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

Creating DataFrames

In [2]:
```
list_of_dicts = [
     {"name": "Ginger", "breed": "Dachshund", "height_cm": 22,"weight_kg": 10, "dat
    {"name": "Scout", "breed": "Dalmatian", "height_cm": 59,"weight_kg": 25, "date_
]
new_dogs = pd.DataFrame(list_of_dicts)
new_dogs
```

Out[2]:

	name	breed	height_cm	weight_kg	date_of_birth
0	Ginger	Dachshund	22	10	2019-03-14
1	Scout	Dalmatian	59	25	2019-05-09

In [3]:
```
dict_of_lists = {
     "name": ["Ginger", "Scout"],
     "breed": ["Dachshund", "Dalmatian"],
     "height_cm": [22, 59],
     "weight_kg": [10, 25],
     "date_of_birth": ["2019-03-14","2019-05-09"]  }
new_dogs = pd.DataFrame(dict_of_lists)
new_dogs
```

Out[3]:

	name	breed	height_cm	weight_kg	date_of_birth
0	Ginger	Dachshund	22	10	2019-03-14
1	Scout	Dalmatian	59	25	2019-05-09

Reading and Writing CSVs

In [4]:
```
avacado = pd.read_csv(r"C:\Users\Sonu\OneDrive\Desktop\NIT\29th- REGRESSION PROJECT
avacado.head()
```

Out[4]:

	Unnamed: 0	Date	AveragePrice	Total Volume	4046	4225	4770	Total Bags	Small Bags
0	0	2015-12-27	1.33	64236.62	1036.74	54454.85	48.16	8696.87	8603.62
1	1	2015-12-20	1.35	54876.98	674.28	44638.81	58.33	9505.56	9408.07
2	2	2015-12-13	0.93	118220.22	794.70	109149.67	130.50	8145.35	8042.21
3	3	2015-12-06	1.08	78992.15	1132.00	71976.41	72.58	5811.16	5677.40
4	4	2015-11-29	1.28	51039.60	941.48	43838.39	75.78	6183.95	5986.26

Read CSV and assign index

In [5]:
```
# read CSV from using pandas and assigning Date as index of the dataframe
avocado = pd.read_csv(r"C:\Users\Sonu\OneDrive\Desktop\NIT\29th- REGRESSION PROJECT
# print the first few rows of the dataframe
avocado.head()
```

Out[5]:

Date	Unnamed: 0	AveragePrice	Total Volume	4046	4225	4770	Total Bags	Small Bags	I
2015-12-27	0	1.33	64236.62	1036.74	54454.85	48.16	8696.87	8603.62	
2015-12-20	1	1.35	54876.98	674.28	44638.81	58.33	9505.56	9408.07	
2015-12-13	2	0.93	118220.22	794.70	109149.67	130.50	8145.35	8042.21	1
2015-12-06	3	1.08	78992.15	1132.00	71976.41	72.58	5811.16	5677.40	1
2015-11-29	4	1.28	51039.60	941.48	43838.39	75.78	6183.95	5986.26	1

Remove index from dataframe.reset_index(drop)

In [6]:
```
avocado=avocado.reset_index(drop=True)
avacado.head()
```

Out[6]:

	Unnamed: 0	Date	AveragePrice	Total Volume	4046	4225	4770	Total Bags	Small Bags
0	0	2015-12-27	1.33	64236.62	1036.74	54454.85	48.16	8696.87	8603.62
1	1	2015-12-20	1.35	54876.98	674.28	44638.81	58.33	9505.56	9408.07
2	2	2015-12-13	0.93	118220.22	794.70	109149.67	130.50	8145.35	8042.21
3	3	2015-12-06	1.08	78992.15	1132.00	71976.41	72.58	5811.16	5677.40
4	4	2015-11-29	1.28	51039.60	941.48	43838.39	75.78	6183.95	5986.26

In [7]:
```
avacado.to_csv("test_write.csv")
```

Some useful pandas function

In [8]:
```
avocado=pd.read_csv(r"C:\Users\Sonu\OneDrive\Desktop\NIT\29th- REGRESSION PROJECT\2
avocado.head()
```

Out[8]:

	Unnamed: 0	Date	AveragePrice	Total Volume	4046	4225	4770	Total Bags	Small Bags
0	0	2015-12-27	1.33	64236.62	1036.74	54454.85	48.16	8696.87	8603.62
1	1	2015-12-20	1.35	54876.98	674.28	44638.81	58.33	9505.56	9408.07
2	2	2015-12-13	0.93	118220.22	794.70	109149.67	130.50	8145.35	8042.21
3	3	2015-12-06	1.08	78992.15	1132.00	71976.41	72.58	5811.16	5677.40
4	4	2015-11-29	1.28	51039.60	941.48	43838.39	75.78	6183.95	5986.26

In [9]:
```
avocado.tail(10)
```

Out[9]:

	Unnamed: 0	Date	AveragePrice	Total Volume	4046	4225	4770	Total Bags	Sr B
18239	2	2018-03-11	1.56	22128.42	2162.67	3194.25	8.93	16762.57	1651
18240	3	2018-03-04	1.54	17393.30	1832.24	1905.57	0.00	13655.49	1340
18241	4	2018-02-25	1.57	18421.24	1974.26	2482.65	0.00	13964.33	1369
18242	5	2018-02-18	1.56	17597.12	1892.05	1928.36	0.00	13776.71	1355
18243	6	2018-02-11	1.57	15986.17	1924.28	1368.32	0.00	12693.57	1243
18244	7	2018-02-04	1.63	17074.83	2046.96	1529.20	0.00	13498.67	1306
18245	8	2018-01-28	1.71	13888.04	1191.70	3431.50	0.00	9264.84	894
18246	9	2018-01-21	1.87	13766.76	1191.92	2452.79	727.94	9394.11	935
18247	10	2018-01-14	1.93	16205.22	1527.63	2981.04	727.01	10969.54	1091
18248	11	2018-01-07	1.62	17489.58	2894.77	2356.13	224.53	12014.15	1198

