Report

US Real Estate Analysis

By:
Sina Tijani.

US Real Estate Data Cleaning

Libraries:

1. Pandas
2. Numpy
3. Seaborn
4. Scipy
5. Matplotlib

Dataframes

I created 5 main data frames (df, df2, df3, df4, and df5) throughout our data cleaning process. I used inplace=True to save changes to some data frames, but I mostly resorted to creating new ones as that allowed me to easily make corrections without breaking my code and having to restart the kernel to run all the code blocks again.

1. I imported the data set by inserting the file path into r''. I named it df.

```
# Importing the data set

df = pd.read_csv(r'/Users/sinatijani/Library/Mobile Documents/com~apple~CloudDocs/Document/Coventry University (Class)/Business Statistics/Course Work 1/CW1(2324SepJan)_USRealEstateData.csv')
df
```

✓ 0.5s

I checked for 1:1 duplicated entries and sorted the result by full address using the sort_values method. By 1:1, I mean the data across every column was equal.

```
df[df.duplicated()].sort_values(by='full_address')
```

✓ 0.5s

To know the percentage of perfect duplicates out of the whole data set, I used len() to count the number of duplicates and divided it by the len of the data frame (df).

```
duplicates_percent = len(df[df.duplicated()]) / len(df) * 100
print(f"The dataset has {duplicates_percent:.2f}% perfect duplicate entries.")
```

✓ 0.2s

```
The dataset has 86.73% perfect duplicate entries.
```

86.73% of the data set were perfect duplicates. By checking the data frame, I realised the street, city, state, and ZIP Code were all also included in the full address column. I decided I would only use city and state and therefore I was going to drop the street and ZIP Code column.

4. In one line of code, I used .drop_duplicates to remove the duplicates and passed street and City through .drop to eliminate the two columns. Then, I used .reset_index to reset the index while dropping the old index column. Vin named the new data frame df2.

```
df2 = df.drop_duplicates().drop(columns=['street', 'zip_code']).reset_index(drop=True)
df2
```
✓ 0.2s

5. I checked for missing entries across all columns using .isnull and .sum.

```
df2.isnull().sum()
```
✓ 0.0s

6. There were still more duplicates in the data frame. I checked for duplicates on the full_address column and sorted using the column, which found 772 more duplicates.

```
next_duplicates = df2[df2.full_address.duplicated()].sort_values(by='full_address')
next_duplicates
```
✓ 0.0s

7. My inspection showed it is tricky to remove all these duplicates correctly. So I created a columns_sorter using full address, price, house size, and sold date to sort while arranging the result in descending order. I reset the index and created a data frame df3. Df3 arrangement allowed me to keep the first duplicate and delete the rest while allowing me to retain as much data as possible across the columns. I named this data frame df4.

```
columns_sorter = ['full_address', 'price', 'house_size', 'sold_date']
columns_sorter
```
✓ 0.0s

```
df4 = df3.drop_duplicates(subset='full_address', keep='first').reset_index(drop=True)
df4
```
✓ 0.0s

8. I confirmed there were no duplicates on the full address column and no null entries.

```
df4.full_address.duplicated().any()
```
✓ 0.0s

```
df4.isnull().sum()
```
✓ 0.0s

9. I checked for descriptive statistics on df4, rounding the result to one decimal place to make it easier to read. There was an old chill, and there was a huge outlier in the price.

```
df4.describe().round(1)
```
✓ 0.0s

10. I plotted df4 on a scatter plot, box plot, and histogram to examine the outliers.

```
# Scatter Plot
plt.figure(figsize=(10, 6))
sns.scatterplot(x='price', y='bath', data=df4)
plt.title('Scatter Plot')
plt.xlabel('Price')
plt.ylabel('Number of Bathrooms')
plt.show()
```
✓ 0.1s

```
# Box Plot
plt.figure(figsize=(10, 6))
sns.boxplot(x='price', data=df4)
plt.title('Box Plot')
plt.xlabel('Price')
plt.show()
```
✓ 0.1s

```
# Plotting histogram
plt.figure(figsize=(15, 15))
sns.histplot(df4['price'], kde=True)
plt.title('Histogram')
plt.xlabel('Price')
plt.ylabel('Frequency')
plt.show()
```
✓ 11.8s

11. I sorted the data frame by price and realised they were entries of 0, 1, and other very low prices, which were also outliers.

```
df4.sort_values(by='price').head(10)
```
✓ 0.0s

12. I used .quantile to calculate the interquartile range. I started with 1.5 for both lower and upper bounds. However, it was capturing about 18% of my data as outliers while completely missing the extremely low prices. So I adjusted the multiplier several times and agreed to use 0.45 for my lower bound and 3.7 for my upper bound. This ensures I am able to capture rows with very low prices as well as those with extremely high prices.

