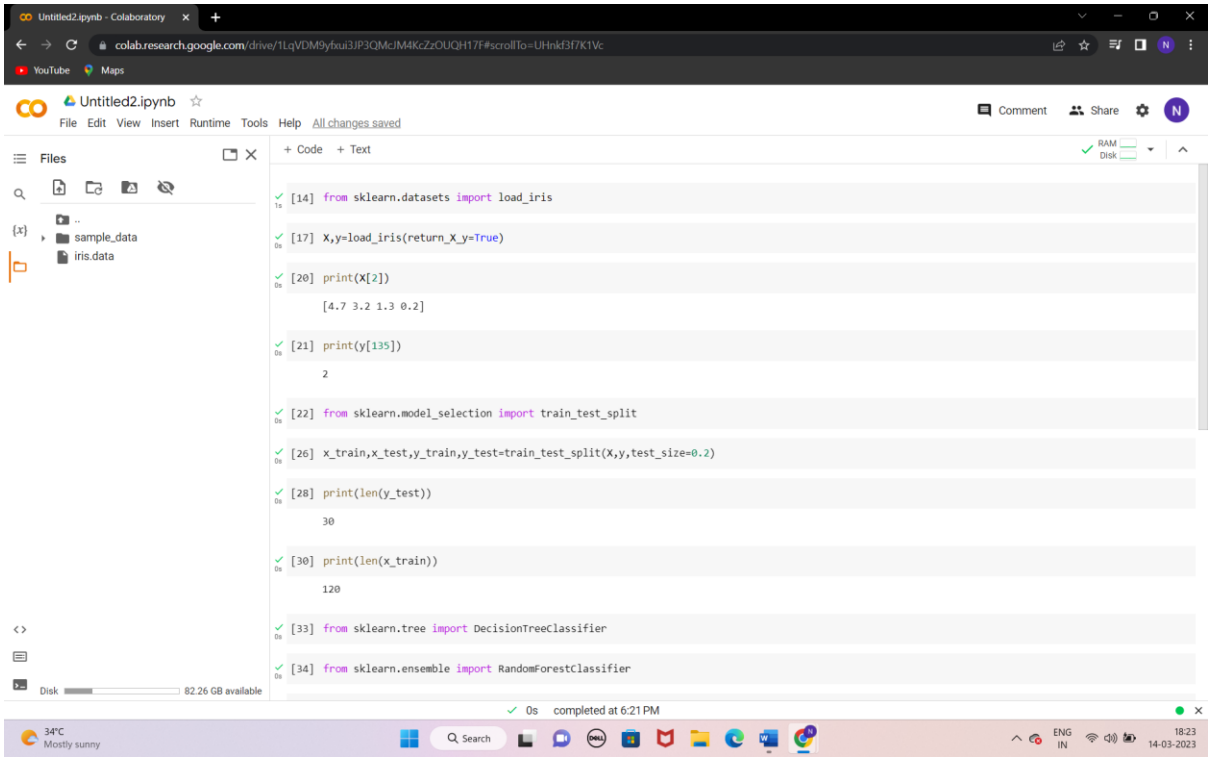
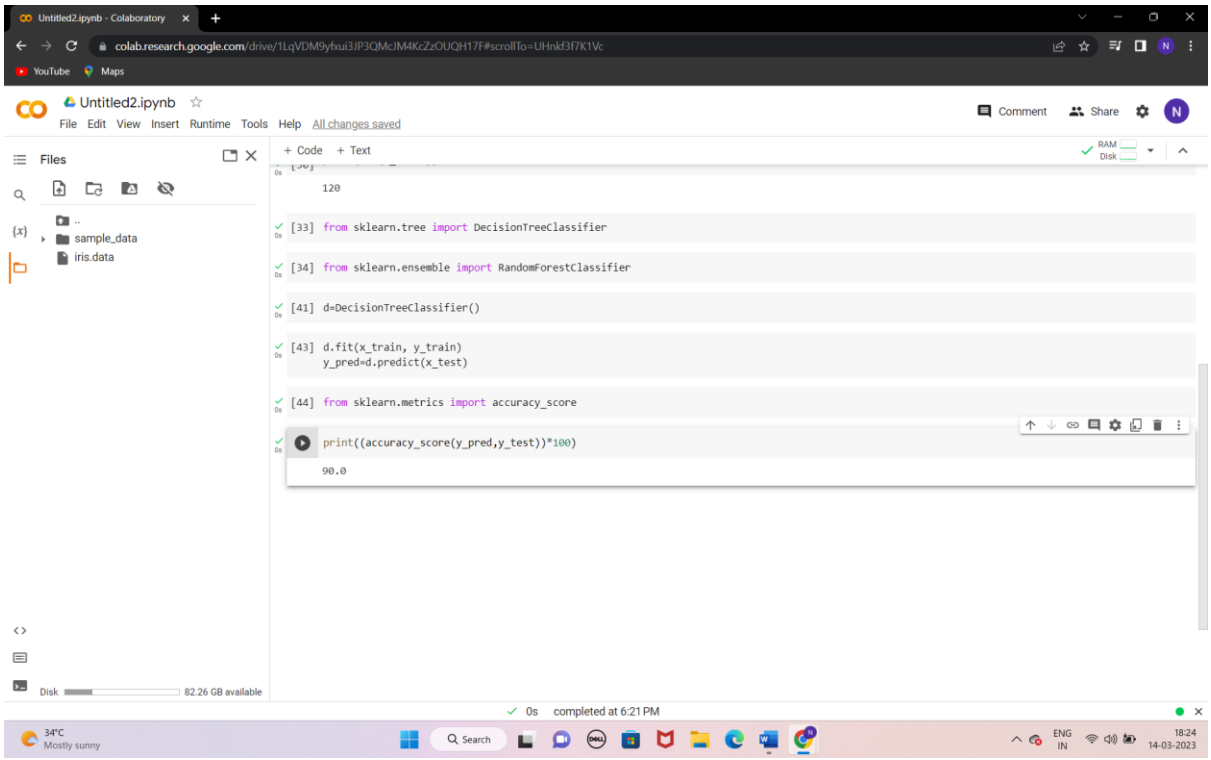
