Oregon Real Estate Analysis by Nareg Koshanian

Introduction:

The real estate market has always been an interest of mine, so in this project we will analyze Oregon's real estate data over the past two decades. This research will look into the dataset, which contains key details such as sales prices, property descriptions, transaction dates, and others. Our goal is to look at the past and find trends within the Oregon real estate market.

Primary Setup:

To begin, there are certain modules we will need to import.

```python
import csv
import pandas as pd
import os
import matplotlib.pyplot as plt
import seaborn as sns
```

Data Preparation:

As I examined the csv file, I realized that there were many entries that lacked information and others that were replicas. In turn, I decided to clean up my data using the following code.

```python
''' Cleaning data:
-removing time from sale date
-removing when sale price <= 1000
-removing when the description says DO NOT USE!!
-removing duplicates

```

```
These changes led to the removal of about 4000 unnecessary entries
'''
filtered_rows = []

# Open the input CSV file
with open('Latest_Sales(2002to2024).csv', mode='r', newline='') as infile:
    csvreader = csv.reader(infile, delimiter=',')
    rows = list(csvreader)  # Read all rows into a list

    # Modify the date-time strings in each row
    for row in rows:

        # Cleaning according to line 1 description
        if row[5] <= '1000' or row[5] == '' or row[4] =='' or row[11] ==
'DO NOT USE!!':
            continue
        else:
            # Remove the time part from the date-time string
            date_time = row[4].split()[0]
            row[4] = date_time
            filtered_rows.append(row)

# Open the output CSV file and write the modified data
with open('Cleaned_Sales.csv', mode='w', newline='') as outfile:
    csvwriter = csv.writer(outfile, delimiter=',')
    csvwriter.writerows(filtered_rows)  # Write all rows back to the file

# Read the CSV file
df = pd.read_csv('Cleaned_Sales.csv')

# Remove duplicates based on specific columns
df = df.drop_duplicates(subset=['DocumentNumber', 'SalesDate',
'SalesPrice', 'Grantor'], keep='first')

# Save the result to a new CSV file
df.to_csv('NoDupes.csv', index=False)

# Rename the temporary file to the original file's name
os.replace('NoDupes.csv', 'Cleaned_Sales.csv')
```

Data Analysis:

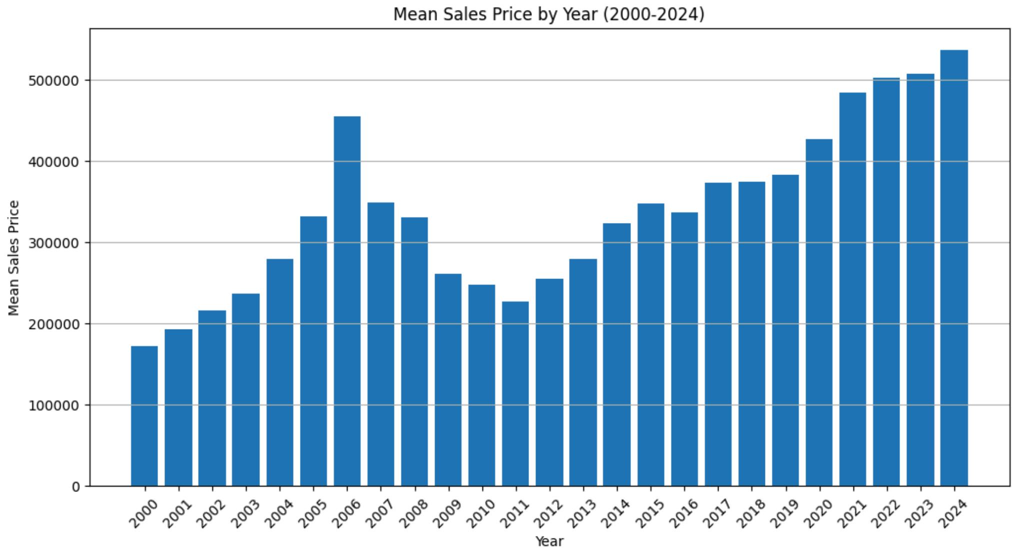
In this portion, we will use pandas to create the data frame and examine it.

```
# Read the CSV file into a pandas DataFrame
df = pd.read_csv('Cleaned_Sales.csv')

# Select only the "SalesPrice" and "SalesDate" columns
selected_columns = df[['SalesPrice', 'SalesDate']]

# Print the table containing only the selected columns
#print(selected_columns)
selected_columns.describe()
```

