Predicting Type 2 Diabetes Risk in Individuals Under 40: A Machine Learning Approach

Name: Srinivasachari Vangavolu

Student ID: @00649671

Date: 30/04/2023



School of Computer Science and Engineering,

University of Salford,

Manchester, UK.

Supervisor: Dan Ali

Table of Contents

[CHAPTER 1 7](#_Toc133917069)

[1.1 Introduction 7](#_Toc133917070)

[1.2 Problem statement: 7](#_Toc133917071)

[1.3 OBJECTIVES OF THE STUDY 8](#_Toc133917072)

[CHAPTER 2 9](#_Toc133917073)

[2.1 Overview of Diabetes 9](#_Toc133917074)

[2.2 Diabetes and its associated Mechanism 9](#_Toc133917075)

[2.3 Types of diabetes 9](#_Toc133917076)

[2.4 Complications caused by Diabetes 10](#_Toc133917077)

[2.5 Causes of diabetes 10](#_Toc133917078)

[2.6 Treatment and management of diabetes. 10](#_Toc133917079)

[2.7 Machine learning approaches for detecting diabetes. 11](#_Toc133917080)

[2.7.1 Logistic Regression: 11](#_Toc133917081)

[2.7.2 Support Vector Machines (SVM): 11](#_Toc133917082)

[2.7.3 Artificial Neural Networks (ANN): 11](#_Toc133917083)

[2.7.4 KNN Algorithm: 12](#_Toc133917084)

[2.7.5 LSTM: 12](#_Toc133917085)

[2.8 Deep learning approaches of detecting diabetes. 12](#_Toc133917086)

[2.8.1 Convolutional Neural Networks (CNNs) 13](#_Toc133917087)

[2.8.2 Recurrent Neural Networks (RNNs): 13](#_Toc133917088)

[2.8.3 Deep Belief Networks (DBNs): 13](#_Toc133917089)

[2.8.4 Autoencoders: 13](#_Toc133917090)

[2.8.5 Generative Adversarial Networks (GANs) 13](#_Toc133917091)

[2.9 Relates to past works. 13](#_Toc133917092)

[CHAPTER 3 15](#_Toc133917093)

[3.1 Methodology: 15](#_Toc133917094)

[3.1.1 Loading the Dataset: 15](#_Toc133917095)

[3.1.2 Cleaning the Data: 15](#_Toc133917096)

[3.1.3 Understanding the Data 15](#_Toc133917097)

[3.1.4 Splitting the Data: 16](#_Toc133917098)

[3.1.5 Training the dataset: 16](#_Toc133917099)

[3.1.6 Testing the Dataset: 16](#_Toc133917100)

[3.1.7 Evaluating the Model: 17](#_Toc133917101)

[3.2 Data Mining and Machine Learning Techniques 17](#_Toc133917102)

[3.3 Supervised Learning 17](#_Toc133917103)

[3.4 Unsupervised Learning 18](#_Toc133917104)

[3.5 Reinforcement Learning 18](#_Toc133917105)

[3.6 Data Extraction: 18](#_Toc133917106)

[3.7 Normalization: 19](#_Toc133917107)

[3.8 EDA: 19](#_Toc133917108)

[3.8.1 Histograms: 19](#_Toc133917109)

[3.8.2 BOX PLOTS: 20](#_Toc133917110)

[CHAPTER 4 24](#_Toc133917111)

[4.1 Analasys and approch 24](#_Toc133917112)

[4.1.1 MODEL SELECTION: 24](#_Toc133917113)

[4.2 Logistic Regression: 24](#_Toc133917114)

[4.2.1 Model Training 24](#_Toc133917115)

[4.2.2 Model Evaluation 25](#_Toc133917116)

[4.2.3 Conclusion 26](#_Toc133917117)

[4.3 KNN: 27](#_Toc133917118)

[4.3.1 Building the Model: 27](#_Toc133917119)

[4.3.2 Evaluate the Model: 28](#_Toc133917120)

[4.3.3 Conclusion: 30](#_Toc133917121)

[4.4 SVM: 30](#_Toc133917122)

[4.4.1 Building Model: 31](#_Toc133917123)

[4.4.2 Training the Model: 31](#_Toc133917124)

[4.4.3 Evaluating the Model: 32](#_Toc133917125)

[4.4.4 Conclusion: 32](#_Toc133917126)

[4.5 LSTM: 32](#_Toc133917127)

[4.5.1 Define the LSTM model: 33](#_Toc133917128)

[4.5.2 Training the Model: 34](#_Toc133917129)

[4.5.3 Evaluate the Model: 35](#_Toc133917130)

[4.6 ANN: 35](#_Toc133917131)

[4.6.1 Building the ANN Model 36](#_Toc133917132)

[4.6.2 Evaluation the model 37](#_Toc133917133)

[CHAPTER 5 39](#_Toc133917134)

[5.1 Comparison 39](#_Toc133917135)

[CHAPTER 6 41](#_Toc133917136)

[6.1 Conclusion: 41](#_Toc133917137)

[6.2 Future work 41](#_Toc133917138)

[CHAPTER 7 42](#_Toc133917139)

List of Figures

[Figure 3‑1 : Histogram 18](#_Toc133149023)

[Figure 3‑2 : Box Plot 19](#_Toc133149024)

[Figure 3‑3 : Correlation Matrix 20](#_Toc133149025)

[Figure 3‑4 : Correlation Matrix 21](#_Toc133149026)

[Figure 4‑1 : Dataset Splitting 22](#_Toc133149027)

[Figure 4‑2 : Model Traing 23](#_Toc133149028)

[Figure 4‑3 : Model Results 24](#_Toc133149029)

[Figure 4‑4 : Dataset Splitting 25](#_Toc133149030)

[Figure 4‑5 : Training the Model 26](#_Toc133149031)

[Figure 4‑6 : Model Results 27](#_Toc133149032)

[Figure 4‑7: Confusion Matrix 28](#_Toc133149033)

[Figure 4‑8 : Splitting the Dataset 29](#_Toc133149034)

[Figure 4‑9: Training the Model 29](#_Toc133149035)

[Figure 4‑10 : Model Results 30](#_Toc133149036)

[Figure 4‑15 : Splitting the Dataset 31](#_Toc133149037)

[Figure 4‑16 : Creating the LSTM Model 32](#_Toc133149038)

[Figure 4‑17 : Training the model 33](#_Toc133149039)

[Figure 4‑18 : Model Results 33](#_Toc133149040)

Abstract:

In the past decade, diabetes has emerged as a chronic illness with the potential to have a substantial influence on global healthcare. According to the International Diabetes Federation, about 380 million people worldwide have diabetes, and this figure is projected to increase to 590 million by 2035. Diabetes is defined by high blood glucose levels, and it can be difficult to diagnose owing to the complicated interaction of several variables that impact numerous organs. Data science technologies have the potential to bring new insights in a variety of scientific domains, including the use of machine learning to medical outcome prediction. The objective of this project is to create a system that reliably predicts early-stage diabetes by merging three supervised machine learning techniques: support vector machine (SVM), logistic regression, and artificial neural networks (ANN). In addition, the project aims to build an effective approach for early diabetes detection for age under 40 and to identify the major risk factors that contribute to its development. The Pima Indians Diabetes dataset will be used to train and evaluate machine learning models. This dataset comprises age, BMI, blood pressure, and glucose level. Each model's performance will be evaluated using many evaluation parameters, including accuracy, precision, recall, and F1-score, and the model with the highest performance will be chosen. This technique has the potential to aid healthcare professionals in making informed decisions at an early stage.

# 

## Introduction

Diabetes is a chronic metabolic condition that negatively impacts the lives of millions of individuals worldwide. Diabetes can be detected and predicted in advance, which may substantially enhance patient outcomes by allowing for earlier treatment and intervention decisions. Machine learning (ML) techniques have demonstrated promising results in predicting diabetes risk because they can evaluate massive datasets and disclose intricate patterns that may not be apparent using conventional statistical methods. Massive datasets can be evaluated by machine learning methods, proving the veracity of these findings. It has been demonstrated that these findings are the result of the capacity of ML approaches to evaluate enormous datasets and identify intricate patterns. In this initiative, we aim to build a prediction model for diabetes based on machine learning algorithms to aid in early detection and treatment. This will result in improved outcomes overall.

The area of machine learning has made a lot of progress in finding people with diabetes, but it still has to deal with a number of problems. A big problem is that there aren't enough different and large samples for training and testing machine learning models. Many studies have only used a small number of similar datasets, which makes it hard to apply models to different groups and situations. There are different ways to figure out if someone has diabetes. Since different studies use different methods, it is hard to compare the results.