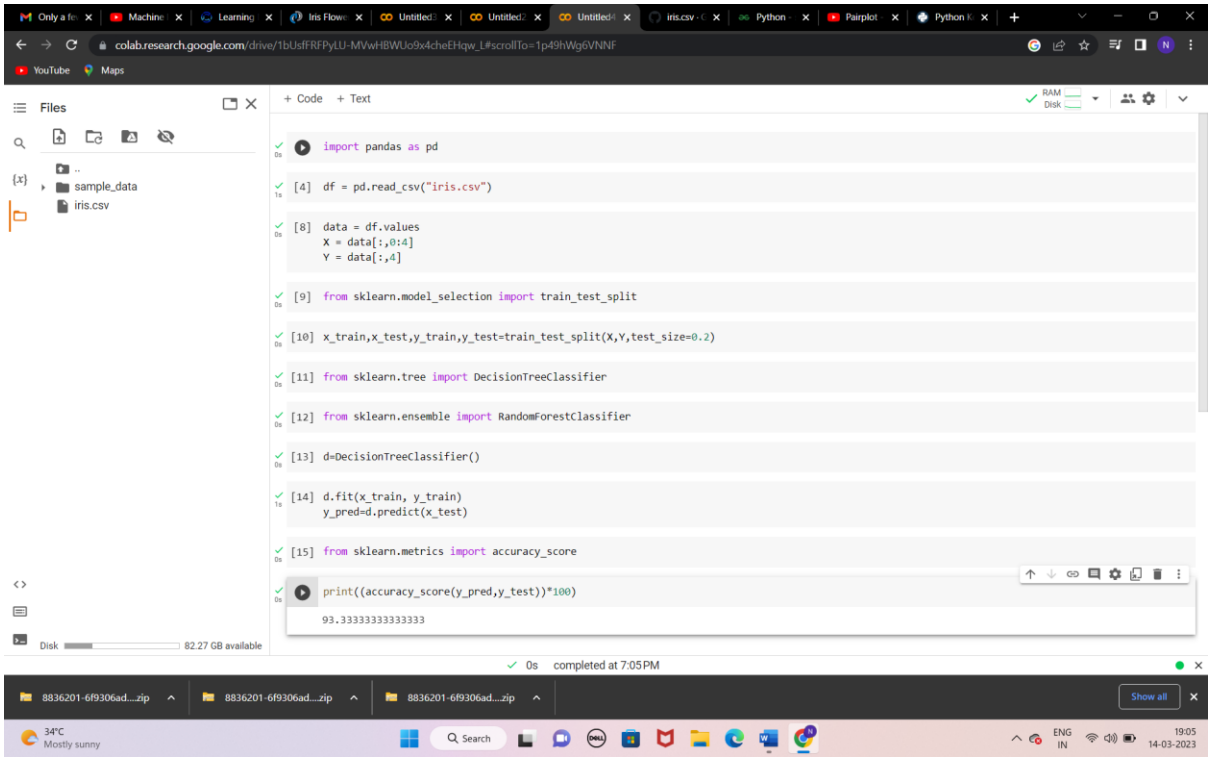
# IRIS FLOWER CLASSIFIER

