Diabetes Prediction Project Documentation

Overview

This project predicts the likelihood of diabetes based on user input using a logistic regression model trained on a real-world dataset. The application is built with Flask and provides a web interface for predictions.

Project Structure

```
.
├── app.py: Flask web application for user interaction and prediction.
├── dpmodel.py: Model training and prediction logic.
├── dpm.pickle: Serialized trained logistic regression model.
├── diabetes_prediction_dataset.csv: Dataset used for training.
├── static/
│   └── main.css: CSS for web interface styling.
└── templates/
    └── index.html: HTML template for the web interface..
```

Dataset: diabetes_prediction_dataset.csv

Columns:

- gender: Female, Male, Other, or No Info (mapped to 0, 1, 2)
- age: Age in years (float)
- hypertension: 1 if present, 0 otherwise
- heart_disease: 1 if present, 0 otherwise
- smoking_history: Categorical (not used in model)
- bmi: Body Mass Index (float)
- HbA1c_level: Not used in model
- blood_glucose_level: Blood glucose level (float)
- diabetes: 1 if diabetic, 0 otherwise (target)

Link: Kaggle

Model Features/Inputs:

Only the following columns are used as features:

- Gender
- Age
- Hypertension
- Heart_disease
- Bmi
- Blood_glucose_level

Model Training: dpmodel.py

Libraries:

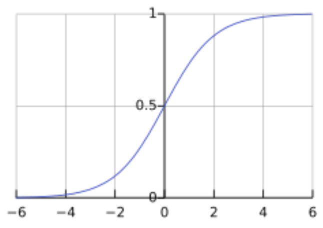
- Pandas
- Numpy
- Scikit-learn
- Pickle

The model is a Logistic Regression Model, which is used to predict the probability of a binary outcome. In the case of this project, we used it to predict the chances of a given person getting diabetes. The probability is predicted by putting the inputs in an equation that would look something like this:

$$z = b + w_1 * x_1 + w_2 * x_2 + \ldots + w_n * x_n$$

b represents bias, which is created when training. Each x value, which are also the inputs, has a weight that is multiplied by the input. This happens for each input, and then they are all added up along with the bias, creating the output. However, this doesn't yet create the probability. Then, you put the output through a sigmoid function, which looks like:

$$f(x) = \frac{1}{1 + \mathrm{e}^{-(x)}}$$



This function always returns a value between 0 and 1, which is ideal for probability.

Steps To Train Model:

1. Load the dataset and drop missing values.
2. Map gender to numeric values: Female=0, Male=1, Other=2.
3. Select features: gender, age, hypertension, heart_disease, bmi, blood_glucose_level.
4. Split data into training and test sets.
5. Train a Logistic Regression model on all the data.
6. Save the trained model to dpm.pickle.
7. Apply a sigmoid function through any result produced by the model

Web Application: app.py

Libraries:

- Flask
- Pickle
- Numpy

Endpoints:

/: Home page with input form.

/prediction: Handles form submission and displays prediction.

Flow:

1. The user submits a form with required fields.
2. Input is parsed and converted to the expected feature vector.
3. The model is loaded from dpm.pickle.
4. The Lrprob function is called to compute the probability. (This function applies both formulas used in a Logistic Regression Model)
5. The result is displayed on the web page.

Web Interface

HTML Template: index.html

- Collects user input for all required features.
- Displays the prediction result.

Styling: main.css

- Provides a clean, modern look using the Poppins font and styled form elements.

Make a Prediction:

Fill out the form and submit to see your diabetes risk probability.