

KIGALI INDEPENDENT UNIVERSITY ULK
SCHOOL OF SCIENCE AND TECHNOLOGY
DEPARTMENT OF COMPUTER SCIENCE
P.O Box 2280 KIGALI

**DIABETES MODEL (SVM) PREDICTION USING MACHINE LEARNING
CASE STUDY: NATIONAL INSTITUTE OF DIABETES AND DIGESTIVE
AND KIDNEY DISEASES (ARIZONA, USA)**

DONE BY

Francis F. Taylor

Roll No: 202110416

Tel: +250 791569103

Supervisor: Dr. MUSABE Jean Bosco

**A dissertation submitted to the School of Sciences and Technology in partial
fulfillment of the academic requirements for the award of Bachelor's Degree
in Computer Sciences**

Kigali, August 2024

DECLARATION

I, **TAYLOR FRANCIS F**, thus certify that the study I have done for my thesis, "**DIABETES MODEL PREDICTION USING MACHINE LEARNING**," is original to me and was done on my own, with guidance from **Dr. MUSABE Jean Bosco** I certify that no other university or institution has accepted this research, in whole or in part, for the award of any other degree or diploma. Through cites and references, every information source and piece of literature used in the study has been appropriately acknowledged.

This study was carried out in compliance with the regulations and ethical standards established by **KIGALI INDEPENDENT UNIVERSITY (ULK)**.

Name: _____

Date: _____

Signed: _____

APPROVAL

This is to certify that the research work titled "**Diabetes Model Prediction Using Machine Learning**" has been conducted and completed by **Taylor Francis F.** under the supervision of

Dr. Musabe Jean Bosco. This research has been examined and is hereby approved as meeting the requirements for the degree of **BSc Computer Science at Kigali Independent University (ULK).**

Supervisor:

Name: _____

Signature: _____

Date: _____

DEDICATION.

In loving memory of my late mother, **Hawa Kollie**, and my beloved aunty, **Sando Kollie Johnson**, as well as my little sister, **Meldosia Angel Cox**, and other family members, I express my deepest gratitude for their unwavering support and guidance. I also extend my heartfelt thanks to the **Bensons family** for their steadfast support throughout this study. In addition to my friends and family, I dedicate this work to **Nennen Korlor**, my future wife and best friend, for her unshakable belief in me. Lastly, I am profoundly grateful to all of my professors and fellow students at **ULK** for their invaluable contributions to my academic journey.

ACKNOWLEDGEMENT

First and foremost, I want to express my gratitude to God for giving me the courage, wisdom, and direction I needed to complete my research.

Prof. Dr. RWIGAMBA BALINDA, the president and founder of ULK (Kigali Independent University), has my sincere gratitude for founding such a distinguished university and giving me the opportunity to achieve my academic goals.

I want to thank everyone who has helped me during this study project from the bottom of my heart. I have the utmost gratitude to my supervisor, **Dr. MUSABE Jean Bosco**, for his wise counsel, perceptive criticism, and unwavering support. The accomplishment of this effort has been made possible in large part by your knowledge and tolerance.

I would also like to thank the faculty and staff at Kigali Independence University (ULK) Computer Science Department for providing me with the resources and knowledge necessary to undertake this research. Your support has been vital to my academic growth.

Special thanks go to my colleagues and fellow students for their camaraderie and collaborative spirit, which made this journey both intellectually stimulating and enjoyable. Your constructive criticism and shared experiences have enriched my work.

I am deeply indebted to my family and friends for their unwavering support and understanding. To my late mother, **Hawa N. Kollie**, and my siblings, thank you for your endless love and encouragement. To my fiancée, **Nennen Korlor**, your belief in me has been my driving force, and I am forever grateful.

Finally, I would like to acknowledge all the participants and organizations who contributed to this study. Your willingness to donate your time and wisdom was essential to this study's accomplishment.

I appreciate your participation in this adventure, everyone.

TABLE OF CONTENT

DECLARATION.....	i
APPROVAL.....	ii
DEDICATION.....	iii
ACKNOWLEDGEMENT.....	iv
TABLE OF CONTENT	v
LIST OF FIGURES	xii
LIST OF TABLES.....	xii
ABBREVIATIONS AND ACRONYMS	xiv
ABSTRACT.....	xv
CHAPTER ONE: GENERAL INTRODUCTION	1
1.1 Introduction to the study	1
1.2 Background of the Project	1
1.3 Statement of the Problem.....	2
1.4 Objective of the project.....	2
1.4.1 General objective	2
1.4.2 Specific objectives	2
1.5. Research Questions.....	2
1.6. Scope of the Study	3
1.6.1. Content scope.....	3
1.6.2. Geographical scope.....	3
1.6.3. Time scope	4
1.7 Project methodology	4
1.8 Significance of the project	5

1.8.1. Personal Interest.....	5
1.8.2. Institutional Interest	5
1.8.3. Public Interest	6
1.9 Limitations of the project.....	6
1.10 Organization of the Project	7
CHAPTER TWO: LITERATURE REVIEW	8
2.0. Introduction.....	8
2.1. Definition of concepts.....	8
2.1.2. Overview of Diabetes prevalence and impact	9
2.1.3. Machine learning for Diabetes prediction.....	9
2.1.4. Challenges and Opportunities	9
2.1.4.1 Optimized Model Development.....	10
2.1.5 Machine Learning Algorithms and Data Preprocessing	10
2.1.6 Advanced Techniques	10
2.2.1 Data Preprocessing Techniques	10
2.2.2 Machine Learning Algorithms	11
2.2.3 Challenges and Solutions.....	11
2.2.4. Model Deployment	11
2.2.4.1. MLOps and Continuous Integration/Continuous Deployment (CI/CD).....	12
2.2.5. Deployment Patterns.....	12
2.2.5.1. Scaling and Optimization.....	12
2.2.5.2. Monitoring and Logging	12
2.2.5.3. Model Management	12
2.2.5.4. Advanced Deployment Strategies	13

2.2.6.1. Ensemble Learning Approaches	13
2.2.6.2. Diabetes Prediction Using Boosting Techniques	13
2.2.6.3. Voting Classifier for Diabetes Prediction:	13
2.2.6.4 Feature Combination and Classification.....	13
2.2.6.5. Optimized Model Development.....	13
2.3. Related works.....	14
2.4. Conclusion	15
CHAPTER THREE: SYSTEM ANALYSIS AND DESIGN.....	16
3.1. Introduction.....	16
3.2. Analysis of the Current System.....	16
3.2.1. Introduction.....	16
3.2.2. Problem of the Current System.....	17
3.3 Analysis of the new system.....	17
3.3.2 System requirements.....	18
3.3.2.1 Functional Requirements	18
3.3.2.2 Non-Functional Requirements	19
3.3.3 Functional Diagram	19
3.3.4 Software Development Methodology	20
3.3.4.1 Advantages of the Agile Model.....	20
3.3.4.2 System Analysis and Design Methodology	21
3.3.4.3. Data collection Techniques	21
3.3.4.3.1 Context Diagram.....	22
3.3.4.3.1. Data Flow Diagram (level 0)	23
3.3.4.3.2. Entity Relationship Diagram (ERD).....	25

3.3.4.3.3. Data Dictionary	26
CHAPTER IV: SYSTEM IMPLEMENTATION	30
4.1. Implementation and Coding.....	30
4.1.1. Introduction.....	30
4.1.2. Description of Implementation Tools and Technology	30
4.1.2.1. Python	30
4.1.2.2: HTML	31
4.1.2.3. CSS	31
4.1.2.4. JavaScript.....	31
4.1.2.5. Bootstrap.....	31
4.1.2.6. Django.....	32
4.1.2.7. SQLite	32
4.1.2.8. Support Vector Machine (SVM)	32
4.1.2.9. IDEs (Jupyter Notebook, and PyCharm)	33
4.2. Machine Learning Analysis from Dataset using Jupyter Notebook	33
4.2.1. Diabetes Dataset Analysis.....	33
4.2.2. Data Analysis (Cleaning, Preprocessing, EDA, Training, Testing and Model Evaluation)	35
4.2.3. Data Visualization	37
4.2.4. Outlier Detection on the datasets	40
4.2.5. Feature Engineering	41
4.2.6. Feature Scaling.....	41
4.2.7. Training and Testing Dataset.....	42
4.2.8. Model Evaluation for Support Vector Machine	42
4.2.9. Logistics Regression model	43

4.2.10. K-Nearest Neighbor Model.....	44
4.2.11. Decision Tree Model.....	44
4.2.12. Random Forest Model.....	45
4.2.13. Machine Learning Evaluation Metrics Performance for Classification Models.	45
4.2.14. Model Comparison.....	46
4.2.15. Saving Mode Using Support Vector Classifier	47
4.2.16. Receiver Operating Characteristic (ROC)	47
4.2.17. Area Under Curve (AUC)	48
4.3.3. Screen shorts and source codes	49
4.3.3.1. Home Menu	49
4.3.3.2. Contact Us Screen Shot	49
4.3.3.3. Signup/ Create user screenshot	50
4.3.3.4. User Sign-in Screenshot.....	50
4.3.3.5. Admin Sign-in.....	51
4.3.3.5. Admin Screen.....	51
4.3.3.6. Changed Password.....	52
4.3.3.7. Predictions.....	52
4.3.3.8. About Us	53
4.3.3.9. Datasets Screenshot	53
4.4. Testing.....	54
4.4.1. Introduction.....	54
4.4.1.1. Objectives of Testing:	54
4.4.2. Unit Testing Outputs	54
4.4.3. Validation Testing outputs.....	56

4.4.4. Integration Testing outputs.....	58
4.4.5. Functional and System Testing	59
4.4.6. Acceptance testing report.....	61
4.4.7. Comparative Results	63
CONCLUSIONS AND RECOMMENDATIONS	64
5.1 Conclusion	64
5.2 Recommendations	64
5.3 Future work.....	65
REFERENCES.....	66
APPENDICES	

LIST OF TABLES

Table 1: Myapp_history	26
Table 2. Myapp_register	27
Table 3. Auth_permission.....	27
Table 4. Auth_user	27
Table 5. Auth_user_user_permission.....	27
Table 6. Auth_user_groups	28
Table 7. Auth_user_group_permissions.....	28
Table 8. Auth_group.....	28
Table 9. Django_admin_log.....	28
Table 10. Django_content_type	28
Table 11. Django_migrations	29
Table 12. Django_session	29
Table 13: Diabetic Dataset	34
Table 14: Exploratory Data Aanalysis	345
Table 15: Dataset Information	346
Table 16: Descriptive Statistics of the Dataset	346
Table 17: Datasets Screenshot	52
Table 18: Unit Testing Cases and Results	55
Table 19: Validation Test Cases and Results.....	57
Table 20: Integration Test Cases and Results.....	58
Table 21: Functional Testing Case and Results	60
Table 22: Acceptance Testing Scenarios	62
Table 23: Comparative Results	63

LIST OF FIGURES

Figure 1: Functional Diagram	19
Figure 2: Context Diagram	22
Figure 3: DFD Level 0 and level 1	23
Figure 4: Extension of data flow diagram level 1	24
Figure 5: Entity Relationship Diagram	25
Figure 6: Libraries for Data Analysis.....	35
Figure 7: Load Dataset.....	35
Figure 8: Dataset Outcome	37
Figure 9: Correlation Matrix.....	49
Figure 10: Dataset Missing Values	49
Figure 11: After Filling Missing Values.....	39
Figure 12: Outlier Detection	39
Figure 13: Feature Engineering	490
Figure 14: Feature Scaling.....	490
Figure 15: Dataset Training and Testing	491
Figure 16: SVM Graph	491
Figure 17: Support Vector Machine Outputs	492
Figure 18: Logistic Regression Outputs	492
Figure 19: K-Nearest Neighbor Outputs.....	493
Figure 20: Decision Tree Outputs	493
Figure 21: Random Forest Outputs.....	494
Figure 22: Model Comparison.....	495
Figure 23: Saving Model	496
Figure 24: Receiver Operating Curve	496

Figure 25: Area Under Curve.....	497
Figure 26: Home Screen	49
Figure 27: Contact us Screen	498
Figure 28: Signup Screen.....	50
Figure 29: User Sign-in.....	50
Figure 30: Admin Sign-in	51
Figure 31: Admin Home Screen	51
Figure 32: Change Password	52
Figure 33: Predictions Screen	52
Figure 34: About Screen	53

ABBREVIATIONS AND ACRONYMS

AI: Artificial Intelligent

AUC: Area Under Curve

BMI: Body Mass Index

CSS: Cascading Style Sheets

DB: Database

DL: Deep Learning

DT: Decision Tree

EDA: Exploratory Data Analysis

EDA: Exploratory Data Analysis

HTML: Hypertext Markup Language

JSON: JavaScript object Notation

LIME: Local Interpretable Model-Agnostic Explanations

LR: Logistics Regression

ML: Machine Learning

MLDB: Machine Learning Database

MLOPS: Machine Learning Operations

NIDDK: National Institute of Diabetes and Digestive and Kendy Disease

RF: Random Forest

ROC: Receiving Operating Characteristics

SSADM: Structured Systems Analysis and Design Method

SVM: Support Vector Machine

ULK: Kigali Independent University

ABSTRACT

A chronic illness that affects millions of individuals worldwide, diabetes poses serious health hazards if left untreated. Improving patient outcomes and limiting serious consequences can be achieved by accurately predicting diabetes and detecting it early. This research aims to create a diabetes prediction model by applying the Support Vector Machine (SVM) algorithm, a potent supervised machine learning method well-known for its efficiency in classification applications. A dataset with numerous health-related characteristics, including blood pressure, body mass index (BMI), age, and glucose levels, was used to train and assess the model. Making a trustworthy and accurate algorithm that could identify people as either diabetics or non-diabetics based on their data was the main goal.

The model was created using Python and other libraries, including Scikit-learn for machine learning tasks, Pandas and NumPy for data processing, and Matplotlib and Seaborn for data visualization. In addition, a Django web application with HTML, CSS, Bootstrap, and JavaScript was created to offer an interactive user interface for data entry and prediction viewing. The backend database utilized to store user data and prediction history was SQLite. The outcomes showed that the SVM model had a good accuracy rate, which made it a useful tool for clinical diabetes prediction. This paper offers a useful implementation methodology for incorporating predictive models into user-friendly online apps and highlights the possibilities of machine learning in the healthcare industry.

To further enhance the accuracy and generalizability of the diabetes prediction model, future research should consider integrating additional features such as genetic data and lifestyle factors, as well as exploring more complex algorithms like deep learning models. Implementing the system in a real clinical setting could also help validate its effectiveness in practice. Additionally, expanding the scope of the study to include different population demographics could ensure that the model is applicable across diverse patient groups. Regular updates and improvements based on user feedback and advances in machine learning techniques are essential to maintaining the system's relevance and effectiveness in predicting diabetes.

CHAPTER ONE: GENERAL INTRODUCTION

1.1 Introduction to the study

Diabetes is a chronic metabolic disorder characterized by elevated blood glucose levels due to the body's inability to produce or effectively use insulin, a hormone essential for regulating blood sugar [1]. This condition is a significant public health concern, with an estimated 135 million people affected worldwide, the number expected to rise to 300 million by 2025 [2].

Machine learning techniques have shown promise in the early prediction and diagnosis of diabetes [3] [4] [2]. One such approach, the support vector machine, has been identified as a particularly effective classifier for predicting diabetes [2]. This study aims to investigate the application of a support vector machine in predicting diabetes outcomes using a case study of the PIMA Indians Diabetes Dataset.

A Python Django web application called the Diabetes Prediction System using a Support Vector Machine makes use of an SQLite database to assess a person's risk of developing diabetes depending on several different factors. The project has two main parts: user and admin. The user module allows users to make predictions, edit their profiles, and log out. The admin module allows administrators to view user and prediction results. The system uses a Support Vector Machine to provide accurate predictions and manage user accounts efficiently.

1.2 Background of the Project

Diabetes is a chronic condition with significant health risks, particularly if left unmanaged. Early prediction and diagnosis are essential in managing diabetes and preventing complications. The Pima Indians Diabetes dataset, collected by the National Institute of Diabetes and Digestive and Kidney Diseases (NIDDK), is widely used in diabetes prediction research. This dataset contains health-related information from 768 Pima Indian dataset that is commonly utilized as a benchmark for testing and evaluating machine learning models designed to predict diabetes.

The dataset includes eight key features: pregnancies, plasma glucose concentration, diastolic blood pressure, skinfold thickness, serum insulin levels, BMI, diabetes pedigree function, and age. While valuable, the dataset faces challenges, such as data imbalance and a limited number of features, which can impact the accuracy and generalizability of predictive models. The current research addresses these issues by applying techniques like SMOTE for balancing data, feature engineering

to enhance the dataset, and regularization to prevent overfitting. Additionally, efforts to integrate diverse datasets aim to improve model applicability across different populations, contributing to more effective diabetes prediction and prevention strategies.

1.3 Statement of the Problem

Despite the increasing prevalence of diabetes, there is a lack of comprehensive data-driven approaches to predicting and managing the disease. The application of machine learning techniques, such as the Support Vector Machine (SVM), can provide valuable insights and improve the accuracy of diabetes prediction, allowing for earlier intervention and more effective treatment [2].

1.4 Objective of the project

The primary objective of this study is to evaluate the effectiveness of using a support vector machine model in predicting diabetes outcomes among patients at the PIMA Indians Dataset.

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.

1.4.1 General objective

The general objective of the project is to explore, analyze, and preprocesses strategies datasets for the design and implement Diabetes prediction using machine algorithms, aiming to enhance the predictions of diabetes disease in the early stages.

1.4.2 Specific objectives

- i. To develop a system that can predict diabetes.
- ii. To Train and Test Machine Learning model for accurate predictions.
- iii. To design a user-friendly interface that allows users to interact with the system.

1.5. Research Questions

- i. How to develop a system that can predict diabetes?
- ii. What Machine Learning Technique is optimal to create a predictive model for accurate diabetes prediction?

- iii. What are the Optimal validation metrics to compare the models' performance?
- iv. How will the interface be designed to promote user-friendly interaction with the system, incorporating usability principles and ensuring accessibility for all users?

1.6. Scope of the Study

The scope of this study involves utilizing the Pima Indians Diabetes dataset to develop and evaluate machine learning models for predicting diabetes risk. It includes addressing challenges such as data imbalance and overfitting, enhancing model accuracy through advanced feature engineering and data augmentation, and improving interpretability to ensure that predictions are transparent and understandable. The study aims to identify the most effective predictive models, assess their generalizability to diverse populations, and explore their practical implications for early diabetes detection and prevention in real-world healthcare settings.

1.6.1. Content scope

The study will cover the following aspects:

Using the support vector machine model, the study will evaluate the accuracy and performance of predicting diabetes outcomes among patients at the PIMA Indians Diabetes Dataset, and compare its effectiveness to other machine learning algorithms, such as random forest and logistic regression.

The study will identify the key risk factors and variables that contribute to the development of diabetes, as determined by the support vector machine model, and provide insights into the underlying drivers of the disease in the given context.

