SHIVANGEE DURGADAS KULKARNI
skulkarni10@hawk.iit.edu

1. Python script for training model using KNN and Decision Tree
   KNN:

```
from sklearn.datasets import load_iris
from sklearn import metrics

#Load the data from the module scikitlearn
iris = load_iris()

#Store features in x variable
x= iris.data

#Store response in y variable
y = iris.target

#Print the shape of x
print("Shape of x: ",x.shape)

#Print the shape of y
print("Shape of y:" ,y.shape)

#Steps of a Machine Learning model
#Step 1: Import the class Neighbors from sklearn for the KNeighborsClassifier
from sklearn.neighbors import KNeighborsClassifier

#Step 2: Instantiating the estimator
est_inst = KNeighborsClassifier(n_neighbors=5)

print("Estimator for n=5 is:" ,est_inst)

#Step 3: Fit the model with data using x and y
est_inst.fit(x,y)

#Step 4: Predict the response for a new set of values
#est_inst.predict([3,5,4,2])

new = [[3,5,4,2],[5,4,3,2]]
est_inst.predict(new)

y_predict = est_inst.predict(x)
print("Accuracy of a model when n=5: ",metrics.accuracy_score(y,y_predict))

#list(iris.target_names)
#print(type(iris.data))
#type(iris.target)

#print(iris.data)
#print(iris.target)

#print(iris.target_names)
#print(iris.feature_names)

#Using the value n=1
est_inst = KNeighborsClassifier(n_neighbors=1)
```

```
print("Estimator for n=1: ",est_inst)

#Step 3: Fit the model with data using x and y
est_inst.fit(x,y)

#Step 4: Predict the response for a new set of values
#est_inst.predict([3,5,4,2])
y_predict = est_inst.predict(x)
print("Accuracy of a model when n=1: ",metrics.accuracy_score(y,y_predict))
est_inst.predict(new)
```

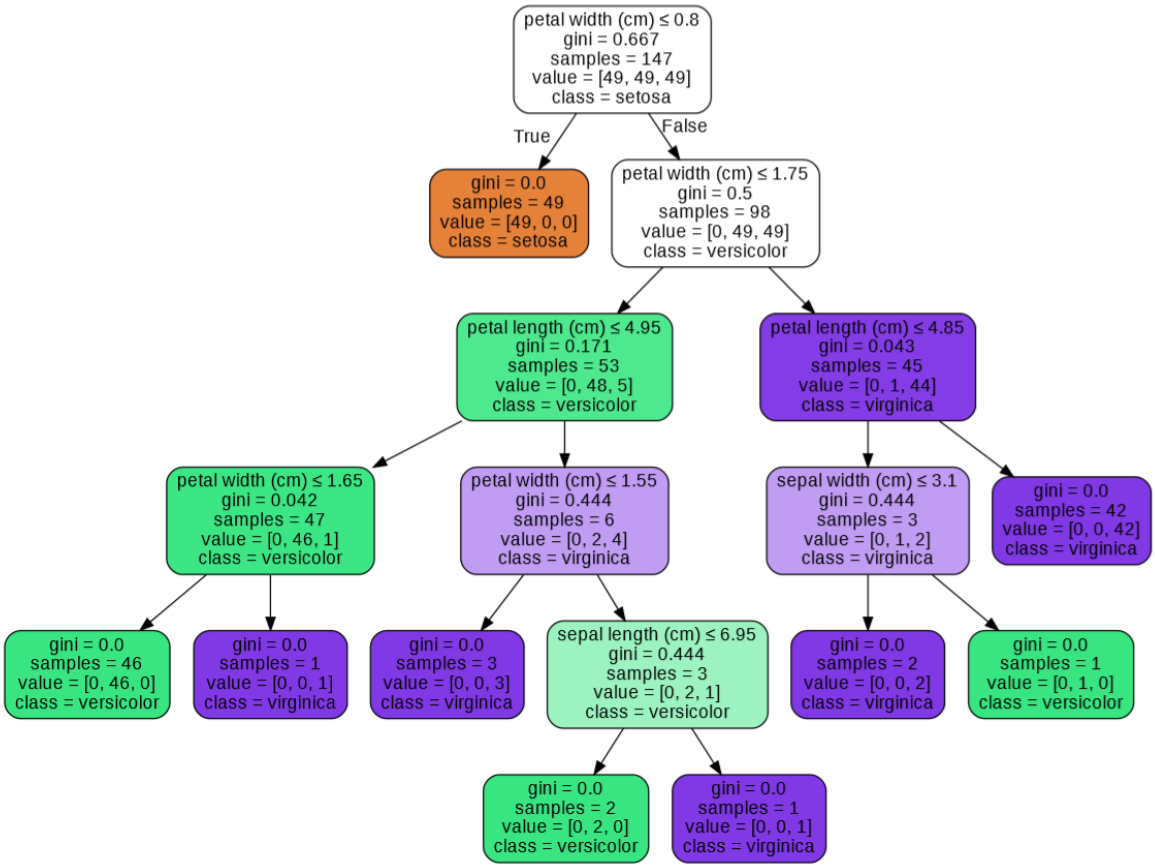
```
Shape of x:  (150, 4)
Shape of y: (150,)
Estimator for n=5 is: KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
                     metric_params=None, n_jobs=None, n_neighbors=5, p=2,
                     weights='uniform')
Accuracy of a model when n=5:  0.9666666666666667
Estimator for n=1:  KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
                     metric_params=None, n_jobs=None, n_neighbors=1, p=2,
                     weights='uniform')
Accuracy of a model when n=1:  1.0
```