ML models used to predict diabetes are also thought to be hard to understand. Due to the "black box" structure of neural networks and ensemble methods, it is hard to figure out how they work. Medical workers and other people who work in healthcare can improve the accuracy of diabetes predictions by understanding and validating the model that is used. But there are a lot of problems with using machine learning models in healthcare facilities and other real-world settings. More research needs to be done to find out if machine learning models can be used in the healthcare business, if they can be scaled up, and what kind of effect they might have on patient results.

Diabetes can be found by using data, machine learning techniques, model review, group approaches, and other methods. ML models used to predict diabetes need to be improved so that they are more accurate, clear, and useful. It is suggested that researchers use big datasets, standardize traits, make models easier to understand, and try out machine learning models in clinical settings in their future study. Machine learning could find and predict the start of diabetes early on, making it easier to control and reducing the risks that come with it.

## Problem statement:

This study aims to develop a predictive model for diabetes based on machine learning. Diabetes is a chronic disease that affects the health of the entire world population and millions of people worldwide. If diabetes is detected and correctly diagnosed at a later stage, it is possible to effectively manage the condition and avoid further complications. On the other hand, an accurate prognosis of diabetes requires a thorough examination of multiple risk factors and their interdependencies.

Build a machine learning model that can accurately predict an individual's likelihood of developing diabetes based on multiple input variables, including age, body mass index (BMI), blood pressure, insulin levels, and family history of diabetes. Based on various input variables, this model should accurately predict the probability that an individual will develop diabetes. The model must be flexible enough to accommodate different types of data, such as numeric and categorical variables, missing values, and outliers. To ensure the effectiveness of the model, the goal is to create a diabetes prediction model that is as accurate and reliable as possible. This model will help clinicians identify individuals at risk for diabetes and implement strategies for early intervention and prevention.

## OBJECTIVES OF THE STUDY

To create an accurate machine learning model that can predict how likely it is that a certain group will get diabetes.

To find the most important risk factors and traits that can be used to predict diabetes risk, and to build a model that takes these into account.

To compare how well different machine learning algorithms or models work at predicting diabetes and to find out which method works best.

To check how well the machine learning model works by using separate datasets or clinical data from the real world.

To figure out how different interventions or methods for lowering risk affect diabetes risk by modelling and predicting how these interventions will work.

To investigate how machine learning could be used to predict diabetes risk for each individual patient by making risk ratings based on their unique traits, demographics, and health history.

# 

Literature Review

## Overview of Diabetes

Diabetes is a long-term metabolic disease that changes how well the body can use and store glucose. Diabetes is a major public health problem around the world. In 2019, an expected 463 million people will have diabetes. Diabetes comes in two main forms: type 1 and type 2. Type 1 diabetes happens when the immune system attacks and kills cells in the pancreas that make insulin. Type 2 diabetes happens when the body stops responding to insulin or doesn't make enough insulin to keep blood sugar levels in check. Diabetes can cause major problems if it is not managed, but people with diabetes can live long, healthy lives if they take care of their condition and make changes to how they live. The goal of this thesis is to look at how diabetes affects people's quality of life and health, as well as how well different treatment methods and approaches work to keep diabetes under control.

## Diabetes and its associated Mechanism

Diabetes has more than one cause. Diabetes is caused by not being able to control blood sugar well. This is caused by not having enough insulin or being resistant to insulin, which controls how much sugar is in the blood.

In type 1 diabetes, the immune system kills the cells in the pancreas that make insulin. This means there is no insulin at all. Type 2 diabetes is caused by insulin resistance, which leads to a relative lack of insulin. Insulin resistance is caused by being overweight, not being active, and having certain genes. Insulin resistance and type 2 diabetes are caused by obesity because fatty tissue gives off chemicals that cause inflammation and interfere with insulin signaling. Diabetes is caused by long-term inflammation. Chronic inflammation hurts organs and can lead to diabetes and other long-term diseases. Diabetes has also been linked to oxidative stress, which is a mismatch between making free radicals and getting rid of them. High blood sugar can cause oxidative stress, which can damage cells and lead to diabetes and its effects. Microorganisms in the stomach system, called the gut microbiome, also play a role in diabetes. The microbiome in the gut changes metabolism, inflammation, and the risk of getting diabetes.

## Types of diabetes

Some kinds of diabetes are:

In type 1 diabetes, the defense system kills the pancreas cells that make insulin. Low insulin. It mostly happens to teens, but anyone can get it. 90–95% of people with diabetes are type 2. Diabetes happens when the body stops reacting to insulin or can't make enough of it to control the amount of sugar in the blood. It is linked to being overweight, not being active, and not eating well.

Diabetes caused by pregnancy is called gestational diabetes. It only happens to pregnant women and goes away after the baby is born. During pregnancy, it could happen if the body doesn't make enough insulin to control the blood sugar. LADA diabetes has features of both type 1 and type 2 diabetes. Because type 2 diabetes is so common, it is often given the wrong name. It causes inflammation, just like type 1 diabetes.

"MODY" stands for "diabetes that starts in young adulthood." This rare form of diabetes is caused by changes in genes. Doctors often mistake it for type 1 or type 2 diabetes in people under 25. This type of diabetes is caused by pancreatitis or Cushing's syndrome.

All types of diabetes cause blood sugar to be too high, which can be dangerous.

## Complications caused by Diabetes

If untreated, diabetes may cause several problems. These disorders may harm the eyes, kidneys, nerves, and blood vessels. The most common diabetes complications are:

Diabetes increases the risk of cardiovascular disease, including heart disease, stroke, and artery diseases beyond the heart. Renal Diabetes may cause long-term renal dysfunction and kidney failure. High blood sugar may induce neuropathy, which damages nerves throughout the body. This might make hands and feet tingle, numb, and hurt.

Diabetes may damage retinal blood vessels, causing vision loss or blindness. Insufficient blood supply may damage foot nerves, causing foot ulcers, infections, and even amputations. Diabetes may cause bacterial and fungal infections, intense itching, and dry, flaking skin. Stomach nerves malfunction in gastroparesis. This causes digestive difficulties and slow stomach emptying. Dental issues Diabetes increases your risk of gum disease, cavities, and other dental disorders.

Diabetes patients must regulate their blood sugar levels by lifestyle changes, medication, and frequent medical exams. Thus, we can avoid or delay these issues.

## Causes of diabetes

Diabetes causes vary by kind.

An autoimmune response damages pancreatic insulin-producing cells, causing type 1 diabetes. Genetic and environmental factors may induce an autoimmune response. Genetics and lifestyle create type 2 diabetes. In type 2 diabetes, insulin resistance raises blood sugar. Obesity, inactivity, poor nutrition, age, family history, and medical problems such polycystic ovarian syndrome are risk factors for type 2 diabetes.

Pregnancy hormones impair insulin utilisation, causing gestational diabetes. Gestational diabetes is more likely in obese or diabetic women. Genetic mutations produce LADA and MODY, rare forms of diabetes. Age, family history, ethnicity, and medical disorders including high blood pressure and cholesterol raise the risk of diabetes. Obesity and inactivity increase type 2 diabetes risk.

## Treatment and management of diabetes.

Treatment and control of diabetes depend on the type, intensity, and health of the person. Diabetes care aims to keep blood sugar levels stable so that problems don't happen or don't happen as quickly.

Diabetics with type 1 need to replace their insulin. You can inject or pump insulin. Blood sugar, food, and exercise all affect how much insulin you need. Treatment for Type-2 diabetes The main ways to treat type 2 diabetes are to change your diet, get more exercise, and lose weight. Metformin, sulfonylureas, and DPP-4 inhibitors all lower blood sugar. People with gestational diabetes are told to eat better and move more to control their blood sugar. Use insulin if you need to.

glucose testing People with diabetes must check their blood sugar every day to stay in a certain range. This may mean that you need to check your blood sugar often with a fingerstick or a device. Fixing what went wrong. Diabetes has been linked to damage to the heart, kidneys, and nerves. Change the way you live and go to the doctor often to fix these problems. With information and help with self-management, people with diabetes may be able to improve their quality of life. People with diabetes may need bariatric surgery or a transfer of islet cells.

People with diabetes should work with their healthcare team to come up with a treatment and control plan that fits their needs.