The study will provide recommendations for the effective implementation of the support vector machine-based diabetes prediction model at the PIMA Indians Diabetes Dataset to enhance early detection and management of the disease.

1.6.2. Geographical scope

The geographical scope of this study focuses on the Pima Indian population residing in the Phoenix, Arizona area, where the original diabetes dataset was collected. By using this specific demographic, the study aims to understand diabetes prediction within this particular population's context.

However, the research also considers the broader applicability of the developed models by evaluating their performance on data from diverse populations and geographical regions. This approach ensures that the findings and predictive models can be generalized beyond the Pima Indian community, contributing to more universal strategies for diabetes detection and management across different demographic and geographic settings.

1.6.3. Time scope

The time scope of this study encompasses the period from the collection of the Pima Indians Diabetes dataset, which dates back to the early 1990s, up to the present day. This includes an analysis of historical data and its relevance in the context of current advancements in machine learning and predictive modeling. The research also evaluates how recent developments and techniques in data science and healthcare can enhance diabetes prediction using this dataset. By bridging the historical data with contemporary methods, the study aims to provide insights that are both relevant to the past and applicable to current and future diabetes management practices.

1.7 Project methodology

The project methodology for a diabetes prediction model using machine learning typically begins with problem definition and data collection. In this stage, the main goal is to predict whether a patient has diabetes based on various health-related features such as glucose levels, BMI, insulin levels, etc. After defining the objective, a literature review is conducted to identify how previous models have approached diabetes prediction and to understand the challenges they faced. Once data is collected, it undergoes preprocessing, including cleaning, handling missing values, and normalizing features, which ensures that the dataset is suitable for training. The data is then split into training and testing sets to evaluate the model fairly.

The next step involves selecting appropriate machine learning algorithms such as Support Vector Machines (SVM), Logistic Regression, Decision Trees, and Random Forest. The model is trained using the training set, and its performance is evaluated using metrics like accuracy, precision, recall, and the F1-score. After initial evaluation, hyperparameter tuning and feature engineering are applied to optimize the model's performance. The finalized model can be deployed into a health management system or a web application to assist in real-time diabetes prediction. Additionally, tools like LIME can be integrated to make the model's decisions interpretable to healthcare professionals.

1.8 Significance of the project

This study's findings will contribute to the ongoing efforts to improve diabetes management and early detection in resource-limited settings. The application of the support vector machine model in this context can provide valuable insights into the key risk factors and variables that contribute to the development of diabetes, enabling targeted interventions and more effective disease management strategies [5].

Moreover, the comparative analysis of the support vector machine model's performance against other machine learning algorithms will provide evidence-based guidance for healthcare providers and policymakers in selecting the most appropriate predictive model for diabetes management in similar settings [9].

The successful implementation of the support vector machine-based diabetes prediction model at the PIMA Indians Diabetes Dataset can serve as a case study for other healthcare facilities in the region, potentially leading to the adoption of more effective and data-driven approaches for early detection and management of diabetes [8].

1.8.1. Personal Interest

I have always been focused by the significant uses of Artificial Intelligent, Machine Learning, and Deep Learning in healthcare as a growing Data scientist. I think that by predicting and diagnosing diseases like diabetes using machine-learning techniques, we can greatly enhance patient outcomes and progress healthcare as a whole. Furthermore, the demand for precise and effective prediction models is urgent due to the rising incidence of diabetes and its complications globally. I am very interested in this research on diabetes prediction using machine learning techniques because it reflects my passion for applying data-driven methods to solve challenging real-world issues.

1.8.2. Institutional Interest

Institutional interest in this study lies in advancing the field of diabetes prediction and management through the application of machine learning techniques. Institutions such as healthcare organizations, research institutions, and universities are keen on improving early diagnosis and prevention strategies for diabetes, a condition with significant public health implications. By leveraging the Pima Indians Diabetes dataset and applying cutting-edge data science methods, the study aligns with institutional goals of enhancing clinical decision-making, developing robust

predictive models, and contributing to evidence-based healthcare practices. The findings are expected to support institutional efforts in implementing more effective diabetes management solutions and advancing research in the intersection of healthcare and technology.

1.8.3. Public Interest

The process of using machine learning in healthcare and others areas will benefit the public in several ways. By accurately determining a person's risk for diabetes, the program can aid in the disease's early detection and prevention. This can reduce the prevalence and burden of diabetes in the general population by enabling people to seek the appropriate attention and alter their lifestyles as needed. The project can also advance customized therapy by identifying specific risk factors and genetic markers associated with diabetes. By identifying these components, people can receive personalized interventions and treatment plans that will improve their health outcomes. Policy and administration in healthcare may also benefit from the project.

With the use of accurate diabetes risk projections and insights, policymakers can make well-informed decisions regarding public health campaigns, preventative actions, and budget allocation. Both the broad growth of machine learning and its applications in healthcare can be aided by the public. The project's implementation at PIMA Indians Diabetes Dataset Hospital can serve as a case study for researchers and other healthcare organizations, inspiring them to adopt similar tactics in their own settings.

1.9 Limitations of the project

The study's retrospective design introduces potential biases like information bias, which are common in this type of research. Additionally, the study's reliance on historical patient data from the PIMA Indians Diabetes Dataset may be constrained by data completeness and accuracy. Furthermore, the findings may have limited generalizability, as they are specific to the context of this Dataset, necessitating further research to evaluate the applicability of the support vector machine model in other healthcare settings in or the surrounding region.

1.10 Organization of the Project

This study is structured and articulated into five chapters sequentially:

Chapter 1: INTRODUCTION TO THE STUDY

This chapter gives an introduction and background of this research. It introduces the main research purpose including, the statement of the problem, Research objectives, research questions, scope and the limitations.

Chapter 2: LITERATURE REVIEW

The main purpose of this chapter is to describe the key terms or concepts used in our study, to review the existing related systems and how previous researchers addressed data exchange problems.

CHAPTER 3: SYSTEM ANALYSIS AND DESIGN

This chapter will present the analysis of the system Vs the new system to be implemented along with research methodologies used and the system design and overview.

Chapter 4: SYSTEM IMPLEMENTATION

In this chapter will describe the tools and technologies used for implementation and system implementation flow and Specifications.

Chapter 5: CONCLUSION AND RECOMMENDATIONS

This chapter offers the conclusion of the study and suggestions to call institutions to adhere on the importance and advantages of using machine learning algorithms in Organization.

CHAPTER TWO: LITERATURE REVIEW

2.0. Introduction

This chapter presents a comprehensive review of existing literature on the application of machine learning, particularly the support vector machine algorithm, in predicting and managing diabetes.

2.1. Definition of concepts

Diabetes: Diabetes is a chronic metabolic disorder characterized by high blood sugar levels (hyperglycemia) due to the body's inability to produce sufficient insulin or effectively use the insulin it makes. If left untreated, it might result in major side effects like heart disease, renal failure, and visual loss [5].

Machine Learning: Machine learning is a subset of artificial intelligence (AI) that entails giving computers the ability to learn from data and make predictions or judgments without having to be specifically programmed to do so by employing statistical models and algorithms. It is extensively employed in many different fields, including marketing, finance, and healthcare.

Support Vector Machine (SVM): Support Vector Machine (SVM) is a supervised machine learning algorithm used for classification and regression tasks. It works by finding a hyperplane that best separates the data points of different classes in a high-dimensional space, maximizing the margin between them. SVM is known for its effectiveness in handling complex classification problems [25].

Pima Indians Diabetes Dataset: The Pima Indians Diabetes Dataset is a publicly available dataset that contains health-related data from 768 Pima Indian women, including features such as blood pressure, glucose levels, BMI, and age. It is widely used in machine learning research for diabetes prediction due to its clear binary classification outcome: diabetic or non-diabetic [23].

Prediction Model: A prediction model in machine learning is a statistical or computational tool designed to predict an outcome or category based on input data. In the context of diabetes prediction, the model analyzes health-related features to classify individuals as either diabetic or non-diabetic [24].

Django: Django is a high-level Python web framework that allows for the rapid development of secure and maintainable websites. In this study, Django was used to build a web application that provides an interactive interface for users to input their health data and receive predictions about their diabetes status.

Scikit-learn: is an open-source Python machine learning framework that offers effective and user-friendly tools for data mining and analysis. It includes various classification, regression, and clustering algorithms, making it a go-to library for machine-learning tasks.

SQLite: SQLite is a lightweight, serverless, and self-contained database engine that is commonly used in embedded systems and applications. In this research, SQLite was used to store user data and prediction history for the web application.

2.1.2. Overview of Diabetes prevalence and impact

Diabetes is a global health concern, with an estimated 537 million adults living with the condition as of 2021 [9]. The prevalence of diabetes is particularly high in developing countries, where access to healthcare and early detection mechanisms may be limited. The PIMA Indians Diabetes Dataset serves as a significant population and plays a crucial role in managing the growing burden of diabetes in the region [4].

2.1.3. Machine learning for Diabetes prediction

The application of machine learning, specifically the support vector machine algorithm, has shown promising results in predicting and managing diabetes. These studies have demonstrated the superiority of the support vector machine model in terms of sensitivity, specificity, and overall classification accuracy when compared to other machine learning algorithms.

2.1.4. Challenges and Opportunities

While the support vector machine algorithm has been successfully applied in diabetes prediction, the implementation of such models in resource-limited settings, such presents unique challenges.

The availability and quality of patient data, the requirement for strong feature engineering, and the incorporation of these models into clinical decision-making processes are just a few of the obstacles that must be overcome in order to fully realize the potential advantages of employing machine learning for diabetes prediction [10].

However, especially in environments with limited resources like PIMA and other areas, there are opportunities to construct more accurate and dependable prediction models for diabetes treatment due to the increased availability of digital health data and the growing computational capacity of current computing systems [11].

2.1.4.1 Optimized Model Development

Studies have developed optimized models using ensemble techniques and two-stage model selection processes. These models are fine-tuned for better accuracy, utilizing advanced techniques like data balancing and feature selection [12].

2.1.5 Machine Learning Algorithms and Data Preprocessing

Studies highlight the use of logistic regression, SVM, and neural networks. Effective preprocessing methods like data normalization and oversampling techniques like SMOTE are also essential to enhance model performance [13].

2.1.6 Advanced Techniques

Recent research includes comprehensive reviews of machine learning methods for diabetes prediction, emphasizing the importance of feature selection, hyperparameter tuning, and cross-validation techniques [3].

2.2.1 Data Preprocessing Techniques

Handling Missing Values: This involves strategies like removing rows with missing values or imputing them using statistical measures such as mean, median, or mode [14]

Encoding Categorical Data: Transforming non-numeric data into numeric formats using techniques like one-hot encoding or label encoding to make them suitable for machine learning algorithms [15].

Feature Scaling: is a vital preprocessing step in machine learning that involves transforming numerical features to a common scale. Common methods include min-max scaling, standard scaling (Z-score normalization), and robust scaling, especially when dealing with outliers [15].

Splitting Datasets: Dividing data into training, validation, and test sets to properly evaluate model performance [14]

2.2.2 Machine Learning Algorithms

Supervised Learning Algorithms: Such as linear regression, logistic regression, and support vector machines which benefit significantly from well-preprocessed data.

Unsupervised Learning Algorithms: Such as k-means clustering rely on the normalization of features for accurate distance measurements.

Advanced Techniques: Including ensemble methods like Random Forests and Gradient Boosting Machines, which combine multiple learning algorithms to improve predictive performance [14][15].

2.2.3 Challenges and Solutions

Imbalanced Data: Techniques like oversampling, under-sampling, and hybrid methods (e.g., SMOTE) are used to address class imbalances in datasets [14].

Feature Engineering: Creating new features from existing data to enhance model performance, such as decomposing date-time features into more meaningful components like "day of the week" or "time of day" [14].

Dealing with High Cardinality: For categorical variables with many unique values, methods like frequency encoding or dimensionality reduction techniques like PCA can be employed to manage computational complexity [15].

These preprocessing steps and algorithmic choices are fundamental to building robust machine learning models, ensuring that the data fed into these models is clean, well-structured, and appropriately scaled for the best performance. For a detailed exploration of these topics, you can refer to the latest guidelines and best practices on platforms like Pure Storage and Lakefs [14][15].

2.2.4. Model Deployment

Model deployment is a critical phase in the machine learning lifecycle, involving the transition of models from development to a production environment. This phase ensures that the models are accessible to end-users and applications in a scalable, reliable, and efficient manner.

2.2.4.1. MLOps and Continuous Integration/Continuous Deployment (CI/CD)

MLOps extends traditional DevOps practices to machine learning, incorporating tasks such as model training, testing, deployment, and monitoring. These procedures are automated by CI/CD pipelines, guaranteeing dependable and consistent updates and deployments. For this, programs like GitHub Actions, GitLab CI, and Jenkins are frequently used [16].

2.2.5. Deployment Patterns

Single-Model Endpoints: This pattern involves deploying one model to serve all predictions, suitable for initial deployments. Once the model is stable and validated, it can handle the prediction traffic for business applications [16].

Model Variants: Tools like Amazon SageMaker support deploying multiple versions of a model simultaneously scaling, enabling A/B testing and shadow testing. This approach helps in comparing model versions under real-world conditions without affecting production traffic [16].

2.2.5.1. Scaling and Optimization

Horizontal scaling (adding more instances) and vertical scaling (enhancing existing instances) ensure that the model can handle increased demand. Load balancers like NGINX and HAProxy distribute incoming requests to improve reliability and performance [17].

Optimizing models through techniques like quantization, pruning, and distillation reduces computational requirements, making them suitable for deployment on resource-constrained devices [17].

2.2.5.2. Monitoring and Logging

Effective monitoring and logging are crucial for maintaining model performance in production. Tools like Prometheus and Grafana provide real-time insights into model behavior; while logging frameworks such as the ELK Stack capture detailed operational data for debugging and analysis [17].

2.2.5.3. Model Management

Platforms like MLflow offer comprehensive management of the machine learning lifecycle, including experiment tracking, model packaging, and deployment. MLflow supports various

deployment options, such as REST API serving, batch inference, and real-time streaming, and integrates with tools like Jupyter Notebooks, Apache Spark, and Kubernetes [18].

2.2.5.4. Advanced Deployment Strategies

Workflows for data science can be built and implemented more easily with Package, thanks to tools like Netflix's Metaflow. Metaflow offers a modular approach to process organization, allows versioning and caching, and connects smoothly with cloud systems. [18].

2.2.6.1. Ensemble Learning Approaches

This method combines multiple algorithms to improve prediction accuracy. Techniques like boosting algorithms, extreme gradient boosting (XGBoost), and random forests have shown promising results [13].

2.2.6.2. Diabetes Prediction Using Boosting Techniques

An ensemble learning approach combining multiple boosting algorithms has shown high accuracy in predicting diabetes. Techniques like XGBoost, AdaBoost, and gradient boosting improve prediction accuracy by combining weak learners into a strong predictor [13].

2.2.6.3. Voting Classifier for Diabetes Prediction:

A study used a voting classification based on the ensemble method to predict diabetes. This approach combined several algorithms to achieve higher accuracy, utilizing the Pima Indians Diabetes dataset for validation [13].

2.2.6.4 Feature Combination and Classification

The application of ensemble learning to improve illness prediction has been studied. Combining different features and classifiers, such as in the XGBoost model, resulted in improved accuracy and performance [13].

2.2.6.5. Optimized Model Development

Studies have developed optimized models using ensemble techniques and two-stage model selection processes. These models are fine-tuned for better accuracy, utilizing advanced techniques like data balancing and feature selection [13].

2.3. Related works

The existing writing on the utilize of support vector machines for diabetes expectation gives a solid establishment for this consider. A ponder in Indonesia compared the execution of support vector machines and altered adjusted arbitrary woodland calculations in identifying diabetes patients, finding that the support vector machine demonstrates berated the other strategy in terms of precision, affectability, and specificity [15]. Another think about in Iran inspected the real-world execution of different data mining strategies, counting support vector machines, in prediction diabetes. The analysts concluded that the support vector machine show positioned to begin with among the tried classifiers in terms of affectability, specificity, and by and large classification precision, making it a promising approach for diabetes expectation [16]. Ahamed B. Arya M, Nancy V A have made a demonstrate to identify whether an individual is influenced with DM illness. The concept of machine learning (ML) is utilized for the location strategies. The PIMA dataset is utilized for the consider. The calculations utilized are LR, Vaive bayes (NB), and K-nearest neighbor (KNN). The exactness gotten are 84, 79, and 69% from these calculations. The measures such as exactness, review, and F-measure are taken into thought and LR is considered to deliver the most elevated exactness [18].

Ahmed N and others in 2021, have utilized ML calculations, specifically, DT, KNN, NB, RF, GB, LR, and bolster vector machine (SVM) for anticipating Diabetes Mellitus. Preprocessing strategies, such as label–encoding–normalization, are utilized to increment the precision. Two diverse datasets are utilized. One dataset gives the most elevated exactness for SVM with 80.26% and for the moment dataset, the most noteworthy precision is given by DT and RF with 86.81% [19].

Kumari S, have utilized two datasets counting PIDD and breast cancer dataset, which were taken from the UC Irvine (UCI) Store. Three ML classifiers are utilized for forecast. They are RF, LR, and Naive Bayes. The precision gotten is the most noteworthy for both datasets with a rate of 79.08% for PIMA information and 87.27% for breast cancer information utilizing delicate voting classifier [19].

Pélagie Hounguè A have created a prediction show for DM malady. A dataset was collected for the consider comprising of 952 occurrences and 18 properties. The machine learning classifiers utilized are RF, LR, KNN, SVM, NB, and DT. The accuracy score gotten was the most noteworthy

for RF with a rate of 84.10% for collected information and 75% for PIMA dataset [20]. The Stacked Generalization approach will be utilized in this work to make strides the execution of unremitting illness predicted models. In this work, the stacking generalization collecting method was utilized for five essential classifiers: Support Vector Machines (SVM), K-Nearest Neighbors (KNN), Logistic Regression (LR), Naive Bayes (NB), and Decision Tree (DT) with cross-validation. The test comes about uncover that the Stacked Generalization procedure moves forward forecast framework unwavering quality, specificity, affectability, positive esteem for expectation, and antagonistic prescient esteem by minimizing change blunder and anticipating overfitting. The expectation models made in this work seem be utilized in the future to productively analyze five unremitting ailments in essential care settings: cardiovascular illness, diabetes, breast cancer, hepatitis, and inveterate kidney infection [21].

2.4. Conclusion

The writing audit highlights the solid establishment of existing inquire about on the utilize of support vector machines (SVM) for diabetes prediction, reliably illustrating the model's prevalent execution compared to other machine learning calculations over different thinks about. SVM models have appeared higher precision, affectability, and specificity in predicting diabetes in distinctive datasets and settings, recommending their vigor and adequacy in this space. Also, the utilize of other calculations like calculated logistics regression, decision tree, Navie bayes, and K-nearest neighbor in combination with preprocessing strategies advance upgrades precision. The stacked generalization approach, which combines numerous classifiers, too appears guarantee in making strides prediction framework unwavering quality and minimizing mistakes, making it a profitable methodology for unremitting illness determination, counting diabetes. In general, the survey underscores the potential of SVM and outfit strategies in creating dependable prediction models for diabetes and other constant maladies.

