Diabetes prediction using machine learning

Project report

Submitted by

Syed Farman Ali

Enrollment no:21045110034

Under supervision of

prof. Dr. Manzoor Ahmad

PG DEPARTMENT OF COMPUTER SCIENCES

UNIVERSITY OF KASHMIR

Hazratbal,Srinagar,190006

(2023)

Diabetes prediction using machine learning

Project report submitted in partial fulfilment of the requirements

Degree of

Master of computer applications (MCA)

by

Syed Farman Ali

Enrollment no:21045110034

Under supervision of

prof. Dr. Manzoor Ahmad

PG DEPARTMENT OF COMPUTER SCIENCES

UNIVERSITY OF KASHMIR

Hazratbal,Srinagar,190006

(2023)

cerificate

This is to certify that the project report entitled *'Diabetes Prediction Using Machine Learning*,' prepared by Syed Farman Ali (Enrollment No. 21045110034), is submitted in partial fulfilment of the requirements for the degree of Master of Computer Science at the Post Graduate Department of Computer Sciences, University of Kashmir. The project was carried out during the academic year 2021-2023. This document is a bonafide record of the work conducted under the guidance and supervision of the respective project guide.

Date:

(Guide/Supervisor) (External/Examinar) (Head of Department)

Acknowlegement

This project required technical guidance and professional supervision given its vast challenging nature and degree of complication . I would like to express my profound gratitude to prof. Dr. Manzoor Ahmad to his continuous supervision , expert guidance and feedback on subject of matter , for Providing useful inputs ,suggestions and the necessary knowledge that helped to complete this project and achieve the required target.

I would to also like to extend my gratitude to other faculty members of Post Graduate Department of Computer Sciences for their useful feedback and cooperation.

Syed Farman Ali

Abstract

The escalating global prevalence of diabetes underscores the importance of early detection for effective prevention and management. This research focuses on leveraging machine learning techniques to predict the likelihood of diabetes based on a comprehensive set of health-related features. The dataset includes lifestyle factors such as age, BMI, number of pregnancies, diabetes pedigree function, blood pressure, skin thickness, fasting glucose levels, and insulin levels.

The proposed predictive model employs advanced machine learning algorithms, including Random forests, decision trees, and support vector machines. Feature selection and hyperparameter tuning techniques are applied to optimize the model's performance. Evaluation metrics such as accuracy, sensitivity, specificity, and the area under the receiver operating characteristic curve (AUC-ROC) provide insights into the effectiveness of the predictive algorithms.

This study aims to contribute to personalized healthcare by providing a reliable and interpretable tool for early diabetes prediction. The results obtained from the model can assist healthcare professionals in identifying high-risk individuals and implementing targeted interventions. Ethical considerations, limitations, and potential future directions of the proposed approach are also discussed.

The accuracy of each algorithm is calculated and compared Based on the analysis, the proposed method outperformed all other classifiers and achieved the highest accuracy of 98.%, Precision of 100% and Recall of 98.41%. All the work is done in the Google Colab environment and on my local Pc based on python programming language and Scikit-learn library.

Contents

CHAPTER 1
INTRODUCTION

INTRODUCTION

Diabetes, a chronic metabolic disorder, is characterized by elevated blood glucose levels, resulting from inadequate insulin production or ineffective utilization. With a growing global prevalence, diabetes poses significant health challenges. Two main types, Type 1 and Type 2, differ in their etiology. Type 1 stems from an autoimmune response affecting insulin-producing cells, while Type 2 involves insulin resistance. Lifestyle factors, genetics, and environmental influences contribute to diabetes risk. Uncontrolled diabetes can lead to severe complications, emphasizing the importance of early diagnosis and effective management strategies to mitigate its impact on individuals' health and well-being.

The World Health Organization (WHO) highlights diabetes as a major public health concern, with a substantial impact on global morbidity and mortality. According to the latest WHO report, an estimated 422 million adults worldwide live with diabetes, and the numbers are rising. This chronic condition significantly contributes to cardiovascular diseases, kidney disorders, and other serious complications. The report emphasizes the urgent need for comprehensive prevention and management strategies, including public health initiatives, education, and accessible healthcare. Addressing diabetes on a global scale requires collaborative efforts to reduce its societal burden and improve the quality of life for those affected.

In India, diabetes has reached epidemic proportions, with a particularly alarming increase in recent years. According to the International Diabetes Federation (IDF), India is home to over 77 million adults living with diabetes, making it the second-largest diabetic population globally. Factors contributing to this surge include genetic predisposition, sedentary lifestyles, urbanization, and dietary changes.

The impact is profound, as diabetes significantly contributes to the country's disease burden, leading to complications such as cardiovascular diseases, neuropathy, and retinopathy. Addressing diabetes in India requires targeted interventions, public awareness campaigns, and improved access to healthcare resources, aligning with global efforts to curb the diabetes epidemic.

Machine learning plays a pivotal role in early diabetes prediction by leveraging advanced algorithms to analyze diverse datasets containing key health parameters. Through the application of sophisticated models such as random forests, decision trees, and support vector machines, machine learning can discern intricate patterns and relationships within data. This enables the identification of early indicators of diabetes risk.

The integration of features like age, BMI, number of pregnancies, diabetes pedigree function, blood pressure, skin thickness, fasting glucose, and insulin levels into predictive models enhances accuracy. Feature selection and hyperparameter tuning optimize model performance, providing healthcare professionals with valuable tools to identify high-risk individuals early on. Machine learning's ability to adapt and learn from data empowers personalized healthcare, facilitating timely interventions and improved management strategies for individuals at risk of diabetes.

CHAPTER 2
Problem statement and objectives

Problem Statement

The prevalence of diabetes is escalating globally, posing a significant health challenge. In [specific region or population], the incidence of diabetes has [increased/substantially grown] in recent years. Timely identification of individuals at risk and effective management strategies are critical to mitigating the impact of diabetes on public health. Existing approaches may lack precision in predicting diabetes onset, leading to delayed interventions and increased healthcare burdens.

Objectives

1. Develop a Predictive Model:

Develop a machine learning predictive model that utilizes a comprehensive dataset, including demographic information, medical history, and lifestyle factors, to accurately predict the likelihood of diabetes.

2. Optimize Model Performance:

Implement feature selection and hyperparameter tuning techniques to enhance the predictive performance of the model, ensuring robust and reliable results.

3. Early Identification:

Enable early identification of individuals at risk by incorporating key features such as age, BMI, number of pregnancies, diabetes pedigree function, blood pressure, skin thickness, fasting glucose, and insulin levels.

4. Evaluate Model Effectiveness:

Assess the accuracy, sensitivity, specificity, and area under the receiver operating characteristic curve (AUC-ROC) to measure the effectiveness of the predictive model in comparison to existing methods.

5.Contribute to Personalized Healthcare:

Provide a valuable tool for healthcare professionals to identify high-risk individuals early on, facilitating personalized interventions and proactive management strategies.

6.Ethical Considerations:

Address ethical considerations related to patient data privacy, informed consent, and the responsible use of machine learning in healthcare.

7.Knowledge Dissemination:

Disseminate knowledge gained from the project through publications, presentations, and collaborations, contributing to the broader scientific and healthcare community.

8.Future Directions:

Explore potential future directions, including the integration of emerging technologies, scalability of the model, and its applicability in diverse healthcare settings.

By addressing these objectives, this project aims to make a substantial contribution to the early prediction and management of diabetes, ultimately improving health outcomes and reducing the societal burden associated with this chronic condition.

CHAPTER 3
Description of the Dataset

Context

This dataset is originally from the National Institute of Diabetes and Digestive and Kidney Diseases. The objective of the dataset is to diagnostically predict whether or not a patient has diabetes, based on certain diagnostic measurements included in the dataset. Several constraints were placed on the selection of these instances from a larger database. In particular, all patients here are females at least 21 years old of Pima Indian heritage.

Content

Several constraints were placed on the selection of these instances from a larger database. In particular, all patients here are females at least 21 years old of Pima Indian heritage.

Pregnancies: Number of times pregnant

Glucose: Plasma glucose concentration a 2 hours in an oral glucose tolerance test

BloodPressure: Diastolic blood pressure (mm Hg)

SkinThickness: Triceps skin fold thickness (mm)

Insulin: 2-Hour serum insulin (mu U/ml)

BMI: Body mass index (weight in kg/(height in m)^2)

DiabetesPedigreeFunction: Diabetes pedigree function

Age: Age (years)

Outcome: Class variable (0 or 1)

Sources:

(a) Original owners: National Institute of Diabetes and Digestive and Kidney Diseases
(b) Donor of database: Vincent Sigillito (vgs@aplcen.apl.jhu.edu)
Research Center, RMI Group Leader Applied Physics Laboratory
The Johns Hopkins University Johns Hopkins Road Laurel, MD 20707
(301) 953-6231
(c) Date received: 9 May 1990.

Number of Instances: 2001

Number of Attributes: 8 plus class

For Each Attribute: (all numeric-valued)

Number of times pregnant

Plasma glucose concentration a 2 hours in an oral glucose tolerance test

Diastolic blood pressure (mm Hg)

Triceps skin fold thickness (mm)

2-Hour serum insulin (mu U/ml)

Body mass index (weight in kg/(height in m)^2)

Diabetes pedigree function

Age (years)

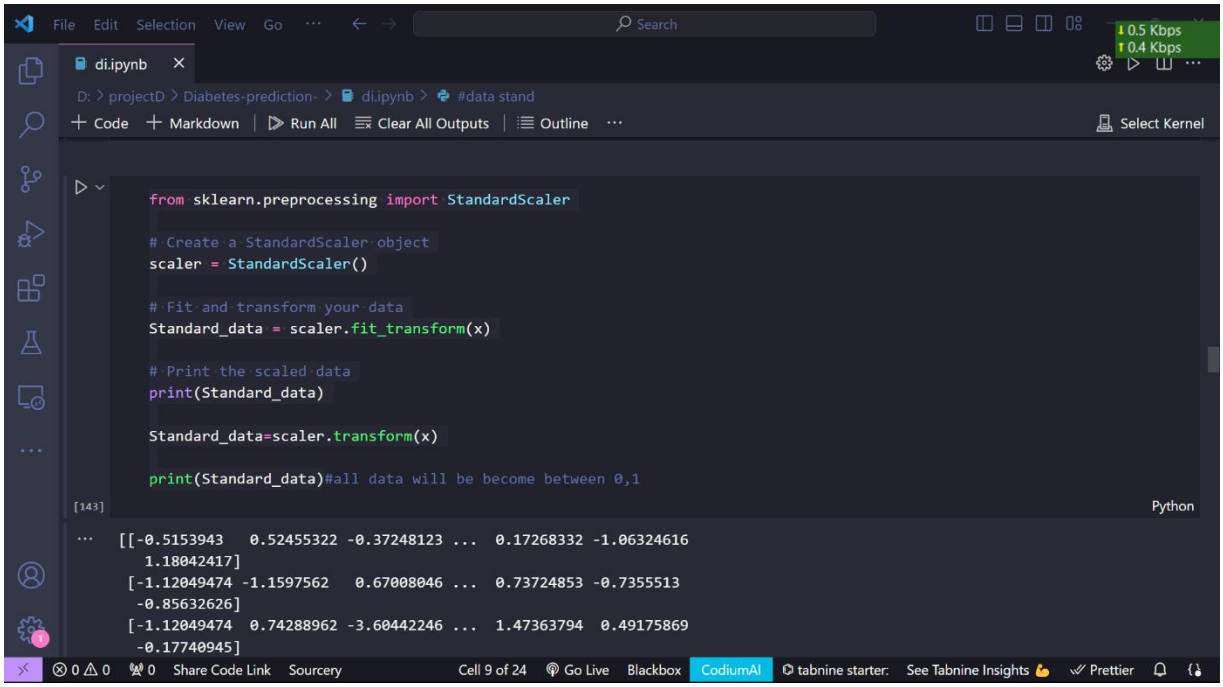
Class variable (0 or 1)

Missing Attribute Values: Yes

Class Distribution: (class value 1 is interpreted as "tested positive for diabetes")

preprocessing of the dataset

This phase of model handles inconsistent data in order to get more accurate and precise results. This dataset contains missing values. So we imputed missing values for few selected attributes like Glucose level, Blood Pressure, SkinThickness, BMI and Age because these attributes cannot have values zero. Then we scale the dataset to normalize all values.



