**Code & Giải Thích Code**

**Khai Báo Thư Viện**

import numpy as np

import matplotlib

import matplotlib.pyplot as plt

import pandas as pd

from pandas.plotting import scatter\_matrix

from matplotlib import pyplot

import seaborn as sns; sns.set()

**Tải Bộ Dữ Liệu Lên**

url = "https://raw.githubusercontent.com/jbrownlee/Datasets/master/iris.csv"

names = ['sepal-length', 'sepal-width', 'petal-length', 'petal-width', 'class']

dataset = pd.read\_csv(url, names=names)  #đọc dữ liệu

# Ma Trận Dữ Liệu và 5 Phần Tử Đầu

print(dataset.shape)

print(dataset)

# Mô Tả Dữ Liệu

**Mô tả chức năng hàm**

**pd.read\_csv(url,names=names):** đọc data từ đường link url dưới dạng file csv

**Mô Tả Thành Phần Của Bộ Dữ Liệu**

print(dataset.describe())

print(dataset.groupby('class').size())

**Mô tả chức năng hàm**

**dataset.describe() :** mô tả về dataset bằng các giá trị

**dataset.groupby('class').size():** nhóm các loại hoa với nhau và đưa ra số lượng mỗi loại

**Tách 1 Ít Phần Tử Để Làm Thành Bộ Test**

hide\_data=dataset.iloc[145:150,:]

hide\_data

**Mô tả chức năng hàm**

**dataset.iloc[145:150,:]:** lấy data từ dataframe hide\_data từ index 145 đến 149 theo toàn bộ giá trị cột.

**Khai Báo Bộ Phân Tách Dữ Liệu Train & Thực Hiện Phân Tách**

from sklearn.model\_selection import train\_test\_split

array = dataset.iloc[0:145,:].values

X = array[:,0:4]

y = array[:,4]

X\_train, X\_validation, Y\_train, Y\_validation = train\_test\_split(X, y, test\_size=0.3, random\_state=1, shuffle=True)

**Mô tả chức năng hàm**

**train\_test\_split()** để chia tỉ lệ bộ train,test theo tỉ lệ tùy chọn

**Dùng biểu đồ để xem mối quan hệ giữa các đại lượng**

scatter\_matrix(dataset)

plt.show()

**Mô tả chức năng hàm**

**scatter\_matrix():** biểu diễn dữ liệu theo ma trận scatter ( ma trận biểu diễn quan hệ theo từng cặp giá trị)

**plt.show()** : visualize ma trận

**Thực Hiện Việc Kiểm Tra Và Đánh Giá Các Mô Hình Lên Hoa Iris bằng các thuật toán classification cơ bản do sklearn hỗ trợ**

# Logistic Regression

from sklearn.linear\_model import LogisticRegression

# Decision Tree & Random Forest

from sklearn.tree import DecisionTreeClassifier

from sklearn.ensemble import RandomForestClassifier

# K-NN

from sklearn.neighbors import KNeighborsClassifier

# Gausian Naive Bayes

from sklearn.naive\_bayes import GaussianNB

# SVM

from sklearn.svm import SVC

models = []

models.append(('LR', LogisticRegression(solver='liblinear', multi\_class='ovr')))

models.append(('KNN', KNeighborsClassifier()))

models.append(('RF', RandomForestClassifier()))

models.append(('DTC', DecisionTreeClassifier()))

models.append(('NB', GaussianNB()))

models.append(('SVM', SVC(gamma='auto')))

# Đánh Giá Mô Hình bằng phương pháp đánh giá chéo

from sklearn.model\_selection import cross\_val\_score

results = []

names = []

for name, model in models:

    cv\_results = cross\_val\_score(model, X\_train, Y\_train, cv=10, scoring='accuracy')

    results.append(cv\_results)

    names.append(name)

    print(name, cv\_results.mean())  #cv\_results.mean: trung bình hiệu suất của 10 lần fold