CHAPTER THREE: SYSTEM ANALYSIS AND DESIGN

3.1. Introduction

Requirement analysis focuses on the task that determines the needs or conduction to meet new projects taking account of the possible conflicting requirements of stakeholders, analyzing, documenting, validating, and managing system requirements. It encompasses every action taken to identify different stakeholders' needs.

Analysis is the process of looking at the different parts of a topic, how they fit together, and what actions to take. This means breaking things down into their elements and answering the WHY & HOW using your brain. It analyses, reviews the target existing system, and notes down its requirements. System analysis gives analysts more freedom in comprehending the goal and operations of the current system.

Data analysis is a process of examining, cleaning, transforming, and modeling data to discover useful information; conclusion-drawing support; and decision-making. Data analysis consists of using a variety of techniques under different titles in disparate sectors, such as science or social sciences. Data Mining is a subset of data analysis that includes some parts related to knowledge gathering and modeling to predict what might happen. This research will push the boundaries of machine learning-based diabetes prediction utilizing a variety of data analysis methodologies. Finally using the system analysis and design (tools & techniques) introduced earlier, this chapter will dissect the present-day predictive health delivery script by looking at problems encountered or faced today by the public/patients/healthcare professionals in providing healthcare possibly with new designs for a modified approach towards operating within those systems while drawing up functions diagrams, Entity-relationship chart also ER diagram as popularly termed, and Data flow diagram.

3.2. Analysis of the Current System

3.2.1. Introduction

The existing manual system for diabetes diagnosis involves a time-consuming and subjective process of evaluating symptoms and conducting physical examinations. Doctors depend on old ways of knowing if a person has diabetes or not but with the help of machine learning these problems can be solved. When common knowledge is lacking, studies are summarized after some

numbers of cases have been studied, we notice that the manual system is not accurate enough to predict well.

3.2.2. Problem of the Current System

The Dataset faces several challenges, including weak institutional capacity and governance issues that impede effective administration. The absence of a database management system (DBMS) leads to data loss, as patient information cannot be properly stored. Additionally, significant time is wasted in predicting patient status, as staff must visit offices and halt other activities to consult with authorities. The current system is also vulnerable to corruption, making it easy to falsify patient results through bribery, which can leave patients in serious jeopardy.

3.3 Analysis of the ew system

The Diabetes Model Prediction System utilizing a Support Vector Machine seeks to forecast a person's risk of getting diabetes by applying machine learning techniques, particularly the Support Vector Machine algorithm. The system aims to assist healthcare professionals and individuals in assessing the risk of diabetes early on, enabling timely interventions, personalized care, and preventive measures.

The primary purpose of the system can be summarized as follows

1. Early Detection: The system aims to identify individuals who are at high risk of developing diabetes at an early stage. By analyzing various parameters and patterns from input data, the system provides predictions that can help in the early detection and prevention of diabetes-related complications.
2. Risk Assessment: The system assesses the individual's risk of developing diabetes based on their demographic and health-related parameters. This allows healthcare professionals to provide personalized recommendations, lifestyle modifications, and interventions to manage and reduce the risk of diabetes.
3. Support Decision-Making: The system provides healthcare professionals with a data-driven tool to support their decision-making process. By considering multiple factors and utilizing the power of the Support Vector Machine algorithm, the system offers accurate predictions that can aid in clinical decision-making and treatment planning.

4. Patient Empowerment: The system empowers individuals by providing them with insights into their potential risk of diabetes. It encourages individuals to take proactive steps toward preventive measures, such as adopting a healthy lifestyle, monitoring their health parameters, and seeking appropriate advice.

5. Research and Public Health: The system generates valuable data and insights that can contribute to research on diabetes, risk factors, and disease management. The collected data can be analyzed to identify trends, patterns, and correlations, helping in the formulation of effective public health strategies and policies related to diabetes prevention and control.

Our proposed Diabetes Model Prediction System overcomes these limitations by automating the diagnosis using machine learning algorithms, specifically Support Vector Machine (SVM). By analyzing relevant features such as age, glucose levels, BMI, and family history, our system provides accurate and objective predictions, leading to early detection of diabetes. This data-driven approach improves the accuracy of diagnosis, reduces the chances of misdiagnosis, and allows for timely intervention and management. Additionally, our web-based application ensures accessibility to healthcare professionals and patients, regardless of their geographical location, thereby improving healthcare outcomes.

3.3.2 System requirements

3.3.2.1 Functional Requirements

These define what the system should do in terms of specific functionalities or features it must possess to meet user needs and stakeholder objectives. Below are the functional requirements for a Diabetes Model prediction System using machine learning.

Users will need to enter their username and password to log in to the system as part of the user authorization process.

- Users will be able to register for accounts on the system.
- The system can store all the information and retrieve it like a user password.
- The system will be able to backup users' information.
- The user will be able to log out after they finish using the system.
- Users will be able to examine and print the predictions.

- The system will allow the admin to view and make decisions.

3.3.2.2 Non-Functional Requirements

These specify the criteria that characterize how well the system performs its functions, including aspects such as security, reliability, performance, usability, scalability, and compliance.

- Security: All user information is secure and stored in the database.
- User-friendly: The system will be user-friendly, understandable, and easy to use.
- Privacy: The system shall be able to protect the user's privacy
- Availability: All users should have round-the-clock access to the system.
- Performance: The system must have a quick performance and should be able to respond to requests in a reasonable amount of time.
- Accessibility: Patients can access their results from any location (as long as they are within a network service reception area).
- Recoverability: The system should be able to recover from any disturbance.
- Environmental: The system should be able to run on an Android operating system

3.3.3 Functional Diagram

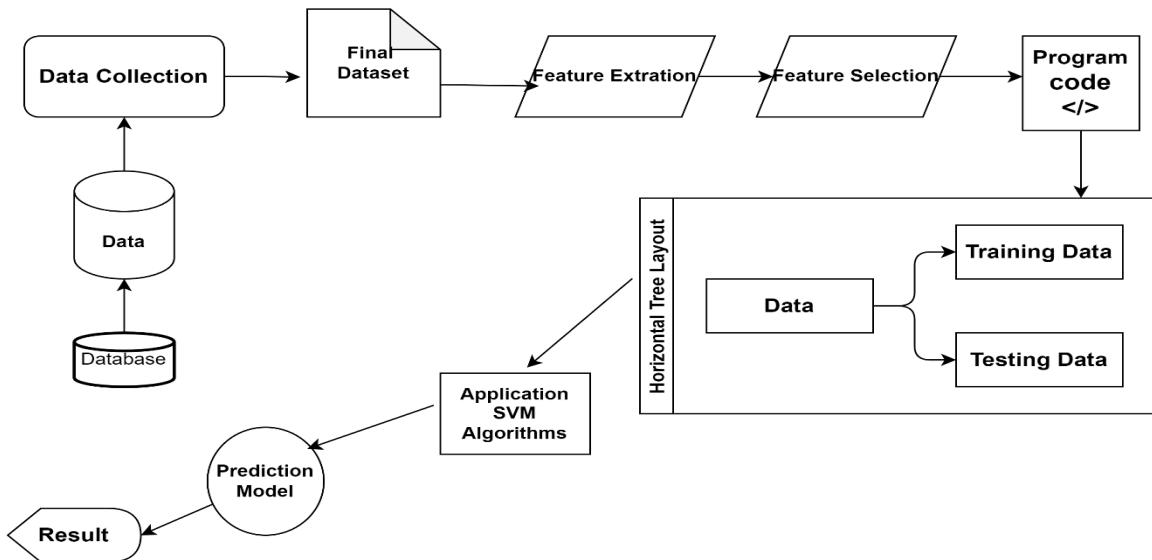


Figure 1: Functional Diagram

Initially, the study's data are gathered and entered into the database. The JFK datasets and the PIMA dataset are sourced from Kaggle for implementation. Following that, the dataset is pre-processed using several exploratory data analysis methods. The dataset will be divided into two 80% for training and 20% for testing. The best working algorithm generating the highest accuracy is then determined to be the best predictive model for Diabetes disease prediction after the various algorithms listed are compared.

3.3.4 Software Development Methodology

Agile methodology promotes flexibility and customer satisfaction by delivering small, functional pieces of the software incrementally.

Teams using Agile methodology often hold regular meetings, called sprints, to review progress and plan the next steps. By promoting ongoing feedback and adaptation, the Agile methodology makes sure that the finished product closely satisfies the requirements and expectations of users.

The Agile methodology promotes ongoing input and adjustment, guaranteeing that the finished result closely complies with user requirements and expectations. This methodology fosters close collaboration among team members and stakeholders, ensuring that everyone is aligned and working towards common goals. Agile emphasizes the value of people and relationships over procedures and equipment, placing a strong emphasis on cooperation and communication. The methodology assists groups in recognizing and resolving issues early on, which improves results. Agile practices often include daily stand-up meetings, where team members briefly discuss their progress, plans for the day, and any obstacles they face. This regular communication helps to quickly identify and resolve issues, maintaining momentum and alignment. Retrospectives at the end of each sprint or iteration allow teams to reflect on their performance and identify areas for improvement, fostering a culture of continuous learning and adaptation.

3.3.4.1 Advantages of the Agile Model

Numerous advantages come with using an agile methodology, such as increased flexibility, quicker time to market, more customer collaboration, transparency, adaptability, and continual improvement. Regular feedback and stakeholder interaction guarantee alignment with user needs and expectations, while frequent iterations and adjustments meet shifting requirements and

priorities. Working software can be delivered incrementally, enabling earlier product launches and quicker answers to market requests.

Transparency, trust, and accountability are fostered within the team through regular demonstrations and open communication. It is simpler to manage risks and take advantage of opportunities when agile teams can react quickly to obstacles and possibilities. Several technologies and tools have been used in the system's development, including Python for programming, Django, a Python framework for backend and machine learning deployment, HTML for frontend development, JavaScript for functionality and validation, and SQLite, a well-known open-source database for configuring a local web server environment.

3.3.4.2 System Analysis and Design Methodology

Structured Systems Analysis and Design Method (SSADM) is a system that uses a systematic and disciplined methodology, breaking down systems development into stages that include feasibility study, requirements analysis, design, and implementation. SSADM emphasizes thorough documentation and detailed modeling techniques to ensure that the final system meets user requirements and is delivered on time and within budget. It is particularly useful for large, complex projects where a structured approach can help manage risks and ensure quality.

SSADM utilizes several key tools for system development. Data Flow Diagrams (DFDs) illustrate the flow of data within a system, showing how data is processed through inputs and outputs. Entity-relationship diagrams (ERDs) provide a clear view of the systems on how they are connected to others. In order to maintain uniformity throughout the project, a data dictionary acts as a centralized repository for information on data, including its meaning, relationships, origin, usage, and format.

3.3.4.3. Data collection Techniques

Data collection Techniques: The study will utilize a retrospective analysis of patient data collected from the PIMA Indians Diabetes Dataset. The dataset will include demographic information, clinical characteristics, and diagnostic test results for patients with and without diabetes.

Data Preprocessing: The data will undergo various preprocessing steps, including handling missing values, removing duplicates, and normalizing the features. Feature engineering will be conducted to select the most relevant attributes for the predictive model.

Machine Learning Modeling: The study will primarily focus on the support vector machine algorithm for predicting diabetes. The performance of the support vector machine model will be compared to other machine learning algorithms, such as logistic regression and decision trees, to assess its relative effectiveness.

3.3.4.3.1 Context Diagram

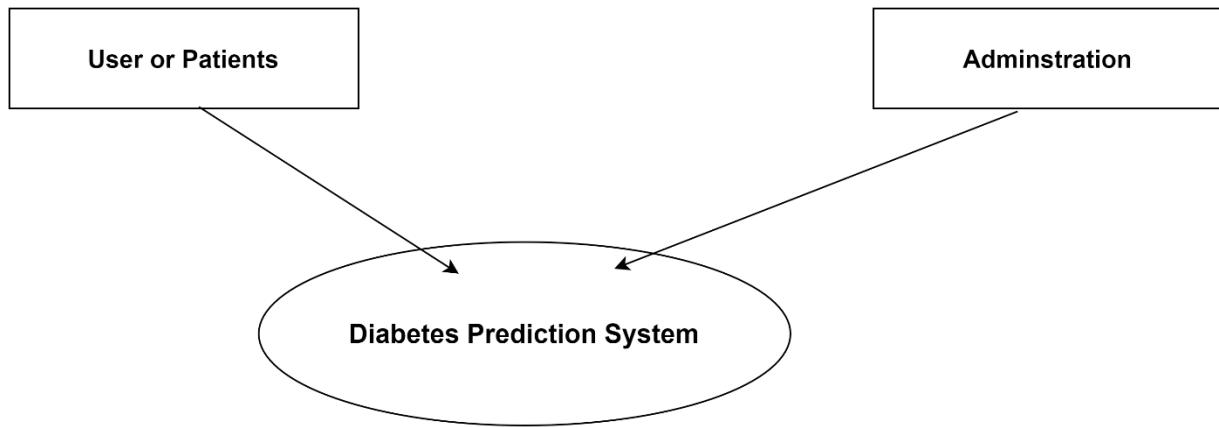


Figure 2: Context Diagram

The system at the Dataset of the diagram is depicted, along with its interactions with two important external entities: users and administrators. Users/Patients can seek diabetes predictions, view prediction history, manage their profiles, register, log in, and execute other interactions with the system. To maintain security and optimal operation of the system, administrators monitor system logs, manage user accounts, and set permissions. This context diagram offers a high-level perspective of the system and its main interactions, which aids in comprehending the border of the system and its principal external relationships.

3.3.4.3.1. Data Flow Diagram (level 0)

DFD Level 0

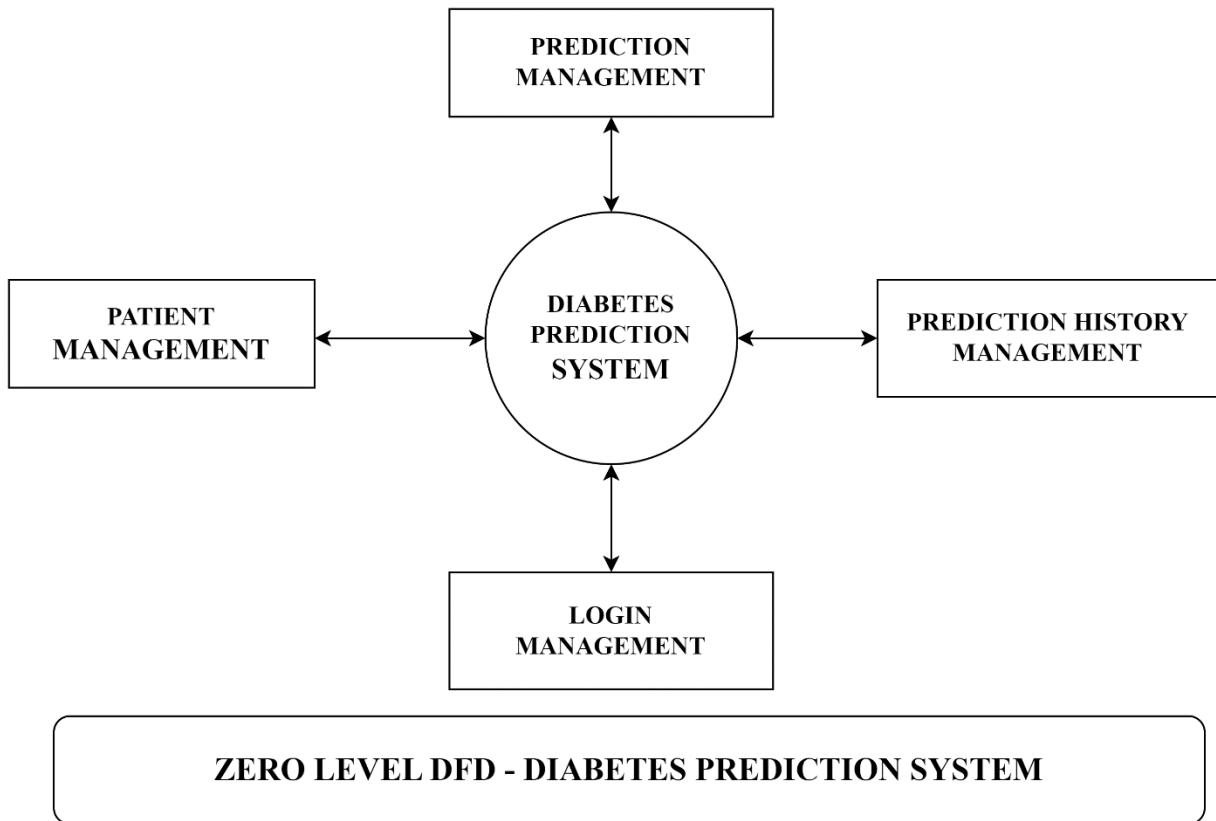


Figure 3: DFD Level 0 and level 1

The figure shows a diabetes prediction system's zero-level data flow diagram (DFD). The Diabetes Prediction System is the central component, interacting with User Management, Prediction History Management, Login Management, and Prediction Management as its four main external entities. These objects stand in for the various system functionalities. Prediction History Management tracks and stores the history of predictions, Patient Management manages user-related operations, Prediction Management generates forecasts, and Login Management is in charge of user authentication. The data flow between these external entities and the core system is depicted in the diagram.

