Coursework_O2_solved

September 8, 2019

1 Coursework Output 2

Instructions: In this coursework, you will show your domain of the data related Python skills required for business analytics. To do so, you will use packages such as *pandas* and *matplotlib*.

1.1 Importing and Visualising a Business Case in Python

Each student will be assigned a different dataset in a different format. The goal is to import the dataset into Python so that you can wrangle and visualise the data in better ways. This will allow you to get your own conclusions and start building up knowledge regarding on how you could potentially learn from data to predict or classify future instances.

You must create a program which allows you to select the following options: 1. Import your dataset into Python as a Pandas data frame. 2. Query and print an instance of the dataset (by row number or by row name). 3. Create a "reduced" dataset (with less columns) by indicating a list of columns to bring upon this newly created dataset. 4. Randomly split the original or the reduced dataset into two substes called *training* and *testing* according to a ratio specified by the user. 5. Visualise a dataset by means of a scatterplot which relates two variables/columns specified by the user. The plot has to show the x and y axis labels and use the *target/class* column (i.e. the last one) as the colour variable. 6. Exit the program.

1.2 Additional Considerations

- The program has to check that every input option added by the user is valid.
- No option can be executed until option 1 is executed first.
- Whenever a dataset or subdataset is imported or created, print it for the user to visually inspect it.
- In option 2, the row to query can be specified either using the row number or the row name.
- In option 3, the list of columns has to be specified by column name.
- Option 5 can receive as input either the original dataset or the training/testing ones (if these have been already created).

1.3 Submission Instructions

- Once that you have finished your program, run all cells and run the main program cell using the sequence of options 0(wrong option)-2(has to fail)-1-2(by row index)-2(by row name)-3-4-5-6 (option 0 is purposely created to verify that your program can handle the error).
- Then, without clearing the kernel, generate a html **OR** pdf file from the Jupyter notebook.

- Name both the Jupyter notebook and the html/pdf file with your id number and submit them to the corresponding Moodle's dropbox before **12th December, 2019**.

In [1]:
```
## Use this cell to import all necessary packages
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.model_selection import train_test_split
import sys
```

In [2]:
```
## Use this cell to define the function corresponding to OPTION 1
def option1():
    '''With this function you import the dataset.'''
    dataset = pd.read_csv('iris.csv')
    print('The dataset has been imported!')
    print(dataset)
    return dataset

# dataset = option1()
```

In [3]:
```
## Use this cell to define the function corresponding to OPTION 2

def option2(dataset):
    '''This function queries and prints an instance of the dataset.'''
    ans = True
    while ans:
        print('Select 1 to query by row number or 2 to query by row name:')
        ans = input('>> ')
        if ans=='1' or ans =='2':
            if ans=='1':
                ans2 = True
                while ans2:
                    print('Select the row number to query (indexes start in 0):')
                    ans2 = input('>> ')
                    if int(ans2)<=len(dataset)-1:
                        print(dataset.iloc[int(ans2)])
                        ans2 = False
                    else:
                        print('The index number is unvalid. Please try again.')
            else:
                ans2 = True
                while ans2:
                    print('Select the row index to query:')
                    ans2 = input('>> ')
                    if ans2 in dataset.index:
                        print(dataset.loc[ans2])
                        ans2 = False
                    else:
```

```
                            print('The index name is unvalid. Please try again.')
                    ans = False
                else:
                    print('Wrong option. Please try again.')
            return

        #option2(dataset)
```

