As parameters, we are passing x, y, test_size, random_state.

```
x_train,x_test,y_train,y_test = train_test_split(X,y,test_size=0.15, random_state=13)

scalar = StandardScaler()
X_train = scalar.fit_transform(x_train)
X_test = scalar.fit_transform(x_test)
```

Milestone 4: Model Building

Activity 1: Training the model in multiple algorithms

Now our data is cleaned and it's time to build the model. We can train our data on different algorithms. For this project, we are applying three classification algorithms. The best model is saved based on its performance.

Activity 1.1: Random Forest Regressor

A function named random forest regressor is created and train and test data are passed as the parameters. Inside the function, a random forest regressor algorithm is initialized and training data is passed to the model with the fit() function. Test data is predicted with predict () function and saved in a new variable.

```
rfc = RandomForestClassifier(n_estimators=20,criterion='entropy',random_state=13)

rfc.fit(X_train,y_train)

RandomForestClassifier
RandomForestClassifier(criterion='entropy', n_estimators=20, random_state=13)

y_pred1 = rfc.predict(X_test)
```

Activity 1.2: Logistic Regression

To evaluate the performance of a logistic regression model, we need to test it on a separate dataset that it has not seen during training. This separate dataset is called the test data. We typically split the available data into two parts, the training data and the test data. The model is trained on the training data, and then tested on the test data to see how well it generalizes to new, unseen data.

```
lr = LogisticRegression()

lr.fit(X_train,y_train)

LogisticRegression
LogisticRegression()

y_pred2 = lr.predict(X_test)
```

Activity 1.3: XGBClassifier

XGBoost (eXtreme Gradient Boosting) is a powerful and widely-used gradient boosting algorithm that is used to solve many different types of machine learning problems. It is an implementation of gradient boosting that is specifically designed to be efficient and scalable,

making it a popular choice for working with large datasets.

```
[ ] xgb = XGBClassifier(n_estimators=20,random_state=13)

xgb.fit(X_train,y_train)
```

```
XGBClassifier(base_score=None, booster=None, callbacks=None,
              colsample_bylevel=None, colsample_bynode=None,
              colsample_bytree=None, device=None, early_stopping_rounds=None,
              enable_categorical=False, eval_metric=None, feature_types=None,
              gamma=None, grow_policy=None, importance_type=None,
              interaction_constraints=None, learning_rate=None, max_bin=None,
              max_cat_threshold=None, max_cat_to_onehot=None,
              max_delta_step=None, max_depth=None, max_leaves=None,
              min_child_weight=None, missing=nan, monotone_constraints=None,
              multi_strategy=None, n_estimators=20, n_jobs=None,
              num_parallel_tree=None, objective='multi:softprob', ...)
```

```
y_pred3 = xgb.predict(X_test)
```

Activity 2: Testing the model

Here we have tested with Logistic regression and Random Forest algorithms. With the help of predict() function.

```
[ ] y_pred1 = rfc.predict(X_test)
```

```
y_pred2 = lr.predict(X_test)
```

```
y_pred3 = xgb.predict(X_test)
```

Milestone 5: Performance Testing

Activity 1: Testing model with multiple evaluation metrics

Multiple evaluation metrics means evaluating the model's performance on a test set using different performance measures. This can provide a more comprehensive understanding of the model's strengths and weaknesses. We are using evaluation metrics for classification tasks including accuracy, precision, recall, support and F1-score.

```
as1 = accuracy_score(y_test,y_pred1)
as2 = accuracy_score(y_test,y_pred2)
as3 = accuracy_score(y_test,y_pred3)
```

```
[ ] as1
```

```
0.9335983630517393
```

```
[ ] as2
```

```
0.5095469161064017
```

```
[ ] as3
```

```
0.3914995615317159
```

```
[ ] pd.crosstab(y_test,y_pred1)
```

col_0	0.0	1.0	2.0
Diabetes_012			
0.0	23960	573	3918
1.0	0	28438	0
2.0	713	475	27448

```
[ ] pd.crosstab(y_test,y_pred2)
```

Diabetes_012 \ col_0	0.0	1.0	2.0
0.0	18217	5042	5192
1.0	8372	8466	11600
2.0	5085	6655	16896

```
pd.crosstab(y_test,y_pred3)
```

