IBM NAAN MUDHALVAN

ARTIFICIAL INTELLIGENCE PROJECT

AI-BASED DIABETES PREDICTION SYSTEM PHASE 4

INTRODUCTION:

An AI-based diabetes prediction system is to develop a model that can accurately predict the likelihood of an individual developed diabetes based on demographic, lifestyle, and health factors. The system should be able to analyse large amount of data, including medical records, genetic information, and lifestyle choices, to provide personalized predictions. The goal is assist to healthcare professionals in identifying individuals who are high risk of developing diabetes, allowing for early intervention and prevention strategies. The system should also user-friendly and easily accessible to both healthcare professionals and individual looking to access their own risk of diabetes.

Artificial intelligence (AI) has become a powerful tool in the field of healthcare, aiding in the diagnosis and treatment of various diseases. One such application is the prediction diabetes, a chronic condition affecting millions of people worldwide. With the goal of improving the accuracy and efficiency of diabetes prediction, our project focuses on leveraging AI technologies to develop a robust prediction model.

The project utilizes a diverse dataset consisting of various health parameters, demographic information, and historical medical records of individuals diagnosed with diabetes. By employing advanced machine learning algorithms and deep learning techniques, our AI model learns patterns and relationships from the data to create a predictive model.

The proposed AI-based diabetes prediction system aims to accurately identify individuals who may be at risk of developing diabetes in the future. The model will analyze the relevant features and provide a risk score or probability of diabetes onset. This information can potentially enable early intervention and personalized healthcare plans, allowing individuals to make informed lifestyle choices and take preventive measures.

In addition to accurate predictions, our project also focuses on interpretability and explainability of the AI model's decision-making. By incorporating explainable AI techniques, we aim to provide insights into the factors influencing the model's predictions.

GIVEN DATASET:

Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI
6	148	72	35	0	33.6
1	85	66	29	0	26.6
8	183	64	0	0	23.3
1	89	66	23	94	28.1
0	137	40	35	168	43.1
5	116	74	0	0	25.6
3	78	50	32	88	31
10	115	0	0	0	35.3
2	197	70	45	543	30.5
8	125	96	0	0	0
4	110	92	0	0	37.6
10	168	74	0	0	38
10	139	80	0	0	27.1
1	189	60	23	846	30.1
5	166	72	19	175	25.8
7	100	0	0	0	30
0	118	84	47	230	45.8
7	107	74	0	0	29.6
1	103	30	38	83	43.3
1	115	70	30	96	34.6
3	126	88	41	235	39.3
8	99	84	0	0	35.4
7	196	90	0	0	39.8
9	119	80	35	0	29
11	143	94	33	146	36.6
10	125	70	26	115	31.1
7	147	76	0	0	39.4
1	97	66	15	140	23.2
13	145	82	19	110	22.2
5	117	92	0	0	34.1
5	109	75	26	0	36
3	158	76	36	245	31.6
3	88	58	11	54	24.8
6	92	92	0	0	19.9
10	122	78	31	0	27.6
4	103	60	33	192	24
11	138	76	0	0	33.2
9	102	76	37	0	32.9
2	90	68	42	0	38.2
4	111	72	47	207	37.1
3	180	64	25	70	34
7	133	84	0	0	40.2
7	106	92	18	0	22.7
9	171	110	24	240	45.4
7	159	64	0	0	27.4

769 rows x 9 columns

OVERVIEW OF THE PROCESS:

Building an AI-based diabetes prediction system involves several key steps. Here's an overview of the process:

1. *Data Collection:*

- Gather a comprehensive dataset that includes relevant health and demographic information from individuals.

- Ensure the data is accurate, properly labeled, and covers a diverse population.

2. *Data Preprocessing:*

- Clean the data by addressing missing values, outliers, and inconsistencies.

- Normalize or standardize features to ensure they are on a similar scale.

- Handle class imbalance if present, as diabetes prediction datasets often have more non-diabetic cases.

3. *Feature Engineering:*

- Create or modify features to capture valuable information related to diabetes risk. This may include BMI, blood pressure categories, age groups, and more.

4. *Feature Selection:*

- Choose the most relevant features using various methods, such as filter, wrapper, or embedded techniques, as well as domain knowledge.

