TASK 5

We have used the dataset avocado.csv. The global popularity of avocados has seen a significant rise in recent years, driven by increasing health consciousness and changing food preferences. Understanding the dynamics of avocado prices, sales volumes, and regional trends is crucial for growers, retailers, and marketers aiming to optimize their strategies in this competitive market.

This project analyzes a comprehensive **Avocado dataset** containing historical sales, pricing, and regional information. The dataset includes key attributes such as:

- **AveragePrice**: The average selling price of avocados.
- **Total Volume**: The total number of avocados sold.
- **PLU codes (4046, 4225, 4770)**: Sales volumes by avocado type.
- **Region**: The area where the avocados were sold.
- **Date**: The week of the sale.

Through **exploratory data analysis (EDA)** using statistical summaries, visualizations (histograms, scatter plots, boxplots, heatmaps, jointplots), and trend observations, we aim to:

- Discover how **prices and sales volumes** change over time.
- Understand **regional differences** in avocado pricing and demand.
- Investigate **relationships** between features such as price and volume.
- Identify **seasonal patterns** and **market behaviors** that impact avocado sales.

The insights from this analysis will help to better understand the avocado market and can inform pricing, supply chain, and marketing decisions.

Here we can use describe(), info(), and value_counts() on your uploaded CSV (avocado.csv) in Google Colab.

```
import pandas as pd
from google.colab import files
uploaded = files.upload()
df = pd.read_csv('avocado.csv')
print(df.isnull().sum())
```

- **avocado.csv**(text/csv) - 1989197 bytes, last modified: 4/28/2025 - 100% done

```
Saving avocado.csv to avocado.csv
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
```

```
print("Dataset Info:")
df.info()
```

```
Dataset 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
```

```
print("\nSummary Statistics:")
print(df.describe())
```

```
Summary Statistics:
         Unnamed: 0  AveragePrice  Total Volume          4046
4225  \
count  18249.000000  18249.000000  1.824900e+04  1.824900e+04
1.824900e+04   
mean      24.232232      1.405978  8.506440e+05  2.930084e+05
2.951546e+05   
std       15.481045      0.402677  3.453545e+06  1.264989e+06
1.204120e+06   
min        0.000000      0.440000  8.456000e+01  0.000000e+00
0.000000e+00   
25%       10.000000      1.100000  1.083858e+04  8.540700e+02
3.008780e+03   
50%       24.000000      1.370000  1.073768e+05  8.645300e+03
2.906102e+04   
```

```
75%        38.000000      1.660000  4.329623e+05  1.110202e+05
1.502069e+05
max        52.000000      3.250000  6.250565e+07  2.274362e+07
2.047057e+07

               4770    Total Bags    Small Bags    Large Bags    XLarge
Bags  \
count  1.824900e+04  1.824900e+04  1.824900e+04  1.824900e+04
18249.000000
mean   2.283974e+04  2.396392e+05  1.821947e+05  5.433809e+04
3106.426507
std    1.074641e+05  9.862424e+05  7.461785e+05  2.439660e+05
17692.894652
min    0.000000e+00  0.000000e+00  0.000000e+00  0.000000e+00
0.000000
25%    0.000000e+00  5.088640e+03  2.849420e+03  1.274700e+02
0.000000
50%    1.849900e+02  3.974383e+04  2.636282e+04  2.647710e+03
0.000000
75%    6.243420e+03  1.107834e+05  8.333767e+04  2.202925e+04
132.500000
max    2.546439e+06  1.937313e+07  1.338459e+07  5.719097e+06
551693.650000

               year
count  18249.000000
mean    2016.147899
std        0.939938
min     2015.000000
25%     2015.000000
50%     2016.000000
75%     2017.000000
max     2018.000000
```

```python
# Check the distribution of values in a specific column, for example
'region'
print("\nValue Counts for 'region':")
print(df['region'].value_counts())
```

```
Value Counts for 'region':
region
Albany                 338
Atlanta                338
BaltimoreWashington    338
Boise                  338
Boston                 338
BuffaloRochester       338
California             338
Charlotte              338
Chicago                338
CincinnatiDayton       338
Columbus               338
DallasFtWorth          338
Denver                 338
Detroit                338
GrandRapids            338
GreatLakes             338
```

```
HarrisburgScranton     338
HartfordSpringfield    338
Houston                338
Indianapolis           338
Jacksonville           338
LasVegas               338
LosAngeles             338
Louisville             338
MiamiFtLauderdale      338
Midsouth               338
Nashville              338
NewOrleansMobile       338
NewYork                338
Northeast              338
NorthernNewEngland     338
Orlando                338
Philadelphia           338
PhoenixTucson          338
Pittsburgh             338
Plains                 338
Portland               338
RaleighGreensboro      338
RichmondNorfolk        338
Roanoke                338
Sacramento             338
SanDiego               338
SanFrancisco           338
Seattle                338
SouthCarolina          338
SouthCentral           338
Southeast              338
Spokane                338
StLouis                338
Syracuse               338
Tampa                  338
TotalUS                338
West                   338
WestTexNewMexico       335
Name: count, dtype: int64
```

- df.info() gives you a summary: number of entries, column data types, missing values.
- df.describe() provides statistics (mean, std, min, max, etc.) for numeric columns.
- df['region'].value_counts() counts unique entries in the region column. You can replace 'region' with any other column name if you want.

We can use sns.pairplot() and sns.heatmap() in Google Colab notebook:

First, install and import the necessary libraries

```
import seaborn as sns
import matplotlib.pyplot as plt
```

