Assignment 4

Ravi Pant
20214509
CSE6C

Question 1: . Working with the data dictionary, list, and DataFrame:

In [ ]:

```
""" 1. Prepare dataFrame with the following lists_marks detail in different subjects:
Columns_list =[‘Reg_no’,’Name’,’Subject1’, ’Subject2’ ,’Subject3’ ,’Subject4’]
Rows_list=
[
 [2022001,’Abhijeet’,65,65,69,81], [2022002,’Ajeet’,75,75,90,81],
[2022003,’Amit’,75,05,69,87], [2022004,’Ranjeet’,55,65,79,91],
[2022005,’Santosh’,85,85,60,61], [2022006,’Satyam’,73,75,68,51],
[2022007,’Shivam’,85,85,50,40], [2022009,’Shyam’,75,65,69,81] ,
[2022010,’Yash’,85,75,89,61]
]
"""
import numpy as np
import pandas as pd

Columns_list = ['Reg_no', 'Name', 'Subject1', 'Subject2', 'Subject3', 'Subject4']
Rows_list = [
    [2022001, 'Abhijeet', 65, 65, 69, 81],
    [2022002, 'Ajeet', 75, 75, 90, 81],
    [2022003, 'Amit', 75, 5, 69, 87],
    [2022004, 'Ranjeet', 55, 65, 79, 91],
    [2022005, 'Santosh', 85, 85, 60, 61],
    [2022006, 'Satyam', 73, 75, 68, 51],
    [2022007, 'Shivam', 85, 85, 50, 40],
    [2022009, 'Shyam', 75, 65, 69, 81],
    [2022010, 'Yash', 85, 75, 89, 61]
]

# Creating a DataFrame from the data
df = pd.DataFrame(Rows_list, columns=Columns_list)

# Displaying the DataFrame
print(df)
```

```
    Reg_no      Name  Subject1  Subject2  Subject3  Subject4 0
2022001  Abhijeet        65        65        69        81
       1      2022002     Ajeet        75        75        90        81
       2      2022003      Amit        75         5        69        87
       3      2022004   Ranjeet        55        65        79        91
       4      2022005   Santosh        85        85        60        61
       5      2022006    Satyam        73        75        68        51
       6      2022007    Shivam        85        85        50        40
       7      2022009     Shyam        75        65        69        81
       8      2022010      Yash        85        75        89        61 In [
```

```
#2. Add column name ‘Total’ with initially blank entries ‘ ’ against each cell.

df['Total'] = ''
print(df)
```

```
]:
```

```
    Reg_no      Name  Subject1  Subject2  Subject3  Subject4 Total 0
2022001  Abhijeet        65        65        69        81
       1      2022002     Ajeet        75        75        90        81
       2      2022003      Amit        75         5        69        87
       3      2022004   Ranjeet        55        65        79        91
       4      2022005   Santosh        85        85        60        61
       5      2022006    Satyam        73        75        68        51
       6      2022007    Shivam        85        85        50        40
       7      2022009     Shyam        75        65        69        81
       8      2022010      Yash        85        75        89        61
```

```
#3 Fill the Total column with values by taking the mark sum in all subjects
df['Total'] = df[['Subject1', 'Subject2', 'Subject3', 'Subject4']].sum(axis=1) #axi
print(df)
```

```
In [ ]:
```

```
    Reg_no      Name  Subject1  Subject2  Subject3  Subject4  Total 0
2022001  Abhijeet        65        65        69        81    280
       1      2022002     Ajeet        75        75        90        81
321
       2      2022003      Amit        75         5        69        87
236
       3      2022004   Ranjeet        55        65        79        91
290
       4      2022005   Santosh        85        85        60        61
291