Diabetes_012 \ col_0	0	1	2
0.0	26789	972	690
1.0	23940	2582	1916
2.0	21680	2844	4112

```
[ ] classificationreport1 = classification_report(y_test,y_pred1)
    classificationreport2 = classification_report(y_test,y_pred2)
    classificationreport3 = classification_report(y_test,y_pred3)
```

```
print(classificationreport1)
```

	precision	recall	f1-score	support
0.0	0.97	0.84	0.90	28451
1.0	0.96	1.00	0.98	28438
2.0	0.88	0.96	0.91	28636
accuracy			0.93	85525
macro avg	0.94	0.93	0.93	85525
weighted avg	0.94	0.93	0.93	85525

```
print(classificationreport2)
```

	precision	recall	f1-score	support
0.0	0.58	0.64	0.61	28451
1.0	0.42	0.30	0.35	28438
2.0	0.50	0.59	0.54	28636
accuracy			0.51	85525
macro avg	0.50	0.51	0.50	85525
weighted avg	0.50	0.51	0.50	85525

```
[ ] print(classificationreport3)
```

	precision	recall	f1-score	support
0.0	0.37	0.94	0.53	28451
1.0	0.40	0.09	0.15	28438
2.0	0.61	0.14	0.23	28636
accuracy			0.39	85525
macro avg	0.46	0.39	0.30	85525
weighted avg	0.46	0.39	0.30	85525

Activity 1.1: Compare the model
For comparing the below two models, with their R_2 score on training and testing data. the results of models are displayed as output. From the below three models random forest regressor is performing well

Milestone 6: Model Deployment

Activity 1: Save the best model

Saving the best model after comparing its performance using different evaluation metrics means selecting the model with the highest performance This can be useful in avoiding the need to retrain the model every time it is needed and also to be able to use it in the future.

```
[ ]  pickle.dump(rfc,open('diabetes_prediction.pkl','wb'))
```

Activity 2: Integrate with Web Framework

In this section, we will be building a web application that is integrated to the model we built. A UI is provided for the uses where he has to enter the values for predictions. The enter values are given to the saved model and prediction is showcased on the UI.
This section has the following tasks
- Building HTML Pages
- Building server-side script
- Run the web application

Activity 2.1: Building Html Pages:

For this project create two HTML files namely
· index.html
· prediction.html
· result.html

and save them in the templates folder. Refer this ENTER THE LINK for templates, static and python file

Activity 2.2: Build Python code:

Import the libraries in python file

index.html | app.py

app.py > start

```
1   from flask import Flask, render_template,request
2   app = Flask(__name__)
3
4   import pickle
5   import numpy as np
6
7   model = pickle.load(open('diabetes_prediction.pkl','rb'))
8
```

Load the saved model. Importing the flask module in the project is mandatory. An object of Flask class is our WSGI application. Flask constructor takes the name of the current module (__name__) as argument.

Render HTML page:

Here we will be using a declared constructor to route to the HTML page which we have created earlier.

```
model = pickle.load(open('diabetes_prediction.pkl','rb'))
```

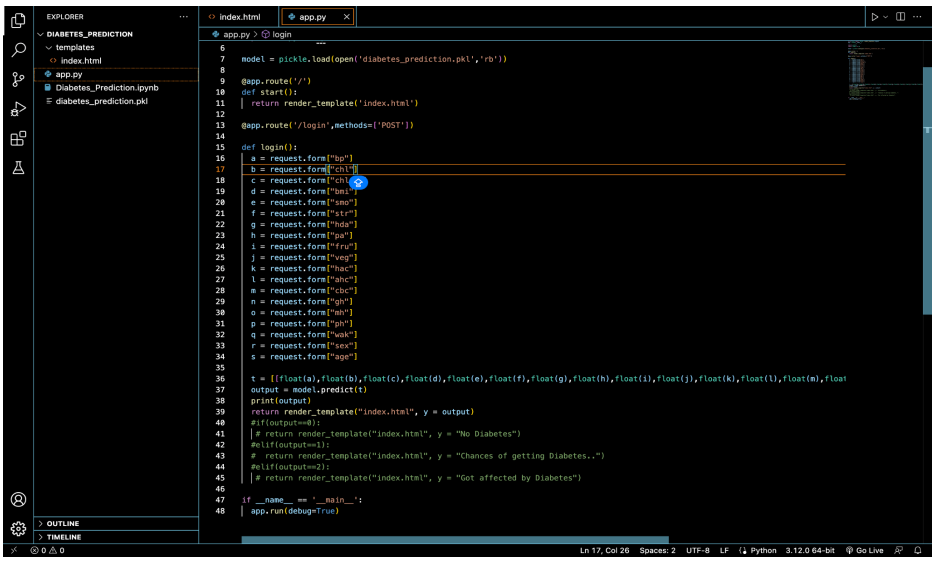
In the above example, '/' URL is bound with the index.html function. Hence, when the home page of the web server is opened in the browser, the html page will be rendered. Whenever you enter the values from the html page the values can be retrieved using POST Method.

