**EXPERIMENT-4**

**Student’s Name: Kanishk Soni UID: 20BCS9398**

**Section/Group: 707\_WM\_B Subject Code: 20CSP-317**

**Subject Name: ML Lab Date of performance:10/10/2022**

**Branch: BE CSE Semester:5th**

**Aim:** Classifying data using Support Vector Machines(SVMs) in Python.

**Objective:** To do SVM on data set.

**Software/Hardware Requirements:** Windows 7 & above version.

**Tools to be used:**

1. Anaconda Jupyter Notebook,
2. numpy, pandas, matplotlib, seaborn, sklearn.

**Introduction to SVMs:**

In machine learning, support vector machines (SVMs, also support vector networks) are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. A Support Vector Machine (SVM) is a discriminative classifier formally defined by a separating hyperplane.

**What is Support Vector Machine?**

An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. In addition to performing linear classification, SVMs can efficiently perform a non-linear classification, implicitly mapping their inputs into high-dimensional feature spaces.

**Code:**

--

import numpy as np

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

get\_ipython().run\_line\_magic('matplotlib', 'inline')

--

iris = pd.read\_csv('Iris.csv')

iris.head()

--

sns.set\_style('whitegrid')

sns.pairplot(iris, hue='Species',palette='Dark2')

--

setosa = iris[iris['Species']=='Iris-setosa']

sns.kdeplot(setosa['SepalWidthCm'], setosa['SepalLengthCm'], cmap="plasma", shade=True, shade\_lowest=False)

--

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

X=iris.drop(['Species'],axis=1)

y=iris['Species']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.30, random\_state=101)

X\_train.head()

--

from sklearn.svm import SVC

from sklearn.metrics import classification\_report,confusion\_matrix

from sklearn.model\_selection import GridSearchCV

--

model=SVC()

model.fit(X\_train,y\_train)

predictions = model.predict(X\_test)

--

print(confusion\_matrix(y\_test,predictions))

print(classification\_report(y\_test,predictions))

--

param\_grid = {'C': [0.1,1, 10, 100], 'gamma': [1,0.1,0.01,0.001]}

grid = GridSearchCV(SVC(),param\_grid,refit=True,verbose=2)

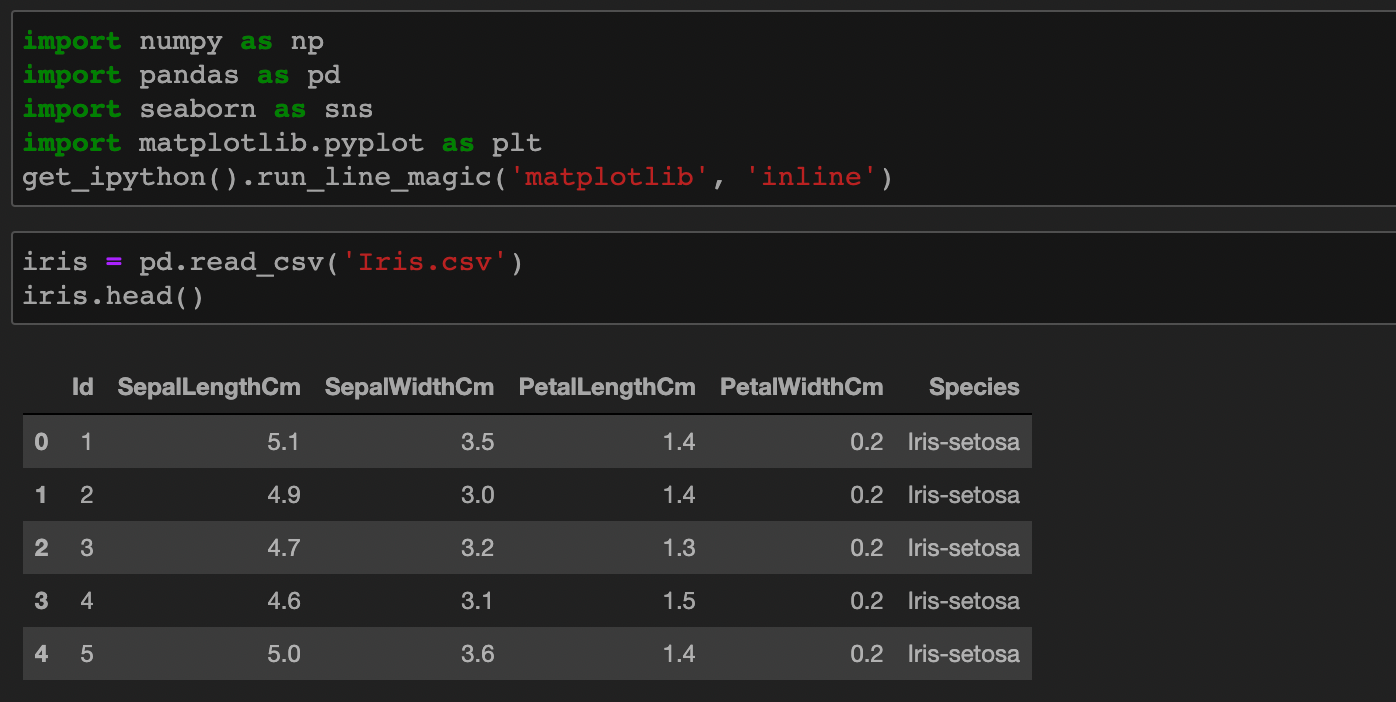
grid.fit(X\_train,y\_train)

