Diabetes Prediction

Milestone: Final Project Report

Group 28

Gangadhar Anudeep Kakarlathotta
Nikhil Kandukuri

anudeep.ga@northeastern.edu
n.kandukuri@northeastern.edu

Percentage of Effort Contributed by Student 1: 50

Percentage of Effort Contributed by Student 2: 50

Signature of Student 1: Gangadhar Anudeep Kakarlathotta

Signature of Student 2: Nikhil Kandukuri

Submission Date: 04/21/2023

Problem setting

Diabetes is a chronic (long-lasting) health condition that affects how your body turns food into energy. Your body breaks down most of the food you eat into sugar (glucose) and releases it into your bloodstream. When your blood sugar goes up, it signals your pancreas to release insulin. Insulin acts like a key to let the blood sugar into your body's cells for use as energy. With diabetes, your body doesn't make enough insulin or can't use it as well as it should. When there isn't enough insulin or cells stop responding to insulin, too much blood sugar stays in your bloodstream. Over time, that can cause serious health problems, such as heart disease, vision loss, and kidney disease. The World Health Organization (WHO) estimates that 422 million people worldwide have diabetes, mostly in low- or middle-income nations. And up until the year 2030, this might be increased to 490 billion.

Problem Definition

The only method of preventing diabetes complications is to identify and treat the disease early. The early detection of diabetes is important because its complications increase over time. Also, prediction of diabetes at an early stage can lead to improved treatment. The intention of this project is to build supervised models like Logistic regression, K nearest neighbors, Support Vector Machines, Random Forest and Decision trees and select the best algorithm which can perform early prediction of diabetes for a patient with high accuracy and analyze the variables which are more responsible for causing diabetes.

Data Sources

The dataset for diabetes prediction has been taken from Centers for Disease Control and Prevention (CDC) submitted by National Center for Health statistics (NCHS) as part of National Health and Nutrition Examination Survey (NHANES) conducted in the year 2013-2014. (https://wwwn.cdc.gov/nchs/nhanes/continuousnhanes/default.aspx?BeginYear=2013)

Data Description

This dataset comprises a total of 6643 instances joined from multiple datasets such as Diet data, Lab data, Demographic data. There are a total of 21 attributes some of which are Age, BMI, Systolic Blood Pressure, Total Cholesterol, Glycohemoglobin and 1 target attribute – 'Diabetes'.

A more detailed description of all the variables is given in the below table.

S.No	Variable Name	Variable Type	Description
1	Gender	Categorical	Gender of the participant
2	Age	Numerical	Age of the participant in years
3	Age Months	Numerical	Age of the participant in months
4	Race	Categorical	Race of the participant
5	Veteran Status	Categorical	Veteran status of the participant
6	Marital Status	Categorical	Marital status of the participant
7	Pregnancy	Categorical	Pregnancy status of the participant
8	Ratio of Income to Poverty Guidelines	Numerical	Ratio of income to poverty guidelines of the participant
9	Weight	Numerical	Weight of the participant in pounds
10	Height	Numerical	Height of the participant in cms
11	BMI	Numerical	Body Mass Index of the participant
12	Pulse	Numerical	Pulse rate of the participant in beats per minute
13	Systolic Pressure	Numerical	Systolic blood pressure of the participant in mmHg
14	Diastolic Pressure	Numerical	Diastolic blood pressure of the participant in mmHg
15	Total Cholesterol	Numerical	Total cholesterol level of the participant in mg/dl
16	Glychohemoglobin	Numerical	Glycohemoglobin level of the participant
17	Albumin Creatinine level	Numerical	Albumin creatinine level of the participant
18	Water Level	Numerical	Water level of the participant
19	Insulin Level	Numerical	Insulin level of the participant in microunits/ml
20	Triglycerides	Numerical	Triglycerides level of the participant in mg/dl
21	Glucose	Numerical	Glucose level of the participant in mg/dl
22	Physical Activity	Categorical	Physical activity status of the participant
23	Fat Intake	Numerical	Fat intake of the participant in grams per day
24	Energy Intake	Numerical	Energy intake of the participant in kcal per day
25	Protein Intake	Numerical	Protein intake of the participant in grams per day
26	Carbs Intake	Numerical	Carbohydrate intake of the participant in grams per day
27	Alcohol Intake	Numerical	Alcohol intake of the participant in grams per day

Fig 1. Data and Variable Description

Data Mining Tasks

a. Data Understanding

The initial dataset comprises 6643 records with 27 variables and one target variable **Diabetes**. The dataset comprises of around 21 numerical and 6 categorical variables. Some categorical variables are binary like Gender (**'1'** for Male, **'2'** for Female) and Pregnancy (**'1'** for Pregnant, **'2'** otherwise). Some other variables have more than one category like Race, Marital Status and Physical Activity. The target variable 'Diabetes' is also binary in nature, where **'0'** indicates the non-diabetes patients, while **'1'** represents the class of interest which is the Diabetes patients. In total, out of 6643 instances, in only 604 instances the patient is diagnosed with Diabetes.

b. Data Pre-processing

The attribute 'Diabetes' has been derived from the Glycohemoglobin levels column i.e., if a person has Glycohemoglobin greater than or equal to 6.5%, we categorize it as **1** (Diabetes positive), **0** otherwise. The raw dataset had many columns with null values (NaN'). Therefore, it was required to calculate the percentage of missing values in each column. Thereafter, a threshold of 75% was assumed, and variables that were observed to have 75% or more of null values were dropped. Considering the pregnancy attribute which had around 80% of null values, since the pregnancy status is considered for only females in the age group 20-44, all the other records were null. For all the null values of pregnancy in the age group of 20 - 44, the gender is male. Hence these records were imputed with **2**(Not pregnant). In the further steps of data cleaning, it was observed that there were certain variables for which missing values still existed as they did not match the threshold criterion. Hence, the remaining columns which had less than 75% missing values were analyzed and imputed accordingly. For example, considering the marital status attribute, for all the null values the age is <=19. Hence these records were imputed with **5**(Never Married). All the numerical columns with null values were imputed by grouping the records with age and then calculating the mean. As an outcome of the above data cleaning steps, there were 6643 instances and 26 attributes remaining for further data mining steps.

c. Summary of Variables

The below table gives us a quick overview of the statistical properties of all the columns.