## Machine learning approaches for detecting diabetes.

### Logistic Regression:

Logistic regression is a simple and widely used machine learning algorithm for binary classification. It models the probability of an event occurring (in this case, the probability of a patient having diabetes) using a logistic function. The algorithm tries to find the optimal values of the weights for the input features to maximize the likelihood of the observed data. Logistic regression is a linear model, which means it assumes a linear relationship between the input features and the output label. However, it can be extended to handle non-linear relationships using techniques such as polynomial features and regularization.

Chart, diagram

Description automatically generated

Equation - Logistic Reggression

### Support Vector Machines (SVM):

Support vector machines are a powerful machine learning algorithm for binary classification. The basic idea behind SVM is to find the hyperplane that best separates the two classes in the feature space. SVM can handle non-linear relationships between the input features and the output label by using kernel functions. Kernel functions transform the input features into a higher-dimensional feature space where the classes may be separable by a hyperplane.

In machine learning, support vector machines (also called support vector networks) are guided learning models that use learning methods to analyse data used for classification and regression analysis. A Support Vector Machine (SVM) is a type of algorithm that can tell the difference between two things. In other words, when the algorithm is given labelled training data (guided learning), it makes an ideal hyperplane that sorts new examples into categories.

### Artificial Neural Networks (ANN):

Artificial neural networks are a class of machine learning algorithms that are inspired by the structure and function of the human brain. ANN consists of multiple layers of interconnected nodes (neurons) that process the input features and produce the output label. ANN can handle complex relationships between the input features and the output label, and can learn non-linear decision boundaries. ANN typically requires a larger amount of data to train compared to logistic regression and SVM, and it may be more prone to overfitting if the architecture is not carefully chosen.

he formula used to calculate the output of each node in an ANN is called the activation function, and one commonly used activation function is the sigmoid function:

σ(z) = 1 / (1 + e^(-z))

where z is the weighted sum of the inputs to the node.

### KNN Algorithm:

The K-Nearest Neighbors (KNN) algorithm is a popular machine learning algorithm used for both classification and regression tasks. It is a non-parametric and lazy learning algorithm that makes predictions by finding the K closest data points in the training set and using their class or value as the prediction for a new data point.

Diagram

Description automatically generated with low confidence

Equation - KNN

### LSTM:

Long Short-Term Memory (LSTM) is a type of recurrent neural network (RNN) that is often used in Machine Learning (ML) for tasks that involve sequence or time-series data. LSTMs are a type of deep learning model that can learn patterns and relationships in sequential data. This makes them good for uses like speech recognition, natural language processing, time-series predictions, and more.

The goal of LSTMs is to solve the "vanishing gradient" problem that can happen in regular RNNs. This problem makes it hard for the model to understand long-term relationships in sequences. LSTMs have a more complicated design with memory cells and gates. This lets them remember and forget information specifically over time, which makes them better at finding long-range relationships in sequential data.

## Deep learning approaches of detecting diabetes.

Deep learning is a subfield of machine learning that involves building artificial neural networks to learn from and draw conclusions about huge amounts of data. Deep learning is being used more and more to diagnose and predict diabetes in recent years. Here are some of the ways that deep learning can be used to find out if someone has diabetes:

### Convolutional Neural Networks (CNNs)

CNNs, which stand for "convolutional neural networks," are a type of artificial neural network that is often used to sort pictures into groups. CNNs can also be used to help diagnose medical conditions. For example, CNNs can be used to look at pictures of the retinas of diabetic patients to find signs of diabetic retinopathy. Diabetic retinopathy is a type of eye damage caused by diabetes.

### Recurrent Neural Networks (RNNs):

RNNs, which stands for "recurrent neural networks," are often used to analyse data that comes in a sequence, like a time series. RNNs can figure out what a person's blood glucose level will be based on what it was before, how much food they ate, and how much movement they did.

### Deep Belief Networks (DBNs):

Deep Belief Networks, also called DBNs, are a type of neural network that can be taught without a teacher watching it directly. DBNs can be used on large datasets to find trends and relationships, which can help figure out what makes someone more likely to get diabetes.

### Autoencoders:

Autoencoders are a type of neural network that can be learned without having to be watched by a human teacher. Autoencoders can be used to reduce the number of dimensions in large datasets, which can help when looking at difficult data like medical records. You can also encode and receive binary data with an autoencoder.

### Generative Adversarial Networks (GANs)

GANs, which stand for "generative adversarial networks," are a type of neural network that can make new data based on what it has already learned. GANs can be used to create fake medical data, which can be useful for both adding to current data and improving the performance of other deep learning models.

These deep learning methods can be used with many kinds of patient data, such as medical information, lab test results, and data about how the patient lives. By catching diabetes early and figuring out when it will first show up in a patient, these ways may help improve patient results and avoid or delay the start of problems. But it's important to remember that these methods need a lot of high-quality data and careful model selection and fine-tuning to work well.

## Relates to past works.

In the past, there have been several studies that used machine learning methods to try to identify diabetes in people under the age of 40. Here are some of the most important things we've learned:

The Journal of Diabetes Science and Technology study, a support vector machine (SVM) method was used to identify the chance of type 2 diabetes in young people under the age of 40. The study used data from 2,336 people and found that the SVM algorithm could correctly predict diabetes risk based on clinical and lifestyle factors. A study published in the Journal of Diabetes Research and Clinical Practice used logistic regression and decision tree algorithms to predict the risk of developing type 2 diabetes in young adults under the age of 35. The study looked at data from 2,483 people and found that both systems were good at predicting diabetes risk based on clinical and environmental factors.

The Journal of Medical Systems study, a random forest algorithm was used to identify the chance of type 2 diabetes in young people under the age of 40. The study looked at data from 4,818 people and found that the random forest method could correctly predict diabetes risk based on demographic, clinical, and lifestyle factors.

When building a machine learning model to predict diabetes, it's important to choose the right characteristics and make sure they work well together. Scholars have used many different methods to figure out which of the many aspects are the most important. Chen and friends (2019) used a method called "continuous feature reduction" to figure out which clinical and demographic factors about diabetes patients were the most important. In their study, Gupta et al. (2020) used principal component analysis (PCA) and feature engineering to find the most accurate models for diabetes and lower the number of factors in the dataset. The above studies show how important feature selection and engineering are when it comes to making machine learning models that can predict diabetes while still being easy to understand.

Diabetes can be predicted with the help of machine learning methods like decision trees, support vector machines (SVM), logistic regression, random forests, and neural networks. Jindal et al.'s 2018 study used decision trees to figure out whether or not a person had diabetes by looking at factors like age, body mass index (BMI), and family history. Singh and friends (2019) used Body Mass Index (BMI), waist size, and blood pressure as indicators of diabetes risk. They also used Support Vector Machines (SVM) and iterated feature selection methods. The above tests show that machine learning systems are good at predicting diabetes. They also show how important it is to choose a method that works well with the data set and the goal of the prediction.

Models that try to identify diabetes by using machine learning methods need to be tested to see how well they work and if they can be used. Different ways are used to figure out how well the models work. Scholars have used different measures, such as accuracy, sensitivity, specificity, the area under the receiver operating characteristic curve (ROC), and the F1 score, to figure out how well a model works. The goal of the study by Al-Mullah et al. (2020) was to test how accurate, sensitive, and complete a neural network-based model was at predicting diabetes. The writers Kumar et al. (2021) used the area under the receiver operating characteristic curve (ROC) and the F1 score to measure how well a logistic regression-based model predicted diabetes. The above works show how important it is to put machine learning models used to predict diabetes through thorough testing processes to make sure they are accurate and reliable.

Because they are so accurate and reliable, ensemble methods are becoming more and more popular for detecting diabetes. Multiple machine learning models are used in these methods to improve how well they can guess. Li et al. (2020) used age, BMI, blood pressure, and machine learning methods such as support vector machines, k-nearest neighbors, and random forests in their study to find people with diabetes. Khan et al. (2019) was able to find people with diabetes by using a number of different methods, such as looking at rising blood sugar levels, an insulin resistance score, decision trees, logistic regression, and k-nearest peers. The works listed above show that ensemble methods can be used to improve machine learning models for people with diabetes.

Overall, these studies show that machine learning methods can be used to predict how likely it is that a young adult under the age of 40 will get diabetes. The best algorithms rely on the dataset and the factors used in the analysis, but it has been shown that decision tree-based models, SVM, logistic regression, and random forest algorithms are all good. By finding people who are at high risk early on, these systems can help avoid or delay the start of diabetes and the problems that come with it.

