NAME: SHUBHAM TAKANKHAR
CLASS: TY MCA
ROLL NO: 54
GR. NO: 119C0046

Read and Write CSV file(at first create CSV file using note pad ,then read same file and write to same file)

In [ ]:
```
#reading writing to csv file w/o csv package

f = open("texts.csv","w")

f.write("name,class,college")
f.write("\n")
f.write("shubham,mcaty,vit pune")

f.close()

f = open("texts.csv","r")

for line in f:
  print(line)

f.close()
```

```
name,class,college

shubham,mcaty,vit pune
```

Read and Write CSV file(download CSV file from UCI/Kaggal ,then read same file and write to same file)

In [ ]:
```
# download athletes of olympic 2021 dataset from Kaggle

import csv

with open("Athletes.csv",encoding='utf-8-sig') as athletess:
  athletes_dict = csv.DictReader(athletess)
  for athlete in athletes_dict:
    if athlete["NOC"] == "India" and athlete["Discipline"] == "Badminton":
      print(athlete["Name"] + "  >  " + athlete["Discipline"] )

athletes.close()

with open("Athletes.csv",encoding='utf-8-sig',mode = "a+") as athletes:
  athletes_writer = csv.writer(athletes)
  athletes_writer.writerow(["Shubham Takankhar","India","Badminton"])

  athletes_dict = csv.DictReader(athletes)
  for athlete in athletes_dict:
    if athlete["NOC"] == "India" and athlete["Discipline"] == "Badminton":
      print(athlete["Name"] + "  >  " + athlete["Discipline"] )

athletes.close()
```

```
B. Sai Praneeth  >  Badminton
PUSARLA V. Sindhu  >  Badminton
SHETTY Chirag  >  Badminton
Shubham Takankhar  >  Badminton
```

Apply 5 data preprocessing techniques on csv file and display file after preprocessing

DataSet used: Choclate Dataset with a bit of adaptation with ~1700 items http://flavorsofcacao.com/chocolate_database.html (http://flavorsofcacao.com/chocolate_database.html)

1. replace missing value by mean
2. replace missing value by interpolation
3. binning by mean
4. binning by boundary
5. normalization

1. replace missing value by mean

processing on the *Rating Column*

In [73]:
```
import pandas as pd
import numpy as np

df = pd.read_csv("flavors_of_cacao.csv")

df.head(10)
```

Out[73]:

	Company\n(Maker-if known)	Specific Bean Origin\nor Bar Name	REF	Review\nDate	Cocoa\nPercent	Company\nLocation	Rating	Bean\nType	Broad Bean\nOrigin
0	A. Morin	Agua Grande	1876	2016	63%	France	3.75		Sao Tome
1	A. Morin	Kpime	1676	2015	70%	France	2.75		Togo
2	A. Morin	Atsane	1676	2015	70%	France	3.00		Togo
3	A. Morin	Akata	1680	2015	70%	France	3.50		Togo
4	A. Morin	Quilla	1704	2015	NaN	France	3.50		Peru
5	A. Morin	Carenero	1315	2014	70%	France	2.75	Criollo	Venezuela
6	A. Morin	Cuba	1315	2014	70%	France	3.50		Cuba
7	A. Morin	Sur del Lago	1315	2014	NaN	France	NaN	Criollo	Venezuela
8	A. Morin	Puerto Cabello	1319	2014	70%	France	3.75	Criollo	Venezuela
9	A. Morin	Pablino	1319	2014	70%	France	4.00		Peru

In [89]:
```
print("No.of missing values in Cocoa Rating column:" , df["Rating"].isnull().sum())
# as it is less than 10% or less we can fill or impute mean
```

```
No.of missing values in Cocoa Rating column: 11
```

In [92]:
```
#filling Missing values with averages

df["Rating"] = round(df["Rating"].fillna(df["Rating"].mean()),2)

df.head(10)
```

Out[92]:

	Company\n(Maker-if known)	Specific Bean Origin\nor Bar Name	REF	Review\nDate	Cocoa\nPercent	Company\nLocation	Rating	Bean\nType	Broad Bean\nOrigin
0	A. Morin	Agua Grande	1876	2016	63%	France	3.75		Sao Tome
1	A. Morin	Kpime	1676	2015	70%	France	2.75		Togo
2	A. Morin	Atsane	1676	2015	70%	France	3.00		Togo
3	A. Morin	Akata	1680	2015	70%	France	3.50		Togo
4	A. Morin	Quilla	1704	2015	NaN	France	3.50		Peru
5	A. Morin	Carenero	1315	2014	70%	France	2.75	Criollo	Venezuela
6	A. Morin	Cuba	1315	2014	70%	France	3.50		Cuba
7	A. Morin	Sur del Lago	1315	2014	NaN	France	3.19	Criollo	Venezuela
8	A. Morin	Puerto Cabello	1319	2014	70%	France	3.75	Criollo	Venezuela
9	A. Morin	Pablino	1319	2014	70%	France	4.00		Peru