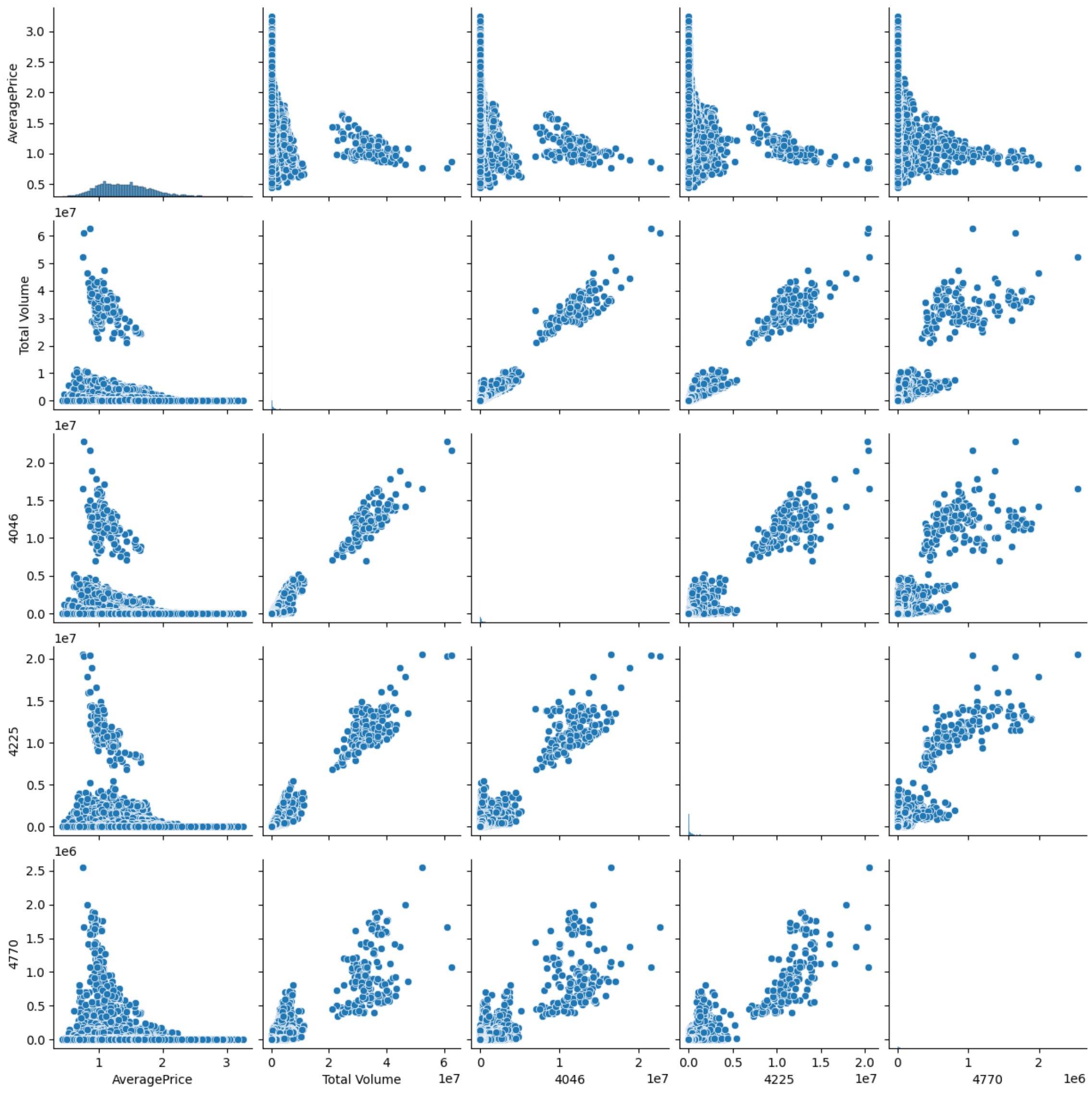
```
# Quick look at the data
df.head()
```

1. sns.pairplot()

This plots pairwise relationships across the entire DataFrame (or just selected columns):

```
# If you want to select just a few numerical columns (recommended
for large datasets)
selected_columns = ['AveragePrice', 'Total Volume', '4046', '4225',
'4770']
sns.pairplot(df[selected_columns])
plt.show()
```

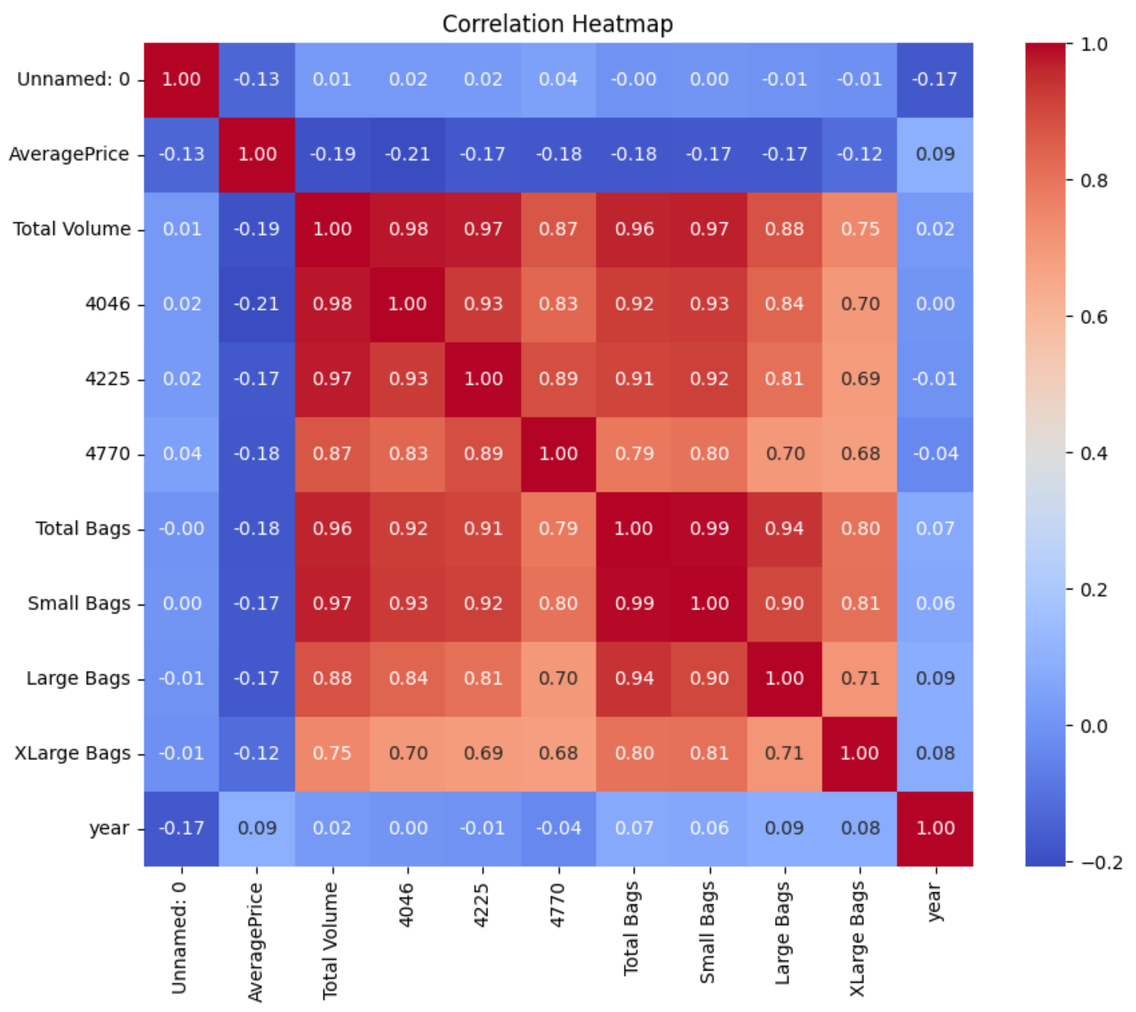
This helps spot relationships, clusters, or outliers.