grid\_predictions = grid.predict(X\_test)

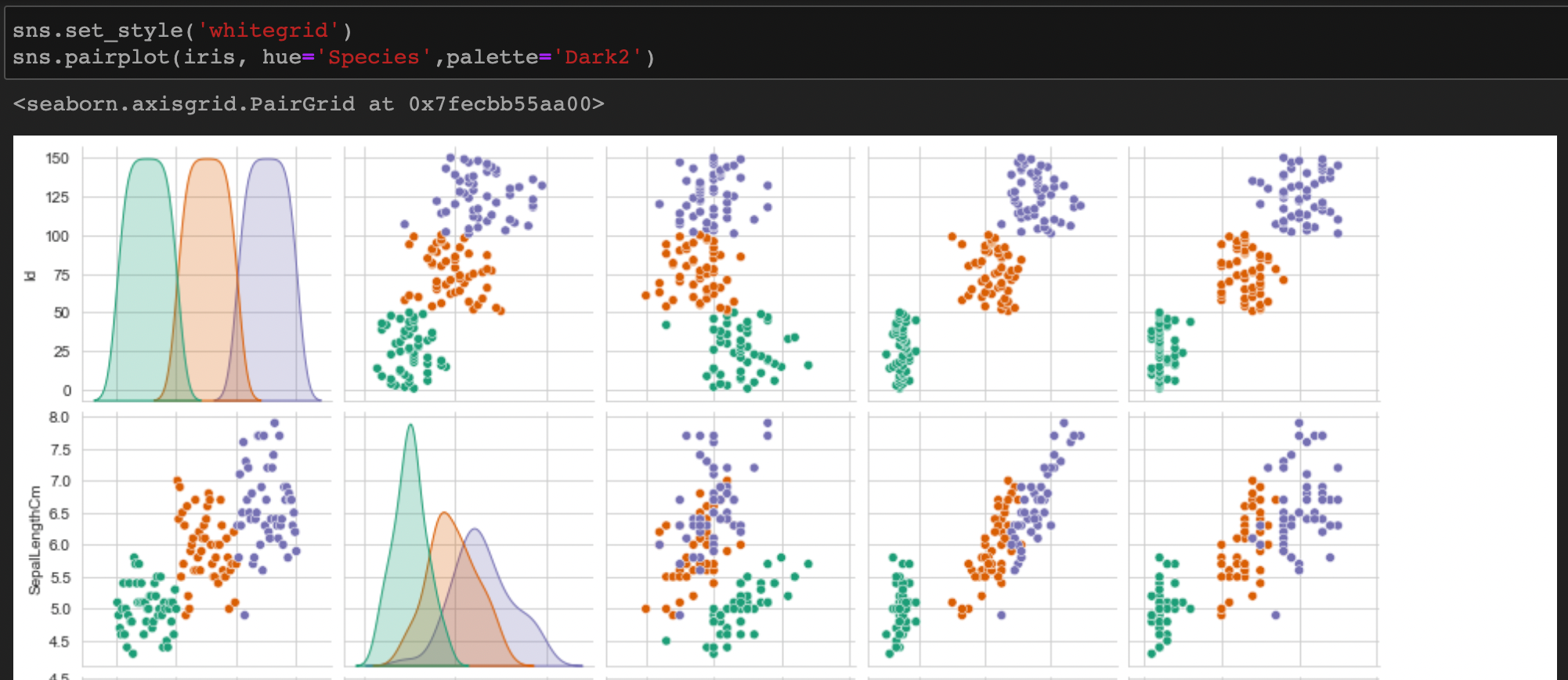
print(confusion\_matrix(y\_test,grid\_predictions))