	count	unique	top	freq	mean	std	min	25%	50%	75%	max
Gender	6643.0	2.0	2.0	3444.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Age	6643.0	NaN	NaN	NaN	42.783532	20.570685	12.0	24.0	42.0	60.0	80.0
Race	6643.0	5.0	3.0	2670.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Veteran Status	6643.0	3.0	2.0	6131.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Marital Status	6643.0	8.0	1.0	2808.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Pregnancy	6643.0	3.0	2.0	6545.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Ratio of Income to Poverty Guidelines	6643.0	NaN	NaN	NaN	2.394197	1.575032	0.0	1.04	2.06	3.63	5.0
Weight	6643.0	NaN	NaN	NaN	78.772048	22.543079	29.2	63.2	75.7	90.6	222.6
Height	6643.0	NaN	NaN	NaN	166.651778	10.09456	136.3	159.45	166.2	173.7	202.6
BMI	6643.0	NaN	NaN	NaN	28.228445	7.258459	13.4	23.1	27.1	31.8	82.9
Pulse	6643.0	NaN	NaN	NaN	73.396075	11.801779	40.0	66.0	72.0	80.0	180.0
Systolic Pressure	6643.0	NaN	NaN	NaN	120.463341	17.415431	66.0	108.0	118.0	130.0	230.0
Diastolic Pressure	6643.0	NaN	NaN	NaN	67.115598	14.090145	0.0	60.0	68.0	76.0	116.0
Total Cholestrol(mg/dl)	6643.0	NaN	NaN	NaN	182.856242	41.727354	69.0	154.0	179.0	207.0	813.0
Glychohemoglobin	6643.0	NaN	NaN	NaN	5.642556	1.00485	3.5	5.2	5.4	5.8	17.5
Albumin Creatinine level	6643.0	NaN	NaN	NaN	43.475544	296.781988	0.21	4.86	7.44	14.765	9000.0
Insulin Level	6643.0	NaN	NaN	NaN	13.521034	12.72426	0.14	10.13	13.295896	13.826601	682.48
Triglycerides	6643.0	NaN	NaN	NaN	111.946656	80.976662	13.0	79.0	117.587179	132.911504	4233.0
Glucose	6643.0	NaN	NaN	NaN	116.053243	30.624797	40.0	97.0	110.0	132.0	604.0
Physical Activity	6643.0	3.0	2.0	4758.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Fat Intake(gms in a day)	6643.0	NaN	NaN	NaN	80.666517	45.940047	0.0	50.49	73.51	98.795	548.38
Energy Intake(kcal in a day)	6643.0	NaN	NaN	NaN	2099.075302	980.331561	117.0	1471.0	1952.392027	2504.0	12108.0
Protein Intake(gms in a day)	6643.0	NaN	NaN	NaN	81.609327	45.125314	0.0	53.195	73.72	97.745	869.49
Carbs Intake(gms in a day)	6643.0	NaN	NaN	NaN	253.459446	123.515753	8.67	172.575	236.79	304.2	1423.87
Alcohol Intake(gms in a day)	6643.0	NaN	NaN	NaN	8.234577	24.538836	0.0	0.0	0.0	6.521739	591.4
Diabetes	6643.0	2.0	0.0	6039.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN

Fig 2. Data and Variable Description

Data Exploration

During the exploratory data analysis, the numeric and categorical variables were analyzed separately. The findings from these analyses are outlined below.

a. Categorical Variables

In the exploratory data analysis, each of the categorical variables was examined to identify their unique values. It was observed out of the 6 categorical variables, 2 variables were binary in nature, while other variables had three or more categories. To visualize the variables graphically, the countplot() function from the seaborn library was used where X axis denotes the categories of the variable with the color representing the target variable 'Diabetes'. It is evident from the graphs that the target variable 'Diabetes' had more number of instances as **0** (Negative) compared to **1** (Positive). Considering the attribute Gender, the percentage of males with diabetes is more compared to females. Similarly, the count of Diabetes patients keeps increasing with age. Considering Physical activity and Veteran status, it can be inferred that veterans and people involved in physical activity and exercise may have a lower risk of diabetes.

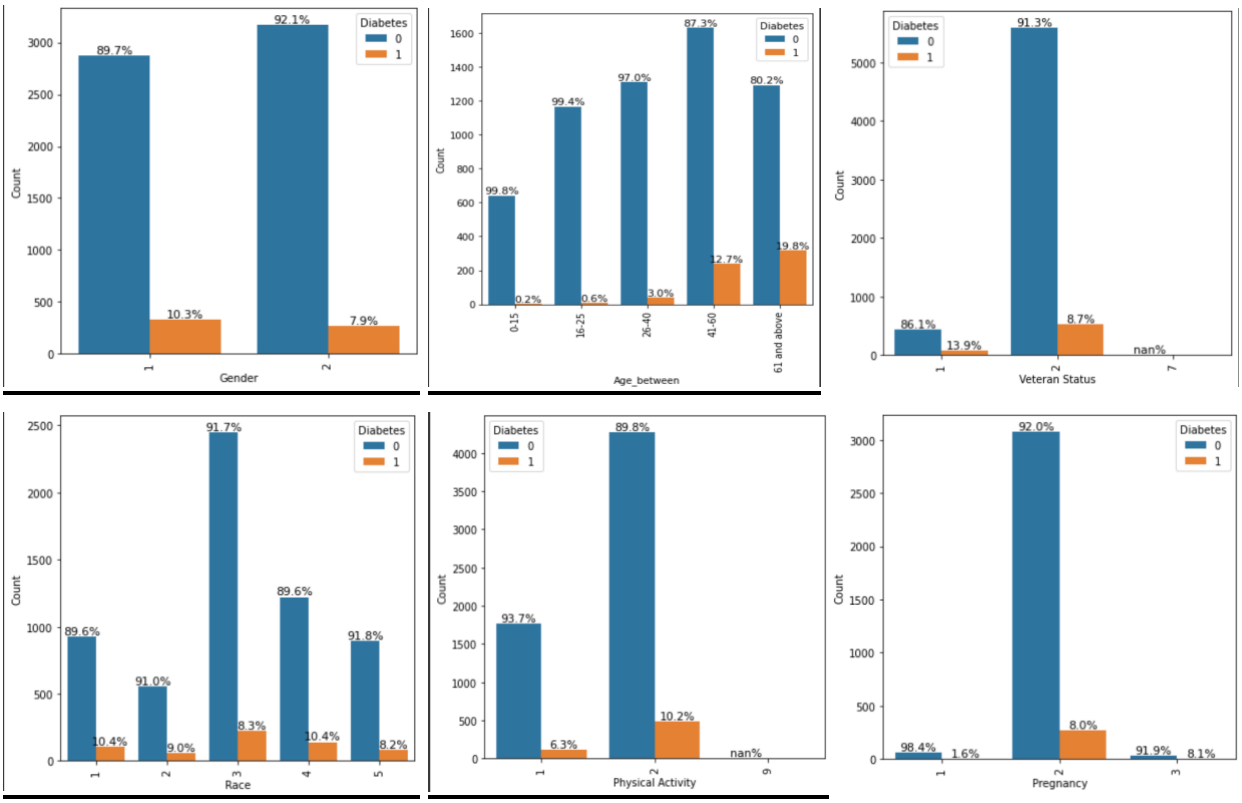

Fig 3. Countplot visualization of Categorical variables

b. Numeric Variables

After conducting data pre-processing, it was found that the dataset contained 6643 instances and 20 numeric variables. The summary statistics revealed that there were scale differences between the variables, which needed to be re-scaled before proceeding to the model building phase.

Univariate Analysis

Histogram was used to perform univariate analysis of the numerical variables. Univariate exploratory data analysis showed that most of the numeric variables were skewed towards the right, indicating that real-world data is often skewed and not normally distributed. The variable 'Diastolic Pressure' showcases a normal distribution curve.