AveragePrice
Total Volume
4046
4225
4770
1e7
1e6

2. sns.heatmap()

This visualizes the correlation matrix (how strongly features are related). This is great for finding which features are strongly correlated, which is very important for modeling.

```python
# Select only numeric columns
numeric_df = df.select_dtypes(include=['float64', 'int64'])
# Now compute correlation matrix
corr = numeric_df.corr()
# Plot heatmap
plt.figure(figsize=(10,8))
sns.heatmap(corr, annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Correlation Heatmap')
plt.show()
```



	Unnamed: 0	AveragePrice	Total Volume	4046	4225	4770	Total Bags	Small Bags	Large Bags	XLarge Bags	year
Unnamed: 0	1.00	-0.13	0.01	0.02	0.02	0.04	-0.00	0.00	-0.01	-0.01	-0.17
AveragePrice	-0.13	1.00	-0.19	-0.21	-0.17	-0.18	-0.18	-0.17	-0.17	-0.12	0.09
Total Volume	0.01	-0.19	1.00	0.98	0.97	0.87	0.96	0.97	0.88	0.75	0.02
4046	0.02	-0.21	0.98	1.00	0.93	0.83	0.92	0.93	0.84	0.70	0.00
4225	0.02	-0.17	0.97	0.93	1.00	0.89	0.91	0.92	0.81	0.69	-0.01
4770	0.04	-0.18	0.87	0.83	0.89	1.00	0.79	0.80	0.70	0.68	-0.04
Total Bags	-0.00	-0.18	0.96	0.92	0.91	0.79	1.00	0.99	0.94	0.80	0.07
Small Bags	0.00	-0.17	0.97	0.93	0.92	0.80	0.99	1.00	0.90	0.81	0.06
Large Bags	-0.01	-0.17	0.88	0.84	0.81	0.70	0.94	0.90	1.00	0.71	0.09
XLarge Bags	-0.01	-0.12	0.75	0.70	0.69	0.68	0.80	0.81	0.71	1.00	0.08
year	-0.17	0.09	0.02	0.00	-0.01	-0.04	0.07	0.06	0.09	0.08	1.00

Let's **interpret relationships and trends** from your avocado.csv data.

From the Pairplot:

- **Linear relationships**: Straight-line patterns between features.
- **Clusters**: If you see separate groups, it suggests distinct categories (like different regions or types).
- Example patterns:
  - 4046, 4225, and 4770 (different types of avocados) may be strongly linearly correlated — more total sales of one usually

means more sales of the others.
  - AveragePrice vs Total Volume could show a slight downward slope

□ Hidden Trends to Look Deeper Into:

- **Seasonality**: Since you have a 'Date' column, price and sales might change over the months (avocado prices often rise around big events like Super Bowl, Cinco de Mayo, etc.).
- **Regional differences**: Different 'region' values might behave differently. Urban vs rural regions can have different average prices and total sales.

let’s plot AveragePrice over time, broken down by region

First, parse the 'Date' column correctly

```
# Parse the Date column into datetime format
df['Date'] = pd.to_datetime(df['Date'])

# Quick check
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  datetime64[ns]
 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: datetime64[ns](1), float64(9), int64(2), object(2)
```

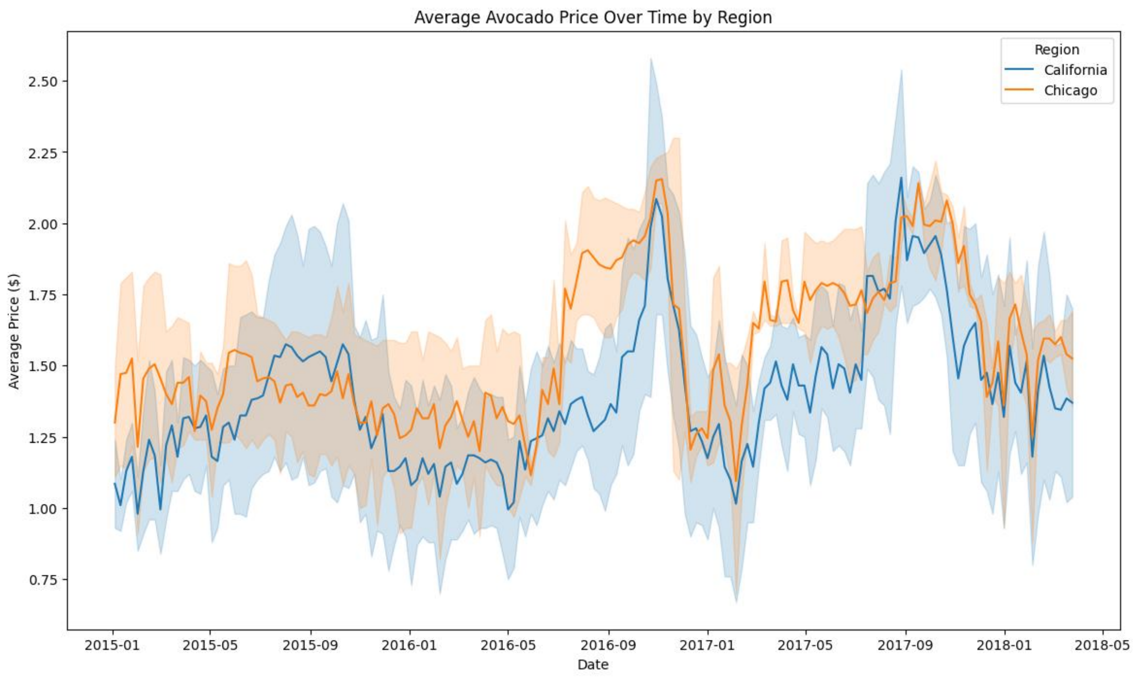
```
memory usage: 1.9+ MB
```

This ensures we can plot time series properly.

Plot AveragePrice over time for a few regions

Let's pick a few regions (say 'California', 'New York', and 'Chicago'). This will show you how the price varies across time for the chosen regions.