Retrieves the value from UI:

```
@app.route('/')
def start():
    return render_template('index.html')
```

Here we are routing our app to prediction () function. This function retrieves all the values from the HTML page using Post request. That is stored in an array. This array is passed to the model Predict() function. This function returns the prediction. And this prediction value will be rendered to the text that we have mentioned in the submit.html page earlier.

Main Function:

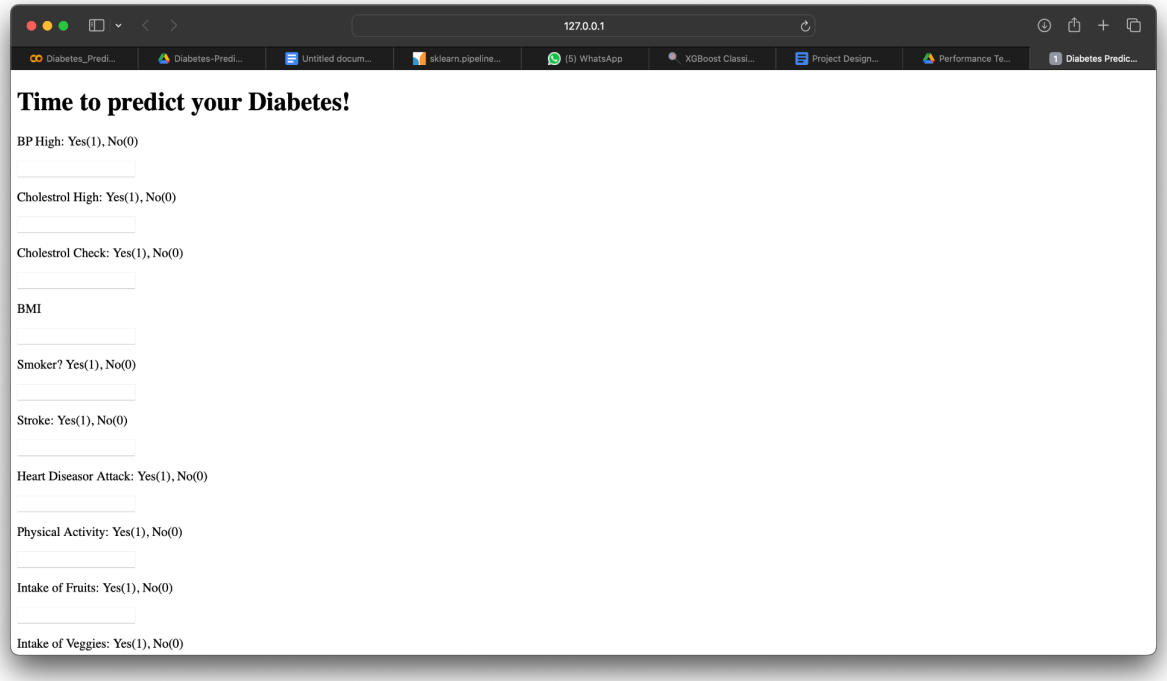


Activity 2.3: Run the web application

- Open anaconda prompt from the start menu
- Navigate to the folder where your python script is.
- Now type “python app.py” command
- Navigate to the localhost where you can view your web page.
- Click on the predict button from the top left corner, enter the inputs, click on the submit button, and see the result/prediction on the web.

```
mohit@Mohiths-MacBook-Pro Diabetes_prediction % python3 app.py
 * Serving Flask app 'app'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 293-040-248
127.0.0.1 - - [08/Nov/2023 18:25:23] "GET / HTTP/1.1" 200 -
[2.]
127.0.0.1 - - [08/Nov/2023 18:26:16] "POST /login HTTP/1.1" 200 -
```