DFD Level 1

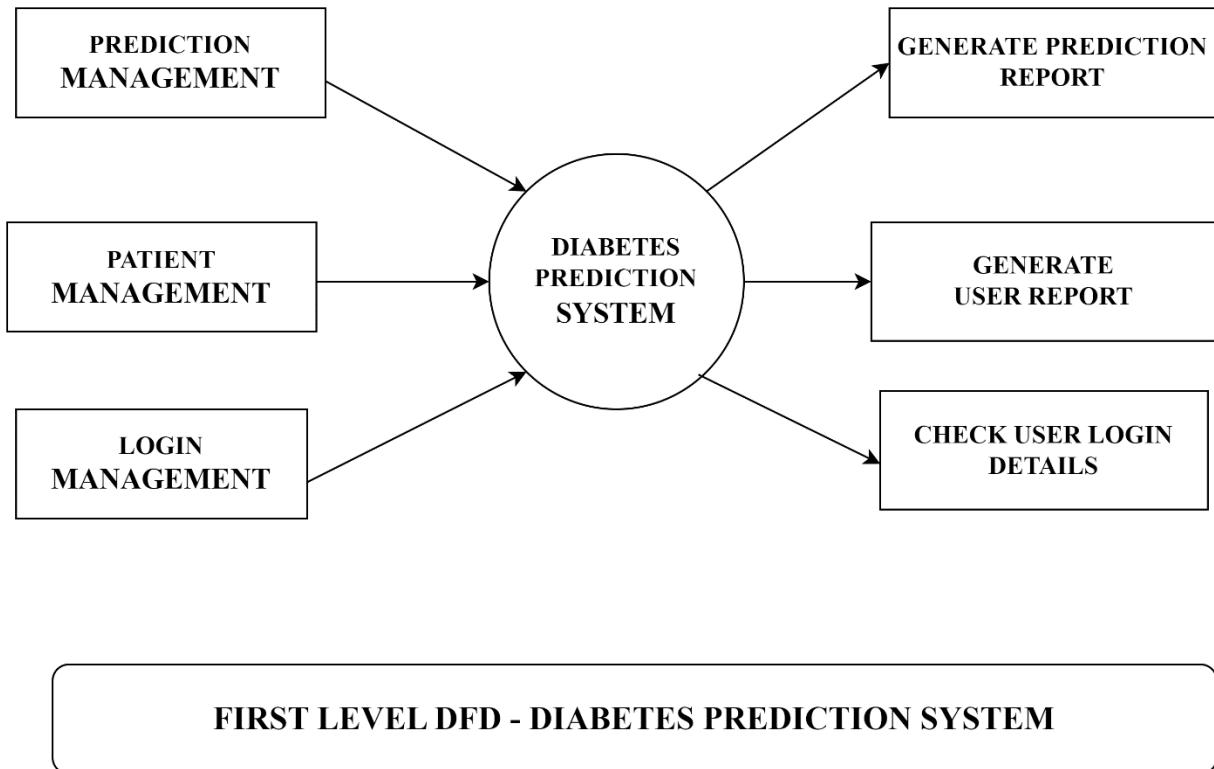


Figure 4: Extension of data flow diagram level 1

In contrast to the Zero Level DFD, the diagram shows a First Level Data Flow Diagram (DFD) for a Diabetes Prediction System, which details more precise interactions and processes. Prediction Management, Patients Management, and Login Management are some of the external entities with which the Diabetes Prediction System, the central system, interacts. Furthermore, it carries out particular tasks including producing forecast reports, creating user reports, and verifying user login credentials. The system's data flow is depicted in the diagram, starting with user and prediction management inputs and ending with the necessary outputs (reports and login credentials). This degree of specificity facilitates comprehension of the many parts of the system and how they interact.

3.3.4.3.2. Entity Relationship Diagram (ERD)

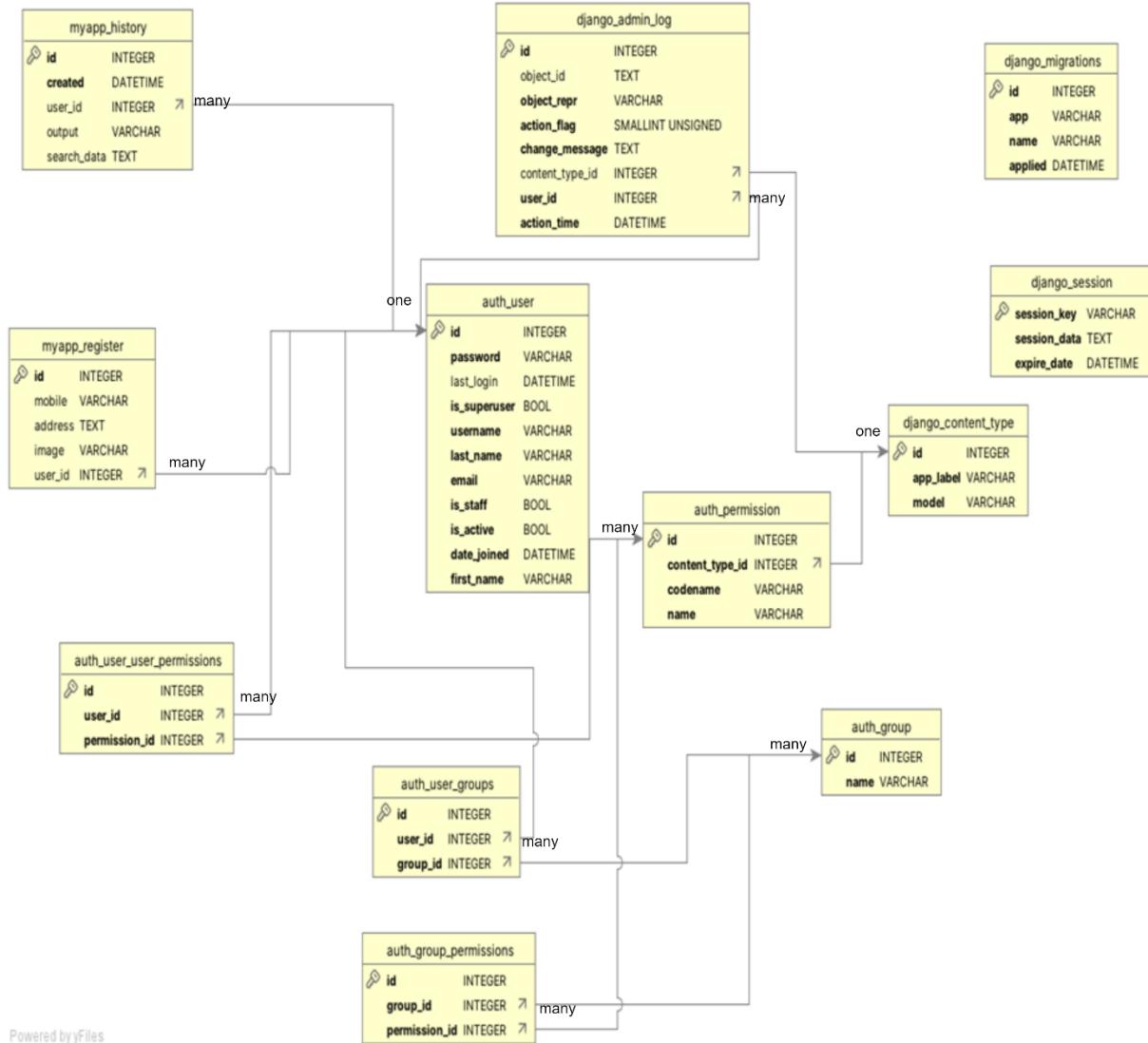


Figure 5: Entity Relationship Diagram

One-to-Many Relationships

auth_user to myapp_register: Each user (auth_user) can be associated with multiple registrations (myapp_register), but each registration is linked to only one user via the user_id field.

auth_user to myapp_history: Each user can have multiple history records, but each history record is linked to only one user.

auth_user to django_admin_log: Each user can be associated with multiple logs, but each log is tied to a single user.

Many-to-Many Relationships

auth_user to auth_permission (via auth_user_user_permissions): Users can have multiple permissions, and each permission can be assigned to multiple users.

auth_group to auth_permission (via auth_group_permissions): Groups can have multiple permissions, and permissions can belong to multiple groups.

auth_user to auth_group (via auth_user_groups): A user can belong to multiple groups, and a group can have multiple users.

One-to-One Relationships

There do not appear to be explicit one-to-one relationships in the diagram. However, Django's **auth_user** table could be seen as having unique entries for each user, making relationships between tables like **auth_user** and **myapp_register** resemble a pseudo one-to-one for unique users with additional related tables.

3.3.4.3.3. Data Dictionary

A data dictionary would act as a central store of knowledge about the data used in the system for a machine learning-based Diabetes Model prediction. The names, definitions, data types, lengths, formats, and relationships between the various data pieces are all provided in detail.

Table 1: Myapp_history

Name	Type	Extra
Id	Integer (10)	Primary key
Created	Datetime	
User_id	Integer(10)	
Output	Varchar(50)	
Search_date	Text (50)	

Table 2. Myapp_register

Name	Type	Extra
Id	Integer(10)	Primary key
Mobile	Varchar(25)	
Address	Text(50)	
Image	Varchar(50)	
User_id	Integer(10)	

Table 3. Auth_permission

Name	Type	Extra
Id	Integer(10)	Primary key
contact_type_id	Integer(10)	Foreign key
Codename	Varchar(50)	
Name	Varchar(50)	

Table 4. Auth_user

Name	Type	Extra
Id	Integer(10)	Primary key
Password	Varchar(50)	
Last_login	DATETIME	
Is_superuser	BOOL	
Username	Varchar(50)	
Last_name	Varchar(50)	
Email	Varchar(50)	
Is_staff	BOOL	
Is_active	BOOL	
Date_joined	DATETIME	
First_name	Varchar(50)	

Table 5. Auth_user_user_permission

Name	Type	Extra
Id	Integer(10)	Primary key
user_id	Integer(10)	Foreign key
Permission_id	Integer(10)	Foreign key

Table 6. Auth_user_groups

Name	Type	Extra
Id	Integer(10)	Primary key
user_id	Integer(10)	Foreign key
group_id	Integer(10)	Foreign key

Table 7. Auth_user_group_permissions

Name	Type	Extra
Id	Integer(10)	Primary key
Permission_id	Integer(10)	Foreign key
group_id	Integer(10)	Foreign key

Table 8. Auth_group

Name	Type	Extra
Id	Integer(10)	Primary key
Name	varchar(50)	

Table 9. Django_admin_log

Name	Type	Extra
Id	Integer(10)	Primary key
object_id	Text	
Object_repr	varchar(50)	
Action_flag	SMALLINT UNSIGNED	
Change_message	Text	
Content_type_id	Integer(10)	Foreign key
User_id	Integer(10)	Foreign key
Action_time	DATETIME	

Table 10. Django_content_type

Name	Type	Extra
Id	Integer(10)	Primary key
app_label	varchar(50)	
Model	varchar(50)	

Table 11. Django_migrations

Name	Type	Extra
Id	Integer(10)	Primary key
App	Varchar(50)	
Name	Varchar(50)	
Applied	Datetime	

Table 12. Django_session

Name	Type	Extra
Session_key	Varchar(50)	Primary key
Session_data	Text	
Session_date	Datetime	

CHAPTER IV: SYSTEM IMPLEMENTATION

4.1. Implementation and Coding

The actual process of creating and programming the support vector machine model using particular tools and libraries is referred to as implementation and coding. In this instance, the scikit-learn Python package was used to create the support vector machine model. The process of evaluating and contrasting the support vector machine model's performance with that of other machine learning techniques is known as evaluation and comparison.

4.1.1. Introduction

To develop the support vector machine model, the scikit-learn Python module was utilized. For training and validation purposes, the dataset was divided into two 65% for train and 35% for testing. Using a grid search technique, the model's hyperparameters the gamma value, regularization parameter, and kernel function were tuned to maximize performance. The performance of the support vector machine model was evaluated using a number of parameters, including test score, F1-score, recall, accuracy, and precision. The findings were compared to see if the support vector machine model performed better than other machine learning techniques, including Random Forests (RF), K-Nearest Neighbors (KNN), Decision Trees (DT), and Logistic Regression (LR).

4.1.2. Description of Implementation Tools and Technology

The study was conducted using Python, HTML, CSS BOOSTRAPT, JAVASCRIPT, DJANGO, SQLite, IDEs (Jupyter Notebook or PyCharm). The following tools will be utilized in the implementation:

4.1.2.1. Python

Python's versatility and ease of use make it a popular choice across a wide range of industries. Reliability in web app development is enabled by frameworks like Flask and Django. In data science and machine learning, libraries such as Pandas, NumPy, and TensorFlow facilitate data processing, visualization, and model building. Python is a popular choice for automation, scripting, and software prototype development because of its easy-to-understand syntax and extensive standard library. Strong community support together with a vibrant ecosystem of third-party packages further assist its widespread acceptance and use across a range of disciplines.

The study will utilize many Python libraries like NumPy, pandas, seaborn, scikit learn, and matplotlib, etc.

Pandas are used for managing tabular data and data preprocessing;

NumPy is used for numeric operations and data manipulation.

Scikit-learn: evolution metrics and machine algorithms

Matplotlib: a tool for visualizing data.

Seaborn: for visual aids with statistics.

4.1.2.2: HTML

The common markup language used to construct web pages is called HTML (Hypertext Markup Language). Using tags, attributes, and elements, developers can specify the organization of material on a webpage thanks to this organized language. Web browsers can comprehend and display text in a structured and ordered fashion thanks to HTML.

4.1.2.3. CSS

Cascading Style Sheet (CSS) is used for styling HTML documents either inner or separated, and it controls the designing and making websites beautiful. It makes it possible for developers to divide a web page's display from its structure. With CSS, web developers can specify a webpage's visual elements, including layout, color scheme, and font style.

4.1.2.4. JavaScript

High-level, interpreted programming languages like JavaScript are used to make interactive websites. It is a client-side scripting language that lets programmers give web pages dynamic features and functionality. JavaScript has the ability to alter HTML and CSS in real time, improving the responsiveness and interactivity of websites.

4.1.2.5. Bootstrap

Bootstrap is a well-liked free and open-source CSS framework for creating mobile-first and responsive websites. It offers pre-made CSS styles and JavaScript plugins that developers may utilize to rapidly and simply construct web pages with a polished appearance. Bootstrap works with all contemporary web browsers and mobile devices.

4.1.2.6. Django

Python-based Django is a free and open-source web framework. Model-view-controller (MVC) architecture is used in its design, which makes it easy and rapid for developers to create web applications. Web application development is made easier by Django's many built-in capabilities, which include an object-relational mapper (ORM), an automated admin interface, and URL routing.

4.1.2.7. SQLite

A serverless, lightweight, and self-contained relational database management system is called SQLite. Due to its low administration and configuration requirements, it is frequently utilized in online and mobile applications. All of the main programming languages are compatible with SQLite, which offers a quick and easy method for storing and retrieving data.

4.1.2.8. Support Vector Machine (SVM)

The machine learning technique employed by the Diabetes Prediction System is called Support Vector Machine (SVM). Using data analysis and pattern recognition, SVM is a supervised learning algorithm that generates predictions. It works especially well for classification tasks, like this one where it has to determine whether diabetes is present or not.

SVM divides the data into several classes according to their features by constructing a hyperplane. It seeks to identify the ideal hyperplane that optimizes the margin between classes, enabling more accurate predictions and improved generalization. By transforming the data into higher-dimensional spaces using various kernel functions, SVM can handle both linear and non-linear data.

SVM is trained on a dataset containing characteristics like age, diabetes pedigree function, body mass index, skin thickness, insulin level, and number of pregnancies in the Diabetes Prediction System. A predictive model that can identify new occurrences as either diabetes or non-diabetic is constructed using these features. The system may learn intricate correlations between the input features and the target variable (diabetes status) by utilizing Support Vector Machines (SVM). It can adjust to various data distributions and handle datasets with a large number of attributes. Support Vector Machine is renowned for its capacity to manage datasets of any size and produce reliable predictions.

Overall, the use of SVM in the Diabetes Prediction System enables accurate classification of individuals based on their health parameters, facilitating early detection and proactive management of diabetes.

4.1.2.9. IDEs (Jupyter Notebook, and PyCharm)

Examples of Integrated Development Environments (IDEs) that are helpful tools for data analysis and programming are Jupyter Notebooks and PyCharm. PyCharm's extensive collection of Python development tools, which includes code completion, debugging, and integrated version control, is quite helpful for large-scale projects. On the other hand, Jupyter Notebooks provide an interactive online environment that is ideal for data exploration, machine learning, and scientific computing. In cells, users can write autonomous code and get instantaneous visual feedback for charts and analysis. Both technologies are essential for various use cases, offering a balance between productivity and flexibility depending on the kind and scope of the project.

4.2. Machine Learning Analysis from Dataset using Jupyter Notebook

Machine learning (ML) has become a transformative tool in the healthcare sector, particularly in areas like disease diagnosis and prediction. One prominent application is in **diabetes prediction**. With the growing prevalence of diabetes worldwide, early detection and personalized treatment plans are crucial. ML models, such as Support Vector Machines (SVM), Decision Trees, and Neural Networks, can be trained on patient data, including lifestyle factors, genetic information, and medical history, to predict the onset of diabetes. These models can analyze complex patterns that may not be obvious to human practitioners, allowing for earlier intervention and potentially reducing the long-term effects of the disease. The ability to process vast amounts of data quickly also makes ML valuable in identifying high-risk individuals and tailoring treatment plans to their unique needs.

4.2.1. Diabetes Dataset Analysis

The datasets (768, 9) consist of several medical predictor (independent) variables and one target (dependent) variable, **Outcome**. Independent variables include the number of pregnancies the patient has had, their BMI, insulin level, age, and so on. The table below displays the features of the datasets and the hers is the link (<https://www.kaggle.com/datasets/uciml/pima-indians-diabetes-database>) [23].

Table 13: Diabetic Dataset

Features name	Categories
Pregnancies	Numeric
Glucose	Numeric
Body mass index (BMI)	Numeric
Blood pressure	Numeric
Skin thickness	Numeric
Insulin	Numeric
Age	Numeric
Diabetes pedigree's function	Numeric
Outcome	Non-diabetic, Diabetic (0, 1)

- Pregnancies: Number of times pregnant
- Glucose: is the method used to measure the level of sugar in the blood.
- Blood Pressure (BP): The pressure of circulating blood against the walls of blood vessels (Hg).
- Skin Thickness: Triceps skin fold thickness (mm)
- Insulin: 2-Hour serum insulin (mu U/ml)
- BMI: is the weight in kilograms (kg) divided by height in meters (m) squared.
- Diabetes Pedigree Function: Diabetes pedigree function
- Age: Age (years)
- Outcome: Target variable (0 or 1)

4.2.2. Data Analysis (Cleaning, Preprocessing, EDA, Training, Testing and Model Evaluation)

Import Libraries

```
import numpy as np
import pandas as pd
import seaborn as sns
import statsmodels.api as sm
import matplotlib.pyplot as plt
from sklearn.preprocessing import scale, StandardScaler
from sklearn.model_selection import train_test_split, GridSearchCV, cross_val_score
from sklearn.metrics import confusion_matrix, accuracy_score, mean_squared_error, r2_score, roc_auc_score, roc_curve, classification_report
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.neural_network import MLPClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import KFold
import warnings
warnings.simplefilter(action='ignore')
sns.set()
plt.style.use("ggplot")
%matplotlib inline
```

Figure 6. Libraries for Data Analysis

```
# Load the dataset
df = pd.read_csv("diabetes.csv")
```

Figure 7. Load Dataset

Table 14. Dataset Head

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome	
0	6	148	72	35	0	33.6		0.627	50	1
1	1	85	66	29	0	26.6		0.351	31	0
2	8	183	64	0	0	23.3		0.672	32	1
3	1	89	66	23	94	28.1		0.167	21	0
4	0	137	40	35	168	43.1		2.288	33	1

Exploratory Data Analysis (EDA) is the process of analyzing and summarizing datasets to understand their main characteristics using visual and statistical techniques. It helps uncover patterns, detect anomalies, test hypotheses, and check assumptions before applying more complex machine learning models.

