Lab 1: Data Loading, Summary, and Visualisation

Type ‘jupyter notebook’ in terminal
A new browser window will be opened listing the directories and files of anaconda.

Choose a new Python 3 file.

1. Data Frame

A data frame is a 2-d data structure similar to matrix. However columns of the matrix can be of different types, Size is mutable, Rows and columns can be labelled. And Arithmetic operations can be performed on rows and columns.

Create a pandas data frame as follows.

```
import numpy
import pandas
myarray = numpy.array([[1,2,3],[4,5,6]])
rownames = ['a','b']
colnames=['f1','f2','f3']
mydataframe = pandas.DataFrame(myarray, index = rownames, columns=colnames)
print(mydataframe)
```

Change the type of data

```
import numpy
import pandas
myarray = numpy.array([['a','sandhya',9.6],[4,'shreya',6.5]])
rownames = ['r1','r2']
colnames=['f1','f2','f3']
mydataframe = pandas.DataFrame(myarray, index = rownames, columns=colnames)
print(mydataframe)
```

2. Load csv file using pandas from a specific path or url

Copy dataset given in https://www.kaggle.com/uciml/pima-indians-diabetes-database to your local folder. Then execute the following.

```
from pandas import read_csv
path='1_diabetes.csv'
data=read_csv(path)
print (data.shape)          #to know size of the data
```

The file can be given column names as follows

```
from pandas import read_csv
url='1_diabetes.csv'
data=read_csv(url)
```

```
colnames=['Pregnancies','Glucose','BloodPressure','SkinThickness','Insulin','BMI','Diab
etesPedigreeFunction','Age','Outcome']
print (data.shape)
```

3. To get statistical summary of the data

(a)

```
description = data.describe()
print(description)
```

This will give statistics of each column in the dataset.

Example

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	\
count	768.000000	768.000000	768.000000	768.000000	768.000000	
mean	3.845052	120.894531	69.105469	20.536458	79.799479	
std	3.369578	31.972618	19.355807	15.952218	115.244002	
min	0.000000	0.000000	0.000000	0.000000	0.000000	
25%	1.000000	99.000000	62.000000	0.000000	0.000000	
50%	3.000000	117.000000	72.000000	23.000000	30.500000	
75%	6.000000	140.250000	80.000000	32.000000	127.250000	
max	17.000000	199.000000	122.000000	99.000000	846.000000	

Here 25%, 50%, gives % of data that falls below a given corresponding value in each column.

(b) Size of matrix

```
print(data.shape)
```

(c) Peek at data

```
print(data.head(4))
```

(d) Group on the basis of a particular attribute

```
print(data.groupby('Outcome').size())
```

4. Data visualization

For plotting pairs of attributes as scattered plot, specify the attributes to be plotted explicitly

```
import matplotlib.pyplot as plt
import pandas
from pandas.plotting import scatter_matrix
scatter_matrix(data[['Pregnancies','Glucose']])
plt.show()
```

For plotting all pairs of attributes in data

```
import matplotlib.pyplot as plt
import pandas
from pandas.plotting import scatter_matrix
scatter_matrix(data)      #scatter plot
plt.show()
data.hist()                #histogram
plt.show()
```

5. Standardization of dataset

```
from sklearn.preprocessing import StandardScaler
import pandas
import numpy
arr=data.values                #convert data frame to array
X=arr[:,0:8]                   #split columns
Y=arr[:,8]
scaler=StandardScaler().fit(X)          #fit data for standardization
rescaledX=scaler.transform(X)           #convert the data as per (x-μ)/σ

numpy.set_printoptions(precision=3)
print(rescaledX[0:2,:])
print(X[0:2,:])
```

6. Normalizing a column in pandas

Create a dataframe for a set of values in an array

```
myarray=numpy.array([1,3,-10,4,5,7,-4,-2,10])
mydataframe = pandas.DataFrame(myarray)
print(mydataframe)
```

plot the data

```
mydataframe.plot(kind='bar')
plt.show()
```

Plot normalized data

```
from sklearn import preprocessing
fl_x=mydataframe.values.astype(float)

#fl_x=mydataframe[['f1']].values.astype(float)   #If specific feature name is to be converted

min_max_scaler=preprocessing.MinMaxScaler()
X_scaled=min_max_scaler.fit_transform(fl_x)
df_normalized=pandas.DataFrame(X_scaled)
print(df_normalized)
df_normalized.plot(kind='bar')
plt.show()
```

Question : Identify the difference in the standardization and normalization of data.

Repeat Q.1 to Q.6 with cancer dataset from https://www.kaggle.com/uciml/breast-cancer-wisconsin-data