Now, Go the web browser and write the localhost url (http://127.0.0.1:5000) to get the below result



8. PERFORMANCE TESTING

Model Performance Testing:

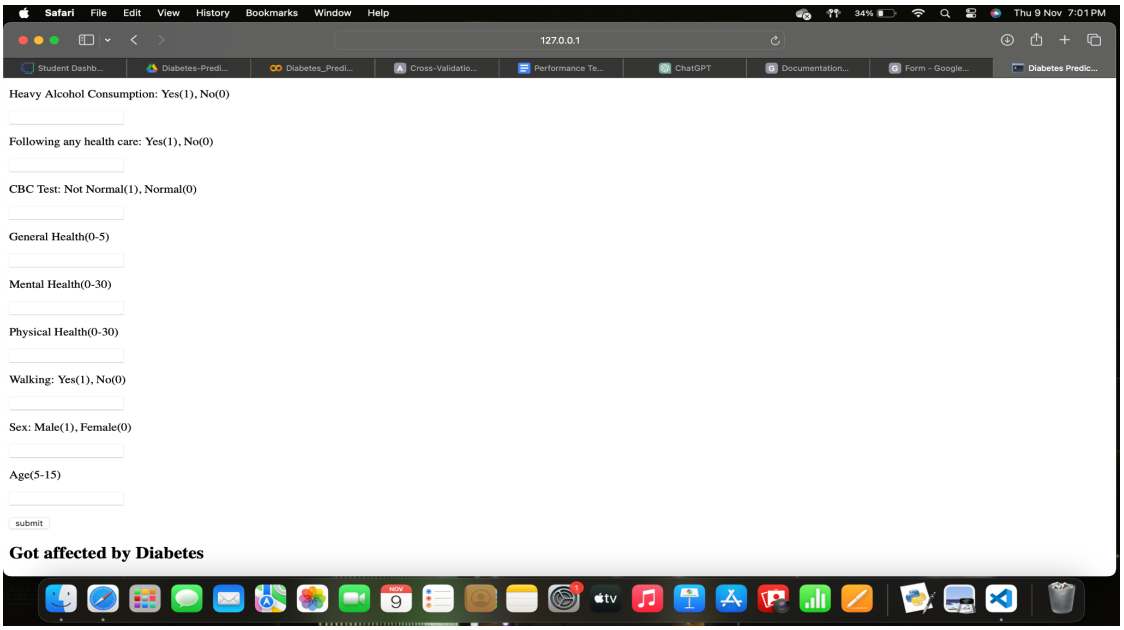
Project team shall fill the following information in the model performance testing template.

