

CSCE 5214.004 SDAI

Group 9: DIABETES PREDICTION TOOL

Phase 2: Description of used Machine Learning & User Interface Design

Aditya Vadrevu - 11601517

Suhas Siddarajgari Tellatakula - 11626111

Srikar Reddy Gunna - 11594012

Sai Praneeth Reddy Avula - 11582402

Indu Shashi Guda - 11618418

1. Machine Learning

Project: <https://github.com/Aditya-Mankar/Diabetes-Prediction>

▪ Used Machine Learning

In this AI system, the author has employed different supervised Machine Learning algorithms like:

- i. Logistic Regression
- ii. K Nearest neighbors
- iii. Support Vector Classifier
- iv. Naive Bayes
- v. Decision tree
- vi. Random Forest

▪ Used Dataset

The dataset used in this system is the *Pima Indians dataset* from the UCI Machine learning repository:

<https://github.com/Aditya-Mankar/Diabetes-prediction/blob/master/diabetes.csv>

This dataset consists of various featuring columns like Pregnancies, Glucose, Blood Pressure, Skin Thickness, Insulin, BMI (Body Mass Index), Diabetes Pedigree Function, Age and finally the Outcome.

	A	B	C	D	E	F	G	H	I
	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
1									
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

■ Data Preparation activities

i. *Data gathering & Importing libraries*

This step involves importing all of the common libraries, including numpy, pandas, matplotlib, and seaborn. For procedures involving linear algebra, data frames are used with pandas, and graphs are plotted using matplotlib and seaborn. The pandas function `read_csv()` is used to import the dataset.

ii. *Data Preprocessing*

The zero values that need to be substituted represent the missing values in this dataset. NaN is used to replace zero values so that the `fillna()` command can quickly impute missing values. We use `MinMaxScaler()` to execute feature scaling on the dataset, which scales the entire set so that it is between 0 and 1. For many algorithms, this preprocessing step is crucial. Age, BMI, Insulin, Glucose, and Insulin are all strongly connected with the result. Therefore, we choose X for these attributes and Y for the result. Following that, the dataset is divided using `train_test_split` in an 80:20 ratio.

iii. *Data Modelling*

A supervised classification model called a "Support Vector Classifier" (SVC) aims to classify data based on a maximal margin hyperplane constructed with support vectors. This hyperplane serves as a decision boundary that divides categories. Support vectors, which are the outliers, are used in its construction. The decision boundary is chosen to be the hyperplane with the largest margin.

By employing a kernel method that implicitly translates the input to high dimensional vector spaces, SVCs can categorize both linear and non-linear data.

With the use of this kernel trick, the lower-dimensional feature space is transformed into a higher-dimensional, linearly separable feature space. For instance, data in a 2D may not be linearly separable but becomes linearly separable when converted to a 3D using the kernel function.

Along with this, the author used 5 other Machine learning models for data modelling like Logistic Regression, K Nearest neighbors, Naive Bayes, Decision tree, Random Forest.

▪ **Validation activities**

The author has chosen three metrics accuracy_score, confusion matrix and classification report for evaluating and validating the models.

- i. accuracy_score
- ii. confusion matrix
- iii. classification report (precision, recall, f1 score, support)

▪ **Visualization and Pipelines**

The author has chosen following visualization techniques to perform the Exploratory Data Analysis and analyze the data graphically using Matplotlib, Seaborn, or Plotly.

- i. Heat Map
- ii. Histograms
- iii. Box plots
- iv. Scatter plots

And, there were no pipelines identified in the existing system.

▪ **Critics**

- i. The overall process of developing the existing prediction system is admirable and should be noted as the author follows the most used and proven methodologies/approach like Data cleaning, preprocessing, modelling, then evaluation.
- ii. Also, the author has employed 6 different machine learning algorithms to predict the diabetes of the candidate which is not usual as it is common practice to model and evaluate only 1 or 2 machine learning algorithms.
- iii. However, there can be more that can be done to improve this system as it has some potential setbacks like limited and poor quality data, poor and minimal designed user interface, lack of performance improvement measures. Further, we will be addressing these issues and propose few ideas to overcome.

