In [1]:
```
import numpy as np
import pandas as pd
import random
```

In [2]:
```
df = pd.read_csv("Avocado.csv")
df
```

Out[2]:

	Date	AveragePrice	Total Volume	4046	4225	4770	Total Bags	Small Bags	Large Bags	XLarge Bags	type	year	region
0	27-12-2015	1.33	64236.62	1036.74	54454.85	48.16	8696.87	8603.62	93.25	0.0	conventional	2015	Albany
1	20-12-2015	1.35	54876.98	674.28	44638.81	58.33	9505.56	9408.07	97.49	0.0	conventional	2015	Albany
2	13-12-2015	0.93	118220.22	794.70	109149.67	130.50	8145.35	8042.21	103.14	0.0	conventional	2015	Albany
3	06-12-2015	1.08	78992.15	1132.00	71976.41	72.58	5811.16	5677.40	133.76	0.0	conventional	2015	Albany
4	29-11-2015	1.29	51039.60	941.48	43838.39	75.78	6183.95	5986.26	197.69	0.0	conventional	2015	Albany
...	...	...	...	...	...	...	...	...	...	...	...	...	...
18245	28-01-2018	1.71	13888.04	1191.70	3431.50	0.00	9264.84	8940.04	324.80	0.0	organic	2018	WestTexNewMexico
18246	21-01-2018	1.87	13766.76	1191.92	2452.79	727.94	9394.11	9351.80	42.31	0.0	organic	2018	WestTexNewMexico
18247	14-01-2018	1.93	16205.22	1527.63	2981.04	727.01	10969.54	10919.54	50.00	0.0	organic	2018	WestTexNewMexico
18248	07-01-2018	1.62	17489.58	2894.77	2356.13	224.53	12014.15	11988.14	26.01	0.0	organic	2018	WestTexNewMexico
18249	18-03-2018	1.56	15896.38	2055.35	1499.55	0.00	12341.48	12114.81	226.67	0.0	organic	2018	WestTexNewMexico

18250 rows × 13 columns

Q-1 Select a subset of relevant attributes from the given dataset that are necessary to know about the total volume of avocados with product lookup codes (PLU) 4046, 4225, 4770) which are of organic type. (Use AVOCADO dataset)

In [4]:
```
new_df = df[['AveragePrice', 'type']]
new_df.loc[df.type=='organic']
```

Out[4]:

	AveragePrice	type
9126	1.83	organic
9127	1.89	organic
9128	1.85	organic
9129	1.84	organic
9130	1.94	organic
...	...	...
18245	1.71	organic
18246	1.87	organic
18247	1.93	organic
18248	1.62	organic
18249	1.56	organic

9124 rows × 2 columns

In [ ]:

2. Discard all duplicate entries in the dataset given and fill all the missing values in the attribute “AveragePrice” as 1.25. Also print the size of the dataset before and after removing duplicates. (Use Trail dataset)

In [5]:
```
# thi is duplicate entry
df = pd.read_csv("Trail.csv")
df[df.duplicated(keep = 'last')]
```

Out[5]:

	Date	AveragePrice	Total Volume	4046	4225	4770	Total Bags	Small Bags	Large Bags	XLarge Bags	type	year	region
0	20-12-2015	1.35	54876.98	674.28	44638.81	58.33	9505.56	9408.07	97.49	0.0	conventional	2015	Albany
1	13-12-2015	0.93	118220.22	794.70	109149.67	130.50	8145.35	8042.21	103.14	0.0	conventional	2015	Albany
2	06-12-2015	1.08	78992.15	1132.00	71976.41	72.58	5811.16	5677.40	133.76	0.0	conventional	2015	Albany
3	22-11-2015	NaN	55979.78	1184.27	48067.99	43.61	6683.91	6556.47	127.44	0.0	conventional	2015	Albany
4	20-12-2015	1.35	54876.98	674.28	44638.81	58.33	9505.56	9408.07	97.49	0.0	conventional	2015	Albany
6	06-12-2015	1.21	78992.15	1132.00	71976.41	72.58	5811.16	5677.40	133.76	0.0	organic	2015	xxxx
7	29-11-2015	1.21	51039.60	941.48	43838.39	75.78	6183.95	5986.26	197.69	0.0	organic	2015	yyyy

In [6]:
```
# size before dropping duplicates..
df.shape
```

Out[6]:
```
(202, 13)
```

In [7]:
```
# size after dropping duplicates
df.drop_duplicates(inplace=True, keep='last')
df.shape
```

Out[7]:
```
(195, 13)
```

In [8]:
```
# fill all the missing values in the attribute “AveragePrice” as 1.25.
col = []
count = 0
for i in range(len(df['AveragePrice'])):
    flag = 1
    for x in str(df['AveragePrice'].iloc[i]):
        if( not (x.isdigit() or x=='.') ):
            col.append(1.25)
            flag = 0
            count += 1
            break

    if(flag):
        col.append( round(float(df['AveragePrice'].iloc[i]), 2) )

df['AveragePrice'] = col
df.head(20)
```

Out[8]:

	Date	AveragePrice	Total Volume	4046	4225	4770	Total Bags	Small Bags	Large Bags	XLarge Bags	type	year	region
5	13-12-2015	1.21	118220.22	794.70	109149.67	130.50	8145.35	8042.21	103.14	0.0	organic	2015	xxxx
8	22-11-2015	1.21	55979.78	1184.27	48067.99	43.61	6683.91	6556.47	127.44	0.0	organic	2015	yyyy
9	06-12-2015	1.08	78992.15	1132.00	71976.41	72.58	5811.16	5677.40	133.76	0.0	conventional	2015	Albany
10	22-11-2015	1.25	55979.78	1184.27	48067.99	43.61	6683.91	6556.47	127.44	0.0	conventional	2015	Albany
11	06-12-2015	1.21	78992.15	1132.00	71976.41	72.58	5811.16	5677.40	133.76	0.0	organic	2015	xxxx
12	29-11-2015	1.21	51039.60	941.48	43838.39	75.78	6183.95	5986.26	197.69	0.0	organic	2015	yyyy
13	20-12-2015	1.35	54876.98	674.28	44638.81	58.33	9505.56	9408.07	97.49	0.0	conventional	2015	Albany
14	13-12-2015	0.93	118220.22	794.70	109149.67	130.50	8145.35	8042.21	103.14	0.0	conventional	2015	Albany
15	22-03-2015	1.12	46346.85	2141.83	34313.56	141.80	9749.66	9252.60	497.06	0.0	conventional	2015	Albany
16	15-03-2015	1.11	43045.79	2128.26	30447.17	99.67	10370.69	9989.59	381.10	0.0	conventional	2015	Albany
17	08-03-2015	1.07	40507.36	795.68	30370.64	159.05	9181.99	8827.55	354.44	0.0	conventional	2015	Albany
18	01-03-2015	0.99	55595.74	629.46	45633.34	181.49	9151.45	8986.06	165.39	0.0	conventional	2015	Albany
19	22-02-2015	1.07	45675.05	1088.38	35056.13	151.00	9379.54	9000.16	379.38	0.0	conventional	2015	Albany
20	15-02-2015	1.06	41567.62	986.66	30045.51	222.42	10313.03	9979.87	333.16	0.0	conventional	2015	Albany
21	08-02-2015	0.99	51253.97	1357.37	39111.81	163.25	10621.54	10113.10	508.44	0.0	conventional	2015	Albany
22	01-02-2015	0.99	70873.60	1353.90	60017.20	179.32	9323.18	9170.82	152.36	0.0	conventional	2015	Albany
23	25-01-2015	1.06	45147.50	941.38	33196.16	164.14	10845.82	10103.35	742.47	0.0	conventional	2015	Albany
24	18-01-2015	1.17	44511.28	914.14	31540.32	135.77	11921.05	11651.09	269.96	0.0	conventional	2015	Albany
25	11-01-2015	1.24	41195.08	1002.85	31640.34	127.12	8424.77	8036.04	388.73	0.0	conventional	2015	Albany
26	04-01-2015	1.22	40873.28	2819.50	28287.42	49.90	9716.46	9186.93	529.53	0.0	conventional	2015	Albany

In [ ]:

3. Binarize the attribute “Year”. Set the threshold above 2016 and print it without truncation. (Use AVOCADO dataset)

In [9]:

```
df = pd.read_csv("Avocado.csv")
df['year'] = np.where(df["year"] > 2016, 1, 0)
df
```

Out[9]:

	Date	AveragePrice	Total Volume	4046	4225	4770	Total Bags	Small Bags	Large Bags	XLarge Bags	type	year	region
0	27-12-2015	1.33	64236.62	1036.74	54454.85	48.16	8696.87	8603.62	93.25	0.0	conventional	0	Albany
1	20-12-2015	1.35	54876.98	674.28	44638.81	58.33	9505.56	9408.07	97.49	0.0	conventional	0	Albany
2	13-12-2015	0.93	118220.22	794.70	109149.67	130.50	8145.35	8042.21	103.14	0.0	conventional	0	Albany
3	06-12-2015	1.08	78992.15	1132.00	71976.41	72.58	5811.16	5677.40	133.76	0.0	conventional	0	Albany
4	29-11-2015	1.29	51039.60	941.48	43838.39	75.78	6183.95	5986.26	197.69	0.0	conventional	0	Albany
...	...	...	...	...	...	...	...	...	...	...	...	...	...
18245	28-01-2018	1.71	13888.04	1191.70	3431.50	0.00	9264.84	8940.04	324.80	0.0	organic	1	WestTexNewMexico
18246	21-01-2018	1.87	13766.76	1191.92	2452.79	727.94	9394.11	9351.80	42.31	0.0	organic	1	WestTexNewMexico
18247	14-01-2018	1.93	16205.22	1527.63	2981.04	727.01	10969.54	10919.54	50.00	0.0	organic	1	WestTexNewMexico
18248	07-01-2018	1.62	17489.58	2894.77	2356.13	224.53	12014.15	11988.14	26.01	0.0	organic	1	WestTexNewMexico
18249	18-03-2018	1.56	15896.38	2055.35	1499.55	0.00	12341.48	12114.81	226.67	0.0	organic	1	WestTexNewMexico

18250 rows × 13 columns

In [10]:
```
# Permanently changes the pandas settings
pd.set_option('display.max_rows', None)

display(df['year'])
pd.reset_option('all')
```

Out[14]:

1	338
41	338
30	338
31	338
32	338
33	338
34	338
35	338
36	338
37	338
38	338
39	338
40	338
42	338
2	338
43	338
44	338
45	338
46	338
47	338
48	338
49	338
50	338
51	338
52	338
53	338
29	338
28	338
27	338
26	338
3	338
4	338
5	338
6	338
7	338
8	338
9	338
10	338
11	338
12	338
13	338
14	338
15	338
16	338
17	338
18	338
19	338
20	338
21	338

```
22    338
23    338
24    338
25    338
54    336
Name: region, dtype: int64
```