       5      2022006    Satyam        73        75        68        51
267
       6      2022007    Shivam        85        85        50        40
260
       7      2022009     Shyam        75        65        69        81
290
```

```
      8     2022010      Yash        85         75         89         61
```

```
""" 3.
Add New Column ‘Grade’ with nominal values{A, B, C, D, E} according to total marks
the formula as given below: Grade A when total marks >=90
Grade B when total marks >=80 and <90
Grade C when total marks >=70 and <80
Grade D when total marks >=50 and <70
Grade E when total marks <50
""" def
getGrade(total_marks):
if total_marks/4 >= 90:
        return 'A'     elif
total_marks/4 >= 80:
        return 'B'     elif
total_marks/4 >= 70:
        return 'C'     elif
total_marks/4 >= 50:
        return 'D'     else:
return 'E' df['Grade'] =
df['Total'].apply(getGrade) print(df)
```

```
310 In [ ]:
    Reg_no      Name  Subject1  Subject2  Subject3  Subject4  Total Grade 0
2022001  Abhijeet        65        65        69        81    280      C
       1      2022002      Ajeet        75        75        90        81
321      B
       2      2022003       Amit        75         5        69        87
236      D
       3      2022004    Ranjeet        55        65        79        91
290      C
       4      2022005    Santosh        85        85        60        61
291      C
       5      2022006     Satyam        73        75        68        51
267      D
       6      2022007     Shivam        85        85        50        40
260      D
       7      2022009      Shyam        75        65        69        81
290      C
       8      2022010       Yash        85        75        89        61
```

```
#5. Prepare subset with [[‘Reg_no’, ’Name’, ’Grade’]
subset = df[['Reg_no', 'Name', 'Grade']] print(subset)
```

```
310      CIn [ ]:
```

```
    Reg_no      Name Grade
0  2022001  Abhijeet     C
       1      2022002      Ajeet     B
       2      2022003       Amit     D
       3      2022004    Ranjeet     C
       4      2022005    Santosh     C
       5      2022006     Satyam     D
       6      2022007     Shivam     D
       7      2022009      Shyam     C
       8      2022010       Yash     CIn [ ]:
```

```
#6. Prepare a list of students according to grades in the separate data file.
grade_A_student = df[df['Grade']=='A'] grade_B_student
= df[df['Grade']=='B'] grade_C_student =
df[df['Grade']=='C'] grade_D_student =
df[df['Grade']=='D']

student_list_according_to_Grade = pd.concat([grade_A_student, grade_B_student, grad
student_list_according_to_Grade.to_csv('student_by_grade.csv',index=False )
print(student_list_according_to_Grade)
```

```
Reg_no      Name  Subject1  Subject2  Subject3  Subject4  Total Grade 1
2022002     Ajeet        75        75        90        81    321      B
0  2022001  Abhijeet        65        65        69        81    280     C
3  2022004   Ranjeet        55        65        79        91    290     C
4  2022005   Santosh        85        85        60        61    291     C
7  2022009     Shyam        75        65        69        81    290     C
8  2022010      Yash        85        75        89        61    310     C
2  2022003      Amit        75         5        69        87    236     D
5  2022006    Satyam        73        75        68        51    267     D
6  2022007    Shivam        85        85        50        40    260     D
```