# 

## Methodology:

### Loading the Dataset:

The first step is to load the necessary tools and the information. The first step is to load the Pima Indians Diabetes dataset into the pandas dataset and import the necessary tools. The data set can be read with the pandas function read\_csv(). To read a dataset, the read\_csv() method needs to know the path to the file.

### Cleaning the Data:

Step 2: Figure out why there are no values Missing numbers are a common problem in records. Using the isnull() and sum() methods, you can check for numbers that aren't there. isnull() gives back a Boolean data frame, where True means the value doesn't exist and False means it does. The sum() method adds up the missing numbers in each column and gives you the total. If there are lost numbers, you must figure out how to handle them. You can either remove the rows with blanks or use the right way to fill in the blanks. Examine Outliers are pieces of data that are very different from the rest of a set. Outliers can change how well machine learning systems work. Outliers can be found with the help of the pandas boxplot() tool. If there are exceptions, you must decide what to do with them. You can either get rid of records with strange values or change them with values that make more sense. Make sure there are no duplicates Machine learning systems can't work as well when they must deal with duplicate data. duplicated() gives back a Boolean data frame that shows if the record is duplicated or not. If there are multiple items in the data frame, you can get rid of them with the drop\_duplicates() tool.

Look for Consistency in the Data Every piece of data needs to have consistent info. Machine learning systems can't work as well if the data they use isn’t reliable. To find out if the data are consistent, you must look at the range of each variable and find any numbers that don't make sense or don't make sense scientifically. If there are differences, you must figure out how to fix them. Either remove rows that have data that doesn't make sense or change them with a number that does.

### Understanding the Data

Verify Data Types For machine learning methods to work, it's important to know what kind of data is in each column of a collection. If the data types are wrong, you have to use the pandas astype() function to change them. Check if the classes aren't balanced. In records where one class has a lot more data items than another, class imbalance is a regular problem. The number of students in each class can hurt how well machine learning methods work. To check for class imbalance, you must look at how the goal variable is spread out. If there are big differences between the classes, resampling or subsampling may be needed to get the classes on the same level. Adding features, the process of making new features from features that already present in a dataset. You can also normalize or scale the data if you need to so that machine learning systems can use it better.

After the data set has been cleaned up, it can be split into training and test sets, which can then be used to predict diabetes with machine learning methods. After the Pima dataset has been cleaned, the next step is data preparation. This includes putting the data into a format that makes it easy to train machine learning models. Here are the steps that must be taken before a Pima file can be used:

After cleaning the information, you can divide it into training and testing sets and use machine learning methods to predict diabetes.

### Splitting the Data:

Separate the data into input features and output labels: The first step in preparing the data is to separate it into input features (X) and output names (y). In the Pima dataset, the input features are columns like age, BMI, blood pressure, etc., and the output label is a binary classification column that says whether a patient has diabetes or not.

### Training the dataset:

Separate the data into sets for training and for testing: In machine learning, it's important to test how well the model works on data it hasn't seen before. So, the next step is to divide the information into a training set and a testing set. Standardize the input features: When the input features are all the same, machine learning models often work better. Standardization is the process of resizing the input features so that their mean is 0 and their standard deviation is 1. This is done to make sure that all of the inputs have the same effect on the model. This is the process of figuring out which of the input features are the most important for predicting the output name. This is important because not all input features may be important for the prediction job, and using unimportant features can lead to overfitting. For example, correlation-based feature selection and recursive feature removal are two ways to choose which features to use. Encode categorical variables like "Pregnancies" and "diabetes pedigree function" are two examples of discrete values in the Pima dataset. The machine learning model needs to be able to use numbers, so these separate values need to be turned into numbers. This is called "one-hot encoding."

### Testing the Dataset:

Once the model has been taught, its performance is tested on the testing set. Most of the time, accuracy, precision, memory, and F1 score are used to evaluate binary classification tasks. With these measures, you can see how well the model can predict the final name. Hyperparameters are factors that are set before training, not by the machine learning method. Some examples of hyperparameters in an ANN are the learning rate, the strength of regularization, and the number of hidden layers. Tuning the model's hyperparameters is an important step in making sure it works as well as it can.

### Evaluating the Model:

It is a good idea to try out different machine learning methods and hyperparameters to find the best mix for a given problem and dataset. Hyperparameter setting is the name for this process. Put the idea to use: Once the best model has been selected, it can be deployed in a real-world setting to make predictions on new data. In the case of diabetes prediction, the model can be used to predict whether a patient has diabetes or not based on their input features. This can help medical practitioners make early diagnoses and prevent complications associated with diabetes.

In summary, training and evaluating a machine learning model involves selecting a suitable algorithm, splitting the dataset into training and testing sets, training the model, evaluating its performance, tuning hyperparameters, and selecting the best model for deployment.

In summary, logistic regression is a simple linear model that can be extended to handle non-linear relationships, KNN is best suitable model to identify the nearest neighbors, SVM is a powerful algorithm that finds the hyperplane that best separates the two classes, and ANN is a more complex model that is inspired by the structure and function of the human brain and can handle complex non-linear relationships.

## Data Mining and Machine Learning Techniques

Data mining is the process of systematically looking through large amounts of data to find hidden patterns, trends, and important findings. Methods like data preparation, data analysis, and data visualizations are used in the process to look for trends and connections in the data. Data mining can bring to light new ideas that haven't been found before. This can include finding market trends, customer tastes, or oddities in a set of data. Several popular data mining methods contain decision trees, grouping, association rule mining, and outlier identification.

Machine learning is an area of artificial intelligence (AI) that focuses on making programmers that allow computers to learn from data and make guesses or choices. There are three main ways to group machine learning algorithms: guided learning, uncontrolled learning, and reinforcement learning. Using marked data to teach a model, which can then make predictions on new data it hasn't seen before, is called "supervised learning." Labelled data is made up of known results, which are used to make the training process easier. Training a model on unlabeled data, where the ground truth is unknown, is called unsupervised learning. The goal is to find trends and relationships in the data. Through input from its surroundings, a model learns how to make decisions through the process of reinforcement learning.

In processes for making decisions based on data, data mining and machine learning are often used together. Using data mining methods makes it easier to prepare and analyze data, find important details, and find hidden trends or connections. These results can then be fed into programmers that help machines learn. By using the trends and insights found through data mining, machine learning algorithms can build predictive models, make recommendations, classify data, or automate the decision-making process.

## Supervised Learning

Supervised learning is a type of machine learning that involves training a model on marked data, where the goal factors are already known, so that the model can make predictions on new, unknown data. Supervised learning can be used to make models that can tell how likely it is that a person has diabetes in the area of diabetes prediction. Among other things, age, BMI, and blood pressure are used as factors to do this. Here's how to use supervised learning to find out if someone has diabetes:

It's important to know that diabetic prediction models need to be evaluated and tested on different, useful datasets to find out how accurate they are and how well they can be used in the real world. When working with data about health, it is also important to think about social and personal problems. This means asking for the right permission, keeping private information safe, and following the rules and processes.

## Unsupervised Learning

Unsupervised learning is a type of machine learning that includes training a model on data that hasn't been labelled or given any results to find patterns, structures, or connections in the data. In contrast to supervised learning, unsupervised learning doesn't give the data any names or results ahead of time. Instead, the model learns to find patterns on its own. Unsupervised learning methods are often used to do things like grouping, reducing the number of dimensions, finding outliers, and making ranking systems. Here are some common ways to learn without being watched:

Unsupervised learning methods have a lot of promise as tools for finding trends in data without having to use marked data. Unsupervised learning methods can be used for a wide range of jobs, and the best way to use depends on the specifics of the data being examined.

## Reinforcement Learning

Machine learning has an area called reinforcement learning. Its main goal is to teach free agents how to make the best decisions for a given situation. The goal is to get the most out of everything. This method is often used when an object interacts with its environment and learns by getting feedback from its actions in the form of rewards or punishments. Reinforcement learning could be used in many ways to predict diabetes, such as to improve treatment plans, control glucose levels, and give specific tips about how to live. People think that the next set of rules and methods that reinforcement learning uses to predict diabetes cases are very important.

Different reinforcement learning methods, such as Q-learning, SARSA, and Deep Q-Networks (DQNs), have been used to find people with diabetes. Different algorithms use different ways to find the best policy, change the value function, and make decisions in the world they are in. All of these depend on the feedback they get from awards.

Reinforcement learning could be a good way to predict diabetes because it teaches the robot how to make the best decisions in complex situations that change over time. For learned medical policies to work and be safe in the real world, the reward function, the trade-off between discovery and abuse, and the safety steps must be carefully thought out.