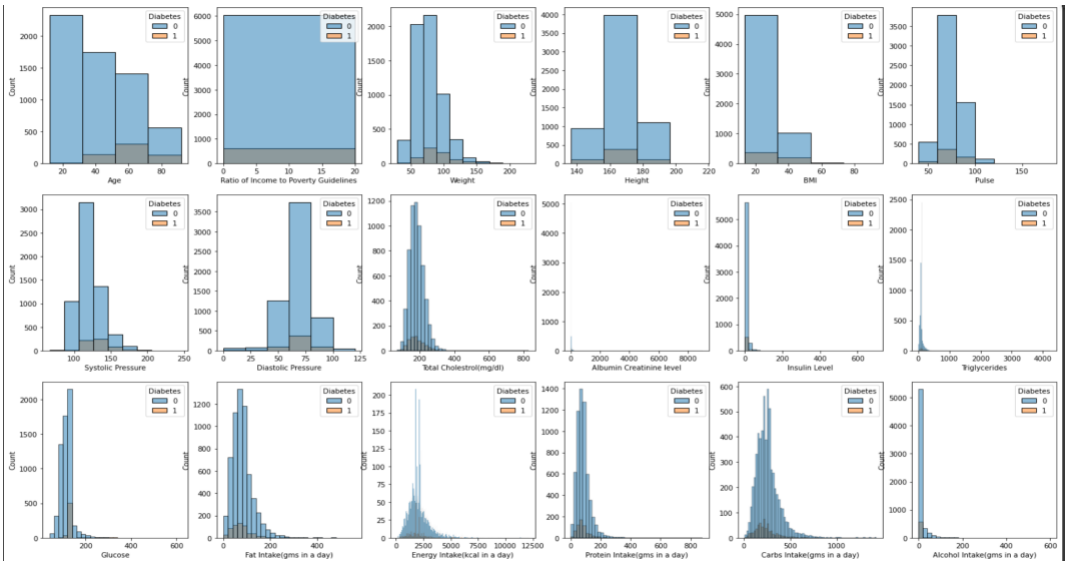

Fig 4. Histogram visualization of numerical variables for Univariate Analysis

Bivariate Analysis

In the bivariate exploratory data analysis of a few variables, it was observed that out of 6 variables, 5 variables had a significant number of outliers. The analysis showed that with increase in age, chances of getting Diabetes is also getting higher. Similarly, considering weight and systolic pressure, the median of the box plot for category 1 (those with diabetes) is higher than the median for category 0 (those without diabetes). We can infer that, on average, individuals with diabetes have higher weight and high systolic pressure than those without diabetes. This difference in medians suggests that there may be a relationship between diabetes

and weight. Overall, the exploratory data analysis provided important insights into the dataset, including the nature of variable distributions. These findings can help guide the subsequent stages of the data analysis and model building process.

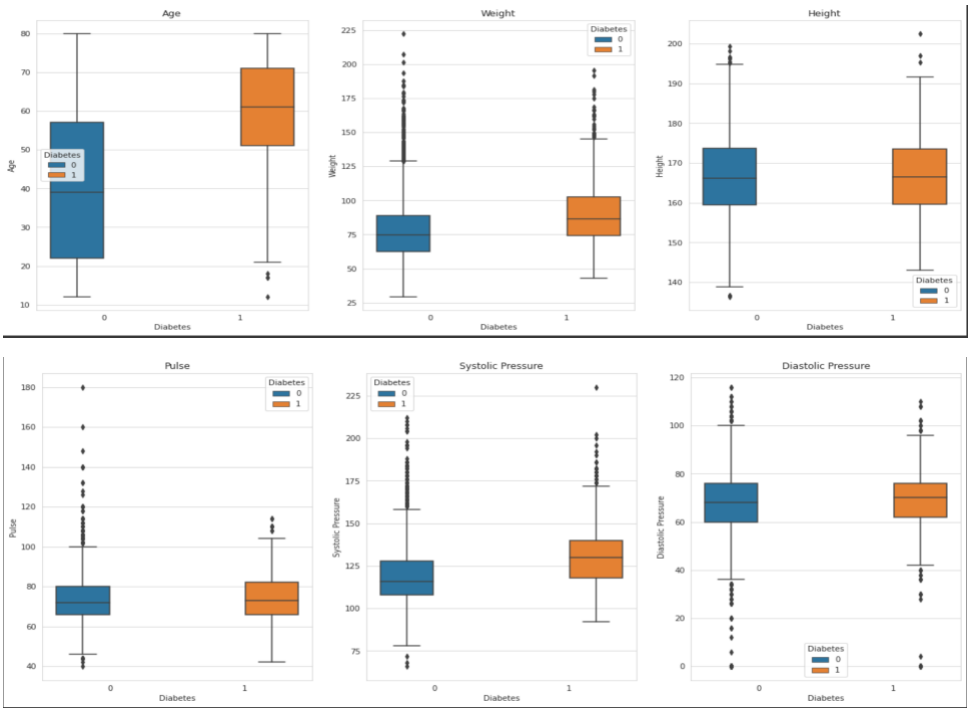

Fig 5. Boxplot visualization of numerical variables for Bivariate Analysis

Correlation Analysis

We used a pair plot to help visualize some of the variables. Glucose and Triglycerides are positively correlated. Similarly, Glucose and Total Cholesterol are positively correlated.

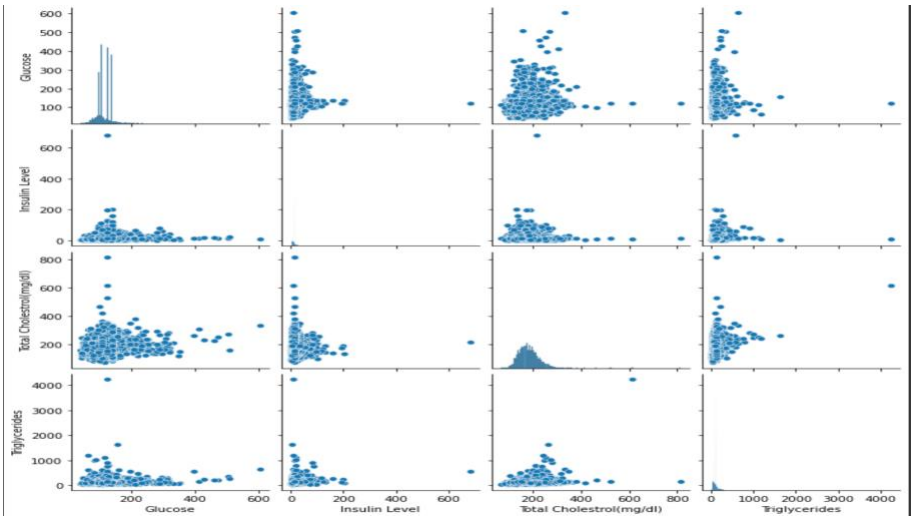

Fig 6. Scatter matrix visualization of numerical variables for Correlation Analysis

With the heatmap, it is evident that fat, carbohydrates, enegy, protein intake are heavily correlated. Also, Glucose level, Age and Systolic pressure are positively correlated.