- **Proposal of alternative approach/extension**

- i. The current data set used is Indians Diabetics, we can extend the dataset set to train from the different Predictor variables like demographic area, biometrics etc.
- ii. In the current code, there are limited UI features, but we will be introducing a report feature having analysis and downloaded options.
- iii. The implementation of a bagging model will be introduced in the context of diabetes prediction. It works by training multiple instances of a base model on different subsets of the training data and combining their predictions to make a final, more robust prediction. When predicting diabetes, employing the bagging method entails training several models, which could include decision trees or alternative classifiers, on distinct subsets of the accessible patient data. Subsequently, these individual predictions are combined to generate a more dependable diabetes risk or diagnosis prediction. This approach serves to alleviate overfitting and enhance the predictive model's accuracy.
- iv. We will introduce cross validation, Hyperparameter Tuning to check the performance of the model.

2. UI Design

- **Description of UI extension**

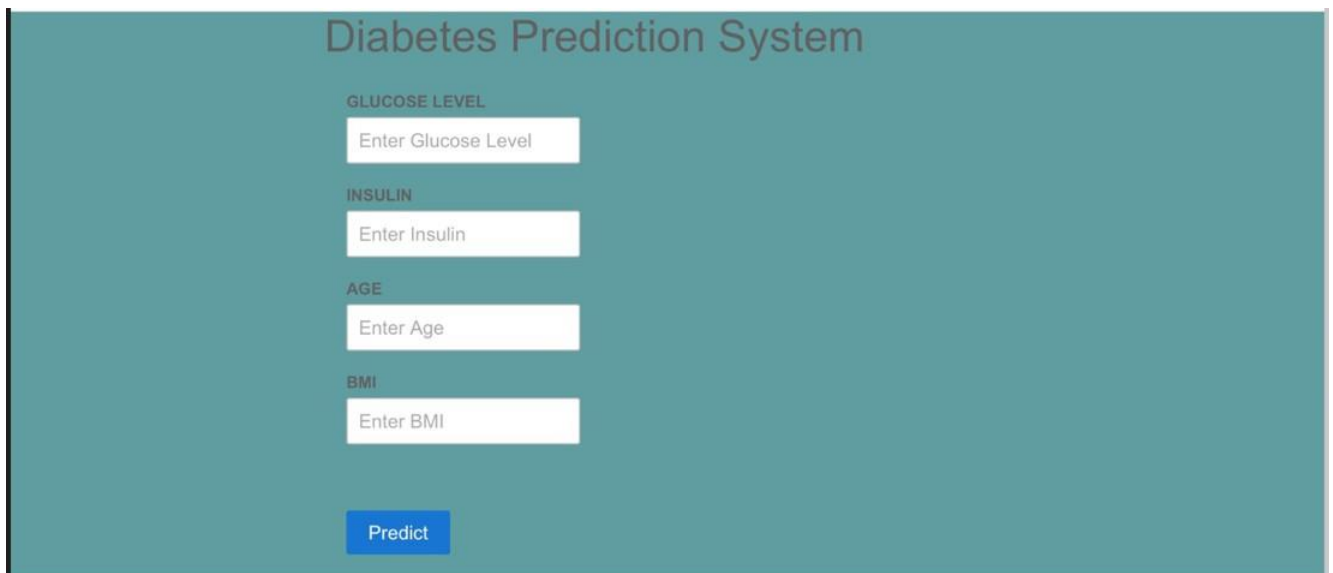
The current existing User Interface works and is simple but is very minimal. So, in order to give the user more information about the prediction, we are going to extend the current User interface with additional features like:

- showcasing the analytics of the Diabetes data on which the tool is trained on.
- allowing the user to view and download a report about his prediction result from the tool.
- enable the user to view some helpful tips and information about Diabetes.
- Creating a range of prediction outcomes as high and normal levels indicating user the risk level of the situation.

We will add these features by creating webpages in html and styling with css and other UI elements using Angular JavaScript framework.