1.

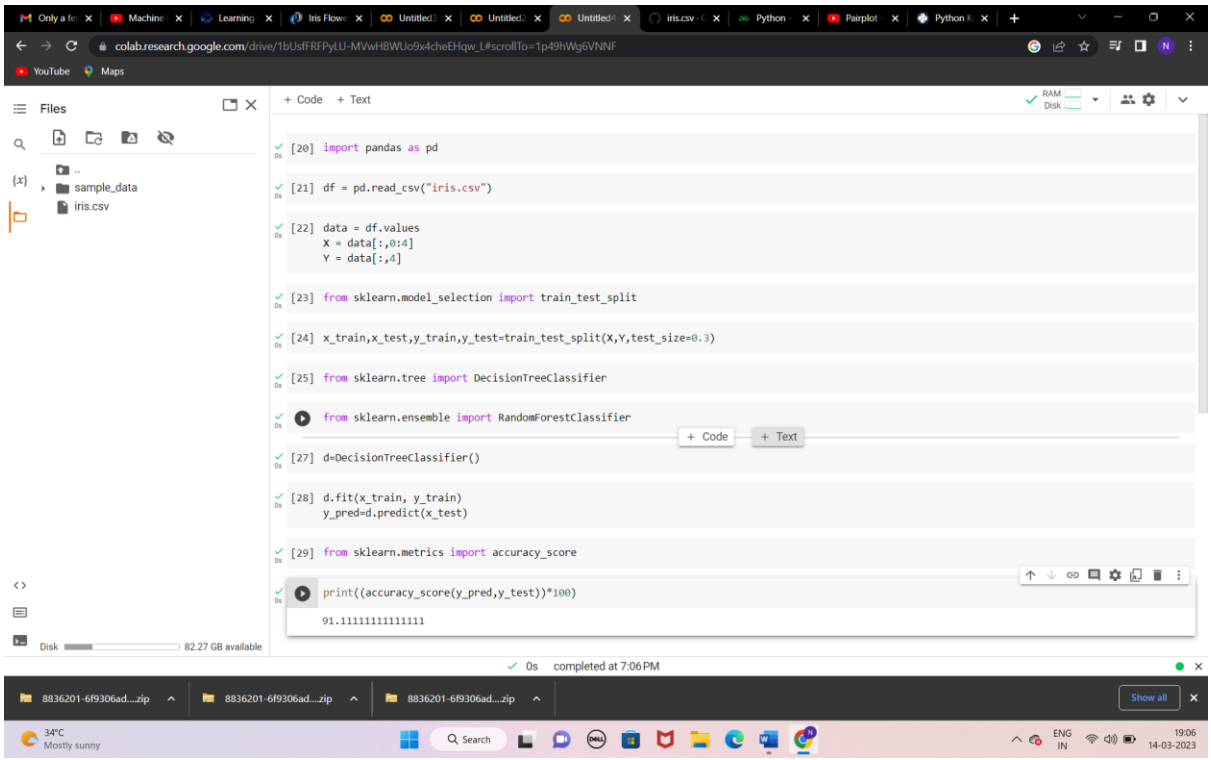


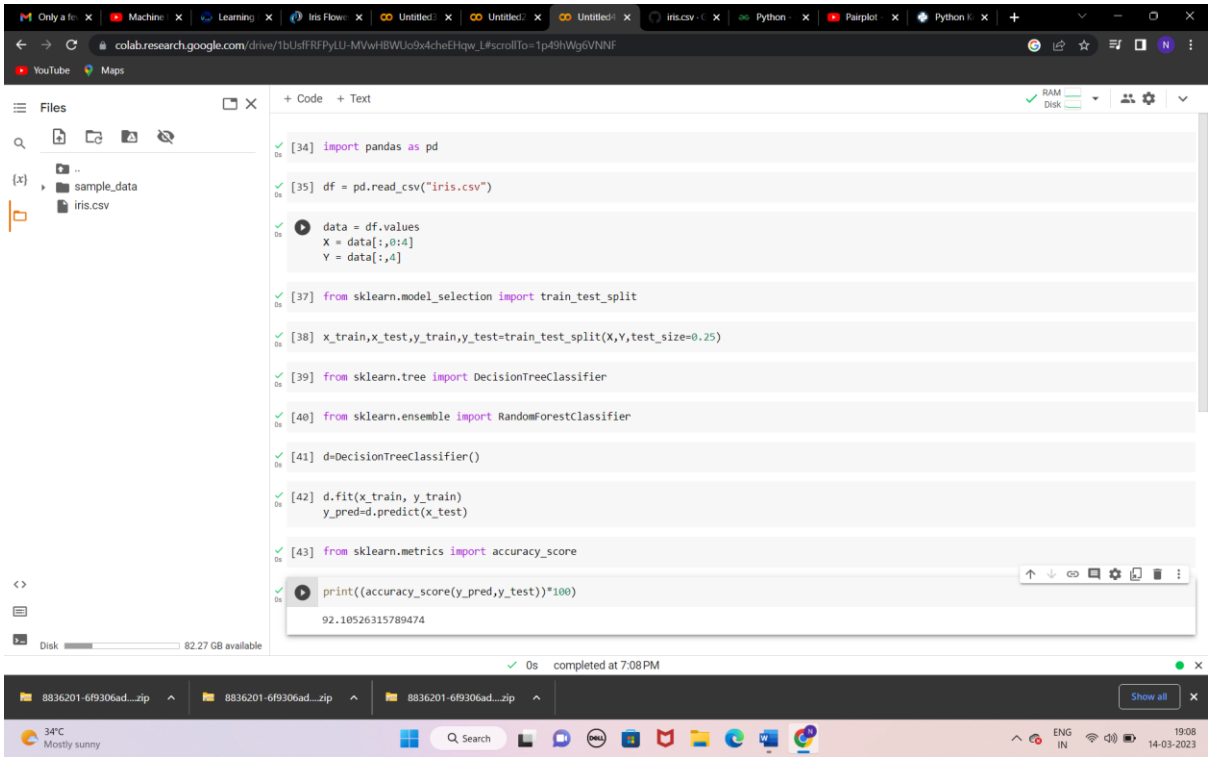



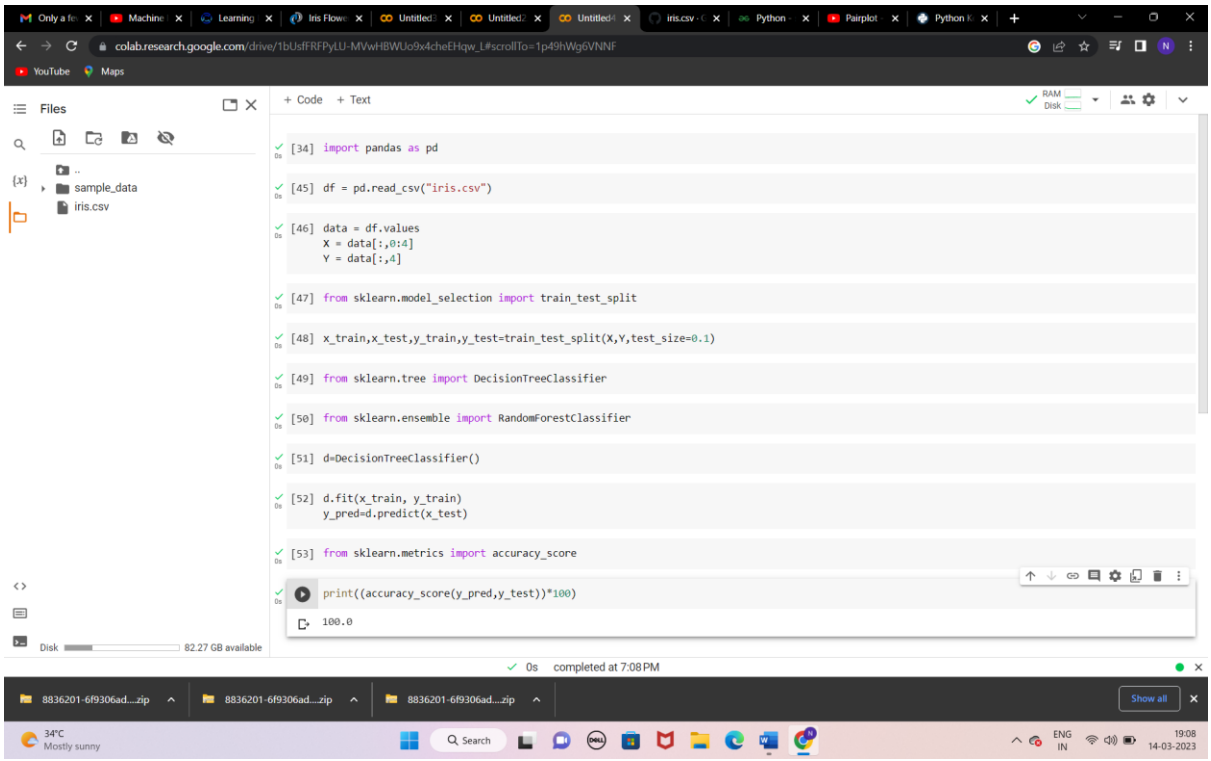
2.


colab.research.google.com/drive/1bUsfFRFPyLU-MVwHBWUo9x4cheEHqw_L#scrollTo=1p49hWg6VNNF
Files
sample_data
iris.csv
+ Code + Text
import pandas as pd
df = pd.read_csv("iris.csv")
data = df.values
X = data[:,0:4]
Y = data[:,4]
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(X,Y,test_size=0.2)
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
d=DecisionTreeClassifier()
d.fit(x_train, y_train)
y_pred=d.predict(x_test)
from sklearn.metrics import accuracy_score
print((accuracy_score(y_pred,y_test))*100)
93.33333333333333
Disk 82.27 GB available
0s completed at 7:05PM