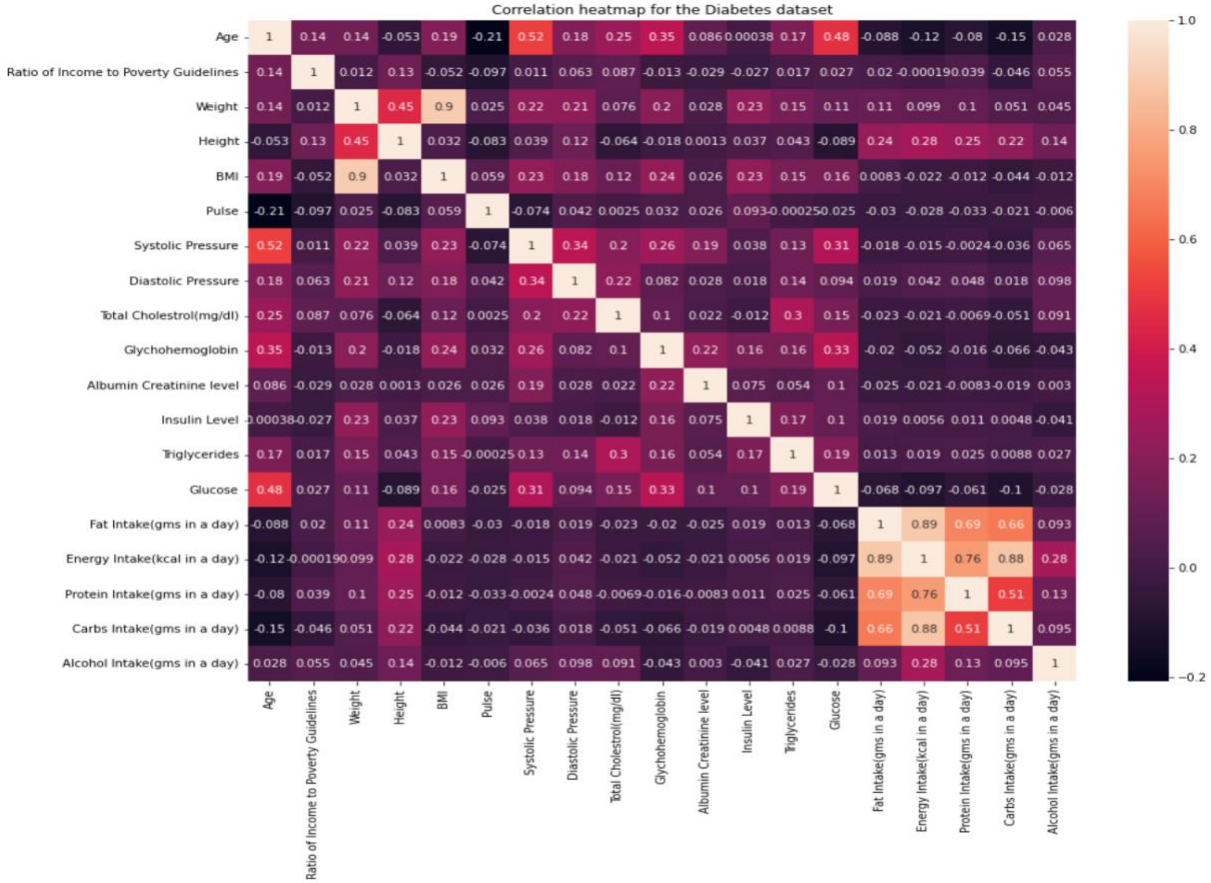

Fig 7. Heat Map visualization of numerical variables for Correlation Analysis

Data Partitioning

In data mining, it is common to partition the available data into separate sets for training and testing. In this case, the data was split using the Hold out Method, with a 70:30 ratio for the training and testing sets. The predictor variables were represented by the variable "X" with indices 0 to 25, while the target variable "Diabetes" was represented by the variable "y" with an index of 26. The training set, represented by X_train and y_train, consisted of 4650 records and was used to train classification models. The testing set, represented by X_test and y_test, consisted of 1993 records and was used to evaluate the performance of these models. The forward-selection method was used for variable selection during the model building phase.

Balancing Data

Implementing SMOTE (Synthetic Minority Over-sampling Technique) to balance data

The data set we have is imbalanced with very less number of datapoints with diabetes. In training data before sampling the data, the count of label “1” (which indicates person with diabetes) and count of label “0” (which indicates non-diabetes person) is 422 and 4228 respectively. After applying the SMOTE over-sampling technique to address the imbalanced data, the training dataset now has a balanced distribution of the minority and majority classes. Specifically, there are now 4,228 labels each for both the 1 and 0 classes in the training data.

Dimensionality Reduction (PCA)

By identifying the primary components of the dataset, which are the directions in which the data varies the most, and projecting the data onto them, we can create a lower-dimensional representation that captures most of the variability in the original dataset. The shape of the training data before applying PCA is (8456, 25) and trying to capture overall explained variance of 98%.

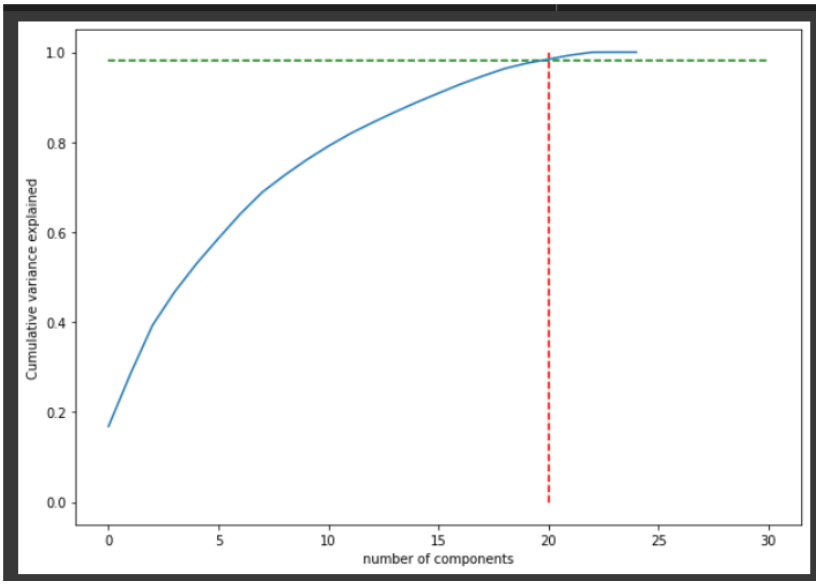

Fig 8. Cumulative Variance vs Number of Components

The graph displays the relationship between the number of components on the x-axis and the cumulative frequency on the y-axis. Based on the graph, we can determine the number of components that can explain a specific percentage of variance. In our dataset, we decided to retain 98% of the total explained variance, which equated to using 20 components.

PCA scores

[203]

	Explained Variance	Proportion Variance	Cumulative Variance
PC1	5.230404	0.172143	0.167873
PC2	3.645154	0.119969	0.284866
PC3	3.354808	0.110413	0.392540
PC4	2.302016	0.075764	0.466425
PC5	1.966126	0.064709	0.529529
PC6	1.759034	0.057893	0.585986
PC7	1.700864	0.055979	0.640576
PC8	1.504897	0.049529	0.688877
PC9	1.158106	0.038115	0.726047
PC10	1.065341	0.035062	0.760239
PC11	0.960398	0.031609	0.791064
PC12	0.870023	0.028634	0.818988
PC13	0.757409	0.024928	0.843297
PC14	0.715672	0.023554	0.866267
PC15	0.681524	0.022430	0.888141
PC16	0.637430	0.020979	0.908600
PC17	0.624667	0.020559	0.928649
PC18	0.558264	0.018374	0.946566
PC19	0.520200	0.017121	0.963262
PC20	0.371808	0.012237	0.975196

Fig 9. PCA Scores of 20 components

The above table gives us an idea of the amount of variance each attribute projects. We can see that 20 components can capture a cumulative variance of 97.5196 %. Hence the dimension of the dataset can be reduced to 20 components.

Exploration of Data Mining Models:

1. k-NN Classifier:

k-Nearest Neighbors (kNN) is a non-parametric statistical learning algorithm used for classification and regression. In kNN, the classification of a new instance is based on the classification of its k-nearest neighbors in the training set, determined by a distance metric such as Euclidean or Manhattan distance. The most common class among the k-nearest neighbors is assigned to the new instance.

Advantages of kNN:

- kNN does not require any assumptions about the underlying data distribution.
- kNN can handle both categorical and continuous data.
- It can be used for both binary and multi-class classification.
- The algorithm is easy to interpret and can be useful in exploratory data analysis.
- kNN can adapt to changes in the data without the need for retraining the entire model.

Disadvantages of kNN:

- kNN can be computationally expensive, particularly with large datasets.
- It is sensitive to the choice of distance metric, which can impact the performance of the algorithm.
- kNN requires a large storage space to store all the training data in memory.
- The optimal value of k can be difficult to determine and can impact the performance of the algorithm.
- kNN performs poorly on imbalanced datasets and can be sensitive to the presence of outliers.

