Project Report

Name: Pima Indians Women Diabetes Prediction Flask Application
Report date: 11-Aug-2021
Internship Batch: LISUM02
Project by: Aynur Cemre Aka
Project Report reviewer:
Dataset location: https://www.kaggle.com/uciml/pima-indians-diabetes-database
Project location: https://github.com/cemreaka/Diabetes-Prediction-Flask-App

Overview

This project aims to predict Pima Indians women diabetes by using a machine learning model. The application diagnostically predict whether a patient has high risk of diabetes, based on certain diagnostic measurements included in the dataset. The prediction is shown after the several medical variables are entered.

This project report consists of 4 parts:

1) Dataset
2) Creating a model
3) Creating the Flask Web Application
4) Styling

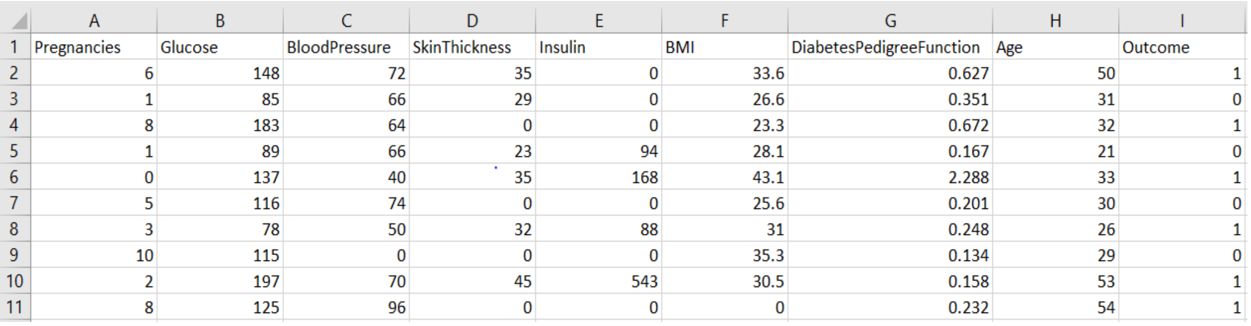
Step 1: Dataset

diabetes.csv consists of 8 medical predictor (independent) variables:

1) Pregnancies: number of times pregnant
2) Glucose: plasma glucose concentration 2 hours in an oral glucose tolerance test
3) BloodPressure: diastolic blood pressure (mm Hg)
4) SkinThickness: triceps skin fold thickness (mm)
5) Insulin: 2-hour serum insulin (mu U/ml)
6) BMI: body mass index (weight in kg/(height in m)^2)
7) DiabetesPedigreeFunction: diabetes pedigree function
8) Age

The target (dependent) variable is Outcome (0 or 1). Totally, this dataset consists of 9 variables. The aim of this dataset is to predict whether a patient has diabetes or not.

First 10 rows of the dataset:

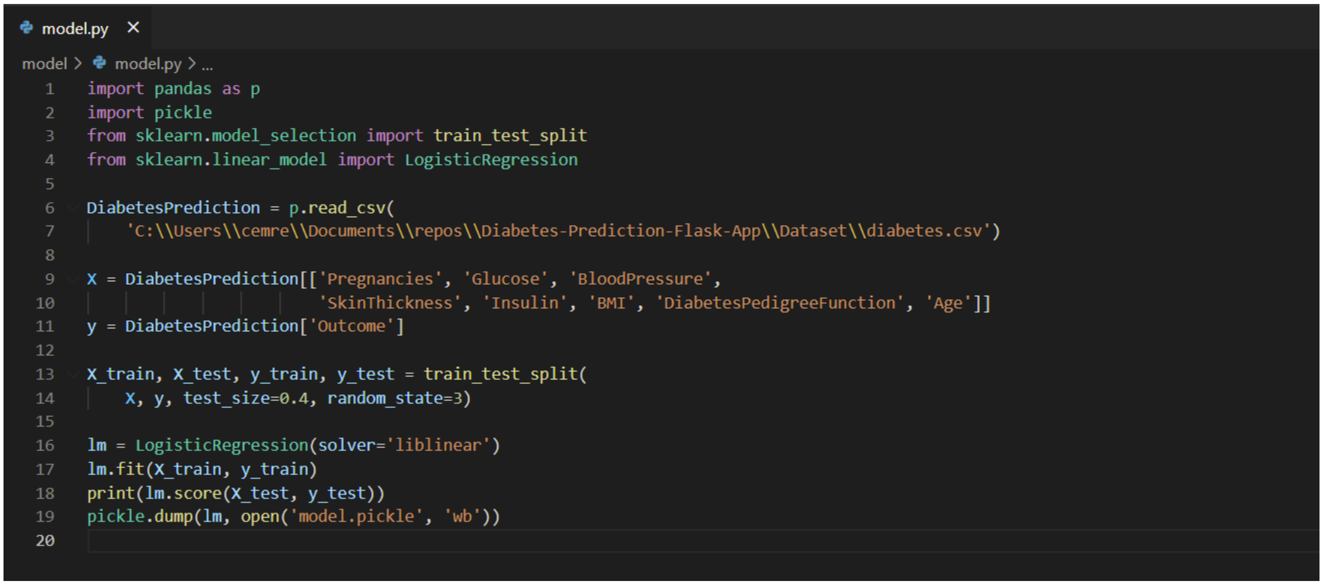


	A	B	C	D	E	F	G	H	I
1	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
2	6	148	72	35	0	33.6	0.627	50	1
3	1	85	66	29	0	26.6	0.351	31	0
4	8	183	64	0	0	23.3	0.672	32	1
5	1	89	66	23	94	28.1	0.167	21	0
6	0	137	40	35	168	43.1	2.288	33	1
7	5	116	74	0	0	25.6	0.201	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

Step 2: Creating a Machine Learning Model

After the dataset has been analyzed, it is divided into 2 parts, which are train (60%) and test (40%) sets. Since the output is binary (0 or 1), the model is created by using logistic regression. The accuracy is calculated as 0.75. After that, the model is serialized and saved as model.pickle file.

model.py file:

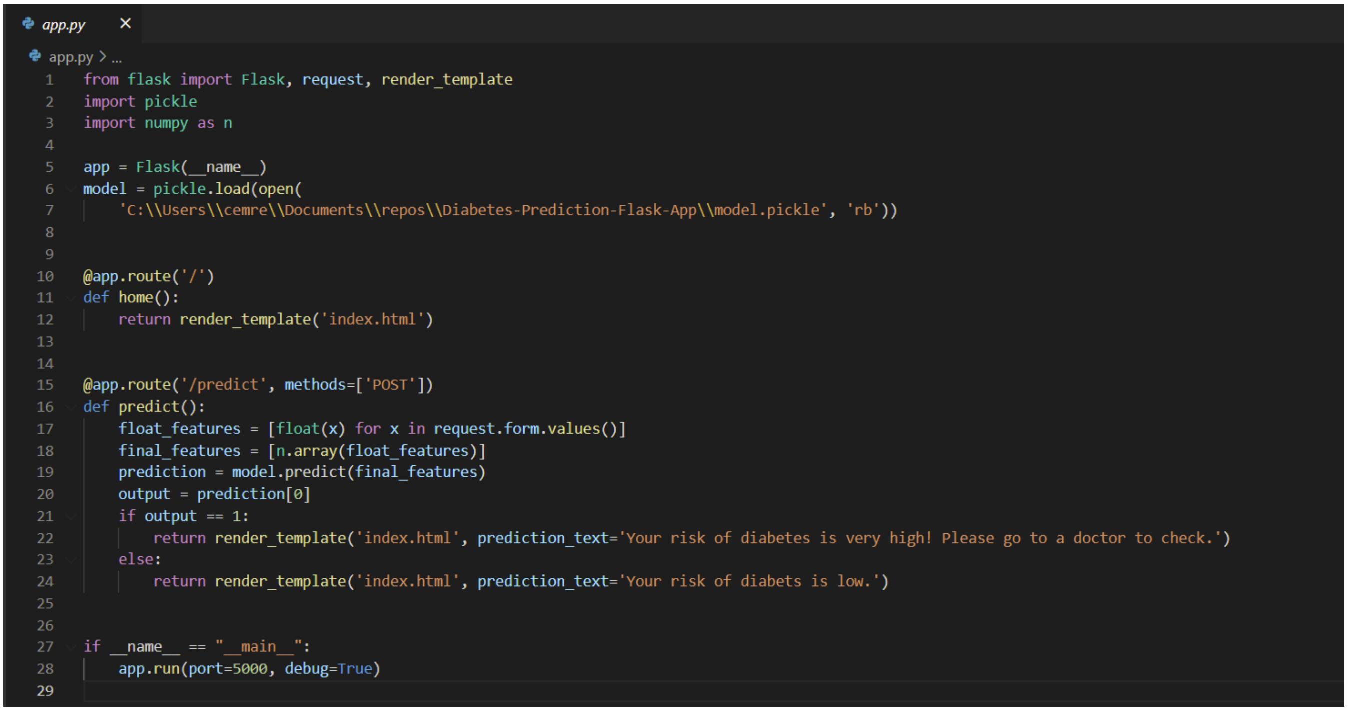


```python
import pandas as p
import pickle
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression

DiabetesPrediction = p.read_csv(
    'C:\\Users\\cemre\\Documents\\repos\\Diabetes-Prediction-Flask-App\\Dataset\\diabetes.csv')

X = DiabetesPrediction[['Pregnancies', 'Glucose', 'BloodPressure',
                        'SkinThickness', 'Insulin', 'BMI', 'DiabetesPedigreeFunction', 'Age']]
y = DiabetesPrediction['Outcome']

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.4, random_state=3)

lm = LogisticRegression(solver='liblinear')
lm.fit(X_train, y_train)
print(lm.score(X_test, y_test))
pickle.dump(lm, open('model.pickle', 'wb'))

```

Step 3: Creating the Web Application by Using Flask

By using the model, which is created before, a Flask web application is created with port number 5000. In app.py file, there are 2 functions which are home and predict. home function works when the application is first opened and after the prediction. predict function is used to gather the data and predict the result. If the prediction result is 1, the patient has a high risk of diabetes and if it is 0, the patient has a low risk of diabetes. When data is entered, the machine learning model starts to work, and the result is shown on the screen.