In [4]:
```
## Use this cell to define the function corresponding to OPTION 3

def option3(dataset):
    '''This function creates a new dataset by indicating which columns to include.'''
    columnstoinclude=[]
    ans = True
    while ans:
        print('Indicate a column to include:')
        print(list(dataset.columns))
        ans = input('>> ')
        if ans in dataset.columns and ans not in columnstoinclude:
            columnstoinclude.append(ans)
            print('Do you want to indicate another column? (Y/N)')
            ans2 = True
            while ans2:
                ans2 = input('>> ')
                if ans2.lower()=='y':
                    ans2=False
                elif ans2.lower()=='n':
                    ans2=False
                    ans=False
                else:
                    print('Wrong option. Please try again.')
        else:
            print('The column indicated does not exit or has already been indicated. Pl
    if columnstoinclude:
        dataset_reduced = dataset[columnstoinclude]
        print('Showing the reduced dataset...')
        print(dataset_reduced)
    return dataset_reduced

#dataset_reduced = option3(dataset)
```

In [5]:
```
## Use this cell to define the function corresponding to OPTION 4

def option4(dataset, dataset_reduced):
    '''This function randomly splits the original or the reduced dataset into train an
    print('Select 1 to use the original dataset or 2 to use the reduced dataset:')
    ans = True
    while ans:
```

```
            select = input('>> ')
            if select == '1':
                ans = False
            elif select == '2':
                if type(dataset_reduced) is not list and len(dataset_reduced)!=0:
                    ans = False
                else:
                    print('There is no reduced dataset. Please select option 1.')
            else:
                print('Wrong option. Please try again.')
        ans = True
        while ans:
            print('Indicate the percentage of the dataset to assign as test data (between (
            ans = input('>> ')
            if 0<float(ans)<1:
                if select == '1':
                    train, test = train_test_split(dataset, test_size=float(ans))
                else:
                    train, test = train_test_split(dataset_reduced, test_size=float(ans))
                print('Showing training data...')
                print(train)
                print('Showing testing data...')
                print(test)
                ans = False
            else:
                print('Wrong input. Please try again.')
        return train, test

    # train, test = option4(dataset, dataset_reduced)
```

In [6]:
```
    ## Use this cell to define the function corresponding to OPTION 5

    def option5(dataset,train,test):
        '''This function visualises the dataset using a scatterplot.'''
        print('Select 1 to use the original dataset, 2 to use the training dataset or 3 to
        ans = True
        while ans:
            select = input('>> ')
            if select == '1':
                vis = dataset
                ans = False
            elif select == '2':
                if type(train) is not list and len(train)!=0:
                    ans = False
                    vis = train
                else:
                    print('There is no training/testing dataset. Please select option 1.')
            elif select == '3':
```

```
                if type(test) is not list and len(test)!=0:
                    ans = False
                    vis = test
                else:
                    print('There is no training/testing dataset. Please select option 1.')
            else:
                print('Wrong option. Please try again.')
        ans = True
        while ans:
            print('Select the variable to use as x axis:')
            print(list(dataset.columns))
            ans = input('>> ')
            if ans in dataset.columns:
                x_axis = ans
                ans = False
            else:
                print('The column does not exist. Please try again.')
        ans = True
        while ans:
            print('Select the variable to use as y axis:')
            print(list(dataset.columns))
            ans = input('>> ')
            if ans in dataset.columns:
                y_axis = ans
                ans = False
            else:
                print('The column does not exist. Please try again.')

        colours = np.array(vis['variety'])
        colours=np.where(colours=='Setosa', 0, colours)
        colours=np.where(colours=='Versicolor', 1, colours)
        colours=np.where(colours=='Virginica', 2, colours)
        plt.scatter(vis[x_axis], vis[y_axis], c=colours)
        plt.xlabel(x_axis)
        plt.ylabel(y_axis)
        plt.show()

        return

    # option5(dataset, train, test)
```

In [7]:
```
## Use this cell to create the "main" part of your program
print('Welcome to Carlos Moreno-Garcia 1813072 business case.')
dataset = []
dataset_reduced = []
train = []
test = []
prev_options=[]
```

```
        valid_options = ['1','2','3','4','5','6']
        ans = True
        while ans:
            print('Select and option')
            ans = input('>> ')
            if ans in valid_options:
                if ans !='1' and '1' not in prev_options:
                    print('The first option to be selected shall be 1. Please try again')
                    ans = True
                elif ans == '1':
                    prev_options.append(ans)
                    dataset = option1()
                    ans = True
                elif ans == '2':
                    prev_options.append(ans)
                    option2(dataset)
                    ans = True
                elif ans == '3':
                    prev_options.append(ans)
                    dataset_reduced = option3(dataset)
                    ans = True
                elif ans == '4':
                    prev_options.append(ans)
                    train, test = option4(dataset, dataset_reduced)
                    ans = True
                elif ans == '5':
                    prev_options.append(ans)
                    option5(dataset, train, test)
                    ans = True
                else:
                    sys.exit()
            else:
                print('Error, try again')
                ans = True
```

```
Welcome to Carlos Moreno-Garcia 1813072 business case.
Select and option
>> 0
Error, try again
Select and option
>> 2
The first option to be selected shall be 1. Please try again
Select and option
>> 1
The dataset has been imported!
     sepal.length  sepal.width  petal.length  petal.width    variety
0             5.1          3.5           1.4          0.2     Setosa
1             4.9          3.0           1.4          0.2     Setosa
```

2	4.7	3.2	1.3	0.2	Setosa
3	4.6	3.1	1.5	0.2	Setosa
4	5.0	3.6	1.4	0.2	Setosa
5	5.4	3.9	1.7	0.4	Setosa
6	4.6	3.4	1.4	0.3	Setosa
7	5.0	3.4	1.5	0.2	Setosa
8	4.4	2.9	1.4	0.2	Setosa
9	4.9	3.1	1.5	0.1	Setosa
10	5.4	3.7	1.5	0.2	Setosa
11	4.8	3.4	1.6	0.2	Setosa
12	4.8	3.0	1.4	0.1	Setosa
13	4.3	3.0	1.1	0.1	Setosa
14	5.8	4.0	1.2	0.2	Setosa
15	5.7	4.4	1.5	0.4	Setosa
16	5.4	3.9	1.3	0.4	Setosa
17	5.1	3.5	1.4	0.3	Setosa
18	5.7	3.8	1.7	0.3	Setosa
19	5.1	3.8	1.5	0.3	Setosa
20	5.4	3.4	1.7	0.2	Setosa
21	5.1	3.7	1.5	0.4	Setosa
22	4.6	3.6	1.0	0.2	Setosa
23	5.1	3.3	1.7	0.5	Setosa
24	4.8	3.4	1.9	0.2	Setosa
25	5.0	3.0	1.6	0.2	Setosa
26	5.0	3.4	1.6	0.4	Setosa
27	5.2	3.5	1.5	0.2	Setosa
28	5.2	3.4	1.4	0.2	Setosa
29	4.7	3.2	1.6	0.2	Setosa
..	...	...	...	...	...
120	6.9	3.2	5.7	2.3	Virginica
121	5.6	2.8	4.9	2.0	Virginica
122	7.7	2.8	6.7	2.0	Virginica
123	6.3	2.7	4.9	1.8	Virginica
124	6.7	3.3	5.7	2.1	Virginica
125	7.2	3.2	6.0	1.8	Virginica
126	6.2	2.8	4.8	1.8	Virginica
127	6.1	3.0	4.9	1.8	Virginica
128	6.4	2.8	5.6	2.1	Virginica
129	7.2	3.0	5.8	1.6	Virginica
130	7.4	2.8	6.1	1.9	Virginica
131	7.9	3.8	6.4	2.0	Virginica
132	6.4	2.8	5.6	2.2	Virginica
133	6.3	2.8	5.1	1.5	Virginica
134	6.1	2.6	5.6	1.4	Virginica
135	7.7	3.0	6.1	2.3	Virginica
136	6.3	3.4	5.6	2.4	Virginica
137	6.4	3.1	5.5	1.8	Virginica
138	6.0	3.0	4.8	1.8	Virginica

```
139           6.9          3.1           5.4          2.1  Virginica
140           6.7          3.1           5.6          2.4  Virginica
141           6.9          3.1           5.1          2.3  Virginica
142           5.8          2.7           5.1          1.9  Virginica
143           6.8          3.2           5.9          2.3  Virginica
144           6.7          3.3           5.7          2.5  Virginica
145           6.7          3.0           5.2          2.3  Virginica
146           6.3          2.5           5.0          1.9  Virginica
147           6.5          3.0           5.2          2.0  Virginica
148           6.2          3.4           5.4          2.3  Virginica
149           5.9          3.0           5.1          1.8  Virginica

[150 rows x 5 columns]
Select and option
>> 2
Select 1 to query by row number or 2 to query by row name:
>> 1
Select the row number to query (indexes start in 0):
>> 65
sepal.length           6.7
sepal.width            3.1
petal.length           4.4
petal.width            1.4
variety         Versicolor
Name: 65, dtype: object
Select and option
>> 3
Indicate a column to include:
['sepal.length', 'sepal.width', 'petal.length', 'petal.width', 'variety']
>> sepla.length
The column indicated does not exit or has already been indicated. Please try again.
Indicate a column to include:
['sepal.length', 'sepal.width', 'petal.length', 'petal.width', 'variety']
>> sepal.length
Do you want to indicate another column? (Y/N)
>> y
Indicate a column to include:
['sepal.length', 'sepal.width', 'petal.length', 'petal.width', 'variety']
>> sepal.width
Do you want to indicate another column? (Y/N)
>> n
Showing the reduced dataset...
     sepal.length  sepal.width
0             5.1          3.5
1             4.9          3.0
2             4.7          3.2
3             4.6          3.1
4             5.0          3.6
```

