Diabetes Prediction - Project Documentation

DIABETES PREDICTION PROJECT - DOCUMENTATION

PROJECT OVERVIEW

This project builds a machine learning model to predict if a person is diabetic or not using medical data.
It uses an SVM (Support Vector Machine) classifier for binary classification and includes data preprocessing, model training, evaluation, and prediction.

LIBRARIES USED

- numpy, pandas
- sklearn.model_selection
- sklearn.preprocessing (StandardScaler)
- sklearn.svm (SVC)
- sklearn.metrics (accuracy_score)

PROJECT FLOW

---

1. Load and explore dataset
2. Split features (X) and labels (y)
3. Standardize data using StandardScaler
4. Split data into train and test sets
5. Train model using Support Vector Machine (linear kernel)
6. Evaluate model performance (train & test accuracy)
7. Predict outcomes for new input data

CODE SECTIONS

```
import numpy as np
import pandas as pd
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn import svm
from sklearn.metrics import accuracy_score

#loading the dataset
diabetes_dataset = pd.read_csv('/content/drive/MyDrive/Data Science/Projects/02 Diabetes Prediction/diabetes.csv')
diabetes_dataset.head(5)

diabetes_dataset.shape

#Getting the statistical measures of the data
diabetes_dataset.describe()

diabetes_dataset['Outcome'].value_counts()

diabetes_dataset.groupby('Outcome').mean()

#seperating the data and labels
x = diabetes_dataset.drop(columns='Outcome', axis=1)
y = diabetes_dataset['Outcome']



scaler = StandardScaler()

standardized_data = scaler.fit_transform(x)
```

```
standardized_data

x = standardized_data
y = diabetes_dataset['Outcome']



x_train, x_test, y_train, y_test = train_test_split(x,y, test_size=0.2, stratify=y, random_state=2)

print(x.shape, x_train.shape, x_test.shape)

classifier = svm.SVC(kernel='linear')

#Training the support vector machine classifier
classifier.fit(x_train, y_train)

#Accuracy score on training data
x_train_prediction = classifier.predict(x_train)
training_data_accuracy = accuracy_score(x_train_prediction, y_train)
training_data_accuracy*100

#On test data
x_test_prediction = classifier.predict(x_test)
test_data_prediction = accuracy_score(x_test_prediction, y_test)
test_data_prediction*100

input_data = x_test[1]

#changing input array to numpy
input_data_as_numpy_array = np.asarray(input_data)

#reshape the array as we are predicting for 1 instance, so that model can know that we are not giving the 768 data
```

```
input_data_reshaped = input_data_as_numpy_array.reshape(1,-1)

#standardize the input data
std_data = scaler.transform(input_data_reshaped)
print(std_data)

prediction = classifier.predict(std_data)
print(f"Output is : {prediction}")

if (prediction[0] == 0):
  print("The person is not diabetic")
else:
  print("The person is diabetic")
```