In [ ]:

5. Transform the attribute = “Region” in the given dataset AVOCADO using One-Hot Encoding.

In [17]:
```
df = pd.read_csv("Avocado.csv")
df = pd.get_dummies(df, columns=['region'], prefix=['encoded'])
df
```

Out[17]:

	Date	AveragePrice	Total Volume	4046	4225	4770	Total Bags	Small Bags	Large Bags	XLarge Bags	...	encoded_SouthCarolina	encoded_SouthCentral	encoded_Southeas
0	27-12-2015	1.33	64236.62	1036.74	54454.85	48.16	8696.87	8603.62	93.25	0.0	...	0	0	
1	20-12-2015	1.35	54876.98	674.28	44638.81	58.33	9505.56	9408.07	97.49	0.0	...	0	0	
2	13-12-2015	0.93	118220.22	794.70	109149.67	130.50	8145.35	8042.21	103.14	0.0	...	0	0	
3	06-12-2015	1.08	78992.15	1132.00	71976.41	72.58	5811.16	5677.40	133.76	0.0	...	0	0	
4	29-11-2015	1.29	51039.60	941.48	43838.39	75.78	6183.95	5986.26	197.69	0.0	...	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
18245	28-01-2018	1.71	13888.04	1191.70	3431.50	0.00	9264.84	8940.04	324.80	0.0	...	0	0	
18246	21-01-2018	1.87	13766.76	1191.92	2452.79	727.94	9394.11	9351.80	42.31	0.0	...	0	0	
18247	14-01-2018	1.93	16205.22	1527.63	2981.04	727.01	10969.54	10919.54	50.00	0.0	...	0	0	
18248	07-01-2018	1.62	17489.58	2894.77	2356.13	224.53	12014.15	11988.14	26.01	0.0	...	0	0	
18249	18-03-2018	1.56	15896.38	2055.35	1499.55	0.00	12341.48	12114.81	226.67	0.0	...	0	0	

18250 rows × 66 columns

In [ ]:

6. Ignore the tuples that hold missing values and print the subset of data from AVOCADO dataset excluding “NaN” values.

In [8]:
```
df = pd.read_csv("Avocado.csv")
df.isnull().sum()
```

Out[8]:
```
Date            0
AveragePrice    28
Total Volume     0
4046             0
4225             0
4770             0
Total Bags       0
Small Bags       0
Large Bags       0
XLarge Bags      0
type             0
year             0
region           0
dtype: int64
```

In [9]:
```
drop_columns = []
for i in df['AveragePrice'].index:
    for x in str(df['AveragePrice'].loc[i]):
        if( not (x.isdigit() or x=='.') ):
            drop_columns.append(i)
            break


df.drop(drop_columns, inplace=True)
df.head(20)
```

Out[9]:

	Date	AveragePrice	Total Volume	4046	4225	4770	Total Bags	Small Bags	Large Bags	XLarge Bags	type	year	region
0	27-12-2015	1.33	64236.62	1036.74	54454.85	48.16	8696.87	8603.62	93.25	0.00	conventional	2015	Albany
1	20-12-2015	1.35	54876.98	674.28	44638.81	58.33	9505.56	9408.07	97.49	0.00	conventional	2015	Albany
2	13-12-2015	0.93	118220.22	794.70	109149.67	130.50	8145.35	8042.21	103.14	0.00	conventional	2015	Albany
3	06-12-2015	1.08	78992.15	1132.00	71976.41	72.58	5811.16	5677.40	133.76	0.00	conventional	2015	Albany
4	29-11-2015	1.29	51039.60	941.48	43838.39	75.78	6183.95	5986.26	197.69	0.00	conventional	2015	Albany
12	04-10-2015	1.31	61007.10	2268.32	49880.67	101.36	8756.75	8379.98	376.77	0.00	conventional	2015	Albany
13	27-09-2015	0.99	106803.39	1204.88	99409.21	154.84	6034.46	5888.87	145.59	0.00	conventional	2015	Albany
14	20-09-2015	1.33	69759.01	1028.03	59313.12	150.50	9267.36	8489.10	778.26	0.00	conventional	2015	Albany
15	13-09-2015	1.28	76111.27	985.73	65696.86	142.00	9286.68	8665.19	621.49	0.00	conventional	2015	Albany
16	06-09-2015	1.11	99172.96	879.45	90062.62	240.79	7990.10	7762.87	227.23	0.00	conventional	2015	Albany
17	30-08-2015	1.07	105693.84	689.01	94362.67	335.43	10306.73	10218.93	87.80	0.00	conventional	2015	Albany
18	23-08-2015	1.34	79992.09	733.16	67933.79	444.78	10880.36	10745.79	134.57	0.00	conventional	2015	Albany
19	16-08-2015	1.33	80043.78	539.65	68666.01	394.90	10443.22	10297.68	145.54	0.00	conventional	2015	Albany
20	09-08-2015	1.12	111140.93	584.63	100961.46	368.95	9225.89	9116.34	109.55	0.00	conventional	2015	Albany
21	02-08-2015	1.45	75133.10	509.94	62035.06	741.08	11847.02	11768.52	78.50	0.00	conventional	2015	Albany
22	26-07-2015	1.11	106757.10	648.75	91949.05	966.61	13192.69	13061.53	131.16	0.00	conventional	2015	Albany
23	19-07-2015	1.26	96617.00	1042.10	82049.40	2238.02	11287.48	11103.49	183.99	0.00	conventional	2015	Albany
24	12-07-2015	1.05	124055.31	672.25	94693.52	4257.64	24431.90	24290.08	108.49	33.33	conventional	2015	Albany
25	05-07-2015	1.35	109252.12	869.45	72600.55	5883.16	29898.96	29663.19	235.77	0.00	conventional	2015	Albany
26	28-06-2015	1.37	89534.81	664.23	57545.79	4662.71	26662.08	26311.76	350.32	0.00	conventional	2015	Albany