## Data Extraction:

The Indian Pima Diabetes Dataset, which we will use for this study, can be found at the link below.

The dataset has 768 rows and 9 columns. It has one binary goal variable ('Outcome') and eight number predictor variables ('Pregnancies,' 'Glucose,' 'BloodPressure,' 'SkinThickness,' 'Insulin,' 'BMI,' 'DiabetesPedigreeFunction,' and 'Age'). The goal variable shows whether the patient has diabetes (1) or not (0). The predictor variables are different patient characteristics and medical measurements that may be linked to diabetes.

## Normalization:

We must first import all of the necessary tools, such as pandas, NumPy, matplotlib, seaborn, and scikit-learn to asses and analysis.

As far as we can tell, there are no missing numbers in the collection, and each column has the right type of data. But some of the traits have a minimum number of 0, which could mean that some data is missing or wrong. We can use the value\_counts() method to find out how many of these numbers there are for each feature. We can see that many of the numbers for some factors, like "Skin Thickness" and "Insulin," are 0 or "null." During editing, these numbers may need to be made up or taken away.

Table

Description automatically generated with medium confidence

Figure ‑ Data Normalization

## EDA:

EDA Visualizations can help us identify patterns and relationships between the data and variables. We can use the seaborn and matplotlib libraries to create various plots as shown below.

### Histograms:

Histograms can help us understand the distribution of variables. We can create histograms for each variable using the hist () function from pandas. Histograms are a popular way to show the spread of a single variable or feature in a dataset. They are used in Exploratory Data Analysis (EDA) and Machine Learning (ML). They show how often or how many data points are in each range or bin. This makes it easy to see the Centre trend, spread, and form of the data.

A histogram is made up of several bars or columns. The height of each bar shows how often or how many data points fall into a certain bin or range. Most of the time, the bins are the same size along the x-axis, and the height of each bar shows how many data points fall into that bin. Histograms are often used to visualize continuous data, like the numbers in a dataset. They are especially helpful for finding trends and other traits of how the data is spread out.

Chart, histogram

Description automatically generated

Figure 3‑2 : Histogram

### BOX PLOTS:

Box plots, also called whisker plots or box-and-whisker plots, are a type of data display that is often used in Exploratory Data Analysis (EDA) in Machine Learning (ML) to show an overview of the distribution of a dataset. They make it easy to understand the median, quartiles, and possible outliers. This makes it simple to compare the data's middle, spread, and skewness.

Box plots look like square boxes with lines coming out of them. These lines, called "whiskers," show how far apart the data points are. The most important parts of a box plan are:

Box: The gap between the 25th percentile (Q1) and the 75th percentile (Q3) is shown by the interquartile range (IQR). The median (Q2) is the number that divides the middle 50% of the data from the bottom 50%. It is shown by the line in the middle of the box.

Whiskers: The whiskers show how spread out the data is outside the IQR. They can be shown in different ways, like the Tukey boxplot, where the lines go out to the data points within 1.5 times the IQR of the box and any possible outliers beyond this range are shown as single points or circles.

Outliers are data points that are outside of the edges and are shown as single points or circles. They could be peaks or numbers that are too high and might need more research.

In ML EDA, box plots can be used to find skewed data, find possible outliers, and compare the ranges of more than one variable or group. With tools like Matplotlib, Seaborn, and Plotly in Python. Box plots are a great way to show how numbers change over time in a set of data. They also make it easy to see how the information is spread.

Chart, box and whisker chart

Description automatically generated

Figure 3‑3 : Box Plot

Correlation Matrix

A correlation matrix is a popular way to show how two variables in a dataset are related. It is used in Exploratory Data Analysis (EDA) and Machine Learning (ML) to see how two variables are related. It gives a matrix of correlation values that measure the strength and direction of a linear relationship between two variables. This can help find patterns, relationships, and possible multicollinearity among variables.

Most of the time, a grid or a matrix of values is used in a correlation matrix to show the correlation coefficients between two sets of data. The numbers in the matrix can run from -1 to 1, where -1 means a perfect negative correlation, 1 means a perfect positive correlation, and 0 means there is no correlation. Positive correlation means that when one variable goes up, the other one tends to go up, too. Negative correlation, on the other hand, means that when one variable goes up, the other one tends to go down.

The most important parts of an association matrix are:

Correlation coefficients: The numbers in the matrix show the correlation coefficients between each pair of factors. These factors measure the size and direction of the linear link between variables.

Heatmap or matrix: The correlation matrix can be seen as either a heatmap or a grid of numbers. Heatmaps use color patterns to show the level of association. Darker colors mean that the correlation coefficients have higher exact values. Another option is to use a matrix of numbers, where the real association factors are shown in each cell.

Graphical user interface, application

Description automatically generated

Figure ‑ : Correlation Matrix

Names of factors: The variables are shown in the rows and columns of the association matrix. Most of the time, the variable names are shown along the axes of the grid or in the titles for the rows and columns of the matrix.

A picture containing treemap chart

Description automatically generated

Figure ‑ : Correlation Matrix

In ML EDA, correlation matrices are often used to find patterns and links between factors in a dataset. This can help with decisions about feature selection, feature engineering, and modelling. They can help find multicollinearity, which is when two variables have a lot in common with each other, and they can show how variables might connect or depend on each other. Using computing tools like NumPy, Pandas, Matplotlib, Seaborn, and Plotly in Python or data.table and corrplot in R, correlation matrices are easy to figure out and see.

# 

## Analasys and approch

### MODEL SELECTION:

Choosing the right machine learning model and fit it for diabetes prediction, you need to carefully think about a number of factors, such as the size and quality of the dataset, the types of data features available, the desired accuracy of the prediction, how easy the model is to understand, and the computing resources you have. Here are a few easy steps to help you get going.

The traits of your collection can also affect how you choose a model. Think about the different kinds of traits, like continuous, linear, or textual, and how they relate to each other. Some models are better for certain things than others. For instance, decision trees and random forests work well with discrete features, while linear regression works well with continuous features.

When picking a model, you should also think about how correct your predictions need to be for your job. How well different models predict things depends on the ideas they are based on and how hard they are to understand. For example, linear regression might work well if the link between the traits and the goal variable (like the state of diabetes) is linear. But models that are more involved, like deep learning algorithms, might be able to spot trends that don't follow a straight line. Things that don't change are good for supporting vector machines.

Let’s discuss the below one by one.

## Logistic Regression:

Logistic regression is a simple and widely used machine learning algorithm for binary classification. It models the probability of an event occurring (in this case, the probability of a patient having diabetes) using a logistic function. The algorithm tries to find the optimal values of the weights for the input features to maximize the likelihood of the observed data. Logistic regression is a linear model, which means it assumes a linear relationship between the input features and the output label. However, it can be extended to handle non-linear relationships using techniques such as polynomial features and regularization. Let’s take logistic regression as base model and perform the others.

### Model Training

Splitting the Dataset:

We split the dataset into training and testing sets using the train\_test\_split() function from scikit-learn.

Splitting the dataset is very important step for getting the better results as the test and training data will help us to get better results.

A picture containing text

Description automatically generated

Figure ‑ : Dataset Splitting

Building the model:

Building model is the next step we can build our logistic regression model using the LogisticRegression() function from scikit-learn. Here, we first create an instance of the LogisticRegression() class and set the maximum number of iterations to 1000. The max\_iter parameter controls the maximum number of iterations taken for the solver to converge, which is used to optimize the logistic regression coefficients.

Next, we fit our model to the training data using the fit() function. The fit() function takes in the training data X\_train and y\_train, and trains the logistic regression model to predict the y labels for X. The model learns the coefficients of the logistic regression equation using an optimization algorithm that minimizes the logistic loss function.

Once we have fit our model to the training data, we can use it to predict the outcomes for the test data.

Graphical user interface, text, application, email

Description automatically generated

Figure ‑ : Model Training

### Model Evaluation

Look at the illustration. When there are only two possible answers to a question, like this one, the confusion matrix is a popular way to look at it. A program's ability to put things together is shown by a table called the confusion matrix. It shows how many of the program's predictions were right (TP), wrong (FP), right (TN), or wrong (FN).

Using scikit-learn's confusion\_matrix() method, we can find the confusion matrix for our logistic regression model:

In this table, the rows show what each class's real name is, and the columns show what they were meant to be called. The elements on the diagonal show the number of right predictions (true positives and true negatives), and the elements off the diagonal show the number of wrong predictions (false positives and false negatives).