Table 15. Dataset Information

```
# Information about the dataset
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column           Non-Null Count  Dtype  
 ---  --  
 0   Pregnancies      768 non-null    int64  
 1   Glucose          768 non-null    int64  
 2   BloodPressure    768 non-null    int64  
 3   SkinThickness    768 non-null    int64  
 4   Insulin          768 non-null    int64  
 5   BMI              768 non-null    float64 
 6   DiabetesPedigreeFunction 768 non-null  float64 
 7   Age              768 non-null    int64  
 8   Outcome          768 non-null    int64  
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
```

The descriptive statistics table provides an overview of the dataset's numerical variables, including measures such as count, mean, standard deviation, minimum, maximum, and quartiles (25%, 50%, 75%). It shows key insights like the average glucose level (120.89), BMI (31.99), and the fact that most variables contain values of zero, which may represent missing or unrecorded data.

Table 16. Descriptive Statistics of the Dataset

# descriptive statistics of the dataset df.describe()									
	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
count	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000
mean	3.845052	120.894531	69.105469	20.536458	79.799479	31.992578	0.471876	33.240885	0.348958
std	3.369578	31.972618	19.355807	15.952218	115.244002	7.884160	0.331329	11.760232	0.476951
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.078000	21.000000	0.000000
25%	1.000000	99.000000	62.000000	0.000000	0.000000	27.300000	0.243750	24.000000	0.000000
50%	3.000000	117.000000	72.000000	23.000000	30.500000	32.000000	0.372500	29.000000	0.000000
75%	6.000000	140.250000	80.000000	32.000000	127.250000	36.600000	0.626250	41.000000	1.000000
max	17.000000	199.000000	122.000000	99.000000	846.000000	67.100000	2.420000	81.000000	1.000000

4.2.3. Data Visualization

The pie chart represents the distribution of diabetes outcomes, where **65.1%** of the patients have a negative outcome (class 0, in blue), meaning they are non-diabetic, and **34.9%** have a positive outcome (class 1, in red), indicating they are diabetic. This class imbalance suggests that a larger portion of the population is non-diabetic, which may need to be considered when training machine learning models to avoid bias toward the majority class.

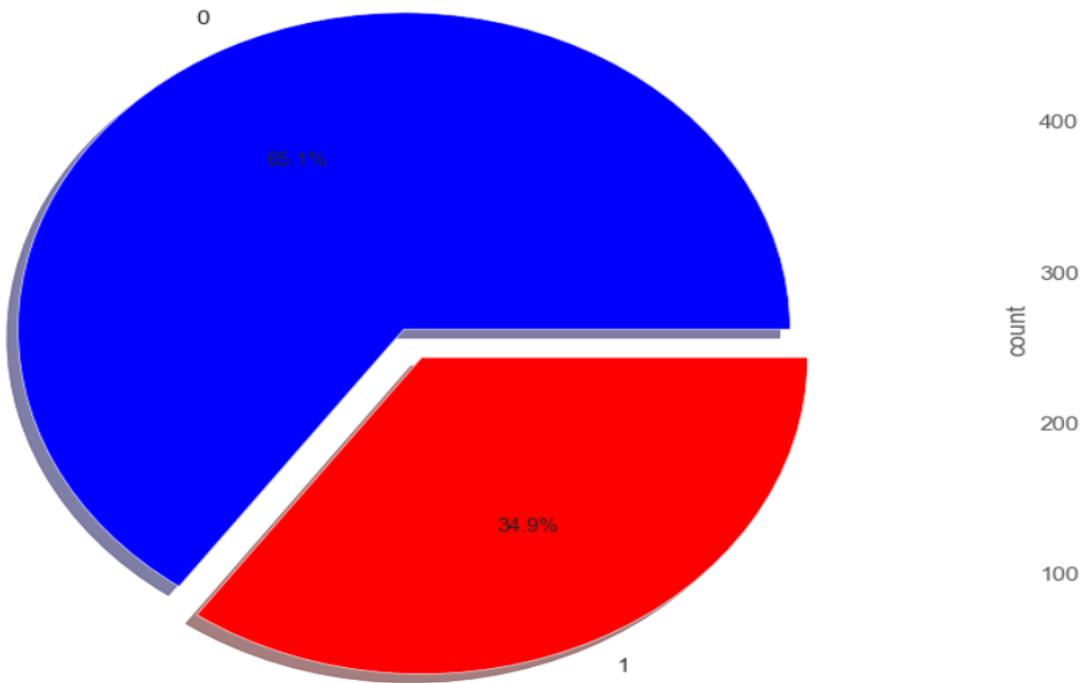


Figure 8. Dataset Outcome

The correlation matrix shows the relationships between different features of a diabetes dataset. The strongest correlation with the Outcome (diabetes presence) is seen with Glucose (0.47), indicating that higher glucose levels are associated with a higher likelihood of diabetes, while other variables like BMI (0.29) and Age (0.24) also show moderate positive correlations.

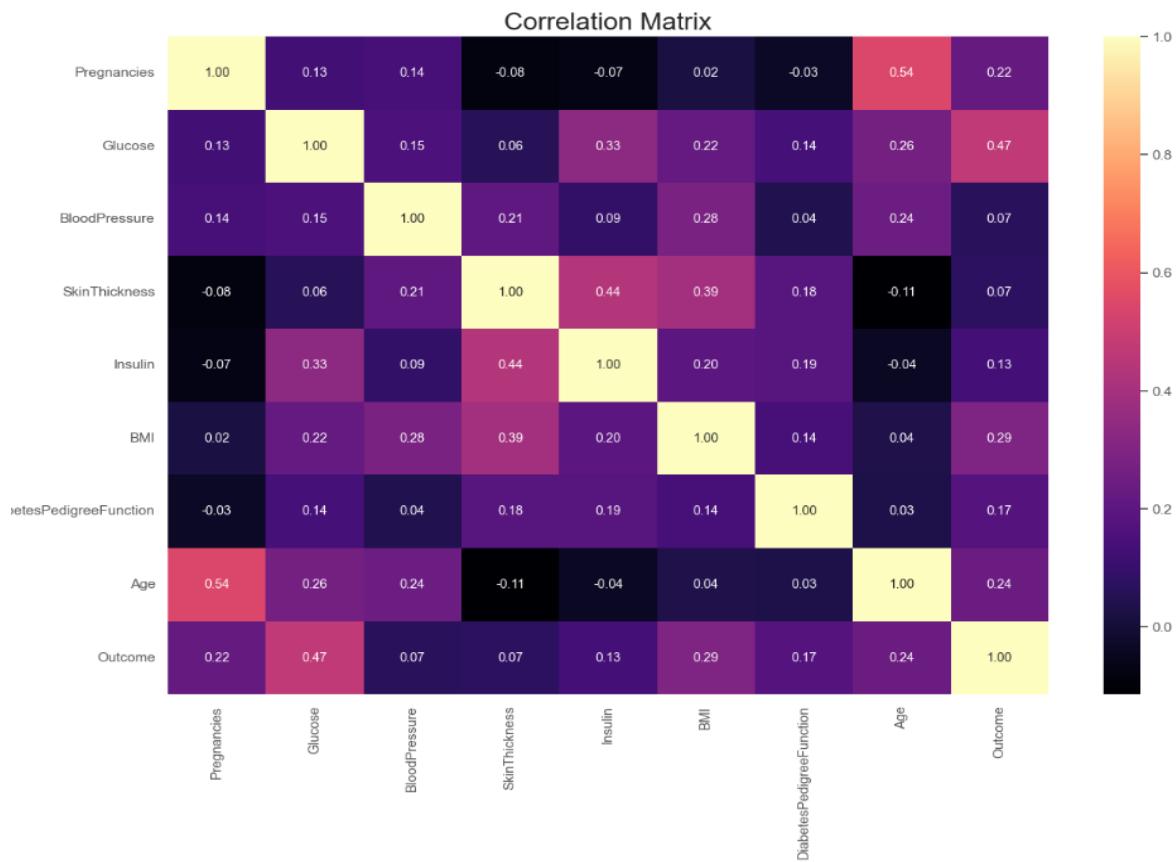


Figure 9. Correlation Matrix

The missing values graph indicates that the dataset has missing values for most features.

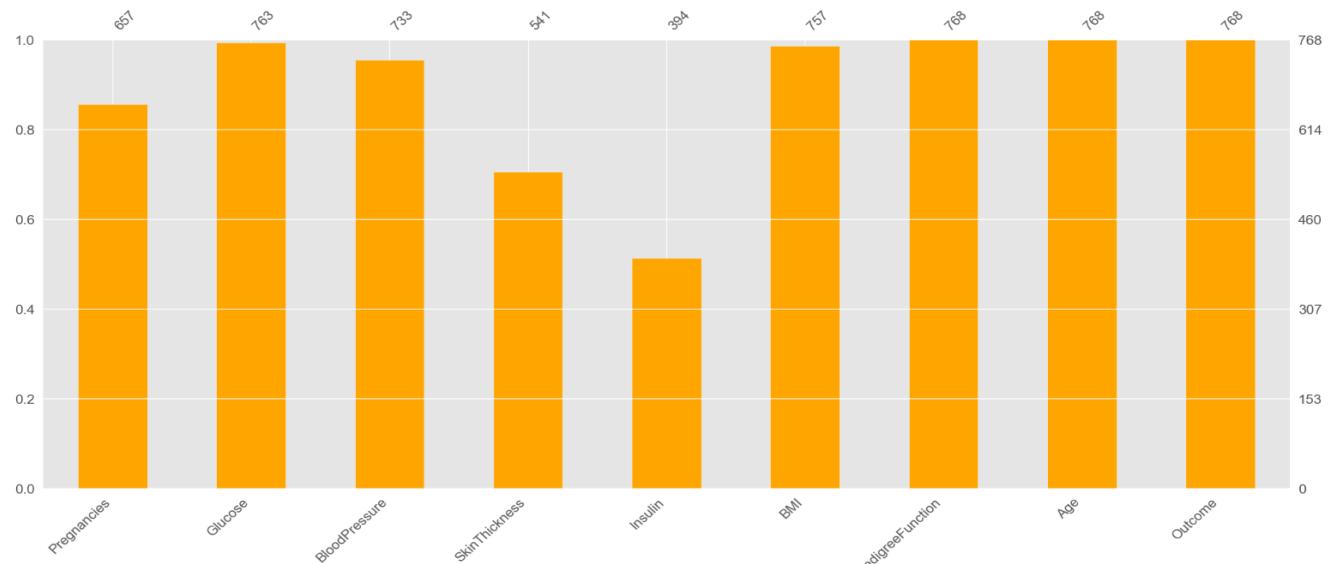


Figure 10. Dataset Missing Values

Filling Missing Values with Statistical method

```
#median
def median_target(var):
    temp = df[df[var].notnull()]
    temp = temp[[var, 'Outcome']].groupby(['Outcome'])[[var]].median().reset_index()
    return temp

columns = df.columns
columns = columns.drop("Outcome")
for i in columns:
    median_target(i)
    df.loc[(df['Outcome'] == 0) & (df[i].isnull()), i] = median_target(i)[i][0]
    df.loc[(df['Outcome'] == 1) & (df[i].isnull()), i] = median_target(i)[i][1]
```

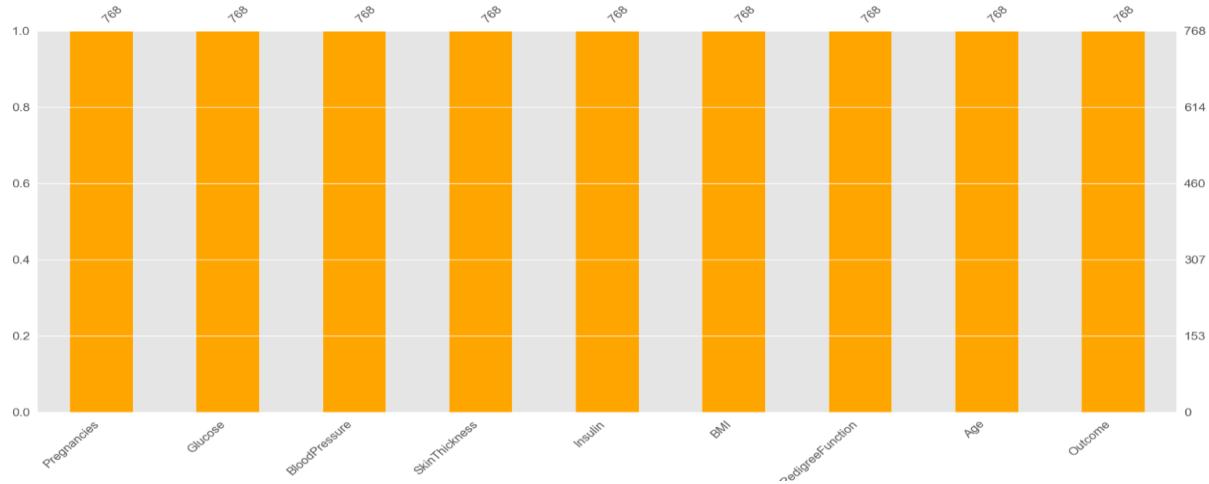


Figure 11. Filling Missing Values

4.2.4. Outlier Detection on the datasets

```
: #Outlier detection is an important step in data preprocessing and analysis.
#Outliers can significantly affect the performance of machine Learning models, so detecting and addressing them is crucial.
# Outlier Detection
# IQR+Q1
# 50%
# 24.65->25%+50%
# 24.65->25%
for feature in df:
    Q1 = df[feature].quantile(0.25)
    Q3 = df[feature].quantile(0.75)
    IQR = Q3-Q1
    lower = Q1-1.5*IQR
    upper = Q3+1.5*IQR
    if df[(df[feature]>upper)].any(axis=None):
        print(feature, "yes")
    else:
        print(feature, "no")

Pregnancies yes
Glucose no
BloodPressure yes
SkinThickness yes
Insulin yes
BMI yes
DiabetesPedigreeFunction yes
Age yes
Outcome no
```

Figure 12. Outlier Detection

4.2.5. Feature Engineering

```
|: # Feature Engineering is the process of transforming raw data into meaningful features that can improve the performance of machine Learning models.
|: NewBMI = pd.Series(["Underweight", "Normal", "Overweight", "Obesity 1", "Obesity 2", "Obesity 3"], dtype = "category")
|: 
|: NewBMI
|: 0    Underweight
|: 1      Normal
|: 2   Overweight
|: 3   Obesity 1
|: 4   Obesity 2
|: 5   Obesity 3
|: dtype: category
|: Categories (6, object): ['Normal', 'Obesity 1', 'Obesity 2', 'Obesity 3', 'Overweight', 'Underweight']
|: 
|: df['NewBMI'] = NewBMI
|: df.loc[df['BMI']>18.5, "NewBMI"] = NewBMI[0]
|: df.loc[(df['BMI']>18.5) & df['BMI']<=24.9, "NewBMI"] = NewBMI[1]
|: df.loc[(df['BMI']>24.9) & df['BMI']<=29.9, "NewBMI"] = NewBMI[2]
|: df.loc[(df['BMI']>29.9) & df['BMI']<=34.9, "NewBMI"] = NewBMI[3]
|: df.loc[(df['BMI']>34.9) & df['BMI']<=39.9, "NewBMI"] = NewBMI[4]
|: df.loc[df['BMI']>39.9, "NewBMI"] = NewBMI[5]
```

Figure 13. Feature Engineering

The diagram demonstrates feature engineering where the **BMI** values are transformed into meaningful categories, such as "Underweight," "Normal," "Overweight," and various obesity levels, using a categorical series. This technique helps convert continuous data into discrete classes, which can improve the performance and interpretability of machine learning models.

4.2.6. Feature Scaling

```
from sklearn.preprocessing import StandardScaler

# Initialize StandardScaler
transformer = StandardScaler().fit(X)

# Transform the data using the fitted StandardScaler
X = transformer.transform(X)

# Convert the scaled data back to a DataFrame
X = pd.DataFrame(X, columns=cols, index=index)
```

Figure 14. Feature Scaling

Feature scaling is the process of normalizing or standardizing the range of independent variables or features in a dataset so that they fall within a similar range. This ensures that models sensitive to the magnitude of features, like SVM LR, RF, DT or KNN, perform optimally without one feature dominating due to larger numeric values

4.2.7. Training and Testing Dataset

```
X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.2, random_state=0)

scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

Figure 15. Dataset Training and Testing

The code splits the dataset into training and testing sets, with 20% of the data allocated for testing and a random state of 0 to ensure reproducibility. It then applies StandardScaler to standardize the features in both the training and testing sets, ensuring they have a mean of 0 and a standard deviation of 1 for improved model performance.

4.2.8. Model Evaluation for Support Vector Machine

To solve this, we use the mathematical formular $w^T x + b = 0$

W: is the Weight vector

X: is the feature vector

B: is the bias term

The diagram below illustrates how Support Vector Machine (SVM) finds the optimal hyperplane that maximizes the margin between two classes (green and blue points). The support vectors are the data points closest to the hyperplane, which influence its position and orientation. SVM creates two parallel hyperplanes that separate the classes, maximizing the margin between these boundaries to achieve the best classification.

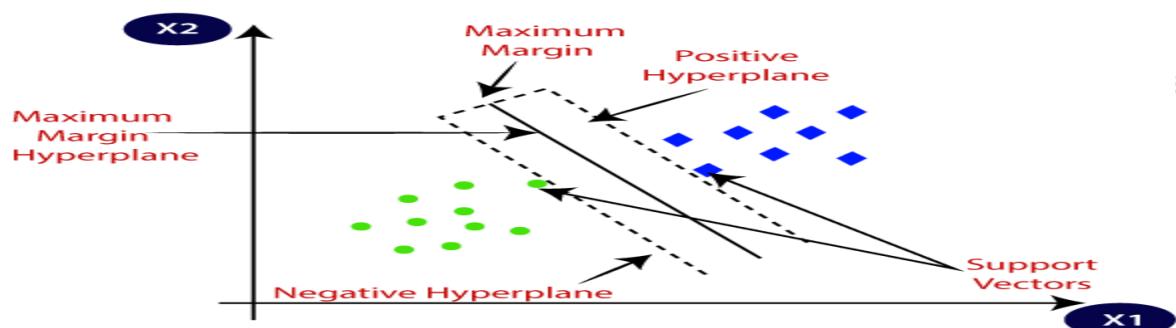


Figure 16. SVM Graph

```

svc = SVC(C=10, gamma = 0.01, probability=True)
svc.fit(X_train, y_train)
y_pred = svc.predict(X_test)
print(accuracy_score(y_train, svc.predict(X_train)))
svc_acc = accuracy_score(y_test, svc.predict(X_test))
print(accuracy_score(y_test, svc.predict(X_test)))
print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred))

0.875
0.9078947368421053
[[90  8]
 [ 6 48]]
      precision    recall  f1-score   support
          0       0.94      0.92      0.93      98
          1       0.86      0.89      0.87      54

   accuracy                           0.91      152
  macro avg       0.90      0.90      0.90      152
weighted avg       0.91      0.91      0.91      152

```

Figure 17. SVM Outputs

4.2.9. Logistics Regression model

```

#Logistic Regression
log_reg = LogisticRegression()
log_reg.fit(X_train, y_train)

▼ LogisticRegression
  LogisticRegression()

y_pred = log_reg.predict(X_test)

accuracy_score(y_train, log_reg.predict(X_train))

