Practical 3

DATA PREPROCESSING

SET A

In [ ]:
```
#Q1.   $Data.csv
import pandas as pd
data=pd.read_csv("C:\\Data.csv")
```

In [ ]:
```
data
```

In [ ]:
```
#Q1.a
data.describe()
```

In [ ]:
```
#b.)
print("Size = {} \n Shape of DataFrame Object = {}\n Number of rows = {} \n Number of Columns = {}".
format(data.size, data.shape, data.shape[0], data.shape[1]))
```

In [ ]:
```
#c.)
print("\n first 3 rows from Dataset")
data.head(3)
```

In [ ]:
```
#Q2.
#Handling Missing values
data.fillna(data.mean())
```

In [ ]:
```
#Q3. a. Applying OneHot Encoding on Country Column
from sklearn.preprocessing import OneHotEncoder
enc = OneHotEncoder(handle_unknown='ignore')
enc_data= pd.DataFrame(enc.fit_transform(data[['Country']]).toarray())
enc_data
```

In [ ]:
```
data_merge= data.join(enc_data)
data_merge
```

In [ ]:
```
#Q3. b. Applying label encoding on purchased column
from sklearn.preprocessing import LabelEncoder
labelencoder = LabelEncoder()
data['Purchased'] = labelencoder.fit_transform(data['Purchased'])
data
```

In [ ]:
```
#The purchased labels are replaces by numbers 0 and 1,
# where 'No' is assigned 0, and 'Yes' is assigned 1.
```

SET B

In [ ]:
```
#Q1.
import pandas as pd
data=pd.read_csv("C:\\winequality-red.csv",sep=";")
```

In [ ]:
```
data.shape
```

In [ ]:
```
#Q2. Rescaling Data
import pandas, scipy, numpy
from sklearn import preprocessing
from sklearn.preprocessing import MinMaxScaler
array=data.values
#Separating data into input and output components
data_scaler=preprocessing.MinMaxScaler(feature_range=(0,1))
data_scaled = data_scaler.fit_transform(array)
print("\n Min Max Scaled Data \n \n ")
print(data_scaled.round(3))
```

This gives us values between 0 and 1.

Rescaling data proves of use with neural networks,

optimization algorithms and those that use distance measures like

k-nearest neighbors and weight inputs like regression.

In [ ]:
```
#Q3. Standardizing Data
from sklearn.preprocessing import StandardScaler
import scipy.stats as s
scaler=StandardScaler().fit(data)
std_data=scaler.transform(data)
print("\n Standardized Data \n ")
print(std_data)
print("\n Standardized Mean : ",-s.tmean(std_data).round(2))
print(" Standardized Standard Deviation : ",round(std_data.std(),2))
```

In [ ]:
```
#Q4. Normalizing Data
import numpy as np
import pandas as pd
import scipy.stats as s
from sklearn import preprocessing
norm_data=preprocessing.normalize(data,norm='l1')
print("\n Normalized Data \n ")
norm_data
```

In [ ]:
```
#Q5. Binarizing Data
binarized_data=preprocessing.Binarizer(threshold=0.0).fit(data).transform(data)
print("\n Binarized Data \n ")
binarized_data
```

SET C

In [ ]:
```
#Q1.
import pandas as pd
data=pd.read_csv("C:\\Student_bucketing.csv",sep=",")
```

In [ ]:
```
#Q2.
print("First 5 Rows of the dataset \n ")
data.head(5)
```

In [ ]:
```
#Q3.
import pandas as pd
data['bucket']=pd.cut(data['marks'],5,
                      labels=['Poor','Below_average','Average','Above_average','Excellent'])
data.head(10)
```