In [10]:
```
df.shape
```

Out[10]:
```
(18202, 13)
```

In [ ]:
```

```

7. Drop the attribute that has high nullity as it facilitates efficient prediction. (Use AVOCADO dataset)

In [28]:
```
# attribute which has high null or invald values..
df = pd.read_csv("Avocado.csv")
df.isnull().sum()
```

Out[28]:
```
Date            0
AveragePrice    28
Total Volume     0
4046             0
4225             0
4770             0
Total Bags       0
Small Bags       0
Large Bags       0
XLarge Bags      0
type             0
year             0
region           0
dtype: int64
```

In [32]:
```
# we can see only AveragePrice has null value, so drop it
df = df.drop(['AveragePrice'], axis=1)
df
```

Out[32]:

	Date	Total Volume	4046	4225	4770	Total Bags	Small Bags	Large Bags	XLarge Bags	type	year	region
0	27-12-2015	64236.62	1036.74	54454.85	48.16	8696.87	8603.62	93.25	0.0	conventional	2015	Albany
1	20-12-2015	54876.98	674.28	44638.81	58.33	9505.56	9408.07	97.49	0.0	conventional	2015	Albany
2	13-12-2015	118220.22	794.70	109149.67	130.50	8145.35	8042.21	103.14	0.0	conventional	2015	Albany
3	06-12-2015	78992.15	1132.00	71976.41	72.58	5811.16	5677.40	133.76	0.0	conventional	2015	Albany
4	29-11-2015	51039.60	941.48	43838.39	75.78	6183.95	5986.26	197.69	0.0	conventional	2015	Albany
...	...	...	...	...	...	...	...	...	...	...	...	...
18245	28-01-2018	13888.04	1191.70	3431.50	0.00	9264.84	8940.04	324.80	0.0	organic	2018	WestTexNewMexico
18246	21-01-2018	13766.76	1191.92	2452.79	727.94	9394.11	9351.80	42.31	0.0	organic	2018	WestTexNewMexico
18247	14-01-2018	16205.22	1527.63	2981.04	727.01	10969.54	10919.54	50.00	0.0	organic	2018	WestTexNewMexico
18248	07-01-2018	17489.58	2894.77	2356.13	224.53	12014.15	11988.14	26.01	0.0	organic	2018	WestTexNewMexico
18249	18-03-2018	15896.38	2055.35	1499.55	0.00	12341.48	12114.81	226.67	0.0	organic	2018	WestTexNewMexico

18250 rows × 12 columns

In [ ]:

8. Study the entire dataset and report the complete statistical summary about the data (Use AVOCADO dataset)

In [3]:
```
df = pd.read_csv("Avocado.csv")

# Dimension of the dataset
df.shape
```

Out[3]:
```
(18250, 13)
```

In [4]:
```
df['AveragePrice'].value_counts()
```

Out[4]:
```
1.15    201
1.18    199
1.08    193
1.26    192
1.13    191
       ...
nil       1
0.48      1
3.05      1
3.12      1
3.17      1
Name: AveragePrice, Length: 262, dtype: int64
```

In [ ]:
```

```

In [34]:
```
# Most frequently occurring value under every attribute.
for each in df:
    print(f"{each} - {df[each].value_counts().index[0]} - {df[each].value_counts().iloc[0]}")
```

```
Date - 18-03-2018 - 109
AveragePrice - 1.15 - 201
Total Volume - 4103.97 - 2
4046 - 0.0 - 242
4225 - 0.0 - 61
4770 - 0.0 - 5498
Total Bags - 0.0 - 15
Small Bags - 0.0 - 159
Large Bags - 0.0 - 2370
XLarge Bags - 0.0 - 12049
type - conventional - 9126
year - 2017 - 5722
region - Albany - 338
```

In [35]:
```
# Datatype of every attribute
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 18250 entries, 0 to 18249
Data columns (total 13 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   Date          18250 non-null  object
 1   AveragePrice  18222 non-null  object
 2   Total Volume  18250 non-null  float64
 3   4046          18250 non-null  float64
 4   4225          18250 non-null  float64
 5   4770          18250 non-null  float64
 6   Total Bags    18250 non-null  float64
 7   Small Bags    18250 non-null  float64
 8   Large Bags    18250 non-null  float64
 9   XLarge Bags   18250 non-null  float64
 10  type          18250 non-null  object
 11  year          18250 non-null  int64
 12  region        18250 non-null  object
dtypes: float64(8), int64(1), object(4)
memory usage: 1.8+ MB
```

In [36]:
```
df.describe()
```

Out[36]:

	Total Volume	4046	4225	4770	Total Bags	Small Bags	Large Bags	XLarge Bags	year
count	1.825000e+04	1.825000e+04	1.825000e+04	1.825000e+04	1.825000e+04	1.825000e+04	1.825000e+04	18250.000000	18250.000000
mean	8.505983e+05	2.929925e+05	2.951385e+05	2.283848e+04	2.396267e+05	1.821854e+05	5.433512e+04	3106.256292	2016.148000
std	3.453456e+06	1.264956e+06	1.204089e+06	1.074613e+05	9.862168e+05	7.461591e+05	2.439596e+05	17692.424825	0.940013
min	8.456000e+01	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000	2015.000000
25%	1.083963e+04	8.542100e+02	3.008097e+03	0.000000e+00	5.089083e+03	2.850323e+03	1.275800e+02	0.000000	2015.000000
50%	1.073655e+05	8.643200e+03	2.905888e+04	1.849750e+02	3.974118e+04	2.635161e+04	2.647270e+03	0.000000	2016.000000
75%	4.329527e+05	1.110087e+05	1.501663e+05	6.242055e+03	1.107811e+05	8.333621e+04	2.201828e+04	132.432500	2017.000000
max	6.250565e+07	2.274362e+07	2.047057e+07	2.546439e+06	1.937313e+07	1.338459e+07	5.719097e+06	551693.650000	2018.000000

In [37]:
```
# Find whether the class distribution of dataset is imbalanced. (Note: Fix the class label as “Type” in the given dataset)

df['type'].value_counts()

# its almost balanced..
```

Out[37]:
```
conventional    9126
organic         9124
Name: type, dtype: int64
```

In [38]:
```
# Correlation matrix
df.corr()
```

Out[38]:

	Total Volume	4046	4225	4770	Total Bags	Small Bags	Large Bags	XLarge Bags	year
Total Volume	1.000000	0.977863	0.974181	0.872203	0.963047	0.967238	0.880640	0.747158	0.017165
4046	0.977863	1.000000	0.926110	0.833390	0.920057	0.925280	0.838645	0.699378	0.003328
4225	0.974181	0.926110	1.000000	0.887855	0.905788	0.916031	0.810016	0.688809	-0.009584
4770	0.872203	0.833390	0.887855	1.000000	0.792315	0.802733	0.698472	0.679862	-0.036550
Total Bags	0.963047	0.920057	0.905788	0.792315	1.000000	0.994335	0.943009	0.804233	0.071520
Small Bags	0.967238	0.925280	0.916031	0.802733	0.994335	1.000000	0.902589	0.806845	0.063883
Large Bags	0.880640	0.838645	0.810016	0.698472	0.943009	0.902589	1.000000	0.710859	0.087858
XLarge Bags	0.747158	0.699378	0.688809	0.679862	0.804233	0.806845	0.710859	1.000000	0.081005
year	0.017165	0.003328	-0.009584	-0.036550	0.071520	0.063883	0.087858	0.081005	1.000000

In [39]:
```
# Skewness of every attribute.
df.skew()
```

Out[39]:
```
C:\Users\Sumit\AppData\Local\Temp\ipykernel_10036\2513495534.py:2: FutureWarning: Dropping of nuisance columns in DataFrame reductions (with 'numeric
_only=None') is deprecated; in a future version this will raise TypeError.  Select only valid columns before calling the reduction.
  df.skew()
Total Volume     9.007930
4046             8.648456
4225             8.942706
4770            10.159671
Total Bags       9.756334
Small Bags       9.540917
Large Bags       9.796719
XLarge Bags     13.140106
year             0.215371
dtype: float64
```

In [ ]:
```

```

Question - 9

In [3]:
```
random_list = np.random.randint(100, size=(20))
random_list
```

Out[3]:
```
array([ 4, 16, 45, 50,  7, 12, 67, 79,  2, 72, 68, 55, 20, 19, 67, 31, 31,
       48, 34, 21])
```

In [4]:
```
random_df = pd.Series(random_list)
probability = random_df.value_counts(normalize=True)
probability
```

Out[4]:
```
31    0.10
67    0.10
4     0.05
68    0.05
34    0.05
48    0.05
19    0.05
20    0.05
55    0.05
72    0.05
16    0.05
2     0.05
79    0.05
12    0.05
7     0.05
50    0.05
45    0.05
21    0.05
dtype: float64
```

In [5]:
```
print("GINI Index = ", (1-(np.sum(np.square(probability)))))
entropy = (-1*np.sum(np.log2(probability)*probability))

print("Entropy = ", entropy)
print("Information Gain : ", (1-entropy))
```

```
GINI Index =  0.94
Entropy =  4.1219280948873624
Information Gain :  -3.1219280948873624
```

In [ ]:
```

```

Question - 10 Test drive the implementation support in your platform of choice for data preprocessing phases such as cleaning, selection, transformation, integration in addition to the earlier exercises.

