**CSNB614: Machine Learning & Data Analytics**

**LAB 2 – Introduction to Machine Learning and Toolkit**

This week, we are going to learn some Python libraries, NumPy and Pandas. You have to refer to the lecture slide.

1. Launch the tool and environment
   1. Launch Anaconda Navigator
   2. Launch Anaconda Prompt
   3. Create virtual environment:
      1. conda create –n tensorflow\_env tensorflow
   4. Proceed ([y]/n)? Y
   5. Activate this environment: conda activate tensorflow\_env
2. Download the exercise file, Introduction\_to\_Machine\_Learning\_and\_Toolkit.ipynb and data.zip. Extract the zip file. Move the extracted file into the same folder as you wish to be used with the Python program later. Open the .ipynb file using Jupyter Notebook. Write your answer (code) according to the question.

|  |  |
| --- | --- |
| **Question** | **Code** |
|  | import numpy as np  import pandas as pd  # assigning file path  filepath= 'C:\\Users\\SW0104233\\Desktop\\wong\\data\\Iris\_Data.csv'  # print out top 5 of the lists  data = pd.read\_csv(filepath)  # Read csv files  data.head()  # Number of data points  data.shape[0]  # Column names  data.columns.tolist()  # Data types for each column  data.dtypes |
|  | # Rename the species by removing the 'Iris-' in front of all species  # data->species->string->replace(string to be replaced, string that need to replaced to)  data['species'] = data.species.str.replace('Iris-','')  data.head() |
|  | # Get description of data  stats\_df = data.describe()  # Get maximum value :: stats\_df.loc['max']  # Get minimum value :: stats\_df.loc['min']  # Get range of data  stats\_df.loc['range'] = stats\_df.loc['max'] - stats\_df.loc['min']  # The datasets needed to be viewed  out\_fields = ['mean', '25%', '50%', '75%', 'range']  # Locate the datasets needed to be viewed  stats\_df = stats\_df.loc[out\_fields]  # Rename '50%' to 'median'  stats\_df.rename({'50%':'median'}, inplace=True)  # Print the datasets  stats\_df |
|  | # Mean for each species  data.groupby('species').mean()  # Median for each species  data.groupby('species').median()  # Applying multiple functions at once throughh aggregration  data.groupby('species').agg(['mean','median']) # 1  data.groupby('species').agg([np.mean,np.median]) # 2  # 1 and 2 are resulted in same output |
|  | # Import the matplotlib library for plotting graph  import matplotlib.pyplot as plt  %matplotlib inline  # Create the axes  ax = plt.axes()  # Scatter the data in (x,y) format  ax.scatter(data.sepal\_length, data.sepal\_width)  # Labelling  ax.set(  xlabel = 'Sepal Length (cm)',  ylabel = 'Sepal Width (cm)',  title = 'Sepal Length VS Width'  ) |
|  | # Plotlib - Sepal Length  ay = plt.axes()  ay.hist(data.sepal\_length, bins=25) # sepal length  ay.set(  xlabel = 'Sepal Length(cm)',  ylabel = 'Frequency',  title = 'Distribution of Sepal Length'  )  # Pandas - petal length  plpd = data.petal\_length.plot.hist(bins=25)  plpd.set(  xlabel = 'Petal Length(cm)',  title = 'Distribution of Petal Length'  )  # Plotlib - sepal width  swpl = plt.axes()  swpl.hist(data.sepal\_width, bins=25) # sepal length  swpl.set(  xlabel = 'Sepal Width(cm)',  ylabel = 'Frequency',  title = 'Distribution of Sepal Width'  )  # Pandas - petal width  pwpd = data.petal\_width.plot.hist(bins=25)  pwpd.set(  xlabel = 'Petal Width(cm)',  title = 'Distribution of Petal Width'  ) |
|  | # seaborn  try:  import seaborn as sb  except:  print('Please install seaborn through "conda install seaborn"')  axnew = data.plot.hist(bins=25, alpha=0.5)  axnew.set\_xlabel('Size (cm)')  # pandas  axnew2 = data.hist(bins=25)  for axpd in axnew2.flatten():  if axpd.is\_last\_row():  axpd.set\_xlabel('Size(cm)')  if axpd.is\_first\_col():  axpd.set\_ylabel('Frequency') |
|  | # pandas  data.boxplot(by='species') |
|  | # step 1  plot\_data = (  data.set\_index('species').  stack().  to\_frame().  reset\_index().  rename(columns={0:'size','level\_1':'measurement'})  )  plot\_data.head()  # step 2  sb.set\_style('white')  sb.set\_palette('dark')  f = plt.figure(figsize=(6,4))  sb.boxplot(x='measurement', y='size', hue='species', data=plot\_data) |
|  | sb.pairplot(data, hue='species') |