Final Year Project Report

**Full Unit – Final Report**

**Diabetes Prediction Android Application**

Debabrata Bhattacharya 161112015005

Poulamee Thakur 161212015009

Trisha Ghosh 161212015014

Soumya Dutta 161112015020

**B.Tech. in Computer Science & Engineering**

**Supervisor:** Dr. Saikat Maity



Department of Computer Science

JIS UNIVERSITY

April 21, 2020

**Declaration**

We, the students of this group, hereby submit the following declaration:

This report has been prepared on the basis of our own work. Where other published and unpublished source materials have been used, these have been acknowledged.

Student Names:

Debabrata Bhattacharya 161112015005

Poulamee Thakur 161212015009

Trisha Ghosh 161212015014

Soumya Dutta 161112015020

Date of Submission: 25th April 2020

Signature:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Debabrata Bhattacharya

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Poulamee Thakur

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Trisha Ghosh

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Soumya Dutta

**Acknowledgement**

We would like to express our special thank of gratitude to our mentor Dr. Saikat Maity ,Professor & HOD of Computer Science & Engineering who gave us the opportunity to make this project on the topic **Diabetes Prediction Android Application**, which helped us in doing a lot of research and we came to know about many new things so we are very thankful to them. With all respect and gratitude, we would like to thank all the people, who have helped us directly or indirectly to make our work complete and successful.

Table of Contents

[Abstract 5](#_Toc38345678)

[List of Figures 6](#_Toc38345679)

[Project Specification 7](#_Toc38345680)

[Chapter 1: Introduction 8](#_Toc38345681)

[1.1 Objective 8](#_Toc38345682)

[Chapter 2: System Analysis 9](#_Toc38345683)

[2.1 Existing System 9](#_Toc38345684)

[2.2 Proposed System 9](#_Toc38345685)

[2.3 Feasibility Study 10](#_Toc38345686)

[2.4 System Specification 11](#_Toc38345687)

[Chapter 3: Software Description 12](#_Toc38345688)

[3.1 Backend Design 12](#_Toc38345689)

[3.2 Frontend Design 14](#_Toc38345690)

[Chapter 4: Project Description 15](#_Toc38345691)

[4.1 Modules 15](#_Toc38345692)

[4.2 Design Patterns 16](#_Toc38345693)

[4.3 Pseudocode and Code Examples 18](#_Toc38345694)

[4.4 Diagrams and Figures 19](#_Toc38345695)

[Chapter 5: Project Data 22](#_Toc38345696)

[5.1 Overview 22](#_Toc38345697)

[5.2 Columns 22](#_Toc38345698)

[5.3 Dataset Overview 22](#_Toc38345699)

[5.4 Data Snapshot 23](#_Toc38345700)

[5.5 Data Comparisons: 24](#_Toc38345701)

[Chapter 6: System Maintenance 27](#_Toc38345702)

[6.1 Software Updates 27](#_Toc38345703)

[Chapter 7: Conclusion 29](#_Toc38345704)

[Chapter 8: Future Enhancements 30](#_Toc38345705)

[8.1 Expansion of MVC framework: 30](#_Toc38345706)

[8.2 Data Engineering: 30](#_Toc38345707)

[8.3 Data Anonymity: 30](#_Toc38345708)

[Chapter 9: Bibliography 31](#_Toc38345709)

[Chapter 7: References 32](#_Toc38345710)

Abstract

Diabetes is a disease that occurs when your blood glucose, also called blood sugar, is too high. It results in an excessive amount of glucose in the blood associating with a range of severe complications including renal, cardiovascular diseases and blindness. If the diabetes is controlled properly, the severity of the disease is decreased. For this reason, it requires correct educational awareness and routine health check. Therefore, this report presents the development of an android application with a friendly user-interface, for the diagnosis of diabetes.

List of Figures

Chapter 3.4: Diagrams and figures…………………………………………………19

3.4.1 DFD…………………………………………………………………...19

3.4.2 Screenshots………………………………………………………….20

5.5: Data Comparisons…………………………………………………….24

Project Specification

i. It might have happened so many times that you or someone yours need doctors help immediately, but they are not available due to some reason. To support this, we have built a Diabetes Disease Prediction application.

ii. The Diabetes Disease Prediction application is an end user support and online consultation project.

iii. Here, we propose a web application that allows users to get instant guidance on their Diabetes disease through an intelligent system online.

iv. The application is fed with various details and the Diabetes disease associated with those details.

v. The application allows user to share their Diabetes related issues.

vi. Here we use some intelligent data mining techniques to guess the most accurate illness that could be associated with patient’s details and share the accuracy with user.

vii. The system can be used in case of emergency online.

viii. System can return quick diagnosis of presence of diabetes as well as type of diabetes.