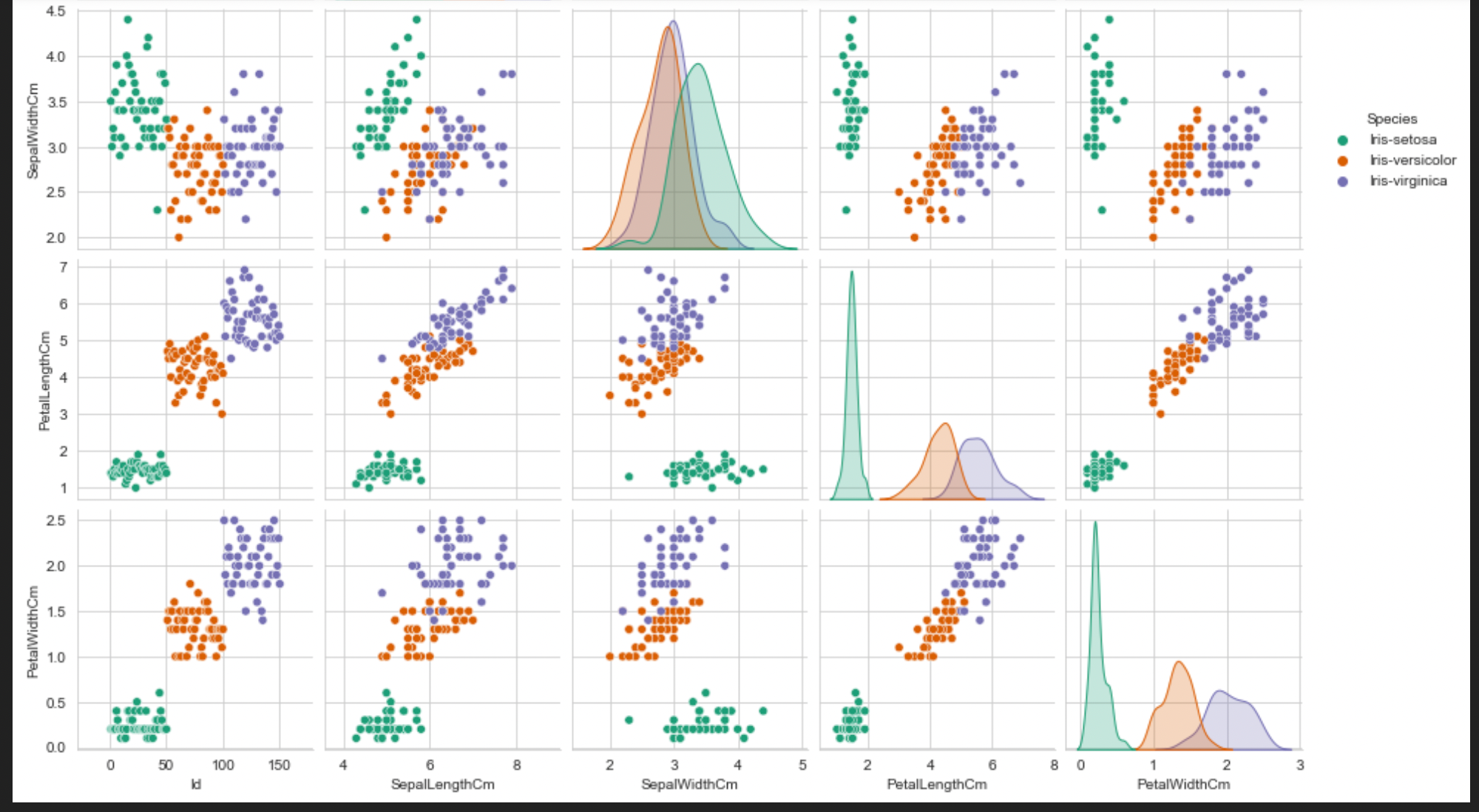
print(classification\_report(y\_test,grid\_predictions))

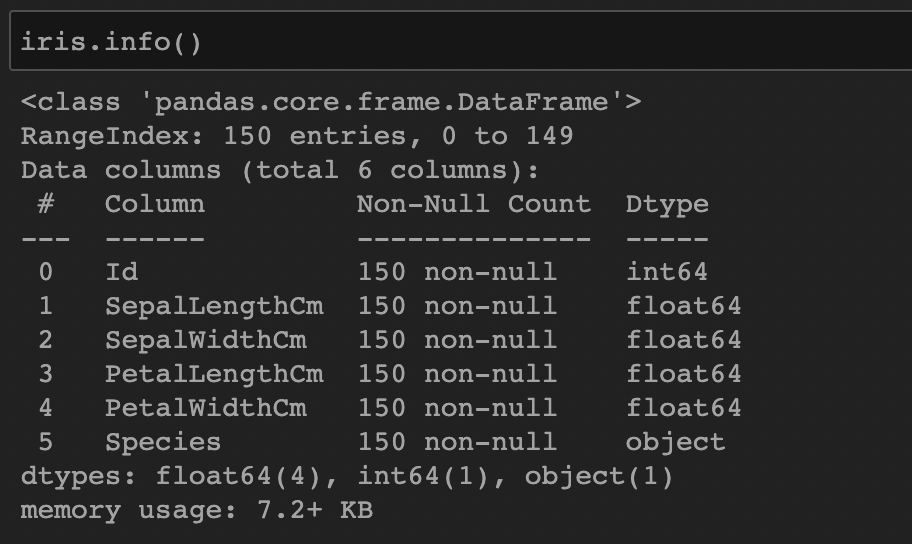
**Output:**

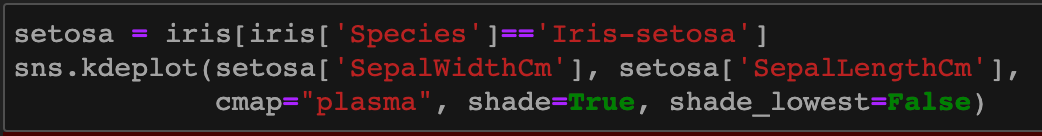


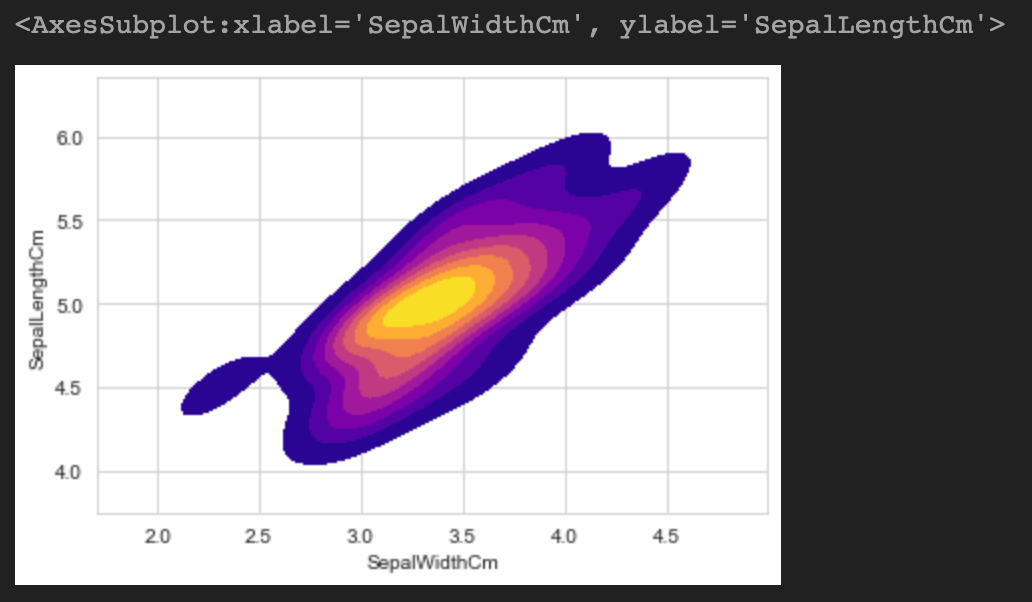