5	5.4	3.9
6	4.6	3.4
7	5.0	3.4
8	4.4	2.9
9	4.9	3.1
10	5.4	3.7
11	4.8	3.4
12	4.8	3.0
13	4.3	3.0
14	5.8	4.0
15	5.7	4.4
16	5.4	3.9
17	5.1	3.5
18	5.7	3.8
19	5.1	3.8
20	5.4	3.4
21	5.1	3.7
22	4.6	3.6
23	5.1	3.3
24	4.8	3.4
25	5.0	3.0
26	5.0	3.4
27	5.2	3.5
28	5.2	3.4
29	4.7	3.2
..	...	...
120	6.9	3.2
121	5.6	2.8
122	7.7	2.8
123	6.3	2.7
124	6.7	3.3
125	7.2	3.2
126	6.2	2.8
127	6.1	3.0
128	6.4	2.8
129	7.2	3.0
130	7.4	2.8
131	7.9	3.8
132	6.4	2.8
133	6.3	2.8
134	6.1	2.6
135	7.7	3.0
136	6.3	3.4
137	6.4	3.1
138	6.0	3.0
139	6.9	3.1
140	6.7	3.1
141	6.9	3.1

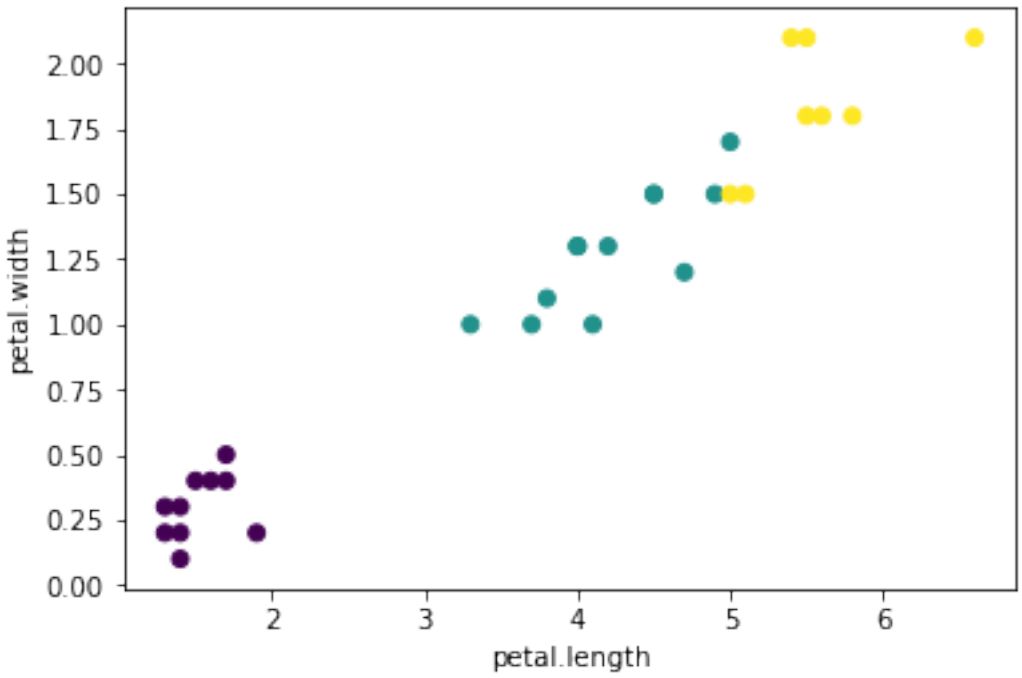
```
142           5.8          2.7
143           6.8          3.2
144           6.7          3.3
145           6.7          3.0
146           6.3          2.5
147           6.5          3.0
148           6.2          3.4
149           5.9          3.0

[150 rows x 2 columns]
Select and option
>> 4
Select 1 to use the original dataset or 2 to use the reduced dataset:
>> 1
Indicate the percentage of the dataset to assign as test data (between 0 and 1):
>> 1.1
Wrong input. Please try again.
Indicate the percentage of the dataset to assign as test data (between 0 and 1):
>> 0.2
Showing training data...
     sepal.length  sepal.width  petal.length  petal.width     variety
117           7.7          3.8           6.7          2.2   Virginica
31            5.4          3.4           1.5          0.4      Setosa
44            5.1          3.8           1.9          0.4      Setosa
113           5.7          2.5           5.0          2.0   Virginica
148           6.2          3.4           5.4          2.3   Virginica
149           5.9          3.0           5.1          1.8   Virginica
127           6.1          3.0           4.9          1.8   Virginica
87            6.3          2.3           4.4          1.3  Versicolor
141           6.9          3.1           5.1          2.3   Virginica
135           7.7          3.0           6.1          2.3   Virginica
17            5.1          3.5           1.4          0.3      Setosa
4             5.0          3.6           1.4          0.2      Setosa
9             4.9          3.1           1.5          0.1      Setosa
45            4.8          3.0           1.4          0.3      Setosa
78            6.0          2.9           4.5          1.5  Versicolor
50            7.0          3.2           4.7          1.4  Versicolor
65            6.7          3.1           4.4          1.4  Versicolor
124           6.7          3.3           5.7          2.1   Virginica
62            6.0          2.2           4.0          1.0  Versicolor
54            6.5          2.8           4.6          1.5  Versicolor
138           6.0          3.0           4.8          1.8   Virginica
94            5.6          2.7           4.2          1.3  Versicolor