# Introduction

Diabetes is rapidly increasing in developing countries including Middle-East, Sub-Saharan Africa, South Asia and Latin America. A study among 191 World Health Organization (WHO) member states in 2000 confirms that 2.8% people for all age groups had diabetes, and this is expected to be 4.4% by the year 2030.

That is the total number of people with diabetes is projected to rise from 171 million in 2000 to 366 million in 2030. Diabetes is reaching epidemic proportions in Bangladesh and in some sectors of our society more than 10% of people have diabetes.

There are severe life-threatening complications of diabetes, such as hypoglycaemic coma, blurred vision, loss of memory, severe impairment of renal function, insulin allergy, acute neuropathy, etc. Every year, 3.2 million people die worldwide because of diabetes-related causes.

## Objective

Prediction of a disease is a procedure by which a doctor searches for the reasons of disease that best explains the symptoms of a patient. Our android-based system is designed for performing prediction based on patient data where patient data can be demographic or clinical. Demographic data is related to the information such as patient’s age, sex, location, income, etc. Clinical data is may be physical signs and laboratory results. Physical signs are those detected by a physical examination of patient, like BMI (body-mass index), pulse rate and blood pressure. Laboratory results are those detected via laboratory tests, like blood test, urine test, etc. The diagnosis of the system is based on the patient data. For example:

1. Test urine for glucose and ketones.
2. Diabetes exhibits if random or fasting blood glucose level:
   1. Fasting plasma glucose >= 7.0 mmol/l
   2. Random plasma glucose >= 11.0 mmol/l.

# System Analysis

Analysis can be defined as breaking up of any whole so as to find out their nature, function etc. It defines design as to make preliminary sketches of; to sketch a pattern or outline for plan. To plan and carry out especially by artistic arrangement or in a skilful wall. System analysis and design can be characterized as a set of techniques and processes, a community of interests, a culture and an intellectual orientation.

The various tasks in the system analysis include the following:

i. Understanding application.

ii. Planning.

iii. Scheduling.

iv. Developing candidate solution.

v. Performing trade studies.

vi. Performing cost benefit analysis.

vii. Recommending alternative solutions.

viii. Selling of the system.

ix. Supervising, installing and maintaining the system.

For proper prediction of diabetes, it will be beneficial if an automated system is developed. Nowadays, mobile-based system has gained tremendous attractions, as it is very user-friendly. Android is a popular mobile operating system. Therefore, the aim of this project is to develop an android-based application for the prediction of diabetes effectively and in a user-friendly platform.

## Existing System

The Existing system is a manual entry for the students. Here the data will be carried out in the hand written registers. It will be a tedious job to maintain the record for the user. The human effort is more here. The retrieval of the information is not as easy as the records are maintained in the hand written registers.

This application requires correct feed on input into the respective field. Suppose the wrong inputs are entered, the application resist to work. So, the users find it difficult to use.

## Proposed System

To overcome the drawbacks of the existing system, the proposed system has been evolved. This project aims to reduce the paper work and saving time to generate accurate results from the client’s data. The system provides with the best user interface. The efficient reports can be generated by using this proposed system.

### Advantages of Proposed System:

1. User can search for diagnosis at any point of time.
2. User can talk about their Diabetes Disease and get instant diagnosis.
3. Doctors get more clients online.
4. Very useful in case of emergency.
5. User can know accuracy of diagnosis instantly.
6. Can be diagnosed about presence and type of diabetes.

## Feasibility Study

Feasibility analysis begins once the goals are defined. It starts by generating broad possible solutions, which are possible to give an indication of what the new system should look like. This is where creativity and imagination are used. Analysts must think up new ways of doing things- generate new ideas. There is no need to go into the detailed system operation yet. The solution should provide enough information to make reasonable estimates about project cost and give users an indication of how the new system will fit into the organization. It is important not to exert considerable effort at this stage only to find out that the project is not worthwhile or that there is a need significantly change the original goal.

Feasibility of a new system means ensuring that the new system, which we are going to implement, is efficient and affordable.

### Technical Feasibility

The technical requirement for the system is economic and it does not use any other additional Hardware and software. Technical evaluation must also assess whether the existing systems can be upgraded to use the new technology and whether the organization has the expertise to use it. This application depends on Android Software Development Kit and it uses Flask Micro Services Framework, Python, Scikit-Learn for the prediction work.

### Economic Feasibility

Development of this application is highly economically feasible. The only thing to be done is making an environment with an effective supervision. It is cost effective in the sense that has eliminated the paper work completely. The system is also time effective because the calculations are automated.

### Operational Feasibility

The system working is quite easy to use and learn due to its simple but attractive interface. User requires no special training for operating the system. Technical performance includes issues such as determining whether the system can provide the right information for the Department personnel student details, and whether the system can be organized so that it always delivers this information at the right place and on time using intranet services. Acceptance revolves around the current system and its personnel.

## System Specification

### HARDWARE REQUIREMENTS (Minimum Requirement)

I. Minimum RAM: 1GB

II. Hard Disk: 128 GB

III. Processor: Intel Pentium 4(1.50 GHZ) or above

### SOFTWARE REQUIREMENTS (Minimum Requirement)

I. Operating system: Windows XP/7/8/10/ Linux/ Unix Variants and Android OS 7.x+

II. Backend End Design: Flask Micro Services Framework, Python, Scikit-Learn

III. Front End Design: Android Software Development Kit

# Software Description

## Backend Design

### Flask Micro Services Framework

Microservices are an application architecture style where independent, self-contained programs with a single purpose each can communicate with each other over a network. Typically, these microservices are able to be deployed independently because they have strong separation of responsibilities via a well-defined specification with significant backwards compatibility to avoid sudden dependency breakage.

Flask is a web framework. This means flask provides you with tools, libraries and technologies that allow you to build a web application. This web application can be some web pages, a blog, a wiki or go as big as a web-based calendar application or a commercial website.

Flask is part of the categories of the micro-framework. Micro-framework is normally framework with little to no dependencies to external libraries. This has pros and cons. Pros would be that the framework is light, there are little dependency to update and watch for security bugs, cons is that some time you will have to do more work by yourself or increase yourself the list of dependencies by adding plugins. In the case of Flask, its dependencies are:

1. Werkzeug a WSGI utility library
2. jinja2 which is its template engine

### Python

Python is a popular programming language. It is used for:

1. web development (server-side),
2. software development,
3. mathematics,
4. system scripting.

### What can Python do?

1. Python can be used on a server to create web applications.
2. Python can be used alongside software to create workflows.
3. Python can connect to database systems. It can also read and modify files.
4. Python can be used to handle big data and perform complex mathematics.
5. Python can be used for rapid prototyping, or for production-ready software development.

### Why Python?

1. Python works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc).
2. Python has a simple syntax similar to the English language.
3. Python has syntax that allows developers to write programs with fewer lines than some other programming languages.
4. Python runs on an interpreter system, meaning that code can be executed as soon as it is written. This means that prototyping can be very quick.
5. Python can be treated in a procedural way, an object-orientated way or a functional way.

### Scikit-Learn

Scikit-learn is a free machine learning library for Python. It features various algorithms like support vector machine, random forests, and k-neighbours, and it also supports Python numerical and scientific libraries like NumPy and SciPy. Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use. The library is focused on modelling data. It is not focused on loading, manipulating and summarizing data. Some popular groups of models provided by Scikit-learn include:

1. **Clustering**: for grouping unlabelled data such as KMeans.
2. **Cross Validation**: for estimating the performance of supervised models on unseen data.
3. **Datasets**: for test datasets and for generating datasets with specific properties for investigating model behaviour.
4. **Dimensionality Reduction**: for reducing the number of attributes in data for summarization, visualization and feature selection such as Principal component analysis.
5. **Ensemble methods**: for combining the predictions of multiple supervised models.
6. **Feature extraction**: for defining attributes in image and text data.
7. **Feature selection**: for identifying meaningful attributes from which to create supervised models.
8. **Parameter Tuning**: for getting the most out of supervised models.
9. **Manifold Learning**: For summarizing and depicting complex multi-dimensional data.
10. **Supervised Models**: a vast array not limited to generalized linear models, discriminate analysis, Naive Bayes, lazy methods, neural networks, support vector machines and decision trees.

## Frontend Design

### Android Software Development Kit

The Android SDK (software development kit) is a set of development tools used to develop applications for Android platform. The Android SDK includes the following:

1. Required libraries
2. Debugger
3. An emulator
4. Relevant documentation for the Android application program interfaces (APIs)
5. Sample source code
6. Tutorials for the Android OS

Every time Google releases a new version of Android, a corresponding SDK is also released. To be able to write programs with the latest features, developers must download and install each version’s SDK for the particular phone.

The development platforms that are compatible with SDK include operating systems like Windows (XP or later), Linux (any recent Linux distribution) and Mac OS X (10.4.9 or later). The components of Android SDK can be downloaded separately. Third party add-ons are also available for download. Although the SDK can be used to write Android programs in the command prompt, the most common method is by using an integrated development environment (IDE). The recommended IDE is Eclipse with the Android Development Tools (ADT) plug-in. However, other IDEs, such as NetBeans or IntelliJ, will also work. Most of these IDEs provide a graphical interface enabling developers to perform development tasks faster. Since Android applications are written in Java code, a user should have the Java Development Kit (JDK) installed.

# Project Description

This system developed will reduce the manual work and avoid redundant data. By maintaining the data manually, then efficient diagnosis cannot be done. The system can work efficiently based on the clients’ data. As the data are maintained in registers it is a tough task for admin and the doctors to maintain it for long time. Instead the software can keep long and retrieve the information when needed.

## Modules

The system comprises of 2 major modules as follows:

### Admin Module

1. Add Training Data
2. View User Details
3. View Accuracy

### User Module

1. Register (With Details like Age, Sex, etc.)
2. Check Diabetes (By providing Details like

* Age in Year
* Gender
* Chest Pain Type
* Fasting Blood Sugar
* Resting Electrographic Results
* Exercise Induced Angina
* The slope of the peak exercise ST segment
* CA – Number of major vessels colour by fluoroscopy
* Test Blood Pressure
* Serum Cholesterol
* Maximum Diabetes rate achieved
* ST depression induced by exercise
* Whether stomach is empty.

## Design Patterns

Design patterns are used to represent some of the best practices adapted by software developers. A design pattern systematically names, motivates, and explains a general design that addresses a design problem in software systems. It describes the problem, the solution, when to apply the solution, and its consequences. It also gives implementation hints and examples.

Our design pattern follows an MVC approach (Model-View-Controller). Being a primarily mobile application, this approach allows us to build a swiftly executing, reactive, and user-guided application.

### The Model

Our model is based on a python script that executes on a server. The server executes this model script for each query that is passed on through the input design.

### View

The view layer is an android based application that receives input from the user and displays the relevant output.

### Controller

All modern applications follow the PWA (Progressive Web App) approach to designing the Controller view. The server backend guides the View layer through user interactions, and informs the model layer of the input. It parses the input from the view layer, call the model to perform computations, and sends appropriate output back to the view layer.

### Input Design

Input design is part of overall system design that requires special attention designing input data is to make the data entered easy and free from errors. The input forms are designed using the controls available in python and web frameworks. Validation is made for each and every data that is entered.

Input design is the process of converting the user originated inputs to a computer-based format. A system user interacting through a workstation must be able to tell the system whether to accept the input to produce reports. The collection of input data is considered to be most expensive part of the system design. Since the input has to be planned in such a manner so as to get relevant information, extreme care is taken to obtain pertinent information

This project, following the MVC approach (outlined in the 3.3 Design Patterns section), receives input from the user in the form of text and integer data through an Android Application Interface. Each data unit is then converted into a single data object block by android libraries (Bundle), sliced up into streams (Http Intents), and sent off to the server.

### Output Design

Output design for this application generally refers to the results and information that are generated by the system for many end-users; output is the main reason for developing the system and the basis on which they evaluate the usefulness of the application.

The output is designed in such a way that it is attractive, convenient and informative. Forms are designed with various features, which make the console output more pleasing.

As the outputs are the most important sources of information to the users, better design should improve the system’s relationships with us and also will help in decision making. Form design elaborates the way output is presented and the layout available for capturing information.

One of the most important factors of the system is the output it produces. This system refers to the results and information generated. Basically, the output from a computer system is used to communicate the result of processing to the user.

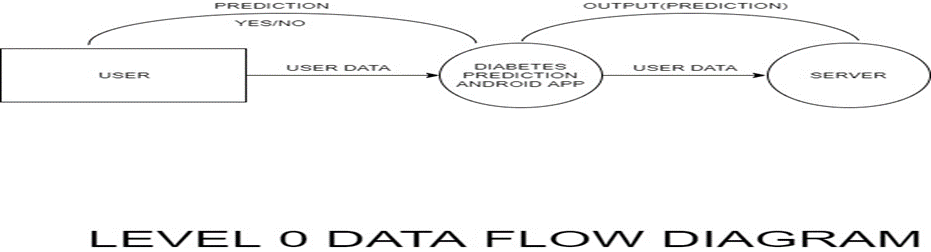
This project, following the MVC approach (outlined in the 3.3 Design Patterns section), receives input from the user in View Layer through an Android Application Interface. Data is then processed by the Model Layer, and a binary output is generated which displays to the user whether they have diabetes or not. This output is transmitted from the server (Controller Layer) to the Android application (View Layer) through a Flask application interface.