5. *Data Split:*

- Divide the dataset into training, validation, and test sets. Common splits are 70-80% for training, 10-15% for validation, and 10-15% for testing.

6. *Model Selection:*

- Choose a suitable machine learning algorithm for binary classification, such as logistic regression, decision trees, random forests, support vector machines, or deep learning models.

7. *Model Training:*

- Train the selected model on the training dataset, tuning hyperparameters to optimize performance.

- Implement k-fold cross-validation to assess generalization performance and prevent overfitting.

8. *Model Evaluation:*

- Use the test set to evaluate the model's predictive accuracy using metrics like accuracy, precision, recall, F1-score, and ROC-AUC.

9. *Model Deployment:*

- Deploy the trained model in a healthcare or clinical setting, ensuring compliance with medical and privacy regulations.

The process of building an AI-based diabetes prediction system is iterative and requires collaboration with healthcare professionals and domain experts to ensure its effectiveness and ethical use. Additionally, compliance with healthcare regulations, data privacy standards, and ethical guidelines is crucial throughout the development and deployment phases.

PROCEDURE:

Feature selection:

Feature selection for an AI-based diabetes prediction system involves choosing the most relevant variables or features from your dataset. Here are some steps specific to building a diabetes prediction system:

1. *Data Collection:* Gather a dataset that contains relevant information for diabetes prediction. This may include features like age, BMI, blood pressure, family history, glucose levels, and more.

2. *Data Preprocessing:* Clean the data by handling missing values, outliers, and any inconsistencies. Ensure that your data is properly labeled to distinguish between individuals with and without diabetes.

4. *Feature Selection Methods:* Choose appropriate feature selection methods for your diabetes prediction task:

a. *Filter Methods:* Use statistical measures like correlation coefficients or chi-squared tests to rank the importance of each feature in relation to the target variable (diabetes).

b. *Wrapper Methods:* Employ techniques like forward selection, backward elimination, or recursive feature elimination with a diabetes prediction model to determine the best subset of features.

c. *Embedded Methods:* Some machine learning algorithms for classification tasks, such as decision trees or random forests, can provide feature importance scores. You can use these scores to select relevant features.

d. *Domain Knowledge:* Leverage domain expertise to identify features known to be strongly associated with diabetes, like fasting blood sugar levels or insulin resistance.

6. *Model Selection:* Choose a machine learning algorithm suitable for classification tasks, such as logistic regression, support vector machines, or decision trees.

7. *Model Building:* Train and fine-tune your selected model using the subset of features you've chosen.

9. *Deployment:* Deploy the diabetes prediction system in a healthcare or clinical setting, ensuring compliance with all relevant regulations and data privacy standards.

Feature selection is critical to improve model accuracy and reduce overfitting, especially in cases like diabetes prediction, where not all available features may be equally informative. Remember that the choice of features and the quality of data play a significant role in the effectiveness of your AI-based diabetes prediction system.

Feature selection:

Pearson's correlation method is a popular method to find the most relevant attributes/features. The correlation coefficient is calculated in this method, which correlates with the output and input attributes. The coefficient value remains in the range between −1 and 1. The value above 0.5 and below −0.5 indicates a notable correlation, and the zero value means no correlation. In Weka, the correlation filter is used to find the correlation coefficient, and the results are shown in Table 3. We used 0.2 as a cut-off for relevant attributes. Hence SkinThickness, BP, DPF features are removed. Glucose, BMI, Insulin, Preg, and Age are our most relevant five input attributes.

The correlation between input and output attributes.

Attributes	**Correlation coefficient**
Glucose	0.484

Attributes	Correlation coefficient
BMI	0.316
Insulin	0.261
Preg	0.226
Age	0.224
SkinThickness	0.193
BP	0.183
DPF	0.178

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn import svm
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import ConfusionMatrixDisplay

RED = "\033[91m"
GREEN = "\033[92m"
YELLOW = "\033[93m"
BLUE = "\033[94m"
RESET = "\033[0m"

df = pd.read_csv("/kaggle/input/diabetes-data-set/diabetes.csv")
```

DATA CLEANING

```
  print(BLUE + "\nDATA CLEANING" + RESET)
```

Check for missing values

```
missing_values = df.isnull().sum()
print(GREEN + "Missing Values : " + RESET)
print(missing_values)
```

Handle missing values

mean_fill = df.fillna(df.mean())

df.fillna(mean_fill, inplace=True)

Check for duplicate values