19            5.1          3.8           1.5          0.3      Setosa
29            4.7          3.2           1.6          0.2      Setosa
25            5.0          3.0           1.6          0.2      Setosa
92            5.8          2.6           4.0          1.2  Versicolor
13            4.3          3.0           1.1          0.1      Setosa
```

```
35             5.0          3.2           1.2          0.2      Setosa
1              4.9          3.0           1.4          0.2      Setosa
89             5.5          2.5           4.0          1.3  Versicolor
..             ...          ...           ...          ...         ...
16             5.4          3.9           1.3          0.4      Setosa
128            6.4          2.8           5.6          2.1   Virginica
56             6.3          3.3           4.7          1.6  Versicolor
146            6.3          2.5           5.0          1.9   Virginica
93             5.0          2.3           3.3          1.0  Versicolor
110            6.5          3.2           5.1          2.0   Virginica
58             6.6          2.9           4.6          1.3  Versicolor
30             4.8          3.1           1.6          0.2      Setosa
137            6.4          3.1           5.5          1.8   Virginica
7              5.0          3.4           1.5          0.2      Setosa
47             4.6          3.2           1.4          0.2      Setosa
69             5.6          2.5           3.9          1.1  Versicolor
11             4.8          3.4           1.6          0.2      Setosa
10             5.4          3.7           1.5          0.2      Setosa
37             4.9          3.6           1.4          0.1      Setosa
122            7.7          2.8           6.7          2.0   Virginica
123            6.3          2.7           4.9          1.8   Virginica
90             5.5          2.6           4.4          1.2  Versicolor
104            6.5          3.0           5.8          2.2   Virginica
136            6.3          3.4           5.6          2.4   Virginica
102            7.1          3.0           5.9          2.1   Virginica
88             5.6          3.0           4.1          1.3  Versicolor
59             5.2          2.7           3.9          1.4  Versicolor
144            6.7          3.3           5.7          2.5   Virginica
120            6.9          3.2           5.7          2.3   Virginica
28             5.2          3.4           1.4          0.2      Setosa
109            7.2          3.6           6.1          2.5   Virginica
72             6.3          2.5           4.9          1.5  Versicolor
22             4.6          3.6           1.0          0.2      Setosa
143            6.8          3.2           5.9          2.3   Virginica

[120 rows x 5 columns]
Showing testing data...
     sepal.length  sepal.width  petal.length  petal.width     variety