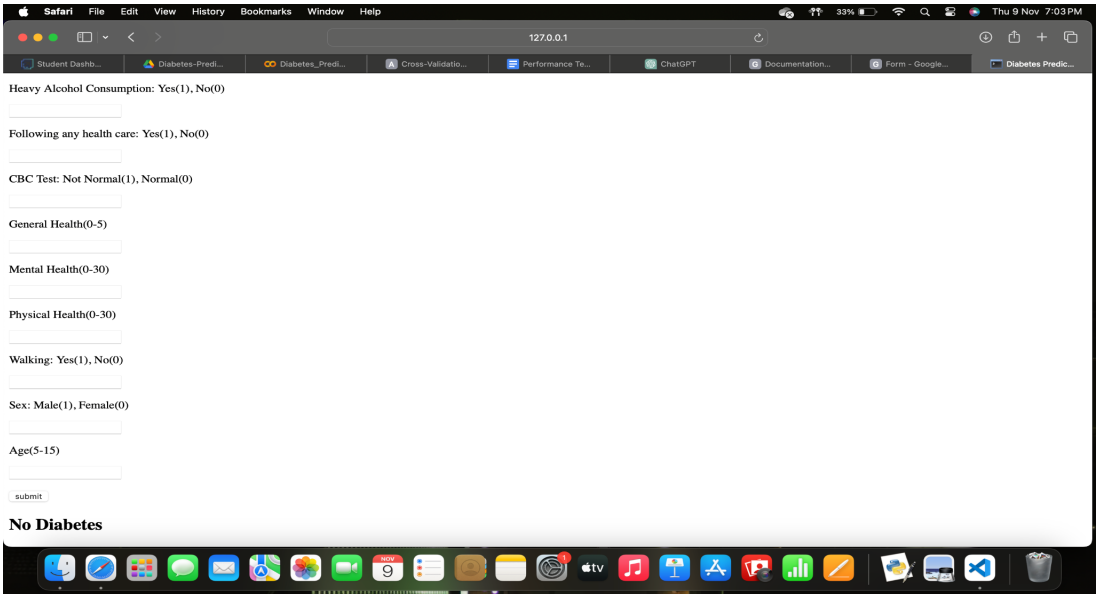
S.No.	Parameter	Values	Screenshot

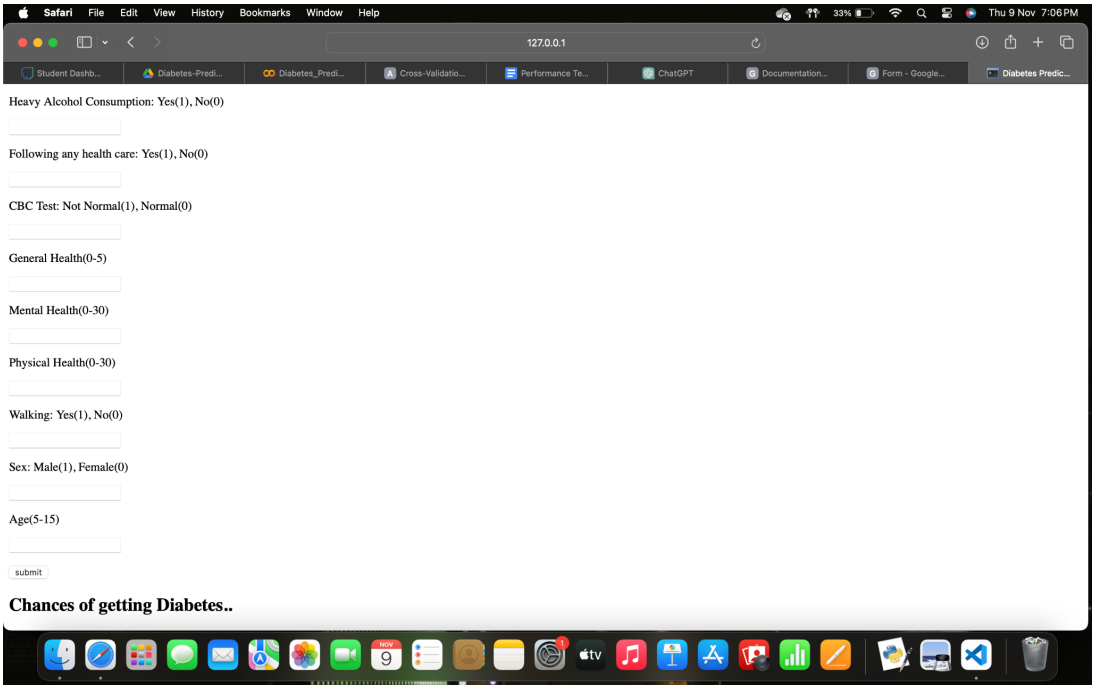
1.	Metrics	**Regression Model:** MAE, MSE , RMSE , R2 score
		
		Classification Model: Confusion Matrix, Accuray Score & Classification Report
			`[32] rfc = RandomForestClassifier(n_estimators=20,criterion='entropy',random_state=13)`
			
			`[33] rfc.fit(X_train,y_train)`
			
			`RandomForestClassifier(criterion='entropy', n_estimators=20, random_state=13)`
			
			`[34] y_pred1 = rfc.predict(X_test)`
			
			Regression Model:
			
			`from sklearn.svm import SVC`
			`from sklearn.datasets import make_classification`
			`from sklearn.pipeline import Pipeline`
			`from sklearn.metrics import mean_absolute_error`
			`from sklearn.metrics import mean_squared_error`
			`from sklearn.metrics import r2_score`
			
