Part 1 Question 2: Analyze the avocado data

Tasks:

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

1. Read the data from the CSV file into a DataFrame.

In [ ]:
```
df = pd.read_csv('datasets/avocado.csv')
df
```

Out[ ]:

	Unnamed: 0	Date	AveragePrice	Total Volume	4046	4225	4770	To Ba
0	0	2015-12-27	1.33	64236.62	1036.74	54454.85	48.16	8696
1	1	2015-12-20	1.35	54876.98	674.28	44638.81	58.33	9505
2	2	2015-12-13	0.93	118220.22	794.70	109149.67	130.50	8145
3	3	2015-12-06	1.08	78992.15	1132.00	71976.41	72.58	5811
4	4	2015-11-29	1.28	51039.60	941.48	43838.39	75.78	6183
...	...	...	...	...	...	...	...	
18244	7	2018-02-04	1.63	17074.83	2046.96	1529.20	0.00	13498
18245	8	2018-01-28	1.71	13888.04	1191.70	3431.50	0.00	9264
18246	9	2018-01-21	1.87	13766.76	1191.92	2452.79	727.94	9394
18247	10	2018-01-14	1.93	16205.22	1527.63	2981.04	727.01	10969
18248	11	2018-01-07	1.62	17489.58	2894.77	2356.13	224.53	12014

18249 rows × 14 columns

2. Display type memory consumption and null count information using the info() method.

```
In [ ]: df.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
```

3. Display the number of unique values in each column.

```
In [ ]: df.nunique()
```

```
Out[ ]: Unnamed: 0         53
        Date              169
        AveragePrice      259
        Total Volume    18237
        4046            17702
        4225            18103
        4770            12071
        Total Bags      18097
        Small Bags      17321
        Large Bags      15082
        XLarge Bags      5588
        type                2
        year                4
        region             54
        dtype: int64
```

4. Display all the rows of data that JupyterLab displays by default.

```
In [ ]: df
```

Out[ ]:

	Unnamed: 0	Date	AveragePrice	Total Volume	4046	4225	4770	To Ba
0	0	2015-12-27	1.33	64236.62	1036.74	54454.85	48.16	8696
1	1	2015-12-20	1.35	54876.98	674.28	44638.81	58.33	9505
2	2	2015-12-13	0.93	118220.22	794.70	109149.67	130.50	8145
3	3	2015-12-06	1.08	78992.15	1132.00	71976.41	72.58	5811
4	4	2015-11-29	1.28	51039.60	941.48	43838.39	75.78	6183
...	...	...	...	...	...	...	...	
18244	7	2018-02-04	1.63	17074.83	2046.96	1529.20	0.00	13498
18245	8	2018-01-28	1.71	13888.04	1191.70	3431.50	0.00	9264
18246	9	2018-01-21	1.87	13766.76	1191.92	2452.79	727.94	9394
18247	10	2018-01-14	1.93	16205.22	1527.63	2981.04	727.01	10969
18248	11	2018-01-07	1.62	17489.58	2894.77	2356.13	224.53	12014

18249 rows × 14 columns

5. Display the first and last five rows of data and the first and last four columns of data.

In [ ]:

```
pd.set_option('display.max_rows', 10)
pd.set_option('display.max_columns', 8)
df
```

Out[ ]:

	Unnamed: 0	Date	AveragePrice	Total Volume	...	XLarge Bags	type	year	
0	0	2015-12-27	1.33	64236.62	...	0.0	conventional	2015	
1	1	2015-12-20	1.35	54876.98	...	0.0	conventional	2015	
2	2	2015-12-13	0.93	118220.22	...	0.0	conventional	2015	
3	3	2015-12-06	1.08	78992.15	...	0.0	conventional	2015	
4	4	2015-11-29	1.28	51039.60	...	0.0	conventional	2015	
...	...	...	...	...	...	...	...	...	
18244	7	2018-02-04	1.63	17074.83	...	0.0	organic	2018	We
18245	8	2018-01-28	1.71	13888.04	...	0.0	organic	2018	We
18246	9	2018-01-21	1.87	13766.76	...	0.0	organic	2018	We
18247	10	2018-01-14	1.93	16205.22	...	0.0	organic	2018	We
18248	11	2018-01-07	1.62	17489.58	...	0.0	organic	2018	We

18249 rows × 14 columns

6. Choose any three columns access them with bracket notation and display the first five rows of this data.

In [ ]:

```
df[['Date', 'AveragePrice', 'Total Volume']].head(5)
```

Out[ ]:

	Date	AveragePrice	Total Volume
0	2015-12-27	1.33	64236.62
1	2015-12-20	1.35	54876.98
2	2015-12-13	0.93	118220.22
3	2015-12-06	1.08	78992.15
4	2015-11-29	1.28	51039.60

7. Select one column and access it with dot notation.

In [ ]:

```
df.Date
```

Out[ ]:

```
0        2015-12-27
1        2015-12-20
2        2015-12-13
3        2015-12-06
4        2015-11-29
            ...
18244    2018-02-04
18245    2018-01-28
18246    2018-01-21
18247    2018-01-14
18248    2018-01-07
Name: Date, Length: 18249, dtype: object
```

8. Multiply the Total Volume and AveragePrice columns and store the result in a new column called EstimatedRevenue. Then display the first five rows of this data to confirm that the column was added and has the correct values.

In [ ]:

```
df['EstimatedRevenue'] = df['AveragePrice'] * df['Total Volume']
df.head(5)
```

Out[ ]:

	Unnamed: 0	Date	AveragePrice	Total Volume	...	type	year	region	Estimate
0	0	2015-12-27	1.33	64236.62	...	conventional	2015	Albany	8
1	1	2015-12-20	1.35	54876.98	...	conventional	2015	Albany	7
2	2	2015-12-13	0.93	118220.22	...	conventional	2015	Albany	10
3	3	2015-12-06	1.08	78992.15	...	conventional	2015	Albany	8
4	4	2015-11-29	1.28	51039.60	...	conventional	2015	Albany	6

5 rows × 15 columns

9. Create a DataFrame that's grouped by region and type and that includes the average price for the grouped columns. Then reset the index and display the first five rows.

In [ ]:

```
df[['region', 'type', 'AveragePrice']].groupby(['region', 'type']).mean().re
```

Out[ ]:

	region	type	AveragePrice
0	Albany	conventional	1.348757
1	Albany	organic	1.773314
2	Atlanta	conventional	1.068817
3	Atlanta	organic	1.607101
4	BaltimoreWashington	conventional	1.344201

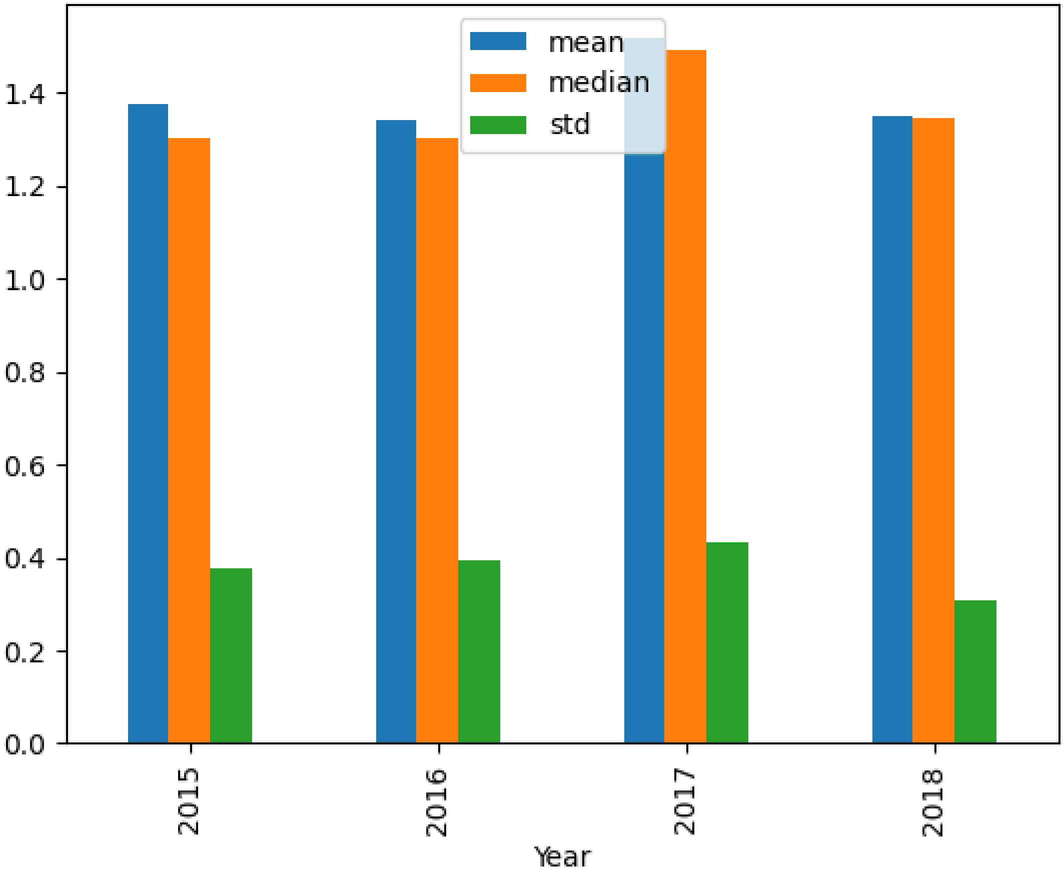
10. Create a bar plot that shows the mean median and standard deviation of the Total Volume column by year.

In [ ]:

```
df['Year'] = pd.to_datetime(df['Date']).dt.year
plot_df = df[['Year', 'AveragePrice']].groupby('Year').agg(['mean', 'median'
plot_df.plot.bar(x='Year', y='AveragePrice')
```

Out[ ]:

```
<Axes: xlabel='Year'>
```



Questions:

1. How many unique regions are there?

```
In [ ]: df['region'].nunique()
```

```
Out[ ]: 54
```

There are 54 unique regions in the dataset

2. What is the average price for each type of avocado (organic and conventional)? Be sure to include just the type and AveragePrice columns in the results.

```
In [ ]: df[['type', 'AveragePrice']].groupby('type').mean().reset_index()
```

Out[ ]:

	type	AveragePrice
0	conventional	1.158040
1	organic	1.653999

The average price for each type of avocado is as follows:

1. conventional - $1.16
2. organic - $1.65

3. Which region has the lowest average price for organic avocados? Hint: Create wide data from the grouped data that you created in task 8.

```
In [ ]: wide_df = df[['region', 'type', 'AveragePrice']].groupby(['region', 'type'])
        wide_df[wide_df['type'] == 'organic'].sort_values('AveragePrice', ascending=
```

Out[ ]:

	region	type	AveragePrice
37	Houston	organic	1.270769

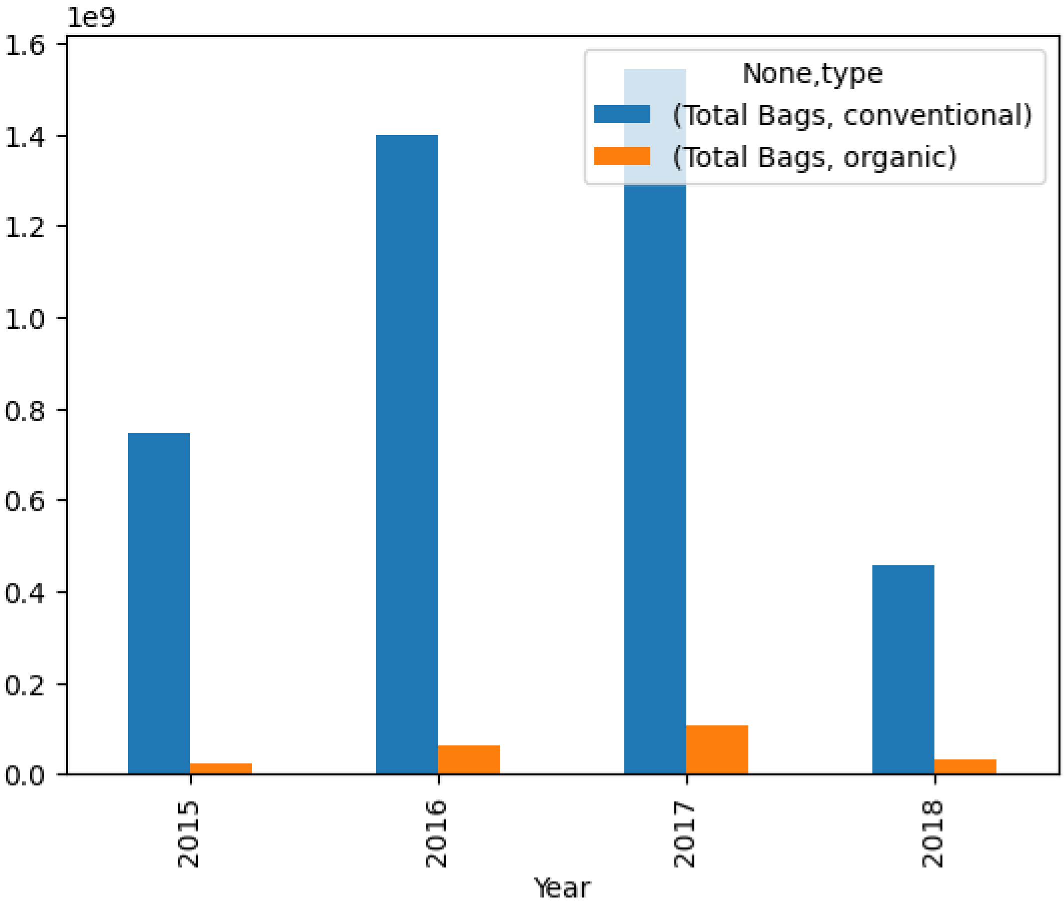
The region with the lowest average organic avocado price is Houstan at $1.27

4. Have the Total Bags sold per year of each type of avocado become more or less consistent over time?

```
In [ ]: plot_df = df[['Year', 'type', 'Total Bags']].groupby(['Year', 'type']).sum()

        plot_df.plot.bar()
```

```
Out[ ]: <Axes: xlabel='Year'>
```



The graph above represents the total bags sold per year of each type of avocado. From this graph it is clear that the total bags sold per year has become less consistent over time, this can be seen with the significant drop from 2017 to 2018.