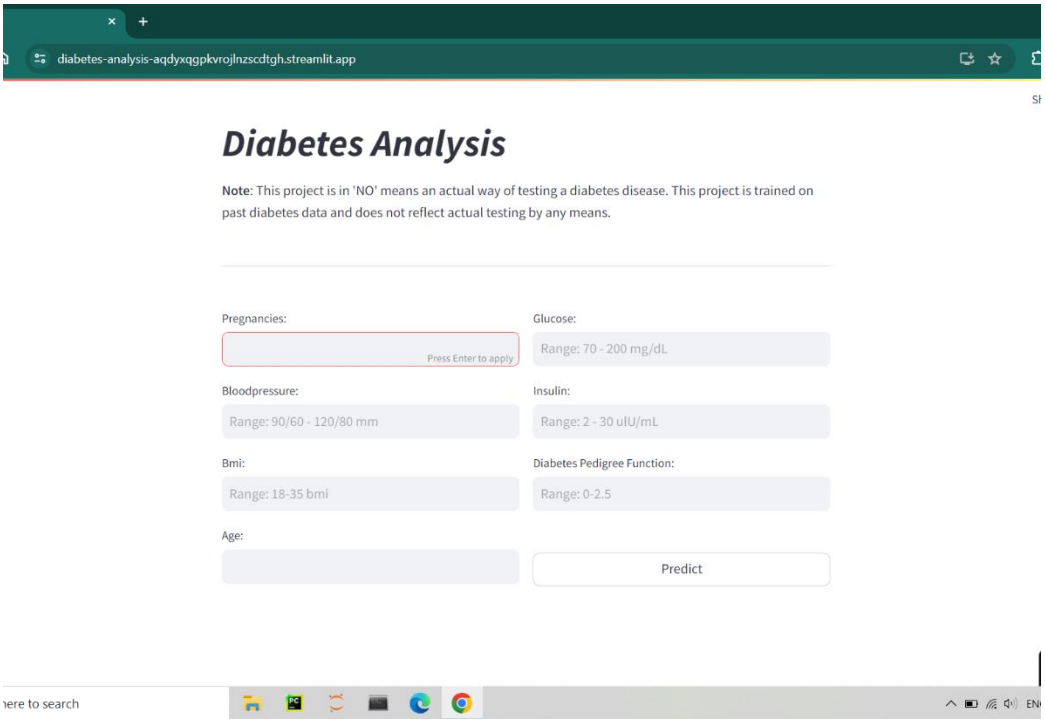
DIABETES

Analysis

System



Date: 04/10/2023

By: Yash Keshari

Document Version Control

Date Issued	Version	Description	Author
03/10/2023	1	Documentation- V 0.1	Yash Keshari

Project link:
https://diabetes-analysis-aqdyxqgpkvrojlnzscdtgh.streamlit.app/

Github link:
https://github.com/yash1314/learning_buds_intern_project

Content :

1 Introduction

Diabetes is a chronic metabolic disorder that affects millions of individuals worldwide. It is characterized by elevated levels of blood glucose, which can lead to various health complications. Understanding and analyzing diabetes data is of paramount importance for both medical professionals and researchers. Through data analysis techniques and machine learning, we can gain insights into factors influencing diabetes risk, identify early markers, and develop predictive models for disease management. This analysis aims to contribute to the same.

2. Dataset Description

The diabetes dataset contains health-related information for individuals, with a focus on Pima Indian women over the age of 21. It is widely used in medical research and machine learning for developing predictive models for diabetes diagnosis.

3. Columns in the Dataset

Pregnancies: Represents the number of pregnancies, providing insight into potential gestational diabetes risk.

Glucose: Indicates blood sugar levels, a crucial diagnostic factor for diabetes.

Blood Pressure: Measures blood pressure, which is a risk factor for diabetes.

Skin Thickness: Represents skinfold thickness, used to estimate body fat percentage.

Insulin: Indicates insulin levels, a hormone critical for regulating blood sugar.