			`[53] MAE = mean_absolute_error(y_test,y_pred1)`
			`MAE.round(2)`
			
			`0.12`
			
			`[54] MSE = mean_squared_error(y_test,y_pred1)`
			`MSE.round(2)`
			
			`0.23`
			
			`[55] RMSE = MSE**(1/2)`
			`RMSE.round(2)`
			
			`0.48`
			
			`[56] R2 = r2_score(y_test,y_pred1)`
			`R2`
			
			`0.654524878682704`
			
			Classification Model:
			
			`pd.crosstab(y_test,y_pred1)`
			
			col_0 | 0.0 | 1.0 | 2.0
			Diabetes_012
			0.0 | 23983 | 571 | 3897
			1.0 | 0 | 28421 | 17
			2.0 | 759 | 510 | 27367
			
			`[41] as1 = accuracy_score(y_test,y_pred1)`
			
			`[42] as1`
			
			`0.9327214264834843`

			[48] classificationreport1 = classification_report(y_test,y_pred1) print(classificationreport1) precision recall f1-score support 0.0 0.97 0.84 0.90 28451 1.0 0.96 1.00 0.98 28438 2.0 0.87 0.96 0.91 28636 accuracy 0.93 85525 macro avg 0.94 0.93 0.93 85525 weighted avg 0.94 0.93 0.93 85525
2.	Tune the Model	**Hyperparameter Tuning, Validation Method**	**Hyperparameter Tuning:** from sklearn.pipeline import make_pipeline from sklearn.model_selection import GridSearchCV pipe = make_pipeline((RandomForestClassifier())) grid_param = [{"randomforestclassifier": [RandomForestClassifier()], "randomforestclassifier__n_estimators":[10,20,50], "randomforestclassifier__random_state":[6,7,13]}] gridsearch = GridSearchCV(pipe,grid_param,cv=5,verbose=0,n_jobs=-1) [65] best_model =gridsearch.fit(X_train,y_train) best_model.score(X_test,y_test) 0.9347208418591055 **Validation Method:** KFold Method [68] from sklearn.model_selection import KFold kfold = KFold(5,shuffle=True) for train,test in kfold.split(y): print("Train:",train,"Test:",test) Train: [0 2 3 ... 570162 570163 570164] Test: [1 4 5 ... 570130 570148 570161] Train: [0 1 2 ... 570161 570162 570163] Test: [3 11 15 ... 570154 570156 570164] Train: [0 1 2 ... 570160 570161 570164] Test: [8 14 18 ... 570158 570162 570163] Train: [1 3 4 ... 570162 570163 570164] Test: [0 2 10 ... 570146 570150 570157] Train: [0 1 2 ... 570162 570163 570164] Test: [6 7 9 ... 570153 570159 570160]

9. OUTPUT SCREENSHOTS







10. ADVANTAGES & DISADVANTAGES

Advantages:

Early Diagnosis: Machine learning models can identify individuals at risk of diabetes even before clinical symptoms appear, enabling early intervention and treatment.

Personalization: Predictive models can offer personalized risk assessments and recommendations for individuals based on their unique characteristics, improving patient outcomes.

Efficiency: Machine learning can process and analyze large data sets quickly, making it a cost-effective

and a time-efficient way to predict diabetes risk.
Improved Public Health: Early prediction and prevention can reduce the overall burden of diabetes and its complications, leading to better public health.
Data-Driven Insights: These models can provide valuable insights into risk factors and disease mechanisms, aiding medical research.
Resource Allocation: Healthcare resources can be allocated more efficiently, targeting high-risk individuals for screenings and interventions.
Continuous Monitoring: Real-time monitoring and alerts can help individuals manage their health more effectively.
Scalability: Machine learning models can be scaled to serve large populations and healthcare systems.

Disadvantages:

Data Privacy: Handling sensitive health data raises privacy and security concerns, requiring robust protection measures.
Data Quality: Model accuracy relies on the quality of data, and incomplete or inaccurate data can lead to unreliable predictions.
Bias: Models can inherit biases from the training data, leading to unfair predictions, especially for underrepresented groups.
Overfitting: Models may overfit to training data, performing well in training but poorly on unseen data.
Clinical Validation: Real-world clinical validation is necessary to confirm the effectiveness of predictive models.
Infrastructure and Cost: Implementing machine learning systems may require significant infrastructure and financial investments.
User Engagement: Encouraging individuals to actively engage with and follow recommendations from predictive systems can be a challenge.
Model Updates: Ensuring that models remain accurate over time requires ongoing updates and maintenance.