Decision Tree:

```
from sklearn.datasets import load_iris
from sklearn import tree
import numpy as np
import graphviz
from IPython.display import Image
from sklearn.tree import export_graphviz
import pydotplus

#Load the data from the module scikitlearn
iris = load_iris()

#Removing the one from each class for test data
removed = [0,50,100]
new_target = np.delete(iris.target,removed)
new_data = np.delete(iris.data,removed,axis=0)

classifier = tree.DecisionTreeClassifier()
classifier = classifier.fit(new_data,new_target)
prediction = classifier.predict(iris.data[removed])

print("Original target labels:",iris.target[removed])
print("Algorithm target labels:",prediction)

from sklearn.metrics import accuracy_score
y_predict = classifier.predict(new_data)
print("Accuracy of the decision tree is : ",accuracy_score(new_target,y_predict))

#Create a dot data
graph_data = export_graphviz(classifier,out_file =
None,feature_names=iris.feature_names,class_names =
iris.target_names,filled=True,rounded=True,special_characters=True)

#graph = pydotplus.graph_from_dot_data(graph_data)
graph = pydotplus.graph_from_dot_data(graph_data)

#Display graph
```

```
graphtree = graphviz.Source(graph_data)
graphtree.render("iris")
```

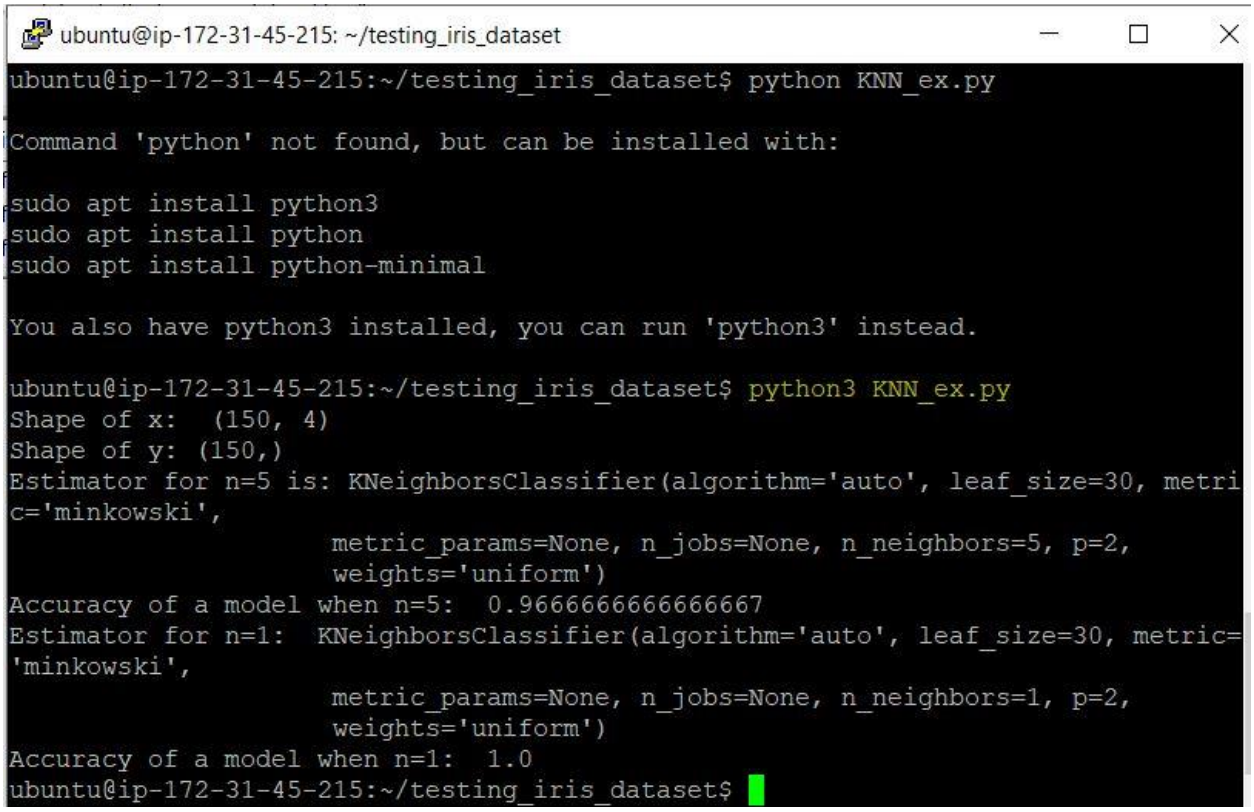
```
Original target labels: [0 1 2]
Algorithm target labels: [0 1 2]
Accuracy of the decision tree is :  1.0
'iris.pdf'
```



