Problem Set 3

CED18I039 - Paleti Krishnasai

QUESTION 1

Suppose that the data for analysis includes the attribute age. The age values for the data tuples are (in increasing order) 13, 15, 16, 16, 19, 20, 20, 21, 22, 22, 25, 25, 25, 25, 30, 33, 33, 35, 35, 35, 35, 36, 40, 45, 46, 52, 70.

In [1]:
```
import pandas as pd
import numpy as np
import random
import matplotlib.pyplot as plt
```

In [2]:
```
age = np.array([13, 15, 16, 16, 19, 20, 20, 21, 22, 22, 25, 25, 25, 25, 30, 33, 33, 35, 35, 35, 35, 36, 40, 45, 46, 52, 70])
```

In [3]:
```
# Min Max
age_min = min(age)
age_max = max(age)
min_max_age = [(i-age_min)/(age_max-age_min) for i in age]
```

In [4]:
```
#Z score
mean_age = np.mean(age)
std_age = np.std(age)

z_score_age = [(i-mean_age)/std_age for i in age]
```

In [5]:
```
#as we can see that the dataset has at max 2 digit numbers, we can easily realize that j = 2
decimal_scale_age = [i/100 for i in age]

Output = pd.DataFrame({"Age":age , "Min-Max":min_max_age , "Z-Score":z_score_age, "Decimal Scale":decimal_scale_age})
```

In [6]:
```
display(Output)
```

	Age	Min-Max	Z-Score	Decimal Scale
0	13	0.000000	-1.335646	0.13
1	15	0.035088	-1.178168	0.15
2	16	0.052632	-1.099429	0.16
3	16	0.052632	-1.099429	0.16
4	19	0.105263	-0.863212	0.19
5	20	0.122807	-0.784473	0.20
6	20	0.122807	-0.784473	0.20
7	21	0.140351	-0.705734	0.21
8	22	0.157895	-0.626995	0.22

	Age	Min-Max	Z-Score	Decimal Scale
9	22	0.157895	-0.626995	0.22
10	25	0.210526	-0.390779	0.25
11	25	0.210526	-0.390779	0.25
12	25	0.210526	-0.390779	0.25
13	25	0.210526	-0.390779	0.25
14	30	0.298246	0.002916	0.30
15	33	0.350877	0.239133	0.33
16	33	0.350877	0.239133	0.33
17	35	0.385965	0.396611	0.35
18	35	0.385965	0.396611	0.35
19	35	0.385965	0.396611	0.35
20	35	0.385965	0.396611	0.35
21	36	0.403509	0.475350	0.36
22	40	0.473684	0.790306	0.40
23	45	0.561404	1.184001	0.45
24	46	0.578947	1.262740	0.46
25	52	0.684211	1.735173	0.52
26	70	1.000000	3.152474	0.70

QUESTION 2

Dataset Description

It is a well-known fact that Millenials LOVE Avocado Toast. It's also a well known fact that all Millenials live in their parents basements.

Clearly, they aren't buying home because they are buying too much Avocado Toast!

But maybe there's hope... if a Millenial could find a city with cheap avocados, they could live out the Millenial American Dream. Help them to filter out the clutter using some pre-processing techniques.

Some relevant columns in the dataset:

- Date - The date of the observation
- Average Price - the average price of a single avocado
- type - conventional or organic
- year - the year
- Region - the city or region of the observation
- Total Volume - Total number of avocados sold
- 4046 - Total number of avocados with PLU* 4046 sold
- 4225 - Total number of avocados with PLU* 4225 sold
- 4770 - Total number of avocados with PLU* 4770 sold

(Product Lookup codes (PLU’s))

a. Sort the attribute “Total Volume” in the given dataset and distribute the data into equal sized/frequency bins. Let the number of bins be 250. Smooth the sorted data by