0.8470394736842105

log_reg_acc = accuracy_score(y_test, log_reg.predict(X_test))

confusion_matrix(y_test, y_pred)

array([[88, 10],
       [ 6, 48]])

print(classification_report(y_test, y_pred))

      precision    recall  f1-score   support
          0       0.94      0.90      0.92      98
          1       0.83      0.89      0.86      54

   accuracy                           0.89      152
  macro avg       0.88      0.89      0.89      152
weighted avg       0.90      0.89      0.90      152

```

Figure 18. Logistic Regression Outputs

4.2.10. K-Nearest Neighbor Model

```
[]: # K-Nearest Neighbors
knn = KNeighborsClassifier()
knn.fit(X_train, y_train)
y_pred = knn.predict(X_test)
print(accuracy_score(y_train, knn.predict(X_train)))
knn_acc = accuracy_score(y_test, knn.predict(X_test))
print(accuracy_score(y_test, knn.predict(X_test)))
print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred))

0.875
0.881578947368421
[[88 10]
 [ 8 46]]
      precision    recall   f1-score   support
          0       0.92      0.90      0.91       98
          1       0.82      0.85      0.84       54

   accuracy                           0.88       152
  macro avg       0.87      0.87      0.87       152
weighted avg       0.88      0.88      0.88       152
```

Figure 19. K-Nearest Neighbor Outputs

4.2.11. Decision Tree Model

```
[]: DT = grid_search_dt.best_estimator_
y_pred = DT.predict(X_test)
print(accuracy_score(y_train, DT.predict(X_train)))
dt_acc = accuracy_score(y_test, DT.predict(X_test))
print(accuracy_score(y_test, DT.predict(X_test)))
print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred))

0.8980263157894737
0.875
[[92  6]
 [13 41]]
      precision    recall   f1-score   support
          0       0.88      0.94      0.91       98
          1       0.87      0.76      0.81       54

   accuracy                           0.88       152
  macro avg       0.87      0.85      0.86       152
weighted avg       0.87      0.88      0.87       152
```

Figure 20. Decision Tree Outputs

4.2.12. Random Forest Model

```

y_pred = rand_clf.predict(X_test)
print(accuracy_score(y_train, rand_clf.predict(X_train)))
rand_acc = accuracy_score(y_test, rand_clf.predict(X_test))
print(accuracy_score(y_test, rand_clf.predict(X_test)))
print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred))

0.993421052631579
0.8947368421052632
[[89  9]
 [ 7 47]]
      precision    recall  f1-score   support
          0       0.93      0.91      0.92       98
          1       0.84      0.87      0.85       54

   accuracy                           0.89      152
  macro avg       0.88      0.89      0.89      152
weighted avg       0.90      0.89      0.90      152

```

Figure 21. Random Forest Outputs

4.2.13. Machine Learning Evaluation Metrics Performance for Classification Models.

Accuracy Score: Accuracy is the ratio of correctly predicted observations to the total observations. Accuracy gives the overall correctness of the model. However, it can be misleading for imbalanced datasets where one class dominates.

$$\text{Formula: } \text{Accuracy Score} = \frac{TN+TP}{TP + TN + FN + FP}$$

Precision: Precision is the ratio of correctly predicted positive observations to the total predicted positives. **Formula:** $Precision = \frac{TP}{TP + FP}$

Recall: Recall is the ratio of correctly predicted positive observations to all observations in the actual class. Recall shows how well the model identifies positive cases, useful when missing positive cases is more detrimental than false positives. **Formula:** $Recall = \frac{TP}{TP + FN}$

F1-Score: The F1-score is the harmonic mean of Precision and Recall, balancing both metrics. The F1-score is particularly useful when the dataset is imbalanced, as it takes both false positives

and false negatives into account, providing a more nuanced view than accuracy alone.

$$\text{Formula: } F1 - Score = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

True Positive (TP): The model correctly predicted the positive class.

Example: The model predicted "diabetic" and the patient is actually diabetic.

True Negative (TN): The model correctly predicted the negative class.

Example: The model predicted "non-diabetic" and the patient is actually non-diabetic.

False Positive (FP): The model incorrectly predicted the positive class (also known as a Type I error). Example: The model predicted "diabetic," but the patient is not diabetic.

False Negative (FN): The model incorrectly predicted the negative class (also known as a Type II error). Example: The model predicted "non-diabetic," but the patient is actually diabetic.

4.2.14. Model Comparison

```
# Model Comparison
models = pd.DataFrame({
    'Model': ['Logistic Regression', 'KNN', 'SVM', 'Decision Tree Classifier', 'Random Forest Classifier'],
    'Score': [100*round(log_reg_acc,4), 100*round(knn_acc,4), 100*round(svc_acc,4), 100*round(dt_acc,4), 100*round(rand_acc,4)]
})
models.sort_values(by = 'Score', ascending = False)
```

	Model	Score
2	SVM	90.79
0	Logistic Regression	89.47
4	Random Forest Classifier	89.47
1	KNN	88.16
3	Decision Tree Classifier	87.50

Figure 22. Model Comparison

This diagram shows a comparison of different machine learning models based on their accuracy scores. The models compared include Logistic Regression, K-Nearest Neighbors (KNN), Support Vector Machine (SVM), Decision Tree, and Random Forest. The SVM model performs the best with an accuracy score of 90.79%, followed by Logistic Regression and Random Forest, both achieving scores around 89.47%. KNN and Decision Tree have slightly lower scores of 88.16% and 87.50%, respectively, indicating that SVM outperforms the other models in this specific evaluation.

4.2.15. Saving Mode Using Support Vector Classifier