- The lower bound price of IQR outliers was 3355.0.
- The upper bound of IQR outliers was 2939370.0.
- The number of IQR outliers was 3967.
- The outliers represent 4.92% of the df4 data frame.

```
Q1 = df4.price.quantile(0.25)
Q3 = df4.price.quantile(0.75)

IQR = Q3 - Q1
```
✓ 0.0s

```
lower_bound = Q1 - 0.45 * IQR
upper_bound = Q3 + 3.7 * IQR
print(lower_bound)
print(upper_bound)
```
✓ 0.0s

```
3355.0
2939370.0
```

```
print(f"The minimum price of IQR outliers is {lower_bound}.")
print(f"The maximum price of IQR outliers is {upper_bound}.")
print(f"The number of IQR outliers are {iqr_outliers.price.count()}.")
```
✓ 0.0s

```
The minimum price of IQR outliers is 3355.0.
The maximum price of IQR outliers is 2939370.0.
The number of IQR outliers are 3967.
```

13. I also used scipy.stats to calculate for outliers so I could decide what to use to drop the outliers.

I ran stats.zscore on the data frame df4, starting with a threshold of 2, and adjusted until I settled on 0.24. This was the only multiplier that would reasonably capture the lowest prices while also capturing the extreme ones. But the outlier percentage was 10.23% of df4, which was too much for me. So I used the IQR, which resulted in only 4.92% of the data frame for outliers.

```
z_scores = stats.zscore(df4.price)
z_scores
```
✓ 0.0s

```
threshold = 0.24
```
✓ 0.0s

```
z_scores_outliers = df4[abs(z_scores) > threshold]
z_scores_outliers.sort_values(by='price', ascending=False)
```
✓ 0.0s

```
len(z_scores_outliers.index)/len(df4.index)
```
✓ 0.0s

```
0.10234618295512352
```

```
iqr_outliers_perc = len(iqr_outliers.index)/len(df4.index)
print(f"The outlier's represent {iqr_outliers_perc * 100 :.2f}% of the cleaned dataframe")
```

```
The outlier's represent 4.92% of the cleaned dataframe
```

14. I dropped the outliers from df4 and named it df5 - my last main data frame. I checked for null entries on df5 and now I had comparatively fewer nulls.

```
df5 = df4.drop(iqr_outliers.index).reset_index(drop=True)
df5
```
✓ 0.0s

```
df5.isnull().sum()
```
✓ 0.0s

15. I also checked for descriptive statistics.

```
df5.describe().round(1)
```
✓ 0.0s

16. From the summaries, I ran 4 lines of code using .fillna to fill the null entries in bed, bath, acre_lot, and house_size with their median values. I saved each using .inplace=True.

```
df5.bed.fillna(df5.bed.median(), inplace=True)
df5.bath.fillna(df5.bath.median(), inplace=True)
df5.acre_lot.fillna(df5.acre_lot.median(), inplace=True)
df5.house_size.fillna(df5.house_size.median(), inplace=True)
```
✓ 0.0s

17. At this point, I only had 21 null entries for City and 39,536 null entries for sold_date. I decided not to fill them, nor drop the sold_date column as the null entries showed the properties have not been sold, which is a good statistic.

```
df5.isnull().sum()
```
✓ 0.0s

18. I ran .unique on the bed column and saw they had no decimals. So I used .astype to change the bed datatype from float64 to int64. Then I confirmed the change using .info().

```
df5.bed.unique()
```
✓ 0.0s

```
df5.bed = df5.bed.astype(dtype=int)
df5.sold_date = pd.to_datetime(df5.sold_date)
```
✓ 0.0s

```
df5.info()
```
✓ 0.0s

