AI BASED DIABETES PREDICTION SYSTEM

952721104304 – Selva Esacky.V.K (aut104301)

ABSTRACT

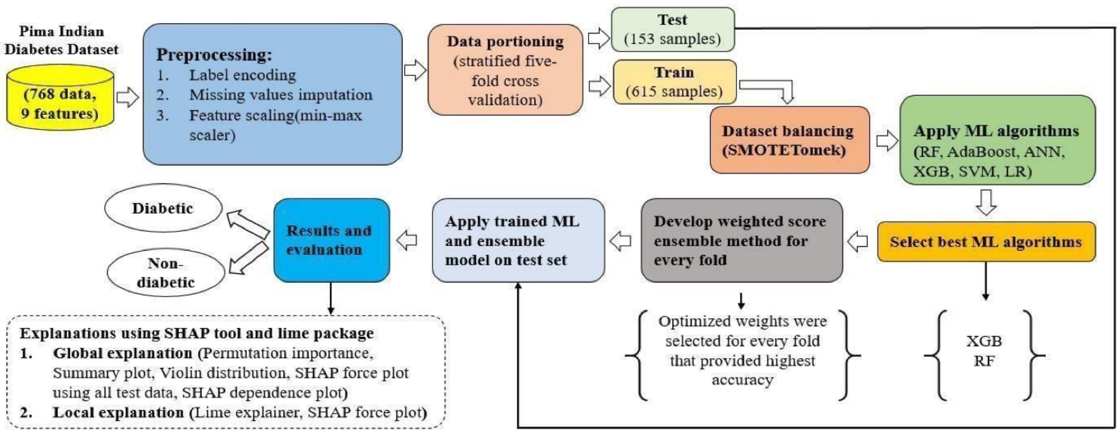
Artificial intelligence (AI) has the potential to revolutionize the way we diagnose and manage diabetes. AI-based diabetes prediction systems can be used to identify people at high risk of developing diabetes, allowing for early intervention and prevention. These systems can also be used to personalize diabetes management, helping people to better control their blood sugar levels and reduce their risk of complications.

MODULES

An AI-based diabetes prediction system typically consists of the following modules:

- Data collection and pre-processing: This module collects and prepares data from a variety of sources, such as electronic health records, wearable devices, and patient surveys. The data is then cleaned and pre-processed to ensure that it is consistent and suitable for machine learning.

- Feature engineering: This module extracts features from the pre-processed data that are relevant to diabetes prediction. These features may include demographic data, medical history, lifestyle factors, and laboratory results.
- Model training: This module trains a machine learning model to predict the risk of diabetes. The model is trained on a dataset of known diabetes cases and controls.
- Model deployment: This module deploys the trained model to production, so that it can be used to predict the risk of diabetes in new patients.
- User interface: This module provides a user interface for interacting with the system. This may include a web application, mobile app, or integration with an electronic health record system.



EXAMPLE WORKFLOW

The following is an example of how an AI-based diabetes prediction system might be used:

1. A patient visits their doctor for a routine check up. The doctor collects some basic demographic data and medical history from the patient.

2. The doctor also orders some laboratory tests, such as a blood glucose test.

3. The results of the laboratory tests and the other collected data are then entered into the AI based diabetes prediction system.

4. The system uses the data to predict the patient's risk of developing diabetes.
5. The doctor reviews the system's prediction and discusses it with the patient.
6. If the patient is at high risk of developing diabetes, the doctor can develop a plan to help the patient prevent or manage the condition.

BENEFITS

AI-BASED DIABETES PREDICTION SYSTEMS OFFER A NUMBER OF BENEFITS, INCLUDING:

- Early detection: AI-based systems can identify people at high risk of developing diabetes earlier than traditional methods. This allows for early intervention and prevention, which can improve outcomes.
- Personalized care: AI-based systems can be used to personalize diabetes management for each patient. This can help people to better control their blood sugar levels and reduce their risk of complications.
- Improved efficiency: AI-based systems can help to improve the efficiency of diabetes care. For example, AI-based systems can be used to automate tasks such as screening patients for diabetes and developing personalized care plans.

CONCLUSION

AI-based diabetes prediction systems have the potential to significantly improve the way we diagnose and manage diabetes. These systems can help to identify people at high risk of developing diabetes earlier, personalize diabetes management, and improve the efficiency of diabetes care.

1. PROBLEM DEFINITION:

Clearly define the problem you want to address, such as predicting blood glucose levels based on various features like age, BMI, diet, and physical activity.

2. DATA COLLECTION:

Gather a dataset that contains relevant features (independent variables) and the target variable (blood glucose levels). Data can be collected from sources like medical records, health sensors, or surveys.