Thus we have use StandardScaler ,

`StandardScaler` is a preprocessing technique provided by scikit-learn, specifically designed to standardize or normalize numerical features in a dataset. This process transforms the data distribution to have a mean of 0 and a standard deviation of 1. Key points about `StandardScaler` include:

1. Standardization:

`StandardScaler` standardizes features by removing the mean and scaling to unit variance. It transforms data to the Z-score distribution, facilitating comparisons and analysis across features.

2. Z-Score Normalization:

The process is akin to Z-score normalization, making it suitable for machine learning algorithms that perform better when input features are on a similar scale. This is crucial for algorithms sensitive to feature magnitudes, such as support vector machines and k-means clustering.

3. Preventing Algorithm Bias:

Many machine learning algorithms assume that features are centered and have similar magnitudes. `StandardScaler` helps prevent bias in these algorithms due to differences in feature scales, ensuring a fair representation of each feature's importance.

4. Usage in scikit-learn:

Implementation is straightforward with scikit-learn. An instance of `StandardScaler` is created, fitted to the data, and then used to transform the features. It is a crucial step in the data preprocessing pipeline before training machine learning models.

5. Applicability:

`StandardScaler` is applicable when features have different scales, preventing certain features from dominating the learning process. It is commonly applied before feeding data into algorithms like linear regression, logistic regression, or neural networks.

In summary, `StandardScaler` is a valuable tool for ensuring consistency in feature scales, promoting the stability and performance of machine learning models across various algorithms and applications.

How `StandardScaler` helps me in data preprocessing :-

1. **Normalization of Features:** Features in the diabetes dataset, such as glucose concentration, blood pressure, BMI, and others, may have different scales and units. Standardizing these features using StandardScaler ensures that they all have a mean of 0 and a standard deviation of 1, bringing them to a common scale. This is crucial for machine learning algorithms that are sensitive to the magnitude of input features.
2. **Enhanced Model Performance:** Many machine learning models, including algorithms like support vector machines and k-nearest neighbors, perform better when features are on a similar scale. Standardizing the data helps prevent certain features from dominating the learning process, leading to more accurate and reliable predictions.
3. **Improved Convergence:** Standardizing features aids in the convergence of iterative optimization algorithms, which are commonly used in machine learning models. It facilitates faster convergence and can prevent issues such as slow convergence or divergence during training.
4. **Alleviating Algorithm Bias:** StandardScaler helps prevent biases in algorithms that are sensitive to feature scales. For instance, distance-based algorithms may produce biased results if features have different magnitudes. Standardizing features mitigates such biases, providing a fair representation of the importance of each feature in the model.
5. **Facilitating Interpretability:** Standardizing features also aids in the interpretability of model coefficients. With features on a common scale, it becomes easier to compare the impact of different features on the model's predictions.

Feature Selection

The features in the dataset are very large, it leads to usage of more memory and execution time. So the important features are selected based on the correlation of features. Since there are 9 features in the dataset, the important features are extracted based on the Pearson's correlation of features with each other. The Pearson correlation coefficient (r) is the most common way of measuring a linear correlation. It is a number between-1 and +1 that measures the strength and direction of the relationship between two variables. The formula for calculating the Pearson's Coefficient r is given below.

$$r = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2 \sum (y_i - \bar{y})^2}}$$

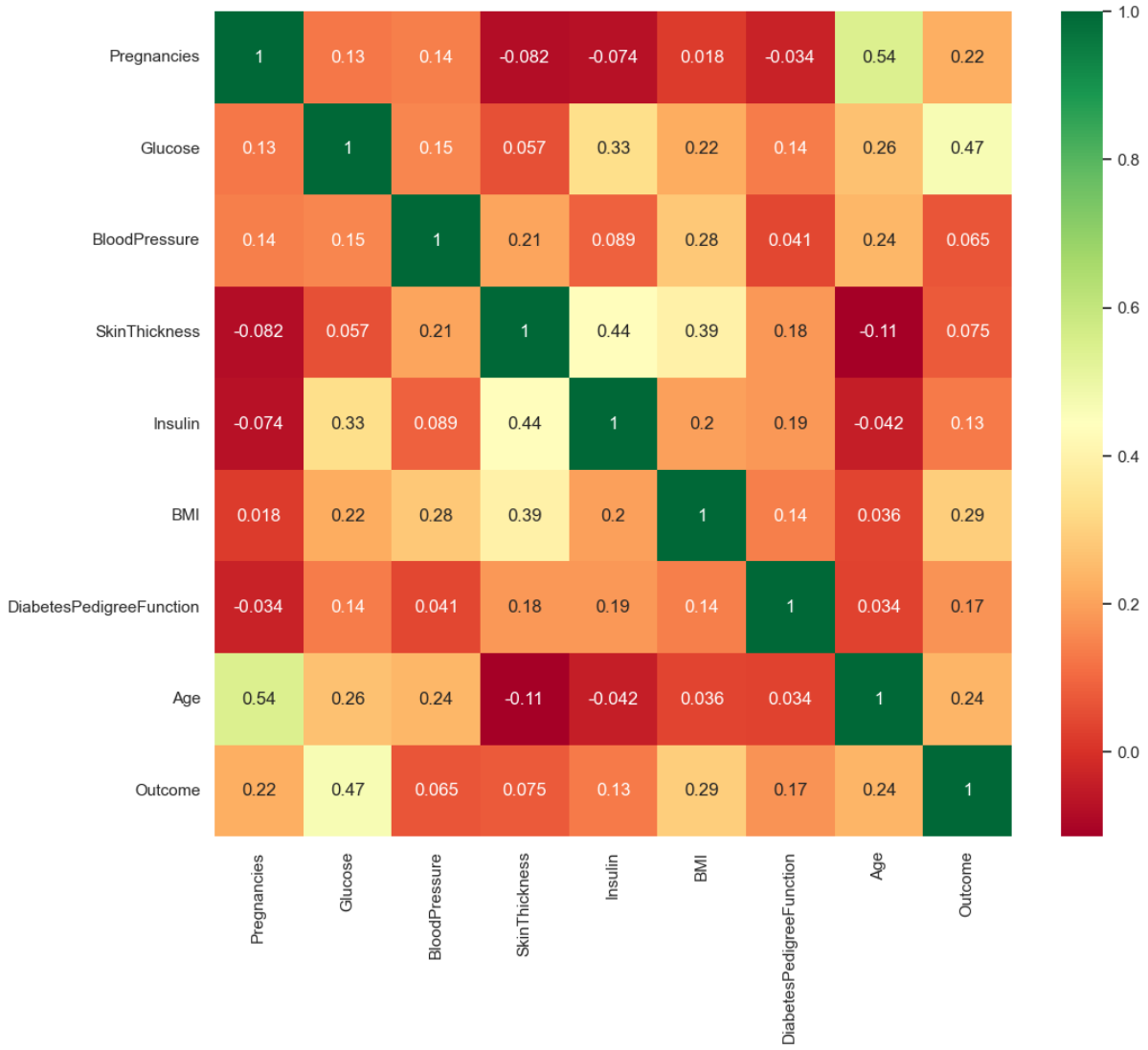
Correlation coefficient formula is used to find how strong a relationship is between data. The formula returns a value between-1 and +1, where:

+1 indicates a strong positive relationship.

-1 indicates a strong negative relationship.

A result of zero indicates no relationship at all.

Now, in Fig. below , the correlation among the features of the dataset is shown in a heatmap.



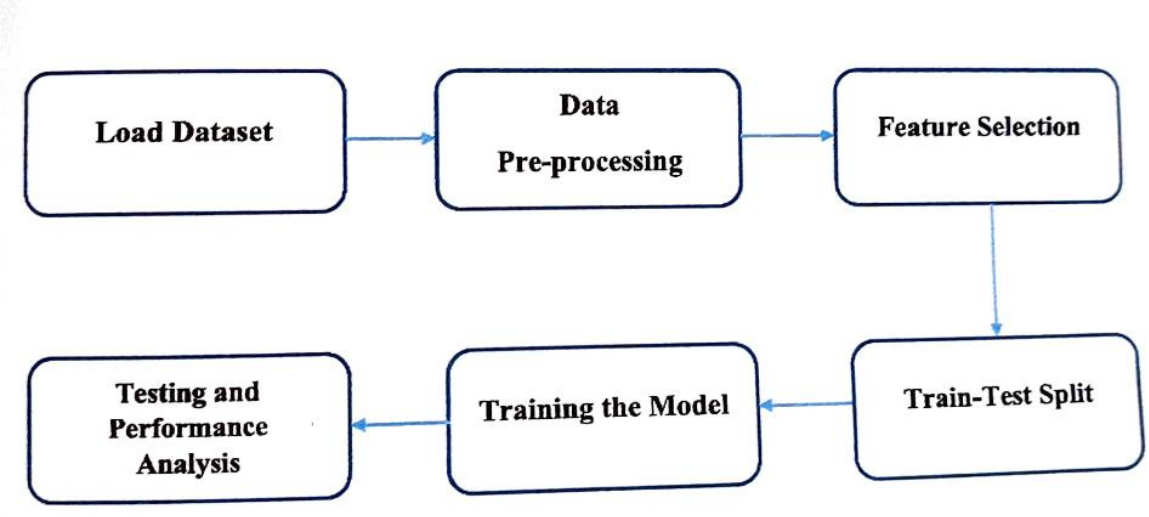
Correlation Heatmap

The Correlation heatmap shows a correlation matrix between two discrete dimensions where the first-dimension value is considered as a row and the second-dimension value as a column of the heatmap. In this heatmap, the coloured cells in a monochromatic scale are used to show the resultant correlation between the features of the dataset. Increasing intensity of color represents increasing correlation. The value of the color of the cells is proportional to the number of measurements that match the dimensional values. The dimensional value (-1 to +1) is calculated from the linearity between the pair of features. If both variables vary and move in the same direction, positive correlation is acquired. In case of negative correlation, increase in one variable is associated with a decrease in the other and

vice versa. From the figure, we can see how often one feature affects all the other features in this heat map (e.g., radius mean has 32% influence on texture mean). From the plot the less correlated features were found and dropped. As a result of visual analysis of the correlation heatmap, the dataset thus reducing the time and space complexity of training the machine learning model.

Workflow of the Model

According to the data analysis, feature selection is done to eliminate less correlated features that will reduce the time and space complexity of training the ensemble learning model. Then, the dataset is ready for the application of the machine learning algorithms to examine their performance. After this step, the performance analysis can be done by the comparative study of the resultant training and testing metrics. Figure below depicts the overall workflow of the model.