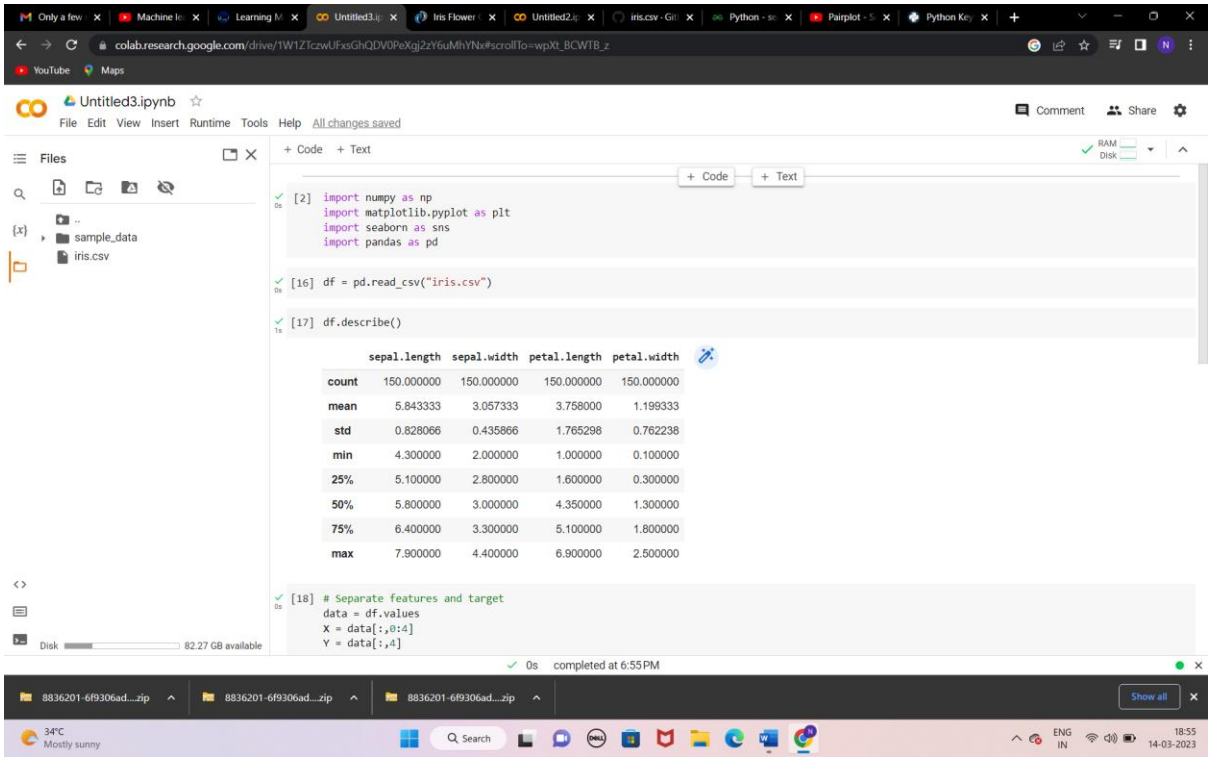
3.