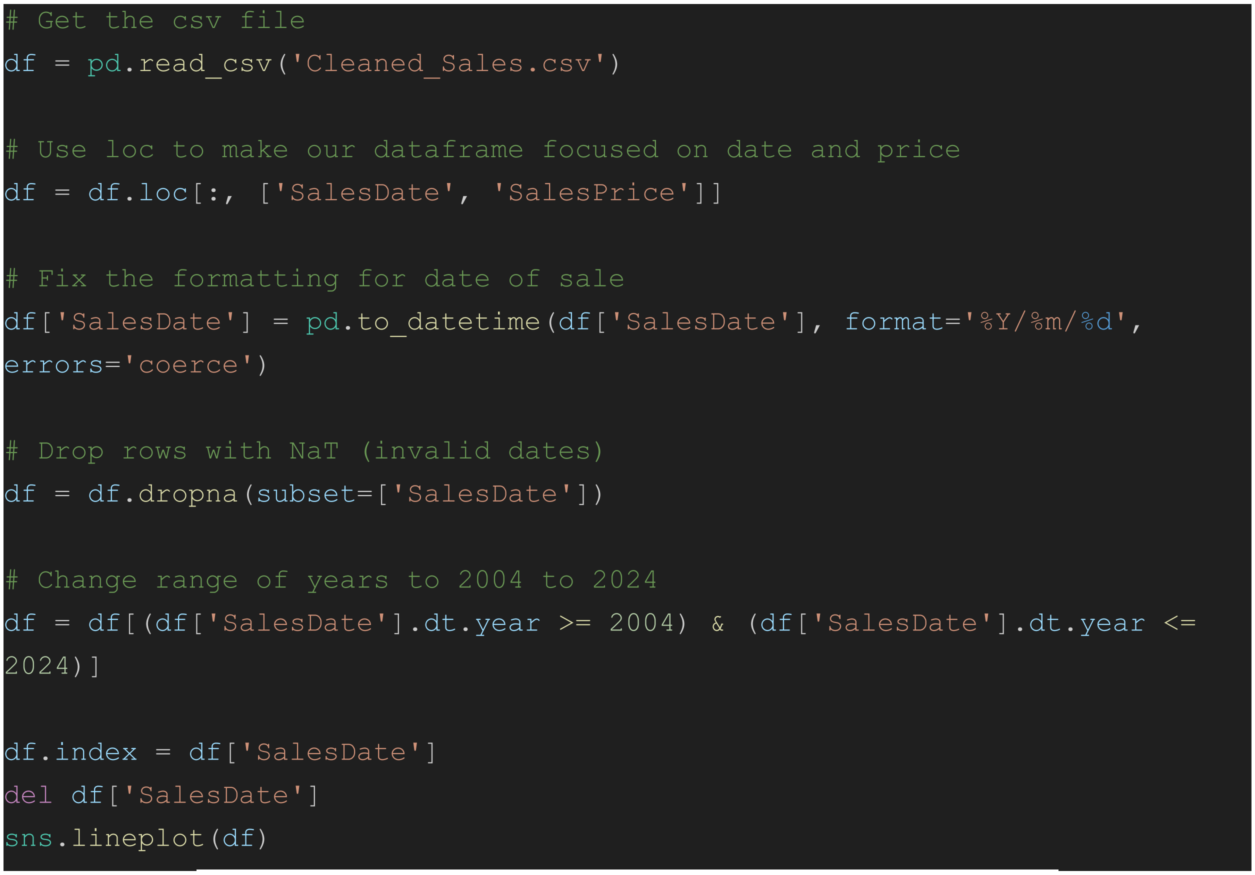
count	mean	std	min	25%	50%	75%	max
4.427600e+04	3.765550e+05	7.473410e+05	1.080000e+02	2.065000e+05	3.000000e+05	4.240000e+05	6.005913e+07