In [10]:
```
avocado.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 18249 entries, 0 to 18248
Data columns (total 14 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   Unnamed: 0    18249 non-null  int64  
 1   Date          18249 non-null  object 
 2   AveragePrice  18249 non-null  float64
 3   Total Volume  18249 non-null  float64
 4   4046          18249 non-null  float64
 5   4225          18249 non-null  float64
 6   4770          18249 non-null  float64
 7   Total Bags    18249 non-null  float64
 8   Small Bags    18249 non-null  float64
 9   Large Bags    18249 non-null  float64
 10  XLarge Bags   18249 non-null  float64
 11  type          18249 non-null  object 
 12  year          18249 non-null  int64  
 13  region        18249 non-null  object 
dtypes: float64(9), int64(2), object(3)
memory usage: 1.9+ MB
```

In [11]:
```
print(avocado.shape)
```

```
(18249, 14)
```

In [12]:
```
avocado.describe()
```

Out[12]:

	Unnamed: 0	AveragePrice	Total Volume	4046	4225	477
count	18249.000000	18249.000000	1.824900e+04	1.824900e+04	1.824900e+04	1.824900e+0
mean	24.232232	1.405978	8.506440e+05	2.930084e+05	2.951546e+05	2.283974e+0
std	15.481045	0.402677	3.453545e+06	1.264989e+06	1.204120e+06	1.074641e+0
min	0.000000	0.440000	8.456000e+01	0.000000e+00	0.000000e+00	0.000000e+0
25%	10.000000	1.100000	1.083858e+04	8.540700e+02	3.008780e+03	0.000000e+0
50%	24.000000	1.370000	1.073768e+05	8.645300e+03	2.906102e+04	1.849900e+0
75%	38.000000	1.660000	4.329623e+05	1.110202e+05	1.502069e+05	6.243420e+0
max	52.000000	3.250000	6.250565e+07	2.274362e+07	2.047057e+07	2.546439e+0

In [13]:
```
avocado.values
```

Out[13]:
```
array([[0, '2015-12-27', 1.33, ..., 'conventional', 2015, 'Albany'],
       [1, '2015-12-20', 1.35, ..., 'conventional', 2015, 'Albany'],
       [2, '2015-12-13', 0.93, ..., 'conventional', 2015, 'Albany'],
       ...,
       [9, '2018-01-21', 1.87, ..., 'organic', 2018, 'WestTexNewMexico'],
       [10, '2018-01-14', 1.93, ..., 'organic', 2018, 'WestTexNewMexico'],
       [11, '2018-01-07', 1.62, ..., 'organic', 2018, 'WestTexNewMexico']],
      dtype=object)
```

In [14]:
```
print(avocado.columns)
```

```
Index(['Unnamed: 0', 'Date', 'AveragePrice', 'Total Volume', '4046', '4225',
       '4770', 'Total Bags', 'Small Bags', 'Large Bags', 'XLarge Bags', 'type',
       'year', 'region'],
      dtype='object')
```

Appending & Concatenating Series

In [15]:
```
import pandas as pd

even = pd.Series([2,4,6,8,10])
odd = pd.Series([1,3,5,7,9])

res = pd.concat([even, odd])
print(res)
```

```
0     2
1     4
2     6
3     8
4    10
0     1
1     3
2     5
3     7
4     9
dtype: int64
```

Observe index got messed up

In [16]:
```
res.reset_index(drop=True)
```

Out[16]:
```
0     2
1     4
2     6
3     8
4    10
5     1
6     3
7     5
8     7
9     9
dtype: int64
```

Sorting

In [17]:
```
# sort values based on "AveragePrice" (ascending) and "year" (descending)
avocado.sort_values(["AveragePrice", "year"], ascending=[True, False])
```

Out[17]:

	Unnamed: 0	Date	AveragePrice	Total Volume	4046	4225	4770	T
15261	43	2017-03-05	0.44	64057.04	223.84	4748.88	0.00	5908
7412	47	2017-02-05	0.46	2200550.27	1200632.86	531226.65	18324.93	45036
15473	43	2017-03-05	0.48	50890.73	717.57	4138.84	0.00	4603
15262	44	2017-02-26	0.49	44024.03	252.79	4472.68	0.00	3929
1716	0	2015-12-27	0.49	1137707.43	738314.80	286858.37	11642.46	10089
...	...	...	...	...	...	...	...	
16720	18	2017-08-27	3.04	12656.32	419.06	4851.90	145.09	724
16055	42	2017-03-12	3.05	2068.26	1043.83	77.36	0.00	94
14124	7	2016-11-06	3.12	19043.80	5898.49	10039.34	0.00	310
17428	37	2017-04-16	3.17	3018.56	1255.55	82.31	0.00	168
14125	8	2016-10-30	3.25	16700.94	2325.93	11142.85	0.00	323

18249 rows × 14 columns

Subsetting

In [18]:
```
avocado["AveragePrice"]
```

Out[18]:
```
0        1.33
1        1.35
2        0.93
3        1.08
4        1.28
         ...
18244    1.63
18245    1.71
18246    1.87
18247    1.93
18248    1.62
Name: AveragePrice, Length: 18249, dtype: float64
```

Subsetting multiple columns

In [19]:
```
avocado[["AveragePrice","Date"]]
```

Out[19]:

	AveragePrice	Date
0	1.33	2015-12-27
1	1.35	2015-12-20
2	0.93	2015-12-13
3	1.08	2015-12-06
4	1.28	2015-11-29
...	...	...
18244	1.63	2018-02-04
18245	1.71	2018-01-28
18246	1.87	2018-01-21
18247	1.93	2018-01-14
18248	1.62	2018-01-07

18249 rows × 2 columns

Subsetting rows

In [20]:
```
avocado["AveragePrice"]<1
```

Out[20]:
```
0        False
1        False
2         True
3        False
4        False
         ...
18244    False
18245    False
18246    False
18247    False
18248    False
Name: AveragePrice, Length: 18249, dtype: bool
```

In [21]:
```
avocado[avocado["AveragePrice"]<1]
```

Out[21]:

	Unnamed: 0	Date	AveragePrice	Total Volume	4046	4225	4770	Total Bags
2	2	2015-12-13	0.93	118220.22	794.70	109149.67	130.50	8145.35
6	6	2015-11-15	0.99	83453.76	1368.92	73672.72	93.26	8318.86
7	7	2015-11-08	0.98	109428.33	703.75	101815.36	80.00	6829.22
13	13	2015-09-27	0.99	106803.39	1204.88	99409.21	154.84	6034.46
43	43	2015-03-01	0.99	55595.74	629.46	45633.34	181.49	9151.45
...	...	...	...	...	...	...	...	...
17169	43	2017-03-05	0.99	155011.12	35367.23	5175.81	5.91	114462.17
17170	44	2017-02-26	0.99	171145.00	34520.03	6936.39	0.00	129688.58
17536	39	2017-04-02	0.98	402676.23	34093.33	58330.53	207.85	310044.52
17537	40	2017-03-26	0.90	456645.91	36169.35	51398.72	139.55	368938.29
17540	43	2017-03-05	0.99	367519.17	61166.48	55123.99	126.80	251101.90

2796 rows × 14 columns

Subsetting based on text data

In [22]:
```
avocado[avocado["type"]=="organic"]
```

Out[22]:

	Unnamed: 0	Date	AveragePrice	Total Volume	4046	4225	4770	Total Bags	Sr B
9126	0	2015-12-27	1.83	989.55	8.16	88.59	0.00	892.80	89
9127	1	2015-12-20	1.89	1163.03	30.24	172.14	0.00	960.65	96
9128	2	2015-12-13	1.85	995.96	10.44	178.70	0.00	806.82	80
9129	3	2015-12-06	1.84	1158.42	90.29	104.18	0.00	963.95	94
9130	4	2015-11-29	1.94	831.69	0.00	94.73	0.00	736.96	73
...	...	...	...	...	...	...	...	...	
18244	7	2018-02-04	1.63	17074.83	2046.96	1529.20	0.00	13498.67	1306
18245	8	2018-01-28	1.71	13888.04	1191.70	3431.50	0.00	9264.84	894
18246	9	2018-01-21	1.87	13766.76	1191.92	2452.79	727.94	9394.11	935
18247	10	2018-01-14	1.93	16205.22	1527.63	2981.04	727.01	10969.54	1091
18248	11	2018-01-07	1.62	17489.58	2894.77	2356.13	224.53	12014.15	1198

9123 rows × 14 columns

Subsetting based on dates

In [23]:
```
# it will print all the rows with "Date" <= 2015-02-04
avocado[avocado["Date"]<="2015-02-04"]
```

Out[23]:

	Unnamed: 0	Date	AveragePrice	Total Volume	4046	4225	4770	Total Bags	S
47	47	2015-02-01	0.99	70873.60	1353.90	60017.20	179.32	9323.18	91
48	48	2015-01-25	1.06	45147.50	941.38	33196.16	164.14	10845.82	101
49	49	2015-01-18	1.17	44511.28	914.14	31540.32	135.77	11921.05	116
50	50	2015-01-11	1.24	41195.08	1002.85	31640.34	127.12	8424.77	80
51	51	2015-01-04	1.22	40873.28	2819.50	28287.42	49.90	9716.46	91
...	...	...	...	...	...	...	...	...	
11928	46	2015-02-01	1.77	7210.19	1634.42	3012.44	0.00	2563.33	25
11929	47	2015-01-25	1.63	7324.06	1934.46	3032.72	0.00	2356.88	23
11930	48	2015-01-18	1.71	5508.20	1793.64	2078.72	0.00	1635.84	16
11931	49	2015-01-11	1.69	6861.73	1822.28	2377.54	0.00	2661.91	26
11932	50	2015-01-04	1.64	6182.81	1561.30	2958.17	0.00	1663.34	16

540 rows × 14 columns

Subsetting based on multiple conditions

In [24]:
```
# it will print all the rows with "Date" before 2015-02-04 and "type" == "organic"
avocado[(avocado["Date"]<"2015-02-04") & (avocado["type"]=="organic")]
```

Out[24]:

	Unnamed: 0	Date	AveragePrice	Total Volume	4046	4225	4770	Total Bags	Small Bags
9173	47	2015-02-01	1.83	1228.51	33.12	99.36	0.0	1096.03	1096.03
9174	48	2015-01-25	1.89	1115.89	14.87	148.72	0.0	952.30	952.30
9175	49	2015-01-18	1.93	1118.47	8.02	178.78	0.0	931.67	931.67
9176	50	2015-01-11	1.77	1182.56	39.00	305.12	0.0	838.44	838.44
9177	51	2015-01-04	1.79	1373.95	57.42	153.88	0.0	1162.65	1162.65
...	...	...	...	...	...	...	...	...	...
11928	46	2015-02-01	1.77	7210.19	1634.42	3012.44	0.0	2563.33	2563.33
11929	47	2015-01-25	1.63	7324.06	1934.46	3032.72	0.0	2356.88	2320.00
11930	48	2015-01-18	1.71	5508.20	1793.64	2078.72	0.0	1635.84	1620.00
11931	49	2015-01-11	1.69	6861.73	1822.28	2377.54	0.0	2661.91	2656.66
11932	50	2015-01-04	1.64	6182.81	1561.30	2958.17	0.0	1663.34	1663.34

270 rows × 14 columns

Subsetting using .isin()

In [25]:
```
# subset the avocado in the region Boston or SanDiego
regionFilter = avocado["region"].isin(["Boston", "SanDiego"])
avocado[regionFilter]
```

Out[25]:

	Unnamed: 0	Date	AveragePrice	Total Volume	4046	4225	4770	Total Bags
208	0	2015-12-27	1.13	450816.39	3886.27	346964.70	13952.56	86012.86
209	1	2015-12-20	1.07	489802.88	4912.37	390100.99	5887.72	88901.80
210	2	2015-12-13	1.01	549945.76	4641.02	455362.38	219.40	89722.96
211	3	2015-12-06	1.02	488679.31	5126.32	407520.22	142.99	75889.78
212	4	2015-11-29	1.19	350559.81	3609.25	272719.08	105.86	74125.62
...	...	...	...	...	...	...	...	...
18100	7	2018-02-04	1.81	17454.74	1158.41	7388.27	0.00	8908.06
18101	8	2018-01-28	1.91	17579.47	1145.64	8284.41	0.00	8149.42
18102	9	2018-01-21	1.95	18676.37	1088.49	9282.37	0.00	8305.51
18103	10	2018-01-14	1.81	21770.02	3285.98	14338.52	0.00	4145.52
18104	11	2018-01-07	2.06	16746.82	5150.82	9366.31	0.00	2229.69