```
duplicate_values = df.duplicated().sum()
print(GREEN + "Duplicate Values : " + RESET)
print(duplicate_values)
```

Drop duplicate values

```
df.drop_duplicates(inplace=True)
```

Support Vector Machine Modelling

```
print(BLUE + "\nMODELLING" + RESET)
X = df.drop("Outcome", axis=1)
y = df["Outcome"]
# --- Splitting the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
)
# --- Splitting the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
)
# --- Standardize Features
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
# --- init and train SVM model
model = svm.SVC(kernel="linear")
model.fit(X_train, y_train)
# --- Predict on test data
y_pred = model.predict(X_test)
# --- Evaluate model performance
accuracy = model.score(X_test, y_test)
print(GREEN + "Model Accuracy : " + RESET)
print(accuracy)
# --- Classification Report and Confusion Matrix
print(GREEN + "Classification Report : " + RESET)
print(classification_report(y_test, y_pred))
print(GREEN + "Confusion Matrix : " + RESET)
cm = ConfusionMatrixDisplay.from_predictions(y_test, y_pred)
sns.heatmap(cm.confusion_matrix, annot=True, cmap="Blues")
plt.show()
print("Displayed")

# SAVING THE FILE
df.to_csv("/kaggle/working/cleaned_diabetes.csv", index=False)
print(BLUE + "\nDATA SAVING" + RESET)
print(GREEN + "Data Cleaned and Saved !" + RESET)
print("\n")
```

DATA CLEANING

Missing Values :

```
Pregnancies                 0
Glucose                     0
BloodPressure               0
SkinThickness               0
Insulin                     0
BMI                         0
DiabetesPedigreeFunction    0
Age                         0
Outcome                     0
dtype: int64
```

Duplicate Values :

0

DATA ANALYSIS

Summary Statistics :

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin \
count	768.000000	768.000000	768.000000	768.000000	768.000000
mean	3.845052	120.894531	69.105469	20.536458	79.799479
std	3.369578	31.972618	19.355807	15.952218	115.244002
min	0.000000	0.000000	0.000000	0.000000	0.000000
25%	1.000000	99.000000	62.000000	0.000000	0.000000
50%	3.000000	117.000000	72.000000	23.000000	30.500000
75%	6.000000	140.250000	80.000000	32.000000	127.250000
max	17.000000	199.000000	122.000000	99.000000	846.000000

	BMI	DiabetesPedigreeFunction	Age	Outcome
count	768.000000	768.000000	768.000000	768.000000
mean	31.992578	0.471876	33.240885	0.348958
std	7.884160	0.331329	11.760232	0.476951
min	0.000000	0.078000	21.000000	0.000000
25%	27.300000	0.243750	24.000000	0.000000
50%	32.000000	0.372500	29.000000	0.000000
75%	36.600000	0.626250	41.000000	1.000000
max	67.100000	2.420000	81.000000	1.000000

Class Distribution :