--

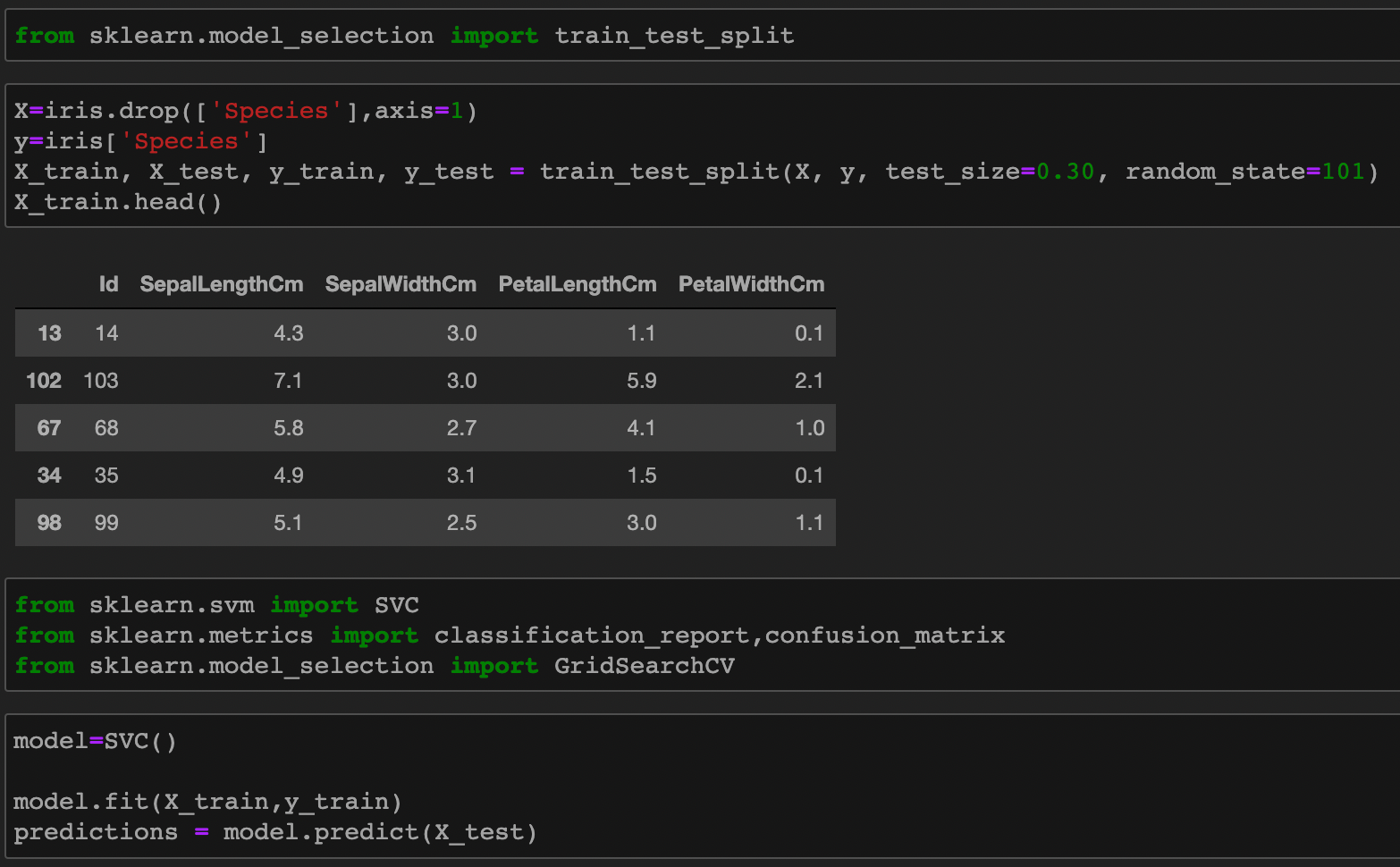


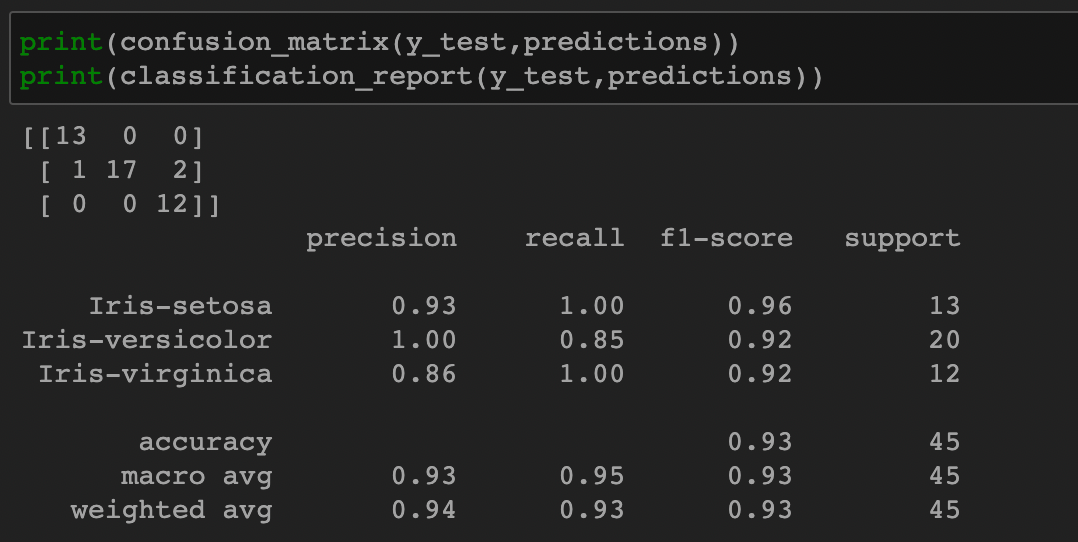


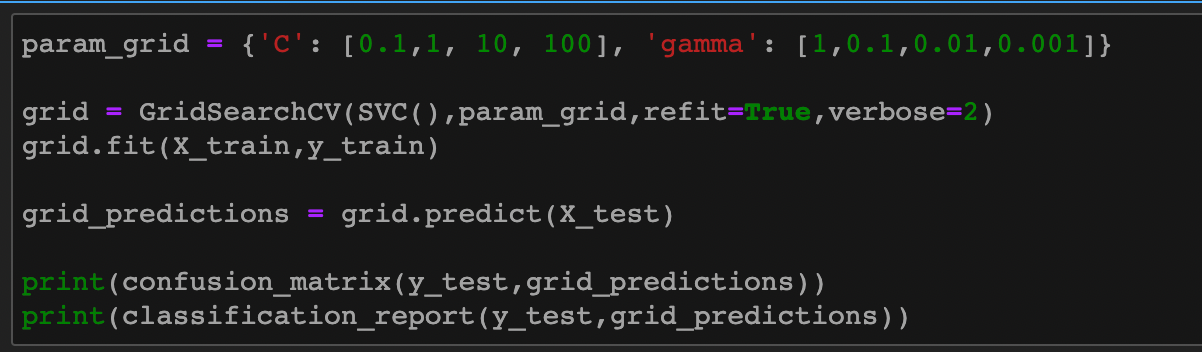


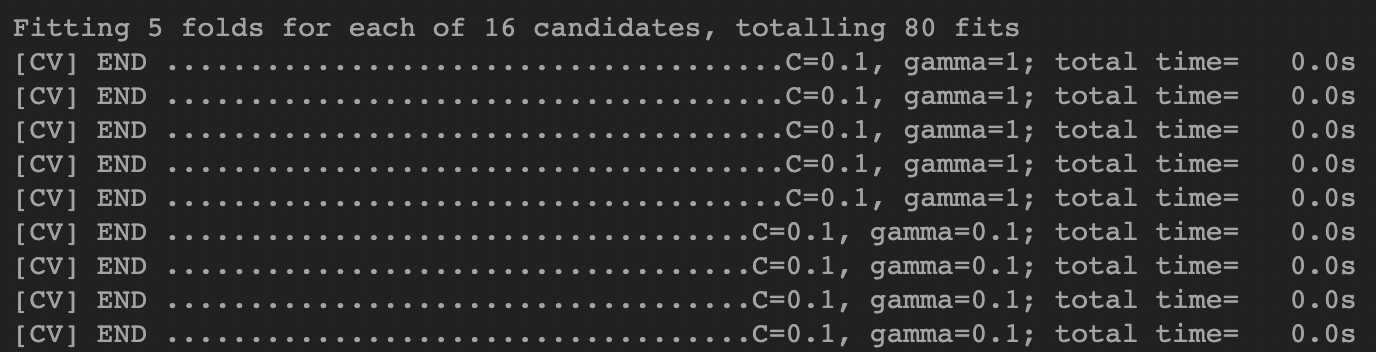




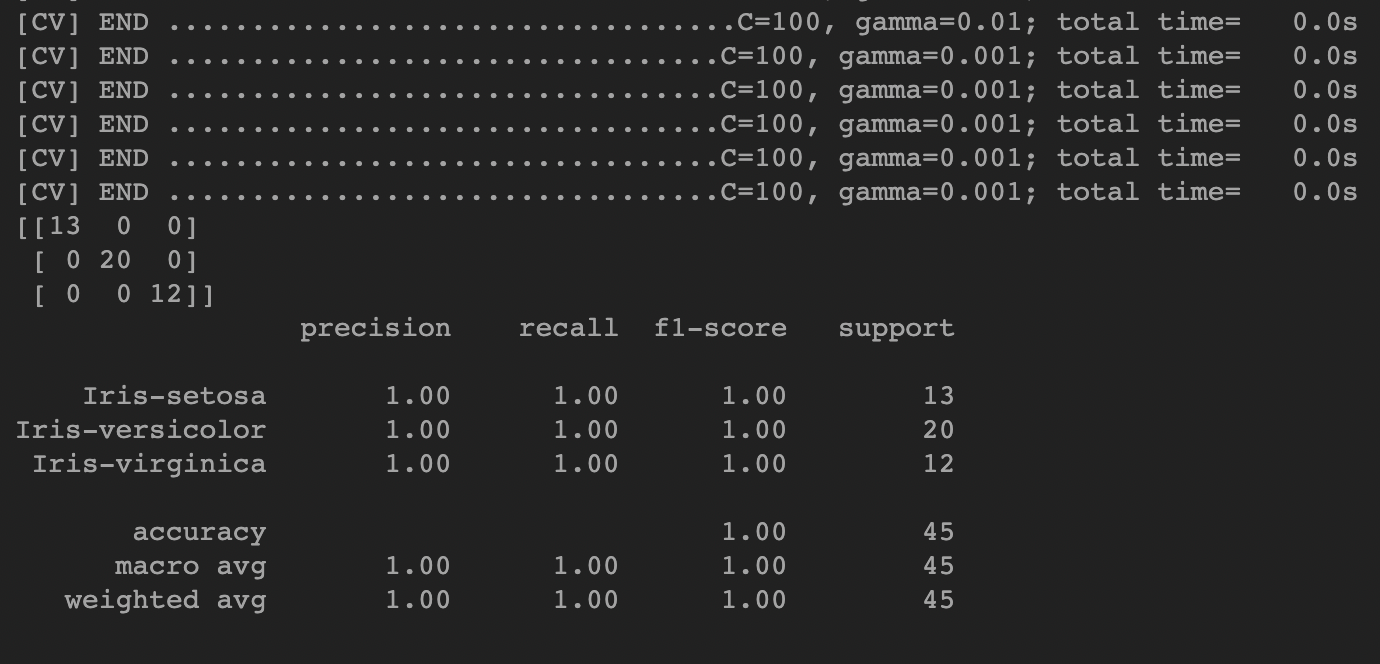








••• •••



**Learning Outcomes:**

1. We learned about Support Vector Machine.
2. We learned about how to do find hyperplane for linear and non-linear data-sets.