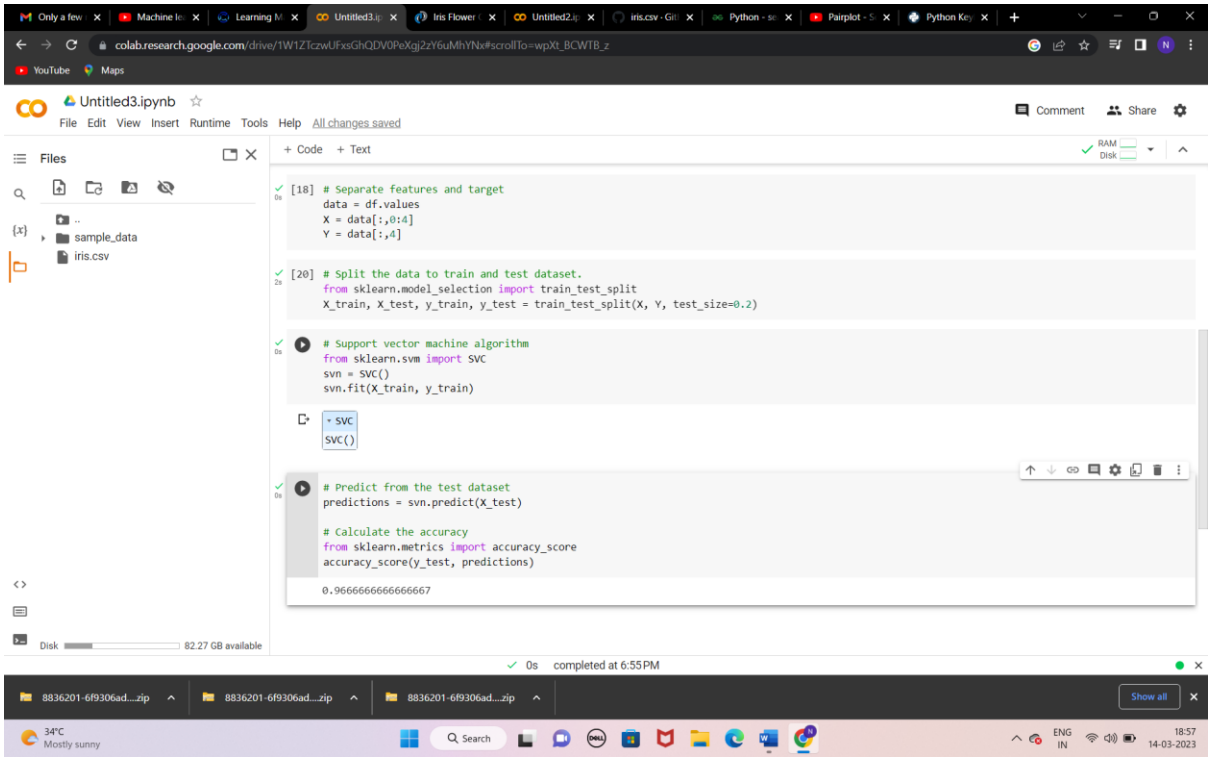

colab.research.google.com/drive/1bUsfFRFPyLU-MVwHBWUo9x4cheEHqw_L#scrollTo=1p49hWg6VNNF
Files
sample_data
iris.csv
+ Code + Text
import pandas as pd
df = pd.read_csv("iris.csv")
data = df.values
X = data[:,0:4]
Y = data[:,4]
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(X,Y,test_size=0.3)
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
d=DecisionTreeClassifier()
d.fit(x_train, y_train)
y_pred=d.predict(x_test)
from sklearn.metrics import accuracy_score
print((accuracy_score(y_pred,y_test))*100)
91.11111111111111
Disk 82.27 GB available
0s completed at 7:06PM


colab.research.google.com/drive/1bUsfFRFPyLU-MVwHBWUo9x4cheEHqw_L#scrollTo=1p49hWg6VNNF
Files
sample_data
iris.csv
+ Code + Text
[34] import pandas as pd
[35] df = pd.read_csv("iris.csv")
data = df.values
X = data[:,0:4]
Y = data[:,4]
[37] from sklearn.model_selection import train_test_split
[38] x_train,x_test,y_train,y_test=train_test_split(X,Y,test_size=0.25)
[39] from sklearn.tree import DecisionTreeClassifier
[40] from sklearn.ensemble import RandomForestClassifier
[41] d=DecisionTreeClassifier()
[42] d.fit(x_train, y_train)
y_pred=d.predict(x_test)
[43] from sklearn.metrics import accuracy_score
print((accuracy_score(y_pred,y_test))*100)
92.10526315789474
Disk 82.27 GB available
0s completed at 7:08 PM