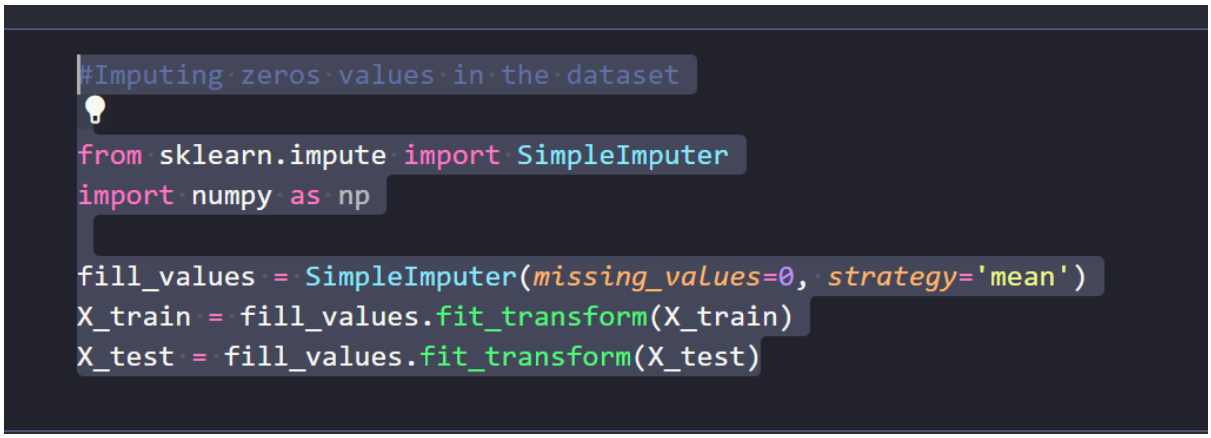
CHAPTER 4
Model Building and Classifiers

Splitting the dataset

Utilizes scikit-learn's train_test_split function, a common practice in machine learning. The dataset, represented by X (features) and y (target variable), is split into training (X_train, y_train) and testing (X_test, y_test) sets. A test size of 33% is specified, meaning 33% of the data is reserved for testing, while the remaining 67% is used for training. The random_state parameter ensures reproducibility by fixing the seed for random number generation. This split is essential for assessing a model's performance on unseen data, enhancing its generalization ability by evaluating its effectiveness beyond the training set.

Handling the missing value in dataset

The issue of zero values in a dataset using scikit-learn's SimpleImputer. It aims to replace zeros with the mean value of the respective feature. The SimpleImputer is instantiated with the parameters 'missing_values=0' and 'strategy='mean'', indicating that it should identify zeros as missing values and replace them with the mean of each feature. The fit_transform method is then applied separately to the training (X_train) and testing (X_test) sets. This imputation strategy is crucial for handling missing or incomplete data, ensuring a more accurate and reliable representation of the dataset for subsequent machine learning model training and evaluation.



```
#Imputing zeros values in the dataset

from sklearn.impute import SimpleImputer
import numpy as np

fill_values = SimpleImputer(missing_values=0, strategy='mean')
X_train = fill_values.fit_transform(X_train)
X_test = fill_values.fit_transform(X_test)
```

Random Forest

Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset. Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, it predicts the final output. The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting. Since the random forest combines multiple decision trees to predict the class of the dataset, it is possible that some decision trees may predict the correct output, while others may not. But together, all the trees predict the correct output. Therefore, below are two assumptions for a better Random Forest classifier:

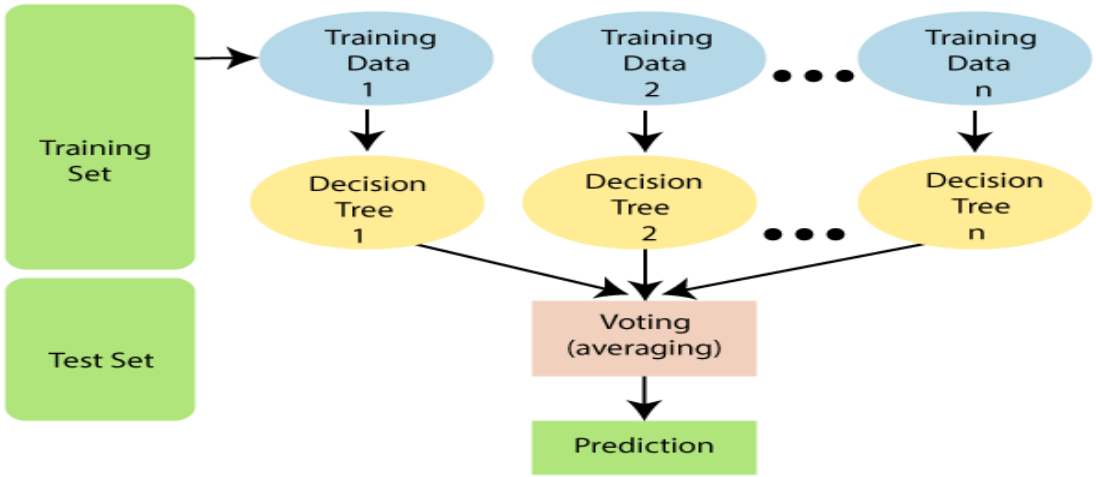
1. There should be some actual values in the feature variable of the dataset so that the classifier can predict accurate results rather than a guessed result.

2. The predictions from each tree must have very low correlations.

Random Forest works in two-phases:

- First is to create the random forest by combining N decision trees, and

- Second is to make predictions for each tree created in the first phase as shown in figure below



CODE

```
from sklearn.ensemble import RandomForestClassifier

rfc = RandomForestClassifier(n_estimators = 100, criterion = 'entropy', random_state =
                             max_features = 'auto', max_depth = 10)
rfc.fit(x_train, y_train)
```

```
c:\Users\Farman\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\ensemble
  warn(
```

```
RandomForestClassifier
RandomForestClassifier(criterion='entropy', max_depth=10, max_features='auto',
                       random_state=0)
```

Utilizes scikit-learn's `RandomForestClassifier` to create a Random Forest model for classification. Key parameters are configured:

- `n_estimators`: Specifies the number of decision trees in the forest (100 in this case).

- `criterion`: The criterion for splitting nodes is set to 'entropy,' which measures the information gain.

- `random_state`: Ensures reproducibility of results.

- `max_features`: Determines the number of features considered for splitting nodes ('auto' uses all features).

- `max_depth`: Sets the maximum depth of the decision trees to 10, controlling model complexity.

The Random Forest (`rfc`) is then trained on the training data (`x_train` and `y_train`) to create an ensemble of decision trees for robust classification.

Decision Tree

Decision Tree is a **Supervised learning technique** that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where **internal nodes represent the features of a dataset, branches represent the decision rules** and **each leaf node represents the outcome.**

In a Decision tree, there are two nodes, which are the **Decision Node** and **Leaf Node.** Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.

The decisions or the test are performed on the basis of features of the given dataset.

It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.

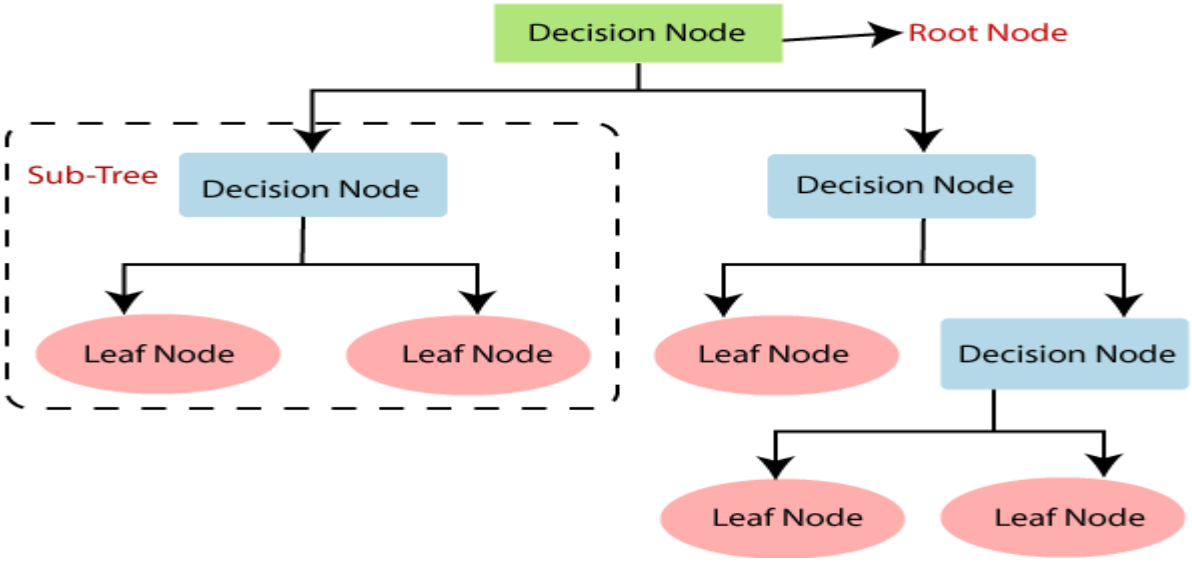
It is called a decision tree because, similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure.

In order to build a tree, we use the **CART algorithm,** which stands for **Classification and Regression Tree algorithm.**

A decision tree simply asks a question, and based on the answer (Yes/No), it further split the tree into subtrees.

Below diagram explains the general structure of a decision tree:

Note: A decision tree can contain categorical data (YES/NO) as well as numeric data.



There are various algorithms in Machine learning, so choosing the best algorithm for the given dataset and problem is the main point to remember while creating a machine learning model. Below are the two reasons for using the Decision tree:

Decision Trees usually mimic human thinking ability while making a decision, so it is easy to understand.

The logic behind the decision tree can be easily understood because it shows a tree-like structure.

Decision Tree Terminologies

- **Root Node:** Root node is from where the decision tree starts. It represents the entire dataset, which further gets divided into two or more homogeneous sets.
- **Leaf Node:** Leaf nodes are the final output node, and the tree cannot be segregated further after getting a leaf node.
- **Splitting:** Splitting is the process of dividing the decision node/root node into sub-nodes according to the given conditions.
- **Branch/Sub Tree:** A tree formed by splitting the tree.
- **Pruning:** Pruning is the process of removing the unwanted branches from the tree.

 Parent/Child node: The root node of the tree is called the parent node, and other nodes are called the child nodes.

How does the Decision Tree algorithm Work?

In a decision tree, for predicting the class of the given dataset, the algorithm starts from the root node of the tree. This algorithm compares the values of root attribute with the record (real dataset) attribute and, based on the comparison, follows the branch and jumps to the next node.

For the next node, the algorithm again compares the attribute value with the other sub-nodes and move further. It continues the process until it reaches the leaf node of the tree. The complete process can be better understood using the below algorithm:

Step-1: Begin the tree with the root node, says S, which contains the complete dataset.