SAMPLE DATA SET

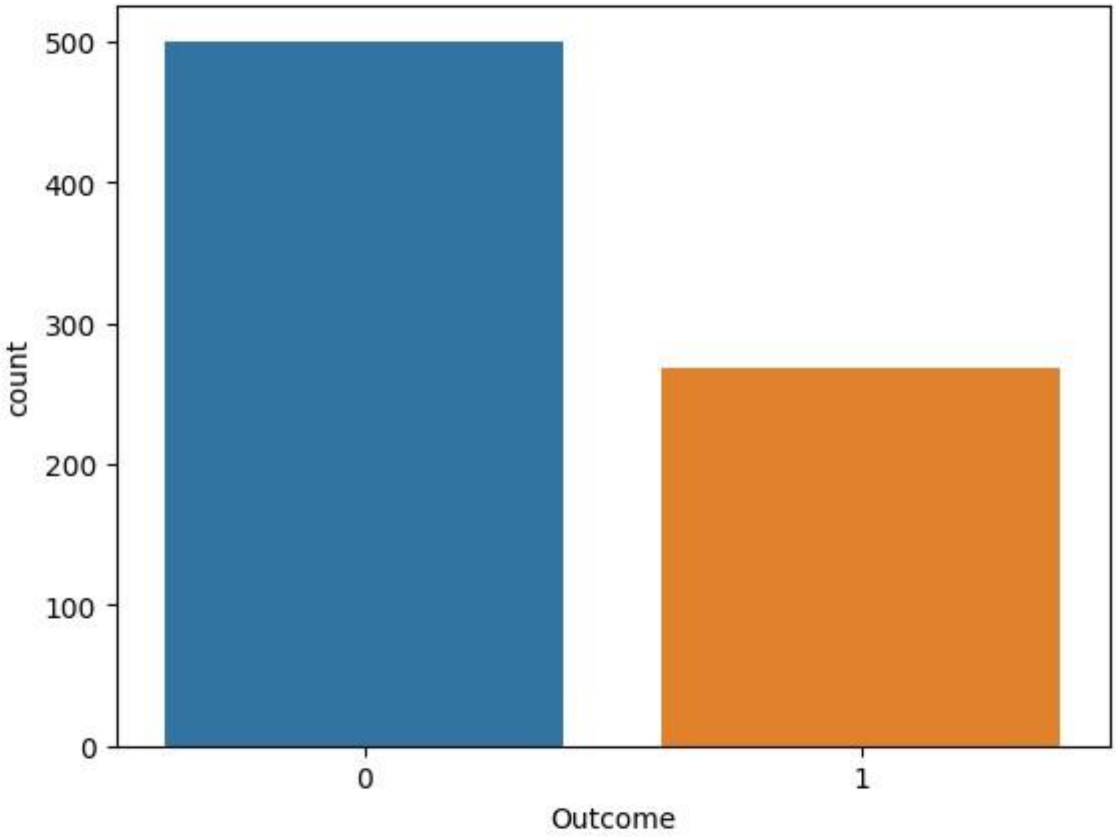
	Pregnancies	Glucose	Blood Pressure	Skin Thickness	Insulin	BMI	Diabetes Pedigree Function	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1

3. DATA PRE-PROCESSING:

Clean and pre-process the dataset to make it suitable for regression modelling. This may involve handling missing values, outlier detection and treatment, and feature scaling.

PROGRAM:

```
#check if null value is
present
dataset.isnull().values.any()
dataset.info()
dataset.describe()
dataset.isnull().sum() #data
visualization
sns.countplot(x = 'Outcome',data = dataset)
```



4. FEATURE SELECTION AND ENGINEERING:

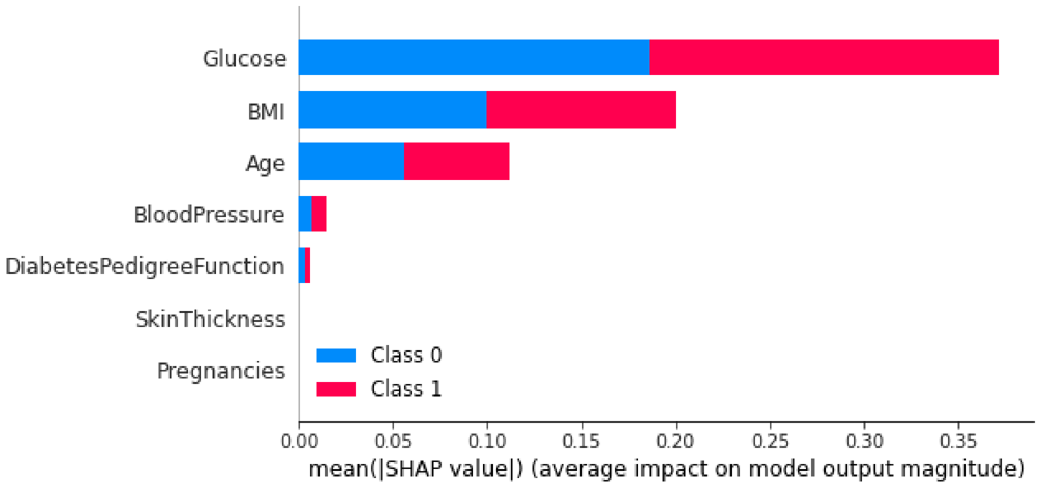
Select and engineer features that are relevant to the prediction task. You might also create new features, like interactions between variables or aggregations.