11. CONCLUSION

The main aim of this project was to design and implement Diabetes Prediction Using Machine Learning Methods and Performance Analysis of those methods and it has been achieved successfully. The proposed approach uses various classification and ensemble learning methods in which Random Forest, Logistic Regression and XGBClassifier are used. And 93.4% classification accuracy has been achieved. The Experimental results can assist health care to take early predic- tion and make early decisions to cure diabetes and save humans life.

12. FUTURE SCOPE

The future scope for diabetes prediction using machine learning is promising and multifaceted. Advancements in machine learning algorithms and the availability of vast and diverse healthcare datasets will lead to more accurate and personalized predictions of diabetes risk. Machine learning models will not only predict diabetes risk but also differentiate between subtypes like Type 1 and Type 2 diabetes at earlier stages.

Integration with wearable devices and continuous glucose monitoring systems will enable real-time data collection, improving the precision and timeliness of predictions. The growth of telemedicine will facilitate remote risk assessments and monitoring, particularly in underserved areas.

Predictive models will offer more detailed lifestyle recommendations, such as personalized diet and exercise plans, to reduce diabetes risk. Machine learning will play a crucial role in population-level health management by identifying at-risk populations and informing public health policies.

User-friendly mobile applications will integrate machine learning for easy access to risk assessment tools, and ongoing model updates will ensure accuracy. Ethical considerations, fairness, and interpretability will be prioritized, along with the development of regulatory frameworks specific to AI in healthcare. The future of diabetes prediction using machine learning holds immense potential to transform diabetes management and prevention, with global implications.

13. APPENDIX

Source Code

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Diabetes Prediction</title>
</head>
<body>
<h1>Time to predict your Diabetes!</h1>

<form action ="/login" method ="post">
<p>BP High: Yes(1), No(0)</p>
<p><input type="text" name="bp"/></p>

<p>Cholestrol High: Yes(1), No(0)</p>
<p><input type="text" name="chl"/></p>
```

```
<p>Cholestrol Check: Yes(1), No(0)</p>
<p><input type="text" name="chlck"/></p>

<p>BMI</p>
<p><input type="text" name="bmi"/></p>

<p>Smoker? Yes(1), No(0)</p>
<p><input type="text" name="smo"/></p>

<p>Stroke: Yes(1), No(0)</p>
<p><input type="text" name="str"/></p>

<p>Heart Diseasor Attack: Yes(1), No(0)</p>
<p><input type="text" name="hda"/></p>

<p>Physical Activity: Yes(1), No(0)</p>
<p><input type="text" name="pa"/></p>

<p>Intake of Fruits: Yes(1), No(0)</p>
<p><input type="text" name="fru"/></p>

<p>Intake of Veggies: Yes(1), No(0)</p>
<p><input type="text" name="veg"/></p>

<p>Heavy Alcohol Consumption: Yes(1), No(0)</p>
<p><input type="text" name="hac"/></p>

<p>Following any health care: Yes(1), No(0)</p>
<p><input type="text" name="ahc"/></p>

<p>CBC Test: Not Normal(1), Normal(0)</p>
<p><input type="text" name="cbc"/></p>

<p>General Health(0-5)</p>
<p><input type="text" name="gh"/></p>

<p>Mental Health(0-30)</p>
<p><input type="text" name="mh"/></p>

<p>Physical Health(0-30)</p>
<p><input type="text" name="ph"/></p>
```

```
<p>Walking: Yes(1), No(0)</p>
<p><input type="text" name="wak"/></p>

<p>Sex: Male(1), Female(0)</p>
<p><input type="text" name="sex"/></p>

<p>Age(5-15)</p>
<p><input type= "text" name="age"/></p>

<p><input type="submit" value="submit"/></p>

<h2><b>{{y}}</b></h2>
</form>
</body>
</html>
```

GitHub Link & Project Demo Link

GitHub Link: Diabetes_prediction.zip
Project Demo Link: https://drive.google.com/file/d/1z6X8BDoNEflzRqJ1aA6v2AhNbIJfNeS5/view?usp=share_link