19. I checked again for descriptive statistics on df5, the detail looked better. However, acre_lot and house_size still had huge outliers. But I left it because I realised it will remove too much data again from the data set. Besides, I was not going to perform a lot of analysis using them.

```
df5.describe().round(1)
```
✓ 0.0s

20. I checked for correlation on df5 using .corr. And then I visualised the correlation on a heat map.

```
sns.heatmap(df3.corr(numeric_only=True), annot=True)

plt.show()
```
✓ 0.0s

```
sns.heatmap(df3.corr(numeric_only=True), annot=True)

plt.show()
```
✓ 0.0s

21. My data was cleaned at this point.

22. I ran .to_excel on my final dataframe df5 and exported the clean dataset to Excel.

```
df5.to_excel('CW1 Group 2 Business Statistics (USRealEstateData) cleaned.xlsx')
```

Descriptive Statistics

1. Mean
2. Median
3. Mode
4. Standard Deviation
5. Quartile
6. Interquartile Range

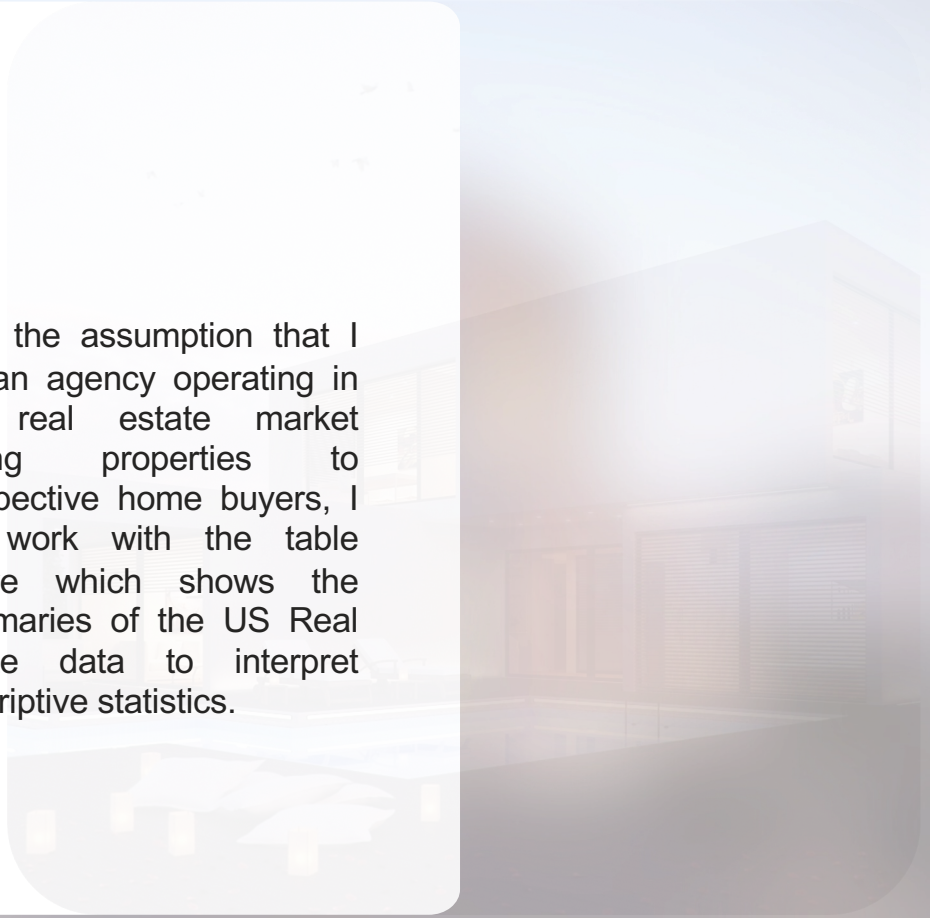
In simple terms...

Descriptive statistics refers to summaries which show characteristics of a given dataset.

When the summaries show the likelihood of the data being grouped among themselves like the average (mean), middle number (median), or frequent numbers (mode), it is termed as a measure of central tendency.