In [26]:
```
df = pd.read_csv("Avocado.csv")
df
```

Out[26]:

	Date	AveragePrice	Total Volume	4046	4225	4770	Total Bags	Small Bags	Large Bags	XLarge Bags	type	year	region
0	27-12-2015	1.33	64236.62	1036.74	54454.85	48.16	8696.87	8603.62	93.25	0.0	conventional	2015	Albany
1	20-12-2015	1.35	54876.98	674.28	44638.81	58.33	9505.56	9408.07	97.49	0.0	conventional	2015	Albany
2	13-12-2015	0.93	118220.22	794.70	109149.67	130.50	8145.35	8042.21	103.14	0.0	conventional	2015	Albany
3	06-12-2015	1.08	78992.15	1132.00	71976.41	72.58	5811.16	5677.40	133.76	0.0	conventional	2015	Albany
4	29-11-2015	1.29	51039.60	941.48	43838.39	75.78	6183.95	5986.26	197.69	0.0	conventional	2015	Albany
...	...	...	...	...	...	...	...	...	...	...	...	...	...
18245	28-01-2018	1.71	13888.04	1191.70	3431.50	0.00	9264.84	8940.04	324.80	0.0	organic	2018	WestTexNewMexico
18246	21-01-2018	1.87	13766.76	1191.92	2452.79	727.94	9394.11	9351.80	42.31	0.0	organic	2018	WestTexNewMexico
18247	14-01-2018	1.93	16205.22	1527.63	2981.04	727.01	10969.54	10919.54	50.00	0.0	organic	2018	WestTexNewMexico
18248	07-01-2018	1.62	17489.58	2894.77	2356.13	224.53	12014.15	11988.14	26.01	0.0	organic	2018	WestTexNewMexico
18249	18-03-2018	1.56	15896.38	2055.35	1499.55	0.00	12341.48	12114.81	226.67	0.0	organic	2018	WestTexNewMexico

18250 rows × 13 columns

In [ ]:

In [27]:
```
# finding null values..
df.isnull().sum()
```

Out[27]:
```
Date             0
AveragePrice    28
Total Volume     0
4046             0
4225             0
4770             0
Total Bags       0
Small Bags       0
Large Bags       0
XLarge Bags      0
type             0
year             0
region           0
dtype: int64
```

In [28]:

```
# replacing all the null values of AveragePrice with its region mean...

col = []
count = 0
for i in range(len(df['AveragePrice'])):
    flag = 1
    for x in str(df['AveragePrice'].iloc[i]):
        if( not (x.isdigit() or x=='.') ):
            col.append(float(0))
            flag = 0
            count += 1
            break

    if(flag):
        col.append( float(df['AveragePrice'].iloc[i]) )

df['AveragePrice'] = col

df_group_region = df.groupby(['region']).mean()

df1 = df.loc[df.AveragePrice==0]['region']
for i in df1.index:
#     print(df1.loc[index])
    df.loc[df.index == i, 'AveragePrice'] = round(df_group_region.loc[df_group_region.index == df1.loc[i]]['AveragePrice'].values[0], 2)


df.head(20)
```

Out[28]:

	Date	AveragePrice	Total Volume	4046	4225	4770	Total Bags	Small Bags	Large Bags	XLarge Bags	type	year	region
0	27-12-2015	1.33	64236.62	1036.74	54454.85	48.16	8696.87	8603.62	93.25	0.0	conventional	2015	Albany
1	20-12-2015	1.35	54876.98	674.28	44638.81	58.33	9505.56	9408.07	97.49	0.0	conventional	2015	Albany
2	13-12-2015	0.93	118220.22	794.70	109149.67	130.50	8145.35	8042.21	103.14	0.0	conventional	2015	Albany
3	06-12-2015	1.08	78992.15	1132.00	71976.41	72.58	5811.16	5677.40	133.76	0.0	conventional	2015	Albany
4	29-11-2015	1.29	51039.60	941.48	43838.39	75.78	6183.95	5986.26	197.69	0.0	conventional	2015	Albany
5	22-11-2015	1.54	55979.78	1184.27	48067.99	43.61	6683.91	6556.47	127.44	0.0	conventional	2015	Albany
6	15-11-2015	1.54	83453.76	1368.92	73672.72	93.26	8318.86	8196.81	122.05	0.0	conventional	2015	Albany
7	08-11-2015	1.54	109428.33	703.75	101815.36	80.00	6829.22	6266.85	562.37	0.0	conventional	2015	Albany
8	01-11-2015	1.54	99811.42	1022.15	87315.57	85.34	11388.36	11104.53	283.83	0.0	conventional	2015	Albany
9	25-10-2015	1.54	74338.76	842.40	64757.44	113.00	8625.92	8061.47	564.45	0.0	conventional	2015	Albany
10	18-10-2015	1.54	84843.44	924.86	75595.85	117.07	8205.66	7877.86	327.80	0.0	conventional	2015	Albany
11	11-10-2015	1.54	64489.17	1582.03	52677.92	105.32	10123.90	9866.27	257.63	0.0	conventional	2015	Albany
12	04-10-2015	1.31	61007.10	2268.32	49880.67	101.36	8756.75	8379.98	376.77	0.0	conventional	2015	Albany
13	27-09-2015	0.99	106803.39	1204.88	99409.21	154.84	6034.46	5888.87	145.59	0.0	conventional	2015	Albany
14	20-09-2015	1.33	69759.01	1028.03	59313.12	150.50	9267.36	8489.10	778.26	0.0	conventional	2015	Albany
15	13-09-2015	1.28	76111.27	985.73	65696.86	142.00	9286.68	8665.19	621.49	0.0	conventional	2015	Albany
16	06-09-2015	1.11	99172.96	879.45	90062.62	240.79	7990.10	7762.87	227.23	0.0	conventional	2015	Albany
17	30-08-2015	1.07	105693.84	689.01	94362.67	335.43	10306.73	10218.93	87.80	0.0	conventional	2015	Albany
18	23-08-2015	1.34	79992.09	733.16	67933.79	444.78	10880.36	10745.79	134.57	0.0	conventional	2015	Albany
19	16-08-2015	1.33	80043.78	539.65	68666.01	394.90	10443.22	10297.68	145.54	0.0	conventional	2015	Albany