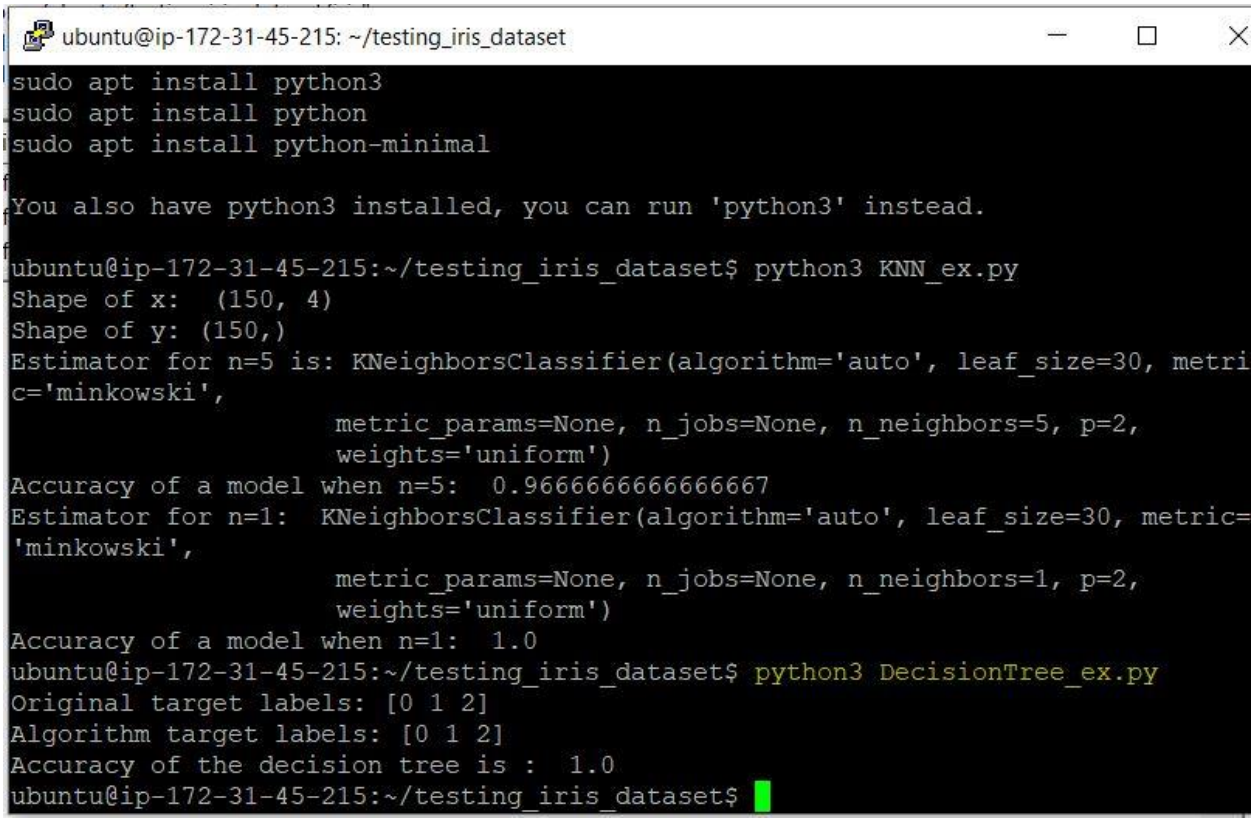
2. Created an AWS and created a user for generating the access key and secret key. Then I created an EC2 instance and connected AWS with Python by generating the private key using PuttyGen. Then I connected Python and AWS using Putty. Also then I transferred files from System to the EC2 instance using the FileZilla. The following are the outputs of the models after running in EC2:

A	B	C	D	E	F	G
User name	Password	Access key ID	Secret access key	Console login link		
testuser	7}cF[oFCWuGZ	AKIAZEYVWM64XRG3XDLL	[redacted]	https://628721346489.signin.aws.amazon.com/console		

KNN:



Decision Tree:



Code snippet for adding the output in S3 bucket:
KNN:

```
import boto3
from sklearn.datasets import load_iris
from sklearn import metrics

#Creating a file to store the output to the S3 bucket
f = open("KNN_output.txt","w+")

#Load the data from the module scikitlearn
iris = load_iris()

#Store features in x variable
x= iris.data

#Store response in y variable
y = iris.target

#Print the shape of x
print("Shape of x: ",x.shape)
f.write(str(x.shape))
f.write("\n")

#Print the shape of y
print("Shape of y:" ,y.shape)
f.write(str(y.shape))
f.write("\n")

#Steps of a Machine Learning model
#Step 1: Import the class Neighbors from sklearn for the KNeighborsClassifier
from sklearn.neighbors import KNeighborsClassifier

#Step 2: Instantiating the estimator
est_inst = KNeighborsClassifier(n_neighbors=5)

print("Estimator for n=5 is:" ,est_inst)
f.write(str(est_inst))
f.write("\n")

#Step 3: Fit the model with data using x and y
est_inst.fit(x,y)

#Step 4: Predict the response for a new set of values
#est_inst.predict([3,5,4,2])

new = [[3,5,4,2],[5,4,3,2]]
est_inst.predict(new)

y_predict = est_inst.predict(x)
print("Accuracy of a model when n=5: ",metrics.accuracy_score(y,y_predict))
f.write(str(metrics.accuracy_score(y,y_predict)))
f.write("\n")

#list(iris.target_names)
#print(type(iris.data))
#type(iris.target)

#print(iris.data)
#print(iris.target)

#print(iris.target_names)
```

```
#print(iris.feature_names)

#Using the value n=1
est_inst = KNeighborsClassifier(n_neighbors=1)

print("Estimator for n=1: ",est_inst)
f.write(str(est_inst))
f.write("\n")

#Step 3: Fit the model with data using x and y
est_inst.fit(x,y)

#Step 4: Predict the response for a new set of values
#est_inst.predict([3,5,4,2])
y_predict = est_inst.predict(x)
print("Accuracy of a model when n=1: ",metrics.accuracy_score(y,y_predict))
f.write(str(metrics.accuracy_score(y,y_predict)))
f.write("\n")

est_inst.predict(new)
f.close()

#Adding the output in S3 bucket
s3 = boto3.client("s3")
s3.upload_file("KNN_output.txt","demoiris","KNN_output.txt")
```

Output file KNN_output:

Decision Tree:

```
import boto3
from sklearn.datasets import load_iris
from sklearn import tree
import numpy as np
import graphviz
from IPython.display import Image
from sklearn.tree import export_graphviz
import pydotplus
```

```
#Creating a file to store the output to the S3 bucket
f = open("DecisionTree_output.txt","w+")

#Load the data from the module scikitlearn
iris = load_iris()

#Removing the one from each class for test data
removed = [0,50,100]
new_target = np.delete(iris.target,removed)
new_data = np.delete(iris.data,removed,axis=0)

classifier = tree.DecisionTreeClassifier()
classifier = classifier.fit(new_data,new_target)
prediction = classifier.predict(iris.data[removed])

print("Original target labels:",iris.target[removed])
f.write(str(iris.target[removed]))
f.write("\n")

print("Algorithm target labels:",prediction)
f.write(str(prediction))
f.write("\n")

from sklearn.metrics import accuracy_score
y_predict = classifier.predict(new_data)
print("Accuracy of the decision tree is : ",accuracy_score(new_target,y_predict))
f.write(str(accuracy_score(new_target,y_predict)))
f.write("\n")

#Create a dot data
graph_data = export_graphviz(classifier,out_file =
None,feature_names=iris.feature_names,class_names =
iris.target_names,filled=True,rounded=True,special_characters=True)

#graph = pydotplus.graph_from_dot_data(graph_data)
graph = pydotplus.graph_from_dot_data(graph_data)

#Display graph
graphtree = graphviz.Source(graph_data)
#graphtree.render("iris")

f.close()

#Adding the output in S3 bucket
s3 = boto3.client("s3")
s3.upload_file("DecisionTree_output.txt","demoiris","DecisionTree_output.txt")
```

Output file DecisionTree_output.txt:

KNN_ex.py | KNN_output.txt | DecisionTree_ex.py | DecisionTree_output.txt

```
[0 1 2]
[0 1 2]
1.0
```