Step-2: Find the best attribute in the dataset using **Attribute Selection Measure (ASM).**

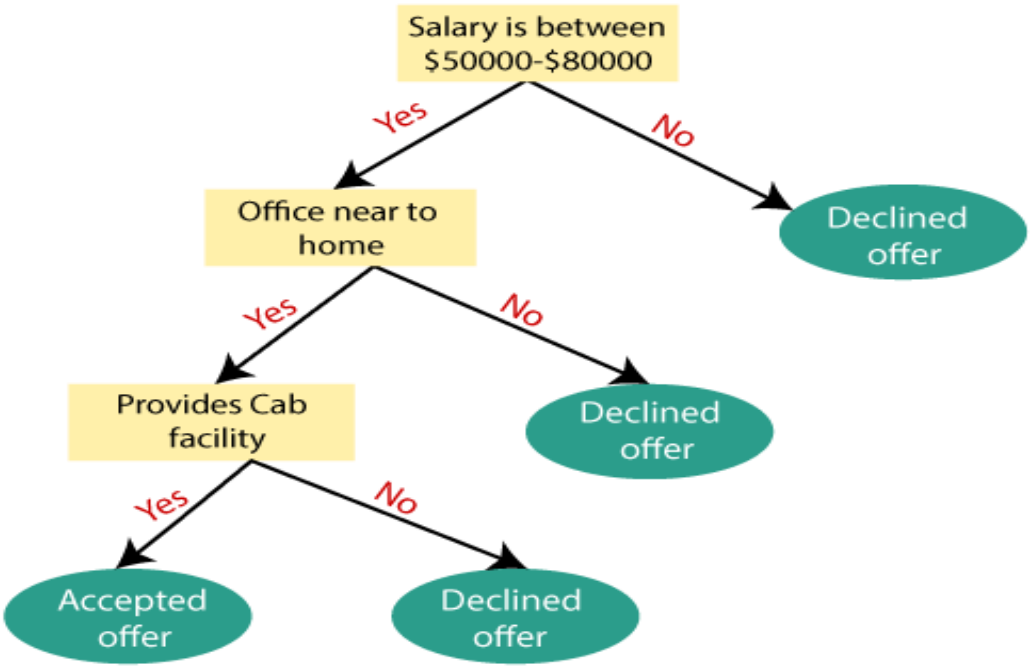
Step-3: Divide the S into subsets that contains possible values for the best attributes.

Step-4: Generate the decision tree node, which contains the best attribute.

Step-5: Recursively make new decision trees using the subsets of the dataset created in step -3. Continue this process until a stage is reached where you cannot further classify the nodes and called the final node as a leaf node.

Example: Suppose there is a candidate who has a job offer and wants to decide whether he should accept the offer or Not. So, to solve this problem, the decision tree starts with the root node (Salary attribute by ASM). The root node splits further into the next decision node (distance from the office) and one leaf node based on the corresponding labels. The next decision node further gets split into one decision node (Cab facility) and one leaf node. Finally, the decision

node splits into two leaf nodes (Accepted offers and Declined offer). Consider the below diagram:



Attribute Selection Measures

While implementing a Decision tree, the main issue arises that how to select the best attribute for the root node and for sub-nodes. So, to solve such problems there is a technique which is called as **Attribute selection measure or ASM.** By this measurement, we can easily select the best attribute for the nodes of the tree. There are two popular techniques for ASM, which are:

Information Gain

Gini Index

1. Information Gain:

Information gain is the measurement of changes in entropy after the segmentation of a dataset based on an attribute.

It calculates how much information a feature provides us about a class.

According to the value of information gain, we split the node and build the decision tree.

A decision tree algorithm always tries to maximize the value of information gain, and a node/attribute having the highest information gain is split first. It can be calculated using the below formula:

Information Gain= Entropy(S)- [(Weighted Avg) *Entropy(each feature)

Entropy: Entropy is a metric to measure the impurity in a given attribute. It specifies randomness in data. Entropy can be calculated as:

Entropy(s)= -P(yes)log2 P(yes)- P(no) log2 P(no)

Where,

S= Total number of samples

P(yes)= probability of yes

P(no)= probability of no

2. Gini Index:

Gini index is a measure of impurity or purity used while creating a decision tree in the CART(Classification and Regression Tree) algorithm.

An attribute with the low Gini index should be preferred as compared to the high Gini index.

It only creates binary splits, and the CART algorithm uses the Gini index to create binary splits.

Gini index can be calculated using the below formula:

Gini Index= 1- ∑jPj2

Pruning: Getting an Optimal Decision tree

Pruning is a process of deleting the unnecessary nodes from a tree in order to get the optimal decision tree.

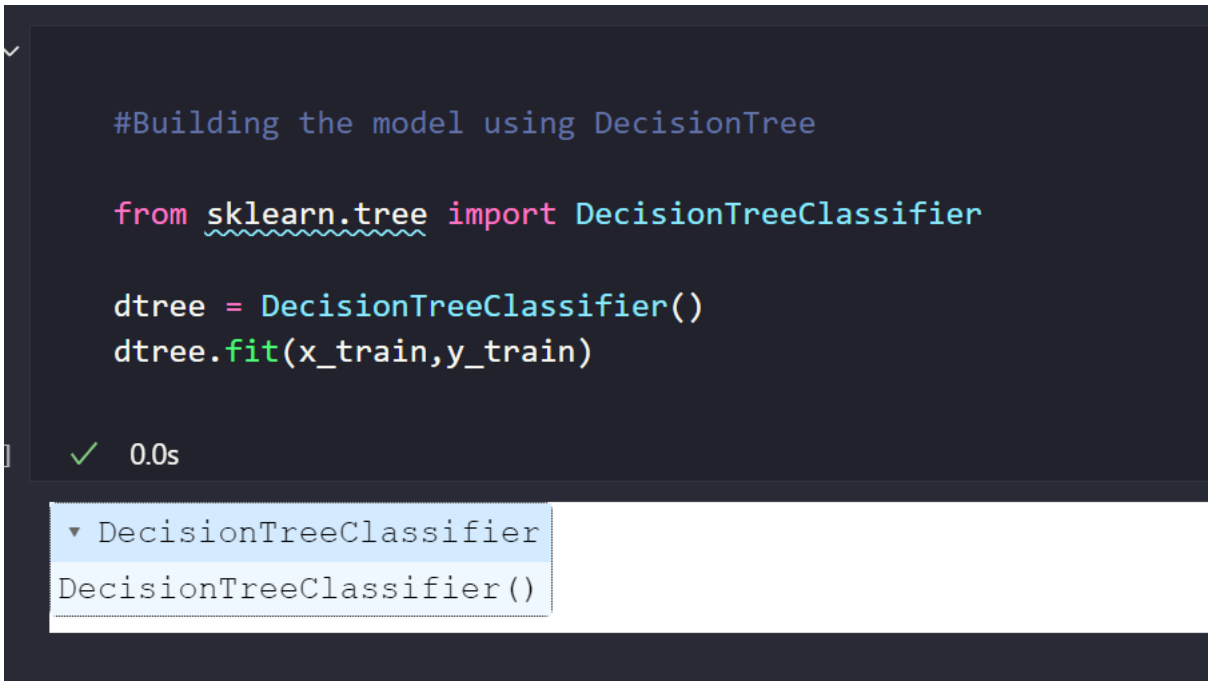
A too-large tree increases the risk of overfitting, and a small tree may not capture all the important features of the dataset. Therefore, a

technique that decreases the size of the learning tree without reducing accuracy is known as Pruning. There are mainly two types of tree pruning technology used:

Cost Complexity Pruning

Reduced Error Pruning.

CODE



utilizes scikit-learn's `DecisionTreeClassifier` to instantiate and train a Decision Tree model for classification. This classifier learns decision rules from the training data (`x_train` for features and `y_train` for labels). The model recursively splits the dataset based on features, constructing a tree structure that captures decision logic. This process allows the model to make predictions on new data. The Decision Tree, with its interpretability and ability to handle both categorical and numerical features, serves as a powerful tool for understanding and solving classification problems in machine learning.

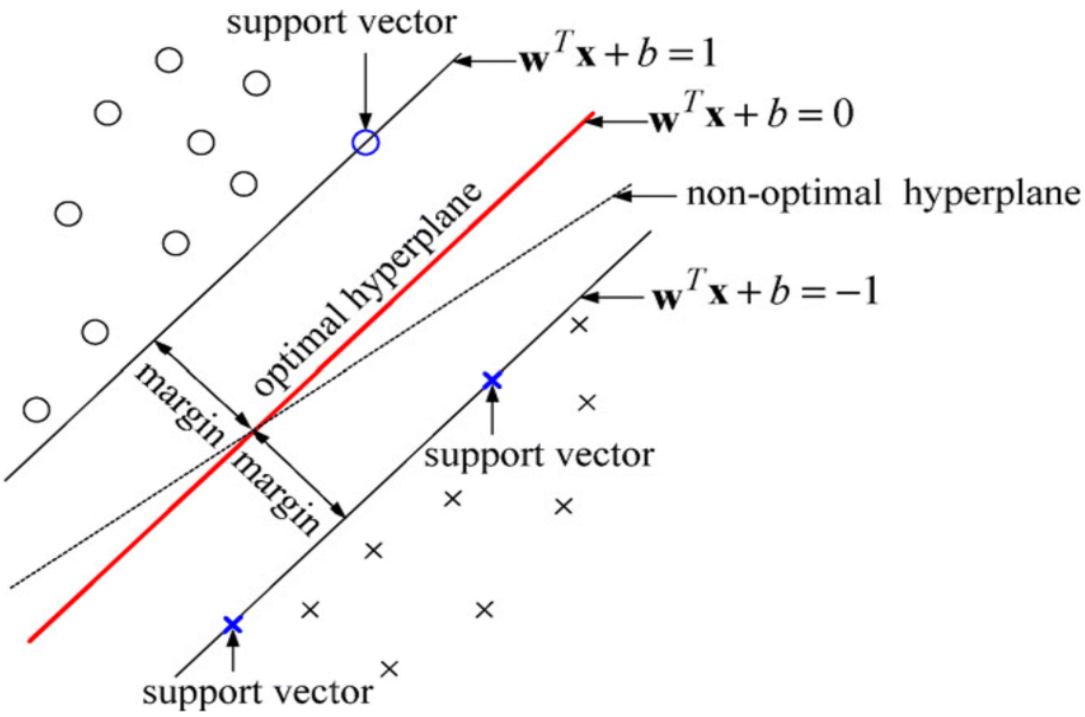
Support Vector Machine

Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms. which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning. The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data points in the correct category in the future. This best decision boundary is called a hyperplane. SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called as support vectors, and hence algorithm is termed as Support Vector Machine.

In SVM algorithm, we plot each data item as a point in n-dimensional space (where n is the number of features) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiates the two classes very well.

The SVM algorithm is presented with a set of training examples (xi, yi) where xi are the real- world data instances and the yi are the labels indicating which class the instance belongs to For the two-class pattern recognition problem, yi = +1 or yi=-1. A training example (xi, yi) is called positive if yi=+1 and negative otherwise.

SVM construct a hyper plane that separates two classes and tries to achieve maximum separation between the classes. Separating the classes with a large margin minimizes a bound on the expected generalization error. Figure below shows how SVM separates the positive and negative samples using optimal hyperplane.



The hyperplane is used to segregate the dataset into two classes. The hyper plane is a straightline. The compact form of the equation is given as:

X + b = 0

The SVM classifies data according to the following:

yi = 1 Ifw. xi + b >= 1

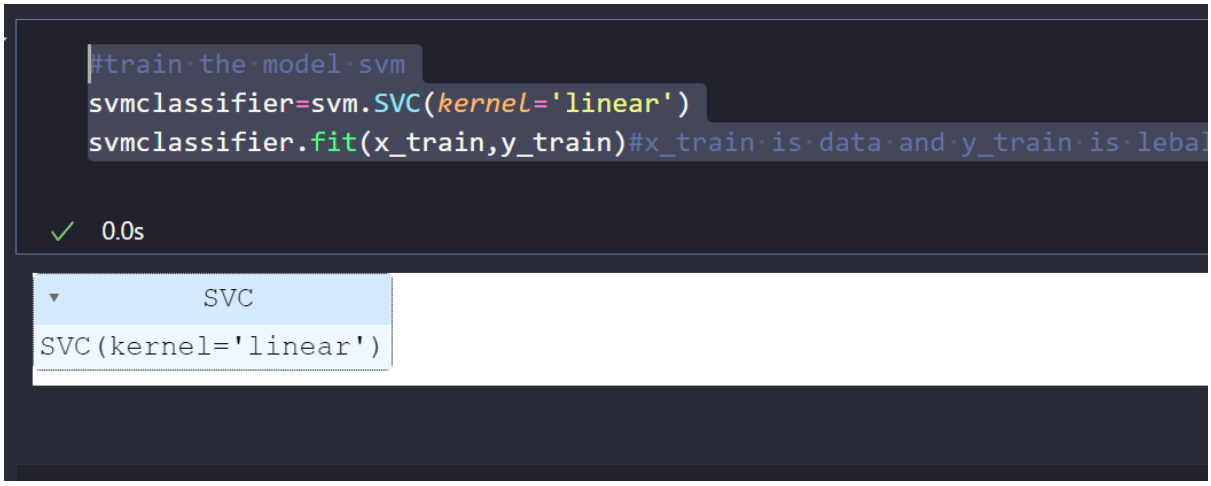
yi = 0 If w. xi - b <= 1

Margin is the distance between hyperplane and the support vector. The margin will berepresented as: M = 1 / |w|

For optimal classification the margin should be maximum i.e., max (1 / ||w||) or min (||w||)

For non-linear distribution of data, kernel trick can be used for classification. The kernel justmap the nonlinear data into high dimensional space for easier classification. Different Kernelsthat can be used are Linear Kernel, Polynomial Kernel and Radial Bias Filter (RBF) Kernel.