676 rows × 14 columns

Multiple parameter Filtering

In [26]:
```
# subset the avocado in the region Boston or SanDiego in the year 2016 or 2017
regionFilter = avocado["region"].isin(["Boston", "SanDiego"])
#yearFilter = avocado["year"].isin(["2016", "2017"])
avocado[regionFilter]
```

Out[26]:

	Unnamed: 0	Date	AveragePrice	Total Volume	4046	4225	4770	Total Bags
208	0	2015-12-27	1.13	450816.39	3886.27	346964.70	13952.56	86012.86
209	1	2015-12-20	1.07	489802.88	4912.37	390100.99	5887.72	88901.80
210	2	2015-12-13	1.01	549945.76	4641.02	455362.38	219.40	89722.96
211	3	2015-12-06	1.02	488679.31	5126.32	407520.22	142.99	75889.78
212	4	2015-11-29	1.19	350559.81	3609.25	272719.08	105.86	74125.62
...	...	...	...	...	...	...	...	...
18100	7	2018-02-04	1.81	17454.74	1158.41	7388.27	0.00	8908.06
18101	8	2018-01-28	1.91	17579.47	1145.64	8284.41	0.00	8149.42
18102	9	2018-01-21	1.95	18676.37	1088.49	9282.37	0.00	8305.51
18103	10	2018-01-14	1.81	21770.02	3285.98	14338.52	0.00	4145.52
18104	11	2018-01-07	2.06	16746.82	5150.82	9366.31	0.00	2229.69

676 rows × 14 columns

Detecting missing values.isna()

In [27]:
```
avocado.isna()
```

Out[27]:

	Unnamed: 0	Date	AveragePrice	Total Volume	4046	4225	4770	Total Bags	Small Bags	Large Bags	X
0	False	False	False	False	False	False	False	False	False	False	
1	False	False	False	False	False	False	False	False	False	False	
2	False	False	False	False	False	False	False	False	False	False	
3	False	False	False	False	False	False	False	False	False	False	
4	False	False	False	False	False	False	False	False	False	False	
...	...	...	...	...	...	...	...	...	...	...	
18244	False	False	False	False	False	False	False	False	False	False	
18245	False	False	False	False	False	False	False	False	False	False	
18246	False	False	False	False	False	False	False	False	False	False	
18247	False	False	False	False	False	False	False	False	False	False	
18248	False	False	False	False	False	False	False	False	False	False	

18249 rows × 14 columns

In [28]:
```
avocado.isna().any()
```

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

Counting missing values

In [29]:
```
avocado.isna().sum()
```

Out[29]:
```
Unnamed: 0      0
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
```

Removing missing values

In [30]:
```
# Luckily we don't have any NaN but if we have we can use any of the two methods

avocado.dropna()

# ****  OR  ****

meanVal = avocado["AveragePrice"].mean()
avocado.fillna(meanVal)
```

Out[30]:

	Unnamed: 0	Date	AveragePrice	Total Volume	4046	4225	4770	Total Bags	
0	0	2015-12-27	1.33	64236.62	1036.74	54454.85	48.16	8696.87	8
1	1	2015-12-20	1.35	54876.98	674.28	44638.81	58.33	9505.56	9
2	2	2015-12-13	0.93	118220.22	794.70	109149.67	130.50	8145.35	8
3	3	2015-12-06	1.08	78992.15	1132.00	71976.41	72.58	5811.16	5
4	4	2015-11-29	1.28	51039.60	941.48	43838.39	75.78	6183.95	5
...	...	...	...	...	...	...	...	...	
18244	7	2018-02-04	1.63	17074.83	2046.96	1529.20	0.00	13498.67	13
18245	8	2018-01-28	1.71	13888.04	1191.70	3431.50	0.00	9264.84	8
18246	9	2018-01-21	1.87	13766.76	1191.92	2452.79	727.94	9394.11	9
18247	10	2018-01-14	1.93	16205.22	1527.63	2981.04	727.01	10969.54	10
18248	11	2018-01-07	1.62	17489.58	2894.77	2356.13	224.53	12014.15	11

18249 rows × 14 columns

Adding a new column

In [31]:
```
avocado["AveragePricePer100"] = avocado["AveragePrice"] * 100
avocado
```

Out[31]:

	Unnamed: 0	Date	AveragePrice	Total Volume	4046	4225	4770	Total Bags	
0	0	2015-12-27	1.33	64236.62	1036.74	54454.85	48.16	8696.87	8
1	1	2015-12-20	1.35	54876.98	674.28	44638.81	58.33	9505.56	9
2	2	2015-12-13	0.93	118220.22	794.70	109149.67	130.50	8145.35	8
3	3	2015-12-06	1.08	78992.15	1132.00	71976.41	72.58	5811.16	5
4	4	2015-11-29	1.28	51039.60	941.48	43838.39	75.78	6183.95	5
...	...	...	...	...	...	...	...	...	
18244	7	2018-02-04	1.63	17074.83	2046.96	1529.20	0.00	13498.67	1
18245	8	2018-01-28	1.71	13888.04	1191.70	3431.50	0.00	9264.84	8
18246	9	2018-01-21	1.87	13766.76	1191.92	2452.79	727.94	9394.11	9
18247	10	2018-01-14	1.93	16205.22	1527.63	2981.04	727.01	10969.54	10
18248	11	2018-01-07	1.62	17489.58	2894.77	2356.13	224.53	12014.15	11

18249 rows × 15 columns

Deleting columns in dataframe

In [32]:
```
avocado.drop(["AveragePricePer100"],axis = 1)
```

Out[32]:

	Unnamed: 0	Date	AveragePrice	Total Volume	4046	4225	4770	Total Bags	
0	0	2015-12-27	1.33	64236.62	1036.74	54454.85	48.16	8696.87	8
1	1	2015-12-20	1.35	54876.98	674.28	44638.81	58.33	9505.56	9
2	2	2015-12-13	0.93	118220.22	794.70	109149.67	130.50	8145.35	8
3	3	2015-12-06	1.08	78992.15	1132.00	71976.41	72.58	5811.16	5
4	4	2015-11-29	1.28	51039.60	941.48	43838.39	75.78	6183.95	5
...	...	...	...	...	...	...	...	...	
18244	7	2018-02-04	1.63	17074.83	2046.96	1529.20	0.00	13498.67	13
18245	8	2018-01-28	1.71	13888.04	1191.70	3431.50	0.00	9264.84	8
18246	9	2018-01-21	1.87	13766.76	1191.92	2452.79	727.94	9394.11	9
18247	10	2018-01-14	1.93	16205.22	1527.63	2981.04	727.01	10969.54	10
18248	11	2018-01-07	1.62	17489.58	2894.77	2356.13	224.53	12014.15	11

18249 rows × 14 columns

Summary statistics

In [33]:
```
avocado["AveragePrice"].mean()
```

Out[33]:
```
1.405978409775878
```

Summaring dates

In [34]:
```
avocado["Date"].max()
```

Out[34]:
```
'2018-03-25'
```

.agg() method

In [35]:
```
def pct30(column):
    #return the 0.3 quartile
    return column.quantile(0.3)
```

```
def pct50(column):
    #return the 0.5 quartile
    return column.quantile(0.5)

avocado[["AveragePrice","Total Bags"]].agg([pct30,pct50])
```

Out[35]:

	AveragePrice	Total Bags
pct30	1.15	7316.634
pct50	1.37	39743.830

Dropping duplicate names.drop_duplicates(lst)

In [36]:
```
temp = avocado.drop_duplicates(subset=["year"])
temp
```

Out[36]:

	Unnamed: 0	Date	AveragePrice	Total Volume	4046	4225	4770	Total Bags
0	0	2015-12-27	1.33	64236.62	1036.74	54454.85	48.16	8696.87
2808	0	2016-12-25	1.52	73341.73	3202.39	58280.33	426.92	11432.09
5616	0	2017-12-31	1.47	113514.42	2622.70	101135.53	20.25	9735.94
8478	0	2018-03-25	1.57	149396.50	16361.69	109045.03	65.45	23924.33

Count categorical data

In [37]:
```
# count number of avocado in each year in descending order
avocado["year"].value_counts(sort=True, ascending = False)
```

Out[37]:
```
year
2017    5722
2016    5616
2015    5615
2018    1296
Name: count, dtype: int64
```

Grouped summaries

In [38]:
```
# group by multiple columns and perform multiple summary statistic operations
avocado.groupby(["year","type"])["AveragePrice"].agg([min,max,np.mean,np.median])
```

```
C:\Users\Sonu\AppData\Local\Temp\ipykernel_21124\3377443975.py:2: FutureWarning: The
provided callable <built-in function min> is currently using SeriesGroupBy.min. In a
future version of pandas, the provided callable will be used directly. To keep curre
nt behavior pass the string "min" instead.
  avocado.groupby(["year","type"])["AveragePrice"].agg([min,max,np.mean,np.median])
C:\Users\Sonu\AppData\Local\Temp\ipykernel_21124\3377443975.py:2: FutureWarning: The
provided callable <built-in function max> is currently using SeriesGroupBy.max. In a
future version of pandas, the provided callable will be used directly. To keep curre
nt behavior pass the string "max" instead.
  avocado.groupby(["year","type"])["AveragePrice"].agg([min,max,np.mean,np.median])
C:\Users\Sonu\AppData\Local\Temp\ipykernel_21124\3377443975.py:2: FutureWarning: The
provided callable <function mean at 0x00000247A73A25C0> is currently using SeriesGro
upBy.mean. In a future version of pandas, the provided callable will be used directl
y. To keep current behavior pass the string "mean" instead.
  avocado.groupby(["year","type"])["AveragePrice"].agg([min,max,np.mean,np.median])
C:\Users\Sonu\AppData\Local\Temp\ipykernel_21124\3377443975.py:2: FutureWarning: The
provided callable <function median at 0x00000247A74D9080> is currently using SeriesG
roupBy.median. In a future version of pandas, the provided callable will be used dir
ectly. To keep current behavior pass the string "median" instead.
  avocado.groupby(["year","type"])["AveragePrice"].agg([min,max,np.mean,np.median])
```

Out[38]:

		min	max	mean	median
year	**type**				
2015	**conventional**	0.49	1.59	1.077963	1.08
	organic	0.81	2.79	1.673324	1.67
2016	**conventional**	0.51	2.20	1.105595	1.08
	organic	0.58	3.25	1.571684	1.53
2017	**conventional**	0.46	2.22	1.294888	1.30
	organic	0.44	3.17	1.735521	1.72
2018	**conventional**	0.56	1.74	1.127886	1.14
	organic	1.01	2.30	1.567176	1.55

Pivot table

In [39]:
```
# this is the same table we build in the previous cell but using pivot table
avocado.pivot_table(index=["year","type"], aggfunc=[min,max,np.mean,np.median], val
```

```
C:\Users\Sonu\AppData\Local\Temp\ipykernel_21124\762502195.py:2: FutureWarning: The
provided callable <built-in function min> is currently using DataFrameGroupBy.min. I
n a future version of pandas, the provided callable will be used directly. To keep c
urrent behavior pass the string "min" instead.
  avocado.pivot_table(index=["year","type"], aggfunc=[min,max,np.mean,np.median], va
lues="AveragePrice")
C:\Users\Sonu\AppData\Local\Temp\ipykernel_21124\762502195.py:2: FutureWarning: The
provided callable <built-in function max> is currently using DataFrameGroupBy.max. I
n a future version of pandas, the provided callable will be used directly. To keep c
urrent behavior pass the string "max" instead.
  avocado.pivot_table(index=["year","type"], aggfunc=[min,max,np.mean,np.median], va
lues="AveragePrice")
C:\Users\Sonu\AppData\Local\Temp\ipykernel_21124\762502195.py:2: FutureWarning: The
provided callable <function mean at 0x00000247A73A25C0> is currently using DataFrame
GroupBy.mean. In a future version of pandas, the provided callable will be used dire
ctly. To keep current behavior pass the string "mean" instead.
  avocado.pivot_table(index=["year","type"], aggfunc=[min,max,np.mean,np.median], va
lues="AveragePrice")
C:\Users\Sonu\AppData\Local\Temp\ipykernel_21124\762502195.py:2: FutureWarning: The
provided callable <function median at 0x00000247A74D9080> is currently using DataFra
meGroupBy.median. In a future version of pandas, the provided callable will be used
directly. To keep current behavior pass the string "median" instead.
  avocado.pivot_table(index=["year","type"], aggfunc=[min,max,np.mean,np.median], va
lues="AveragePrice")
```