Now let’s make a bar graph to see the trend of home prices over the past 2.5 decades.

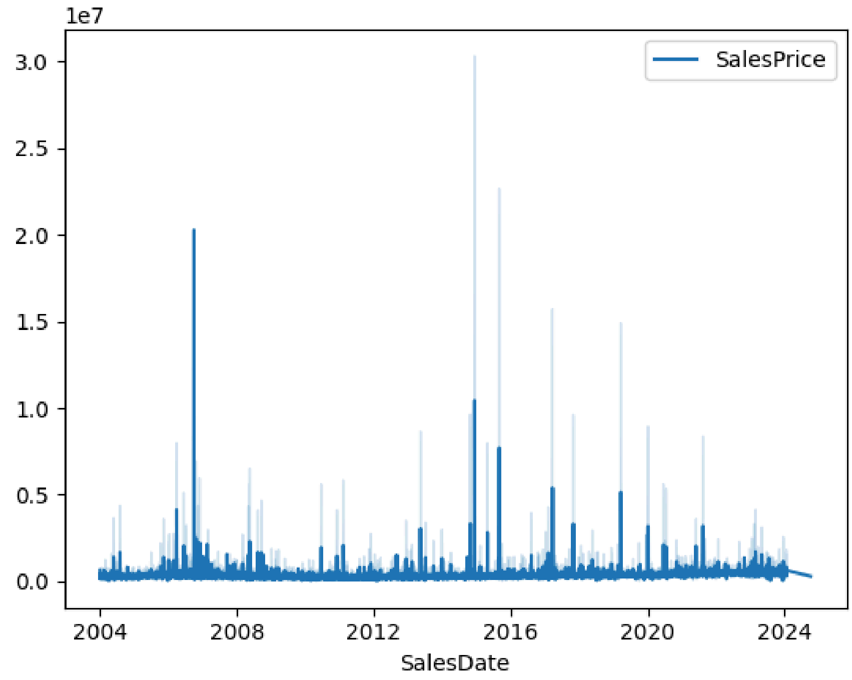
```
# Convert the 'SalesDate' column to datetime format
df['SalesDate'] = pd.to_datetime(df['SalesDate'], format='%Y/%m/%d',
errors='coerce')

# Drop rows with NaT (invalid dates)
df = df.dropna(subset=['SalesDate'])

# Extract the year from the 'SalesDate' column
df['Year'] = df['SalesDate'].dt.year
# Removing data before 2000 or after 2024
df = df[(df['Year'] >= 2000) & (df['Year'] <= 2024)]

# Group by year and calculate the mean sale price for each year
yearly_sales = df.groupby('Year')['SalesPrice'].mean()

# Create a bar graph with years on the x-axis and mean sale price on the
y-axis
plt.figure(figsize=(12, 6))
plt.bar(yearly_sales.index, yearly_sales.values)
plt.xlabel('Year')
plt.ylabel('Mean Sales Price')
plt.title('Mean Sales Price by Year (2000-2024)')
plt.xticks(range(2000, 2025), rotation=45)
plt.grid(axis='y')
plt.show()
```