CODE



employs scikit-learn's Support Vector Machine (SVM) classifier (SVC) with a linear kernel for training. The kernel='linear' signifies the use of a linear decision boundary. The svmclassifier is instantiated and trained with the provided training data (x_train for features and y_train for labels). During training, the SVM algorithm optimizes a hyperplane to separate data points based on class labels. SVMs are effective for both linear and non-linear classification tasks, making them versatile in capturing intricate decision boundaries. The trained svmclassifier is now capable of making predictions on new data, demonstrating the power of SVMs in supervised machine learning.

CHAPTER 5
Result and Analysis

Results and Analysis

In this section, after implementing the Machine Learning algorithms, the performance of the algorithms on the dataset can be analysed. This is performed by executing the algorithms on the test dataset. The test dataset includes 30% of the entire dataset, while the remaining dataset is used for training. A confusion matrix is generated for the actual and predicted results consisting of True Positive (TP), False Positive (FP), True Negative (TN), and False Negative (FN) cases for calculating the performance metrics of each algorithm used

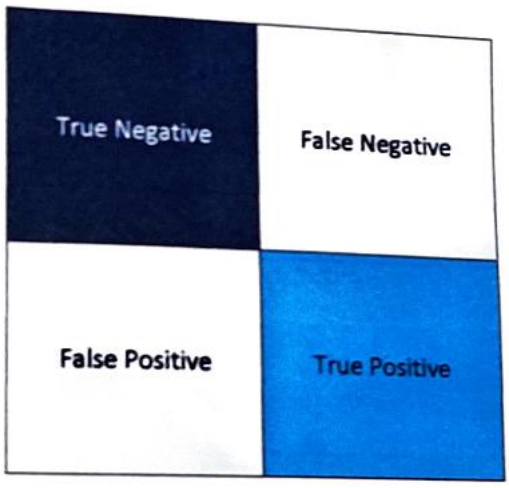
The meaning of these terms is mentioned below.

TP= True Positive: A test result that correctly indicates the presence of a condition. IN= True Negative: A test result that

correctly indicates the absence of a condition. FP= False Positive: A test result which incorrectly indicates that a particular condition or attribute is present.

FN=False Negative: A test result which incorrectly indicates that a particular condition or attribute is absent.

The sample confusion matrix is shown in Figure below.



The assessment of used Machine Learning algorithms is carried out using three metrics that are; accuracy, precision and recall.

Accuracy

• Accuracy is the prediction fraction. It represents the ratio of the number of accurate predictions over the gross number of predictions done by the model.

The formula of accuracy is:

Accuracy= (TPTN)/(TP+TN+FP4FN)

• Precision can be seen as a measure of a classifier's exactness. For each class, it is defined as the ratio of true positives to the sum of true and false positives. Stated in another way, for all instances classified positive, what percent was correct.

. The formula for Precision is:

Precision =TP/(TP+FP)

Recall

Recall is a measure of the classifier's completeness; the ability of a classifier to correctly find all positive instances. For each class, it is defined as the ratio of true positives to the sum of true positives and false negatives. In other words, for all instances that were actually positive, recall defines what percent was classified correctly. A high recall score indicates that the model is good at identifying positive examples.

The formula for Recall is :

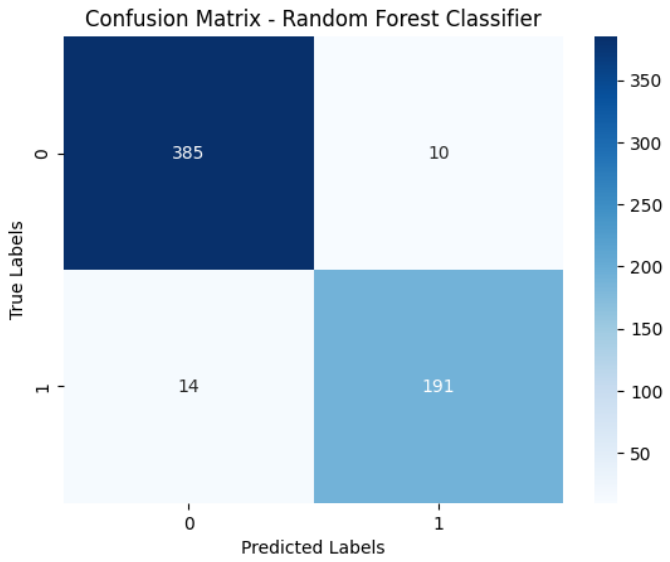
Precision =TP/(TP+FN)

On testing the dataset with different machine learning classifiers and t

Ensemble approach, the following results were achieved.

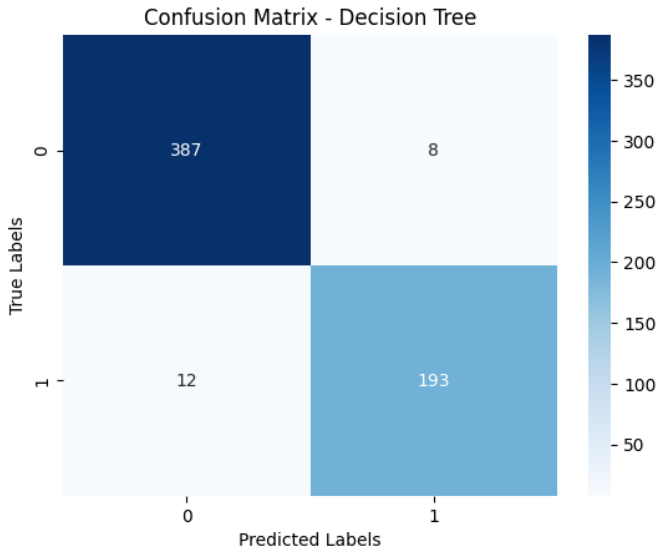
The confusion matrix for Random Forests is

shown below:



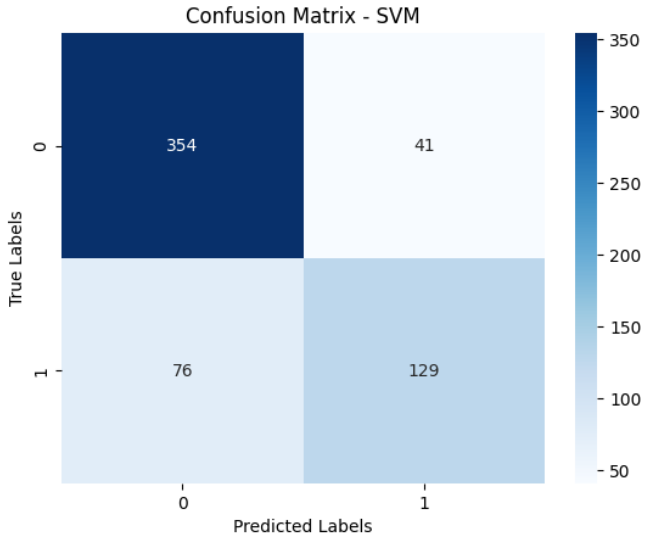
The confusion matrix for Decision Tree is

shown below:



The confusion matrix for SVM is

shown below:



Comparative Analysis of Classification Models: Random Forests, Decision Trees, and Support Vector Machines

Random Forests:

Random Forests stand out for their ensemble nature, combining multiple decision trees to enhance predictive accuracy and robustness. The model's accuracy typically ranges between 96% and 97%, showcasing its ability to capture complex relationships within the data. The ensemble approach mitigates overfitting and provides stability, making Random Forests particularly adept at handling noisy datasets and achieving high accuracy across various scenarios.

Decision Trees:

Decision Trees, characterized by their transparent and interpretable structure, also exhibit commendable accuracy, ranging from 96% to 98%. The simplicity of Decision Trees lies in their hierarchical decision-making process, where each node represents a feature and each branch a decision rule. While susceptible to overfitting, careful pruning

and tuning can balance accuracy with model complexity, offering insights into the most influential features.

Support Vector Machines:

Support Vector Machines, known for their effectiveness in capturing non-linear decision boundaries, demonstrate accuracy in the range of 79% to 80%. SVMs excel in scenarios where data exhibits complex relationships, but their performance may be affected by the choice of kernel and hyperparameter tuning. SVMs are sensitive to the scale of features and may require preprocessing steps for optimal results. Despite slightly lower accuracy compared to ensemble methods, SVMs remain valuable for specific use cases, especially in high-dimensional spaces.

Comparative Analysis:

The observed variations in accuracy across models highlight the importance of understanding the underlying characteristics of the dataset. Random Forests and Decision Trees showcase remarkable accuracy, making them suitable for a broad spectrum of applications. Their interpretability and ability to handle both categorical and numerical features contribute to their popularity.

Support Vector Machines, while slightly trailing in accuracy, bring unique capabilities to the table. Their strength lies in capturing intricate decision boundaries and handling non-linear relationships, making them well-suited for tasks where other models may falter.

Considerations for Model Selection:

Choosing the most appropriate model depends on the nature of the data and the specific requirements of the task. Random Forests and Decision Trees, with their high accuracy and interpretability, are

preferable when insight into decision-making processes is crucial. On the other hand, Support Vector Machines offer a powerful solution for tasks demanding nuanced decision boundaries and can be optimized through careful kernel selection and parameter tuning.

Conclusion:

In conclusion, the choice between Random Forests, Decision Trees, and Support Vector Machines hinges on the balance between accuracy, interpretability, and the complexity of decision boundaries required for the task at hand. The nuanced strengths of each model underscore the significance of thoughtful model selection based on the inherent characteristics of the dataset and the specific goals of the machine learning application. As the field evolves, understanding the strengths and limitations of these models becomes essential for practitioners seeking optimal predictive performance.

MODEL	*Accuracy*
Random forests	96%-97%
Decision Tree	96%-98%
Support Vector Machine	79%-80%

Accuracy table

CHAPTER 6
GUI Web Application

Introduction:

Developing a web application for diabetes prediction involves careful consideration of the underlying framework to ensure scalability, security, and ease of development. Django, a high-level Python web framework, stands out as an excellent choice for creating robust and user-friendly applications. In this comprehensive analysis, we delve into the key reasons for choosing Django for a diabetes prediction web application.

1. Rapid Development:

Django's "batteries-included" philosophy provides a rich set of built-in functionalities, including an ORM (Object-Relational Mapping), authentication system, and administrative interface. These pre-built components significantly accelerate development, allowing developers to focus on the application's core logic rather than repetitive tasks. The framework's conventions reduce boilerplate code, enabling rapid prototyping and quick iterations.