Out[39]:

		min	max	mean	median
		AveragePrice	AveragePrice	AveragePrice	AveragePrice
year	type				
2015	conventional	0.49	1.59	1.077963	1.08
	organic	0.81	2.79	1.673324	1.67
2016	conventional	0.51	2.20	1.105595	1.08
	organic	0.58	3.25	1.571684	1.53
2017	conventional	0.46	2.22	1.294888	1.30
	organic	0.44	3.17	1.735521	1.72
2018	conventional	0.56	1.74	1.127886	1.14
	organic	1.01	2.30	1.567176	1.55

Explict indexes

In [40]:
```
regionIndex = avocado.set_index(["region"])
regionIndex
```

Out[40]:

region	Unnamed: 0	Date	AveragePrice	Total Volume	4046	4225	4770
Albany	0	2015-12-27	1.33	64236.62	1036.74	54454.85	48.16
Albany	1	2015-12-20	1.35	54876.98	674.28	44638.81	58.33
Albany	2	2015-12-13	0.93	118220.22	794.70	109149.67	130.50
Albany	3	2015-12-06	1.08	78992.15	1132.00	71976.41	72.58
Albany	4	2015-11-29	1.28	51039.60	941.48	43838.39	75.78
...	...	...	...	...	...	...	...
WestTexNewMexico	7	2018-02-04	1.63	17074.83	2046.96	1529.20	0.00
WestTexNewMexico	8	2018-01-28	1.71	13888.04	1191.70	3431.50	0.00
WestTexNewMexico	9	2018-01-21	1.87	13766.76	1191.92	2452.79	727.94
WestTexNewMexico	10	2018-01-14	1.93	16205.22	1527.63	2981.04	727.01
WestTexNewMexico	11	2018-01-07	1.62	17489.58	2894.77	2356.13	224.53

18249 rows × 14 columns

In [41]:
```
# Insted of doing this
avocado[avocado["region"].isin(["Albany", "WestTexNewMexico"])]
```

Out[41]:

	Unnamed: 0	Date	AveragePrice	Total Volume	4046	4225	4770	Total Bags	
0	0	2015-12-27	1.33	64236.62	1036.74	54454.85	48.16	8696.87	8
1	1	2015-12-20	1.35	54876.98	674.28	44638.81	58.33	9505.56	9
2	2	2015-12-13	0.93	118220.22	794.70	109149.67	130.50	8145.35	8
3	3	2015-12-06	1.08	78992.15	1132.00	71976.41	72.58	5811.16	5
4	4	2015-11-29	1.28	51039.60	941.48	43838.39	75.78	6183.95	5
...	...	...	...	...	...	...	...	...	
18244	7	2018-02-04	1.63	17074.83	2046.96	1529.20	0.00	13498.67	13
18245	8	2018-01-28	1.71	13888.04	1191.70	3431.50	0.00	9264.84	8
18246	9	2018-01-21	1.87	13766.76	1191.92	2452.79	727.94	9394.11	9
18247	10	2018-01-14	1.93	16205.22	1527.63	2981.04	727.01	10969.54	10
18248	11	2018-01-07	1.62	17489.58	2894.77	2356.13	224.53	12014.15	11

673 rows × 15 columns

In [42]:
```
# we can simply do
regionIndex.loc[["Albany", "WestTexNewMexico"]]
```

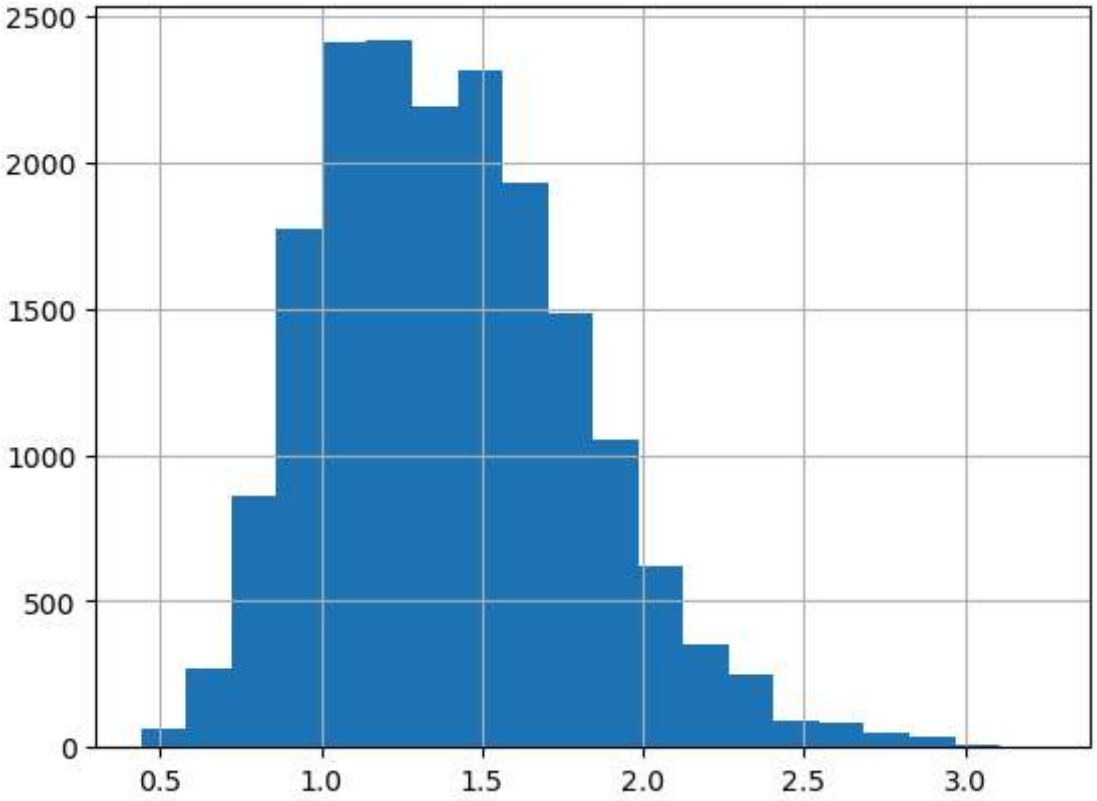
Out[42]:

region	Unnamed: 0	Date	AveragePrice	Total Volume	4046	4225	4770
Albany	0	2015-12-27	1.33	64236.62	1036.74	54454.85	48.16
Albany	1	2015-12-20	1.35	54876.98	674.28	44638.81	58.33
Albany	2	2015-12-13	0.93	118220.22	794.70	109149.67	130.50
Albany	3	2015-12-06	1.08	78992.15	1132.00	71976.41	72.58
Albany	4	2015-11-29	1.28	51039.60	941.48	43838.39	75.78
...	...	...	...	...	...	...	...
WestTexNewMexico	7	2018-02-04	1.63	17074.83	2046.96	1529.20	0.00
WestTexNewMexico	8	2018-01-28	1.71	13888.04	1191.70	3431.50	0.00
WestTexNewMexico	9	2018-01-21	1.87	13766.76	1191.92	2452.79	727.94
WestTexNewMexico	10	2018-01-14	1.93	16205.22	1527.63	2981.04	727.01
WestTexNewMexico	11	2018-01-07	1.62	17489.58	2894.77	2356.13	224.53

673 rows × 14 columns

Visualizing your data

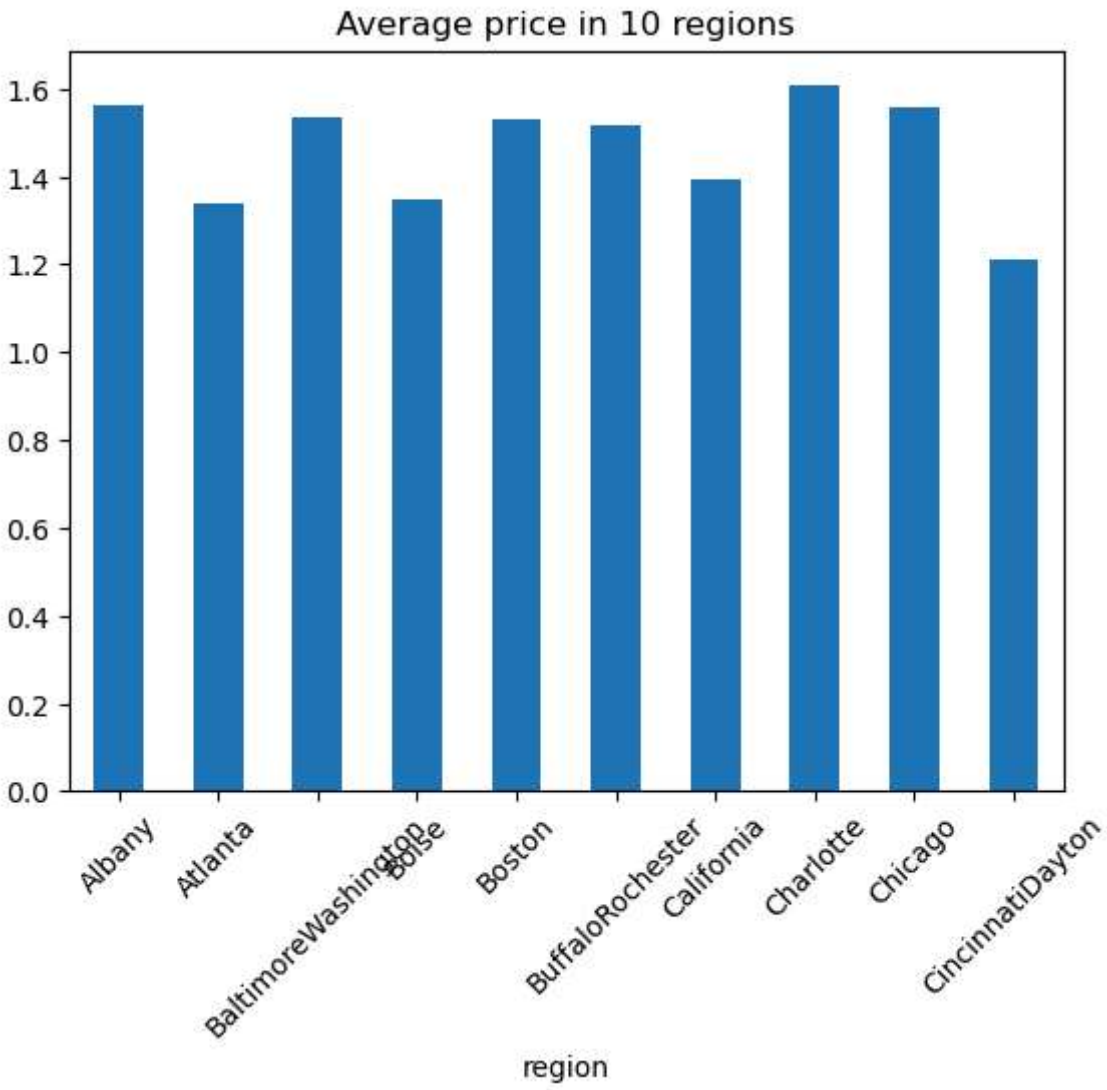
In [43]:
```
# Histograms
avocado["AveragePrice"].hist(bins=20)
plt.show()
```