When the summaries reveal how the data is dispersed or scattered among themselves like standard deviation, variance, and quartiles, then it is termed a measure of variability.

With the assumption that I am an agency operating in the real estate market selling properties to prospective home buyers, I will work with the table above which shows the summaries of the US Real estate data to interpret descriptive statistics.



1. Mean

	price	bed	bath	acre_lot	house_size	sold_date	year
count	76632.0	76632.0	76632.0	76632.0	76632.0	37096	37096.0
mean	599656.0	3.0	2.0	15.0	1904.0	2009-05-10 18:02:08.100064768	2009.0
min	3900.0	1.0	1.0	0.0	100.0	1901-01-01 00:00:00	1901.0
25%	249999.0	2.0	2.0	0.0	1382.0	2003-04-28 00:00:00	2003.0
50%	447000.0	3.0	2.0	0.0	1638.0	2010-08-31 00:00:00	2010.0
75%	750000.0	4.0	3.0	1.0	1969.0	2017-10-27 00:00:00	2017.0
max	2937500.0	49.0	36.0	100000.0	400149.0	2022-10-31 00:00:00	2022.0
std	531688.0	2.0	1.0	881.0	2144.0	NaN	10.0

This is also known as average. From the table, there are 76,632 properties in the US real estate market shown as “count”. When the price of each property is added together, we get a total of $45,952,865,713. When this total is divided by 76,632, we get $599,656 which is referred to as the mean.

Applying the same concept to the rest, when a customer makes an enquiry, but they are not sure what they want, we can quickly tell them, “We have various properties to serve your needs. With $599,656, we can get you a house with three bedrooms and two bathrooms for you. The house size will be about 1,904 square feet covering about 15 acres.” The customer may not want a 3-bedroom house, but this quickly gives them something to think about and to get the conversation going. Hopefully, we make a sale!

2. Median

	price	bed	bath	acre_lot	house_size	sold_date	year
count	76632.0	76632.0	76632.0	76632.0	76632.0	37096	37096.0
mean	599656.0	3.0	2.0	15.0	1904.0	2009-05-10 18:02:08.100064768	2009.0
min	3900.0	1.0	1.0	0.0	100.0	1901-01-01 00:00:00	1901.0
25%	249999.0	2.0	2.0	0.0	1382.0	2003-04-28 00:00:00	2003.0
50%	447000.0	3.0	2.0	0.0	1638.0	2010-08-31 00:00:00	2010.0
75%	750000.0	4.0	3.0	1.0	1969.0	2017-10-27 00:00:00	2017.0
max	2937500.0	49.0	36.0	100000.0	400149.0	2022-10-31 00:00:00	2022.0
std	531688.0	2.0	1.0	881.0	2144.0	NaN	10.0

This is also known as the middle value. When all the properties are listed in ascending order using their prices, the value of the property in the middle is referred to as the median. In simple terms, “the property in the middle”. This is shown as 50% on the table representing $447,000.

Because the mean is gotten by dividing the total value by the count, it can be influenced by extreme prices. Also, as a measure of central tendency, the median is more robust to this. It tells us the value of the property in the middle of the data set. With this, we can tell a customer that “half of our properties are below $447,000. Based on the location, you could have a 3-bedroom and a 2-bathroom house sitting on 1638 square feet.”

3. Mode

	price	bed	bath	acre_lot	house_size	sold_date	year
count	76632.0	76632.0	76632.0	76632.0	76632.0	37096	37096.0
mean	599656.0	3.0	2.0	15.0	1904.0	2009-05-10 18:02:08.100064768	2009.0
min	3900.0	1.0	1.0	0.0	100.0	1901-01-01 00:00:00	1901.0
25%	249999.0	2.0	2.0	0.0	1382.0	2003-04-28 00:00:00	2003.0
50%	447000.0	3.0	2.0	0.0	1638.0	2010-08-31 00:00:00	2010.0
75%	750000.0	4.0	3.0	1.0	1969.0	2017-10-27 00:00:00	2017.0
max	2937500.0	49.0	36.0	100000.0	400149.0	2022-10-31 00:00:00	2022.0
std	531688.0	2.0	1.0	881.0	2144.0	NaN	10.0

In our context, this is known as the most occurring property sold. The mode was not given in the earlier table. So, we have calculated it below.