## Pseudocode and Code Examples

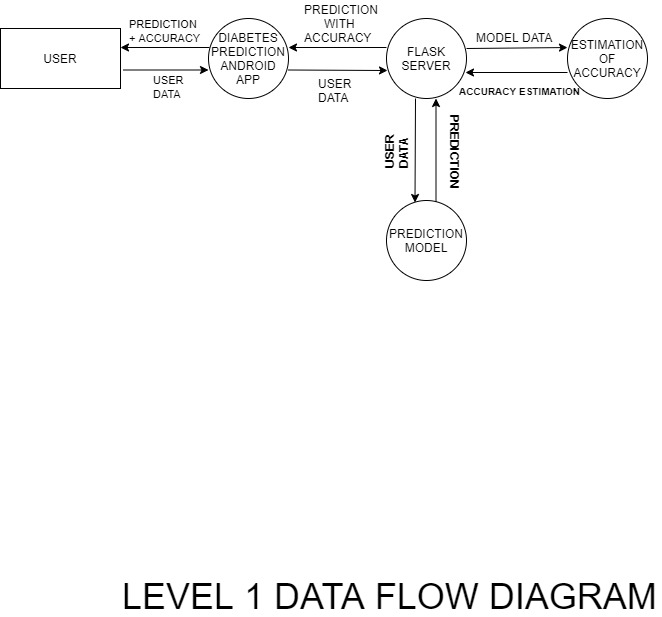
|  |  |
| --- | --- |
|  | from sklearn.model\_selection import train\_test\_split |
|  | from sklearn.neural\_network import MLPClassifier |
|  | from sklearn.preprocessing import StandardScaler |
|  | from sklearn.externals.joblib import dump |
|  | from sklearn.externals.joblib import load  import pandas as pd |
|  | diabetes = pd.read\_csv('diabetes.csv') |
|  | X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(diabetes.loc[:, diabetes.columns != 'Outcome'], diabetes['Outcome'], stratify=diabetes['Outcome'], random\_state=66) |
|  | scaler = StandardScaler() |
|  | X\_train\_scaled = scaler.fit\_transform(X\_train) |
|  | X\_test\_scaled = scaler.fit\_transform(X\_test) |
|  | model = MLPClassifier(random\_state=0) |
|  | model.fit(X\_train\_scaled, Y\_train) |
|  | print("Accuracy on training set: {:.3f}".format( |
|  | model.score(X\_train\_scaled, Y\_train))) |
|  | print("Accuracy on test set: {:.3f}".format(model.score(X\_test\_scaled, Y\_test))) |
|  | # save the model to disk |
|  | filename = 'finalized\_deep\_learning\_model.sav' |
|  | dump(model, filename) |
|  | # sometime later... |
|  | # load the model from disk |
|  | loaded\_model = load(filename) |
|  | print("Test set score: {:.3f}".format(loaded\_model.score(X\_test\_scaled, Y\_test))) |
|  | ''' |
|  | Accuracy on training set: 0.823 |
|  | Accuracy on test set: 0.802 |
|  | Test set score: 0.802 |
|  | ''' |