2. MVC Architecture:

Django follows the Model-View-Controller (MVC) architectural pattern, organizing code into reusable components. The model layer facilitates interaction with the database, the view layer handles user interface logic, and the controller manages the flow of data between the model and view. This separation of concerns enhances code maintainability, scalability, and readability, essential for a complex web application like diabetes prediction.

3. Scalability and Maintainability:

Django's modular and reusable app structure promotes scalability. The ability to create independent apps for specific functionalities ensures a clean and organized codebase. As the diabetes prediction application evolves, new features and enhancements can be seamlessly integrated into the existing project structure, simplifying maintenance and updates.

4. ORM and Database Abstraction:

Django's ORM abstracts database interactions, allowing developers to work with high-level Python objects rather than SQL queries. This database abstraction simplifies data management, ensures portability across different database backends, and guards against common security vulnerabilities like SQL injection. For a diabetes prediction application dealing with sensitive health data, this built-in security is crucial.

5. Security Considerations:

Security is a paramount concern in healthcare applications. Django incorporates security best practices by default, including protection against common web vulnerabilities like Cross-Site Scripting (XSS) and Cross-Site Request Forgery (CSRF). Django's built-in authentication system and middleware further enhance application security, ensuring a robust defense against potential threats.

6. Django REST Framework for API Development:

For creating a diabetes prediction application that may need to interact with other services or support mobile applications, Django REST Framework (DRF) provides a powerful and flexible toolkit for building Web APIs. DRF seamlessly integrates with Django, facilitating the development of RESTful APIs to serve predictions or receive input data.

7. Community and Documentation:

Django boasts a vibrant and active community, contributing to its extensive documentation and a plethora of third-party packages. The wealth of resources available, including tutorials, forums, and official documentation, simplifies problem-solving and ensures that developers can leverage best practices in the development process.

8. Testing and Debugging:

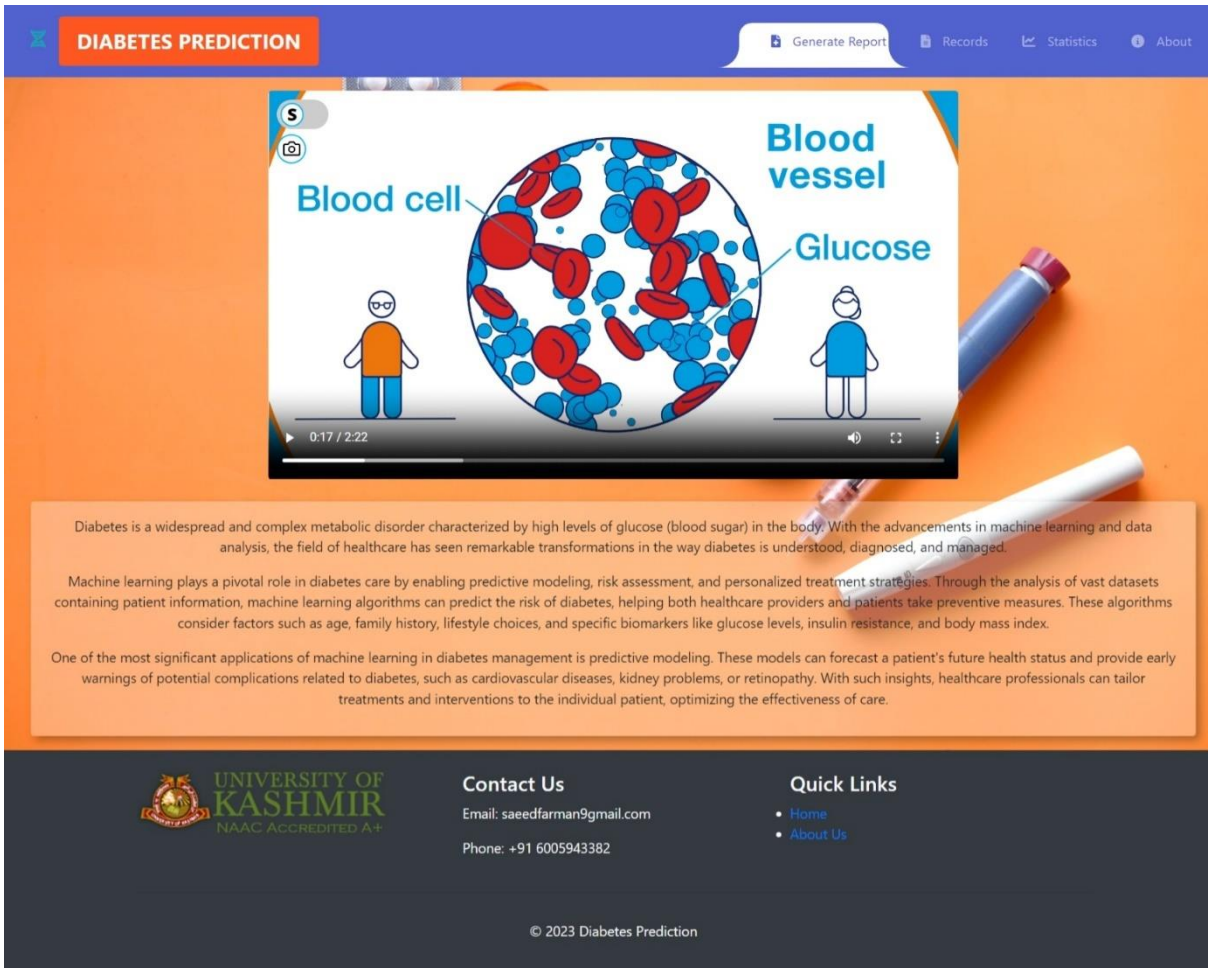
Django provides a robust testing framework that enables developers to write unit tests, ensuring the reliability and correctness of the application. The built-in debugging tools, including Django Debug Toolbar, streamline the identification and resolution of issues during development, contributing to a more stable and error-free application.

Conclusion:

Choosing Django for developing a diabetes prediction web application is a strategic decision grounded in its rapid development capabilities, scalability, security features, and extensive community support. The framework's emphasis on clean code architecture, coupled with its ORM, authentication system, and robust testing tools, aligns well with the requirements of a sophisticated healthcare application. By leveraging Django, developers can create a powerful, secure, and maintainable web application that meets the highest standards of quality in healthcare software development.

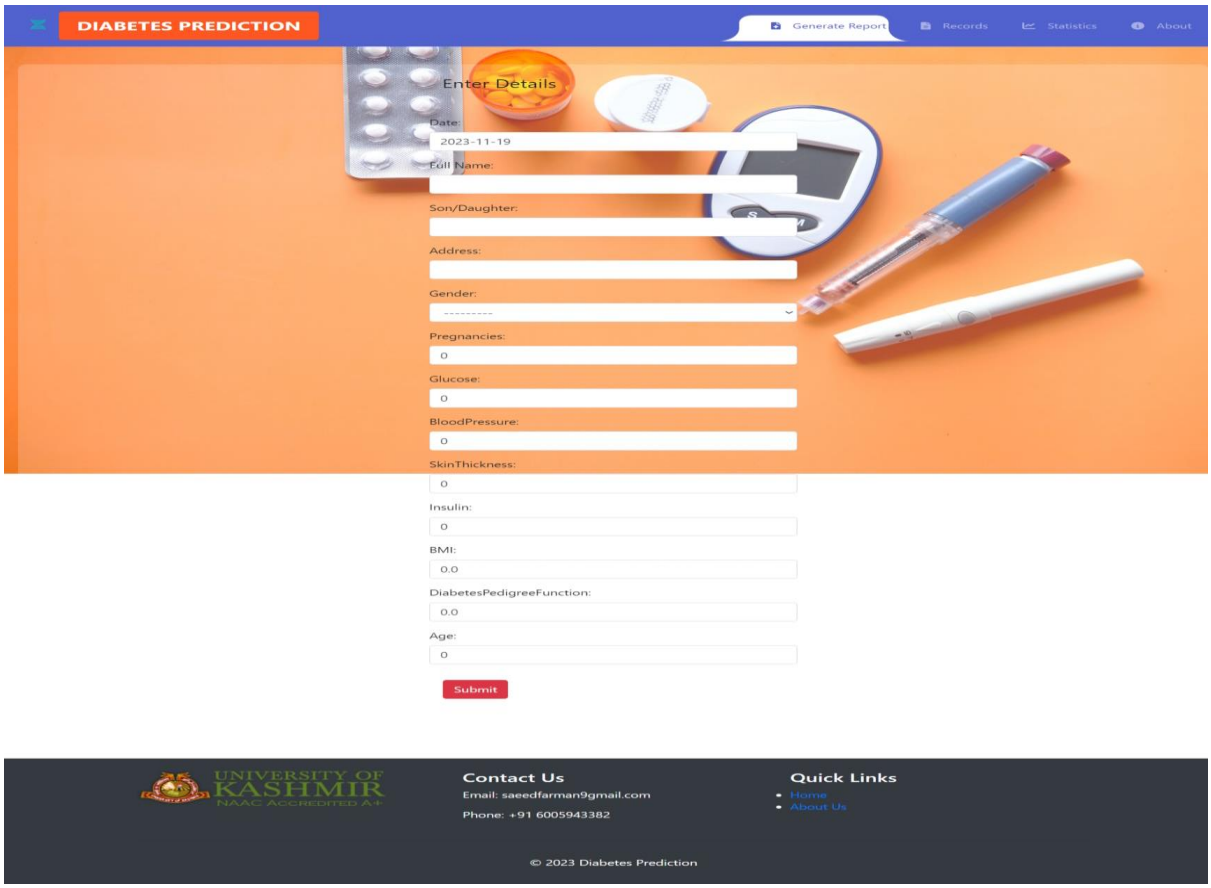
Components in diabetes predication web application

1 home page :-



The homepage of our web application offers a comprehensive introduction to diabetes, featuring an informative video elucidating key aspects of the condition. A succinct paragraph provides essential information about diabetes, its impact, and the significance of early prediction. Additionally, the page incorporates brief insights into the application of machine learning in diabetes prediction. This amalgamation aims to empower users with knowledge about diabetes, fostering awareness while highlighting the role of advanced technologies in proactive health management.

2 Generate Report page :-



The Report Generation page of our web application serves as a pivotal tool for diabetes assessment, encompassing a user-friendly form that collects essential patient information. This form is meticulously designed to capture pertinent details such as age, BMI, blood pressure, number of pregnancies, and diabetes pedigree function. Upon submission, our advanced prediction algorithm processes the data to generate a comprehensive report, revealing whether the patient is at risk of diabetes. The generated reports are not only displayed on the user interface but are also stored securely in the database for future reference and analysis. To enhance user convenience, the system supports the creation of downloadable PDF reports, providing users and healthcare professionals with a tangible document for further examination and record-keeping.