```
import pickle
model = svc
pickle.dump(model, open("diabetes.pkl", 'wb'))
```

Figure 23. Saving Model

The image shows the process of saving a trained machine learning model (in this case, an SVM model) using the Python pickle module. The model (SVC) is serialized into a binary file named diabetes.pkl, which can later be loaded to make predictions without retraining. This step is crucial for web deployment as it allows the model to be reused efficiently in a web application.

4.2.16. Receiver Operating Characteristic (ROC)

The Receiver Operating Characteristic (ROC) curve is a graphical representation of a classification model's performance. It plots the True Positive Rate (Sensitivity) against the False Positive Rate (1-Specificity) at various threshold settings. The curve helps visualize the trade-off between sensitivity (detecting positives) and specificity (avoiding false positives).

The ROC curve compares the performance of five models (Logistic Regression, Decision Tree, SVM, KNN, and Random Forest) in predicting diabetes by plotting sensitivity (True Positive Rate) against 1-specificity (False Positive Rate). The SVM and Random Forest models show the best performance with an AUC (Area Under the Curve) of 0.90, while the Decision Tree has the lowest AUC of 0.85, indicating relatively weaker classification ability.

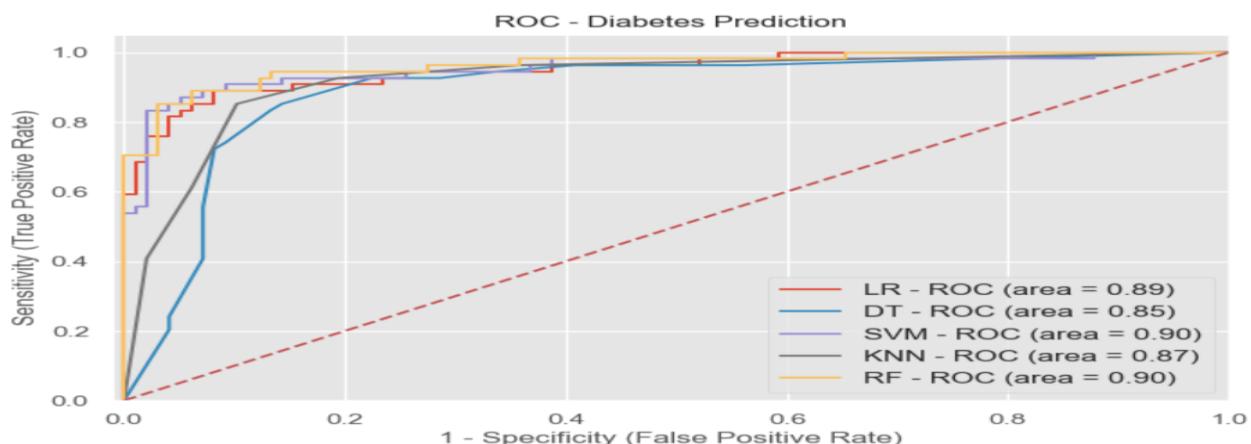


Figure 24. Receiver Operating Characteristic (ROC)

4.2.17. Area Under Curve (AUC)

The Area Under the Curve (AUC) refers to the area under the ROC curve, and it provides a single scalar value to summarize the performance of a classification model. It measures the ability of the model to distinguish between positive and negative classes.

The Area Under the Curve (AUC) is a performance metric used to evaluate the ability of a classification model to distinguish between classes. It represents the area under the Receiver Operating Characteristic (ROC) curve, where a higher AUC value indicates better performance in distinguishing between positive and negative classes, with a perfect model achieving an AUC of 1.0.

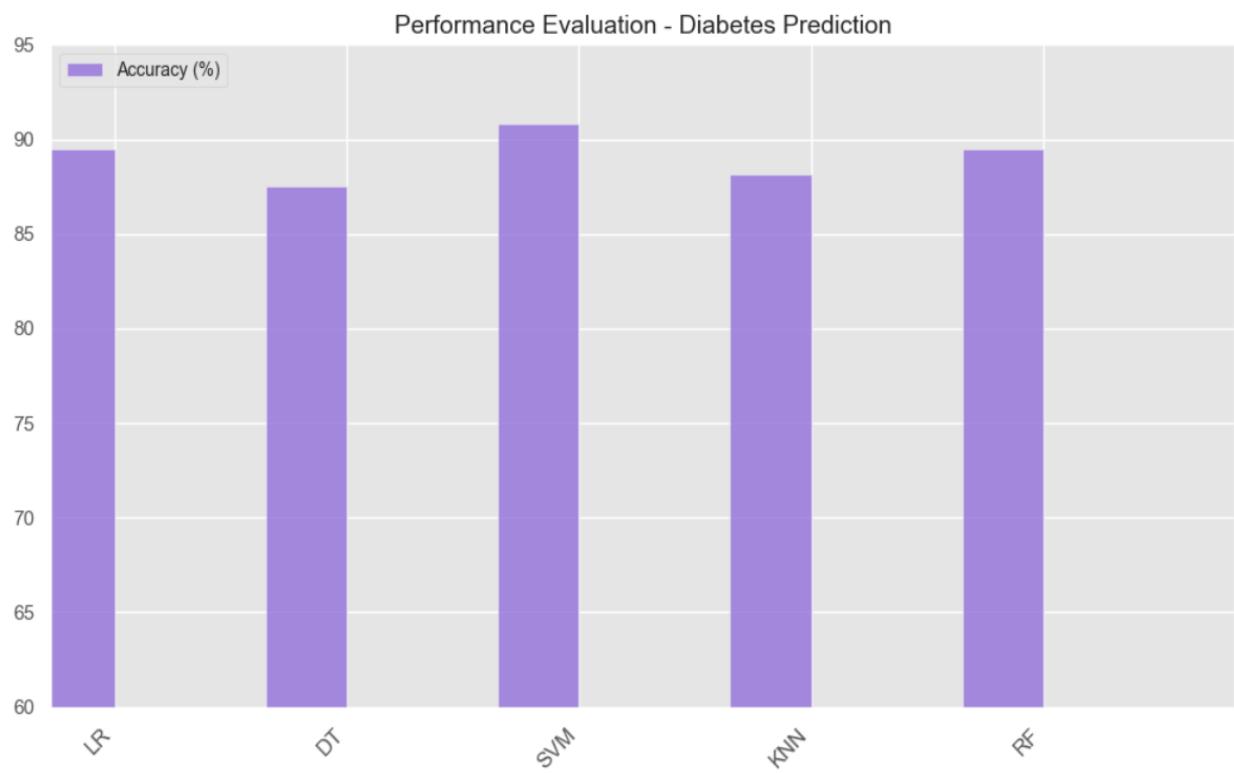


Figure 25. Area Under Curve (AUC)

4.3.3. Screen shorts and source codes

4.3.3.1. Home Menu

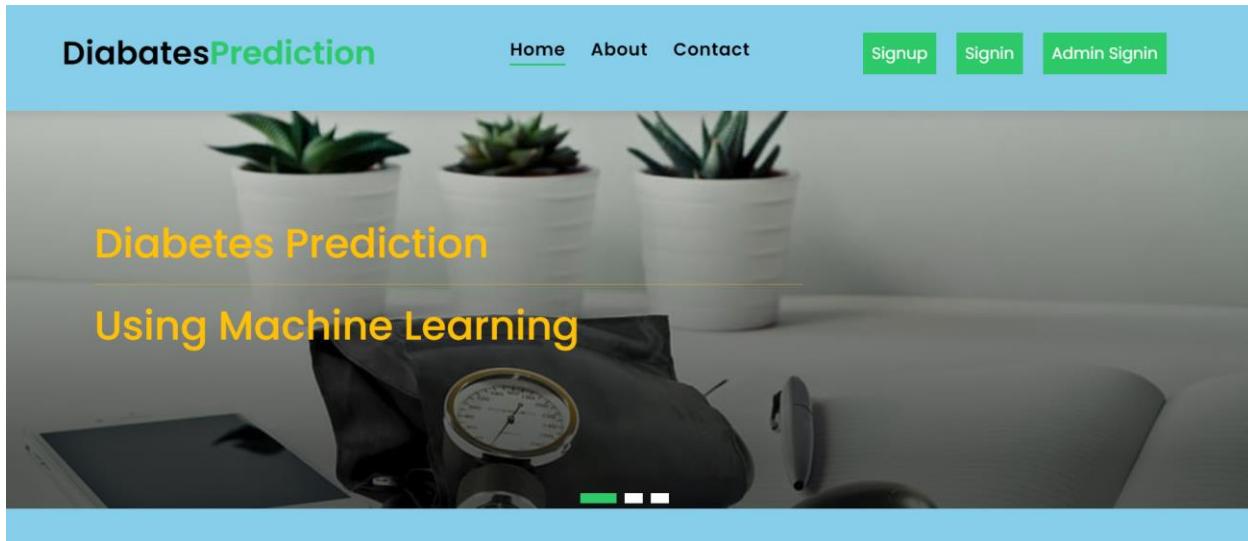


Figure 26: Home Screen

The homepage of the "Diabetes Prediction" shows It features a navigation menu with links to "Home," "About," "Contact," and buttons for "Signup," "Signin," and "Admin Signin."

4.3.3.2. Contact Us Screen Shot

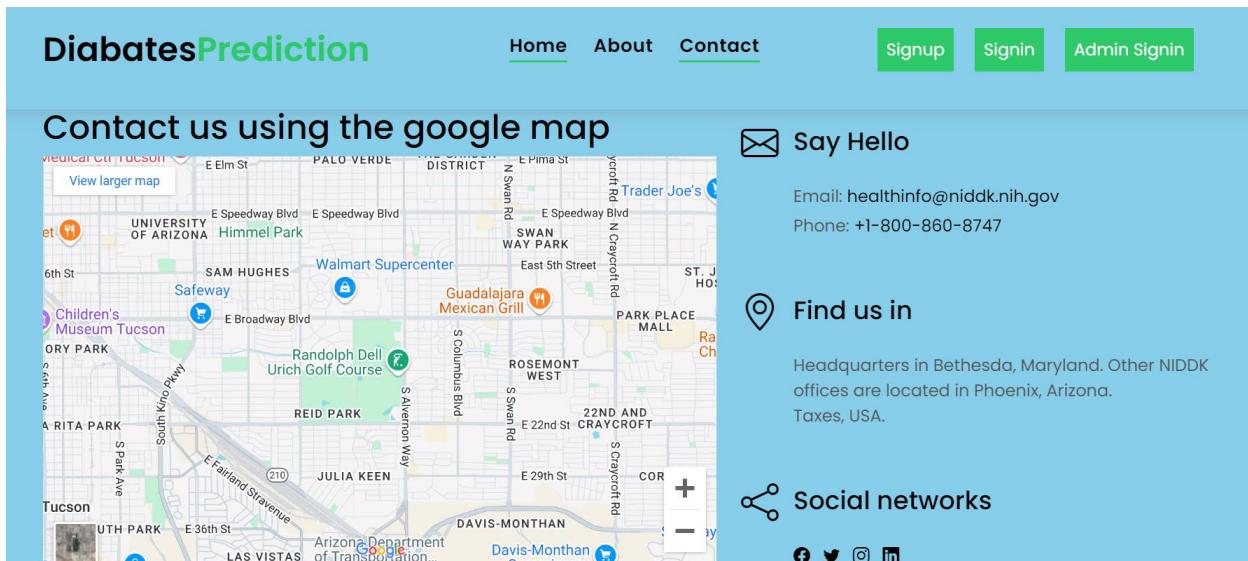


Figure 27: Contact us Screen

The "Contact" page of the "Diabetes Prediction" show the map of Monrovia, along with contact information, location details in Monrovia, , and links to social networks.

4.3.3.3. Signup/ Create user screenshot

DiabatesPrediction

Home About Contact [Signup](#) [Signin](#) [Admin Signin](#)

First name Last name

Email Password

Image No file chosen Mobile

Address

[Register](#)

Figure 28: Signup Screen

The registration page of the "Diabetes Prediction" website, where users can enter their first name, last name, email, password, mobile number, and address, and optionally upload an image before clicking the "Register" button to create an account.

4.3.3.4. User Sign-in Screenshot

DiabatesPrediction

Home About Contact [Signup](#) [Signin](#) [Admin Signin](#)

User Sign In

Email

Password

[Sign In](#) [Forgot Password](#)

Figure 29: User Sign-in

The figure above shows, where a user can sign-in form with fields for email and password, and click on the sign-in buttons with the correct information.

4.3.3.5. Admin Sign-in

Figure 30: Admin Sign-in

The figure above shows, where a super user/admin can sign-in form with fields for email and password, and click on the sign-in buttons with the correct information.

4.3.3.5. Admin Screen

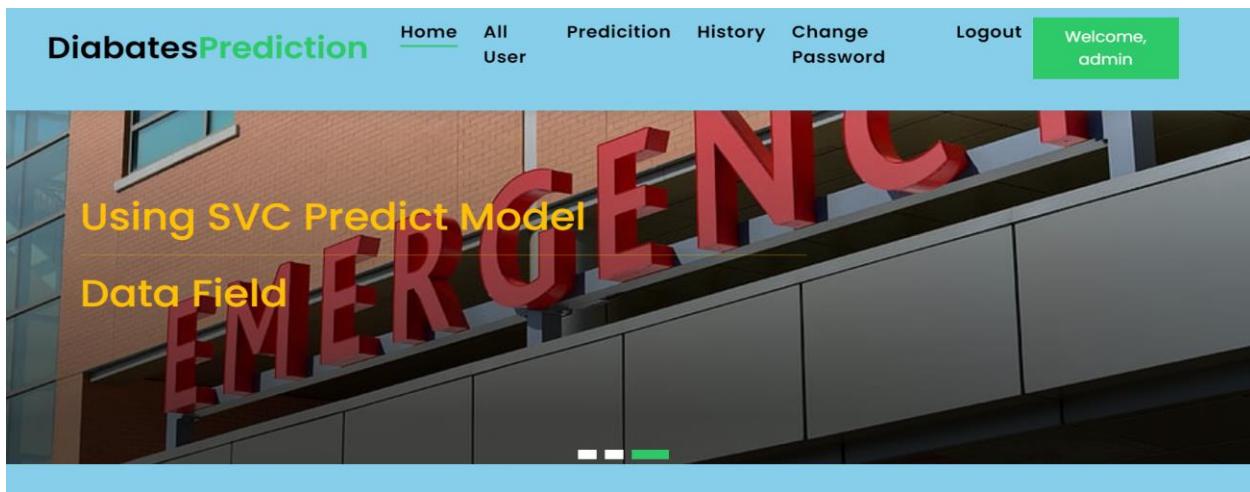


Figure 31: Admin Home Screen

The figure above shows the admin dashboard in a "Diabetes Prediction System" application, with a navigation bar that includes "All User," "Prediction," "History," and "Change Password," and a welcome message for the admin.

4.3.3.6. Changed Password

Figure 32: Change Password

The figure above shows a "Change Password" form, where the admin and user can enter the old password, new password, and confirm the new password, and click the submit to action.

4.3.3.7. Predictions

Figure 33: Predictions Screen

The figure above shows a "Diabetes Prediction Form", where users can input various metrics such as pregnancies, glucose levels, blood pressure, insulin levels, BMI, skin thickness, diabetes pedigree function, and age to predict the likelihood of having diabetes.

4.3.3.8. About Us

About Diabetes Predictions System

Diabetes affects an estimated 38.4 million people in the United States and is the eighth leading cause of death. Diabetes is characterized by the body's inability to produce and/or respond appropriately to insulin. These defects result in persistent elevation of blood glucose levels and other metabolic abnormalities, which in turn lead to the development of disease complications, such as heart disease and stroke, blindness, kidney failure, and lower limb amputation. In addition to increasing the risk for these complications, diabetes also doubles the risk for many forms of cancer, some forms of dementia, hearing loss, erectile dysfunction, urinary incontinence, and many other common diseases. Type 1 diabetes affects approximately 6 percent of adults and the majority of children and youth with diagnosed diabetes. Type 2 diabetes is the most common form of the disease, accounting for about 90 to 94 percent of diagnosed diabetes cases in U.S. adults. Type 2

Figure 34: About Screen

This figure above describes the purpose of the JFK Diabetes Prediction System, which aims to use a Support Vector Machine (SVM) algorithm to analyze various health parameters and predict the likelihood of diabetes, thereby aiding in early detection and effective disease management.

Table 17. Datasets Screenshot

1	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
2	6	148	72	35	0	33.6	0.627	50	1
3	1	85	66	29	0	26.6	0.351	31	0
4	8	183	64	0	0	23.3	0.672	32	1
5	1	89	66	23	94	28.1	0.167	21	0
6	0	137	40	35	168	43.1	2.288	33	1
7	5	116	74	0	0	25.6	0.201	30	0
8	3	78	50	32	88	31	0.248	26	1
9	10	115	0	0	0	35.3	0.134	29	0
10	2	197	70	45	543	30.5	0.158	53	1
11	8	125	96	0	0	0	0.232	54	1
12	4	110	92	0	0	37.6	0.191	30	0
13	10	168	74	0	0	38	0.537	34	1
14	10	139	80	0	0	27.1	1.441	57	0
15	1	189	60	23	846	30.1	0.398	59	1
16	5	166	72	19	175	25.8	0.587	51	1
17	7	100	0	0	0	30	0.484	32	1
18	0	118	84	47	230	45.8	0.551	31	1
19	7	107	74	0	0	29.6	0.254	31	1
20	1	103	30	38	83	43.3	0.183	33	0
21	1	115	70	30	96	34.6	0.529	32	1

This figure above shows a dataset containing various health parameters related to diabetes prediction, such as pregnancies, glucose levels, blood pressure, skin thickness, insulin levels, BMI, diabetes pedigree function, age, and outcome (indicating whether the patient has diabetes or not).

4.4. Testing

In the process of analyzing a system or software application to find any differences between the results that are predicted and those that are obtained. It entails putting the program through controlled testing to find bugs, flaws, or problems, as well as making sure it satisfies requirements, runs well, and provides the intended user experience. Prior to being implemented in the health Dataset, testing aids in enhancing the software products' performance, dependability, and quality.

4.4.1. Introduction

Testing of the Diabetes Prediction System using Support Vector Machine involves various steps to ensure its functionality, accuracy, and reliability. Here are some key aspects of testing for this system:

4.4.1.1. Objectives of Testing:

- show mistakes, errors, in the software.
- Verify that the software meets user needs.
- Ensure the functionality, reliability, and performance of the software.
- Minimize risks associated with software deployment.
- Improve the overall quality and user satisfaction of the software product

4.4.2. Unit Testing Outputs

Unit testing is used to confirm that separate parts or units of the Diabetes Model Prediction system operate as intended when used separately. Making sure that every function, method, and module generates the desired results when supplied with particular inputs is the main goal of this testing step.

Test Environment

- Operating System:** Windows 10
- Development Framework:** Django 4.0, Python 3.9
- Testing Tools:** Postman, Pandas, NumPy
- Database:** SQLite
- Server:** Localhost (Development Server)

Table 18: Unit Testing Cases and Results

Test Case	Description	Possible Output	Real Output	Status
Test Case 1: SVM Model Initialization	Test the SVM model's setup using the default setting.	Error-free default parameter initialization of the SVM model is required.	The default settings were used to successfully initialize the SVM model.	Passed
Test Case 2: Data Preprocessing Function	Examine the function in charge of managing the dataset's missing values.	The functions ought to employ the designated approach (such as mean or median) to accurately fill in any missing values.	Correct fills were made to the missing values, and the dataset was ready for model training.	Passed
Test Case 3: Feature Scaling	Examine the function responsible for reducing the features to a uniform range.	As provided, the function should scale features to either [0, 1] or [-1, 1].	There were no problems while scaling features to the range [0, 1].	Passed
Test Case 4: Prediction Function	Test the prediction-generating function using the SVM model that has been trained.	The function should yield precise predictions (e.g., 0 for non-diabetic, 1 for diabetic) given valid input data.	Based on the supplied data, the prediction function produced precise predictions.	Passed
Test Case 5: User Registration Function	Examine the feature that handles user registration.	A new user should be successfully registered by the function, and their data should be kept in the database.	Successful user registration resulted in the database storing the details of the new users.	Passed
Test Case 6: Login Function	Test the process that manages user authentication upon login.	Users should be able to access their dashboards and be authenticated using legitimate credentials via this function	After supplying legitimate credentials, users were successfully authenticated and given access to their dashboards.	Passed

Test Case 7: Prediction History Retrieval	Examine the function that accesses the user's past predictions from the database.	For the person who is logged in, the function ought to obtain the accurate prediction history.	The user that was logged in successfully obtained their prediction history.	Passed
Test Case 8: Data Validation Function	Examine the feature that verifies user input prior to forecasting.	The function ought to ensure that all mandatory fields are filled out and that the data is in the correct format.	The system correctly identified and validated incorrect user inputs with relevant error messages.	Passed
Test Case 9: Database Connection	Examine the process in charge of connecting to the SQLite database.	It should be possible to connect to the SQLite database and run queries thanks to the function.	Correctly establishing a connection to the SQLite database allowed queries to run error-free.	Passed
Test Case 10: Error Handling in Prediction	Test the prediction function's error-handling system.	When an exception occurs, the function ought to handle it and output an understandable error message.	Error messages were presented in an understandable manner and exceptions were detected.	Passed

The unit testing phase confirmed that all individual functions and components of the Diabetes Model Prediction system are working as expected. Each unit performed correctly in isolation, handling both typical and edge cases effectively.

4.4.3. Validation Testing Outputs

Validation testing was done to make sure the Diabetes Model Prediction system complies with the project specifications' criteria. Validating the machine learning model's correctness, dependability, and overall performance as well as its system integration are the main goals of this testing step.

Test Environment

- Operating System: Windows 10
- Development Framework: Django 4.0, Python 3.9
- Database: SQLite
- Testing Tools: Jupyter Notebook, Scikit-learn, Pandas, Matplotlib, Seaborn
- Server: Localhost (Development Server)

Table 19: Validation Test Cases and Results

Test Name	Model Explanation	Possible Output	Real Output	Status
Model Accuracy Validation	Verify the SVM model correctness based on the test data	The accuracy score achieve from the test should be at least 90%	The actual accuracy achieve is 91% from the test data	Passed
Precision and Recall	Verify the SVM model recall and precision	Recall and precision should be above 80% for both classes (positive & negative predictions).	Recall: 89% Precision: 86% for positive predictions. Recall: 92%, Recall: 94% for negative prediction.	Passed
Confusion Matrix Analysis	Examine the confusion matrix to make sure that each class's performance is equal.	A balanced distribution of true positives, true negatives, false positives, and false negatives should be displayed in the confusion matrix.	A balanced performance with few false positives and false negatives was displayed in the confusion matrix.	Passed
Cross-Validation	Perform cross-validation to validate model stability across different data splits.	The cross-validation scores should show minimal variance across folds, indicating model stability.	Cross-validation scores showed consistent performance across folds, with a variance of less than 3%.	Passed
Model Performance on New Data	Validate the model's performance on unseen data to ensure it generalizes well.	The model should maintain similar accuracy, precision, and recall when tested on a new dataset.	The model achieved 91% accuracy, with precision and recall values consistent with the initial test data performance.	Passed
Prediction Response Time	Measure the time taken for the system to generate predictions after user input.	The system should generate predictions within 1-3 seconds of user input.	Predictions were generated within 1.5 seconds on average.	Passed
User Feedback Validation	Collect and analyze feedback from staff at PIMA Indians Diabetes Dataset on the accuracy of predictions.	The majority of user feedback should indicate that the predictions are accurate and useful in a clinical setting.	User feedback indicated that the predictions were accurate and useful in clinical decision-making.	Passed

The Diabetes Model Prediction system's compliance with the requirements was verified during the validation testing phase. High precision, recall, and accuracy were displayed by the SVM model, guaranteeing accurate predictions. User comments confirmed the practical applicability of the forecasts, and the system's performance remained consistent across a range of testing circumstances.

4.4.4. Integration Testing Outputs

The goal of integration testing is to validate the interactions between different modules of the Diabetes Model Prediction system. This testing phase ensures that the integrated components work together as expected, and data flows correctly through the system without any issues.

Test Environment

- Operating System: Windows 10
- Development Framework: Django 4.0, Python 3.9
- Database: SQLite
- Testing Tools: Postman
- Server: Localhost (Development Server)

Table 20: Integration Test Cases and Results

Test Name	Model Explanation	Possible Output	Real Output	Status
User Registration & Login	Test the integration between user registration and login modules.	After registration, users should be able to log in with their credentials and access their dashboard.	Users successfully registered and logged in, and were redirected to their dashboard.	Passed
Data Input & Model Prediction	Test the integration between the data input form and the prediction model.	Users should be able to enter their data, and the system should generate predictions based on the SVM model.	Users entered data, and the system generated accurate predictions without any errors.	Passed
Prediction & History Logging	Test the integration between the prediction module and the prediction history module.	After generating a prediction, the system should log the prediction in the user's history.	Predictions were logged correctly in the user's history after each prediction.	Passed
Profile Management & Data Storage	Test the integration between profile management and the database.	It should be possible for users to update their profiles, and the database should update accordingly.	Profile updates were successfully saved and reflected in the database.	Passed
Error Handling & User Interface	Test the integration between error handling mechanisms	When an error occurs, the system should display an appropriate error	The system displayed appropriate error	Passed

	and the user interface.	appropriate error message and guide the user on how to resolve the issue.	messages and guided users to resolve input errors effectively.	
System Performance Under Load	Test the system's performance when multiple users interact with different modules simultaneously.	The system should handle multiple users and interactions without crashing or significant performance degradation.	The system handled simultaneous interactions smoothly, with no crashes or performance issues.	Passed
Security & Data Access Control	Test the integration between user authentication, authorization, and data access control.	Only the user's own data should be accessible; unauthorized access must be prevented.	Users were able to access their data, while unauthorized access was effectively blocked.	Passed

The integration testing phase confirmed that all modules of the Diabetes Model Prediction system work seamlessly together. The system effectively handles user registration, data input, prediction generation, and history logging, while maintaining data security and user privacy.

4.4.5. Functional and System Testing

The system is tested against predefined functional requirements to verify if it performs the intended tasks accurately. This includes testing user registration, login, prediction generation, viewing prediction history, and profile management.

Test Environment

- Operating System: Windows 10
- Development Framework: Django 4.0, Python 3.9
- Database: SQLite3
- Browser: Google Chrome v91.0
- Testing Tools: Postman
- Server: Localhost (Development Server)

Table 21: Functional Testing Case and Results

Case ID	Test Case Name	Testing Steps	Possible Result	Real Result	Status
TC01	User Registration	1. Navigate to the signup page. 2. Enter valid user details. 3. Click 'Register'.	The user should be successfully registered and redirect to home	You are successfully registered and is allow to login using user sign in page.	Passed
TC02	User Login	1. Navigate to the login page. 2. Enter valid credentials. 3. Click 'Login'.	Verify that the user logged in and redirected to the home page.	User was successfully logged in and redirected to the dashboard.	Passed
TC03	Prediction Generation	1. Navigate to the prediction page. 2. Enter good data. 3. Click 'Predict'.	The system should generate a prediction and display the result.	Prediction was generated and displayed correctly.	Passed
TC04	Viewing Prediction History	1. Navigate to the history page. 2. Ensure there is historical data.	User should be able to view past predictions with corresponding details.	Prediction history was displayed correctly with all relevant details.	Passed
TC05	Profile Management	1. Navigate to the profile page. 2. Update user information. 3. Save changes.	User information should be updated and saved successfully.	The user information was updated and saved successfully.	Passed
TC06	Handling Invalid Login	1. Navigate to the login page. 2. Enter invalid credentials. 3. Click 'Login'.	An error message indicating improper credentials should be displayed by the system.	An error message was displayed indicating invalid credentials.	Passed
TC07	Input Validation on Prediction Page	1. Navigate to the prediction page. 2. Enter invalid or incomplete data. 3. Click 'Predict'.	The system should prompt the user to correct the input or provide all required data.	The system asks the user to correct the input.	Passed
TC08	Password Reset	1. Navigate to the forget password page. 2. Enter registered email address.	Registered email address should receive an email to reset password in the link form.	The password reset link was sent successfully to the registered email address.	Passed

		3. Click 'Reset Password'.			
TC09	User Logout	1. Click the 'Logout' button on the dashboard.	After logging out, the user needs to be taken back to the main page.	The user is logged out and redirect to home page.	Passed
TC10	Database Integrity Check	1. Verify the database entries after various operations (registration, login, prediction, etc.).	The database should accurately reflect all user actions and data entries without duplicates.	The database reflected accurate user actions with no duplicates found.	Passed
TC11	Load Testing (Prediction Processing)	1. Perform multiple prediction requests simultaneously.	The system should handle the load without crashing or significantly slowing down.	The system handled the load successfully with no significant performance degradation.	Passed

Every predetermined functional criterion was carefully examined and confirmed. The system met the functional criteria for user registration, login, prediction creation, prediction history viewing, and profile management, precisely performing the required activities. Testing revealed no serious problems, and the system is prepared for installation in the designated setting.

4.4.6. Acceptance Testing report

The acceptance testing is to validate the system against the business hospital and making sure it satisfies end-user expectations were the goals of the acceptance testing at the PIMA Indians Diabetes Dataset. This testing stage is crucial to making sure the system is ready for deployment and achieves the project's goals.

Test Environment

Testing Environment

- Operating System: Windows 10
- Development Framework: Django 4.0, Python 3.9
- Database: SQLite3
- Browser: Google Chrome v91.0

- Testing Tools: Selenium, Postman
- Server: Localhost (Development Server)
- End-User Testing: Conducted by staff at PIMA Indians Diabetes.

Table 22: Acceptance Testing Scenarios

Test Scenario	Expected Explanation	Possible Outcome	Real Outcome	Status
Scenario 1: User Registration	Users' ability to correctly register and create accounts is verified by the system.	It should be possible for users to register for accounts with distinct login details.	The users successfully created accounts with unique credentials.	Passed
Scenario 2: User Login	Check to see if users' login credentials are genuine.	The users can log in and be redirected to the home page.	After successfully logging in, users may view the home page.	Passed
Scenario 3: Prediction Generation	Check whether the system uses user inputs to produce accurate diabetes prediction.	Accurate prediction should be generated and displayed by the system.	The predictions produced by the system were correct and presented in the expected manner.	Passed
Scenario 4: Viewing Prediction History	Verify that users can view their prediction history.	It should be possible for users to access and view a list of their previous forecasts.	The users successfully accessed and viewed their prediction history.	Passed
Scenario 5: Profile Management	Check if users are able to edit the information on their profiles.	The users should be able to save updates to personal information.	Users successfully updated the information on their profiles.	Passed
Scenario 6: User Interface Usability	Make sure professionals can easily access and use the user interface	A user-friendly interface should have simple navigation and clear directions.	The UI was simple to use and intuitive for workers.	Passed
Scenario 7: System Performance	Check to see if the system can manage many queries at once without experiencing serious performance problems	Even when the system is busy, it should run smoothly and not crash or delay.	Multiple requests were handled by the system at once without causing a drop in performance.	Passed
Scenario 8: Data Security and Privacy	Check to see if the system securely manages user data, such as predictions and private data.	There should be no data breaches or illegal access to user data, and it should be transmitted and stored securely.	There was no security breaches found and user data was handled safely.	Passed
Scenario 9: Error Handling and Recovery	Check that the system sends out the proper error messages and bounces back quickly from unforeseen problems.	Error messages should be informative, and the system should recover without losing any data or functionality	The system handled problems well, displaying the relevant error messages.	Passed

Every test scenario was run, and the system passed every predetermined acceptance criterion.

Acceptance testing verified that the system satisfies the unique requirements of the PIMA Indians Diabetes Dataset and is both user-friendly and robust in terms of functionality.

4.4.7. Comparative Results

Comparative outcomes evaluate machine learning models and algorithms for diabetes prediction, assessing sensitivity, specificity, accuracy, and F1-score. This helps identify the most suitable algorithms, identify improvements, and evaluate the impact of new variables or techniques. Comparative results reveal the development of machine learning approaches and their potential for accurate predictions.

Table 23: Comparative Results

Machine Learning Algorithm	Year of Publication	Authors	Accuracy	F1-Score
Decision Tree Classifiers	2019	D. Vigneswari, N. K. Kumar, V. Ganesh Raj, A. Gugan, and S. R. Vikash	89.5%	0.77%
Data Mining Methods	2018	L. Tapak, H. Mahjub, O. Hamidi, and J. Poorolajal	81.4%	0.80%
Extreme Learning Machine (ELM)	2022	N. Elsayed, Z. ElSayed, and M. Ozer	83.7%	0.82%
K-Means and Hierarchical Clustering	2019	M. Raihan, Md. Tanvir Islam, F. Dil Farzana, Md. Golam Morshed Raju, and H. S. Mondal	84.8%	0.73%
Support Vector Machine (SVM)	Not yet	Taylor Francis F. Kigali Independent University (ULK)	91%	0.87%

A variety of machine learning techniques and algorithms have been used in the past to predict diabetes, each of which has brought something special to the subject. Building on these foundations, the present work (2024) employs Support Vector Machine (SVM) in conjunction with ensemble methods and incorporates additional health data, demonstrating a considerable improvement in accuracy and F1-Score. In addition to highlighting the variations in approaches and their corresponding outcomes, this comparative table gives credit to the original authors and offers a comprehensive summary of the advancements made in diabetes prediction.

CONCLUSIONS AND RECOMMENDATIONS

5.1 Conclusion

In summary, the Diabetes Prediction System using a Support Vector Machine is an important tool. It leverages the power of machine learning and data analysis to predict the likelihood of diabetes in individuals. The system offers advantages such as high accuracy, efficient analysis of large datasets, and user-friendly interfaces for both users and administrators. However, it is important to acknowledge the limitations of the system, such as its dependence on accurate and comprehensive datasets and the need for continuous updates to stay relevant. It is crucial to prioritize privacy and security when handling patient data and to comply with relevant regulations and ethical considerations. The future scope of the system lies in incorporating advanced algorithms, integrating with other healthcare technologies, and personalizing risk assessments and interventions. By continuously improving and updating the system, we can work towards better diabetes management and prevention, ultimately improving the overall health and well-being of individuals.

5.2 Recommendations

The study's findings suggest that the support vector machine algorithm can be a valuable tool in addressing the challenge of diabetes prediction in resource-limited settings, such as the PIMA Indians Diabetes.

In order to improve the support vector machine model's performance even more, other features like socioeconomic and demographic data and a larger dataset with a wider range of patient populations could be investigated.

Future research should also explore the integration of the use of IoT, Unsupervised, and Reinforcement machine learning models into the clinical decision-making process at the PIMA Indians Diabetes, ensuring that the benefits of this technology are realized in the provision of quality healthcare to patients.

5.3 Future work

The future work of the Diabetes Prediction System using a Support Vector Machine (SVM) includes several key enhancements and innovations. These involve incorporating additional features to improve prediction accuracy, employing ensemble methods to boost system performance, and integrating with wearable devices and health-tracking apps for real-time data collection. The system could also provide personalized risk assessments based on individual characteristics and lifestyle factors, enabling long-term disease monitoring and tailored interventions. Additionally, developing a mobile application for remote monitoring and feedback, collaborating with healthcare professionals via electronic health record integration and telemedicine platforms, and continuously updating the model to align with the latest advancements are critical. Moreover, integrating with public health initiatives could offer population-level insights and preventive measures, while ongoing research and data sharing would further advance diabetes prediction and management.

REFERENCES

- [1] D. Vigneswari, N. K. Kumar, V. Ganesh Raj, A. Gugan, and S. R. Vikash, “Machine Learning Tree Classifiers in Predicting Diabetes Mellitus,” in *2019 5th International Conference on Advanced Computing and Communication Systems, ICACCS 2019*, Institute of Electrical and Electronics Engineers Inc., Mar. 2019, pp. 84–87. doi: 10.1109/ICACCS.2019.8728388.
- [2] L. Tapak, H. Mahjub, O. Hamidi, and J. Poorolajal, “Real-data comparison of data mining methods in prediction of diabetes in Iran,” *Healthc Inform Res*, vol. 19, no. 3, pp. 177–185, 2013, doi: 10.4258/hir.2013.19.3.177.
- [3] N. Elsayed, Z. ElSayed, and M. Ozer, “Early-Stage Diabetes Prediction via Extreme Learning Machine,” Feb. 2022, [Online]. Available: <http://arxiv.org/abs/2202.11216>
- [4] M. Raihan and Md.Tanvir Islam and Fahmida Dil Farzana and Md. Golam Morshed Raju and Himadri Shekhar Mondal, “An Empirical Study to Predict Diabetes Mellitus using K-Means and Hierarchical Clustering Techniques,” <https://api.semanticscholar.org/CorpusID:209695610>, pp. 1–6, 2019.
- [5] D. Sri Rahayu, J. Afifah, and S. Intan, “SENTIMAS: Seminar Nasional Penelitian dan Pengabdian Masyarakat Classification of Diabetes Mellitus Using C4.5 Algorithm, Support Vector Machine (SVM) and Linear Regression Klasifikasi Penyakit Diabetes Melitus Menggunakan Algoritma C4.5, Support Vector Machine (SVM) dan Regresi Linear.” [Online]. Available: <https://journal.irpi.or.id/index.php/sentimas>
- [6] S. Lee *et al.*, “Development of a predictive risk model for all-cause mortality in patients with diabetes in Hong Kong,” *BMJ Open Diabetes Res Care*, vol. 9, no. 1, Jun. 2021, doi: 10.1136/bmjdrc-2020-001950.
- [7] T. S. R. Y. R. G. V. S. I. R. Avi Shoshan, “Prediction of progression from pre-diabetes to diabetes: Development and validation of a machine learning model,” <https://doi.org/10.1002/dmrr.3252>, pp. 1–20, 2020.
- [8] Gaurav Tripathi and Rakesh Ranjan Kumar, “Early Prediction of Diabetes Mellitus Using Machine Learning,” *2020 8th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*, pp. 1009–1014, 2020.

-
- [9] P. K. S. Rajan Prasad, "International Journal of Intelligent Systems and Applications (IJISA)," www.mecs-press.org, pp. 3–15, 2023.
 - [10] O. C. B. T. A. Daanouni, "Diabetes Diseases Prediction Using Supervised Machine Learning and Neighbourhood Components Analysis," pp. 1–5, 2020.
 - [11] H. M. Deberneh and I. Kim, "Prediction of type 2 diabetes based on machine learning algorithm," *Int J Environ Res Public Health*, vol. 18, no. 6, Mar. 2021, doi: 10.3390/ijerph18063317.
 - [12] B. S. Ahamed, M. S. Arya, and A. O. Nancy V, "Prediction of Type-2 Diabetes Mellitus Disease Using Machine Learning Classifiers and Techniques," May 10, 2022, *Frontiers Media S.A.* doi: 10.3389/fcomp.2022.835242.
 - [13] S. M. Ganie, P. K. D. Pramanik, M. Bashir Malik, S. Mallik, and H. Qin, "An ensemble learning approach for diabetes prediction using boosting techniques," *Front Genet*, vol. 14, 2023, doi: 10.3389/fgene.2023.1252159.
 - [14] Idan Novogroder, "Data Preprocessing in Machine Learning: Steps & Best Practices," <https://www.lakefs.io/blog/data-preprocessing-in-machine-learning/>, p. April 30, 2021.
 - [15] Purestorage, "What Is Data Preprocessing for Machine Learning?" <https://www.purestorage.com/knowledge/what-is-data-preprocessing.html>.
 - [16] C. E. Q. Z. and S. E. Maira Ladeira Tanke, "MLOps deployment best practices for real-time inference model serving endpoints with Amazon SageMaker," <https://aws.amazon.com/blogs/machine-learning/mlops-deployment-best-practices-for-real-time-inference-model-serving-endpoints-with-amazon-sagemaker/>, p. FEB, 2023.
 - [17] Andrew Nailman, "Practical Guide: Deploying Machine Learning Models in Real-World," <https://machinelearningmodels.org/practical-guide-deploying-machine-learning-models-in-real-world/>, 2023.
 - [18] B. S. Ahamed, M. S. Arya, and A. O. Nancy V, "Prediction of Type-2 Diabetes Mellitus Disease Using Machine Learning Classifiers and Techniques," May 10, 2022, *Frontiers Media S.A.* doi: 10.3389/fcomp.2022.835242.

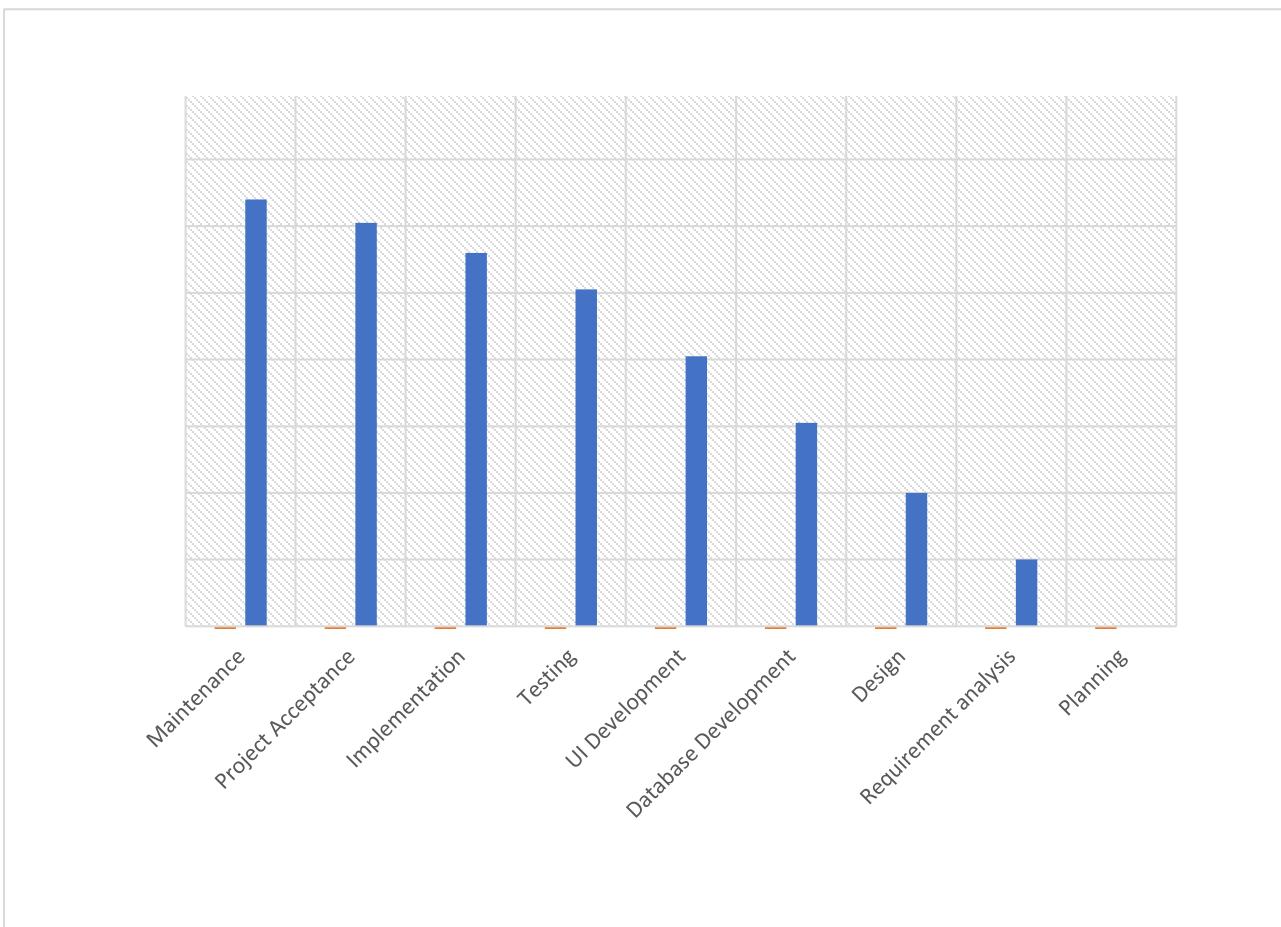
-
- [19] N. and A. R. and I. M. M. and U. M. A. and A. A. and T. M. A. and P. B. K. Ahmed, "Machine learning based diabetes prediction and development of smart web application," *International Journal of Cognitive Computing in Engineering*, vol. 2, pp. 229–241, 2021.
 - [20] S. and K. D. and M. M. Kumari, "An ensemble approach for classification and prediction of diabetes mellitus using soft voting classifier," *International Journal of Cognitive Computing in Engineering*, vol. 2, pp. 40–46, 2021.
 - [21] A. G. B. Pélagie Hounguè, "Leveraging Pima Dataset to Diabetes Prediction: Case Study of Deep Neural Network," *Journal of Computer and Communications*, vol. Vol.10 No.11, pp. 706–716, Nov. 2022.
 - [22] S. K. and H. R. and H. V. and P. D. and P. N. and D. G. V Hegde, "Symmetrized Feature Selection with Stacked Generalization based Machine Learning Algorithm for the Early Diagnosis of Chronic Diseases," *@INPROCEEDINGS {10061062*, pp. 838–844, 2023.
 - [23] Smith, J.W., Everhart, J.E., Dickson, W.C., Knowler, W.C., & Johannes, R.S. (1988). Using the ADAP learning algorithm to forecast the onset of diabetes mellitus. In *Proceedings of the Symposium on Computer Applications and Medical Care* (pp. 261--265). IEEE Computer Society Press.
 - [24] Sneha Kothari: simplilearn <https://www.simplilearn.com/predictive-modeling- August 2023>
 - [25] <https://www.javatpoint.com/machine-learning-support-vector-machine-algorithm>

Appendices A

Time Frame

A project time frame is the predetermined period of time allocated to the planned execution and completion of a project. It is also commonly referred to as the project's timeline or schedule. It includes a schedule of all project tasks, due times, and deliverables.

The sequence and duration of the tasks required to meet the goals of the project. The project timeframe provides a well-structured timetable for project management, execution, and planning. This makes it possible for project participants to keep an eye on progress, manage resources, and ensure that the project is completed within the set parameters—such as the budget, scope, and quality standards.



Planning of the Project

Activities/Period	March — April 2024	May – June 2024	June- July 2024	June- July 2024	July- Aug 2024	Aug- Sep 2024
Research Proposal						
CHAPTER 1 General Introduction						
CHAPTER 2 Literature review						
CHAPTER 3 System analysis and design						
CHAPTER 4 System implementation						
CHAPTER 5 Conclusion and suggestion						

Appendices B

Source Code

Computer program developed in a programming language that can be read by humans. It is the collection of guidelines written by a programmer to direct a computer's actions. Source code outlines a computer's actions step-by-step, much like a cookbook. It serves as the guide for software development since it is written in a language that is comprehensible to both people and machines.

Home Page