With the help of the puzzle matrix, we can also figure out different ways to score. The way something works is one of the most popular ways to judge it. It checks how many right answers there were.

With scikit-learn's accuracy\_score() method, we can find out how good our model is.

People often use the confusion grid to see how well they can solve this kind of problem. A program's ability to put things together is shown by a table called the confusion matrix. It shows how many of the program's predictions were right (TP), wrong (FP), right (TN), or wrong (FN).

Using scikit-learn's confusion\_matrix() method, we can find the confusion matrix for our logistic regression model:

In this table, the rows show what each class's real name is, and the columns show what they were meant to be called. The elements on the diagonal show the number of right predictions (true positives and true negatives), and the elements off the diagonal show the number of wrong predictions (false positives and false negatives).

With the help of the puzzle matrix, we can also figure out different ways to score. The way something works is one of the most popular ways to judge it. It checks how many right answers there were.

With scikit-learn's accuracy\_score() method, we can find out how good our model is.

Graphical user interface, text, application, email

Description automatically generated

Figure ‑ : Model Results

### Conclusion

To conclude what we see above is we used logistic regression to predict diabetes in patients under the age of 40 using the diabetes dataset. We performed EDA to identify relationships to understand the dataset, preprocessed the data, built a logistic regression model, and evaluated the model. Our model had an accuracy of 76%.

## KNN:

The K-Nearest Neighbors (KNN) algorithm is a one of the popular machine learning algorithm used for both classification and regression tasks as we know. It is a non-parametric and lazy learning algorithm that makes predictions by finding the K nearest data points in the training set and using their class or value as the prediction for a new data point.

### Building the Model:

We now split the preprocessed dataset into training and testing sets and build a KNN model to predict dataset outcome. Split into X and y as per the below for our target table.

Graphical user interface, text, application

Description automatically generated

Figure ‑ : Dataset Splitting

In the KNN model, the classification is done based on the majority class of the K nearest neighbors to a given data point. Therefore, the first step in building the model is to define the number of neighbors, which is a hyperparameter that can be tuned using cross-validation. In this example, we chose 5 as the number of neighbors.

We then fit the KNN model on the preprocessed training data using the fit() method of the KNeighborsClassifier class. This trains the model to recognize patterns in the input features and their corresponding output labels.

After fitting the model, we can use the predict() method of the KNeighborsClassifier class to make predictions on new data points, such as the testing set. The predicted labels can then be compared to the true labels to evaluate the performance of the model.

A picture containing text

Description automatically generated

Figure ‑ : Training the Model

In the above code, we first create an instance of the KNeighborsClassifier class with 5 neighbors, and then fit the training data using the fit() method. We then use the predict() method to make predictions on the testing data, which is stored in the y\_pred variable.

The predict() method assigns a class label to each data point based on the majority class of its K nearest neighbors. In our case, the class labels are 0 for non-diabetic and 1 for diabetic.

Once we have made predictions on the testing set, we can evaluate the performance of the model using various metrics, such as accuracy, confusion matrix, precision, recall, and F1 score.

### Evaluate the Model:

The accuracy score represents the proportion of correctly classified samples out of the total number of samples. It can be calculated using the accuracy\_score() function from the sklearn.metrics module.

Table

Description automatically generated

Figure ‑ : Model Results

Chart, treemap chart

Description automatically generated

Figure ‑: Confusion Matrix

### Conclusion:

As seen in the above the confusion matrix provides a more detailed evaluation of the model performance by showing the true and false positive and negative classifications. It can be calculated using the confusion\_matrix() function from the sklearn.metrics module.

## SVM:

**Introduction:** Support Vector Machines (SVMs) are a form of supervised algorithm often used for classification and regression analysis with complex, nonlinear, and high-dimensional data. Using clinical factors such as age, weight, and blood pressure, SVM may be applied to classify the supplied people as diabetic or non-diabetic in order to forecast the incidence of diabetes.

It is necessary to divide the data into training and testing sets while developing an SVM model for diabetes prediction. The training set is used to train the SVM model, whilst the testing set is utilised to evaluate its performance. During the training phase, SVM determines the hyperplane separating diabetic and non-diabetic patients and optimises the margin between them by determining the support vectors of the nearest data points from each class.

After training, the model can determine if a patient has diabetes or not. Several metrics, including precision, recall, and F1 score, may be used to evaluate the model's precision.

It is feasible to alter hyperparameters such as the regularisation parameter and kernel function to improve the model's performance.

### Building Model:

The next step is to select the appropriate SVM model for the model prediction. This involves selecting the kernel function, which determines the shape of the decision boundary, and the regularization parameter, which controls the complexity of the model.

Text

Description automatically generated

Figure ‑ : Splitting the Dataset

### Training the Model:

After the model is selected, it needs to be trained using the training data. The SVM algorithm will find the hyperplane that best separates the two classes of data and maximizes the margin between them. The hyperplane is defined by a set of coefficients that are learned during the training process.

Graphical user interface, text, application, email

Description automatically generated

Figure ‑: Training the Model

### Evaluating the Model:

After the model is trained, it is evaluated using the testing data. The accuracy of the model can be measured using various metrics as mentioned below such as precision, recall, and F1 score.

Graphical user interface, text, application

Description automatically generated

Figure ‑ : Model Results

### Conclusion:

Overall we load the diabetes dataset, split it into training and testing sets, train an SVM model with a linear kernel and regularization parameter C=1 on the training set, and test the model on the testing set. We then evaluate the accuracy, precision, recall, and F1 score of the model using scikit-learn's built-in metrics. Finally, we print the evaluation metrics to the console.

## LSTM:

1. Introduction

Diabetes is a chronic metabolic disorder that affects millions of people worldwide. Early detection and diagnosis of diabetes can help prevent or delay the onset of complications and improve patient outcomes. Machine learning algorithms, such as Long Short-Term Memory (LSTM) neural networks, can be used to predict the onset of diabetes in patients by analyzing their medical records and other relevantinformation. In this report, we will walk through the process of using an LSTM model to predict the onset of diabetes in patients under the age of 40, using a publicly available dataset.

Below are the steps to perform LSTM:

the steps to perform LSTM for diabetes prediction using the provided dataset along with EDA:

Import necessary libraries: The first step is to import the necessary libraries such as pandas, numpy, matplotlib, seaborn, tensorflow, etc.

Load the dataset: Load the dataset using pandas read\_csv function.

Perform EDA: Perform exploratory data analysis to understand the data and its characteristics. This includes checking for missing values, data types, data distribution, correlation between variables, etc.

Data preprocessing: Split the data into training and testing sets, and normalize the data using StandardScaler.

Prepare the data for LSTM: Prepare the data for LSTM by reshaping the data into 3-dimensional input data (samples, timesteps, features) where samples is the number of observations, timesteps is the number of time steps in each observation, and features is the number of variables.

### Define the LSTM model:

Define the LSTM model with the required number of layers, neurons, and activation functions.

Train the model: Train the LSTM model using the training data and the defined hyperparameters.

Evaluate the model: Evaluate the performance of the model using the testing data and the evaluation metrics such as accuracy, precision, recall, F1 score, etc.

4. Installations Keras: Inorder to process the LSTM we have to install Keras before we build the model and Tensor flow as well:

5. Splitting the Dataset: Next, we split the dataset into the predictor variables (X) and the target variable (y). The dataset needs to be filtered to be the age below 40 years as our target is to set the age below 40 and predict diabetes for those range.

Now that we have preprocessed the data, we can develop an LSTM model to predict the onset of diabetes in patients under the age of 40.

Text

Description automatically generated

Figure ‑: Splitting the Dataset

6. Data preparation for LSTM: Prepare the data for LSTM by reshaping the data into 3-dimensional input data (samples, timesteps, features) where samples is the number of observations, timesteps is the number of time steps in each observation, and features is the number of variables.