This refers to the highest number of times, we have sold a particular property. Or simply, the most frequent property sold.

About our business, we can glean from this information and tell a prospective customer with confidence that “3-bedroom houses are higher in demand than a 4 or 2-bedroom house. Usually, they come with 2 bathrooms sitting on an acre size of 0.34. You can get a similar property with a size of 1638 square feet. We have sold many of them for $399,000. We believe you would like this.”

4. Standard Deviation

This is how much the data deviates (is spread out) from the mean. A bigger standard deviation means there are huge distances between the values.

From the estate market, we record a standard deviation (Sd) value of $531,688. Where the mean is $599,656, the Sd shows property prices vary significantly from the mean value.

```
print(f"The standard deviation of price is ${df5.price.std().__round__(2)}")
print(f"The standard deviation of bed is {df5.bed.std().__round__(2)}")
print(f"The standard deviation of bath is {df5.bath.std().__round__(2)}")
print(f"The standard deviation of acre lot is {df5.acre_lot.std().__round__(2)}")
print(f"The standard deviation of house size is {df5.house_size.std().__round__(2)}")
```

✓ 0.0s

```
The standard deviation of price is $531688.11
The standard deviation of bed is 1.55
The standard deviation of bath is 1.2
The standard deviation of acre lot is 881.01
The standard deviation of house size is 2143.9
```

Therefore, customers are likely going to pay much higher or lower for property than the mean. These factors may from higher sales in highly residential area cost higher or a lot of distressed apartments with very low prices. With number of bedrooms mean of 3 and Sd of 1.55, this shows customers are likely going to get fewer bedrooms, or they will have to pay higher for the property since the chances of getting more than 3-bedrooms property is high. The same goes for bathroom with a mean of 2 and a Sd of 1.2.

5. Quartile

	price	bed	bath	acre_lot	house_size	sold_date	year
count	76632.0	76632.0	76632.0	76632.0	76632.0	37096	37096.0
mean	599656.0	3.0	2.0	15.0	1904.0	2009-05-10 18:02:08.100064768	2009.0
min	3900.0	1.0	1.0	0.0	100.0	1901-01-01 00:00:00	1901.0
25%	249999.0	2.0	2.0	0.0	1382.0	2003-04-28 00:00:00	2003.0
50%	447000.0	3.0	2.0	0.0	1638.0	2010-08-31 00:00:00	2010.0
75%	750000.0	4.0	3.0	1.0	1969.0	2017-10-27 00:00:00	2017.0
max	2937500.0	49.0	36.0	100000.0	400149.0	2022-10-31 00:00:00	2022.0
std	531688.0	2.0	1.0	881.0	2144.0	NaN	10.0

As I can see, the word “quarter” from the name, quartile divides the data into 4 equal parts. In the first descriptive statistics summary table, 25% is the first quartile. This is the lower boundary.

50% is the second quartile which is also known as the 50th percentile. And 75% is the third quartile which is also the upper boundary. In terms of number of bedrooms, this tells us that 75% of the properties have 3 or less bedrooms as the 75% quartile are 4 bedrooms. Also, those properties will likely have 2 or fewer bathrooms in them.

6. Interquartile Range (IQR) -a

A business can also use what Interquartile range (IQR) to remove outliers from their data. Outliers, simply are data or values that the company deems as too low or too high to use in their estimations. As real estate agents, we try to create reports to guide how to operate the agency.

```
Q1 = df4.price.quantile(0.25)
Q3 = df4.price.quantile(0.75)

IQR = Q3 - Q1
```

✓ 0.0s

```
lower_bound = Q1 - 0.45 * IQR
upper_bound = Q3 + 3.7 * IQR
print(lower_bound)
print(upper_bound)
```

✓ 0.0s

```
3355.0
2939370.0
```

Many times, we try to generalize and speculate how the market is or will look like in the future. For this, we can collect housing data from different websites to learn from it. From our dataset, we find that some houses in the US were sold at $0, $1, $175, $385, and $445 in 2006, 2007, and other years.