- **UI mockup**

- Header:** The header of the page displays the title "Diabetes Prediction System" indicating the purpose of the web application.
- Input Form:** Users can input the following information:
 - Glucose Level – Users needs to enter their glucose level in the input field provided.
 - Insulin – Users needs to enter their insulin level in the input field.
 - Age – In this tab users can enter their age, which is masked as a password field for privacy.
 - BMI – Users needs to enter their BMI in the input field.
 - Predict – Users can click this button after entering their data to initiate the prediction process.



The image shows a UI mockup for a "Diabetes Prediction System". The title is centered at the top in a large, dark font. Below the title, there are four input fields, each with a label above it: "GLUCOSE LEVEL", "INSULIN", "AGE", and "BMI". Each input field contains the placeholder text "Enter [field name]". The "AGE" field is styled as a password field with a masked input. At the bottom of the form, there is a blue button labeled "Predict". The entire form is set against a teal background.

- iii. **Results:** The page has a "Results" section where the predictions and results are displayed. A table displays the date and time along with the diabetes result. Currently, there is one entry in the table. Users can expect to see their prediction results displayed in a similar format after clicking the "Predict" button. Users can download the detailed report with dates and time by clicking on the Results.

Results

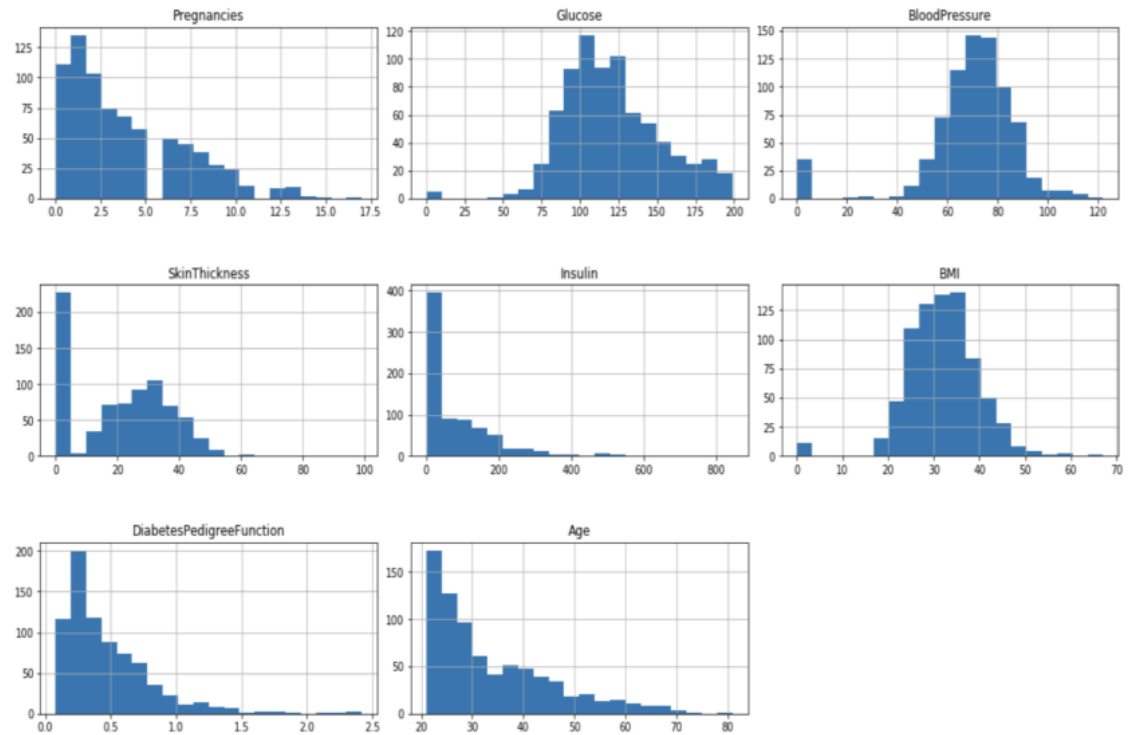
Date and Time	Diabetes Result
10/07/2023	5.23

- iv. **Prediction Table:** Another section titled "Prediction Table" provides information related to different measurement levels and their corresponding categories. This table offers context to users about the interpretation of their results.