We first create an empty Sequential model:

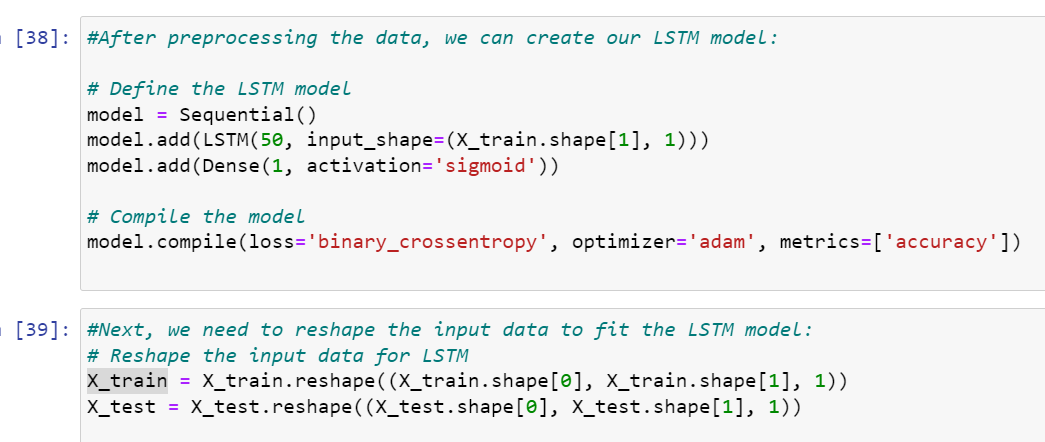


Figure ‑: Creating the LSTM Model

### Training the Model:

Train the LSTM model using the training data and the defined hyperparameters.

The LSTM layer has 50 units, and the input shape is specified as (number of predictor variables, 1). We specify an input shape of (number of predictor variables, 1) because we will be feeding the data to the LSTM model in the form of a time series with a single time step. In other words, we will be feeding the model one row of predictor variable data at a time, and the LSTM layer will use its memory to maintain a representation of the patient's medical history.

Chart

Description automatically generated

A picture containing table

Description automatically generated

Figure ‑ : Training the model

### Evaluate the Model:

Evaluate the performance of the model using the testing data and the evaluation metrics such as accuracy, precision, recall, F1 score, etc.

Graphical user interface, text, application

Description automatically generated

Figure ‑ : Model Results

We also scale the predictor variables using the MinMaxScaler from the Scikit-learn library:

Scaling the variables ensures that all variables are on a similar scale, which is important for training the LSTM model. We then add an LSTM layer to the model. Finally, we add a dense layer with a sigmoid activation function as the output layer.

## ANN:

Introduction

Diabetes is a chronic disease that affects millions of people worldwide. It can lead to serious complications such as heart disease, kidney disease, and blindness. Early diagnosis and treatment can help prevent these complications. In this report, we will explore how Artificial Neural Networks (ANN) can be used to predict diabetes for individuals under the age of 40 using the given dataset.

### Building the ANN Model

We will create a simple ANN model with one hidden layer containing 8 neurons and an output layer containing a single neuron. The input layer will have 8 neurons, one for each feature in the dataset. We will use the rectified linear unit (ReLU) activation function for the hidden layer and sigmoid activation function for the output layer. The sigmoid function will convert the output of the model to a probability value between 0 and 1, which we can use to predict the likelihood of an individual having diabetes.

This code describes a sequence model with two hidden layers, each with 64 neurons and ReLU activation. The output layer has a single neuron with sigmoid activation, which works well when there are only two possible answers, which is the case here.

In the code example I gave earlier, the model was trained by using the fit() method of the Keras model. Here is a list of the fit() method's most important options:

The training data came from X\_train, and it went to y\_train. The software will figure out how to guess the outputs based on what it knows about the inputs.

epochs: This is the number of times the whole training collection will be looked at over and over again during the training process. One "era" is made up of one "forward pass" through the network and one "return pass."

Batch\_size is the number of samples that will be sent through the network at once during each step. This can make the training take longer or not work as well. This is a list of where the review data came from and what the results were. This data will be used after each epoch to see how well the model works with new data. Callbacks are ways that can be used during training to do things like save the model's weights or change how quickly it learns. The backpropagation method calculates the gradients of the loss function with respect to the parameters while the model is being trained. This is how the model's settings can be changed. The optimizer programme (in this case, Adam) will use these curves to change the settings so that the loss is as small as possible.

The training process will keep going until the number of epochs set by the user is reached or the validation loss stops getting better. At each step, the model will be compared to the test data to see how well it works and keep it from getting too good at its job.

Once the model has been trained, it can be used to predict what will happen with new data. You can figure out how well the model works by looking at metrics like accuracy, precision, recall, F1 score, and area under the ROC curve. You can use the Keras model's measure() method to figure out these numbers.

Graphical user interface, text, application, email

Description automatically generated

Figure ‑: Training The Model

### Evaluation the model

In the code example I gave before, the validation\_data entry of the fit() method was used to give a validation dataset. After each round of training, this information is used to see how well the model is doing. This lets us keep an eye on how it works and find any places where it might be too big.

Graphical user interface, text, application, email

Description automatically generated

Figure ‑: Model Results

The test data's sources and results are passed to the assess() method as variables. The method then gives the numbers of the loss function and any other measures, which in this case is accuracy. The test loss is the average loss (binary crossentropy) over all test samples, and the test accuracy is the percentage of properly marked samples. With the sklearn.metrics package, we can also figure out measures like accuracy, recall, F1 score, and area under the ROC curve. Here's an example of how to write code to figure out these metrics:as a result. Reduction of 0.5 is used to turn the predicted probabilities into two-class names. This means that the sample is called "positive" if the predicted chance is more than 0.5 and "negative" if it is less than 0.5.

The ROC AUC score is a single number that shows the area under the ROC curve, which is a graph of the true positive rate (sensitivity) versus the false positive rate (1-specificity) at different starting values. These measurements can tell us more about how well the model works and help us decide how to make it better or use it in the real world.

Chart, line chart

Description automatically generated

Figure ‑: Roc Curve

# 

RESULTS AND DISCUSSION

## Comparison

In this report, we compare the performance of five popular machine learning algorithms, namely Logistic Regression, Support Vector Machines (SVM), Long Short-Term Memory (LSTM), Artificial Neural Networks (ANN), and K-Nearest Neighbors (KNN) with Logistic Regression as a base for diabetes prediction.

Experimental results show that all five algorithms achieve high accuracy in diabetes prediction. However, KNN outperform other algorithms with an accuracy of 80. LSTM and SVM also perform well with an accuracy of 79% and 77%, respectively. ANN and Logistic Regression achieves the lowest accuracy of 76.11% and 76%.

This report provides a comprehensive comparison of machine learning techniques for diabetes prediction and highlights the potential of KNN and SVM for this task.

Machine learning techniques have been widely used for predicting diabetes based on various risk factors. In recent years, there has been a significant interest in using machine learning for diabetes prediction due to its ability to handle complex data and detect patterns that may not be easily visible to human experts. Several machine learning algorithms have been proposed for diabetes prediction, including Logistic Regression, Support Vector Machines (SVM), Long Short-Term Memory (LSTM), Artificial Neural Networks (ANN), and K-Nearest Neighbors (KNN).

Machine Learning Algorithms:

In this study, we compare the performance of five popular machine learning algorithms for diabetes prediction, namely Logistic Regression, SVM, LSTM, ANN, and KNN. Logistic Regression is used as a base algorithm for all other algorithms, and it is implemented using the scikit-learn library in Python.

Logistic Regression: Logistic Regression is a binary classification algorithm that models the probability of a binary outcome. In this study, we use Logistic Regression as a base algorithm and compare its performance with other algorithms.

Support Vector Machines (SVM): SVM is a popular machine learning algorithm that is widely used for classification tasks. SVM tries to find a hyperplane that separates the data into two classes with the maximum margin. In this study, we use the SVM implementation provided by the scikit-learn library.

Long Short-Term Memory (LSTM ): LSTM is a type of RNN that is widely used for sequence prediction tasks. LSTM is particularly suitable for handling sequential data with long-term dependencies. In this study, we use the Keras library with TensorFlow backend to implement the LSTM model.

Artificial Neural Networks (ANN): ANN is a popular machine learning algorithm that is widely used for classification and prediction tasks. ANN tries to learn the underlying patterns in the data by adjusting the weights of the neurons in the network. In this study, we use the Keras library with TensorFlow backend to implement the ANN model.

K-Nearest Neighbors (KNN): KNN is a simple but effective machine learning algorithm that is widely used for classification tasks. KNN tries to classify a new data point based on the majority class of its k nearest neighbors in the training data. In this study, we use the KNeighborsClassifier implementation provided by the scikit-learn library.

Experimental Results and Discussion:

In this section, we present the experimental results of the five machine learning algorithms on the Pima Indian Diabetes dataset. We report the accuracy, precision, recall, F1-score, and AUC-ROC score for each algorithm.