6. Interquartile Range (IQR) -b

With my business knowledge, I know these are errors. Also, some very expensive properties were sold as high as $875,000,000, $120,000, and others. But they are not a lot. They are also not errors; if I add them to my calculations, they will affect my averages and influence other analyses wrongly.

```
Q1 = df4.price.quantile(0.25)
Q3 = df4.price.quantile(0.75)

IQR = Q3 - Q1
```

✓ 0.0s

```
lower_bound = Q1 - 0.45 * IQR
upper_bound = Q3 + 3.7 * IQR
print(lower_bound)
print(upper_bound)
```

✓ 0.0s

```
3355.0
2939370.0
```

To ensure I can get a true picture of the market in general, I decided to create a price ceiling that, only properties sold as low as $3,355 and as high as $2,939,370 would be included in my calculations. Then I go ahead and remove those data. This is an example of how IQR is used in the business world.

Real Estate Report

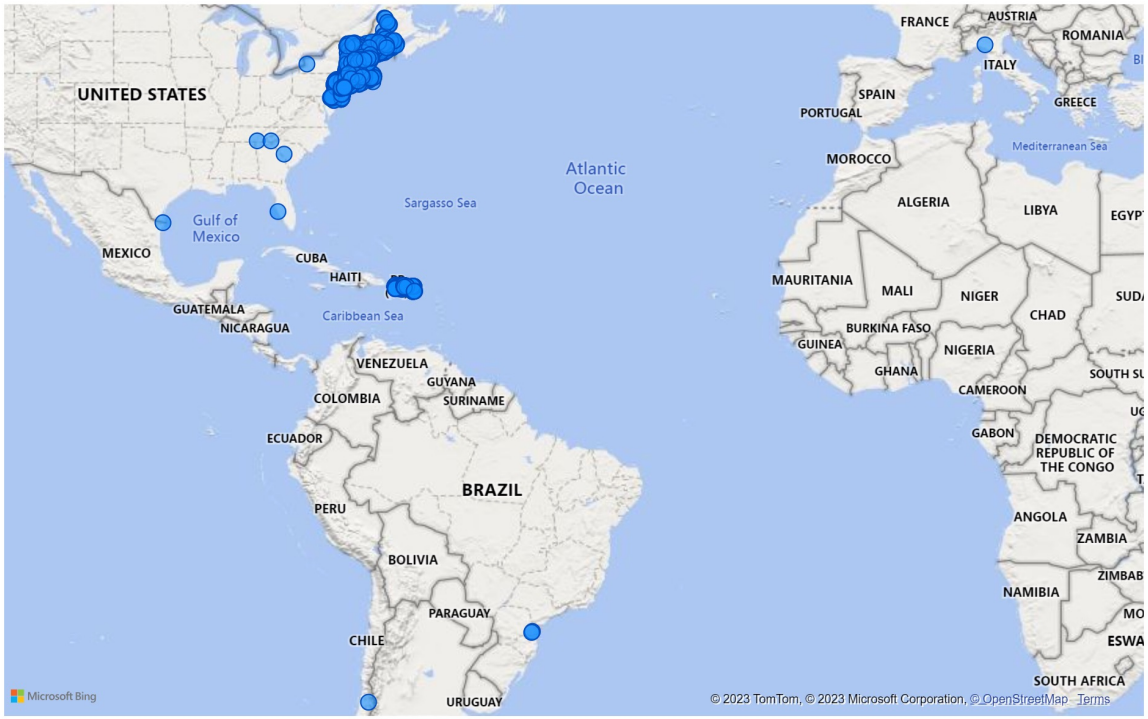
In 67 years from 1901 to 2022, our data showed

- ❖ 76,000 properties,
- ❖ 247,000 beds,
- ❖ 179,000 bathrooms in
- ❖ 18 states and
- ❖ 2,062 cities.
- ❖ In area coverage, the properties totalled 1.14 million acres.

While United States is the primary location, few properties were in South America and Europe.

