Assignment 2

a. Find the correlation matrix on the iris dataset.

b. Plot the correlation plot on the dataset and visualize giving an overview of relationships among data on iris dataset.

Correlation matrix

A correlation matrix is simply a table which displays the correlation coefficients for different variables. The matrix depicts the correlation between all the possible pairs of values in a table. It is a powerful tool to summarize a large dataset and to identify and visualize patterns in the given data.

A correlation matrix consists of rows and columns that show the variables. Each cell in a table contains the correlation coefficient.

In addition, the correlation matrix is frequently utilized in conjunction with other types of statistical analysis. For instance, it may be helpful in the analysis of multiple linear regression models. Remember that the models contain several independent variables. In multiple linear regression, the correlation matrix determines the correlation coefficients between the independent variables in a model.

Correlation plots

A scatterplot shows the relationship between two quantitative variables measured for the same individuals. The values of one variable appear on the horizontal axis, and the values of the other variables appear on the vertical axis. Each individual in the data appears as a point on the graph.

Many research projects are correlational studies because they investigate the relationships that may exist between variables. Prior to investigating the relationship between two quantitative variables, it is always helpful to create a graphical representation that includes both of these variables. Such a graphical representation is called a scatterplot.

*#Importing the required libraries*  
import pandas as pd  
import numpy as np  
import seaborn as sns  
import matplotlib.pyplot as plt  
from google.colab import drive  
  
drive.mount('/content/drive')  
data = pd.read\_csv('/content/drive/MyDrive/Ass1/cigarettes.csv')  
  
*#Displaying info about the data*  
data.info()

Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force\_remount=True).  
<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 46 entries, 0 to 45  
Data columns (total 4 columns):  
 # Column Non-Null Count Dtype   
--- ------ -------------- -----   
 0 Unnamed: 0 46 non-null object   
 1 packs 46 non-null float64  
 2 price 46 non-null float64  
 3 income 46 non-null float64  
dtypes: float64(3), object(1)  
memory usage: 1.6+ KB

* Correlation is a statistical tool that is used to determine the relationship between two variables.
* Covariance evaluates how the mean values (average values) of two random vaiables move together. It measures the direction of relationship.
* Correaltion measures the strength of relation.

R = Sum((xi-x')(yi-y'))/n(s(x)\*s(y)) where s(x) = std. deviation of x s(y) = std. deviation of y

Cov(x,y) = Sum((xi-x')(yi-y'))

*#Description of the dataset in brief*  
data.describe()

packs price income  
count 46.000000 46.000000 46.000000  
mean 4.847844 0.205509 4.775455  
std 0.191458 0.086230 0.142261  
min 4.408590 -0.032600 4.529380  
25% 4.711547 0.140543 4.679075  
50% 4.814950 0.200205 4.758505  
75% 4.984450 0.273485 4.852850  
max 5.379060 0.363990 5.102680

*#Printing the range of the dataset*  
print(data.index)

RangeIndex(start=0, stop=46, step=1)

*#For displaying the correaltion matrix*   
features = data.iloc[:, 0:8]  
target = data['income']  
print(features)  
print("\n")  
print(target)  
print("\n")

Unnamed: 0 packs price income  
0 AL 4.96213 0.20487 4.64039  
1 AZ 4.66312 0.16640 4.68389  
2 AR 5.10709 0.23406 4.59435  
3 CA 4.50449 0.36399 4.88147  
4 CT 4.66983 0.32149 5.09472  
5 DE 5.04705 0.21929 4.87087  
6 DC 4.65637 0.28946 5.05960  
7 FL 4.80081 0.28733 4.81155  
8 GA 4.97974 0.12826 4.73299  
9 ID 4.74902 0.17541 4.64307  
10 IL 4.81445 0.24806 4.90387  
11 IN 5.11129 0.08992 4.72916  
12 IA 4.80857 0.24081 4.74211  
13 KS 4.79263 0.21642 4.79613  
14 KY 5.37906 -0.03260 4.64937  
15 LA 4.98602 0.23856 4.61461  
16 ME 4.98722 0.29106 4.75501  
17 MD 4.77751 0.12575 4.94692  
18 MA 4.73877 0.22613 4.99998  
19 MI 4.94744 0.23067 4.80620  
20 MN 4.69589 0.34297 4.81207  
21 MS 4.93990 0.13638 4.52938  
22 MO 5.06430 0.08731 4.78189  
23 MT 4.73313 0.15303 4.70417  
24 NE 4.77558 0.18907 4.79671  
25 NV 4.96642 0.32304 4.83816  
26 NH 5.10990 0.15852 5.00319  
27 NJ 4.70633 0.30901 5.10268  
28 NM 4.58107 0.16458 4.58202  
29 NY 4.66496 0.34701 4.96075  
30 ND 4.58237 0.18197 4.69163  
31 OH 4.97952 0.12889 4.75875  
32 OK 4.72720 0.19554 4.62730  
33 PA 4.80363 0.22784 4.83516  
34 RI 4.84693 0.30324 4.84670  
35 SC 5.07801 0.07944 4.62549  
36 SD 4.81545 0.13139 4.67747  
37 TN 5.04939 0.15547 4.72525  
38 TX 4.65398 0.28196 4.73437  
39 UT 4.40859 0.19260 4.55586  
40 VT 5.08799 0.18018 4.77578  
41 VA 4.93065 0.11818 4.85490  
42 WA 4.66134 0.35053 4.85645  
43 WV 4.82454 0.12008 4.56859  
44 WI 4.83026 0.22954 4.75826  
45 WY 5.00087 0.10029 4.71169  
  
  
0 4.64039  
1 4.68389  
2 4.59435  
3 4.88147  
4 5.09472  
5 4.87087  
6 5.05960  
7 4.81155  
8 4.73299  
9 4.64307  
10 4.90387  
11 4.72916  
12 4.74211  
13 4.79613  
14 4.64937  
15 4.61461  
16 4.75501  
17 4.94692  
18 4.99998  
19 4.80620  
20 4.81207  
21 4.52938  
22 4.78189  
23 4.70417  
24 4.79671  
25 4.83816  
26 5.00319  
27 5.10268  
28 4.58202  
29 4.96075  
30 4.69163  
31 4.75875  
32 4.62730  
33 4.83516  
34 4.84670  
35 4.62549  
36 4.67747  
37 4.72525  
38 4.73437  
39 4.55586  
40 4.77578  
41 4.85490  
42 4.85645  
43 4.56859  
44 4.75826  
45 4.71169  
Name: income, dtype: float64

In the features square bracket, the first ':' denotes that all columns must be considered and the range which is given as second argument will print columns from 0 to n-1. For finding correlation coefficient, we have "corrcoef" function.

*#Finding the correlation coefficient of specific columns of the dataset*  
matrix = np.corrcoef(data['income'], data['packs'])  
print(matrix)  
print("\n\n")  
print(data[['income', 'packs']].corr())

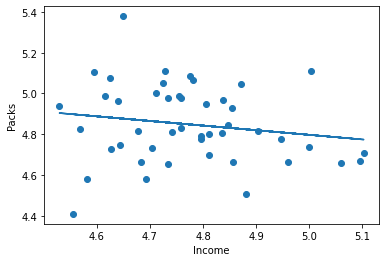
[[ 1. -0.16867283]  
 [-0.16867283 1. ]]  
  
  
  
 income packs  
income 1.000000 -0.168673  
packs -0.168673 1.000000

The .corr() function comes with a dataframe.

*#Correlation matrix of the whole dataset*  
print(data.cov())  
matrix = data.corr()  
print(matrix)

packs price income  
packs 0.036656 -0.008910 -0.004594  
price -0.008910 0.007436 0.006040  
income -0.004594 0.006040 0.020238  
 packs price income  
packs 1.000000 -0.539707 -0.168673  
price -0.539707 1.000000 0.492332  
income -0.168673 0.492332 1.000000

*#Scatter plot of two variables*  
plt.scatter(data['income'], data['packs'])  
m,b = np.polyfit(data['income'], data['packs'], 1)  
plt.plot(data['income'], m\*data['income']+b) *#(y, mx+c) format*  
plt.xlabel("Income")  
plt.ylabel("Packs")  
plt.show()



*#Ploting the corr plot for the whole dataset*  
sns.pairplot(data)

<seaborn.axisgrid.PairGrid at 0x7f3b92f7d910>