In [29]:

```
# in this field we can see, most of the entries of XLarge Bags are 0.0.. so its better to drop them cause its
# easy to assume none use XLarge Bags..

df['XLarge Bags'].value_counts()
```

Out[29]:
```
0.00       12049
3.33          29
6.67          16
1.11          15
5.00          12
            ...
3018.05        1
2739.44        1
9301.67        1
8640.00        1
24.18          1
Name: XLarge Bags, Length: 5588, dtype: int64
```

In [30]:
```
df = df.drop(['XLarge Bags'], axis=1)
df
```

Out[30]:

	Date	AveragePrice	Total Volume	4046	4225	4770	Total Bags	Small Bags	Large Bags	type	year	region
0	27-12-2015	1.33	64236.62	1036.74	54454.85	48.16	8696.87	8603.62	93.25	conventional	2015	Albany
1	20-12-2015	1.35	54876.98	674.28	44638.81	58.33	9505.56	9408.07	97.49	conventional	2015	Albany
2	13-12-2015	0.93	118220.22	794.70	109149.67	130.50	8145.35	8042.21	103.14	conventional	2015	Albany
3	06-12-2015	1.08	78992.15	1132.00	71976.41	72.58	5811.16	5677.40	133.76	conventional	2015	Albany
4	29-11-2015	1.29	51039.60	941.48	43838.39	75.78	6183.95	5986.26	197.69	conventional	2015	Albany
...	...	...	...	...	...	...	...	...	...	...	...	...
18245	28-01-2018	1.71	13888.04	1191.70	3431.50	0.00	9264.84	8940.04	324.80	organic	2018	WestTexNewMexico
18246	21-01-2018	1.87	13766.76	1191.92	2452.79	727.94	9394.11	9351.80	42.31	organic	2018	WestTexNewMexico
18247	14-01-2018	1.93	16205.22	1527.63	2981.04	727.01	10969.54	10919.54	50.00	organic	2018	WestTexNewMexico
18248	07-01-2018	1.62	17489.58	2894.77	2356.13	224.53	12014.15	11988.14	26.01	organic	2018	WestTexNewMexico
18249	18-03-2018	1.56	15896.38	2055.35	1499.55	0.00	12341.48	12114.81	226.67	organic	2018	WestTexNewMexico

18250 rows × 12 columns

In [ ]:
```
```

In [31]:
```
# normalizing of Total Volumn column using min-max algo
columns = ('Total Volume', 'AveragePrice', '4046', '4225', '4770', 'Total Bags', 'Small Bags', 'Large Bags')

for each in columns:
    df[each] = (df[each] - df[each].min()) / (df[each].max() - df[each].min())

df
```

Out[31]:

	Date	AveragePrice	Total Volume	4046	4225	4770	Total Bags	Small Bags	Large Bags	type	year	region
0	27-12-2015	0.316726	0.001026	0.000046	0.002660	0.000019	0.000449	0.000643	0.000016	conventional	2015	Albany
1	20-12-2015	0.323843	0.000877	0.000030	0.002181	0.000023	0.000491	0.000703	0.000017	conventional	2015	Albany
2	13-12-2015	0.174377	0.001890	0.000035	0.005332	0.000051	0.000420	0.000601	0.000018	conventional	2015	Albany
3	06-12-2015	0.227758	0.001262	0.000050	0.003516	0.000029	0.000300	0.000424	0.000023	conventional	2015	Albany
4	29-11-2015	0.302491	0.000815	0.000041	0.002142	0.000030	0.000319	0.000447	0.000035	conventional	2015	Albany
...	...	...	...	...	...	...	...	...	...	...	...	...
18245	28-01-2018	0.451957	0.000221	0.000052	0.000168	0.000000	0.000478	0.000668	0.000057	organic	2018	WestTexNewMexico
18246	21-01-2018	0.508897	0.000219	0.000052	0.000120	0.000286	0.000485	0.000699	0.000007	organic	2018	WestTexNewMexico
18247	14-01-2018	0.530249	0.000258	0.000067	0.000146	0.000286	0.000566	0.000816	0.000009	organic	2018	WestTexNewMexico
18248	07-01-2018	0.419929	0.000278	0.000127	0.000115	0.000088	0.000620	0.000896	0.000005	organic	2018	WestTexNewMexico
18249	18-03-2018	0.398577	0.000253	0.000090	0.000073	0.000000	0.000637	0.000905	0.000040	organic	2018	WestTexNewMexico

18250 rows × 12 columns

In [ ]:
```
```

In [42]:
```
df[['AveragePrice', 'Total Volume', '4046', '4225', 'Total Bags', 'Small Bags', 'Large Bags', 'region']].groupby(['region']).agg({"AveragePrice": "me
```