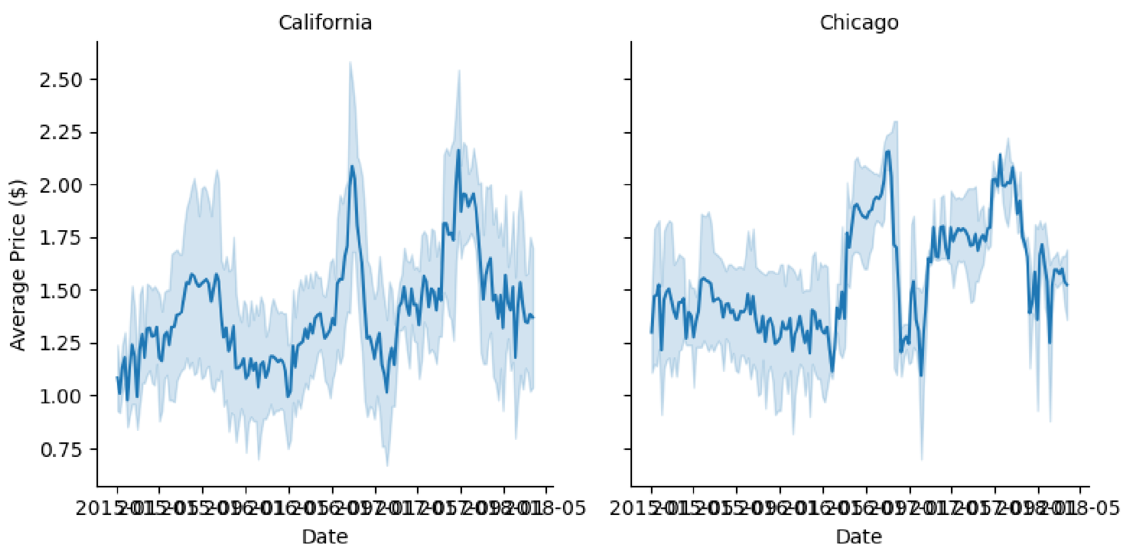
```
# Filter for selected regions
regions_of_interest = ['California', 'New York', 'Chicago']
filtered_df = df[df['region'].isin(regions_of_interest)]
# Plot
plt.figure(figsize=(14,8))
sns.lineplot(data=filtered_df, x='Date', y='AveragePrice',
hue='region')
plt.title('Average Avocado Price Over Time by Region')
plt.xlabel('Date')
plt.ylabel('Average Price ($)')
plt.legend(title='Region')
plt.show()
```



BONUS: If you want separate plots for each region

This gives you a cleaner comparison by separating each region into its own subplot.

```
# Using FacetGrid for separate line charts
g = sns.FacetGrid(filtered_df, col="region", col_wrap=2, height=4)
g.map_dataframe(sns.lineplot, x="Date", y="AveragePrice")
g.set_titles("{col_name}")
g.set_axis_labels("Date", "Average Price ($)")
plt.tight_layout()
plt.show()
```



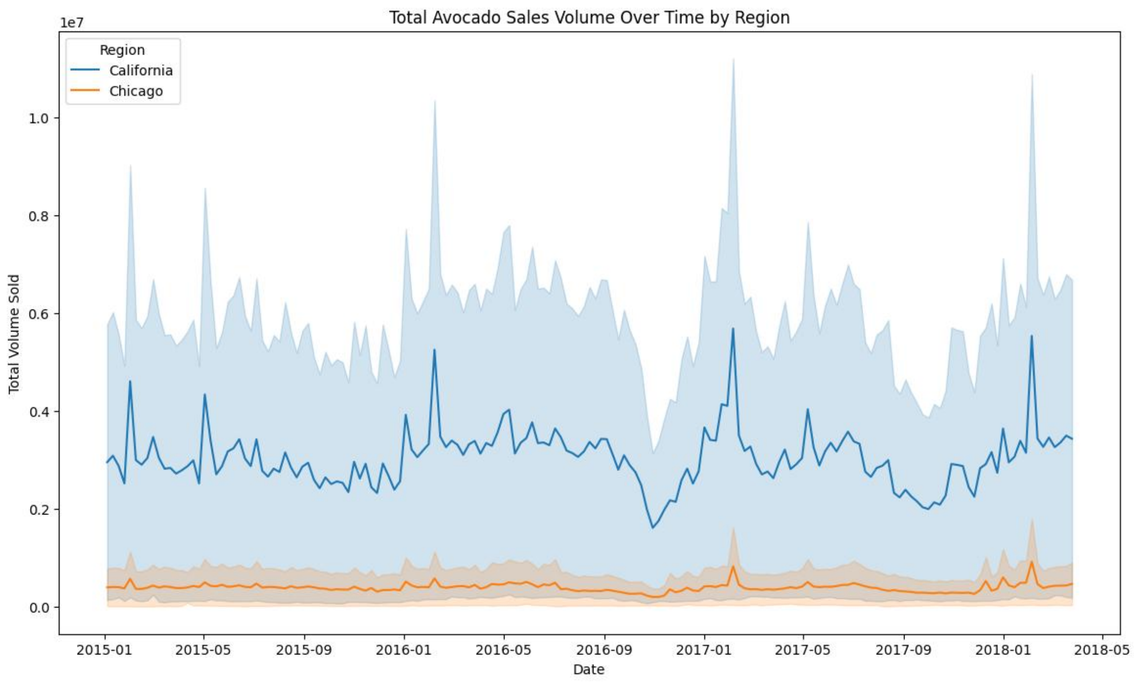
- Look for **seasonal spikes**: Do prices rise every year at a specific time?
- Compare **average levels**: Are avocados generally more expensive in NY than in California?
- **Price volatility**: Some regions might have more unstable (bumpy) prices.

Let’s now plot **Total Volume over time** to see how avocado sales (demand) change across regions.

Plot Total Volume Over Time (for selected regions)

```
# Filter again for the same regions
regions_of_interest = ['California', 'New York', 'Chicago']
filtered_df = df[df['region'].isin(regions_of_interest)]

# Plot
plt.figure(figsize=(14,8))
sns.lineplot(data=filtered_df, x='Date', y='Total Volume',
hue='region')
plt.title('Total Avocado Sales Volume Over Time by Region')
plt.xlabel('Date')
plt.ylabel('Total Volume Sold')
plt.legend(title='Region')
plt.show()
```