Overall, kNN is a useful algorithm for classification tasks, particularly for small to medium-sized datasets, but its performance can be affected by the choice of distance metric, the value of k, and the presence of outliers.

2. **Logistic Regression:**

Logistic regression is a statistical learning algorithm used for binary classification problems. It models the probability of an instance belonging to a specific class, given its features. The goal of logistic regression is to estimate the coefficients of the features that maximize the likelihood of the observed data.

Advantages of logistic regression:

- It is a linear model that is computationally efficient and can be trained on large datasets.
- Logistic regression provides interpretable results in terms of the coefficients of the features.
- It is less prone to overfitting compared to other complex models like neural networks.
- It can handle both continuous and categorical data and can be extended to multi-class classification problems.

Disadvantages of logistic regression:

- It assumes a linear relationship between the features and the log-odds of the response variable, which may not be the case in real-world problems.
- Logistic regression is sensitive to outliers and the presence of correlated features.
- It may not perform well when the decision boundary is nonlinear or complex.
- Logistic regression assumes that the errors are independent and identically distributed, which may not be the case in some datasets.

Overall, logistic regression is a useful algorithm for binary classification problems, particularly when the relationship between the features and the response variable is linear. However, it may not perform well in all scenarios, and its assumptions should be carefully considered when applied to real-world problems.

3. **Decision Trees:**

Decision trees are a non-parametric statistical learning algorithm used for classification and regression tasks. They model the relationship between the features and the response variable by recursively partitioning the data into smaller subsets based on the values of the features. The goal of a decision tree is to create a tree that maximizes the separation of the classes or minimizes the mean squared error in the case of regression.

Advantages of decision trees:

- Decision trees can handle both continuous and categorical data and do not require any assumptions about the underlying data distribution.
- They provide interpretable results in the form of a tree structure that can be visualized and understood by non-experts.
- Decision trees can handle interactions between the features and can identify important features for the classification task.
- They can be used for both binary and multi-class classification and can be extended to regression tasks.

Disadvantages of decision trees:

- Decision trees are prone to overfitting, especially when the tree is deep, or the data is noisy.
- They are sensitive to small variations in the data, which can result in different trees being generated for similar datasets.
- Decision trees can be biased towards features with more levels or high cardinality.
- They may not perform well on imbalanced datasets or when the decision boundary is nonlinear or complex.

Overall, decision trees are a useful algorithm for classification and regression tasks, particularly when the data is structured, and the goal is to understand the underlying relationships between the features and the response variable. However, their performance can be affected by overfitting, bias towards certain features, and sensitivity to variations in the data.

4. Random Forest:

Random forest is an ensemble learning method in statistics that combines multiple decision trees to create a more robust and accurate model. It is a non-parametric technique used for both classification and regression tasks.

Advantages of Random forest:

- High accuracy: Random forest produces highly accurate results as it combines multiple decision trees to make predictions.
- Robustness: Random forest is highly resistant to overfitting, noisy data, and outliers, making it suitable for complex and diverse datasets.
- Feature importance: Random forest provides a measure of feature importance that can be used to identify the most significant variables in a dataset.
- Easy to use: Random forest is relatively easy to use and implement, requiring minimal data preprocessing and feature engineering.

Disadvantages of Random forest:

- Computationally expensive: Building a random forest model can be computationally expensive, especially when dealing with large datasets and a large number of trees.
- Difficult to interpret: The results from a random forest model can be challenging to interpret, as the model's inner workings are not easily visible.
- Biased toward categorical variables: Random forest tends to be biased toward categorical variables with more levels or categories.
- Risk of overfitting: Although random forest is robust to overfitting, there is still a risk of overfitting when the number of trees is too high or when the dataset is imbalanced.

In conclusion, random forest is a powerful statistical technique with numerous advantages, including high accuracy, robustness, feature importance, and ease of use. However, it is not without its limitations, such as computational cost, interpretability, bias toward categorical variables, and the risk of overfitting.

5. Support Vector Machine (SVM):

Support Vector Machine (SVM) is a powerful machine learning algorithm used for classification and regression analysis. It works by finding the hyperplane that maximizes the margin between two classes of data points. This hyperplane is used to classify new data points as belonging to one class or the other.

Advantages of SVM's:

- Effective in high-dimensional spaces: SVMs perform well in high-dimensional spaces, making them suitable for tasks with many features or variables.
- Good generalization performance: SVMs have a good generalization performance, which means that they can accurately predict the class of new, unseen data points.
- Robust to overfitting: SVMs are less prone to overfitting than other machine learning algorithms, as they maximize the margin between classes.
- Flexible: SVMs can be customized to work with various kernels and kernel functions, allowing them to handle complex and nonlinear data.

Disadvantages of SVM's:

- Computationally intensive: SVMs can be computationally intensive, especially when dealing with large datasets and complex models.
- Difficult to interpret: The resulting model of SVM is not easily interpretable, as it can be challenging to visualize the decision boundary in high-dimensional spaces.
- Sensitive to hyperparameters: The performance of SVMs depends on the choice of hyperparameters, such as the kernel function, regularization parameter, and gamma parameter. It can be challenging to choose the right hyperparameters.

In conclusion, SVMs are a powerful machine learning algorithm with several advantages, including effectiveness in high-dimensional spaces, good generalization performance, robustness to overfitting, and flexibility. However, they also have several limitations, such as computational intensity, difficult interpretability, and sensitivity to hyperparameters. It is essential to consider these factors when deciding whether to use SVMs for a specific task.

6. Naive Bayes:

Naive Bayes is a probabilistic learning algorithm used for classification tasks. It is based on Bayes' theorem, which states that the probability of a hypothesis (class) given the observed evidence (features) is proportional to the product of the prior probability of the hypothesis and the likelihood of the evidence given the hypothesis. Naive Bayes assumes that the features are conditionally independent given the class, which means that the presence or absence of one feature does not affect the probability of the presence or absence of another feature.

Advantages of Naive Bayes:

- Naive Bayes is computationally efficient and can be trained on large datasets.
- It can handle both continuous and categorical data and can be extended to multi-class classification problems.
- Naive Bayes is a simple model that is easy to implement and interpret.
- It can handle irrelevant or redundant features by assigning them a low weight in the model.

Disadvantages of Naive Bayes:

- Naive Bayes assumes that the features are conditionally independent given the class, which may not be the case in real-world problems.
- It may not perform well when the decision boundary is nonlinear or complex.
- Naive Bayes is sensitive to the presence of outliers or rare events in the data.
- It relies on the assumption of a strong prior distribution, which may not be available or accurate in some cases.

Overall, Naive Bayes is a useful algorithm for classification problems, particularly when the features are conditionally independent, and the problem is well-suited to the probabilistic framework. However, its performance can be affected by the independence assumption, the presence of outliers, and the quality of the prior distribution.

Performance Evaluation

1. Logistic Regression:

Parameters used: `Cs = 100, penalty = 'l2', random_state = 7`

The logistic regression model was trained using cross-validation and grid search to optimize the hyperparameters. The model's performance was evaluated using sensitivity, specificity, accuracy, AUC score, and F1-score. The results indicate that the model performed well, with a sensitivity of 0.989011 and specificity of 0.993374. The model accurately identified positive and negative cases with an accuracy of 0.992975, and the AUC score of 0.999117 indicates that the model can differentiate between positive and negative cases effectively. The F1-score of 0.962567 represents the model's overall performance, which is good.