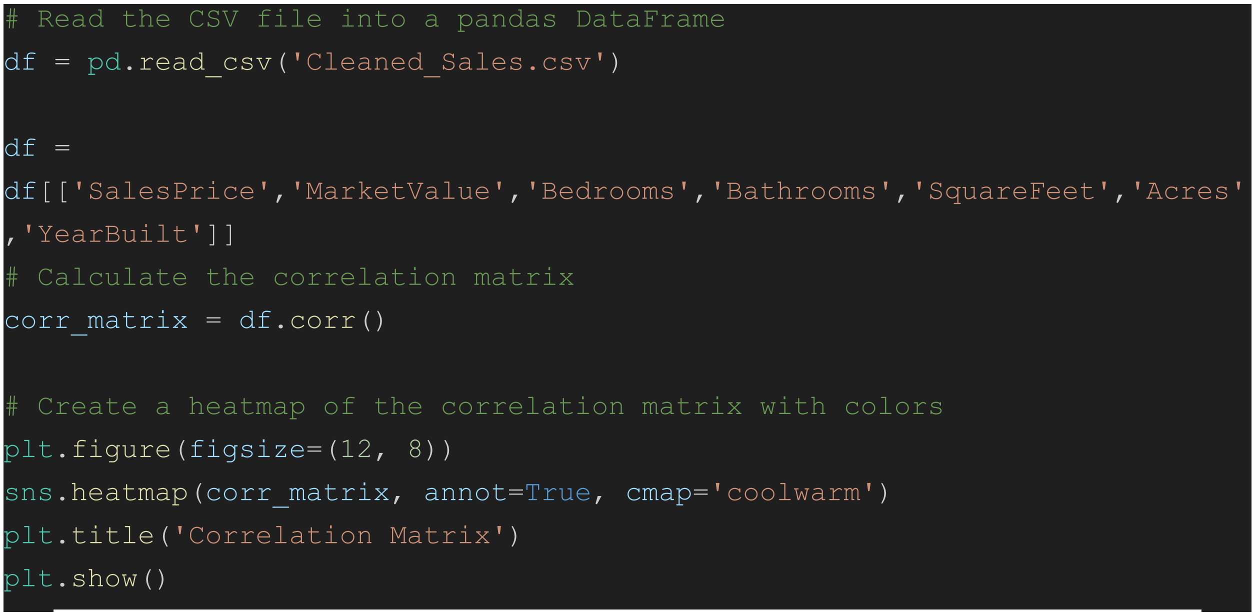
Since we just examined the mean sales price, why don't we plot every sales price recorded over the past two decades?