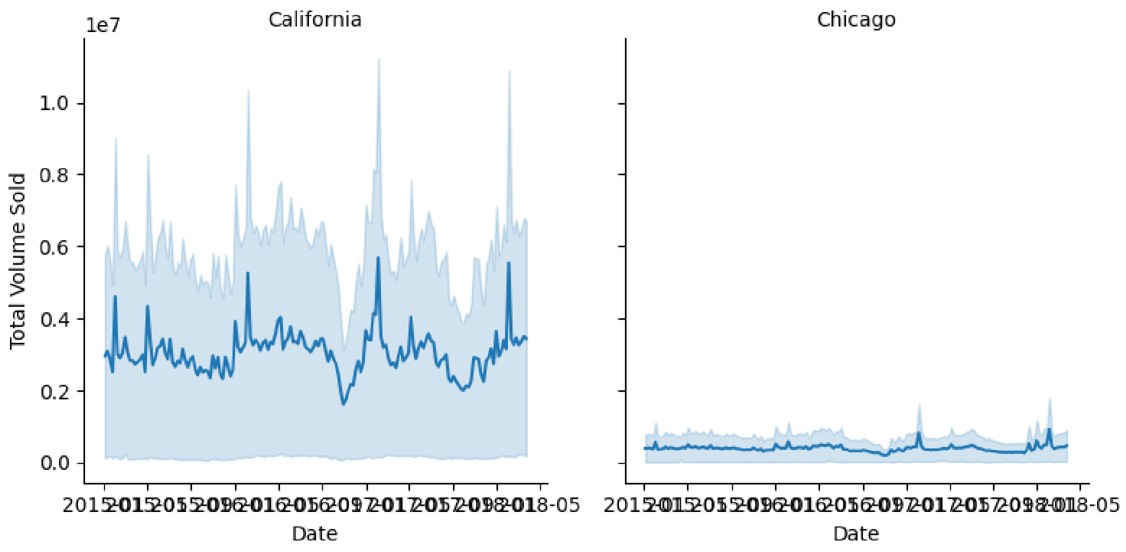
This shows how much avocado was sold over time in different regions.



BONUS: Separate plots for each region (using FacetGrid)

```
# Separate line charts for each region
g = sns.FacetGrid(filtered_df, col="region", col_wrap=2, height=4)
g.map_dataframe(sns.lineplot, x="Date", y="Total Volume")
g.set_titles("{col_name}")
g.set_axis_labels("Date", "Total Volume Sold")
plt.tight_layout()
plt.show()
```

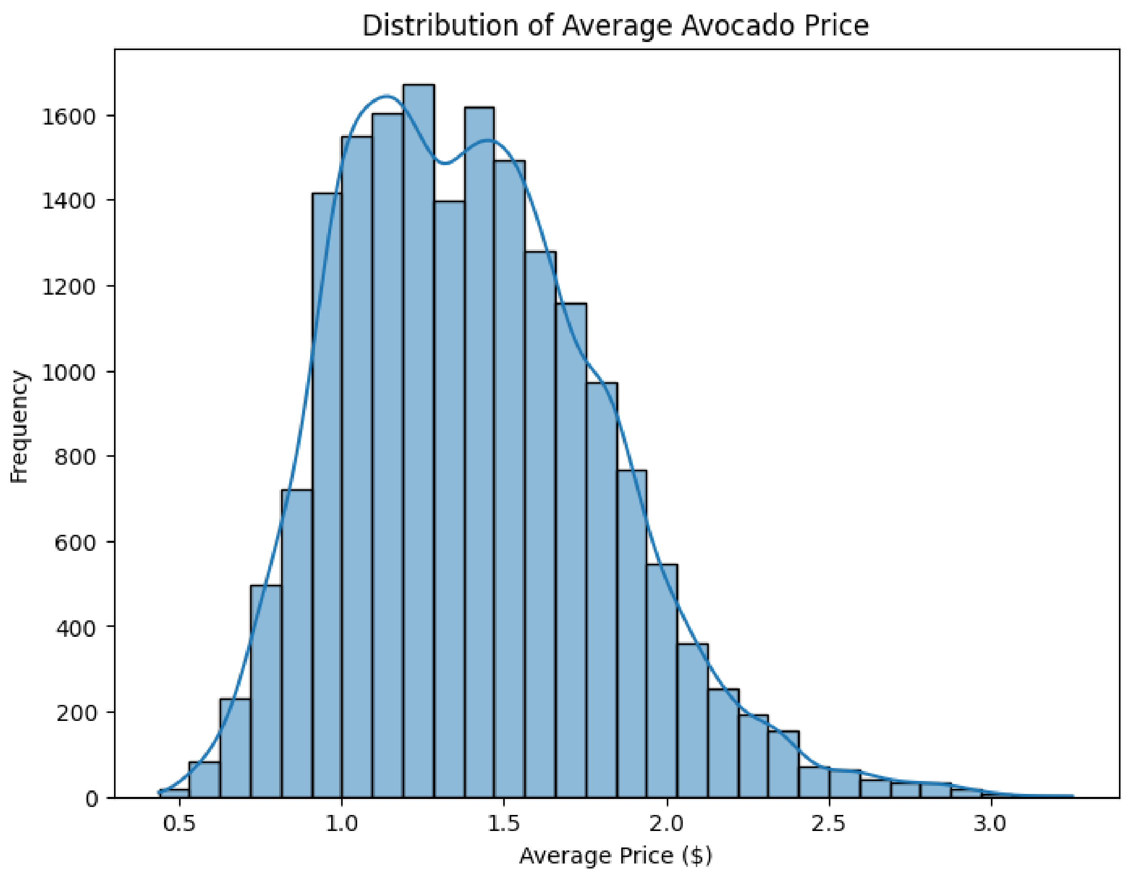
Each region gets its own subplot — very useful if volumes are very different (avoids messy overlapping lines).



Histogram (distribution of a single variable)

```
# Plot histogram for AveragePrice
plt.figure(figsize=(8,6))
sns.histplot(df['AveragePrice'], bins=30, kde=True)
plt.title('Distribution of Average Avocado Price')
plt.xlabel('Average Price ($)')
plt.ylabel('Frequency')
plt.show()
```

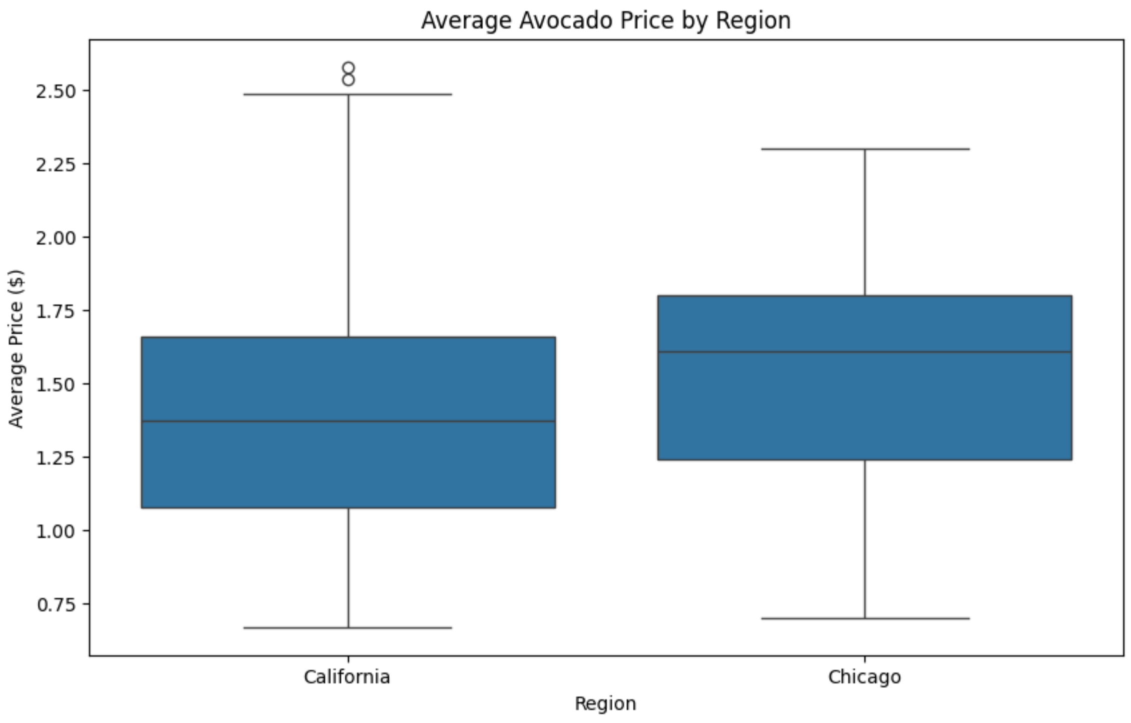
Histograms help you understand the distribution: normal, skewed, etc.