- (i)bin-means
- (ii) bin-medians
- (iii) bin-boundaries

b. The dataset represents weekly retail scan data for National retail volume (units) and price. However, the company is interested in knowing the monthly (total per month) and annual sales (total per year), rather than the total per week. So, reduce the data accordingly.

c. Summarize the number of missing values for each attribute

d. Populate data for the missing values of the attribute= “Average Price” by averaging all the values of the “Avg Price” attribute that fall under the same “REGION” attribute value.

e. Discretize the attribute= “Date” using concept hierarchy into {Old, New, Recent}

(Consider 2015,2016 : Old, 2017: New, 2018: Recent)

In [7]:

```
#Read the dataset

Avacado_data= pd.read_csv('Avocado Dataset.csv')
Avacado_data.head()
```

Out[7]:

	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

In [8]:

```
#Store Total Volume in a list
Total_Vol=list(Avacado_data['Total Volume'])

#Sort
Total_Vol=sorted(Total_Vol)
print(Total_Vol[:10])
```

```
[84.56, 379.82, 385.55, 419.98, 472.82, 482.26, 515.01, 530.96, 542.85, 561.1]
```

A

In [9]:

```
#Create bins of size 250

Bins=[]
NBins = 250
size=int(len(Total_Vol)/NBins)

for i in range( int(NBins) ):
    Bins.append( Total_Vol[size*i: size*i+size] )

print("Bin Size:-",len(Bins[0]))
#print("First Bin:-",Bins[0])
```

```
Bin Size:- 73
```

In [10]:

```
#Bin by mean

Mean_Bins = []

for i in Bins:
    mean = np.array(i).mean()
    Mean_Bins.append([mean]*size) #To put the mean for all values of the bin

#print("First Bin:-",Mean_Bins[0])
```

In [11]:

```
#Similarily,Bin by median

Median_Bins = []

for j in Bins:
    median = np.median(np.array(j))
    Median_Bins.append([median]*size) #To put the Median for all values of the bin

#print("First Bin:-",Median_Bins[0])
```

In [12]:

```
#Bin by Boundary

Boundary_Bins = []
for k in Bins:
    bins = []
    for val in k:
        if((val-k[0])<=(k[size-1]-val)): #Append Closest Boundary For each value
            bins.append(k[0])
        else:
            bins.append(k[size-1])
    Boundary_Bins.append(bins)

#print("First Bin:\n",Boundary_Bins[0])
```

B

In [13]:

```
#Monthly Sales

dates = Avacado_data.loc[:,'Date']
months = list(set([i[3:] for i in dates]))

AveragePrice = pd.to_numeric(Avacado_data.iloc[:,1],errors='coerce')
Avacado_data['AveragePrice'] = AveragePrice

month = []
AvgPrice = []
TotVol = []
region = []

for i in months:
    monthly = Avacado_data.loc[Avacado_data['Date'].str.contains("[0-9][0-9]-"+i,na=False)].groupby('region').sum()
    month.extend([i]*len(monthly.index))
    AvgPrice.extend(list(monthly['AveragePrice']))
    TotVol.extend(list(monthly['Total Volume']))
    region.extend(list(monthly.index.values))

MonthlyData = pd.DataFrame.from_dict({'Month':month,'region':region,'Avg Price':AvgPrice,"Total Volume":TotVol})
MonthlyData.head()
```

Out[13]:

	Month	region	Avg Price	Total Volume
0	12-2017	Albany	14.39	511555.37
1	12-2017	Atlanta	12.83	2695000.64
2	12-2017	BaltimoreWashington	14.32	3893550.53
3	12-2017	Boise	14.30	398794.24
4	12-2017	Boston	15.80	2649927.32