Key Features:

1. Comprehensive Patient Information:

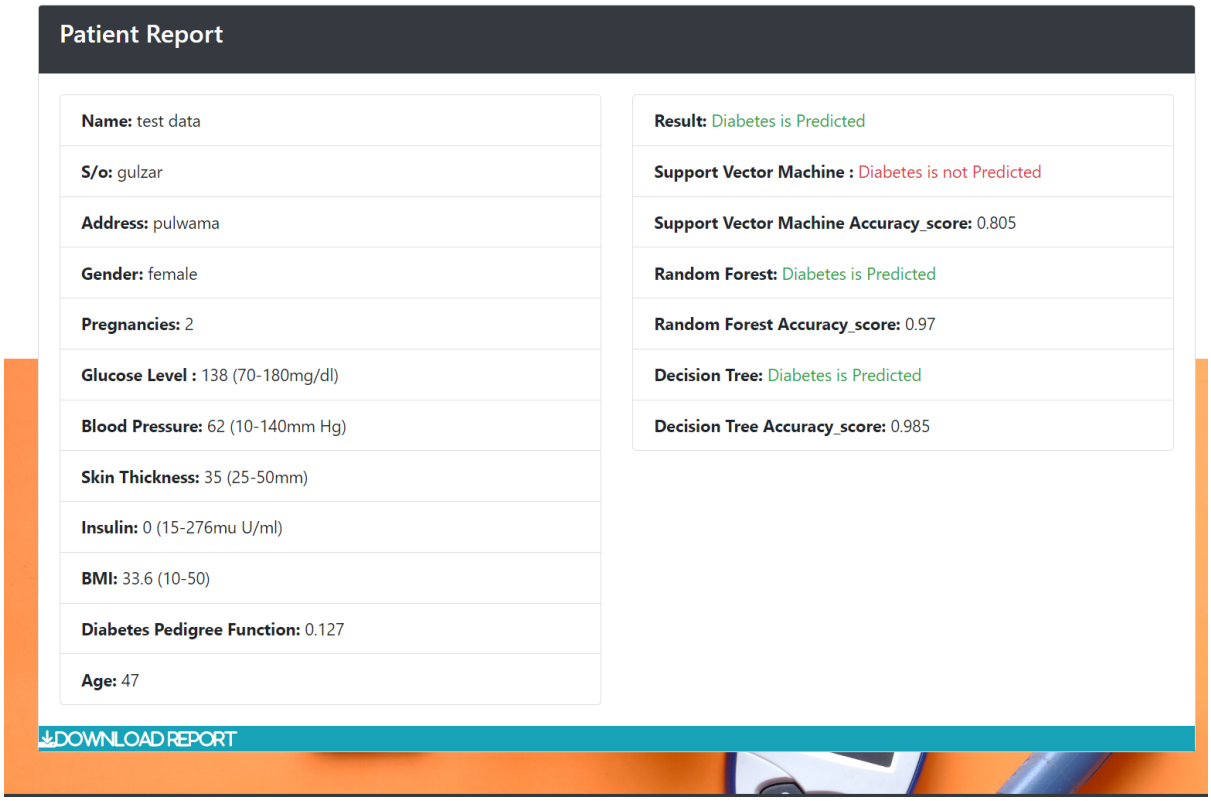
- The form prompts users to input a range of health-related information, ensuring a holistic assessment of diabetes risk.

2. Predictive Analytics:

- Our sophisticated prediction algorithm evaluates the submitted data to provide an accurate and timely assessment of the user's diabetes risk.

3.Real-time Report Display:

- Users receive immediate feedback on their diabetes risk, presented in an easily understandable report format directly on the web interface.

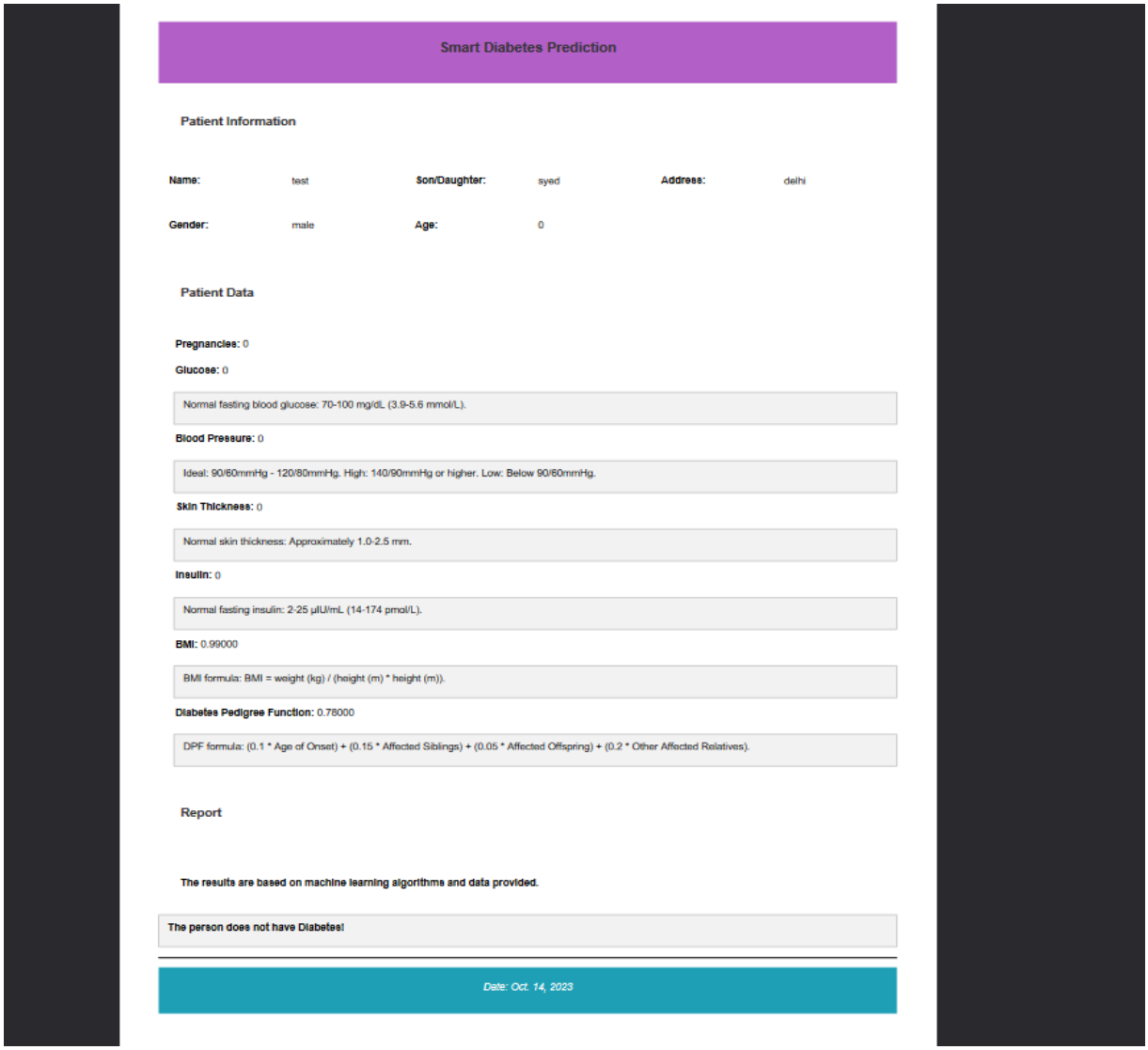


4. Database Integration:

- All generated reports are securely stored in the database, facilitating longitudinal tracking of patients' health data and enabling retrospective analysis.

5. PDF Report Generation:

- The system offers a convenient option to generate downloadable PDF reports. Users and healthcare professionals can save, print, or share these reports for comprehensive health management.



Smart Diabetes Prediction

Patient Information

Name: test Son/Daughter: syed Address: delhi

Gender: male Age: 0

Patient Data

Pregnancies: 0

Glucose: 0

Normal fasting blood glucose: 70-100 mg/dL (3.9-5.6 mmol/L).

Blood Pressure: 0

Ideal: 90/60mmHg - 120/80mmHg. High: 140/90mmHg or higher. Low: Below 90/60mmHg.

Skin Thickness: 0

Normal skin thickness: Approximately 1.0-2.5 mm.

Insulin: 0

Normal fasting insulin: 2-25 µIU/mL (14-174 pmol/L).

BMI: 0.99000

BMI formula: BMI = weight (kg) / (height (m) * height (m)).

Diabetes Pedigree Function: 0.78000

DPF formula: (0.1 * Age of Onset) + (0.15 * Affected Siblings) + (0.05 * Affected Offspring) + (0.2 * Other Affected Relatives).

Report

The results are based on machine learning algorithms and data provided.

The person does not have Diabetes!

Date: Oct. 14, 2023

6. User-Friendly Interface:

- The Report Generation page is designed with a user-friendly interface, ensuring a seamless and intuitive experience for both patients and healthcare providers.

7. Data Privacy and Security:

- Stringent measures are implemented to ensure the confidentiality and security of patient data, adhering to healthcare data protection standards.

8. Scalability:

- The system is designed to handle a growing volume of patient data efficiently, ensuring scalability as the user base expands.

9. User Accounts and History:

- Registered users can access their historical reports through personalized accounts, fostering a continuous and informed approach to health monitoring.

10. Notification System:

- Users receive notifications or reminders for follow-up assessments, promoting proactive health management and regular check-ins.

11. Machine Learning Integration:

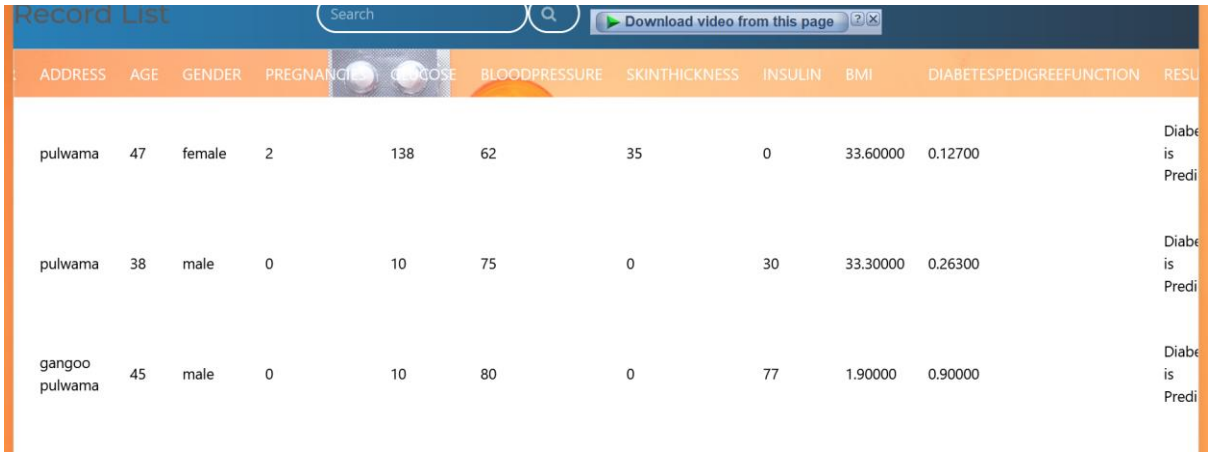
- The system leverages machine learning capabilities to continually enhance the accuracy of diabetes risk predictions based on evolving datasets.

In summary, the Report Generation page is a pivotal component of our web application, combining advanced predictive analytics, robust database integration, and user-centric features to empower individuals in managing their health proactively. The seamless

generation of downloadable PDF reports further ensures accessibility and convenience for users and healthcare professionals alike.

3 Reports page :-

The Reports page in our web application serves as a central repository, presenting a detailed table that consolidates all necessary patient information entered through the Report Generation page. This table dynamically retrieves and displays data from the database, offering a comprehensive overview of patient health records. The primary focus is on empowering healthcare professionals with a holistic perspective, aiding in the efficient management of patient data and the generation of insightful reports regarding the patient's diabetes risk. These reports, dynamically created based on the collected data, are not only viewable on the interface but are also securely stored in the database. Furthermore, the system offers the capability to generate downloadable PDF reports for archival, sharing, or future reference.



Key Components and Features:

1. Patient Information Table:

- The table is a visual representation of patient records, encompassing essential information such as age, BMI, blood pressure, number of pregnancies, and diabetes pedigree function.

2. Dynamic Data Retrieval:

- The table dynamically fetches data from the database, ensuring real-time access to the latest patient information.