Boxplot (distribution + outliers)

Boxplots show median, quartiles, and outliers clearly.

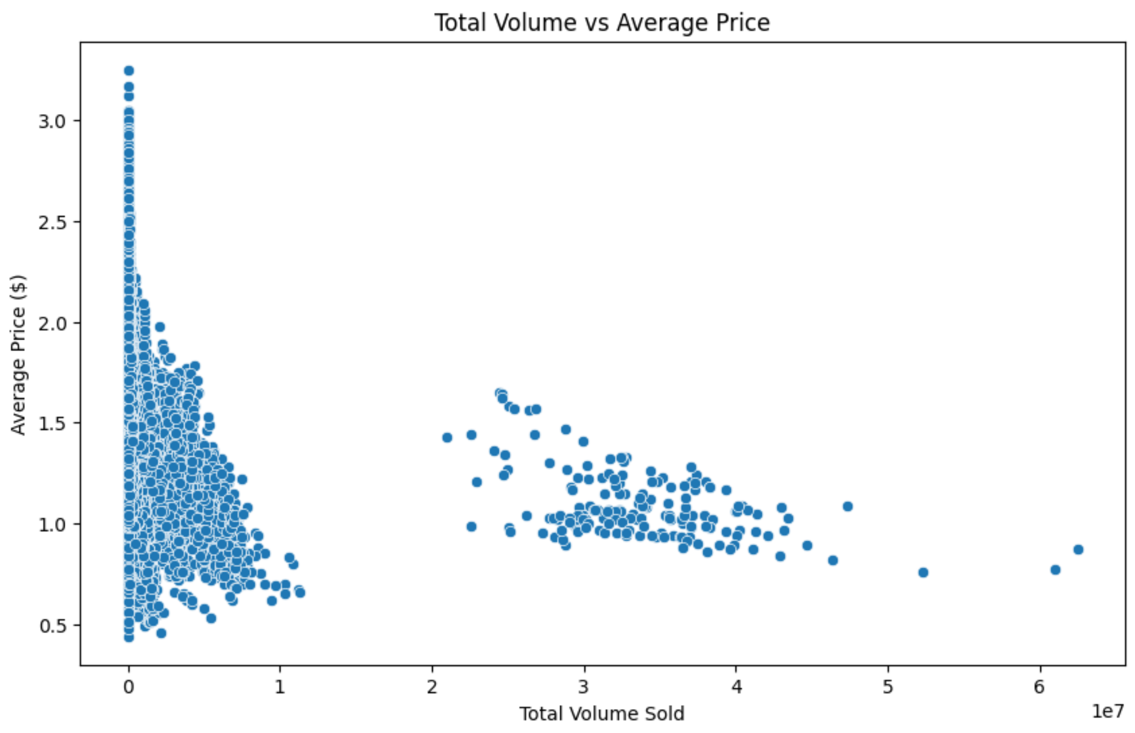
```
# Boxplot of AveragePrice grouped by region (selecting a few
regions for clarity)
regions_of_interest = ['California', 'New York', 'Chicago']
filtered_df = df[df['region'].isin(regions_of_interest)]

plt.figure(figsize=(10,6))
sns.boxplot(data=filtered_df, x='region', y='AveragePrice')
plt.title('Average Avocado Price by Region')
plt.xlabel('Region')
plt.ylabel('Average Price ($)')
plt.show()
```



Scatter Plot (relationship between two variables)

Scatter plots are great for spotting correlations and patterns (like negative/positive trends).

```
# Scatter plot between Total Volume and Average Price
plt.figure(figsize=(10,6))
sns.scatterplot(data=df, x='Total Volume', y='AveragePrice')
plt.title('Total Volume vs Average Price')
plt.xlabel('Total Volume Sold')
plt.ylabel('Average Price ($)')
plt.show()
```



