TECHNOHACKS

(LETS GROW TOGETHER)

An PROJECT

on

"IRIS FLOWERS 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

To classify the iris flowers:

use the famous iris dataset to build a model that can classify iris flowers based on their sepal and petal dimensions.

Certainly! The Iris dataset is a classic dataset in machine learning and is often used for classification tasks. In this case, we'll use Python and the popular machine learning library, scikit-learn, to build a model that can classify Iris flowers based on their sepal and petal dimensions.

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.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix

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

# 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 k-nearest neighbors classifier
k = 3  # You can choose a different value for k
knn_classifier = KNeighborsClassifier(n_neighbors=k)
knn_classifier.fit(X_train_scaled, y_train)

# Make predictions on the test set
y_pred = knn_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}')
```

OUTPUT:

```
Accuracy: 1.0
Confusion Matrix:
[[10  0  0]
 [ 0  9  0]
 [ 0  0 11]]
Classification Report:
              precision    recall  f1-score   support

         0.0       1.00      1.00      1.00        10
         1.0       1.00      1.00      1.00         9
         2.0       1.00      1.00      1.00        11

    accuracy                           1.00        30
   macro avg       1.00      1.00      1.00        30
weighted avg       1.00      1.00      1.00        30
```

[redacted]

This code uses the k-nearest neighbors (KNN) algorithm for classification. You can experiment with different values of **k** and try other classification algorithms provided by scikit-learn.