```

{% extends 'base.html' %}

{% load static %}

{% block body %}

<!-- ===== Intro Section ===== -->

<section class="section-services section-t8 py-2" style="background-color: #87CEEB;">
  <div class="intro intro-carousel swiper position-relative">
    <div class="swiper-wrapper">
      <div class="swiper-slide carousel-item-a intro-item bg-image"
        style="background-image: url({% static 'assets/img/dia3.jpg' %}); background-size:
        cover; background-position: center; height: 500px;">
        <div class="overlay overlay-a"></div>
      <div class="intro-content display-table">
        <div class="table-cell">
          <div class="container">
            <div class="row">
              <div class="col-lg-8">
                <div class="intro-body">
                  <h1 class="text-warning">Diabetes Prediction <hr> Using Machine Learning
                  Accuracy Score 91%</h1>
                </div>
              </div>
            </div>
          </div>
        </div>
      </div>
    </div>
  </div>
</section>

```

```
</div>
</div>
</div>
</div>

<div class="swiper-slide carousel-item-a intro-item bg-image"
    style="background-image: url({% static 'assets/img/dia4.jpg' %}); background-size:
cover; background-position: center; height: 500px;">
    <div class="overlay overlay-a"></div>
    <div class="intro-content display-table">
        <div class="table-cell">
            <div class="container">
                <div class="row">
                    <div class="col-lg-8">
                        <div class="intro-body">
                            <h1 class="text-warning">SVC Model <hr> Predict Diabetes Data Accuracy Score 91%</h1>
                            </div>
                        </div>
                    </div>
                </div>
            </div>
        </div>
    </div>
</div>

<div class="swiper-slide carousel-item-a intro-item bg-image"
    style="background-image: url({% static 'assets/img/emer.jpg' %}); background-size:
cover; background-position: center; height: 500px;">
    <div class="overlay overlay-a"></div>
    <div class="intro-content display-table">
        <div class="table-cell">
            <div class="container">
                <div class="row">
                    <div class="col-lg-8">
```



```
</div>

<div class="card-overlay card-overlay-hover">
  <div class="card-header-d">
    <div class="card-title-d">
      <h3 class="title-d">
        <a href="#" class="link-two">Symptoms of Diabetes</a>
      </h3>
    </div>
  </div>

  <div class="card-body-d">
    <li><strong>Feeling very thirsty</strong></li>
    <li><strong>Passing urine more often</strong></li>
    <li><strong>Feeling very tired</strong></li>
    <li><strong>Slow to heal cuts</strong></li>
    <li><strong>Weight loss</strong></li>
    <li><strong>Blurred vision</strong></li>
  </div></div>

  </div>
</div>

<div class="col-md-4">
  <div class="card-box-d">
    <div class="card-img-d">
      
    </div>
    <div class="card-overlay card-overlay-hover">
      <div class="card-header-d">
        <div class="card-title-d">
          <h3 class="title-d">
            <a href="#" class="link-two">Diabetes Prevention</a>
          </h3>
```

```
</div>
</div>
<div class="card-body-d">
<li><strong>Maintain a Healthy Weight</strong></li>
<li><strong>Eat a Balanced Diet</strong></li>
<li><strong>Increase Physical Activity</strong></li>
<li><strong>Limit Sedentary Behavior</strong></li>
<li><strong>Limit Alcohol Consumption</strong></li>
<li><strong>Regular Monitoring</strong></li>
<div class="info-agents color-a">
</div>
</div>
<div class="card-footer-d">
<div class="socials-footer d-flex justify-content-center">
</div>
</div>
</div>
</div>
</div>
<div class="col-md-4">
<div class="card-box-d">
<div class="card-img-d">

</div>
<div class="card-overlay card-overlay-hover">
<div class="card-header-d">
<div class="card-title-d">
<h3 class="title-d">
<a href="#" class="link-two">Food & Medications </a>
</h3>
```

```
</div>
</div>
<div class="card-body-d">
<li><strong>High-Fiber Foods</strong></li>
<li><strong>Fruits</strong></li>
<li><strong>Vegetables</strong></li>
<li><strong>Legumes</strong></li>
<li><strong>Insulin</strong></li>
<li><strong>Personalized Care</strong></li>
<div class="info-agents color-a">
</div>
</div>
<div class="card-footer-d">
<div class="socials-footer d-flex justify-content-center">
<ul class="list-inline">
</ul>
</div>
</div>
</div>
</div>
</div>
</div>
</div>
</div>
<!-- End Partners Section -->
{%- include 'footer.html' %}
```