77             6.7          3.0           5.0          1.7  Versicolor
112            6.8          3.0           5.5          2.1   Virginica
103            6.3          2.9           5.6          1.8   Virginica
133            6.3          2.8           5.1          1.5   Virginica
80             5.5          2.4           3.8          1.1  Versicolor
84             5.4          3.0           4.5          1.5  Versicolor
53             5.5          2.3           4.0          1.3  Versicolor
139            6.9          3.1           5.4          2.1   Virginica
96             5.7          2.9           4.2          1.3  Versicolor
26             5.0          3.4           1.6          0.4      Setosa
```

```
42            4.4          3.2           1.3          0.2      Setosa
52            6.9          3.1           4.9          1.5  Versicolor
71            6.1          2.8           4.0          1.3  Versicolor
81            5.5          2.4           3.7          1.0  Versicolor
21            5.1          3.7           1.5          0.4      Setosa
119           6.0          2.2           5.0          1.5   Virginica
108           6.7          2.5           5.8          1.8   Virginica
40            5.0          3.5           1.3          0.3      Setosa
66            5.6          3.0           4.5          1.5  Versicolor
49            5.0          3.3           1.4          0.2      Setosa
24            4.8          3.4           1.9          0.2      Setosa
116           6.5          3.0           5.5          1.8   Virginica
23            5.1          3.3           1.7          0.5      Setosa
67            5.8          2.7           4.1          1.0  Versicolor
105           7.6          3.0           6.6          2.1   Virginica
12            4.8          3.0           1.4          0.1      Setosa
5             5.4          3.9           1.7          0.4      Setosa
57            4.9          2.4           3.3          1.0  Versicolor
73            6.1          2.8           4.7          1.2  Versicolor
6             4.6          3.4           1.4          0.3      Setosa
Select and option
>> 5
Select 1 to use the original dataset, 2 to use the training dataset or 3 to use the test datase
>> 3
Select the variable to use as x axis:
['sepal.length', 'sepal.width', 'petal.length', 'petal.width', 'variety']
>> petal.length
Select the variable to use as y axis:
['sepal.length', 'sepal.width', 'petal.length', 'petal.width', 'variety']
>> petal.width
```




```
Select and option
>> 6
```

```
An exception has occurred, use %tb to see the full traceback.
```

```
SystemExit
```

```
C:\ProgramData\Anaconda\lib\site-packages\IPython\core\interactiveshell.py:2870: UserWarning: T
  warn("To exit: use 'exit', 'quit', or Ctrl-D.", stacklevel=1)
```