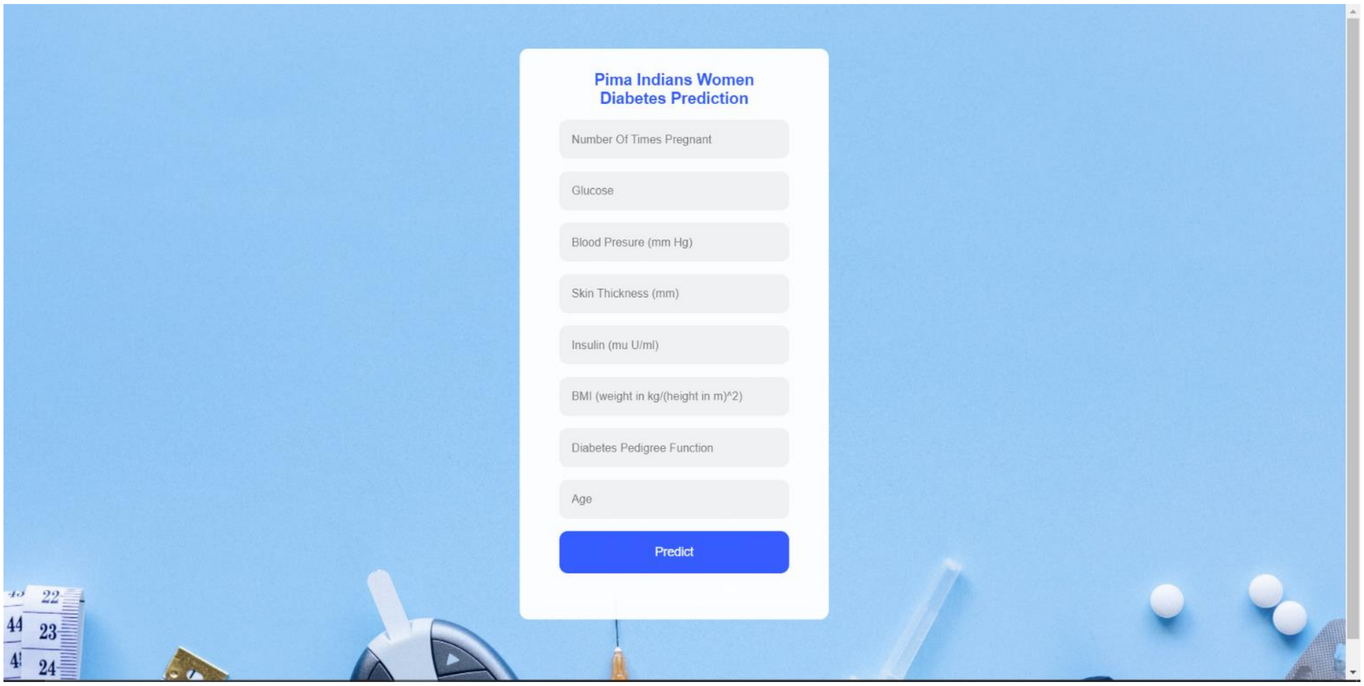
app.py:



```python
from flask import Flask, request, render_template
import pickle
import numpy as n

app = Flask(__name__)
model = pickle.load(open(
    'C:\\Users\\cemre\\Documents\\repos\\Diabetes-Prediction-Flask-App\\model.pickle', 'rb'))


@app.route('/')
def home():
    return render_template('index.html')


@app.route('/predict', methods=['POST'])
def predict():
    float_features = [float(x) for x in request.form.values()]
    final_features = [n.array(float_features)]
    prediction = model.predict(final_features)
    output = prediction[0]
    if output == 1:
        return render_template('index.html', prediction_text='Your risk of diabetes is very high! Please go to a doctor to check.')
    else:
        return render_template('index.html', prediction_text='Your risk of diabets is low.')


if __name__ == "__main__":
    app.run(port=5000, debug=True)

```

Step 4: Designing

After the machine learning model and Flask application is created, index.html and style.css files are created.
Web application:


Pima Indians Women Diabetes Prediction
Number Of Times Pregnant
Glucose
Blood Presure (mm Hg)
Skin Thickness (mm)
Insulin (mu U/ml)
BMI (weight in kg/(height in m)^2)
Diabetes Pedigree Function
Age
Predict