Out[42]:

region	AveragePrice	Total Volume	4046	4225	Total Bags	Small Bags	Large Bags
Albany	0.402177	0.256605	0.027108	0.621183	0.138349	0.167876	0.068172
Atlanta	0.319558	1.417098	2.171489	0.515465	1.474233	1.303196	1.895347
BaltimoreWashington	0.391954	2.154774	0.529898	4.061548	1.822428	2.549023	0.171626
Boise	0.324380	0.230133	0.297516	0.057158	0.278706	0.349501	0.124325
Boston	0.394597	1.555788	0.074226	3.537093	1.109551	1.487564	0.262308
BuffaloRochester	0.383868	0.366910	0.026404	0.522820	0.599322	0.767183	0.220388
California	0.339973	16.461784	17.541934	17.170570	12.733210	17.116530	2.152776
Charlotte	0.414959	0.568381	0.332569	0.604784	0.603823	0.795232	0.172140
Chicago	0.397429	2.138590	0.476797	4.202493	0.870556	1.105350	0.295829
CincinnatiDayton	0.273737	0.711831	0.080425	1.008174	1.078706	0.423016	2.617930
Columbus	0.289246	0.479394	0.553392	0.279214	0.517025	0.602992	0.301600
DallasFtWorth	0.229748	3.333955	4.860989	2.304308	2.398670	3.049906	0.911983
Denver	0.277075	2.221786	1.162034	2.482125	3.043235	1.401882	7.018752
Detroit	0.297543	1.014211	0.825865	0.574590	1.252253	1.489704	0.467201
GrandRapids	0.379004	0.482888	0.021688	0.842259	0.473289	0.562847	0.139323
GreatLakes	0.319769	9.432984	4.120585	13.055703	9.217695	9.211154	8.549539
HarrisburgScranton	0.381952	0.668425	0.330445	1.010707	0.698351	0.954297	0.105220
HartfordSpringfield	0.490619	0.810199	0.058023	1.848879	0.584986	0.805296	0.083378
Houston	0.216345	3.249939	4.386853	2.331368	2.591995	2.430046	3.036103
Indianapolis	0.311030	0.483714	0.106646	0.706567	0.618424	0.493931	0.889346
Jacksonville	0.381120	0.460142	0.680266	0.172283	0.501629	0.324238	0.921999
LasVegas	0.334846	0.869496	0.870008	0.750562	0.931857	0.670743	1.584046
LosAngeles	0.276159	8.125164	9.843613	4.989306	8.685457	11.634734	1.479805
Louisville	0.301312	0.257072	0.031198	0.410964	0.342499	0.210260	0.649250
MiamiFtLauderdale	0.351776	1.562175	2.600294	0.728425	1.213869	0.838276	2.096627
Midsouth	0.343332	8.132409	4.855456	10.863061	8.078901	10.038744	3.631844

region	AveragePrice	Total Volume	4046	4225	Total Bags	Small Bags	Large Bags
Nashville	0.274769	0.569285	0.799736	0.186146	0.669338	0.750325	0.491842
NewOrleansMobile	0.307755	0.730600	1.178561	0.215588	0.736532	0.886061	0.331643
NewYork	0.458211	3.850946	0.335889	7.979935	3.509714	4.363353	1.614415
Northeast	0.413496	11.411022	1.538507	23.178866	10.168122	12.971859	3.856563
NorthernNewEngland	0.369180	1.143967	0.118857	2.618845	0.717748	0.842534	0.440175
Orlando	0.379435	0.938030	1.468144	0.408371	0.870362	0.667027	1.344708
Philadelphia	0.424246	1.148861	0.202923	2.088287	1.232389	1.600126	0.411263
PhoenixTucson	0.279159	3.129557	5.075170	1.912892	2.008360	1.664050	2.881696
Pittsburgh	0.328939	0.300417	0.174662	0.356682	0.373995	0.404060	0.311185
Plains	0.354629	4.978116	6.309962	4.266385	4.018307	5.137299	1.394214
Portland	0.312357	1.768221	1.020415	1.709071	2.555310	3.101225	1.384436
RaleighGreensboro	0.396839	0.770718	0.519742	0.814737	0.809197	1.123717	0.094621
RichmondNorfolk	0.302965	0.675176	0.548587	0.785340	0.645648	0.870890	0.127283
Roanoke	0.287519	0.400179	0.327694	0.389415	0.495066	0.631797	0.184498
Sacramento	0.420487	1.202056	1.031120	2.011517	0.463646	0.649765	0.013414
SanDiego	0.340984	1.436086	1.498560	1.429294	1.204899	1.514861	0.510889
SanFrancisco	0.485481	2.172632	1.500944	4.062078	0.768638	1.083511	0.030938
Seattle	0.356788	1.746814	0.943854	1.760614	2.616582	3.171345	1.429587
SouthCarolina	0.342806	0.971517	1.249158	0.617074	0.937028	1.002149	0.789155
SouthCentral	0.235318	16.178577	23.524904	10.769118	12.047221	13.808083	8.032185
Southeast	0.340932	9.842481	14.935492	4.438588	9.370071	7.987043	12.608216
Spokane	0.357862	0.248565	0.180449	0.240668	0.322838	0.401082	0.153675
StLouis	0.352534	0.512662	0.594050	0.185606	0.760643	0.891462	0.451011
Syracuse	0.384457	0.174610	0.014588	0.312172	0.216404	0.272377	0.088303
Tampa	0.344785	1.055521	1.571771	0.523733	1.006378	0.792559	1.513078
TotalUS	0.312820	93.827036	90.352221	98.434548	84.581088	92.909947	65.322256
West	0.296163	17.386462	17.525381	14.692257	18.947560	16.639235	25.096920

region	AveragePrice	Total Volume	4046	4225	Total Bags	Small Bags	Large Bags
WestTexNewMexico	0.292736	2.311943	3.696837	1.083505	1.822568	1.490063	2.631792

In [ ]:

In [ ]:

In [ ]:

In [ ]: