TECHNOHACKS

(LETS GROW TOGETHER)

An PROJECT

on

“DIABETES CLASSIFICATION”

Submitted in partial fulfillment for the INTERNSHIP

BATCH 23

IN

MACHINE LEARNING DOMAIN

Submitted by

AFSHA SULTHANA

Under the guidance of

Mr.SANDIP GAVIT

TECHNOHACKS EDUTECH INTERNSHIP PROGRAM

2023

DIABETES CLASSIFICATION:

use a dataset containing medical data of pateints to predict if a person has diabetes or not.

Certainly! To build a model to predict whether a person has diabetes or not using a dataset containing medical data, you can follow these general steps. In this example, I'll use the popular diabetes dataset from scikit-learn.

CODE:

```
# Import necessary libraries
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix

# Load the diabetes dataset
from sklearn.datasets import load_diabetes
diabetes = load_diabetes()
data = pd.DataFrame(data=np.c_[diabetes['data'], diabetes['target']], columns=diabetes['feature_names'] + ['target'])

# Assuming the target variable is binary (1 for diabetes, 0 for no diabetes)
data['target'] = data['target'].apply(lambda x: 1 if x > 150 else 0)

# Split the data into features (X) and target variable (y)
X = data.drop('target', axis=1)
y = data['target']
```

```
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

# Standardize the features
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

# Build and train the Logistic Regression classifier
logreg_classifier = LogisticRegression(random_state=42)
logreg_classifier.fit(X_train_scaled, y_train)

# Make predictions on the test set
y_pred = logreg_classifier.predict(X_test_scaled)

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
class_report = classification_report(y_test, y_pred)

print(f'Accuracy: {accuracy}')
print(f'Confusion Matrix:\n{conf_matrix}')
print(f'Classification Report:\n{class_report}')
```

Note: This example assumes that the target variable in the diabetes dataset is continuous. In this case, I've transformed it into a binary variable based on a threshold value (150). You may need to adapt this threshold or use a different approach depending on your specific dataset.

Also, replace the dataset and preprocessing steps with your actual medical dataset. Ensure that your dataset has appropriate features and a binary target variable indicating the presence or absence of diabetes.

OUTPUT:

```
Accuracy: 0.7640449438202247
Confusion Matrix:
[[42  7]
 [14 26]]
Classification Report:
              precision    recall  f1-score   support

           0       0.75      0.86      0.80        49
           1       0.79      0.65      0.71        40

    accuracy                           0.76        89
   macro avg       0.77      0.75      0.76        89
weighted avg       0.77      0.76      0.76        89
```