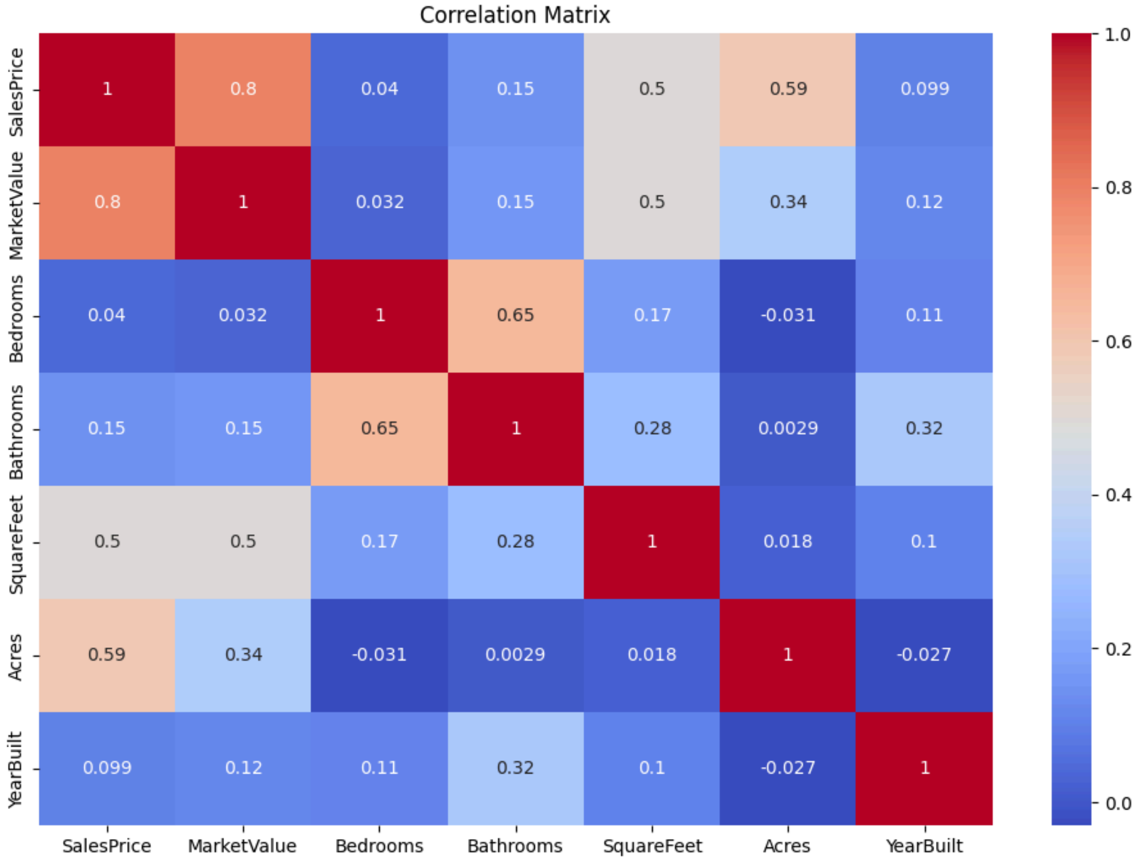
```
# Get the csv file
df = pd.read_csv('Cleaned_Sales.csv')

# Use loc to make our dataframe focused on date and price
df = df.loc[:, ['SalesDate', 'SalesPrice']]

# Fix the formatting for date of sale
df['SalesDate'] = pd.to_datetime(df['SalesDate'], format='%Y/%m/%d',
errors='coerce')

# Drop rows with NaT (invalid dates)
df = df.dropna(subset=['SalesDate'])

# Change range of years to 2004 to 2024
df = df[(df['SalesDate'].dt.year >= 2004) & (df['SalesDate'].dt.year <=
2024)]

df.index = df['SalesDate']
del df['SalesDate']
sns.lineplot(df)
```