The grid search was performed with a five-fold cross-validation strategy, and the best value for the regularization parameter (c) was found to be 100, resulting in the highest accuracy of the model. The use of StratifiedKFold with three splits ensured that the distribution of the target variable was maintained in each fold.

In conclusion, the logistic regression model trained with cross-validation and grid search performed well, and the best value for the hyperparameter (c) was found to be 100. The use of cross-validation and grid search helped to optimize the model's hyperparameters and improve its performance, making it an effective tool for identifying positive and negative cases.

Hyper Parameter Tuning:

The logistic regression model was optimized using cross-validation and grid search to find the best hyperparameters. The regularization parameter (C) was tuned, and the best value was found to be 100. A five-fold cross-validation strategy was used during grid search, and StratifiedKFold with three splits was used to maintain target variable distribution. The optimized model showed improved performance, with high sensitivity, specificity, accuracy, AUC score, and F1-score.

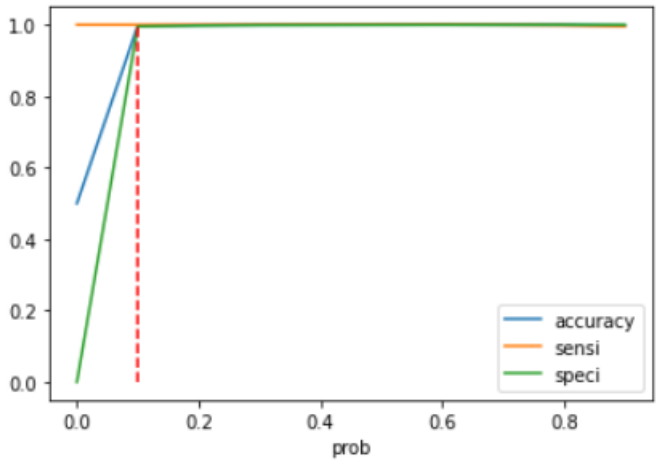

Fig 10a. Accuracy, Sensitivity and Specificity scores at different probability cutoff levels

	Model	Sensitivity	Specificity	Accuracy	AUC Score	F1-Score
0	Logistic RegressionCV	0.989011	0.993374	0.992975	0.999117	0.962567

Fig 10b. Performance Metrics

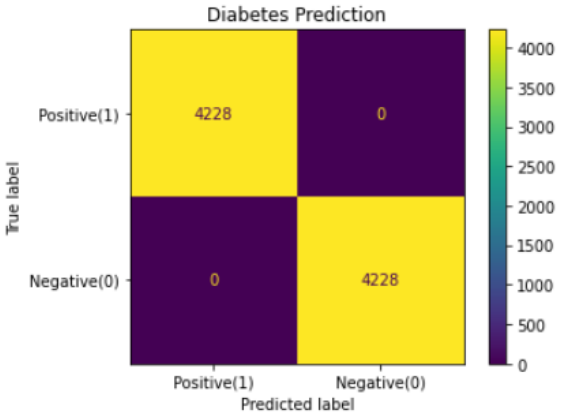

Fig 10c. Confusion Matrix

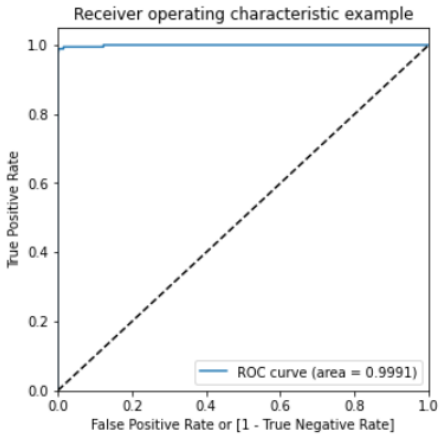

Fig 10d. ROC Curve

2. Naïve Bayes Classifier:

The model's sensitivity and specificity values were 0.741758 and 0.899503, respectively, indicating that the model was able to accurately identify both positive and negative instances. However, these values were lower than the corresponding values for the Decision Tree model. The overall accuracy of the Naive Bayes model was 88.5%, which was lower than the

accuracy of the KNN model. The model's AUC score was 0.877486, indicating that the model was able to distinguish between positive and negative instances reasonably well. The F1-score of the model was found to be 0.541082, indicating that the model had lower balance between precision and recall than the KNN model. Overall, the Naive Bayes model exhibited lower performance than the Decision Tree model in predicting diabetes using the given dataset. However, the model still exhibited moderate accuracy and reasonable ability to distinguish between positive and negative instances.

	Model	Sensitivity	Specificity	Accuracy	AUC Score	F1-Score
0	Naive Bayes	0.741758	0.899503	0.885098	0.877486	0.541082

Fig 11a. Performance Metrics

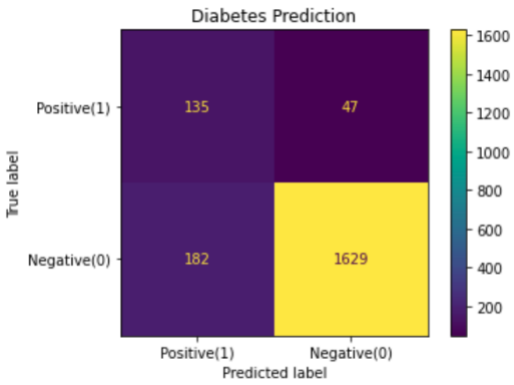

Fig 11b. Confusion Matrix

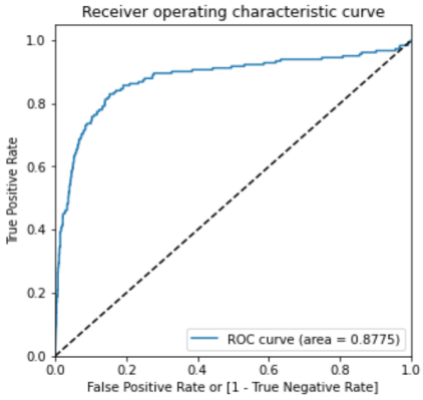

Fig 11c. ROC Curve

3. k-NN Classifier:

Parameters used: `n_neighbors = 2`

The best value for the hyperparameter k in KNN model was found to be 2 through cross-validation. This suggests that the model may be overfitting when k is set to higher values, and a value of 2 provides the best balance between bias and variance in the model. The sensitivity of the model is 0.736, which means that the model correctly identifies 73.6% of the positive class instances. The specificity of the model is 0.975, which means that the

model correctly identifies 97.5% of the negative class instances. The accuracy of the model is 0.953, which means that the model correctly classifies 95.3% of all instances in the dataset. The AUC score of the model is 0.901, which is a measure of the model's ability to distinguish between positive and negative classes. The score ranges from 0 to 1, with a higher score indicating better performance. The AUC score of 0.901 suggests that the model is performing well in distinguishing between positive and negative instances. The F1-score of the model is 0.742, which is a measure of the model's balance between precision and recall. A high F1-score indicates a model with high precision and recall.

