############################### PROBLEM 1 ######################################

# -\*- coding: utf-8 -\*-

"""Untitled0.ipynb

Automatically generated by Colaboratory.

Original file is located at

https://colab.research.google.com/drive/19OaYr1U03xrFTeXl9FIk0m\_prQHvqKqL

1) Please take care of missing data present in the “Data.csv” file using python module

“sklearn.impute” and its methods, also collect all the data that has “Salary” less than “70,000”.

"""

#Loading neccessary libraries

import pandas as pd

from sklearn.impute import SimpleImputer

import numpy as np

#Load the data set

data=pd.read\_csv('/content/Data (2).csv')

#Print the dataset

data

##Checking for missing values

data.isnull().sum()

"""from above we can see that we have multiple missing values in each column.

To avaoid the data loss we will not drop the rows or columns. We will use imputation techniques to deal with missing data.

For numeric columns we will use mean/median imputer, and for categorical columns we will use mode imputer

"""

data.columns

data

#Mean Imputer for the numeric columns with no outliers

mean\_imputer = SimpleImputer(missing\_values=np.nan, strategy='mean')

data['age']= pd.DataFrame(mean\_imputer.fit\_transform(data[['age']]))

data['age'].isna().sum()

##Median Imputer is used if there are any outliers in the dataset

median\_imputer = SimpleImputer(missing\_values=np.nan, strategy='median')

data['Salaries']= pd.DataFrame(mean\_imputer.fit\_transform(data[['Salaries']]))

data['Salaries'].isna().sum()

mode\_imputer = SimpleImputer(missing\_values = np.nan, strategy = 'm# Mode Imputer for categorical columns

ost\_frequent')

data['Position'] = pd.DataFrame(mode\_imputer.fit\_transform(data[['Position']]))

data['State'] = pd.DataFrame(mode\_imputer.fit\_transform(data[['State']]))

data['MaritalDesc'] = pd.DataFrame(mode\_imputer.fit\_transform(data[['MaritalDesc']]))

data['EmploymentStatus'] = pd.DataFrame(mode\_imputer.fit\_transform(data[['EmploymentStatus']]))

data['CitizenDesc'] = pd.DataFrame(mode\_imputer.fit\_transform(data[['CitizenDesc']]))

data['Race'] = pd.DataFrame(mode\_imputer.fit\_transform(data[['Race']]))

data['Department'] = pd.DataFrame(mode\_imputer.fit\_transform(data[['Department']]))

data.isnull().sum() # all Sex, MaritalDesc records replaced by mode

# Constant Value Imputer, it will replace all nan values with constant value 'F'

constant\_imputer = SimpleImputer(missing\_values = np.nan, strategy = 'constant', fill\_value = 'F')

# fill\_value can be used for numeric or non-numeric values

data["Sex"] = pd.DataFrame(constant\_imputer.fit\_transform(data[["Sex"]]))

data.isnull().sum()

# Random Imputer can be used on numeric data. In our data we only have 2 numeric columns therefore we will not perform random imputation technique

#from feature\_engine.imputation import RandomSampleImputer

#random\_imputer = RandomSampleImputer(['age'])

#data["age"] = pd.DataFrame(random\_imputer.fit\_transform(data[["age"]]))

#data["age"].isna().sum() # all records replaced by median

#4. For the dataset “Indian\_cities”,

#a) Find out top 10 states in female-male sex ratio

#b) Find out top 10 cities in total number of graduates

#c) Find out top 10 cities and their locations in respect of total effective\_literacy\_rate.s

#Importing the dataset

import pandas as pd

indian\_cities = pd.read\_csv('C:/Users/user/Documents/Study Material/360digitmg/Python/Assignments/Python Problem Statements/Indian\_cities.csv')

#a.

gp\_s = indian\_cities.groupby('state\_name').sum() #Grouping the data based on state first

gp\_s

st = gp\_s[['sex\_ratio']].sort\_values('sex\_ratio', ascending=False) #After grouping the states we check by sorting the values of the column sexratio in descending order.

st.head(10) #printing the top ten states.

type(st)

#b.

#Same logic as Q1.a

gp\_c= indian\_cities.groupby('name\_of\_city').sum()

city = gp\_c[['total\_graduates']].sort\_values('total\_graduates', ascending=False)

city.head(10)

#c.

# Using nlargest function we get the top 10 records and then specify the column we want to print.

indian\_cities.nlargest(10,['effective\_literacy\_rate\_total'])[['name\_of\_city','location']]

#5. For the data set “Indian\_cities”

#a) Construct histogram on literates\_total and comment about the inferences

#b) Construct scatter plot between male graduates and female graduates

#a)

import matplotlib.pyplot as plt

plt.hist(indian\_cities.literates\_total)

#Inferences from histogram:

#• The data represented on the histogram is not symmetrical.

#• It has a long positive tail. It has a positive skewness.

#• Approximately more than 90% of the data is confined in the range 56998 to 416998.

#• Outliers are present in the dataset.

#b)

#Using matplotlib.pyplot library we create thee scatter plot.

# x= indian\_cities.male\_graduates

#y = indian\_cities.female\_graduates

plt.scatter(indian\_cities.male\_graduates, indian\_cities.female\_graduates, edgecolors=('red'))

#6. For the data set “Indian\_cities”

#a) Construct Boxplot on total effective literacy rate and draw inferences

#b) Find out the number of null values in each column of the dataset and delete them.

#a)

import matplotlib.pyplot as plt #Importing the library to create visualizations

plt.boxplot(indian\_cities.effective\_literacy\_rate\_total)

#Inferences from boxplot:

#• The data represented on the boxplot is not symmetrical.

#• It has negative skewness as the median of the data is close to the upper end of the boxplot.

#• Outliers are present in the dataset beyond the lower whisker.

#• The median of the data is approximately 85.

#• The spread of the data is not much and majority of the data is confined between the range 80% to 90%.

#b)

import numpy as np

indian\_cities.isnull().sum() #There are no missing values in the dataset.

indian\_cities.describe()

indian\_cities.dropna(inplace = True) #We can drop the na values using the