BMI (Body Mass Index): It is a measure of weight relative to height, highlighting obesity risk.

Diabetes Pedigree Function: Quantifies genetic risk to diabetes based on family history.

Age: Represents the individual's age, a key factor in diabetes risk, particularly for type 2 diabetes.

Outcome: The target variable, typically binary (1 for diabetes, 0 for no diabetes), used for predictions and analysis.

4. Data Transformation

These transformations are applied to handle outliers or missing values in the dataset and make the data more suitable for analysis and modeling. The specific values used for the lower bounds and median may vary based on the dataset and analysis requirements. It's a common practice to replace extreme values or missing data with more reasonable values to ensure data quality for subsequent analysis.

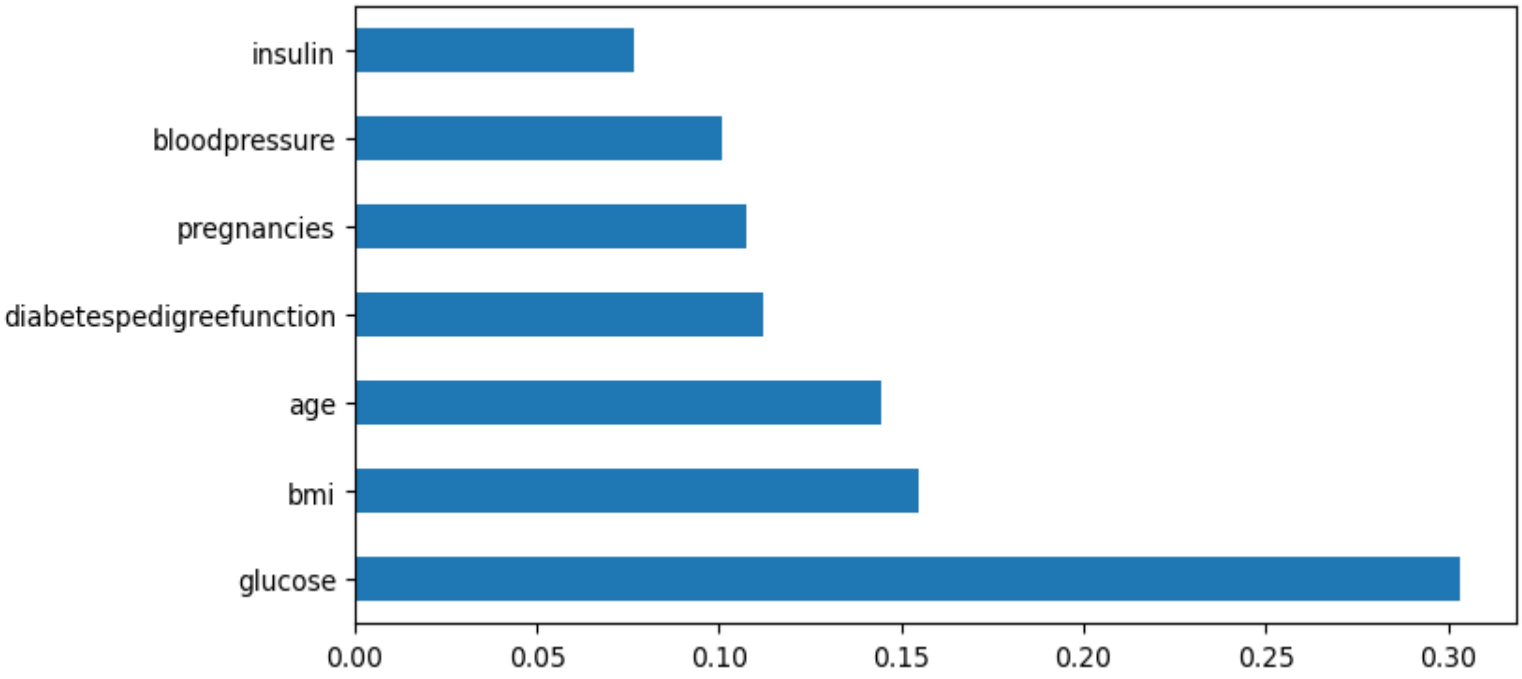
Glucose Transformation:
Calculate a lower bound for the "glucose" column as 2.5 times the standard deviation below the mean.
Replace values in the "glucose" column that are less than the calculated lower bound with the lower bound value.