In [91]:
```
#verify
print("No.of missing values in Cocoa Rating column:" , df["Rating"].isnull().sum())
```

```
No.of missing values in Cocoa Rating column: 0
```

2.replace missing value by interpolation

processing on the Cocoa Percent Column

In [117]:
```
# removing percentage sign and convert string to float
df['Cocoa\nPercent'] = df['Cocoa\nPercent'].str.replace('%','').str.strip()

df['Cocoa\nPercent'] = pd.to_numeric(df['Cocoa\nPercent'],errors='coerce')

df['Cocoa\nPercent'].head()
```

Out[117]:
```
0    63.0
1    70.0
2    70.0
3    70.0
4     NaN
Name: Cocoa\nPercent, dtype: float64
```

In [123]:
```
#calculating no of missing  value which is somewhere around 27 in the entire dataset
df['Cocoa\nPercent'].isnull().sum()
```

Out[123]:
```
0
```

In [122]:
```
#using linear interpolation to fill missing values in the percent column

interpolated_data = pd.Series(df['Cocoa\nPercent']).interpolate()

df['Cocoa\nPercent'] = interpolated_data

df['Cocoa\nPercent'].head(10)
```

Out[122]:
```
0    63.0
1    70.0
2    70.0
3    70.0
4    70.0
5    70.0
6    70.0
7    70.0
8    70.0
9    70.0
Name: Cocoa\nPercent, dtype: float64
```

In [124]:
```
# verifing
df['Cocoa\nPercent'].isnull().sum()
```

Out[124]:
```
0
```

3. binning by mean

In [ ]:

In [133]:
```
#binning according to the ratings

min_rating = df['Rating'].min()

max_rating = df['Rating'].max()

#creating three bins:  low ratings --- average rating --- high rating   as ['awful','decent','excellent']

bins = np.linspace(min_rating,max_rating,4)

print("Bins: ",bins)

labels = ['awful','decent','excellent']
```

```
Bins:  [1.         2.33333333 3.66666667 5.        ]
```

In [139]:
```
#generating the quality column based on the boundary of ratings

df['Quality'] = pd.cut(df['Rating'],bins=bins,labels = labels,include_lowest=True )

df.head()
```

Out[139]:

	Company\n(Maker-if known)	Specific Bean Origin\nor Bar Name	REF	Review\nDate	Cocoa\nPercent	Company\nLocation	Rating	Bean\nType	Broad Bean\nOrigin	Quality
0	A. Morin	Agua Grande	1876	2016	63.0	France	3.75		Sao Tome	excellent
1	A. Morin	Kpime	1676	2015	70.0	France	2.75		Togo	decent
2	A. Morin	Atsane	1676	2015	70.0	France	3.00		Togo	decent
3	A. Morin	Akata	1680	2015	70.0	France	3.50		Togo	decent
4	A. Morin	Quilla	1704	2015	70.0	France	3.50		Peru	decent

4. binning by boundary

In [176]:
```
#binning by boundaries of Percent

data = np.sort(df['Cocoa\nPercent'][:900]) #considering only first 900 values because of large dataset

b1=np.zeros((900,3))

for i in range (0,900,3):
  k=int(i/3)
  for j in range (3):
    if (data[i+j]-data[i]) < (data[i+2]-data[i+j]):
      b3[k,j]=data[i]
    else:
      b3[k,j]=data[i+2]

print("boundary bins\n",b3[:5])
```

```
boundary bins
 [[42. 55. 55.]
 [55. 55. 55.]
 [55. 55. 55.]
 [55. 55. 55.]
 [55. 58. 58.]]
```

5. Normalization

In [184]:
```
# copy the dataframe
df_min_max_scaled = df.copy()

# apply normalization techniques by cocoa percent
column = 'Cocoa\nPercent'
df_min_max_scaled[column] = (df_min_max_scaled[column] - df_min_max_scaled[column].min()) / (df_min_max_scaled[column].max() - df_min_max_scaled[column].min())

# view normalized percentages
display(df_min_max_scaled['Cocoa\nPercent'])
```

```
0       0.362069
1       0.482759
2       0.482759
3       0.482759
4       0.482759
          ...
1790    0.482759
1791    0.396552
1792    0.396552
1793    0.344828
1794    0.396552
Name: Cocoa\nPercent, Length: 1795, dtype: float64
```