Big Data Analytics

Lab Practical and date – Practical 2, Monday 27th July 2020

Name and Roll Number- Het Shah, 17BIT103

Practical Objective- Learning limitation of data analytics by applying Machine Learning Techniques on large amount of data. Write R/Python program to Read data set from any online website, excel file and CSV file and to perform

a) Linear regression and logistic regression on iris dataset.
b) K-means clustering.

Steps Involved-

We perform data scarping using python by reading data from different file format such as excel, csv and also performing data analytics on it

Background

Python is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python's design philosophy emphasizes code readability with its notable use of significant whitespace.

Libraries Used-

1) Pandas-pandas is a software library written for the Python programming language for data manipulation and analysis

2) SciKit-Learn-Scikit-Learn a free software machine learning library for the Python programming language. It features various classification and regression and clustering algorithms including support vector machines, random forests, gradient boosting, *k*-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.
3) Mathplotlib- Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK+.

Data-set

1) Iris DataSet-The data set consists of 50 samples from each of three species of *Iris* (*Iris setosa*, *Irisvirginica* and *Iris versicolor*). Features were measured from each sample: the length and the width of sepals and petals, in centimeters. Based on the combination of these four features, Fisher developed a linear discriminant model to distinguish the species from each other.

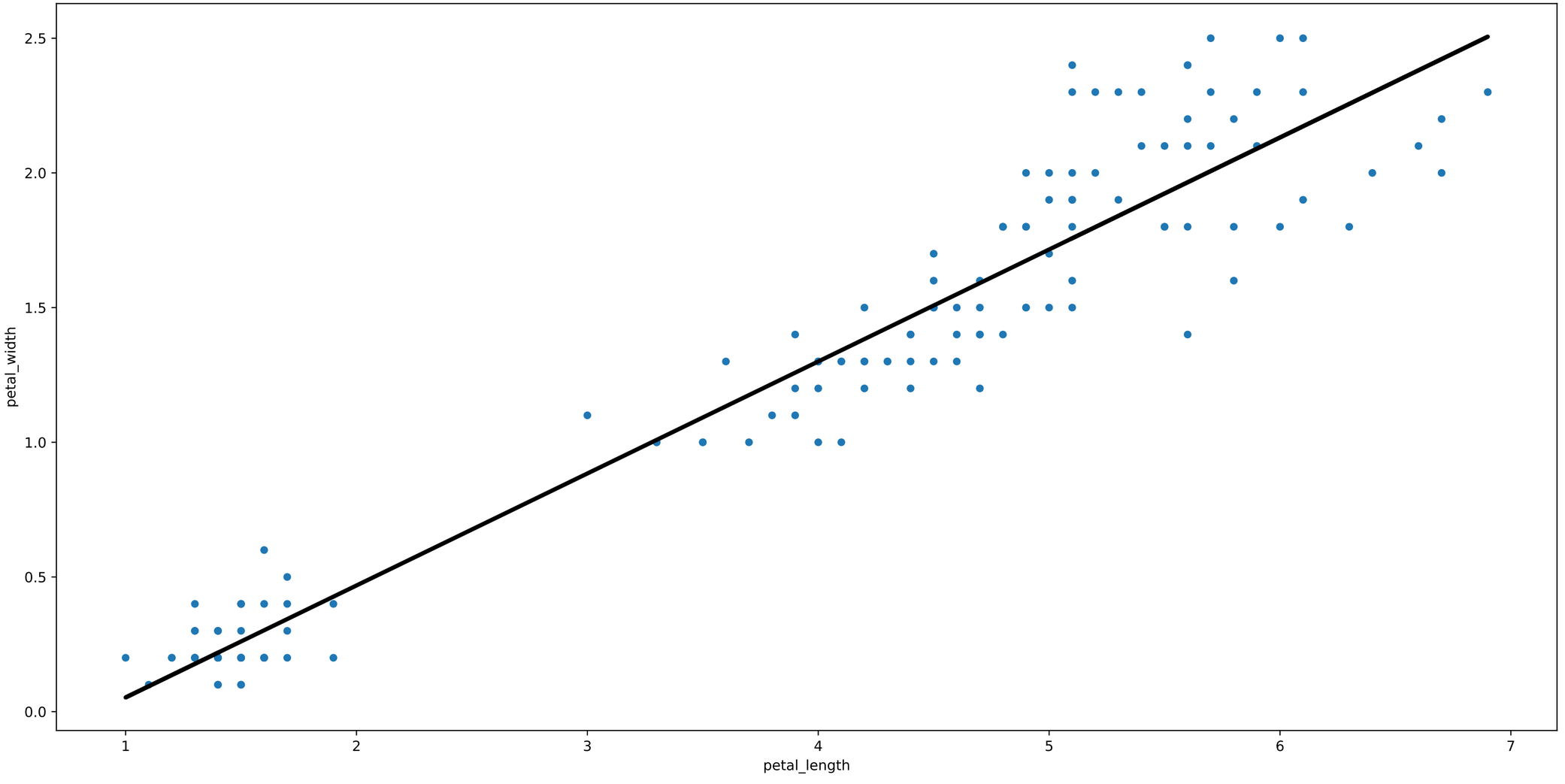
Algorithms

1) Linear regression- linear regression is a linear approach to modeling the relationship between a scalar response and one or more explanatory variables.
2) K-Means-k-means clustering is a method of vector quantization, originally from signal processing, that aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster.
3) Logistic Regression-In statistics, the logistic model is used to model the probability of a certain class or event existing such as pass/fail, win/lose, alive/dead or healthy/sick. This can be extended to model several classes of events such as determining whether an image contains a cat, dog, lion, etc.

Conclusion-

In this Experiment we applied various machine learning algorithms on the given IRIS dataset and plotted the results using the mathplotlib.