Blood Pressure Transformation:
Calculate a lower bound for the "bloodpressure" column as 2 times the standard deviation below the mean.
Replace values in the "bloodpressure" column that are equal to 0 with the calculated lower bound value.

Skin Thickness Transformation:
Calculate the median value of the "skinthickness" column.
Replace values in the "skinthickness" column that are equal to 0 with the calculated median value.

BMI Transformation:
Calculate a lower bound for the "bmi" column as 1.5 times the standard deviation below the mean.
Replace values in the "bmi" column that are equal to 0 with the calculated lower bound value.

5. Feature Importance.



6. Tools used

Tools and Technologies Used -

- **Programming Language**: Python

- **Python Libraries and Frameworks**:
  - NumPy
  - Pandas
  - Scikit-learn
  - MLOps basics

- **Integrated Development Environment (IDE)**:
  - Vs-code

- **Data Visualization**:
  - Seaborn

- **Cloud Computing and Deployment**:
  - Streamlit cloud

- **Front-end Development**:
  - Streamlit.

GitHub is used as version control

7. Evaluation results and analysis

Model: Gradient Boosting Algorithm(defaults)

Performance Metrics:

Recall Score:

Recall is a measure of the model's ability to correctly identify positive instances (True Positives) out of all actual positive instances (True Positives + False Negatives). A recall score of 0.75 (75%) indicates that the model correctly identified 75% of the actual positive cases.

Confusion Matrix: [92 14]
[14 42]
True Positives (TP): 42 cases were correctly identified as positive.
True Negatives (TN): 92 cases were correctly identified as negative.
False Positives (FP): 14 cases were incorrectly identified as positive.
False Negatives (FN): 14 cases were incorrectly identified as negative.

Analysis:

1. The model has a relatively high recall score of 0.75, which means that it is effective at capturing a significant portion of actual positive cases.

2. The number of false negatives (14) indicates that there are cases of actual positive instances that the model missed. This means there is room for improvement in correctly identifying more positive cases.

3. The number of false positives (14) suggests that some negative cases were incorrectly classified as positive. Reducing false positives may be desirable in certain applications, depending on the consequences of false alarms.

4. The number of true negatives (92) indicates that a substantial number of negative cases were correctly identified as negative.

Overall Evaluation:

The model exhibits a relatively good ability to identify actual positive cases (75% recall) but has room for improvement in reducing false negatives and, if necessary, managing false positives. Further analysis and potential model refinement may be required to optimize the performance based on the desired outcomes.

8. Project Folder Architecture

```
|- artifacts/ (Include trained models or other important artifacts)
|
|- notebooks/
| |- notebook.ipynb (Jupyter Notebook for the project)
| |- data/ (Data files or datasets)
|
|- logs/ (Log files or logs directory)
|
|- src/
| |- logger.py (Logging utilities)
| |- exception.py (Custom exception handling)
| |- utils.py (Utility functions)
| |- __init__.py
|
| |- components/
|       -__init__.py
|       - data_ingestion.py (Data loading functions)
|       - data_transformation.py (Data preprocessing, feature engineering)
|       - model_training.py (Machine learning model training)
|
| |- pipelines/
|       - __init__.py
|       - prediction_pipeline.py (prediction pipeline)
|       - training_pipeline.py (Model training pipeline)
```

|- requirements.txt (List of project dependencies)
|-visuals/
| - image.jpeg
|- app.py (Main application script)
|
|- .gitignore (Specify files or directories to ignore in version control)

9. References

- Google
- Documentations – Scikit learn, pandas, streamlit, ect.