Question 2: . Working with Pandas CSV reading\writing and preparing training\testing dataset:

In [ ]:

```
#1. Read weatherNumeric.csv file and assigned it to object df.
import pandas as pd df = pd.read_csv('weather-numeric.csv')
print(df)
```

```
outlook  temperature  humidity  windy play 0
sunny           85        85  False   no
       1     sunny          80        90   True   no
       2     overcast          83        86  False  yes
       3     rainy          70        96  False  yes
       4     rainy          68        80  False  yes
       5     rainy          65        70   True   no
       6     overcast          64        65   True  yes
       7     sunny          72        95  False   no
       8     sunny          69        70  False  yes
       9     rainy          75        80  False  yes
       10    sunny          75        70   True  yes
       11    overcast          72        90   True  yes
       12    overcast          81        75  False  yes
       13    rainy          71        91   True   noIn [ ]:
```

```
#2. Select the last column as a class and assign it to object Y
Y = df.iloc[:,-1] print(Y)
```

```
0      no
1      no
2      yes
3      yes
4      yes
5      no
6      yes
7      no
8      yes
9      yes
10     yes
11     yes
12     yes
13     no
Name: play, dtype: object
```

In [ ]:

```
#3. Select all remaining columns other than the last and assigned them to object X X
= df.iloc[:, :-1] print(X)
```

```
outlook  temperature  humidity  windy 0
sunny           85        85  False
1      sunny          80        90   True
2      overcast          83        86  False
3      rainy          70        96  False
4      rainy          68        80  False
5      rainy          65        70   True
6      overcast          64        65   True
```

```
7       sunny          72        95  False
8       sunny          69        70  False
9       rainy          75        80  False
10      sunny          75        70   True
11      overcast          72        90   True
12      overcast          81        75  False
```

```
""" 4. Split entire both X and Y into training: 80%, testing:20% parts and assigned
it X_test, Y_train, and Y_test respectively
""" from sklearn.model_selection import

train_test_split

X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_sta

print("X_train")
print(X_train)
print()
print("X_test")
print(X_test)
print()
print("Y_train")
print(Y_train)
print()
print("Y_test")
print(Y_test)
```

```
13      rainy          71        91   True
```

```
In [ ]:
```

```
X_train
     outlook  temperature  humidity  windy 12
overcast           81        75  False
5      rainy           65        70   True
8      sunny           69        70  False
2   overcast           83        86  False
1      sunny           80        90   True
13     rainy           71        91   True
4      rainy           68        80  False
7      sunny           72        95  False
10     sunny           75        70   True
3      rainy           70        96  False
6   overcast           64        65   True

X_test
     outlook  temperature  humidity  windy 9
rainy           75        80  False
11  overcast           72        90   True
0      sunny           85        85  False

Y_train
12    yes
5      no
8     yes
2     yes
1      no
13     no
4     yes
7      no
10    yes
3     yes
6     yes
Name: play, dtype: object

Y_test
9     yes
11    yes
0      no
Name: play, dtype: object
```

In [ ]:

```
"""
5.Prepare five different training\testing pairs and use pandas to_csv()
to save these into file names: train1,test1, train2,test2, train3,test3, train4,tes
"""

for i in range(1, 6):
  X_train, X_test = train_test_split(X, test_size=0.2, random_state=i)

  X_train.to_csv(f'X_train{i}.csv', index=False)
X_test.to_csv(f'X_test{i}.csv', index=False)

  print(X_train)
print()
print(Y_train)
print()
```

```
     outlook  temperature  humidity  windy 2
overcast           83        86  False
10     sunny           75        70   True
4      rainy           68        80  False
1      sunny           80        90   True
12  overcast           81        75  False
0      sunny           85        85  False
13     rainy           71        91   True
9      rainy           75        80  False
8      sunny           69        70  False
11  overcast           72        90   True
5      rainy           65        70   True

12    yes
5      no
8     yes
2     yes
1      no
13     no
4     yes
7      no
10    yes
3     yes
6     yes
Name: play, dtype: object

     outlook  temperature  humidity  windy 0
sunny           85        85  False
9      rainy           75        80  False
3      rainy           70        96  False
1      sunny           80        90   True
10     sunny           75        70   True
7      sunny           72        95  False
12  overcast           81        75  False
2   overcast           83        86  False
6   overcast           64        65   True
13     rainy           71        91   True
8      sunny           69        70  False

12    yes
5      no
8     yes
2     yes
1      no
13     no
4     yes
7      no
10    yes
3     yes
6     yes
Name: play, dtype: object

     outlook  temperature  humidity  windy 2
overcast           83        86  False
13     rainy           71        91   True
6   overcast           64        65   True 5      rainy           65
70   True
```

```
0      sunny           85        85  False
11  overcast           72        90   True
12  overcast           81        75  False
3      rainy           70        96  False
9      rainy           75        80  False
8      sunny           69        70  False
10     sunny           75        70   True

12    yes
5      no
8     yes
2     yes
1      no
13     no
4     yes
7      no
10    yes
3     yes
6     yes
Name: play, dtype: object

     outlook  temperature  humidity  windy 9
rainy           75        80  False
6   overcast           64        65   True
13     rainy           71        91   True
2   overcast           83        86  False
0      sunny           85        85  False
12  overcast           81        75  False
8      sunny           69        70  False
1      sunny           80        90   True
5      rainy           65        70   True
7      sunny           72        95  False
10     sunny           75        70   True

12    yes
5      no
8     yes
2     yes
1      no
13     no
4     yes
7      no
10    yes
3     yes
6     yes
Name: play, dtype: object

     outlook  temperature  humidity  windy 2
overcast           83        86  False
10     sunny           75        70   True
13     rainy           71        91   True
11  overcast           72        90   True
4      rainy           68        80  False
8      sunny           69        70  False
9      rainy           75        80  False0      sunny           85
       85  False
12  overcast           81        75  False
```

```
6      overcast          64        65   True
3      rainy          70        96  False

12    yes
5      no
8     yes
2     yes
1      no
13     no
4     yes
7      no
10     yes
3     yes
6      yes
Name: play, dtype: object
```