```
Outcome
0    500
1    268
Name: count, dtype: int64
```

MODELLING

Model Accuracy :

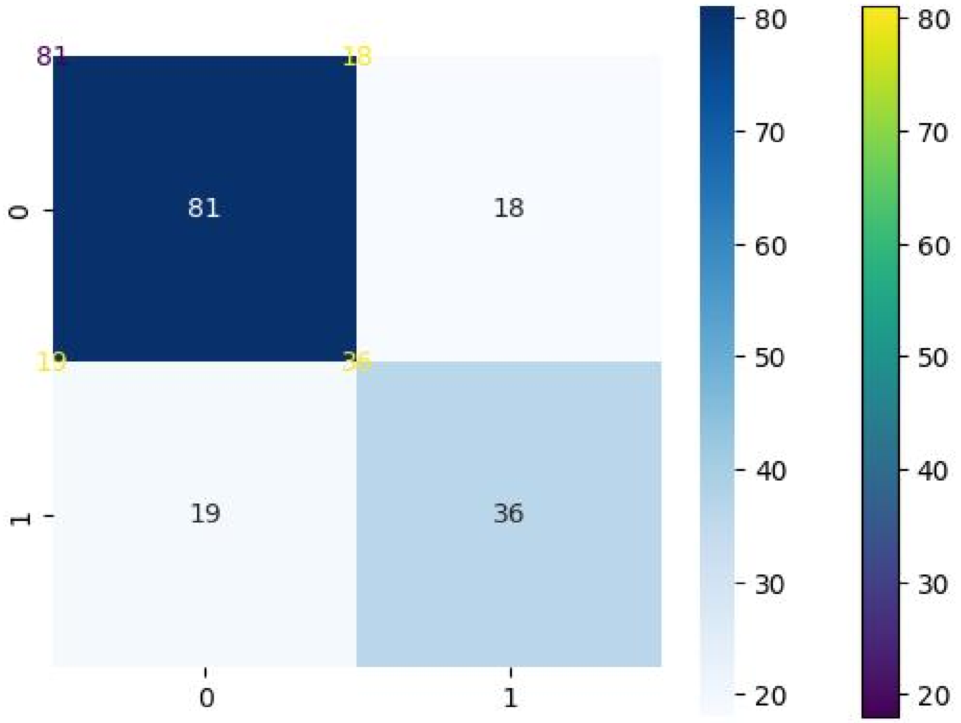
0.7597402597402597

Classification Report :

	precision	recall	f1-score	support
0	0.81	0.82	0.81	99

1	0.67	0.65	0.66	55
accuracy			0.76	154
macro avg	0.74	0.74	0.74	154
weighted avg	0.76	0.76	0.76	154

Confusion Matrix :



Model Training:

Training an AI-based diabetes prediction system typically involves the following steps:

1. Data Collection: Gather a large and diverse dataset that includes relevant features such as patient demographics, medical history, lifestyle factors, and lab test results. The dataset should also include information about whether each individual has diabetes or not.

2. Data Preprocessing: Clean the data by handling missing values, outliers, and formatting issues. Normalize or standardize the data to ensure that different features have the same scale.

3. Feature Selection: Choose the most relevant features that can contribute to diabetes prediction. Feature engineering may also be performed to create new features based on domain knowledge.

4. Model Selection: Choose an appropriate machine learning or deep learning model for the prediction task. Common choices include logistic regression, decision trees, support vector machines, or neural networks.

5. Model Training: Split the dataset into training and testing sets to evaluate model performance. Train the chosen model on the training data, adjusting hyperparameters and optimizing it for the task.

6. Evaluation: Use evaluation metrics such as accuracy, precision, recall, F1 score, and area under the ROC curve to assess the model's performance on the test data. Cross-validation can also be employed for a more robust evaluation.

7. Fine-tuning: Based on the evaluation results, fine-tune the model by adjusting hyperparameters, trying different algorithms, or increasing the dataset size if necessary.

8. Deployment: Once the model performs well, deploy it in a clinical setting or as part of a diabetes management system. This may involve integrating the model into a mobile app or a web platform.

9. Monitoring and Maintenance: Continuously monitor the model's performance and update it as needed to adapt to changes in data distribution or medical knowledge.

10. Privacy and Ethics: Be mindful of patient data privacy and ethical considerations, such as informed consent and data security.

It's important to collaborate with healthcare professionals to ensure that the AI system aligns with medical standards and can provide valuable insights for diabetes management and prediction.

Model evaluation:

Evaluating an AI-based diabetes prediction system is crucial to ensure its effectiveness and reliability. Here are some common evaluation metrics and methods:

1. *Accuracy*: Calculate the percentage of correctly predicted cases. However, accuracy can be misleading if the dataset is imbalanced.

2. *Precision and Recall*: These metrics are useful when dealing with imbalanced datasets. Precision measures the ratio of true positive predictions to all positive predictions, while recall measures the ratio of true positive predictions to all actual positive cases.

3. *F1 Score*: It combines precision and recall into a single metric, providing a balance between the two. It's especially useful when you need to strike a balance between false positives and false negatives.

4. *Receiver Operating Characteristic (ROC) Curve*: Plot the true positive rate against the false positive rate to assess the model's ability to distinguish between classes. The area under the ROC curve (AUC) is also a valuable metric.

5. *Confusion Matrix*: This matrix provides a detailed breakdown of true positives, true negatives, false positives, and false negatives.

6. *Cross-Validation*: Use techniques like k-fold cross-validation to assess the model's performance on different subsets of the data.

7. *Feature Importance Analysis*: Determine which features have the most significant impact on predictions. This can help in feature selection and model improvement.

8. *Domain Expert Review*: Have domain experts evaluate the model's predictions and provide feedback on its clinical relevance and accuracy.

9. *External Validation*: Test the model on new, unseen data to assess its generalizability.

10. *Ethical and Fairness Evaluation*: Check for biases in the model's predictions and ensure it doesn't disproportionately affect certain demographic groups.

11. *Continuous Monitoring*: After deployment, continuously monitor the model's performance and update it as needed.