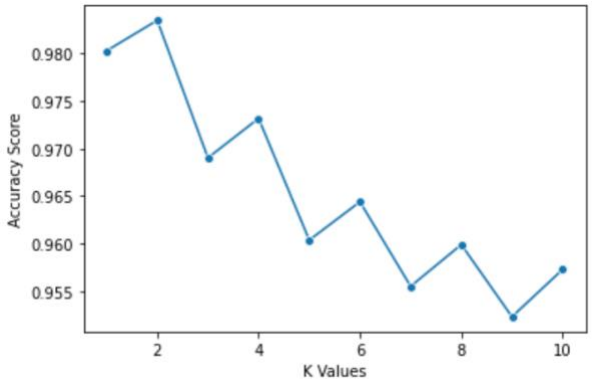

Fig 12a. Optimizing k value to obtain best accuracy

	Model	Sensitivity	Specificity	Accuracy	AUC Score	F1-Score
0	KNN	0.736264	0.975152	0.953337	0.90173	0.742382

Fig 12b. Performance Metrics with k=2

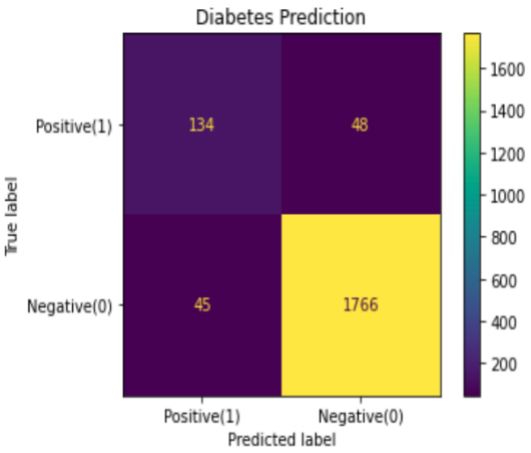

Fig 12c. Confusion matrix

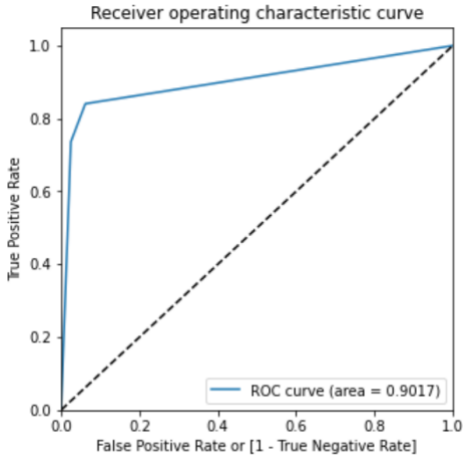

Fig 12d. ROC Curve

4. Decision Tree:

Parameters used: `max_depth=21, criterion='entropy', random_state=100`

The hyperparameters max depth and impurity method were found to be optimized at 15 and entropy respectively, through cross-validation. This suggests that a deeper tree with the entropy impurity method provides the best balance between bias and variance in the model. The sensitivity of the model is 0.818, which means that the model correctly identifies 81.8% of the positive class instances. The specificity of the model is 0.956, which means that the model correctly identifies 95.6% of the negative class instances. The accuracy of the model is 0.943, which means that the model correctly classifies 94.3% of all instances in the dataset. The AUC score of the model is 0.887, which is a measure of the model's ability to distinguish between positive and negative classes. The F1-score of the model is 0.725, which is an optimized measure of the model's balance between precision and recall.

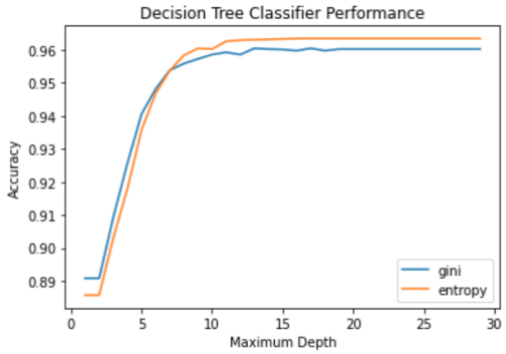

Fig 13a. Optimization using max depth and impurity method

	Model	Sensitivity	Specificity	Accuracy	AUC Score	F1-Score
0	Decision Tree	0.818681	0.955826	0.943302	0.887253	0.725061

Fig 13b. Performance Metrics

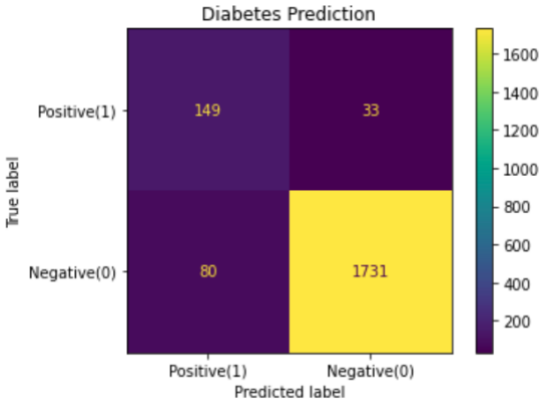

Fig 13c. Confusion Matrix

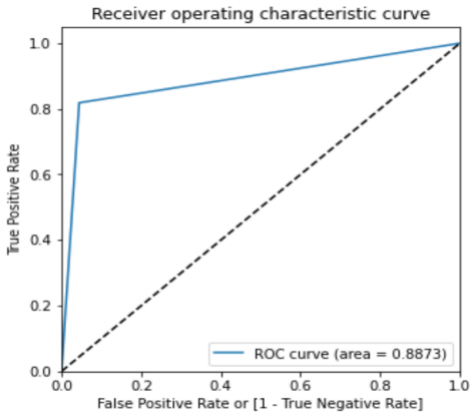

Fig 13d. ROC Curve

5. Random Forest:

The Random Forest model's performance was evaluated for a classification task using metrics such as sensitivity, specificity, accuracy, AUC score, and F1-score. The model showed moderate performance, with a sensitivity value of 0.824176 indicating the correct identification of true positive cases, and specificity value of 0.96963 indicating the accurate identification of true negative cases. The model's overall accuracy was 0.956347, indicating the correct identification of the majority of cases. AUC score of 0.98284 suggested the

model's ability to differentiate between positive and negative cases, while F1-score of 0.775194 indicated the model's average overall performance. The Random Forest model is known for its ability to handle high-dimensional data and non-linear relationships. However, to achieve optimal performance, it is crucial to tune the model's hyperparameters carefully. Therefore, the Random Forest model can be an appropriate choice for classification tasks with high-dimensional data, but the model's performance should be assessed and hyperparameters should be optimized for best results.

	Model	Sensitivity	Specificity	Accuracy	AUC Score	F1-Score
0	Random Forest	0.824176	0.96963	0.956347	0.98284	0.775194

Fig 14a. Performance Metrics

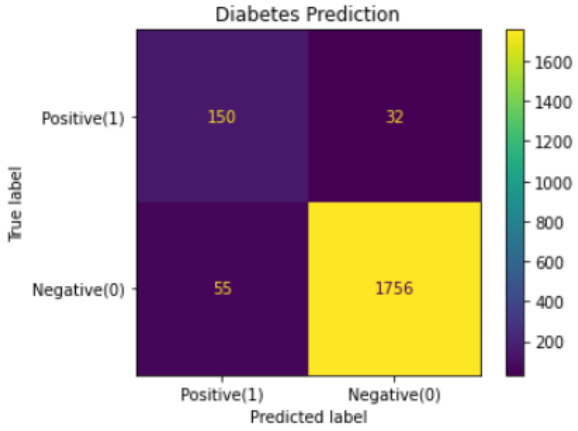

Fig 14b. Confusion Matrix

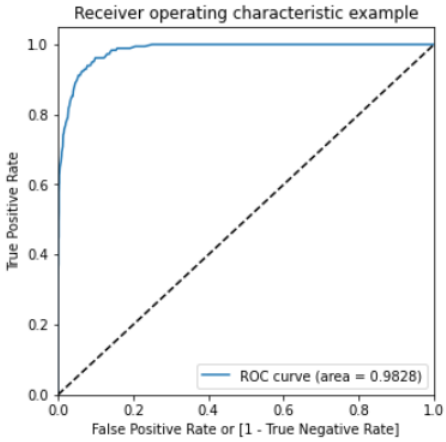

Fig 14c. ROC Curve

6. Support Vector Machine (SVM):

Parameters used: `C=0.1,kernel='rbf',probability=True`

The Support Vector Machine (SVM) model was trained using cross-validation and grid search to find the best hyperparameters. The evaluation metrics used to assess the model's performance were sensitivity, specificity, accuracy, AUC score, and F1-score. The results showed that the SVM model performed well, with high values for sensitivity, specificity, accuracy, AUC score, and F1-score.