5. DATA SPLIT:

Divide the dataset into training, validation, and test sets. Common splits are 70-80% for training, 1015% for validation, and 10-15% for testing.

6. SELECT REGRESSION MODELS:

Choose one or more regression models suitable for the task. Common regression techniques include Linear Regression, Lasso Regression, Ridge Regression, Decision Trees, Random Forests, and Support Vector Regression.



7. MODEL EVALUATION:

Assess the model's performance using appropriate regression metrics such as Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and R-squared (R^2) on the validation dataset.

PROGRAM: #Heatmap

```
sns.heatmap(dataset.corr(), annot = True)

plt.show() # Count of NaN

dataset_new.isnull().sum()

dataset_new.isnull().sum()

dtype: int64
```

Pregnancies	0
Glucose	0
Blood Pressure	0

SkinThickness	0
Insulin	0
BMI	0
DiabetesPedigreeFunction	0
Age	0
Outcome	0

8. MODEL TRAINING:

Train the chosen regression model(s) on the training data. The model will learn the relationships between the features and the target variable (blood glucose levels).

PROGRAM:

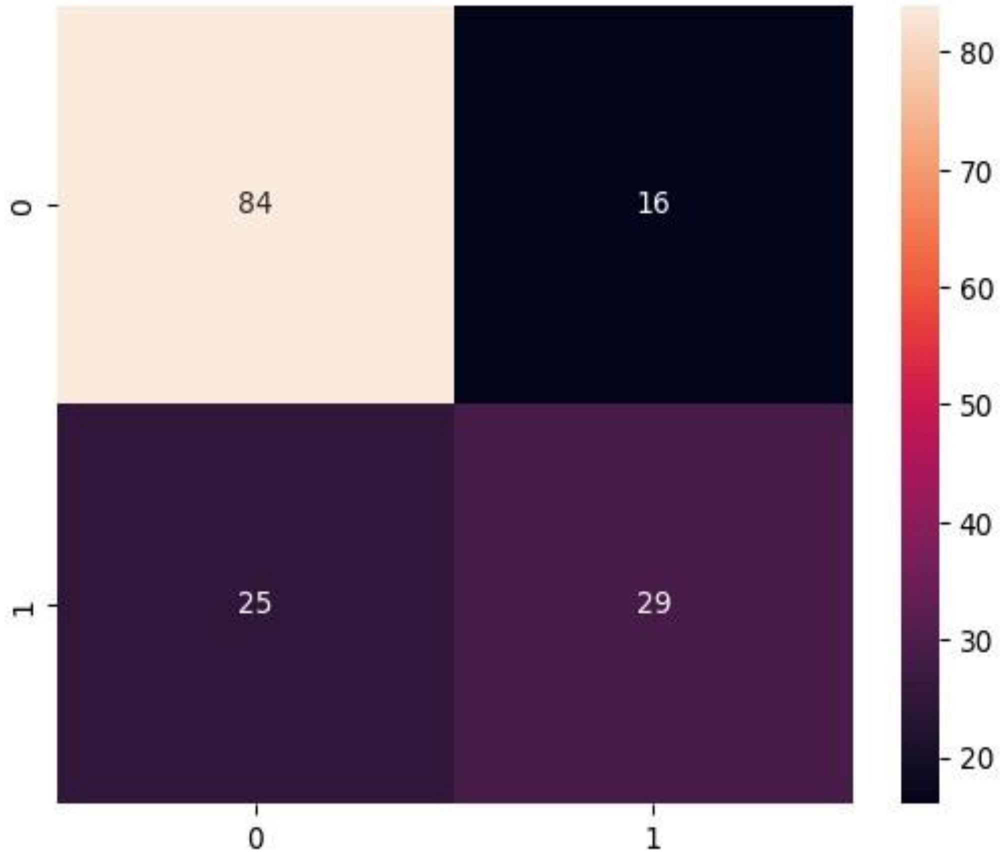
```
# Splitting X and Y from sklearn.model_selection

import train_test_split

X_train, X_test, Y_train, Y_test = train_test_split(X, y, test_size = 0.20, random_state =
42, stratify = dataset_new['Outcome'] ) from

sklearn.linear_model import LogisticRegression

model = LogisticRegression() model.fit(X_train,

Y_train) y_predict = model.predict(X_test)
```

9. HYPER PARAMETER TUNING:

Fine-tune the model's hyper parameters to optimize its performance using techniques like grid search, random search, or Bayesian optimization.

10. MODEL TESTING:

Evaluate the final model on the test dataset to ensure it generalizes well to new, unseen data.



11. INTERPRETABILITY:

Ensure that the model's predictions are interpretable. This can be crucial in a healthcare context to understand the factors contributing to blood glucose predictions.

12. DEPLOYMENT:

Deploy the regression model in a production environment. This could involve creating an API for real-time predictions or incorporating the model into a healthcare system.

13. MONITORING AND MAINTENANCE:

Continuously monitor the model's performance in the real world, and retrain it periodically with new data to maintain its accuracy.