Prediction Table

Measurments	Level
10 to 11.1 millimoles per liter (mmol/L)	High
Less than 180 mg/dL	Normal

- v. **Graph Design:** There is a "Graph Design" section that includes a switch-like element. By interacting with this switch, users may be able to enable or disable specific graph design settings. The exact functionality is not described in the provided code, but it suggests that users can customize the appearance of graphs or charts.



UI Flow:

Users would start by entering their glucose level, insulin level, age, and BMI into the input fields. After entering the data, they would click the "Predict" button. The system would then process the input and display the prediction results in the "Results" section, along with the date and time of the prediction. Users can refer to the "Prediction Table" for context on what the prediction results mean in terms of diabetes levels. If user wants to get additional information, he clicks on graph design switch which shows the graphical visualizations.

3. Challenges and Lessons Learned

Challenges

- **Overfitting:** One of the key challenges in developing a Diabetes prediction system is the model's overfitting of data. When a model learns the training data too well and is unable to generalize to new data, that is referred as overfitting. This may occur if the model is very complicated or if there are insufficient training data. Making sure the model is not overfitted and error-free can be challenging. Overfitting can be avoided utilizing a variety of strategies, including regularization and early termination. Cross-validation is a technique that can help you to avoid overfitting and get a more accurate estimate of your model's performance.
- **Underfitting:** Making the model without any underfitting can be other challenge. Underfitting can happen when a model is extremely straightforward and the dataset has more noise. In this case, model is unable to produce correct predictions because it has not learned the training data sufficiently.
- **Complex Models:** The quantity of parameters in a supervised machine learning model affects the model's complexity. In addition to being more likely to overfit the training set, more complicated models may also be able to identify more intricate connections between the characteristics and the target variable. Hence, employing such complex machine learning models can be challenging.
- **Uncertainty:** It's critical to develop user interfaces that can gracefully accept uncertainty and failures because no machine learning model is flawless. This could be telling users how confident the model is in its predictions or enabling them to fix mistakes in the training set of data.

Lessons Learned

- **Informative UI:** Users occasionally might need to know why the model generated a specific forecast. This can be challenging, particularly for intricate models. It's critical to create user interfaces (UIs) that can succinctly and clearly explain the model's predictions to people. Real-time predictions are made using some supervised machine learning algorithms. This may present new UI design problems, such as making sure the UI is responsive and that the model's predictions are shown promptly.
- **Data PreProcessing:** Usually, more pertinent info is preferable. Your model will generalize to new data more effectively the more data you have to train it on. However, it is crucial to guarantee that the data is accurate and a good representation of the issue in the actual world that you are attempting to solve. The performance of your model will be significantly influenced by the type and quantity of training data you use. A well-labeled dataset indicative of the real-world data you will be using the model to forecast on is essential. To do this, you must eliminate any outliers, deal with missing values, and arrange the data in a way that is appropriate for the machine learning technique you intend to employ.
- **Feature engineering:** This can make huge difference. The process of engineering new features from already-existing data. Well-designed features can help your model understand more intricate data relationships and perform better. You can improve your model by using hyperparameter adjustment. Hyperparameters are factors that affect how the machine learning algorithm behaves. In order to identify the values that result in the greatest model performance, hyperparameter tuning entails experimenting with various combinations of hyperparameter values.
- **Right Machine Learning Algorithm:** The process of training supervised machine learning models is not one-size-fits-all. The optimal strategy will rely on the current issue to resolve and the data at our disposal. There are numerous supervised machine learning algorithms out there, each with unique advantages and disadvantages. It's crucial to pick an algorithm that works well with the data you have and the problem you're trying to solve.
- **Model Evaluation:** Utilize a held-out test set to evaluate the model. It's crucial to assess a model's performance on a held-out test set once you've trained it. The model was not trained with this batch of data. You can get a more accurate prediction of the model's performance on new data by analyzing it on a held-out test set. Ensemble learning combines different models to make predictions that are more accurate.