Remember that the choice of evaluation metrics and methods should align with the specific goals and requirements of the diabetes prediction system and its intended use in a clinical setting. Additionally, it's important to involve healthcare professionals in the evaluation process to ensure the model meets clinical standards and is safe for patients.

Feature Engineering:

Feature engineering is the process of creating new features or modifying existing ones to improve the performance of an AI-based diabetes prediction system. Here are some feature engineering ideas for such a system:

1. *Body Mass Index (BMI):* Calculate BMI from height and weight features if not already present in your dataset. High BMI is often associated with diabetes risk.

2. *Waist-to-Hip Ratio:* Create a feature that represents the waist-to-hip ratio, which can be an indicator of abdominal obesity and diabetes risk.

3. *Age Groups:* Group individuals into different age categories (e.g., young adults, middle-aged, seniors) to capture age-related diabetes trends.

4. *Blood Pressure Categories:* Categorize blood pressure readings as normal, prehypertension, or hypertension to incorporate the impact of hypertension on diabetes risk.

5. *Glucose Metrics:* Create aggregated metrics for glucose levels, such as the average glucose level over a specific time period or the variation in glucose levels.

6. *Family History Score:* Assign scores based on family history of diabetes,where a higher score indicates a stronger family history of the disease.

7. *Dietary Patterns:* If dietary data is available, engineer features to capture dietary patterns like low-carb, high-fiber, or high-sugar diets.

8. *Physical Activity Levels:* Create a metric that quantifies the level of physical activity for each individual, considering factors like daily steps, exercise frequency, and sedentary behavior.

9. *Sleep Patterns:* Incorporate features related to sleep duration and quality, as poor sleep habits can impact diabetes risk.

10. *Medication History:* If relevant, include features that track an individual's history of diabetes medications or insulin use.

11. *HbA1c Trends:* Calculate trends in HbA1c values over time if your dataset includes multiple measurements for each individual.

12. *Fasting Hours:* Create a feature representing the number of fasting hours before a glucose measurement, which can impact glucose levels.

13. *Microbiome Data:* If available, incorporate features related to gut microbiome composition, as emerging research suggests a link between the microbiome and diabetes.

14. *Social and Economic Factors:* Consider features related to an individual's socioeconomic status, access to healthcare, and lifestyle, as these can influence diabetes risk.

15. *Geographic Factors:* If data includes location information, features related to geographic factors like climate, pollution, or access to healthy food options can be relevant.

16. *Interaction Terms:* Create interaction terms between pairs of features to capture potential synergistic effects, e.g., age multiplied by BMI.

17. *Time-based Features:* Incorporate time-related features like the season, day of the week, or holidays, as they might influence eating and exercise patterns.

18. *Composite Scores:* Develop composite scores that combine multiple features to represent an individual's overall risk, such as a diabetes risk score.

Remember to validate the new features through cross-validation and assess their impact on the performance of your diabetes prediction model. Feature engineering is an iterative process, and domain knowledge plays a crucial role in deciding which features are most relevant to the problem.

Conclusion:

In conclusion, the development of an AI-based diabetes predicton system holds signifcant potental for improving healthcare outcomes related to diabetes. By leveraging artfcial intelligence techniques, such as machine learning and data analysis, the system can provide accurate risk assessments and enable early interventon and preventon strategies.

The AI-based diabetes predicton system ofers several advantages. Firstly, it can assist healthcare professionals in identfying individuals at high risk of developing diabetes, allowing for targeted interventons and personalized care plans. Secondly, it can contribute to reducing healthcare costs by focusing resources on preventon and early detecton, which can help mitgate the long-term complicatons associated with diabetes. Additonally, the system can empower individuals to take proactve steps towards managing their health and making informed lifestyle choices