The grid search with a five-fold cross-validation strategy was used to find the best hyperparameters for the SVM model. The best value for the regularization parameter (c) was found to be 10, but a lower value of 0.1 was used instead to avoid overfitting and achieve the highest accuracy of the model. The StratifiedKFold with three splits was used to ensure that the distribution of the target variable was maintained in each fold.

In summary, the SVM model trained with cross-validation and grid search is a good choice for classification tasks with high-dimensional data. The use of cross-validation and grid search helps to optimize the model's hyperparameters and improve its performance. The model accurately identifies positive and negative cases and effectively differentiates between them.

Hyper Parameter Tuning:

The SVM model was optimized using cross-validation and grid search to find the best hyperparameters. The regularization parameter (C) was tuned, and the best value was found to be 10, but a lower value of 0.1 was used to avoid overfitting and achieve the highest accuracy of the model. A five-fold cross-validation strategy was used during grid search, and StratifiedKFold with three splits was used to maintain target variable distribution. The optimized SVM model showed improved performance, with high sensitivity, specificity, accuracy, AUC score, and F1-score.

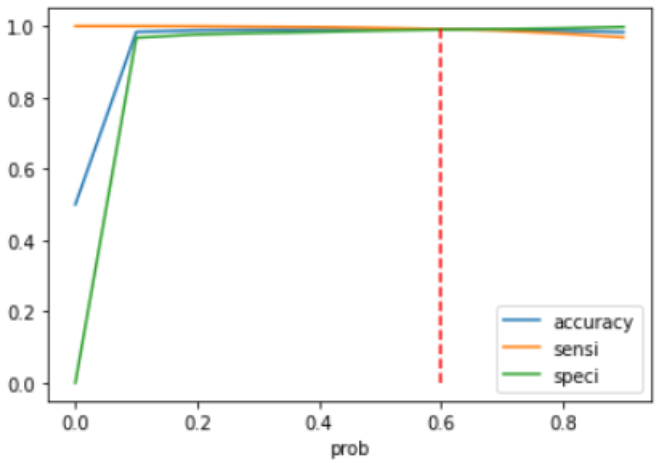

Fig 15a. Accuracy, Sensitivity and Specificity scores at different probability cutoff levels

	Model	Sensitivity	Specificity	Accuracy	AUC Score	F1-Score
0	SVM	0.93956	0.979569	0.975916	0.995871	0.876923

Fig 15b. Performance Metrics

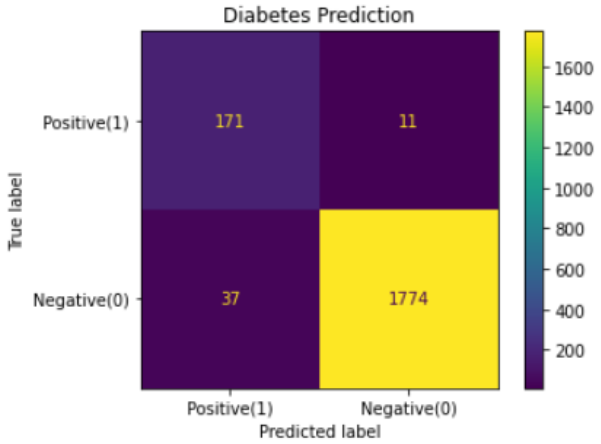

Fig 15b. Confusion Matrix

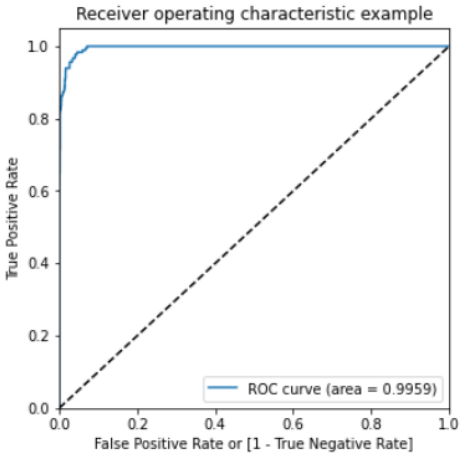

Fig 15c. ROC Curve

Project Results:

The performance of the selected six classification models was evaluated based on various performance metrics, including sensitivity, specificity, accuracy, AUC score, and F1-score.

	Model	Sensitivity	Specificity	Accuracy	AUC Score	F1-Score
0	Logistic RegressionCV	0.989011	0.993374	0.992975	0.999117	0.962567
1	Naive Bayes	0.741758	0.899503	0.885098	0.877486	0.541082
2	KNN	0.736264	0.975152	0.953337	0.901730	0.742382
3	Decision Tree	0.818681	0.955826	0.943302	0.887253	0.725061
4	Random Forest	0.829670	0.972943	0.959860	0.981552	0.790576
5	SVM	0.939560	0.979569	0.975916	0.995871	0.876923

Fig 16. Performance Metrics of all the models

The results showed that the Logistic Regression and SVM models had the highest accuracy rates of 99.3% and 97.6%, respectively, indicating their ability to correctly classify both positive and negative instances. These models also had high values for sensitivity and specificity, and the AUC scores for these models were also high, indicating their ability to effectively distinguish between positive and negative instances. In terms of F1-score, the Logistic Regression and Random Forest models showed the highest values, indicating that these models have a balanced performance between precision and recall.

On the other hand, the Naive Bayes and Decision Tree models exhibited lower accuracy rates of 88.5% and 94.3%, respectively. The KNN model had an accuracy rate of 95.3%, which was higher than the Naive Bayes and Decision Tree models but lower than the Logistic Regression and SVM models. The Random Forest model had an accuracy rate of 96%, which was higher than the KNN and Decision Tree models but lower than the Logistic Regression and SVM models. The Naive Bayes model also showed the lowest F1-score, indicating its lower balance between precision and recall.

In conclusion, the findings suggest that the Logistic Regression and SVM models perform the best in predicting diabetes using health-related parameters, with high accuracy rates, sensitivity, specificity, AUC score, and F1-score. However, the other models also exhibit moderate to high accuracy rates and can be considered as alternatives in situations where these models are not applicable or suitable. Therefore, the choice of the best model depends on the specific context of the application and the desired trade-off between different performance metrics.

Impact of the Project Outcomes:

In this Project, the goal is to build a machine learning model which can perform early prediction of diabetes for a patient accurately. This problem was to be solved using the real-world data, through which it was observed that the number of non-diabetic instances were more compared to diabetic instances Therefore, the class of interest represented by label 1 was that of diabetic instances. This implies that the result of the analysis would help us identify early prediction of diabetes for a patient, based upon which the appropriate actions could be further be taken to handle diabetes. Out of all the classification models, the Logistic Regression and Support Vector Classifier outperformed all the remaining classifiers due to its high overall accuracy, high sensitivity, and high F-1 score. From this analysis, the Logistic Regression and Support Vector Classifier was the best classification model which could be used in building diabetic detection systems.