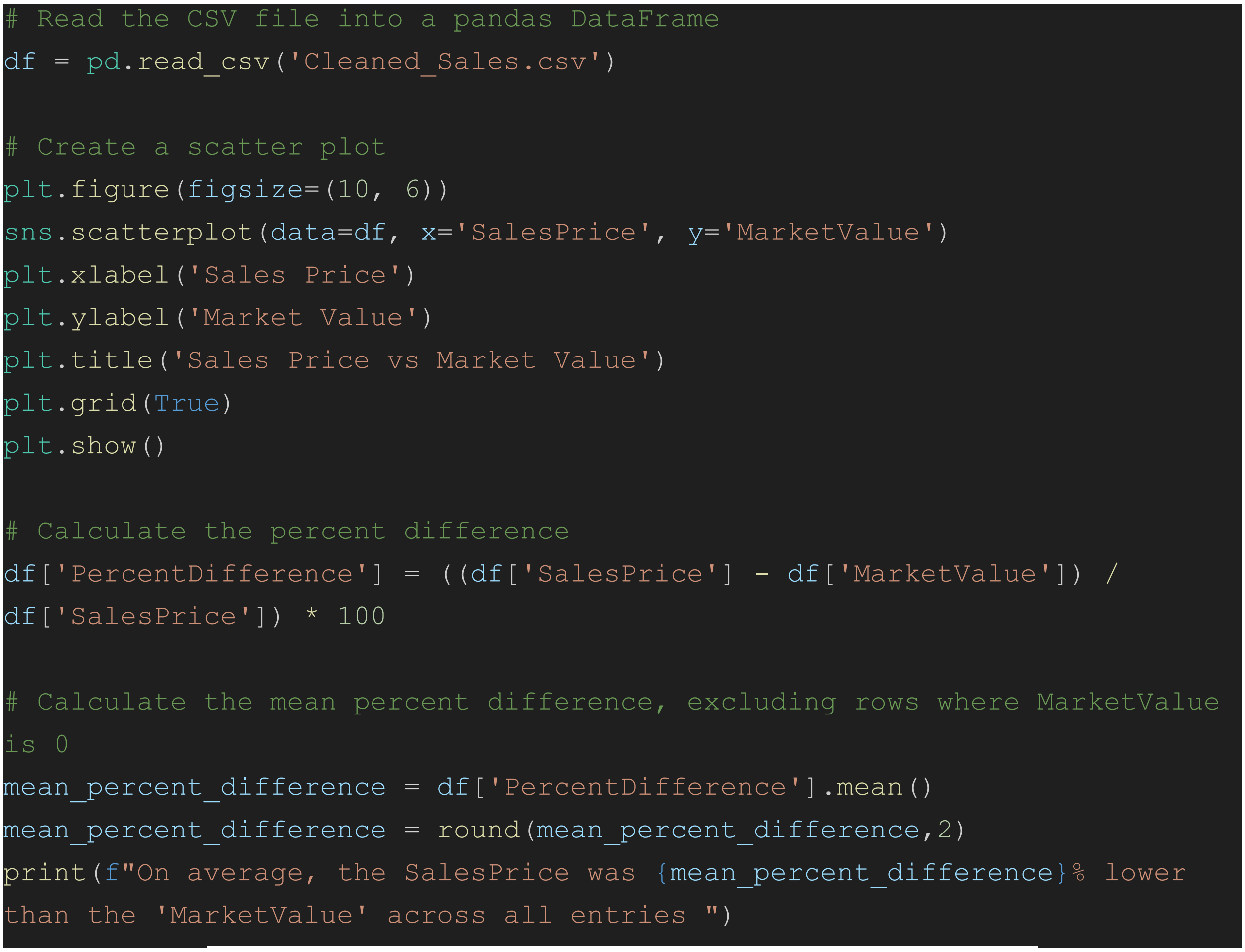
Now, Let’s plot a correlation matrix to explore the relationships between variables in our dataset. This will display the correlation coefficients between pairs of variables, indicating the strength and direction of their linear relationships.



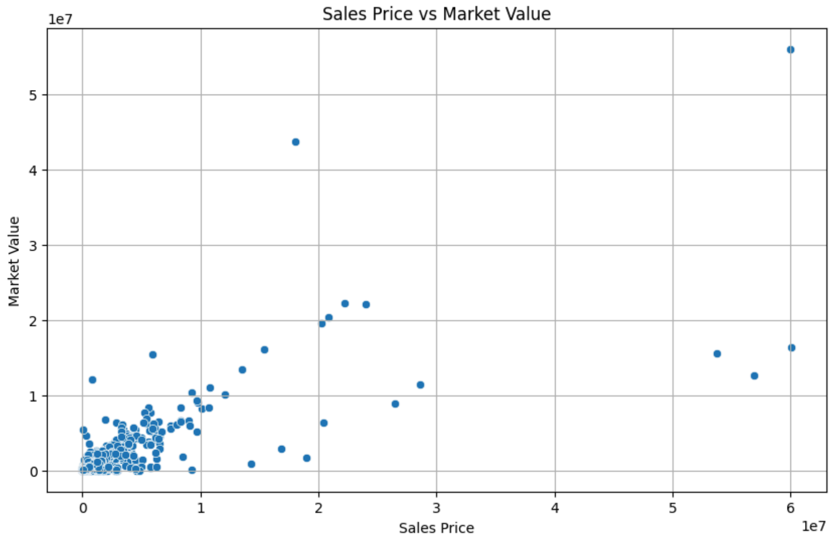
```
# Read the CSV file into a pandas DataFrame
df = pd.read_csv('Cleaned_Sales.csv')

df =
df[['SalesPrice','MarketValue','Bedrooms','Bathrooms','SquareFeet','Acres'
,'YearBuilt']]
# Calculate the correlation matrix
corr_matrix = df.corr()

# Create a heatmap of the correlation matrix with colors
plt.figure(figsize=(12, 8))
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm')
plt.title('Correlation Matrix')
plt.show()
```