1.4 Questions

Please answer the following questions to appraise your level of engagement with the content of the course. Use the Markdown cell corresponding to each question to write your answers.

1. Using any of the two continuous variables of your dataset, show an example of how a linear regression (implemented using an existing Python module) could be applied on the training data to predict the values of one column of the test data. Discuss if there is any metric that can be used to decide which two variables are most correlated.

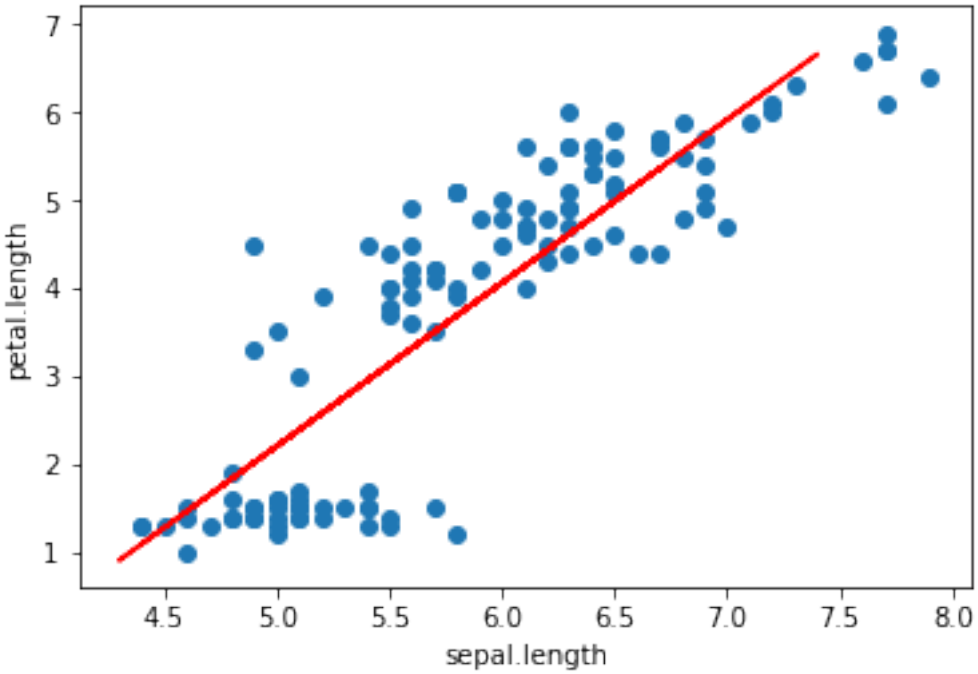
ANSWER: Students are encouraged to implement a code (as below) which allows them to test different X,Y combinations to find a suitable combination that show some level of correlation. Moreover, they can investigate concepts such as R^2 and *mean square error*, which are useful to see the level of correlation between two variables.

```
In [17]: # Use this cell to implement linear regression.
         import numpy as np
         import matplotlib.pyplot as plt  # To visualize
         import pandas as pd  # To read data
         from sklearn.linear_model import LinearRegression
         from sklearn.model_selection import train_test_split

         data_x = 0
         data_y = 2

         dataset = pd.read_csv('iris.csv')
         train, test = train_test_split(dataset, test_size=0.2)
         X = train.iloc[:, data_x].values.reshape(-1, 1)  # values converts it into a numpy ar
         Y = train.iloc[:, data_y].values.reshape(-1, 1)  # -1 means that calculate the dimens
         X_pred = test.iloc[:, data_x].values.reshape(-1, 1)  # values converts it into a nump
         linear_regressor = LinearRegression()  # create object for the class
         linear_regressor.fit(X, Y)  # perform linear regression
         Y_pred = linear_regressor.predict(X_pred)  # make predictions

         plt.scatter(X, Y)
         plt.plot(X_pred, Y_pred, color='red')
         plt.xlabel(dataset.columns[data_x])
         plt.ylabel(dataset.columns[data_y])
         plt.show()
```



2. Using any clustering method available in literature and in a Python module (e.g. hierarchical, k-means), briefly describe the selected method and implement it to classify the data of the original dataset into clusters. How would you verify how accurate is your clustering algorithm with respect to the original dataset target/class?