colab.research.google.com/drive/1bUsfFRFPyLU-MVwHBWUo9x4cheEHqw_L#scrollTo=1p49hWg6VNNF
Files
sample_data
iris.csv
+ Code + Text
[34] import pandas as pd
[45] df = pd.read_csv("iris.csv")
[46] data = df.values
X = data[:,0:4]
Y = data[:,4]
[47] from sklearn.model_selection import train_test_split
[48] x_train,x_test,y_train,y_test=train_test_split(X,Y,test_size=0.1)
[49] from sklearn.tree import DecisionTreeClassifier
[50] from sklearn.ensemble import RandomForestClassifier
[51] d=DecisionTreeClassifier()
[52] d.fit(x_train, y_train)
y_pred=d.predict(x_test)
[53] from sklearn.metrics import accuracy_score
print((accuracy_score(y_pred,y_test))*100)
100.0
Disk 82.27 GB available
0s completed at 7:08 PM

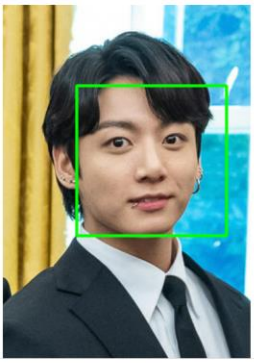
4.


Untitled3.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
Files
sample_data
iris.csv
+ Code + Text
[2] import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
[16] df = pd.read_csv("iris.csv")
[17] df.describe()
sepal.length sepal.width petal.length petal.width
count 150.000000 150.000000 150.000000 150.000000
mean 5.843333 3.057333 3.758000 1.199333
std 0.828066 0.435866 1.765298 0.762238
min 4.300000 2.000000 1.000000 0.100000
25% 5.100000 2.800000 1.600000 0.300000
50% 5.800000 3.000000 4.350000 1.300000
75% 6.400000 3.300000 5.100000 1.800000
max 7.900000 4.400000 6.900000 2.500000
[18] # Separate features and target
data = df.values
X = data[:,0:4]
Y = data[:,4]
Disk 82.27 GB available
0s completed at 6:55PM


Untitled3.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
Files
sample_data
iris.csv
+ Code + Text
[18] # Separate features and target
data = df.values
X = data[:,0:4]
Y = data[:,4]
[20] # Split the data to train and test dataset.
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.2)
# Support vector machine algorithm
from sklearn.svm import SVC
svn = SVC()
svn.fit(X_train, y_train)
SVC
SVC()
# Predict from the test dataset
predictions = svn.predict(X_test)
# Calculate the accuracy
from sklearn.metrics import accuracy_score
accuracy_score(y_test, predictions)
0.9666666666666667
Disk 82.27 GB available
0s completed at 6:55PM

# IMAGE RECOGNITION USING HAAR CASCADE .

```
import cv2
from google.colab.patches import cv2_imshow

face_cascade=cv2.CascadeClassifier(cv2.data.haarcascades + "haarcascade_frontalface_default.xml")

img = cv2.imread('image.jpg')

gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

faces = face_cascade.detectMultiScale(gray, scaleFactor=1.1, minNeighbors=5, minSize=(30, 30))

for (x, y, w, h) in faces:
    cv2.rectangle(img, (x, y), (x+w, y+h), (0, 255, 0), 2)

cv2_imshow(img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```



0s completed at 9:16 PM

# BOSTON HOUSING CLASSIFIER

```
import pandas as pd
from sklearn.tree import DecisionTreeRegressor
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error

# Load the Boston Housing dataset
boston = pd.read_csv('Boston.csv')

# Separate the features (X) and target (y) variables
X = boston.drop(columns=['medv'])
y = boston['medv']


#d.fit_transform(X_train, y_train)

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

# Train the Decision Tree Regressor on the training data
d = DecisionTreeRegressor()
d.fit(X_train, y_train)

# Use the trained model to make predictions on the testing data
y_pred = d.predict(X_test)

# Calculate the mean squared error of the model's predictions
score = mean_squared_error(y_test, y_pred)
score
```

```
13.76
```

0s completed at 10:44 PM