İbrahim Talha ASAN
COE-64170019

Assignment 5 Report

Problem 1:

Definition:

The purpose of Assignment 5 to find the best linear regression model by using the iris data set by selecting the optimum features. In addition, the result of the linear regression model should be made more descriptive with various graphs and evaluations.

Program Code:

```
import pandas as pd
import matplotlib.pyplot as pl
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn import preprocessing
from sklearn import metrics

fileURL = 'C:\\Users\\pc\\.spyder-py3\\openCV\\iris.data'

iris = pd.read_csv (fileURL , names=[ 'Sepal Length' , 'Sepal Width' ,
                            'Petal Length' ,  'Petal Width' ,
                            'Species' ] , header=None )
iris = iris.dropna()

def pairs(data):
   i = 1
   # Divide columns into features and class
   features = list(data.columns)
   classes = features[-1] # create class column
   del features[-1] # delete class column from feature vector
   # Generate an nxn subplot figure, where n is the number of features
   figure = pl.figure(figsize=(5*(len(data.columns)-1), 4*(len(data.columns)-1)))
   for col1 in data[features]:
      for col2 in data[features]:
         ax = pl.subplot(len(data.columns)-1, len(data.columns)-1, i)
         if col1 == col2:
            ax.text(2.5, 4.5, col1, style='normal', fontsize=20)
            ax.axis([0, 10, 0, 10])
            pl.xticks([]), pl.yticks([])
         else:
            for name in data[classes]:
               cond = data[classes] == name
               ax.plot(data[col2][cond], data[col1][cond], linestyle='none', marker='o', label=name)
            #t = plt.title(name)
         i += 1
   pl.show()

#pairs(iris)
```