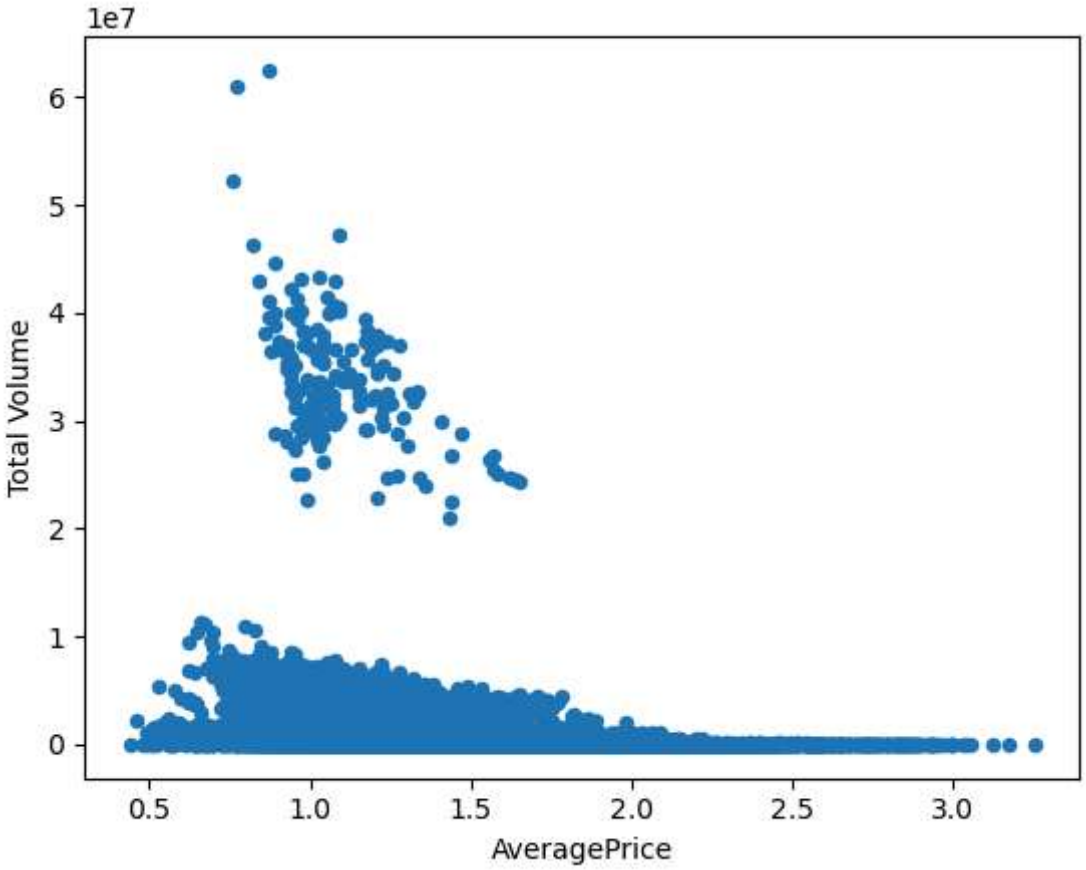
In [44]:
```
#Barplots
regionFilter = avocado.groupby("region")["AveragePrice"].mean().head(10)
regionFilter
```

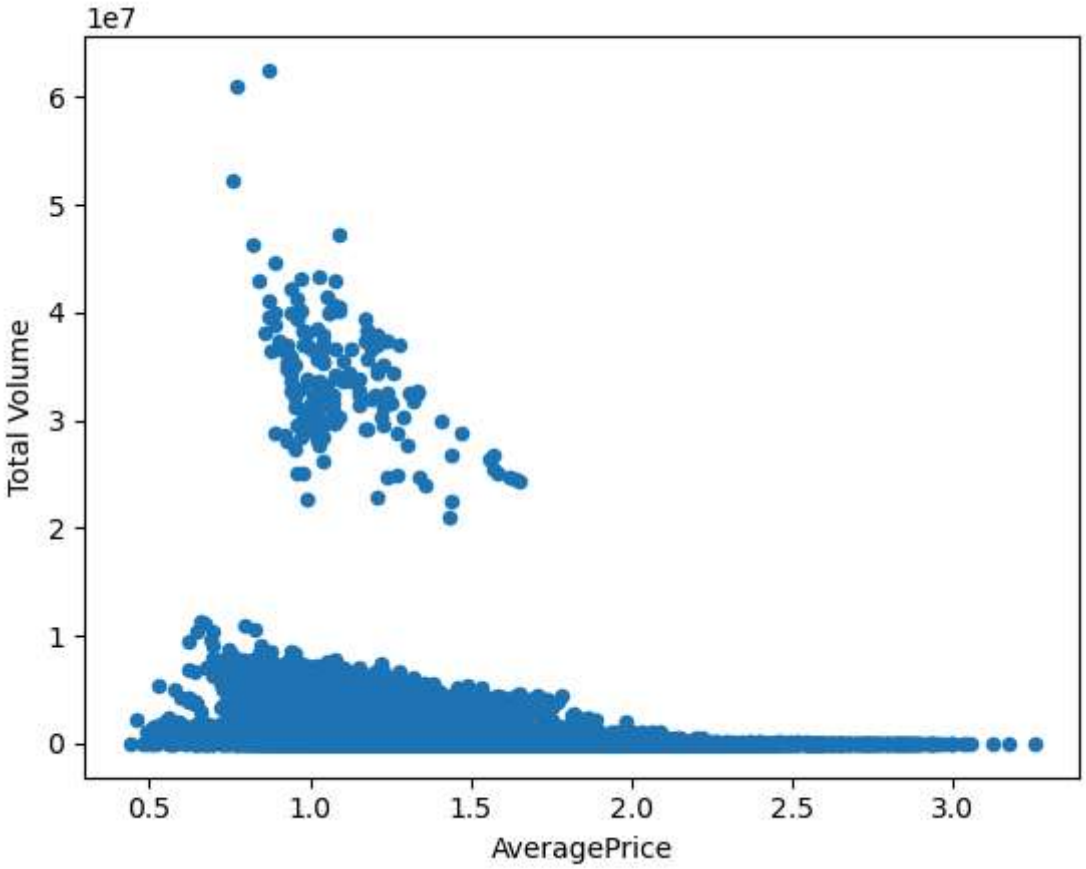
Out[44]:
```
region
Albany                1.561036
Atlanta               1.337959
BaltimoreWashington   1.534231
Boise                 1.348136
Boston                1.530888
BuffaloRochester      1.516834
California            1.395325
Charlotte             1.606036
Chicago               1.556775
CincinnatiDayton      1.209201
Name: AveragePrice, dtype: float64
```

In [46]:
```
regionFilter.plot(kind = "bar",rot=45,title="Average price in 10 regions")
plt.show()
```



In [48]:
```
#scatterplot
avocado.plot(x="AveragePrice", y="Total Volume", kind="scatter")
plt.show()
```





In [49]:
```
# subtract AveragePrice with AveragePrice :P
# Dah its 0
```

```
avocado["AveragePrice"].sub(avocado["AveragePrice"])
```

Out[49]:
```
0        0.0
1        0.0
2        0.0
3        0.0
4        0.0
        ...
18244    0.0
18245    0.0
18246    0.0
18247    0.0
18248    0.0
Name: AveragePrice, Length: 18249, dtype: float64
```

In [ ]:
```

```