**Mô tả chức năng hàm**

**models.append():** thêm các model vào mảng models

**cross\_val\_score():** tiến hành đánh giá chéo theo số lần tùy chọn và cho ra kết quả theo mỗi lần đánh giá

**Thực Hiện Show Bảng Thống Kê So Sánh Các Thuật Toán**

# Compare Algorithms

pyplot.boxplot(results, labels=names)

pyplot.title('Algorithm Comparison')

pyplot.show()

**Mô tả chức năng hàm**

**boxplot():** biển diễn dữ liệu theo biểu đồ hộp

**pyplot.title():** đặt tên cho biểu đồ

**Tiến hành train model bằng thuật toán SVM**

# Kiểm Chứng Dự Đoán Lên Bộ Dữ Liệu Đánh Giá

from sklearn.metrics import accuracy\_score

model = SVC(gamma='auto')

model.fit(X\_train, Y\_train)

predictions = model.predict(X\_validation)

print(accuracy\_score(Y\_validation, predictions))

**Mô tả chức năng hàm**

**model.fit():** đào tạo model

**model.predict():** dự đoán kết quả với bộ dữ liệu tương ứng

**accuracy\_score():** số lần dự đoán đúng / tổng phần tử dự đoán => đánh giá hiệu suất model

**Biểu diễn kết quả dự đoán dưới dạng confusion matrix**

from sklearn.metrics import confusion\_matrix

target\_names=['Iris-setosa','Iris-versicolor','Iris-virginica']

mat=confusion\_matrix(Y\_validation,predictions)

sns.heatmap(mat,square=True,annot=True,cbar=False,xticklabels=target\_names,yticklabels=target\_names)

plt.xlabel('true label')

plt.ylabel('predicted label')

**Mô tả chức năng hàm**

**confusion\_matrix()** **:** biểu diễn số lượng đặc trưng giống và khác nhau giữa 2 bộ dữ liệu

**sns.heatmap():** visualize các giá trị ma trận

**plt.xlabel():** đặt tên cho trục x

**plt.ylabel():** đặt tên cho trục y

**Đánh giá f1-score**

from sklearn.metrics import classification\_report

print(classification\_report(Y\_validation, predictions))

**Mô tả chức năng hàm**

**classification\_report():** tính toán các giá trị precision,recall,f1-score theo từng feature của 2 bộ dữ liệu => đánh giá độ chính xác của model

**Tuning Hyperparameter sử dụng GridSearchCV do sklearn cung cấp**

from sklearn.model\_selection import GridSearchCV

param\_grid={

             'kernel':['linear','rbf','poly'],

            }

search=GridSearchCV(model,param\_grid,n\_jobs=-1)

search.fit(X\_validation,Y\_validation)

search.best\_params\_

**Mô tả chức năng hàm**

**GridSearchCV():** đánh giá và lựa chọn các tham số được cung cấp để tìm ra tham số làm cho model đạt hiệu suất cao nhất

**GridSearchCV().bestparams :** in ra tham số được chọn sau khi thực hiện hàm GridSearchCV()

**Tiến hành train và đánh lại model với tham số kernel thay đổi**

tuning\_model = SVC(gamma='auto',kernel='poly')

tuning\_model.fit(X\_train, Y\_train)

predictions = tuning\_model.predict(X\_validation)

print(accuracy\_score(Y\_validation, predictions))

**In ra báo cáo phân loại**

print(classification\_report(Y\_validation, predictions))

**Hàm dự đoán loại hoa với đầu vào là 4 thông số chiều dài,chiều rộng của cánh hoa và đài hoa**

def Predict\_Func(x,y,z,t,model=tuning\_model,train=X\_train):

    arr=[[x,y,z,t]]

    pred=model.predict(arr)

    print("Loai hoa duoc model du doan la: ",pred[0])

    print("Thank you")

hide\_data

Predict\_Func(6.5,3.0,5.2,2.0)

* **Hướng dẫn sử dụng: gọi hàm với tham số truyền vào lần lượt là giá trị chiều dài chiều rộng của cánh hoa, chiều dài chiều rộng của đài hoa . Output: loại hoa được model dự đoán**