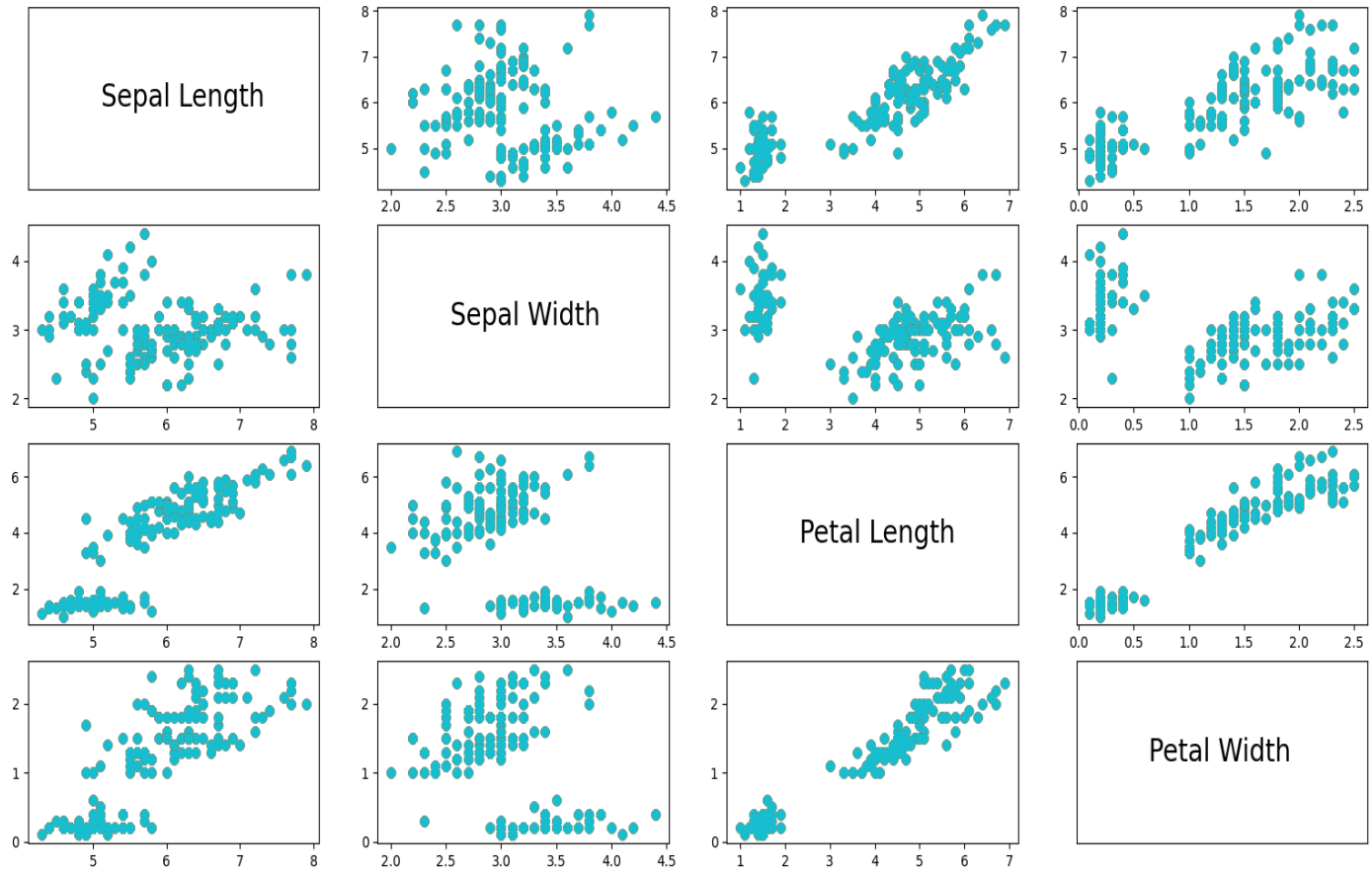
İbrahim Talha ASAN
COE-64170019

```
    def showingCorrelation(iris):

        pl.xlabel('Features')
        pl.ylabel('Species')

        plX = iris.loc[:,'Sepal Length']
        plY = iris.loc[:,'Species']
        pl.scatter(plX,plY,color='blue',label = 'Sepal Length')

        plX = iris.loc[:,'Sepal Width']
        plY = iris.loc[:,'Species']
        pl.scatter(plX,plY,color='green',label = 'Sepal Width')

        plX = iris.loc[:,'Petal Length']
        plY = iris.loc[:,'Species']
        pl.scatter(plX,plY,color='red',label = 'Petal Length')

        plX = iris.loc[:,'Petal Width']
        plY = iris.loc[:,'Species']
        pl.scatter(plX,plY,color='black',label='Petal Width')

        pl.legend(loc=4,prop={'size':8})
        pl.show()

    #showingCorrelation(iris)

    def applyLinearRegressionWithSepalLengthAndPetalLength(iris):
        sepal_length=iris.loc[:,'Sepal Length']
        pedal_length=iris.loc[:,'Petal Length']

        label_Encoder=preprocessing.LabelEncoder()

        iris_X = np.column_stack((sepal_length,pedal_length))
        iris_y = label_Encoder.fit_transform(iris.iloc[:,-1])

        iris_X_train,iris_X_test,iris_y_train,iris_y_test=train_test_split(iris_X,iris_y,test_size=0.2,random_state=0)

        regr = LinearRegression()

        regr.fit(iris_X_train,iris_y_train)

        y_pred = regr.predict(iris_X_test)

        df = pd.DataFrame({'Actual': iris_y_test.flatten(), 'Predicted': y_pred.flatten()})

        print ("Coefficients : \n" , regr.coef_)
        print ( "Residual sum of squares : %.2f" %
        np .mean ((regr.predict ( iris_X_test ) - iris_y_test)** 2))
        print ( "Variance score : %.2f" % regr.score ( iris_X_test , iris_y_test))
        print()
        print('Mean Absolute Error:', metrics.mean_absolute_error(iris_y_test, y_pred))
```

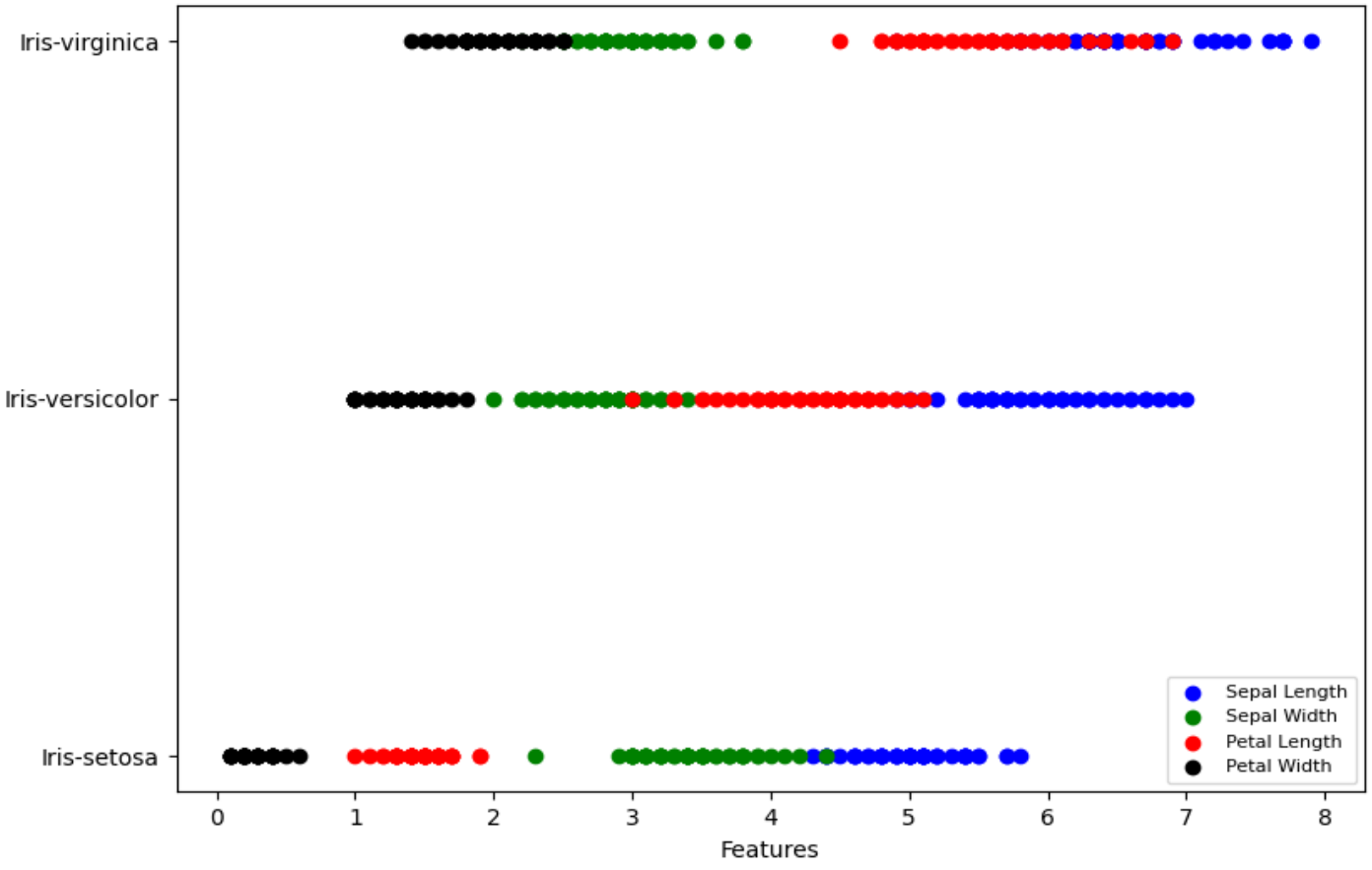
```
        print('Mean Squared Error:', metrics.mean_squared_error(iris_y_test, y_pred))
        print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(iris_y_test, y_pred)))
        print()
        print(df)

        df1 = df.head(25)
        df1.plot(kind='bar',figsize=(16,10))
        pl.grid(which='major', linestyle='-', linewidth='0.5', color='green')
        pl.grid(which='minor', linestyle=':', linewidth='0.5', color='black')
        pl.title('Linear Regression With Sepal Length and Petal Length')
        pl.show()

    #applyLinearRegressionWithSepalLengthAndPetalLength(iris)

    def applyLinearRegressionWithSepalWidthAndPetalLength(iris):
        sepal_width=iris.loc[:,'Sepal Width']
        pedal_length=iris.loc[:,'Petal Length']

        label_Encoder=preprocessing.LabelEncoder()

        iris_X = np.column_stack((sepal_width,pedal_length))
        iris_y = label_Encoder.fit_transform(iris.iloc[:,-1])


        iris_X_train,iris_X_test,iris_y_train,iris_y_test=train_test_split(iris_X,iris_y,test_size=0.2,random_state=0)

        regr = LinearRegression()

        regr.fit(iris_X_train,iris_y_train)

        y_pred = regr.predict(iris_X_test)

        df = pd.DataFrame({'Actual': iris_y_test.flatten(), 'Predicted': y_pred.flatten()})
        print ("Coefficients : \n" , regr.coef_)
        print ( "Residual sum of squares : %.2f" %
        np .mean ((regr.predict ( iris_X_test ) - iris_y_test)** 2))
        print ( "Variance score : %.2f" % regr.score ( iris_X_test , iris_y_test))
        print()
        print('Mean Absolute Error:', metrics.mean_absolute_error(iris_y_test, y_pred))
        print('Mean Squared Error:', metrics.mean_squared_error(iris_y_test, y_pred))
        print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(iris_y_test, y_pred)))
        print()
        print(df)
        df1 = df.head(25)
        df1.plot(kind='bar',figsize=(16,10))
        pl.grid(which='major', linestyle='-', linewidth='0.5', color='green')
        pl.grid(which='minor', linestyle=':', linewidth='0.5', color='black')
        pl.title('Linear Regression With Sepal Width and Petal Length')
        pl.show()

    applyLinearRegressionWithSepalWidthAndPetalLength(iris)
```

İbrahim Talha ASAN
COE-64170019

Program Outputs:



pairwise scatterplot of the dataset features



correlation between dataset features

İbrahim Talha ASAN
COE-64170019

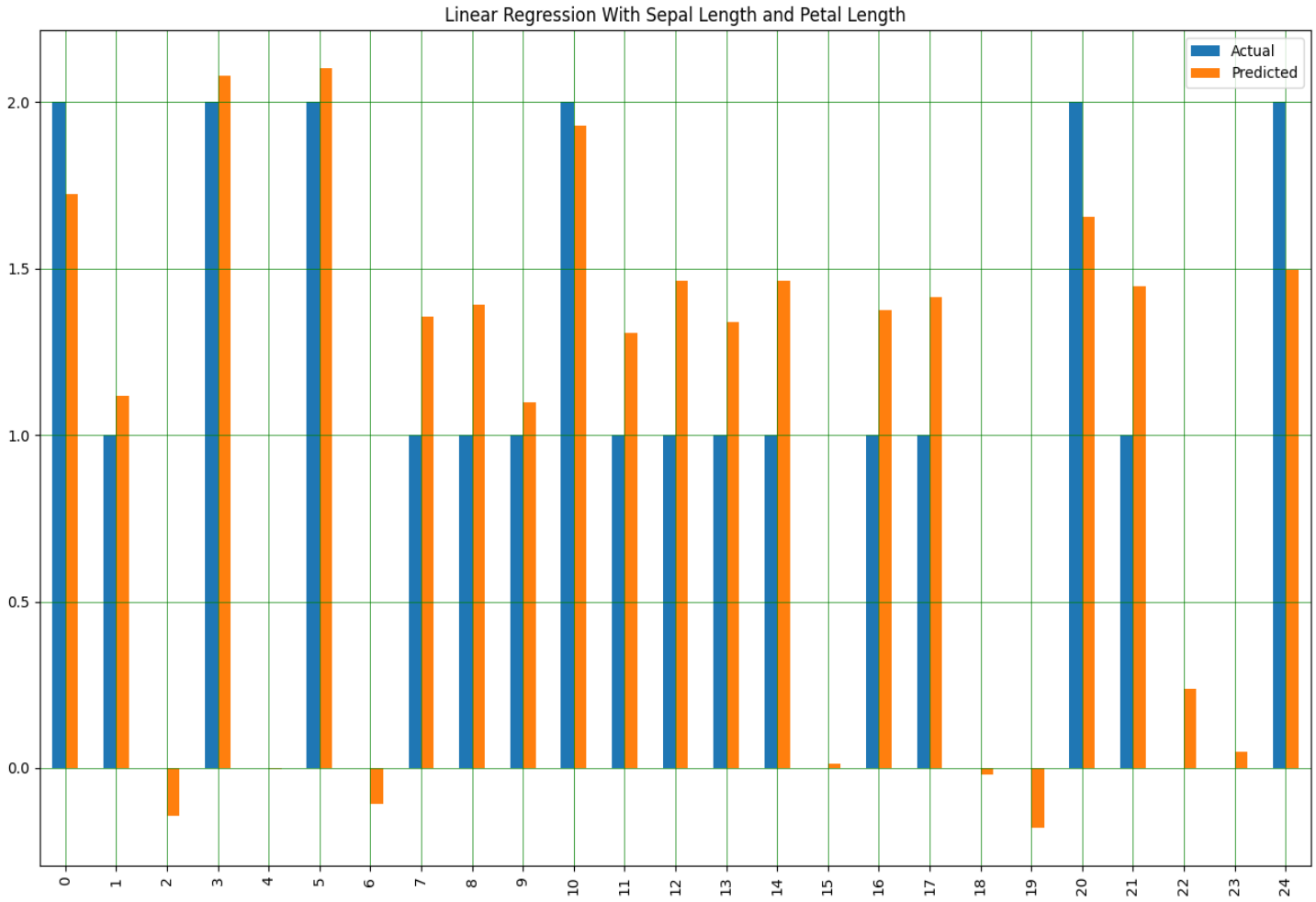
Linear Regression With Sepal Length and Petal Length

Coefficients :
[-0.1769398 0.51912882]
Residual sum of squares : 0.07
Variance score : 0.87

Mean Absolute Error: 0.22116226250993695
Mean Squared Error: 0.07263420492762888
Root Mean Squared Error: 0.2695073374281837

	Actual	Predicted
0	2	1.722617
1	1	1.116188
2	0	-0.145077
3	2	2.080162
4	0	-0.004695
5	2	2.101363
6	0	-0.108520
7	1	1.355720
8	1	1.389939
9	1	1.098494
10	2	1.929100
11	1	1.304976
12	1	1.461884
13	1	1.339195
14	1	1.461884
15	0	0.012999
16	1	1.375752
17	1	1.412309
18	0	-0.021220
19	0	-0.179296
20	2	1.654179
21	1	1.446528
22	0	0.238345
23	0	0.049556
24	2	1.496103
25	0	-0.193483
26	0	0.185263
27	1	1.236538
28	1	0.929737
29	0	0.047218

İbrahim Talha ASAN
COE-64170019



Linear Regression With Sepal Width and Petal Length

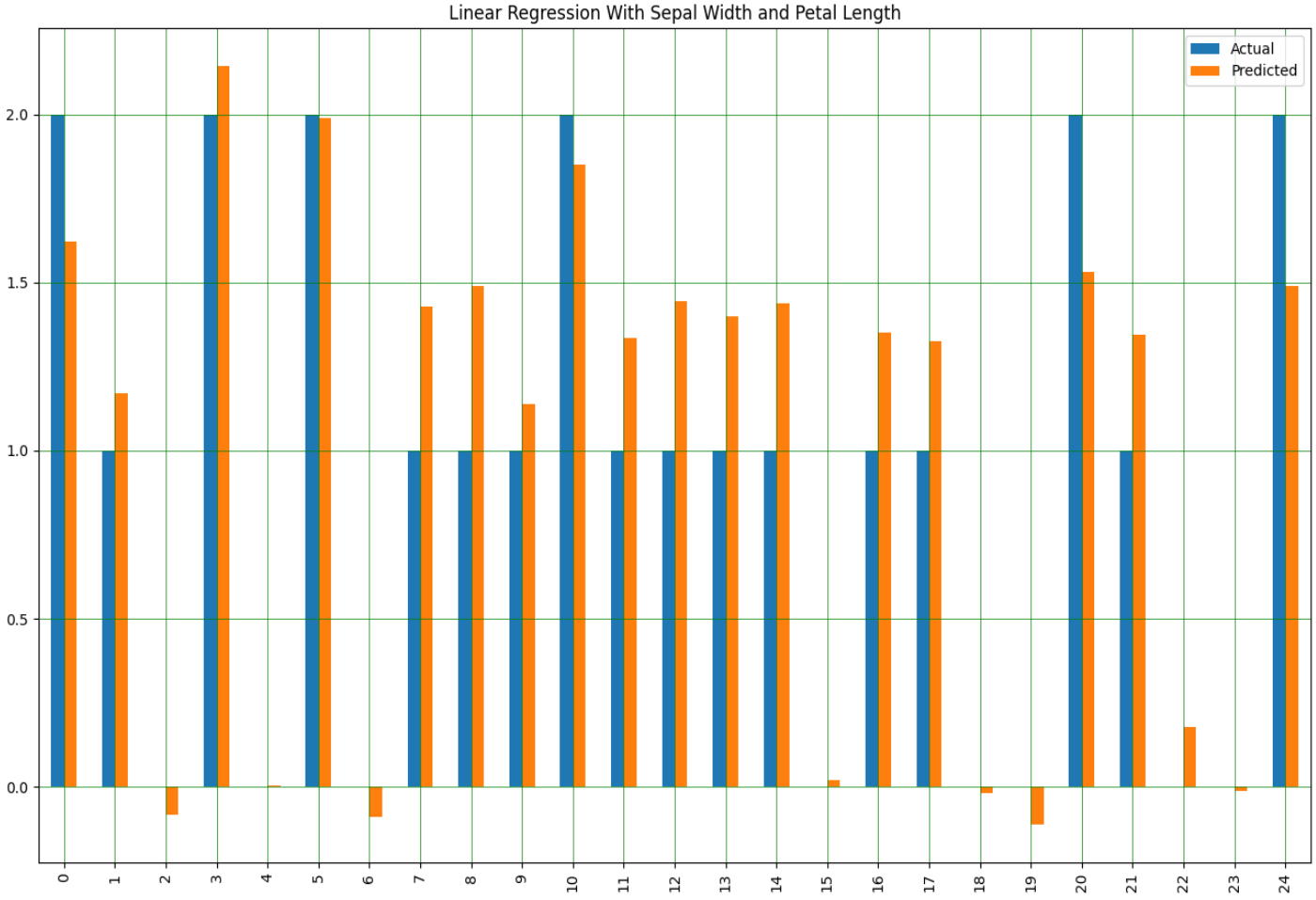
Coefficients :
[-0.0546932 0.44029918]
Residual sum of squares : 0.08
Variance score : 0.85

Mean Absolute Error: 0.22950250618210777
Mean Squared Error: 0.07924165598086128
Root Mean Squared Error: 0.28149894490186156

	Actual	Predicted
0	2	1.620940
1	1	1.169427
2	0	-0.084738
3	2	2.143830
4	0	0.003047
5	2	1.989862
6	0	-0.090482
7	1	1.428412
8	1	1.488850
9	1	1.136611
10	2	1.852028
11	1	1.334883

İbrahim Talha ASAN
COE-64170019

12	1	1.444820
13	1	1.400790
14	1	1.439351
15	0	0.019455
16	1	1.351291
17	1	1.323669
18	0	-0.019106
19	0	-0.112360
20	2	1.532880
21	1	1.345822
22	0	0.179167
23	0	-0.013636
24	2	1.488850
25	0	-0.228041
26	0	0.157289
27	1	1.263231
28	1	0.855748
29	0	0.047077



İbrahim Talha ASAN
COE-64170019

Discussions:

I plot the correlation plot of the properties in the Iris dataset, and based on this graph, I determined the property pairs that were most related to each other (Sepal Length and Petal Length and Sepal Width and Petal Length). Later on linear regression was applied on these feature pairs and the results of the model were added to the report. As a result, the results of the Sepal Length and Petal Length feature pairs were better.