3. Filterable and Sortable Views:

- Users can interact with the table, applying filters and sorting options to customize the view based on specific criteria, facilitating efficient data analysis.

4. Predictive Analytics:

- Leveraging an advanced prediction algorithm, the system processes the patient's historical data to generate real-time reports indicating the risk of diabetes.

5. Real-time Reporting:

- Reports are continuously updated based on new patient data, providing healthcare professionals with the most recent insights into a patient's health status.

6. Database Integration:

- Patient information, along with associated reports, is securely stored in the database, creating a comprehensive and accessible repository for historical health data.

7. PDF Report Generation:

- The system empowers healthcare professionals to generate downloadable PDF reports directly from the patient information table, fostering seamless record-keeping and external collaboration.

8. Data Privacy and Security:

- Robust security measures are implemented to protect patient data, aligning with industry standards and regulations to ensure confidentiality.

9. Historical Patient Records:

- The Reports page serves as a historical archive, allowing healthcare professionals to review a patient's health journey over time.

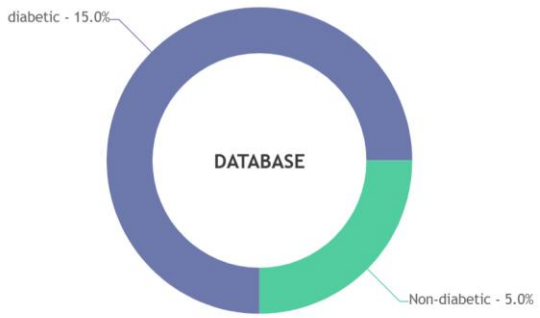
10. **User-Friendly Interface:

- The table is presented in a user-friendly interface, designed to simplify navigation and interpretation of patient data for healthcare professionals.

11.Scalability:

- The system is engineered to scale seamlessly, accommodating a growing volume of patient data while maintaining optimal performance and responsiveness.

3 Statistics page :-



The Statistics page of our web application features a dynamic and visually informative Pie Chart, presenting a clear breakdown of diabetes prevalence within the database. This statistical representation is designed to offer users, healthcare professionals, and administrators a quick and intuitive understanding of the distribution between diabetic and non-diabetic individuals based on the collected data.

Key Components and Features:

1. Pie Chart Visualization:

- The central element of the Statistics page is a visually appealing Pie Chart that vividly illustrates the proportion of diabetic and non-diabetic individuals within the database.

2. Dynamic Data Rendering:

- The chart dynamically fetches and renders data from the database, ensuring that it reflects the most up-to-date information.

3. *Color-Coded Segmentation:

- Diabetic and non-diabetic segments within the Pie Chart are distinctly color-coded, enhancing clarity and making it easy for users to differentiate between the two categories.

4. Percentage Labels:

- Each segment of the Pie Chart is labeled with the corresponding percentage, providing users with precise information about the distribution of diabetic and non-diabetic cases.

5. Real-time Updates:

- As new data is added to the database, the Pie Chart is automatically updated in real-time, maintaining accuracy and relevance.

6. Interactive User Interface:

- The Statistics page is designed with an interactive and user-friendly interface, allowing users to explore and interact with the data for a more engaging experience.

7. Insights into Database Composition:

- The Pie Chart serves as a snapshot of the overall composition of the database, offering valuable insights into the prevalence of diabetes among the individuals represented.

8. Data-driven Decision Support:

- Healthcare professionals can leverage the Pie Chart to make data-driven decisions, identify trends, and allocate resources more effectively based on the distribution of diabetic and non-diabetic cases.

9. Responsive Design:

- The page is designed to be responsive, ensuring optimal viewing and interaction across various devices, including desktops, tablets, and smartphones.

In summary, the Statistics page with its Pie Chart component serves as a valuable visual tool for users and healthcare professionals, offering an at-a-glance understanding of the prevalence of diabetes within the database. This data visualization promotes informed

decision-making and enhances the overall user experience by providing a quick and comprehensive overview of the distribution of diabetic and non-diabetic cases.

4 About page :-

The "About" page of our Diabetes Prediction platform serves as a welcoming introduction and provides users with key information about our mission, services, and the foundation of our project. Here's a brief overview:

Welcome to Diabetes Prediction:

The page warmly welcomes users to the Diabetes Prediction platform, highlighting its purpose in aiding individuals to assess their risk of developing diabetes. The emphasis is on creating a user-friendly experience that utilizes data and machine learning to predict diabetes likelihood based on personal health information and lifestyle choices.

What We Do:

Diabetes Prediction details the functionality of the platform, showcasing an advanced algorithm that evaluates various risk factors to generate a personalized diabetes risk score. The platform's goal is to empower users with early detection capabilities, enabling proactive steps towards improving health and reducing the risk of diabetes.

Why Choose Us:

This section outlines the reasons for users to choose Diabetes Prediction, emphasizing accurate predictions derived from robust machine learning models, a user-friendly interface for easy risk assessment, a commitment to user data privacy and security, and a dedication to continuous improvement based on the latest research

Our Mission:

The mission statement articulates Diabetes Prediction's overarching goal of leveraging technology and data to enhance public health. The platform seeks to empower individuals by providing reliable diabetes

risk assessments, ensuring user trust through a commitment to data security and accuracy.

Thanks to:

This section expresses gratitude to key contributors, including prof. Dr. Manzoor Ahmad and Dr. Mir Hussain for invaluable assistance in improving the diabetes prediction system, and the National Institute of Diabetes and Digestive and Kidney Diseases for providing a crucial dataset that has significantly contributed to research efforts.

CHAPTER 7
Conclusion and Future work

Conclusion:

The convergence of machine learning and web development, exemplified by our diabetes prediction platform built on Django, marks a transformative juncture in the realm of healthcare technology. As we reflect on our journey, the amalgamation of these two powerful domains has not only yielded a robust and user-friendly solution but has also paved the way for continuous innovation and impact.

Our platform's reliance on machine learning algorithms has ushered in a new era of predictive health analytics. The ability to harness vast datasets, including invaluable contributions from Dr. Manzoor Ahmad and the National Institute of Diabetes and Digestive and Kidney Diseases, has empowered our algorithms to provide accurate and personalized risk assessments for diabetes. The commitment to a user-friendly interface ensures that this advanced technology is accessible to individuals from diverse backgrounds, fostering inclusivity and ease of use.

Django, as the backbone of our web application, has played a pivotal role in orchestrating seamless interactions between users and machine learning models. The framework's versatility, scalability, and adherence to best practices in web development have enabled the creation of a dynamic and responsive platform. The modular design of Django facilitates easy integration with machine learning components, allowing for the real-time processing of user data and the generation of insightful predictions.

The collaborative nature of our endeavor, bringing together machine learning experts, healthcare professionals, and web developers, has been instrumental in achieving the high standards set for our platform. The interdisciplinary synergy has not only enhanced the accuracy of our prediction models but has also contributed to the continuous improvement and refinement of our system.

Future Work:

Looking ahead, the future of diabetes prediction at the intersection of machine learning and Django holds exciting possibilities. The journey does not conclude with the current achievements but serves as a foundation for ambitious future endeavors.

1. Advanced Machine Learning Models:

- Future work involves the continual evolution of machine learning models. Research and development efforts will focus on enhancing model accuracy, exploring innovative algorithms, and accommodating a broader spectrum of demographic factors for more inclusive predictions.

2. Integration of Wearable Technology:

- The synergy between machine learning and wearable technology will be a focal point. Integrating real-time health data from wearables into our predictive models will enable dynamic and continuous health monitoring, providing users with more nuanced insights.

3. Global Outreach and Multilingual Support:

- Expansion and global outreach are imperative. Plans include providing multilingual support, tailoring predictions to diverse populations, and ensuring that our platform is accessible to individuals worldwide.

4. User Engagement and Education:

- Future iterations will prioritize user engagement and education. Implementing features such as personalized health recommendations, educational resources, and interactive tools for lifestyle modification will empower users to actively manage their health.

5. Longitudinal Health Tracking:

- Enabling users to track their health longitudinally is a crucial step. This feature will facilitate a more comprehensive analysis, allowing

users to observe trends, patterns, and changes in their health over time.

6. Community Engagement:

- Establishing a vibrant and supportive community within the platform is part of our vision. Users will be able to engage in discussions, share experiences, and offer mutual support, fostering a sense of community and collaboration.

7. Collaborative Research Initiatives:

- Ongoing collaboration with healthcare professionals and researchers is paramount. Participating in and contributing to collaborative research initiatives will keep our platform aligned with the latest advancements in diabetes research.

8. Enhanced Data Security Measures:

- Future work includes the implementation of even more robust data security measures. As we handle sensitive health information, we are committed to staying ahead of evolving cybersecurity challenges.

In conclusion, the fusion of machine learning and Django in our diabetes prediction platform stands as a testament to the transformative potential of interdisciplinary collaboration. Looking forward, we are poised to continue pushing the boundaries of innovation, ensuring that our platform remains at the forefront of technology-driven healthcare solutions. Through ongoing research, development, and user-focused enhancements, we aim to contribute meaningfully to the global efforts in diabetes prevention and public health improvement.

CHAPTER 8
References

References

1] Gauri D. Kalyankar, Shivananda R. Poojara and Nagaraj V. Dharwadkar," Predictive Analysis of Diabetic Patient Data Using Machine

Learning and Hadoop", International Conference On I-SMAC,978-1-5090-3243-3,2017.

[2] Ayush Anand and Divya Shakti," Prediction of Diabetes Based on Personal Lifestyle Indicators", 1st International Conference on Next

Generation Computing Technologies, 978-1-4673-6809-4, September 2015.

[3] B. Nithya and Dr. V. Ilango," Predictive Analytics in Health Care Using Machine Learning Tools and Techniques", International Conference

on Intelligent Computing and Control Systems, 978-1-5386-2745-7,2017.

[4] Dr Saravana kumar N M, Eswari T, Sampath P and Lavanya S," Predictive Methodology for Diabetic Data Analysis in Big Data", 2nd

International Symposium on Big Data and Cloud Computing,2015.

[5] Aiswarya Iyer, S. Jeyalatha and Ronak Sumbaly," Diagnosis of Diabetes Using Classification Mining Techniques", International Journal of

Data Mining & Knowledge Management Process (IJDKP) Vol.5, No.1, January 2015.

[6] P. Suresh Kumar and S. Pranavi "Performance Analysis of Machine Learning Algorithms on Diabetes Dataset using Big Data Analytics",

International Conference on Infocom Technologies and Unmanned Systems, 978-1-5386-0514-1, Dec. 18-20, 2017.

[7] Mani Butwall and Shraddha Kumar," A Data Mining Approach for the Diagnosis of Diabetes Mellitus using Random Forest Classifier",

International Journal of Computer Applications, Volume 120 - Number 8,2015.

[8] K. Rajesh and V. Sangeetha, “Application of Data Mining Methods and Techniques for Diabetes Diagnosis”, International Journal of

Engineering and Innovative Technology (IJEIT) Volume 2, Issue 3, September 2012.

[9]Humar Kahramanli and Novruz Allahverdi,”Design of a Hybrid System for the Diabetes and Heart Disease”, Expert Systems with

Applications: An International Journal, Volume 35 Issue 1-2, July, 2008.

[10] B.M. Patil, R.C. Joshi and Durga Toshniwal,”Association Rule for Classification of Type-2 Diabetic Patients”, ICMLC '10 Proceedings of

the 2010 Second International Conference on Machine Learning and Computing, February 09 - 11, 2010.

[11] Dost Muhammad Khan1, Nawaz Mohamudally2, “An Integration of K-means and Decision Tree (ID3) towards a more Efficient Data

Mining Algorithm ”, Journal Of Computing, Volume 3, Issue 12, December 2011