Next, I plotted a scatter plot which not only showed us the correlation between these two variables, but it also helped find that on average, the sales price was 8.32% lower than the market value across all entries.



```
# Read the CSV file into a pandas DataFrame
df = pd.read_csv('Cleaned_Sales.csv')

# Create a scatter plot
plt.figure(figsize=(10, 6))
sns.scatterplot(data=df, x='SalesPrice', y='MarketValue')
plt.xlabel('Sales Price')
plt.ylabel('Market Value')
plt.title('Sales Price vs Market Value')
plt.grid(True)
plt.show()

# Calculate the percent difference
df['PercentDifference'] = ((df['SalesPrice'] - df['MarketValue']) /
df['SalesPrice']) * 100

# Calculate the mean percent difference, excluding rows where MarketValue
is 0
mean_percent_difference = df['PercentDifference'].mean()
mean_percent_difference = round(mean_percent_difference,2)
print(f"On average, the SalesPrice was {mean_percent_difference}% lower
than the 'MarketValue' across all entries ")
```



Finally, let's take a deeper look at our closest towns and see how they compare

```
df = pd.read_csv('Cleaned_Sales.csv')

# Find all uniques possibilities of SiteCity
df['SiteCity'].unique()

# Isolate sales that are in Ashland, Talent, Phoenix and Medford
df = df[(df['SiteCity'] == 'TALENT') | (df['SiteCity'] == 'ASHLAND') |
        (df['SiteCity'] == 'PHOENIX') | (df['SiteCity'] == 'MEDFORD')]

# Group the data by city and calculate the average for each column,
rounding to 2 decimal places
avg_data = df.groupby('SiteCity').agg({'SalesPrice': lambda x:
round(x.mean(), 2),
                                       'Bedrooms': lambda x:
round(x.mean(), 2),
                                       'Bathrooms': lambda x:
round(x.mean(), 2),
                                       'YearBuilt': lambda x:
round(x.mean(), 2)}).reset_index()
# Rename columns for clarity
avg_data.columns = ['City', 'Avg Sales Price $', 'Avg Bedrooms', 'Avg
Bathrooms', 'Avg Year Built']

# Display the table
print(avg_data)
```

City	Avg Sales Price $	Avg Bedrooms	Avg Bathrooms	Avg Year Built
ASHLAND	475009.19	2.83	2.09	1975.62
MEDFORD	346976.86	3.01	1.97	1979.69
PHOENIX	292293.68	2.98	1.92	1984.47
TALENT	298256.22	2.93	1.96	1991.15

Conclusion:

Making this project was fun and an informative way to learn more about statistical research using python. This analysis provided valuable insights into market trends over the past two decades. These findings would be valuable to real estate professionals, policymakers, and investors who are interested in the potential opportunities in Oregon's real estate market.

Resources:

https://gis.jacksoncountyor.gov/datasets/JCGIS::residential-values/about

https://www.geeksforgeeks.org/working-csv-files-python/

https://medium.com/@filipesampaiocampos/real-estate-data-analysis-and-modeling-using-python-184252d60189

https://pandas.pydata.org/docs/user_guide/cookbook.html