In [14]:

```
#Annual Sales
dates = Avacado_data.loc[:,'Date']
years = list(set([i[6:] for i in dates]))

AveragePrice = pd.to_numeric(Avacado_data.iloc[:,1],errors='coerce')
Avacado_data['AveragePrice'] = AveragePrice

year = []
AvgPrice = []
TotVol = []
region = []

for i in years:
    yearly = Avacado_data.loc[Avacado_data['Date'].str.contains("[0-9][0-9]-[0-9][0-9]-"+i,na=False)].groupby('region').sum()
    year.extend([i]*len(yearly.index))
    AvgPrice.extend(list(yearly['AveragePrice']))
    TotVol.extend(list(yearly['Total Volume']))
    region.extend(list(yearly.index.values))

YearlyData = pd.DataFrame.from_dict({'Year':year,'region':region,'Avg Price':AvgPrice,"Total Volume":TotVol})
YearlyData.head()
```

Out[14]:

	Year	region	Avg Price	Total Volume
0	2015	Albany	152.32	4029896.43
1	2015	Atlanta	143.58	23231698.12
2	2015	BaltimoreWashington	134.21	40645579.54
3	2015	Boise	137.36	3784357.34
4	2015	Boston	137.11	27454991.64

C

In [15]:

```
# missing data
Avacado_data.isnull().sum()
```

Out[15]:

```
Date             0
AveragePrice    48
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
```

D

In [16]:
```
for i in range(len(Avacado_data)):
    if(np.isnan(Avacado_data.iloc[i]['AveragePrice'])):
        Avacado_data.iloc[i,1]=Avacado_data[Avacado_data['region']==Avacado_data.iloc[i]['region']]['AveragePrice'].mean()

#To show there are no more missing values
Avacado_data.isnull().sum()
```

Out[16]:
```
Date            0
AveragePrice    0
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
```

E

In [17]:
```
# hierarchy
DiscreteDate = []
for i in range(len(Avacado_data)):
    year = Avacado_data.iloc[i,0][6:]
    if(int(year)<=2016):
        DiscreteDate.append('Old')
    elif(int(year)==2017):
        DiscreteDate.append('New')
    elif(int(year)==2018):
        DiscreteDate.append('Recent')
    else:
        DiscreteDate.append(np.nan)

Avacado_data = Avacado_data.drop(['Date'],axis=1)
Avacado_data.insert(0,"Date",DiscreteDate)

Avacado_data
```

Out[17]:

	Date	AveragePrice	Total Volume	4046	4225	4770	Total Bags	Small Bags	Large Bags	XLarge Bags	type	year	region
0	Old	1.33	64236.62	1036.74	54454.85	48.16	8696.87	8603.62	93.25	0.0	conventional	2015	Albany
1	Old	1.35	54876.98	674.28	44638.81	58.33	9505.56	9408.07	97.49	0.0	conventional	2015	Albany
2	Old	0.93	118220.22	794.70	109149.67	130.50	8145.35	8042.21	103.14	0.0	conventional	2015	Albany
3	Old	1.08	78992.15	1132.00	71976.41	72.58	5811.16	5677.40	133.76	0.0	conventional	2015	Albany
4	Old	1.29	51039.60	941.48	43838.39	75.78	6183.95	5986.26	197.69	0.0	conventional	2015	Albany
...	...	...	...	...	...	...	...	...	...	...	...	...	...
18245	Recent	1.71	13888.04	1191.70	3431.50	0.00	9264.84	8940.04	324.80	0.0	organic	2018	WestTexNewMexico
18246	Recent	1.87	13766.76	1191.92	2452.79	727.94	9394.11	9351.80	42.31	0.0	organic	2018	WestTexNewMexico

	Date	AveragePrice	Total Volume	4046	4225	4770	Total Bags	Small Bags	Large Bags	XLarge Bags	type	year	region
18247	Recent	1.93	16205.22	1527.63	2981.04	727.01	10969.54	10919.54	50.00	0.0	organic	2018	WestTexNewMexico
18248	Recent	1.62	17489.58	2894.77	2356.13	224.53	12014.15	11988.14	26.01	0.0	organic	2018	WestTexNewMexico
18249	Recent	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 [ ]: