EE2211 Tutorial 2

Dr Feng LIN
feng_lin@nus.edu.sg

Q0

Program for demonstration of one-hot encoding

```
from numpy import array
from numpy import argmax
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import OneHotEncoder
```

Importing Libraries

```
# define example
data = ['cold', 'cold', 'warm', 'cold', 'hot', 'hot', 'warm', 'cold', 'warm', 'hot']
values = array(data)
print(values)
```

Define example data

```
# integer encode
label_encoder = LabelEncoder()
integer_encoded = label_encoder.fit_transform(values)
print(integer_encoded)
```

Integer encode: convert categorical labels into integer labels:

- 'cold' -> 0
- 'warm' -> 2
- 'hot' -> 1

```
# binary encode
onehot_encoder = OneHotEncoder(sparse_output=False)
integer_encoded = integer_encoded.reshape(len(integer_encoded), 1)
onehot_encoded = onehot_encoder.fit_transform(integer_encoded)
print(onehot_encoded)
```

OneHotEncoder converts the integer-encoded array into a one-hot encoded matrix. Each category is represented by a binary vector:

0 -> [1, 0, 0] (cold)
1 -> [0, 1, 0] (hot)
2 -> [0, 0, 1] (warm

```
# invert first example
inverted = label_encoder.inverse_transform([argmax(onehot_encoded[0, :])])
print(inverted)
```

converted back to the original label

01

Q1

(Data Reading and Visualization, simple data structure)

A Comma Separated Values (CSV) file is a plain text file that contains a list of data. These files are often used for exchanging data between different applications.

- Download the file “government-expenditure-on-education.csv” from https://data.gov.sg/dataset/government-expenditure-on-education .
- Plot the educational expenditure over the years. (Hint: you might need “import pandas as pd” and “import matplotlib.pyplot as plt”.)

Q1

```
import pandas as pd
import matplotlib.pyplot as plt
import os
```

Importing Libraries

```
df = pd.read_csv("./GovernmentExpenditureonEducation.csv")
```

Reading the CSV File

```
expenditureList = df ['total_expenditure_on_education'].tolist()
yearList = df ['year'].tolist()
```

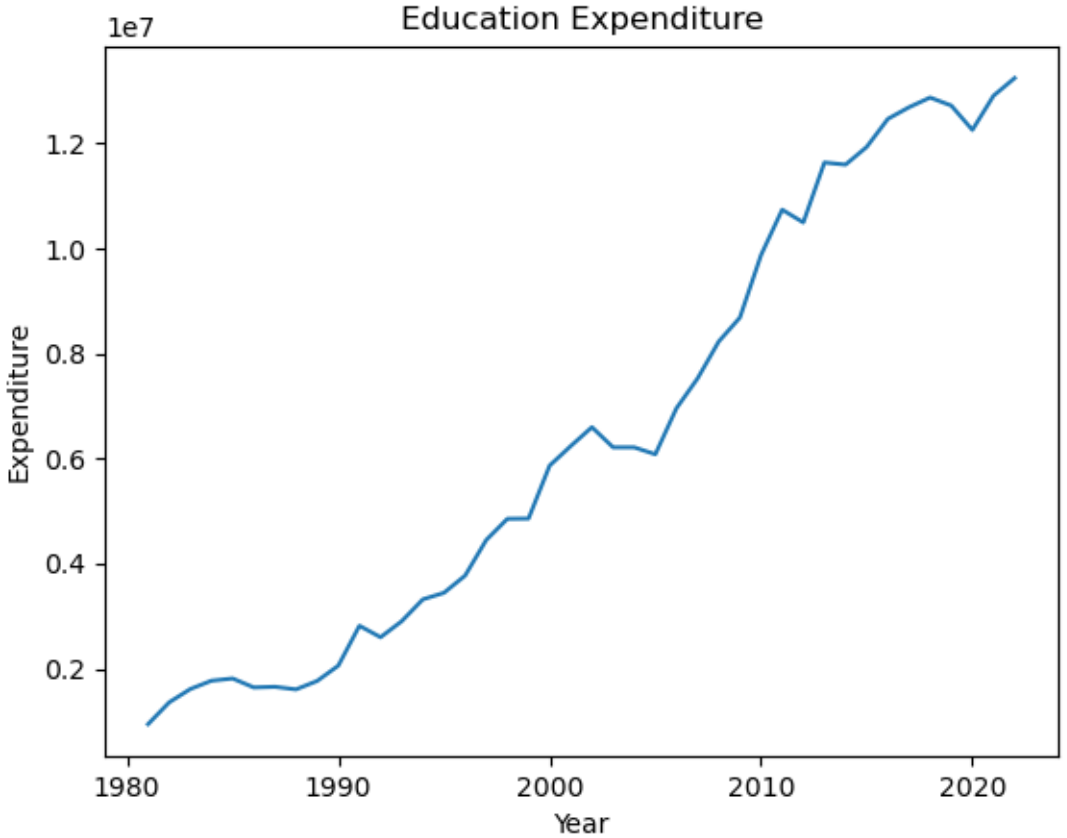
Extract the column from DataFrame and convert it into a list.

```
plt.plot(yearList, expenditureList, label = 'Expenditure over the years')
plt.xlabel('Year')
plt.ylabel('Expenditure')
plt.title('Education Expenditure')
plt.show()
```

Plotting the Data

01

Q1



02

Q2

(Data Reading and Visualization, slightly more complicated data structure.)

- Download the CSV file from https://data.gov.sg/dataset/annual-motor-vehicle-population-by-vehicle-type.
- Extract and plot the number of Omnibuses, Excursion buses and Private buses over the years as shown below.
- (Hint: you might need “import pandas as pd” and “import matplotlib.pyplot as plt”.)

```
import pandas as pd
import matplotlib.pyplot as plt
import os
```

Importing Libraries

```
df = pd.read_csv("AnnualMotorVehiclePopulationbyVehicleType.csv")
```

Loading the Data

```
year = df ['year'].tolist()
category = df ['category'].tolist()
vehtype = df ['type'].tolist()
number = df ['number'].tolist()
```

Convert the columns from the DataFrame into Python List

```
val1 = df.loc[df['type']=='Omnibuses'].index
val2 = df.loc[df['type']=='Excursion buses'].index
val3 = df.loc[df['type']=='Private buses'].index
print(val1)
```

Filter the DataFrame based on the condition provided inside the loc method. '.index' returns the indices where the condition is true.

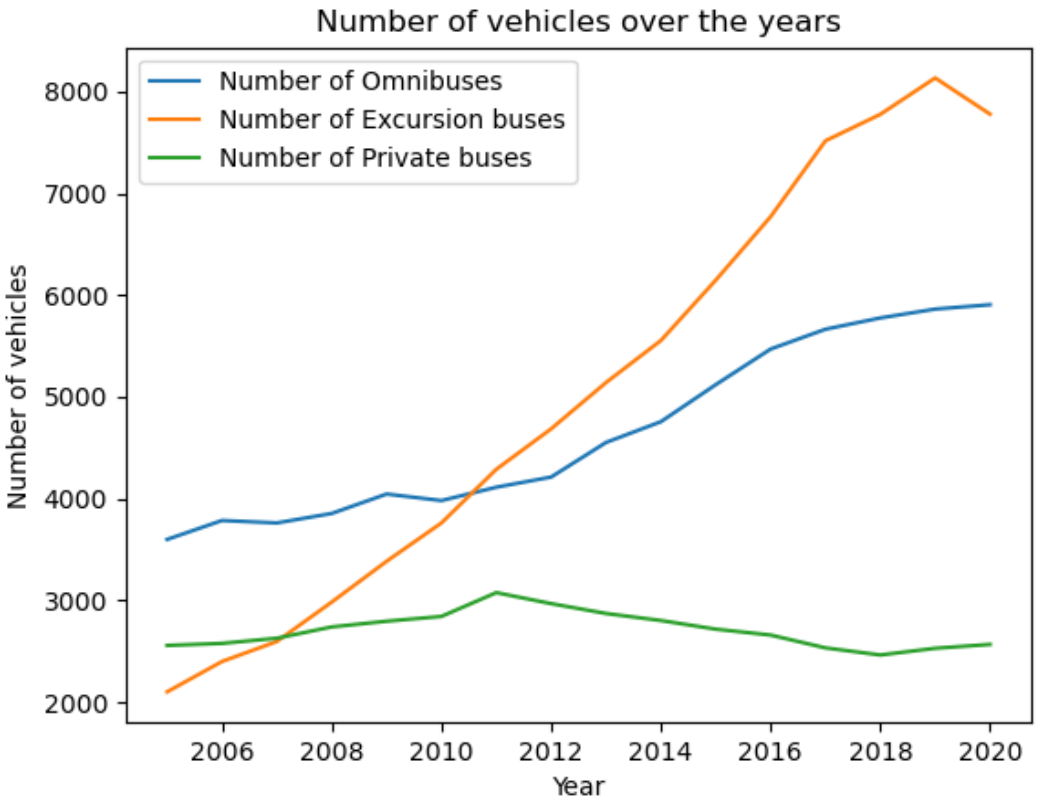
```
List1 = df.loc[val1]; print(List1)
List2 = df.loc[val2]; print(List2)
List3 = df.loc[val3]; print(List3)
```

Create lists contain only the rows corresponding to 'Omnibuses', 'Excursion buses', and 'Private buses' respectively

```
plt.plot(List1['year'], List1['number'], label = 'Number of Omnibuses')
plt.plot(List2['year'], List2['number'], label = 'Number of Excursion buses')
plt.plot(List3['year'], List3['number'], label = 'Number of Private buses')
plt.xlabel('Year')
plt.ylabel('Number of vehicles')
plt.title('Number of vehicles over the years')
plt.legend()
plt.show()
```

02

Q2



Q2 Method 2

```
import pandas as pd
import colorsys
import matplotlib.pyplot as plt
import seaborn as sns
```

Importing Libraries

```
df = pd.read_csv("AnnualMotorVehiclePopulationbyVehicleType.csv")
```

Load the data

```
sns.set_style("darkgrid")
```

Sets the aesthetic style of the plots to "darkgrid", which includes a dark background with gridlines.

```
df3 = df.loc[df['type'].isin(['Omnibuses', 'Excursion buses', 'Private buses’])]
```

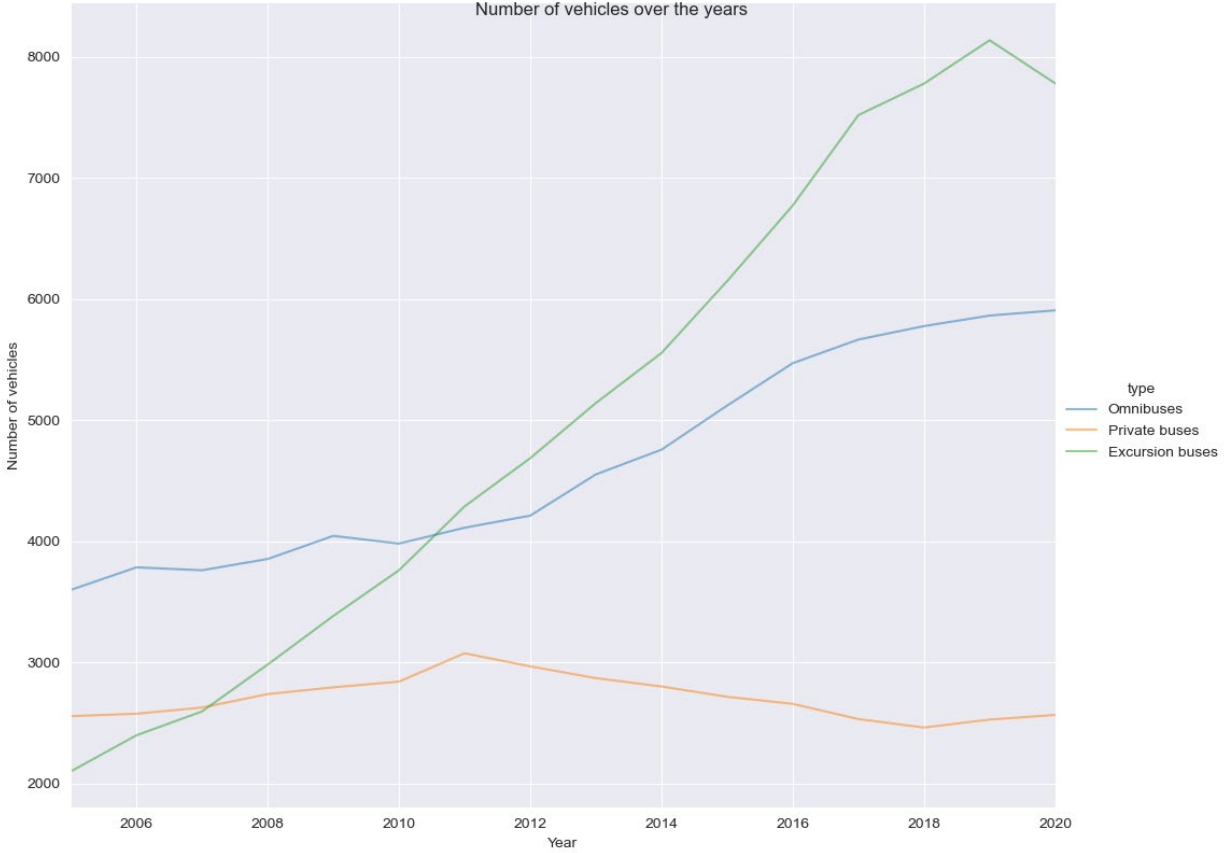
- df.loc[]: This is a label-based indexer for selecting rows and columns from the DataFrame.
- df['type'].isin(): Filters the rows where the specific 'type'

```
g = sns.PairGrid(data=df3, x_vars="year", y_vars="number", hue="type", height=10, aspect=1)
```

Creates a grid for plotting pairwise relationships in the dataset

```
g = g.map(plt.plot, alpha=0.5)
g = g.set(xlim=(df['year'].min(), df['year'].max()))
g = g.add_legend()
g.fig.suptitle('Number of vehicles over the years')
plt.xlabel('Year')
plt.ylabel('Number of vehicles')
```

Q2 Method 2



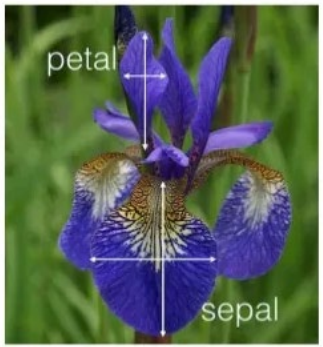
03

Q3

(Data Reading and Visualization, distribution)

The “iris” flower data set consists of measurements such as the length, width of the petals, and the length, width of the sepals, all measured in centimeters, associated with each iris flower.

- Get the data set “from sklearn.datasets import load_iris” and
- Do a scatter plot as shown below. (Hint: you might need “from pandas.plotting import scatter_matrix”)



https://analyticsindiamag.com/ai-mysteries/start-building-first-machine-learning-project-famous-dataset/

https://medium.com/analytics-vidhya/exploration-of-iris-dataset-using-scikit-learn-part-1-8ac5604937f8

Q3 Method 1

```
import pandas as pd; print("pandas version: {}".format(pd.__version__))
import matplotlib.pyplot as plt
import sklearn; print("scikit-learn version: {}".format(sklearn.__version__))
```

Importing Libraries and Printing Versions

```
from sklearn.datasets import load_iris
iris_dataset = load_iris()
from sklearn.model_selection import train_test_split
```

Loading the Iris Dataset

```
X_train, X_test, y_train, y_test = train_test_split( iris_dataset['data'], iris_dataset['target'],
random_state=0)
```

Splitting the dataset into training and testing sets

```
# create dataframe from data in X_train
# label the columns using the strings in iris_dataset.feature_names
iris_dataframe = pd.DataFrame(X_train, columns=iris_dataset.feature_names)
```