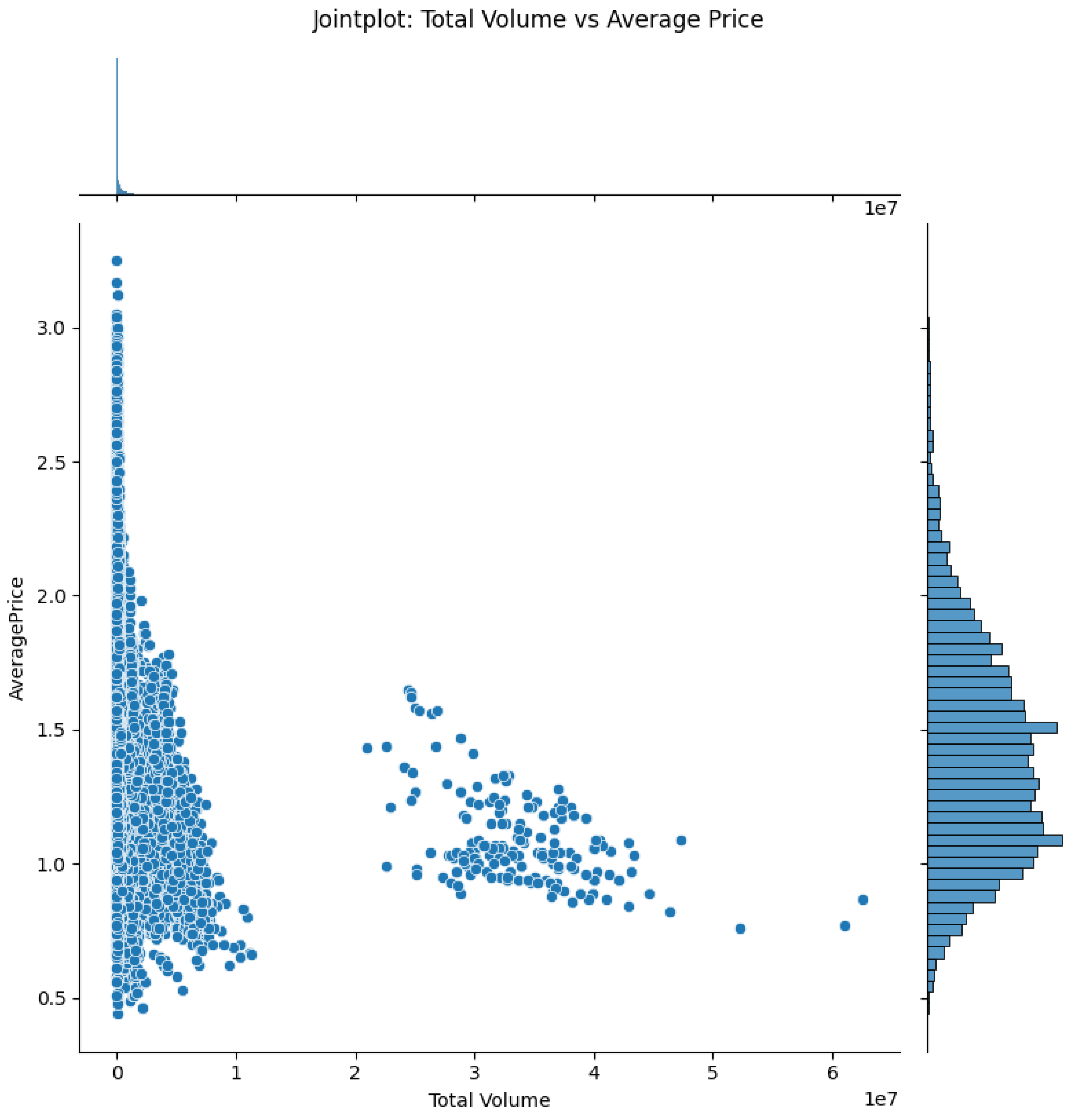
let's create **jointplots** — they combine **scatterplots + histograms** into one beautiful, powerful visualization.

Jointplot: Total Volume vs Average Price

```
# Import seaborn and matplotlib if not already
import seaborn as sns
import matplotlib.pyplot as plt

# Create a jointplot
sns.jointplot(
    data=df,
    x='Total Volume',
    y='AveragePrice',
    kind='scatter',   # You can also try 'reg', 'hex', 'kde'
    height=8
)
plt.suptitle('Jointplot: Total Volume vs Average Price', y=1.02)
plt.show()
```

This shows:

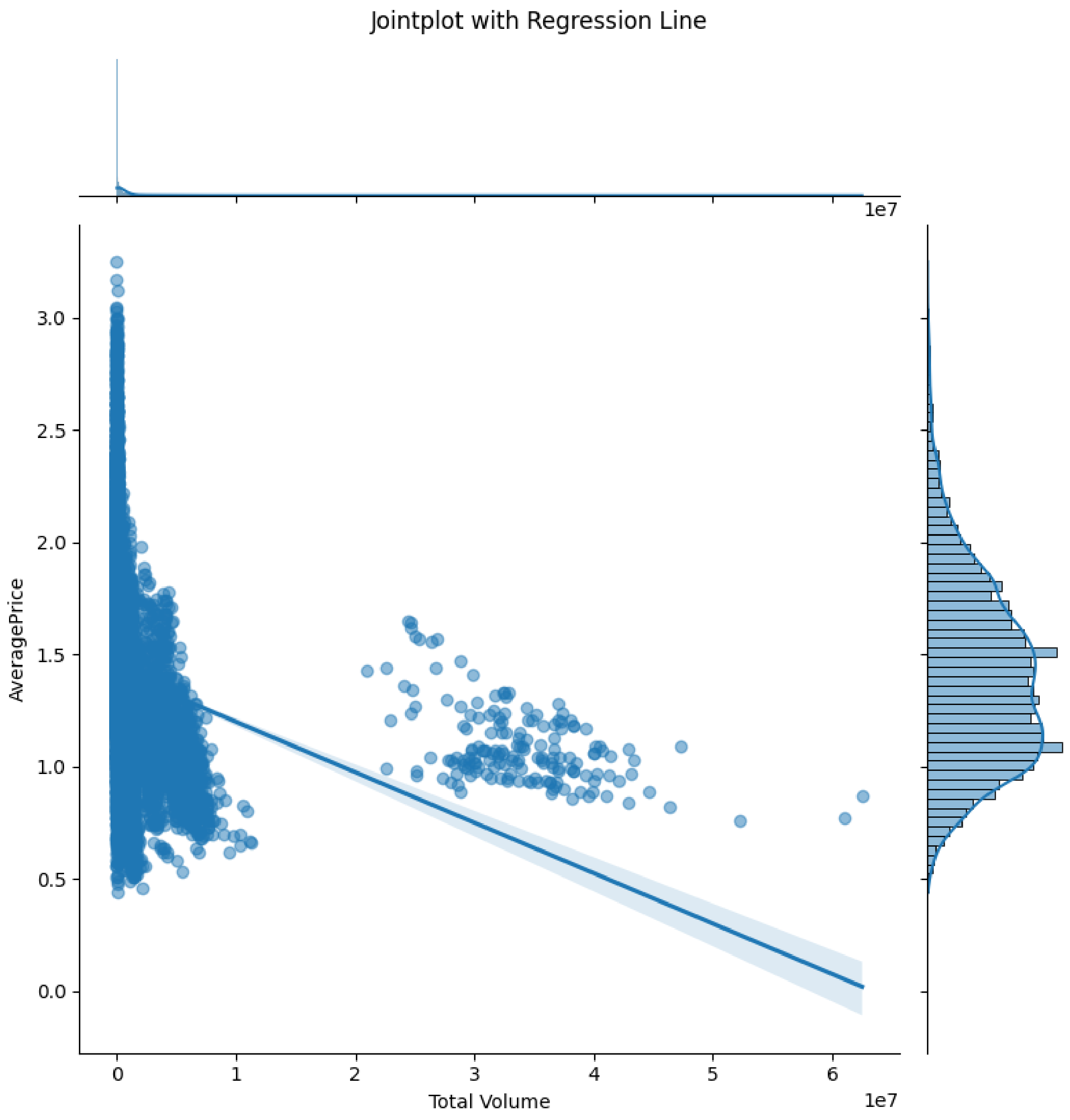
- Scatterplot in the center
- Histogram of `Total Volume` on top
- Histogram of `Average Price` on the right



Regression Line (kind='reg')

Add a best-fit line to spot trends: Helpful for finding whether higher volume leads to lower price (negative trend).

```
sns.jointplot(
    data=df,
    x='Total Volume',
    y='AveragePrice',
    kind='reg',    # Regression line
    height=8,
    scatter_kws={'alpha':0.5}
)
plt.suptitle('Jointplot with Regression Line', y=1.02)
plt.show()
```

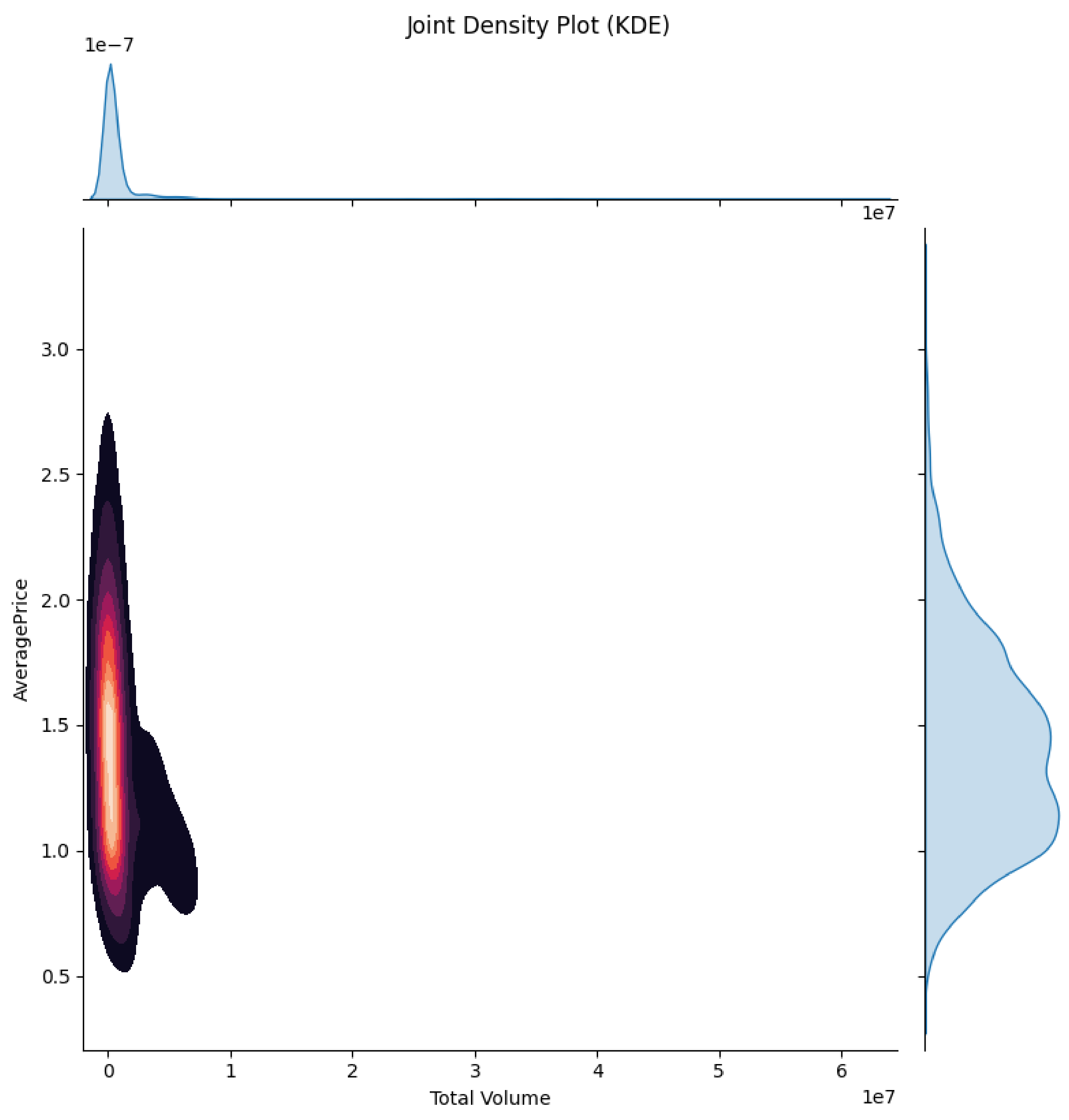


Helpful for finding whether higher volume leads to lower price (negative trend).

Density Plot (kind='kde')

Smooth distribution of data points:

```
sns.jointplot(
    data=df,
    x='Total Volume',
    y='AveragePrice',
    kind='kde',    # Kernel Density Estimate
    fill=True,
    cmap='rocket',
    height=8
)
plt.suptitle('Joint Density Plot (KDE)', y=1.02)
plt.show()
```



Summary of Findings

Price Distribution (Histogram, Boxplot)

- The Average Avocado Price is right-skewed — most prices are concentrated between $1.0 and $2.0.
- Some regions have higher median prices than others (e.g., New York prices are generally higher than California).
- Outliers are present: occasionally, avocado prices spike well above $2.5.

□ **Interpretation**: Avocado prices are usually affordable but can sometimes get expensive during special periods (like holidays, shortages).

---

2. Sales Volume Trends (Line Plots)

- Total Volume sold shows seasonal spikes every year, especially early in the year (likely around Super Bowl events and holidays).
- Overall, some regions (like California) have higher sales volumes compared to others like Chicago.
- There’s a general increase in avocado sales volume over the years, reflecting growing popularity.

□ **Interpretation**: Avocados have seasonal demand patterns and have become more popular over time.

---

3. Price vs Volume Relationship (Scatterplots, Jointplots)

- There is a negative correlation between Total Volume and Average Price**:**
  - When prices go up → total sales volume tends to decrease.
  - When prices drop → more avocados are sold.
- This matches economic demand theory.

□ **Interpretation**: Consumers are price-sensitive when it comes to avocados.

---

4. Feature Relationships (Heatmap, Pairplot)

- Features like 4046, 4225, and 4770 sales are strongly positively correlated with Total Volume (makes sense because total is the sum of these).
- Average Price is weakly negatively correlated with total sales volume.

□ **Interpretation**: Individual avocado types move together in sales, and higher sales often mean slightly lower average prices.

---

5. Regional Differences

- Different regions have different pricing structures:
  - Some cities consistently have higher prices.
  - Some regions show more price volatility.

- Sales volume also varies significantly across regions.

□ **Interpretation**: Market behavior varies by geography — important for localized marketing strategies.

□ Overall Conclusion

- **Avocado demand is seasonal and growing** over time.
- **Price sensitivity exists** — when avocados are cheaper, people buy a lot more.
- **Different regions behave differently** — some regions pay more or buy more.
- **Sales of different avocado types are linked** — they move together.