ANSWER: For this question, students will be encouraged to review scientific sources to explain how a clustering method works, then they can use an existing Python module to implement such method in their dataset. To verify the validity of the solution they may propose a simple comparison between elements or an existing module.

```
In [27]: # Use this cell to implement clustering.

         ## Clustering data using K-means
         import pandas as pd
         from sklearn.cluster import KMeans
         import numpy as np

         dataset = pd.read_csv('iris.csv')
         kmeans = KMeans(n_clusters=3, init='k-means++', n_init=1, max_iter=300,
                         tol=0.0001, precompute_distances='auto', random_state = 0,
                         n_jobs=1, algorithm='auto').fit(dataset[dataset.columns[:-1]])
         target_kmeans = kmeans.labels_
         print('The data is classified in the following clusters', target_kmeans)

         ## Verifying the accuracy
```

```
target = dataset['variety']
target=np.where(target=='Setosa', 0, target)
target=np.where(target=='Versicolor', 2, target)
target=np.where(target=='Virginica', 1, target)
comparison = target_kmeans == target
print('The comparison between target and clusters is: ', comparison)
print('Setosa', comparison[:50],'Accuracy', sum(comparison[:50])/len(comparison[:50]))
print('Versicolor', comparison[50:100],'Accuracy', sum(comparison[50:100])/len(compari
print('Virginica', comparison[100:],'Accuracy', sum(comparison[100:])/len(comparison[
```

```
The data is classified in the following clusters [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 1 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 1 1 1 1 2 1 1 1 1
 1 1 2 2 1 1 1 1 2 1 2 1 2 1 1 2 2 1 1 1 1 1 2 1 1 1 1 2 1 1 1 2 1 1 1 2 1
 1 2]
The comparison between target and clusters is:  [ True  True  True  True  True  True  True  Tru
  True  True  True  True  True  True  True  True  True  True  True  True
  True  True  True  True  True  True  True  True  True  True  True  True
  True  True  True  True  True  True  True  True  True  True  True  True
  True  True False  True False  True  True  True  True  True  True  True
  True  True  True  True  True  True  True  True  True  True  True  True
  True  True  True  True  True  True  True  True  True  True  True  True
  True  True  True  True  True False  True  True  True  True  True  True
  True  True  True  True  True  True  True  True  True  True  True  True
  True  True  True  True  True False  True  True  True  True False  True
  True  True  True  True  True False False  True  True  True  True False
  True False  True False  True  True False False  True  True  True  True
  True False  True  True  True  True False  True  True  True False  True
  True  True False  True  True False]
Setosa [ True  True  True  True  True  True  True  True  True  True  True  True
  True  True  True  True  True  True  True  True  True  True  True  True
  True  True  True  True  True  True  True  True  True  True  True  True
  True  True  True  True  True  True  True  True  True  True  True  True
  True  True] Accuracy 1.0
Versicolor [False  True False  True  True  True  True  True  True  True  True  True
  True  True  True  True  True  True  True  True  True  True  True  True
  True  True  True False  True  True  True  True  True  True  True  True
  True  True  True  True  True  True  True  True  True  True  True  True
  True  True] Accuracy 0.94
Virginica [ True False  True  True  True  True False  True  True  True  True  True
  True False False  True  True  True  True False  True False  True False
  True  True False False  True  True  True  True  True False  True  True
  True  True False  True  True  True False  True  True  True False  True
  True False] Accuracy 0.72
```