## Diagrams and Figures

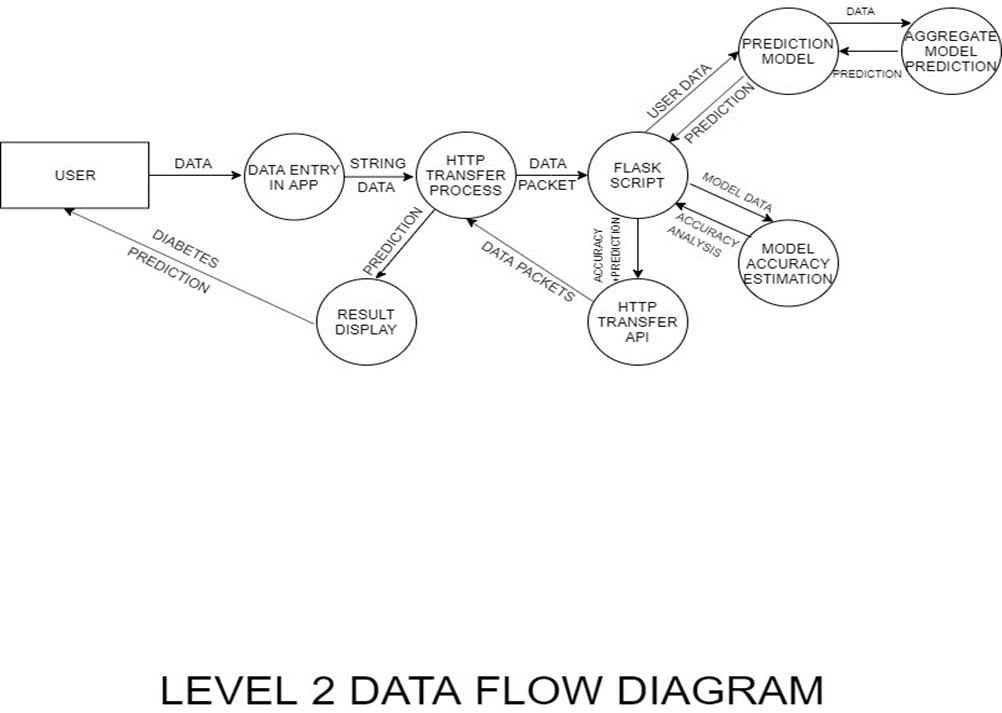
### DFD



1. Level 0 DFD



1. Level 1 DFD



1. Level 2 DFD

### Screenshots

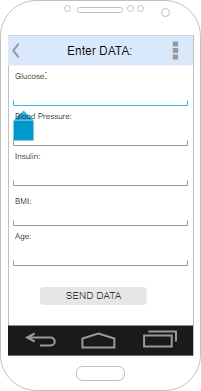
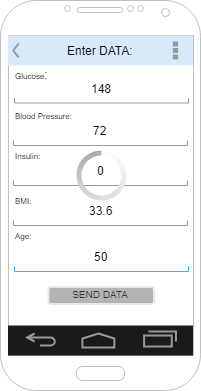
** **

Image I Image II

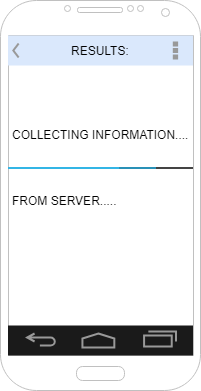
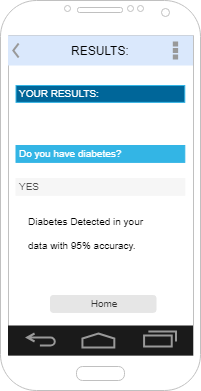
** **

Image III Image IV

# Project Data

## Overview

The primary dataset for this project is the Pima Indians Dataset, hosted at UCI Machine Learning Repository.

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

The datasets consist of several medical predictor variables and one target variable, Outcome. Predictor variables includes the number of pregnancies the patient has had, their BMI, insulin level, age, and so on.

## Columns

**Pregnancies**: Number of times pregnant

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

**Blood Pressure**: Diastolic blood pressure (mm Hg)

**Skin Thickness**: Triceps skin fold thickness (mm)

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

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

**Diabetes Pedigree Function**: Diabetes pedigree function

**Age**: Age (years)

**Outcome**: Class variable (0 or 1) 268 of 768 are 1, the others are 0

## Dataset Overview

Number of variables: 9

Number of observations: 768

Total Missing (%): 0.0%

Total size in memory: 54.1 KiB

Average record size in memory: 72.1 B

### Variables types:

Numeric: 8

Categorical: 0

Boolean: 1

Date: 0

Text (Unique): 0

Rejected: 0

Unsupported: 0

### Warnings:

BMI has 11 / 1.4% zeros

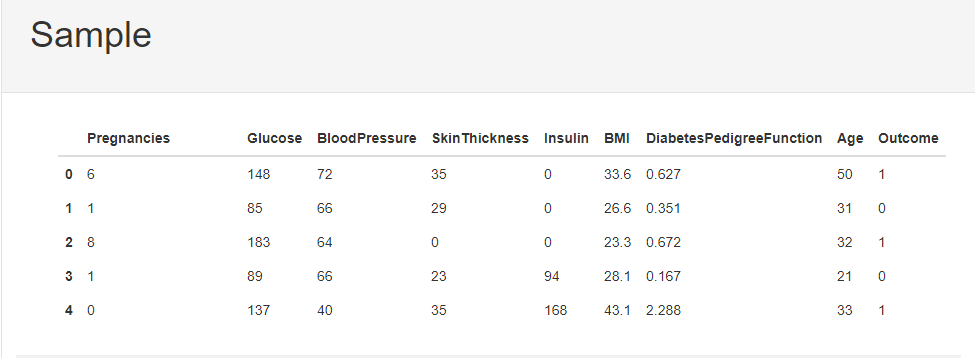
Blood Pressure has 35 / 4.6% zeros

Insulin has 374 / 48.7% zeros

Pregnancies has 111 / 14.5% zeros

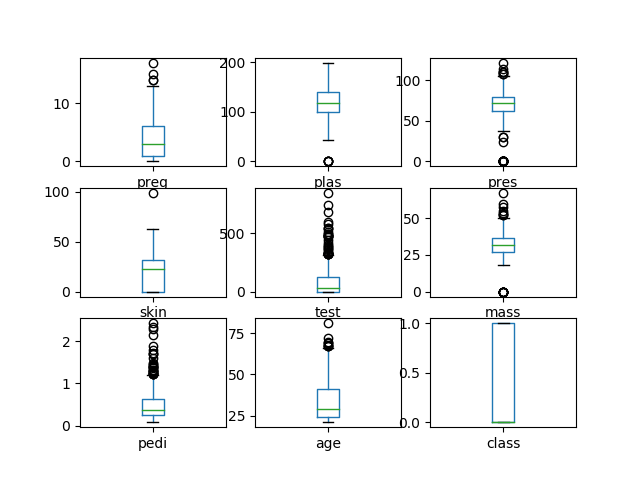
Skin Thickness has 227 / 29.6% zeros

## Data Snapshot

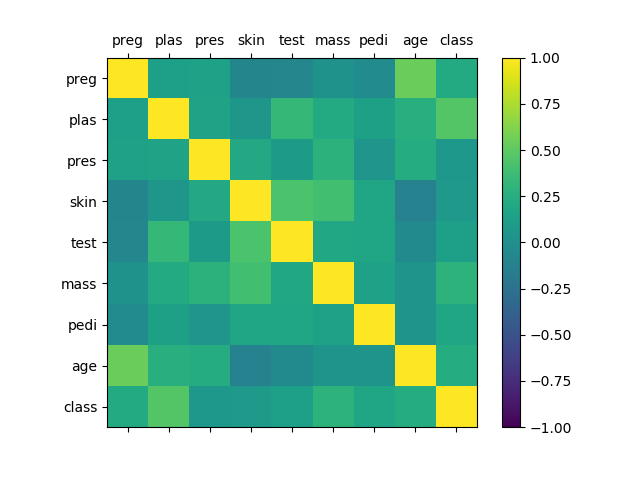


1. Data Snapshot

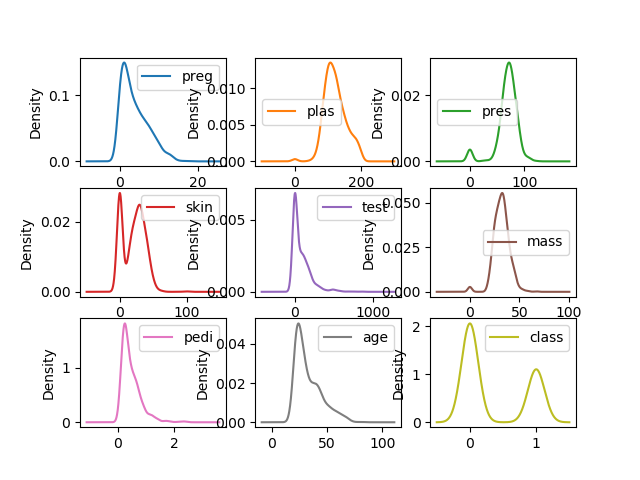
## Data Comparisons:



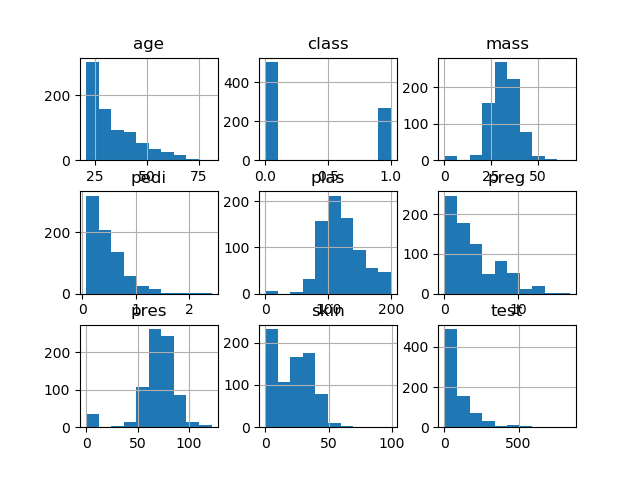
1. Box Plots



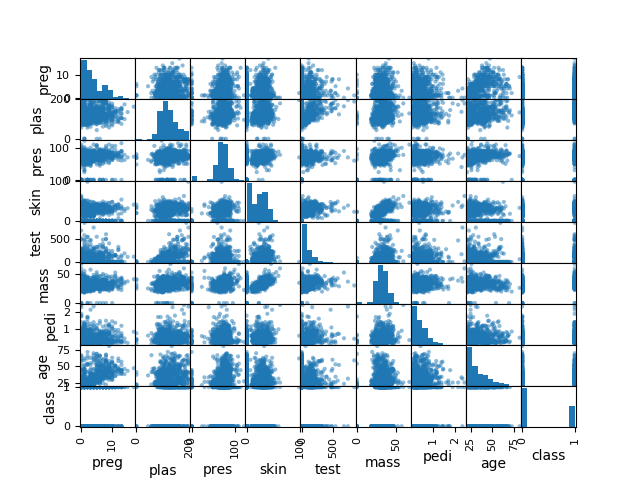
1. Correlation Matrix



1. Density Plots



1. Histograms



1. Scatter Plots

# System Maintenance

Software maintenance is far more than finding mistakes. Provision must be made for environment changes, which may affect either the computer, or other parts of the computer-based systems. Such activity is normally called maintenance. It includes both the improvement of the system functions and the corrections of faults, which arise during the operation of a new system.

It may involve the continuing involvement of a large proportion of computer department recourses. The main task may be to adapt existing systems in a changing environment.

Back up for the entire database files are taken and stored in storage devices like flash drives, pen drives and disks so that it is possible to restore the system at the earliest. If there is a breakdown or collapse, then the system gives provision to restore database files. Storing data in a separate secondary device leads to an effective and efficient maintains of the system. The nominated person has sufficient knowledge of the organization’s computer passed based system to be able to judge the relevance of each proposed change.

## Software Updates

### Android App Updates

Keeping the app up-to-date on our users’ devices enables them to try new features, as well as benefit from performance improvements and bug fixes. Although some users enable background updates when their device is connected to an unmetered connection, other users may need to be reminded to update. In-app updates are a Play Core library feature that introduces a new request flow to prompt active users to update our app.

In-app updates work only with devices running Android 5.0 (API level 21) or higher, and requires Play Core library 1.5.0 or higher

### Check for Update Availability

Before requesting an update, we need to first check if one is available for our app. To check for an update, we use App Update Manager.

The result contains the update availability status. If an update is available and the update is allowed, the returned App Update Info also contains intent to start the update.

If an in-app update is already in progress, the result will also report the status of the in-progress update.

### Start an Update

After checking that we are able to update the app, we can request an update using App Update Manager startUpdateFlowForResult (). However, we should be mindful how often you request updates to avoid annoying or tiring our users. That is, we should limit requesting in-app updates to only the changes that are important to the functionality of our app.

Each App Update Info instance can be used to start an update only once. To retry the update in case of failure, we need to request a new App Update Info and check again that the update is available and allowed.

The type of update we request determines the next steps we need to take.

### Get a call-back for Update Status

After starting an update, we can use an onActivityResult () call-back to handle an update failure or cancellation.

### Python

Dependencies are defined in the YAML file of the project, which is situated in the top-level directory. Anaconda is used as the primary platform for all python applications in this project and all updates are handled by the conda package manager. To update python dependencies, simply type:

conda --upgrade

# Conclusion

This report presents the development of an automated android application for the prediction of diabetes. The system is user friendly and has convenient user interface. The reliability, robustness and correctness of the system exact need more elaborate study.

# Future Enhancements

## Expansion of MVC framework:

MVC framework should be fully implemented. All web based communication should be in the form of API calls between the Android app and the flask server.

## Data Engineering:

Data Engineering should be implemented, now that the model has been built. A data pipeline should be set up so that the model can learn from New data as it comes along.

Job pipeline libraries such as 'pickle' or 'joblib' should be used for this purpose.

## Data Anonymity:

Data should be anonymous to the developer. Using job pipelining libraries, the data pipeline should be implemented such that new data is learned from without bias. This will bring the application in compliance with GDPR and FESTA/SOSTA laws in Europe and the US.

# Bibliography

1. Dave Cohen and Carlos Matos. *Third Year Projects – Rules and Guidelines.* Royal Holloway, University of London, 2013.
2. Adam Brown. *Bright Spots & Landmines: The Diabetes Guide I Wish Someone Had Handed Me*. 1st Edition.
3. Paul Barry. *Head-First Python.* 2nd edition*.*
4. Zed A. Shaw *Learn Python the Hard Way.* 3rd Edition.
5. Bill Philips & Brian Hardy. *Android Programming: The Big Nerd Ranch Guide*. Big Nerd Ranch Guides.
6. Ian G. Clifton. *Android User Interface Design: Turning Ideas and Sketches into Beautifully Designed Apps*.

Chapter 7: References

1. https://blog.feedspot.com/diabetes\_blogs
2. https://www.diabetes.org
3. https://www.cdc.gov/diabetes/basics/
4. https://machinelearningmastery.com
5. https://elitedatascience.com/learn-machine-learning
6. http://developer.android.com/training/