In [ ]:

```
"""
Q3.
  Analysis with result dataset: The datasheet.csv file contains sensitivity score
results generated by Random Forest Tree (RFT) classifiers on 24 equivalent re-sam
of a dataset by 18 different resampling methods. The first column of datasheet.cs
represents the dataset and all other remaining columns represent sensitivity valu
by RFT on different resampling methods. Read datasheet.csv file from the director
and do the following analysis.
  1.Assigned rank to each resampling method, corresponding
  to each sensitivity score on each dataset row by using the following ranking stra
    a.Rank(1:Higher sensitivity score and so on)
    b.Assign the same rank for the two or more similar sensitivity scores
    c.Rank range(1 to 18 in case all sensitivity values in a row are distinct) """

import pandas as pd
data = {
    'Dataset': ['Pima', 'Glass', 'Wisconsin'],
    'ENN': [0.9552, 0.9773, 0.7864],
    'AllKNN': [0.9452, 0.9773, 0.7864],
    'SMOTE': [0.9352, 0.9673, 0.7864]
}

# Convert the dictionary to a DataFrame df
= pd.DataFrame(data)

# Save the DataFrame to a CSV file
df.to_csv('datasheet.csv', index=False) print(df)

# Question 1: Assigned rank to each resampling method, corresponding to each sensit

# Define a function to assign ranks based on the specified strategy def
assign_ranks(row):
    # Sort the sensitivity scores in descending order
sorted_row = row.sort_values(ascending=False)
    # Initialize a dictionary to store ranks
    rank_dict = {}     rank = 1
prev_score = None     for method, score in
sorted_row.items():
        # Check if the score is the same as the previous score
if score != prev_score:             rank = rank
rank_dict[method] = rank         prev_score = score
rank += 1
    return pd.Series(rank_dict)

# Apply the function row-wise to assign ranks to each method's sensitivity score
rank_df = df.drop('Dataset', axis=1).apply(assign_ranks, axis=1) print()
print(rank_df)
```

```
    Dataset     ENN   AllKNN   SMOTE 0
Pima  0.9552  0.9452  0.9352
1      Glass  0.9773  0.9773  0.9673
2      Wisconsin  0.7864  0.7864  0.7864

   ENN  AllKNN  SMOTE
0    1       2      3
1    1       2      3
2    1       2      3
```

```
# Question 2: Compute the average sensitivity rank of each resampling method on the
average_ranks = rank_df.mean()

print("\nAverage Ranks:") print(average_ranks)
```

In [ ]:

```
Average Ranks:
ENN       1.0
AllKNN    2.0
SMOTE     3.0
dtype: float64
```

In [ ]:

```
# Question 3: Identify lowest and highest performing methods
lowest_performer = average_ranks.idxmax() highest_performer =
average_ranks.idxmin()

# Print the result
print("\nThe highest average rank is {:.0f}, therefore {} is the highest performer
```

```
The highest average rank is 1, therefore ENN is the highest performer and SMOTE is t
he lowest performer.
```