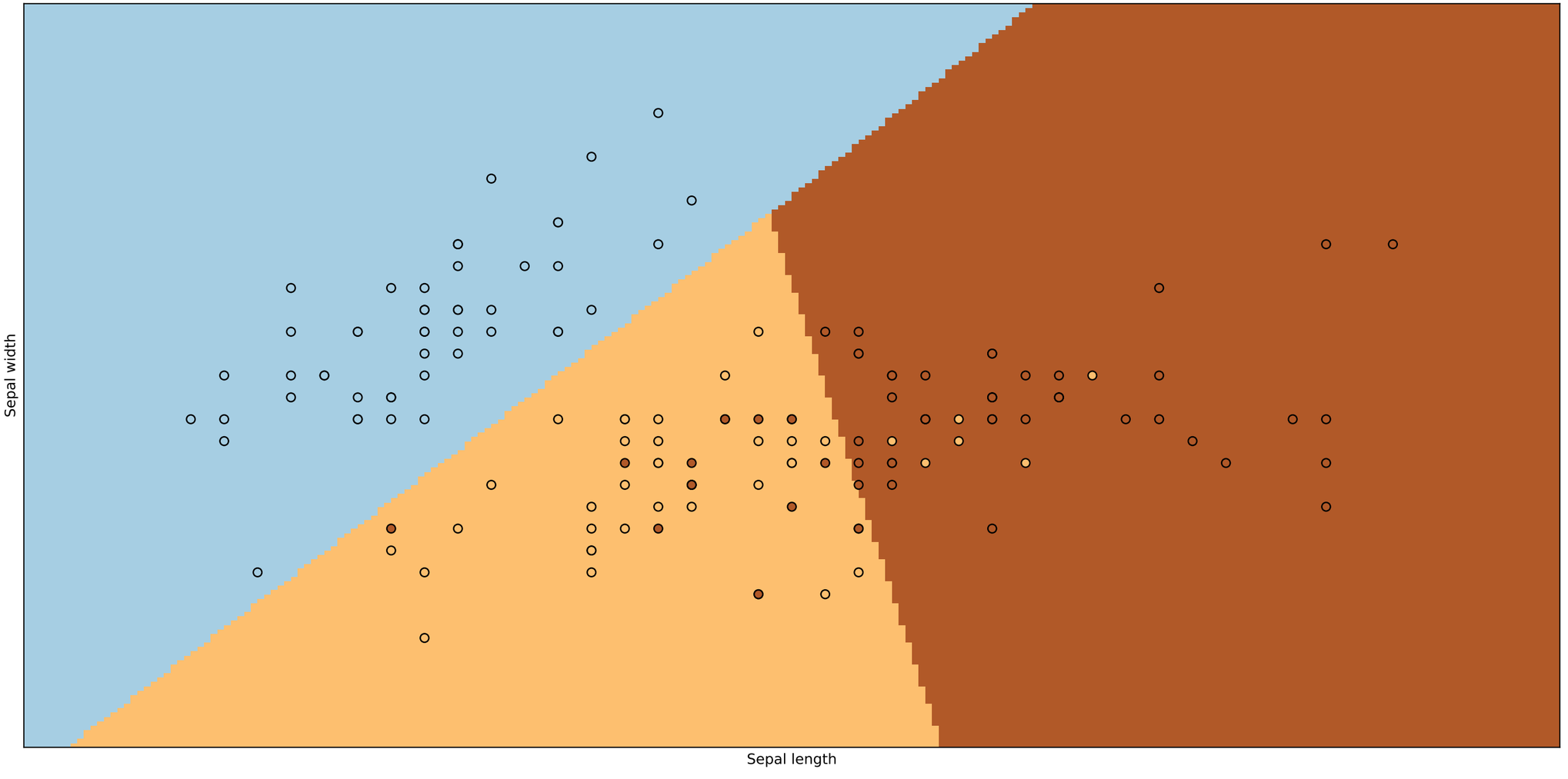
```
# -*- coding: utf-8 -*-
"""
Created on Tue Aug  4 12:47:37 2020

@author: HETSHAH
"""

from sklearn import datasets, linear_model
import pandas as pd
import matplotlib.pyplot as plt
import seaborn.apionly as sns

iris = sns.load_dataset('iris')
fit_data = iris[["petal_length", "petal_width"]].values
x_data = fit_data[:,0].reshape(-1,1)
y_data = fit_data[:,1].reshape(-1,1)

# Create linear regression object
regr = linear_model.LinearRegression()
# once the data is reshaped, running the fit is simple
regr.fit(x_data, y_data)

# we can then plot the data and out fit
axes = iris.plot(x="petal_length", y="petal_width", kind="
scatter")
plt.plot(x_data, regr.predict(x_data), color='black',
linewidth=3)
plt.show()
```


petal_width
petal_length
0.0
0.5
1.0
1.5
2.0
2.5
1
2
3
4
5
6
7

File - E:\Desktop\bda 2 2.py

```
# -*- coding: utf-8 -*-
"""
Created on Tue Aug  4 12:43:16 2020

@author: HETSHAH
"""

import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LogisticRegression
from sklearn import datasets

iris = datasets.load_iris()
X = iris.data[:, :2]  # we only take the first two features.
Y = iris.target

logreg = LogisticRegression(C=1e5)

# Create an instance of Logistic Regression Classifier and fit the data.
logreg.fit(X, Y)

# Plot the decision boundary. For that, we will assign a color to each
# point in the mesh [x_min, x_max]x[y_min, y_max].
x_min, x_max = X[:, 0].min() - .5, X[:, 0].max() + .5
y_min, y_max = X[:, 1].min() - .5, X[:, 1].max() + .5
h = .02  # step size in the mesh
xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min, y_max, h))
Z = logreg.predict(np.c_[xx.ravel(), yy.ravel()])

# Put the result into a color plot
Z = Z.reshape(xx.shape)
plt.figure(1, figsize=(4, 3))
plt.pcolormesh(xx, yy, Z, cmap=plt.cm.Paired)

# Plot also the training points
plt.scatter(X[:, 0], X[:, 1], c=Y, edgecolors='k', cmap=plt.cm.Paired)
plt.xlabel('Sepal length')
plt.ylabel('Sepal width')

plt.xlim(xx.min(), xx.max())
```

```
41 plt.ylim(yy.min(), yy.max())
42 plt.xticks(())
43 plt.yticks(())
44
45 plt.show()
```

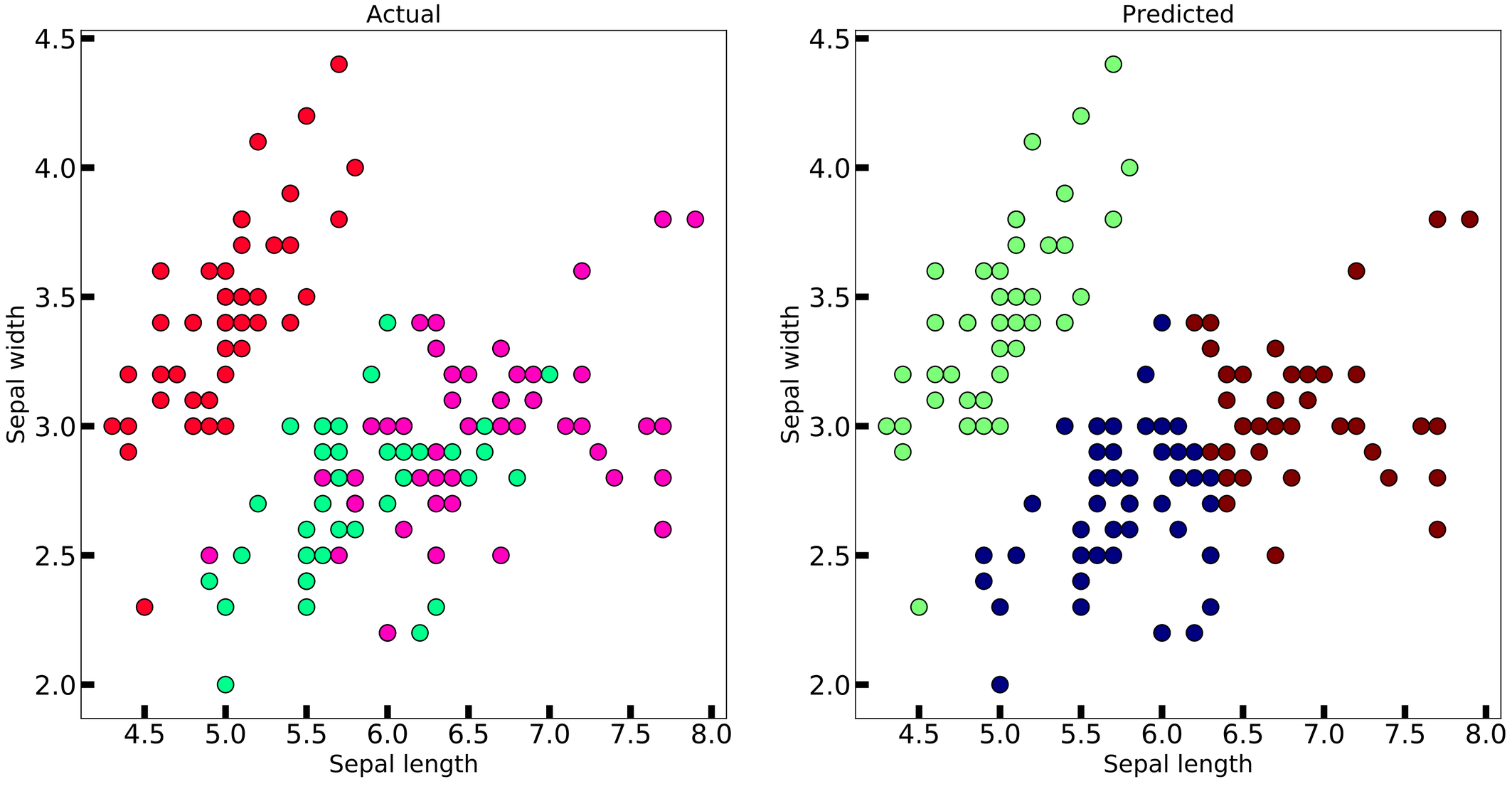

Sepal width
Sepal length

File - E:\Desktop\bda 2.py

```
# -*- coding: utf-8 -*-
"""
Created on Tue Aug  4 10:18:40 2020

@author: HETSHAH
"""


import pandas as pd;
import seaborn as sns
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.datasets import load_iris
from sklearn.linear_model import LinearRegression;
from sklearn.metrics import mean_absolute_error
from sklearn.metrics import mean_squared_error


print("start")
iris = load_iris()

iris_df = pd.DataFrame(data=iris.data, columns=iris.feature_names)
target_df= pd.DataFrame(data=iris.target, columns=['species'])

def converter(specie):
    if(specie == 0):
        return 'setosa'
    elif specie == 1:
        return 'versicolor'
    else:
        return 'virginica'

target_df['species']=target_df['species'].apply(converter)

iris_df = pd.concat([iris_df,target_df],axis=1)

iris_df.info();

sns.pairplot(iris_df, hue= 'species')
iris_df.drop('species', axis= 1, inplace= True)
target_df = pd.DataFrame(columns= ['species'], data= iris.target)
iris_df = pd.concat([iris_df, target_df], axis= 1)
```

```
43
44
45 X= iris_df.drop(labels= 'sepal length (cm)', axis= 1)
46 y= iris_df['sepal length (cm)']
47
48
49 X_train, X_test, y_train, y_test = train_test_split(X, y,
   test_size= 0.33, random_state= 101)
50
51 lr = LinearRegression()
52
53 #train
54 lr.fit(X_train, y_train)
55
56 #predict
57 lr.predict(X_test)
58 pred = lr.predict(X_test)
59
60
61 print('Mean Absolute Error:', mean_absolute_error(y_test,
   pred))
62 print('Mean Squared Error:', mean_squared_error(y_test,
   pred))
63 print('Mean Root Squared Error:', np.sqrt(
   mean_squared_error(y_test, pred)))
64 print("stop")
65
66
67
68
69
70 import matplotlib.pyplot as plt
71 import seaborn as sns
72 from sklearn.linear_model import LogisticRegression
73 from sklearn.metrics import classification_report
74 from sklearn.metrics import accuracy_score
75 from sklearn.model_selection import train_test_split
76 import pandas.util.testing as tm
77
78
79 data = sns.load_dataset("iris")
80 data.head()
81 X = data.iloc[:, :-1]
82 y = data.iloc[:, -1]
83 plt.xlabel('Features')
```

```
 84 plt.ylabel('Species')
 85
 86 pltX = data.loc[:, 'sepal_length']
 87 pltY = data.loc[:,'species']
 88 plt.scatter(pltX, pltY, color='blue', label='sepal_length
    ')
 89
 90 pltX = data.loc[:, 'sepal_width']
 91 pltY = data.loc[:,'species']
 92 plt.scatter(pltX, pltY, color='green', label='sepal_width
    ')
 93
 94 pltX = data.loc[:, 'petal_length']
 95 pltY = data.loc[:,'species']
 96 plt.scatter(pltX, pltY, color='red', label='petal_length'
    )
 97
 98 pltX = data.loc[:, 'petal_width']
 99 pltY = data.loc[:,'species']
100 plt.scatter(pltX, pltY, color='black', label='petal_width
    ')
101
102 plt.legend(loc=4, prop={'size':8})
103 plt.show()
104
105 #Split the data into 80% training and 20% testing
106 x_train, x_test, y_train, y_test = train_test_split(X, y,
     test_size=0.2, random_state=42)
107
108 #Train the model
109 model = LogisticRegression()
110 model.fit(x_train, y_train) #Training the model
111
112 #Test the model
113 predictions = model.predict(x_test)
114 print(predictions)# printing predictions
115
116 print()# Printing new line
117
118 #Check precision, recall, f1-score
119 print( classification_report(y_test, predictions) )
120
121 print( accuracy_score(y_test, predictions))
```


Actual
Predicted
Sepal width
Sepal length
4.5
4.0
3.5
3.0
2.5
2.0
4.5
5.0
5.5
6.0
6.5
7.0
7.5
8.0