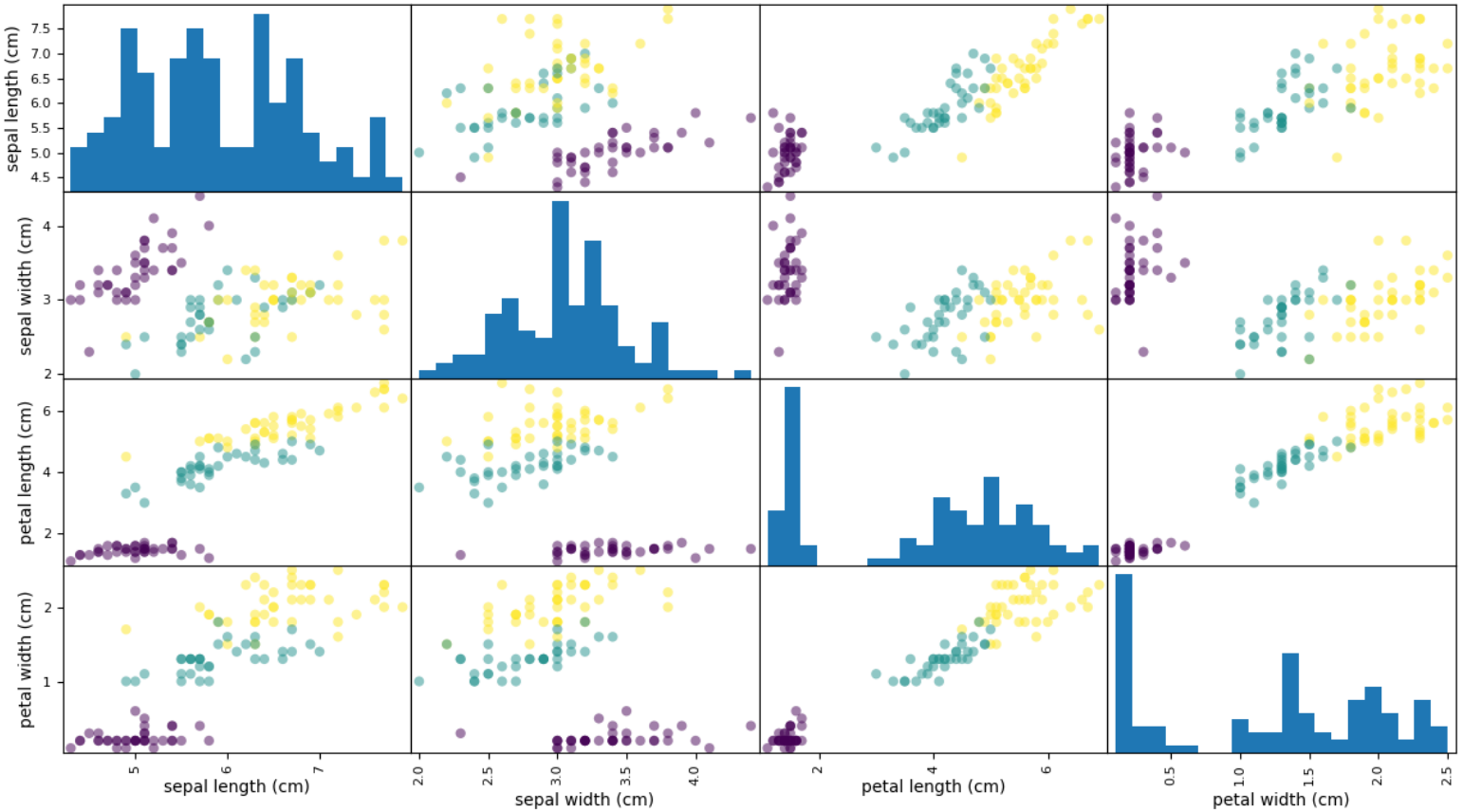
- Create a DataFrame from training data
- Label columns using feature names from iris_dataset (e.g., sepal length, sepal width, petal length, petal width).

Visualizing data with a scatter matrix

```
# create a scatter matrix from the dataframe, color by y_train
from pandas.plotting import scatter_matrix

grr = pd.plotting.scatter_matrix(iris_dataframe, c=y_train, figsize=(15, 15), marker='o',
hist_kwds={'bins': 20})
plt.show()
```

Colors the points according to their class labels


sepal length (cm)
sepal width (cm)
petal length (cm)
petal width (cm)
sepal length (cm)
sepal width (cm)
petal length (cm)
petal width (cm)

Q3 Method 2

```
import pandas as pd
import colorsys
import matplotlib.pyplot as plt
import seaborn as sns
```

Importing Libraries

```
sns.set()
iris = sns.load_dataset("iris")
```

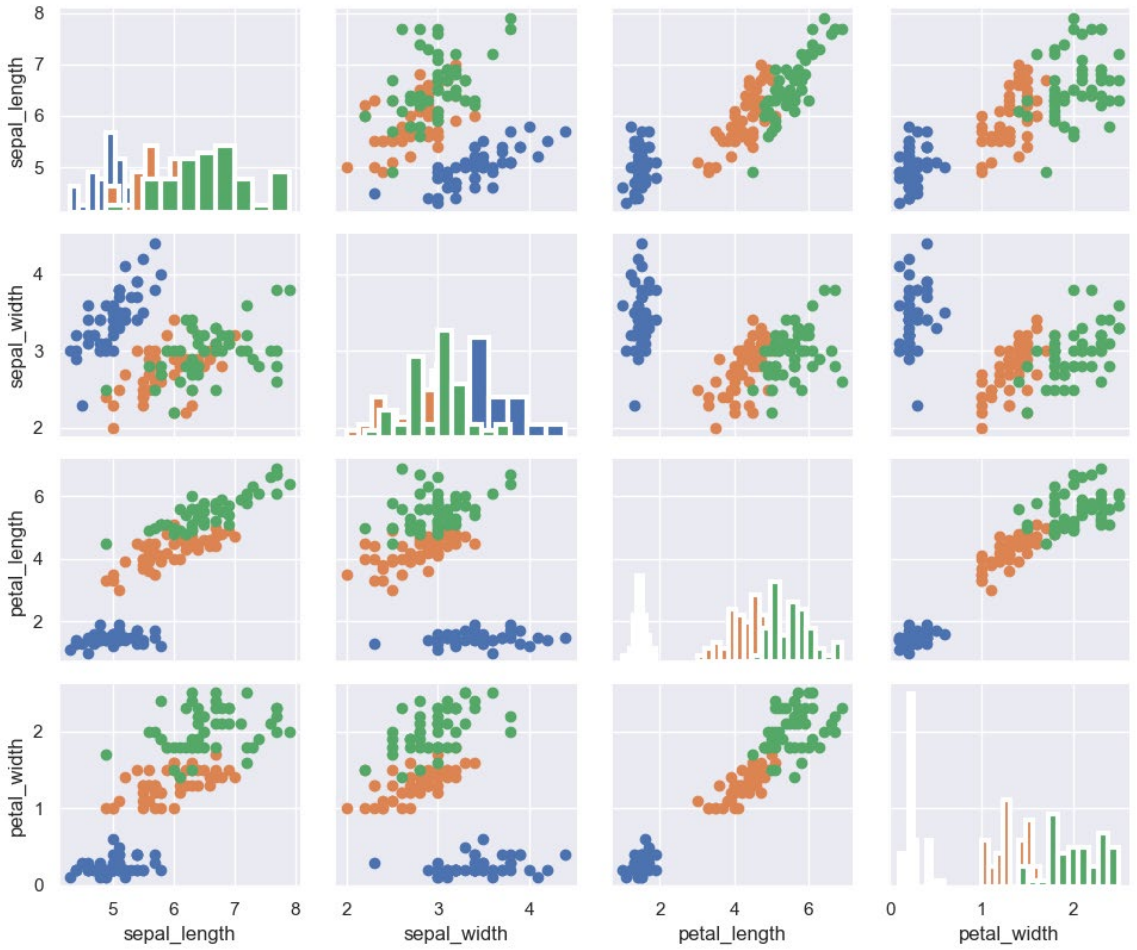
Set Seaborn default style and load the iris dataset

```
g = sns.PairGrid(iris, hue="species")
```

- Create a PairGrid to plot pairwise relationships in a dataset
- Specifies that different species of iris flowers should be distinguished by different colors in the plots.

```
g = g.map_diag(plt.hist, linewidth=3)
g = g.map_offdiag(plt.scatter)
plt.show()
```

- Create histograms on diagonal
- Create scatter plots on the off-diagonal



04

Q 4

(Data Wrangling/Normalization)

You are given a set of data for supervised learning. A sample block of data looks like this:

" 1.2234, 0.3302, 123.50, 0.0081, 30033.81, 1
1.3456, 0.3208, 113.24, 0.0067, 29283.18, -1
0.9988, 0.2326, 133.45, 0.0093, 36034.33, 1
1.1858, 0.4301, 128.55, 0.0077, 34037.35, 1
1.1533, 0.3853, 116.70, 0.0066, 22033.58, -1
1.2755, 0.3102, 118.30, 0.0098, 30183.65, 1
1.0045, 0.2901, 123.52, 0.0065, 31093.98, -1
1.1131, 0.3912, 113.15, 0.0088, 29033.23, -1 "

Each row corresponds to a sample data measurement with 5 input features and 1 response.

(a) What kind of undesired effect can you anticipate if this set of raw data is used for learning?

(b) How can the data be preprocessed to handle this issue?

Q4

Ans:

(a) Those features with very large values may overshadow those with very small values.

(b) We can either use min-max or z-score normalization to resolve the problem.

```
# Tutorial 2 Problem 4
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
```

Importing Libraries

```
data = [ [1.2234, 0.3302, 123.50, 0.0081, 30033.81, 1],
[1.3456, 0.3208, 113.24, 0.0067, 29283.18, -1],
[0.9988, 0.2326, 133.45, 0.0093, 36034.33, 1],
[1.1858, 0.4301, 128.55, 0.0077, 34037.35, 1],
[1.1533, 0.3853, 116.70, 0.0066, 22033.58, -1],
[1.2755, 0.3102, 118.30, 0.0098, 30183.65, 1],
[1.0045, 0.2901, 123.52, 0.0065, 31093.98, -1],
[1.1131, 0.3912, 113.15, 0.0088, 29033.23, -1] ]
```

'data' is a list of lists, where each inner list represents a row of data. Each row contains six values.

```
df = pd.DataFrame(data)
df.head(7)
```

Converts the list of lists into a pandas DataFrame, which is a table-like data structure with rows and columns.

```
from sklearn import preprocessing

# Z-score scaling
df_scaled = preprocessing.scale(df)
print(df_scaled.mean(axis=0))
print(df_scaled.std(axis=0))
```

Applies Z-score scaling to the DataFrame df. Z-score scaling standardizes the data such that each column will have a mean of 0 and a standard deviation of 1.

```
# min-max scaling
mix_max_scale = preprocessing.MinMaxScaler()
df_minax = mix_max_scale.fit_transform(df)
```

Min-Max scaling transforms the data so that all features are within a given range, usually [0, 1]

05 Q5 (Missing Data)

The Pima Indians Diabetes Dataset involves predicting the onset of diabetes within 5 years in Pima Indians given medical details. Download the Pima-Indians-Diabetes data from https://raw.githubusercontent.com/jbrownlee/Datasets/master/pima-indians-diabetes.data.csv.

It is a binary (2-class) classification problem. The number of observations for each class is not balanced. There are 768 observations with 8 input variables and 1 output variable. The variable names are as follows:

0. Number of times pregnant.
1. Plasma glucose concentration a 2 hours in an oral glucose tolerance test.
2. Diastolic blood pressure (mm Hg).
3. Triceps skinfold thickness (mm).
4. 2-Hour serum insulin (mu U/ml).
5. Body mass index (weight in kg/(height in m)^2).
6. Diabetes pedigree function.
7. Age (years).
8. Class variable (0 or 1).

(a) Print the summary statistics of this data set.
(b) Count the number of “0” entries in columns [1,2,3,4,5].
(c) Replace these “0” values by “NaN”.
(Hint: you might need the “.describe()” and “.replace(0, numpy.NaN)” functions “from pandas import read_csv”.)

05 Q5

```
#(a) from pandas import read_csv
import pandas as pd
dataset = pd.read_csv('pima-indians-diabetes.csv', header=None)
print(dataset.describe())
```

Importing the Dataset

```
#(b)
print((dataset[[1,2,3,4,5]] == 0).sum())
```

Counting Zeros in Specific Columns

```
#(c) import numpy
import numpy
# mark zero values as missing or NaN
dataset[[1,2,3,4,5]] = dataset[[1,2,3,4,5]].replace(0, numpy.NaN)
# print the first 20 rows of data
print(dataset.head(20))
print(dataset.isnull().sum())
```

Replacing Zeros with ‘NaN’ and Checking for Missing Values

06

Q6

Disease Outbreak Response System Condition (DORSCON) in Singapore is a colour-coded framework that shows the current disease situation. The framework provides us with general guidelines on what needs to be done to prevent and reduce the impact of infections. There are 4 statuses – Green, Yellow, Orange and Red, depending on the severity and spread of the disease. Which type of data does DORSCON belong to ?

(1) Categorical; (2) Ordinal; (3) Continuous; (4) Interval

07

Q7

A boxplot is a standardized way of displaying the dataset based on a five-number summary: the minimum, the maximum, _BLANK1_, and the first and third quartiles, where the number of data points that fall between the first and third quartiles amounts to _BLANK2_ percent of the total number of data on display.

Ans:

BLANK1: ???

BLANK2: ???

Example of Boxplot

```
# Tutorial 2: Example of Boxplot
import seaborn as sns
import statistics
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
```

Importing Libraries

```
dataPointsWOoutliers=[55,57,57,58,63,66,66,67,68,69,70,70,70,70,72,73,75,76,76,78,79,81.]
dataPointsWoutliers=[35,57,57,58,63,66,66,67,68,69,70,70,70,70,72,73,75,76,76,78,79,99.]]
df_combined = pd.DataFrame()
df_combined['normal'] = dataPointsWOoutliers
df_combined['outliers'] = dataPointsWoutliers
```

Stores both lists of data points in two columns: normal for data without outliers and outliers for data with outliers

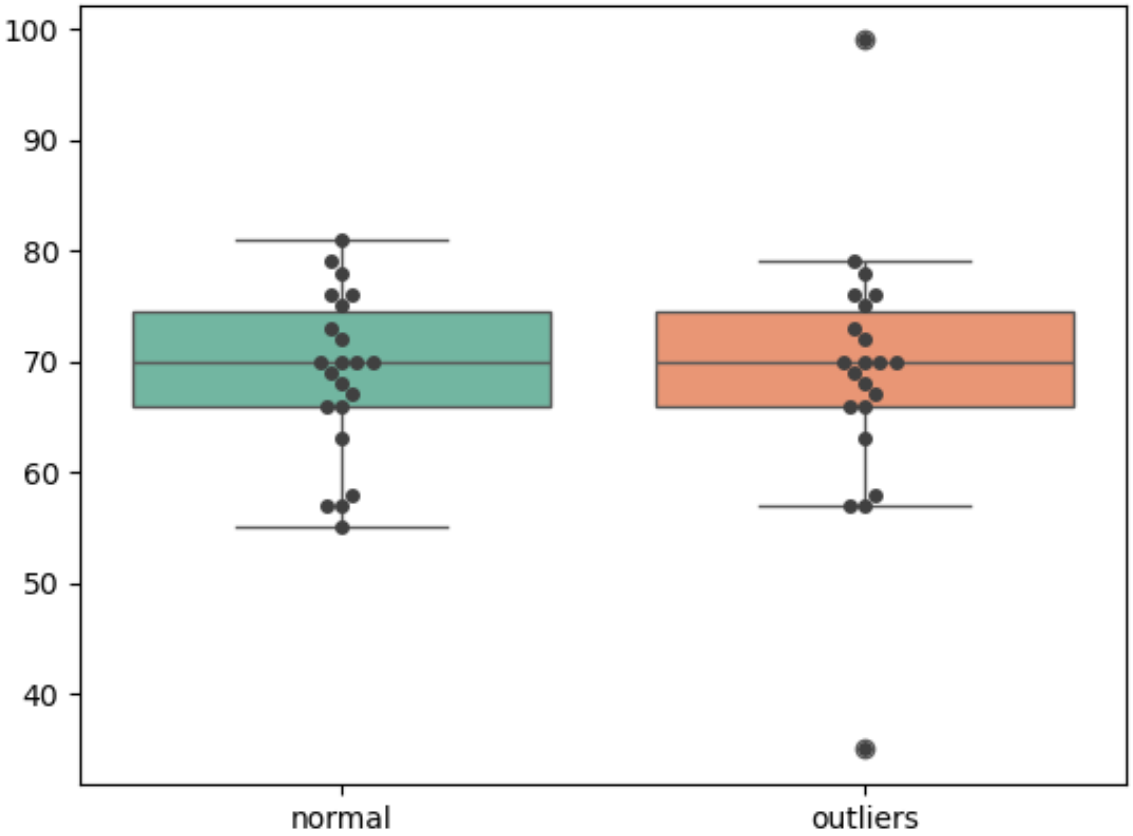
```
ax1 = sns.boxplot(data=df_combined, orient="v", palette="Set2")
```

A boxplot to show the distribution of the data, highlighting the median, quartiles, and any potential outliers.

```
ax1 = sns.swarmplot(data=df_combined, orient="v", color=(".25"))
plt.show()
```

Overlays a swarm plot on top of the boxplot

```
print(statistics.median(dataPointsWOoutliers))
print(statistics.median(dataPointsWoutliers))
```



THANK YOU