**Objective 1**

**[1]**

import pandas as pd

import numpy as np

from sklearn.preprocessing import LabelEncoder from sklearn.preprocessing import OneHotEncoder

*# reading the csv file*

***[2]***df=pd.read\_csv("adult.csv")

#

**[3]**

df

sex age race marital-status education nativecountry \

1. Male 39 White Never-married Bachelors United-

States

1. Male 50 White Married-civ-spouse Bachelors United-

States

1. Male 38 White Divorced HS-grad United-

States

1. Male 53 Black Married-civ-spouse 11th UnitedStates
2. Female 28 Black Married-civ-spouse Bachelors

Cuba

... ... ... ... ... ... .

..

1. Female 27 White Married-civ-spouse Assoc-acdm United-

States

1. Male 40 White Married-civ-spouse HS-grad United-

States

1. Female 58 White Widowed HS-grad United-

States

1. Male 22 White Never-married HS-grad United-

States

1. Female 52 White Married-civ-spouse HS-grad UnitedStates

workclass occupation salary-class 0 State-gov Adm-clerical 38000

1 Self-emp-not-inc Exec-managerial 47500 2 Private Handlers-cleaners 27500

1. Private Handlers-cleaners 27500
2. Private Prof-specialty 50000 ... ... ... ... 30157 Private Tech-support 32000 30158 Private Machine-op-inspct 45000 30159 Private Adm-clerical 38000
3. Private Adm-clerical 38000
4. Self-emp-inc Exec-managerial 47500

[30162 rows x 9 columns]

**[4]**

df.shape

(30162, 9)

*#Pandas groupby is used for grouping the data according to the categories and apply a function to the categories.*

*#It also helps to aggregate data efficiently.*

***[5]***

dataset=df.groupby('sex')

dataset.first()

age race marital-status education native-country workclass \ sex

Female 28 Black Married-civ-spouse Bachelors Cuba Private

Male 39 White Never-married Bachelors United-States State-gov

occupation salary-class sex Female Prof-specialty 50000 Male Adm-clerical 38000

*# Finding the values contained in the "female" group*

**[6]**

df1=dataset.get\_group('Female')

df1.describe()

age salary-class count 9782.000000 9782.000000 mean 36.883459 39642.608873 std 13.532427 6968.553378 min 17.000000 27500.000000 25% 25.250000 36000.000000

50% 35.000000 38000.000000 75% 46.000000 47500.000000 max 90.000000 50000.000000

*#Label Encoding is a popular encoding technique for handling categorical variables.*

*#In this technique, each label is assigned a unique integer based on alphabetical ordering. # creating initial dataframe*

***[7]***

Gender\_type = ('female','male')

Gender\_df = pd.DataFrame(Gender\_type, columns=['Gender\_type']) Gender\_df

Gender\_type

0 female 1 male

*# creating instance of labelencoder*

***[8]***

labelencoder = LabelEncoder()

*# Assigning numerical values and storing in another column*

***[9]***

Gender\_df['Gender\_type\_Cat']=labelencoder.fit\_transform(Gender\_df['Gender\_type'])

Gender\_df

Gender\_type Gender\_type\_Cat

0 female 0 1 male 1

*#With one-hot, we convert each categorical value into a new categorical column and assign*

*#a binary value of 1 or 0 to those columns. # creating instance of one-hot-encoder*

***[10]***

enc =OneHotEncoder(handle\_unknown='ignore')

*# passing Gender-type-cat column (label encoded values of Gender-type)*

***[11]***

enc\_df=pd.DataFrame(enc.fit\_transform(Gender\_df[['Gender\_type\_Cat']]).toarray ())

*# merge with main df Gender\_df on key values*

***[12]***

Gender\_df = Gender\_df.join(enc\_df) Gender\_df

Gender\_type Gender\_type\_Cat 0 1 0 female 0 1.0 0.0 1 male 1 0.0 1.0

**[13]**

Gender\_df.drop(['Gender\_type\_Cat'], axis = 1)

Gender\_type 0 1

0 female 1.0 0.0 1 male 0.0 1.0

**[14]**

dataset.describe()

age \ count mean std min 25% 50% 75% max sex Female 9782.0 36.883459 13.532427 17.0 25.25 35.0 46.0 90.0

Male 20380.0 39.184004 12.873243 17.0 29.00 38.0 48.0 90.0 salary-class

\

count mean std min 25%

50% sex

Female 9782.0 39642.608873 6968.553378 27500.0 36000.0

38000.0

Male 20380.0 39757.090285 6918.763492 27500.0 35000.0 38000.0

75% max sex Female 47500.0 50000.0 Male 47500.0 70000.0

**Objective 2**

*# reading the csv file*

***[1]***

*import pandas as pd*

*import numpy as np*

*from sklearn.preprocessing import MinMaxScaler*

data = pd.read\_csv("iris.csv") data

sepal\_length sepal\_width petal\_length petal\_width species 0 5.1 3.5 1.4 0.2 setosa

1. 4.9 3.0 1.4 0.2 setosa
2. 4.7 3.2 1.3 0.2 setosa
3. 4.6 3.1 1.5 0.2 setosa
4. 5.0 3.6 1.4 0.2 setosa.. ... ... ... ... ... 145 6.7 3.0 5.2 2.3 virginica
5. 6.3 2.5 5.0 1.9 virginica
6. 6.5 3.0 5.2 2.0 virginica
7. 6.2 3.4 5.4 2.3 virginica
8. 5.9 3.0 5.1 1.8 virginica

[150 rows x 5 columns]

**[2]**

data.head()

**[4]**

setosa = data['species'] == 'setosa' print(data[setosa].describe())

sepal\_length sepal\_width petal\_length petal\_width count 50.00000 50.000000 50.000000 50.000000 mean 5.00600 3.428000 1.462000 0.246000 std 0.35249 0.379064 0.173664 0.105386 min 4.30000 2.300000 1.000000 0.100000 25% 4.80000 3.200000 1.400000 0.200000

50% 5.00000 3.400000 1.500000 0.200000 75% 5.20000 3.675000 1.575000 0.300000 max 5.80000 4.400000 1.900000 0.600000

**[3]**

versicolor = data['species'] == 'versicolor' print(data[versicolor].describe())

sepal\_length sepal\_width petal\_length petal\_width count 50.000000 50.000000 50.000000 50.000000 mean 5.936000 2.770000 4.260000 1.326000 std 0.516171 0.313798 0.469911 0.197753 min 4.900000 2.000000 3.000000 1.000000 25% 5.600000 2.525000 4.000000 1.200000

50% 5.900000 2.800000 4.350000 1.300000 75% 6.300000 3.000000 4.600000 1.500000 max 7.000000 3.400000 5.100000 1.800000 virginica = data['species'] == 'virginica' print(data[virginica].describe())

sepal\_length sepal\_width petal\_length petal\_width count 50.00000 50.000000 50.000000 50.00000 mean 6.58800 2.974000 5.552000 2.02600 std 0.63588 0.322497 0.551895 0.27465 min 4.90000 2.200000 4.500000 1.40000 25% 6.22500 2.800000 5.100000 1.80000

50% 6.50000 3.000000 5.550000 2.00000 75% 6.90000 3.175000 5.875000 2.30000 max 7.90000 3.800000 6.900000 2.50000