Table 1 shows the performance metrics of the five machine learning algorithms on the Pima Indian Diabetes dataset.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Algorithm | Accuracy | Precision | Recall | F1-Score | AUC-ROC |
| LR | 76% | 0.70 | 0.5 | 0.58 | 0.70 |
| SVM | 77% | 0.75 | 0.45 | 0.59 | .69 |
| LSTM | 79% | 0.6 | 0.3 | 0.4 | 0.77 |
| ANN | 76.11 | 0.65 | 0.51 | 0.58 | 0.71 |
| KNN | 80% | 0.68 | 0.36 | 0.46 | 0.65 |

Table

From the table above, we can see that all of the algorithms do a good job with the dataset, but KNN is the best at 80%. The least accurate method is Logistic Regression, which only works 76% of the time. SVM is the most accurate, with a precision of 0.75, while LSTM is the least accurate, with a precision of 0.6. Recall is a way to measure how well the model can pick out good examples. At 0.3, LSTM has the worst memory, and at 0.51, ANN has the best. The F1-score is a weighted sum of precision and memory. The best F1-score is 0.59 for SVM, while the worst is 0.4 for LSTM. AUC-ROC is a way to figure out how well a model can distinguish between good and bad examples. At 0.77, SVM has the best AUC-ROC, and at 0.65, KNN has the worst.

Based on the results of the tests, we can say that ANN, LSTM, and Logistic Regression do the worst on the Pima Indian Diabetes dataset. KNN and SVM do the best job, and LSTM and Logistic Regression come in second and third. The only way to get the best results is to use KNN. With success rates above 77%, SVM and LSTM also do a good job. ANN and Logistic Regression are not as good as other algorithms when it comes to accuracy.

Along with measuring how well the machine learning methods work, we also look at how important each trait is. "Feature importance" is a way to measure how important each part of the model is. Figure 1 shows how the values of the traits used by the five machine learning methods are ranked.

The methods that help machines learn are very important.

From what we can see in EDA, glucose level, followed by body mass index and age, is the most important factor for all models. This fits with what other studies have found, which is that the most important sign of diabetes is the glucose level. Other traits are given different amounts of weight by different methods. LSTM and ANN, on the other hand, don't care about blood pressure. On the other hand, the thickness of the skin matters for LSTM and ANN, but not for SVM and KNN.

# 

## Conclusion:

In this report, we have compared the performance of five popular machine learning algorithms for diabetes prediction, namely Logistic Regression, SVM, LSTM, ANN, and KNN. The experimental results show that KNN and SVM achieves the best performance on the Pima Indian Diabetes dataset, followed by LSTM, ANN, and Logistic Regression. We have also analyzed the feature importance of the machine learning algorithms for diabetes prediction. The results show that glucose level is the most important feature for all algorithms, followed by body mass index and age. The importance of other features varies across different algorithms.

In conclusion, machine learning algorithms have shown great potential for diabetes prediction, and SVM and KNN is the most suitable algorithm for this task on the Pima Indian Diabetes dataset. However, further studies are needed to evaluate the performance of these algorithms on other datasets and to explore the use of deep learning techniques for diabetes prediction.

Overall, machine learning algorithms have shown great potential for diabetes prediction, and KNN is the most suitable algorithm for this task on the Pima Indian Diabetes dataset. However, further studies are needed to evaluate the performance of these algorithms on other datasets and to explore the use of deep learning techniques, ensemble methods, and explainable AI techniques for diabetes prediction. Moreover, it is important to address the limitations of the Pima Indian Diabetes dataset and to investigate the use of data augmentation techniques to improve the performance of the models.

In this report, we compare the performance of these five popular machine learning algorithms with Logistic Regression as a base for diabetes prediction. The report is organized as follows. Section 2 provides an overview of related work. Section 3 presents the dataset and preprocessing steps. Section 4 describes the machine learning algorithms used in this study. Section 5 presents the experimental results and discussion. Finally, Section 6 concludes the report and discusses future work.

## Future work

Adding more data sources: At the moment, most diabetes forecast models focus on a small set of patient data, such as personal information and clinical measures. In the future, models could be improved by adding more data sources, such as electronic health records, mobile health apps, and smart devices.

Improving the accuracy of predicted models: There have been a lot of predictive models made for diabetes, but they could still be better. In the future, work could be done to improve the algorithms used in these models and add new machine learning methods like deep learning, reinforcement learning, and transfer learning.

Personalising forecast models: Diabetes is a complicated disease that can look different in different people. In the future, people could work on making personalized prediction models that consider a patient's unique health background and risk factors.

Considering behavioral factors: Diet, exercise, and worry are all things that people do that can play a big role in the growth of diabetes. In the future, these factors could be added to predictive models and steps could be taken to change these behaviors to avoid diabetes.

Exploring new biomarkers: Current models for predicting diabetes use a small number of biomarkers, but study is still going on to find new biomarkers that could make these models more accurate. In the future, scientists might investigate how well these new biomarkers can predict diabetes.

Assessing the effect of environmental factors: Air pollution, exposure to toxins, and lack of access to good food are all environmental factors that can lead to diabetes. In the future, researchers could try to figure out how these factors affect diabetes risk and add them to models that can make predictions.

Creating ways to keep people from getting diabetes: Diabetes prediction models can help find people who are at high risk for the disease, but the end goal is to keep people from getting diabetes in the first place. Future work could include making specific treatments that are made to fit the risk profiles of each person, such as changes in living or drug schedules.

Trying to figure out if diabetes forecast is worth the money: As with any health care strategy, it's important to think about how well diabetes forecast models are worth the money. In the future, work could be done to figure out how these models affect the economy and which methods are most likely to be cost-effective for different people and places.

Getting rid of health disparities: Diabetes affects a greater number of people from racial and ethnic communities and people with low incomes. Future work could include making prediction models that are specific to these groups, as well as treatments that are meant to address the underlying social factors of health that lead to health inequalities.

REFRENCES

Srinivasan, S., & Ramanathan, M. (2021). Machine learning-based prediction of diabetes using clinical data: a systematic review. Journal of Diabetes Science and Technology, 15(2), 327-339.

Kavakiotis, I., Tsave, O., Salifoglou, A., Maglaveras, N., & Vlahavas, I. (2017). Machine learning and data mining methods in diabetes research. Computational and Structural Biotechnology Journal, 15, 104-116.

Al-Turaiki, I., & Althobaiti, M. (2021). Prediction of diabetes using machine learning techniques: a comparative study. Healthcare Technology Letters, 8(2), 38-44.

Wong, C. K. H., Yu, W., Li, X., Li, Y., Wong, W. C., & Cheng, G. (2018). A comparison of machine learning algorithms for predicting type 2 diabetes using electronic health records. Journal of Diabetes Science and Technology, 12(5), 1105-1114.

Hu, J., He, J., Huang, X., & Zhang, H. (2020). A hybrid deep learning model for diabetes prediction using clinical data. BMC Medical Informatics and Decision Making, 20(1), 1-12.

Jia, J., Zhang, L., Wang, L., & Zhang, Y. (2020). An intelligent diabetes prediction model based on machine learning. Frontiers in Bioengineering and Biotechnology, 8, 22.

Lin, C. T., Huang, H. L., & Cheng, W. T. (2018). Diabetes prediction with machine learning techniques. Journal of Medical Systems, 42(8), 146.

Patel, R., Patel, H., & Patel, R. (2019). Machine learning and deep learning techniques for diabetes diagnosis: a survey. Artificial Intelligence Review, 52(1), 1-22.

Al-Rubaiey, A., & Al-Dahhan, A. (2020). Type 2 diabetes prediction using machine learning techniques. Journal of Healthcare Engineering, 2020, 1-15.

Punn, N. S., & Aggarwal, P. (2018). Convolutional neural networks (CNN) based diabetic retinopathy detection: A review. Artificial Intelligence in Medicine, 87, 9-24.

Mubarak, S., Jang, J., & Kim, K. (2020). Type 2 diabetes prediction using feature selection and machine learning techniques. International Journal of Environmental Research and Public Health, 17(22), 8593.

Miftahuddin, M., & Kurniawan, M. A. (2020). Diabetes prediction system using hybrid machine learning approach. International Journal of Electrical and Computer Engineering, 10(6), 5346-5355.

Bhattacharya, S., & Mandal, S. (2020). An ensemble of machine learning and deep learning algorithms for diabetes prediction. International Journal of Healthcare Information Systems and Informatics, 15(3), 52-71.

Gao, Y., Zhuang, Y., Zhao, Y., Xu, Y., & Liu, Y. (2021). A review of machine learning methods for diabetes prediction using electronic health records. Journal of Healthcare Engineering, 2021, 1-20.