$45.95bn Total Valuation	**67** Number of Years
76.63K Number of Properties	**1.14M** Total Acres
247K Number of Beds	**18** Number of States
179K Number of Baths	**2062** Number of Cities

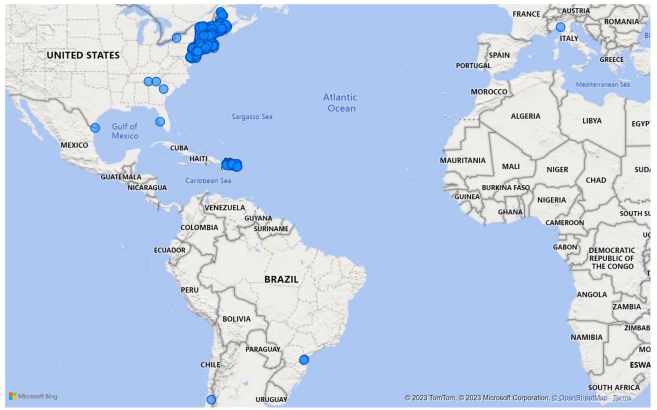
Locations



Real Estate Report Dashboard

$45.95bn Total Valuation	67 Number of Years
76.63K Number of Properties	1.14M Total Acres
247K Number of Beds	18 Number of States
179K Number of Baths	2062 Number of Cities

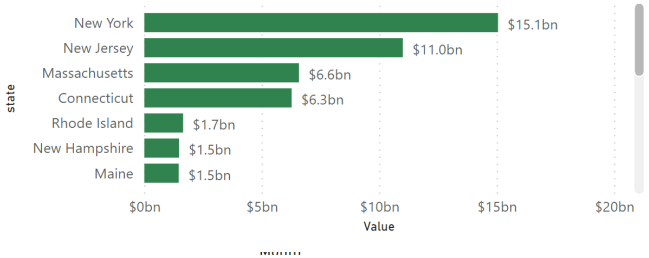
Locations



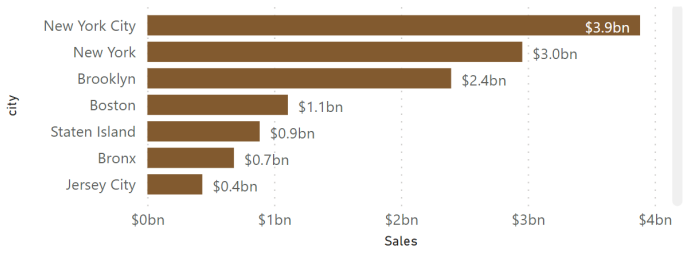
Year	Number of Sales
2022	561
2021	2537
2020	1684
2019	2034
2018	2094
2017	1972
2016	1764
2015	1587
2014	1216
2013	1190
2012	902
Total	**76632**

Month	Number of Sales
January	254
February	187
March	90
April	14
May	11
June	1
July	1
September	1
October	2
Total	**561**

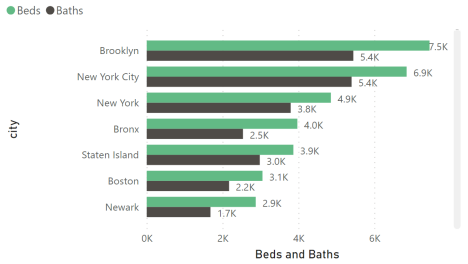
Value by state



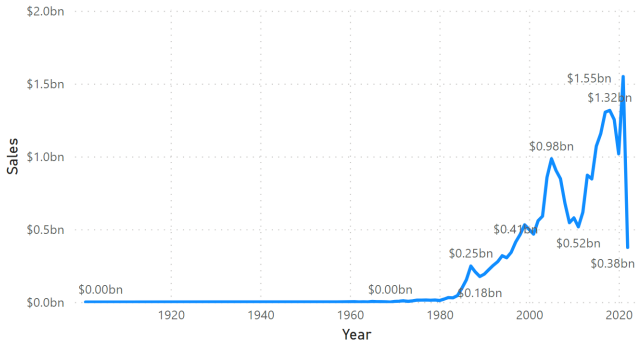
Sales by city



Beds and Baths by city



Sales by Year



Sales by Month

Sales by Day

Limitations

1

Cleaning

This report is limited to my cleaned data, which I carefully cleaned to retain only useful data at best.

2

We Dropped

- 86.7% perfect duplicates of dataset. I saved as “df”
- 0.12% other duplicates
- 0.65% outliers based on my IQR calculations.

3

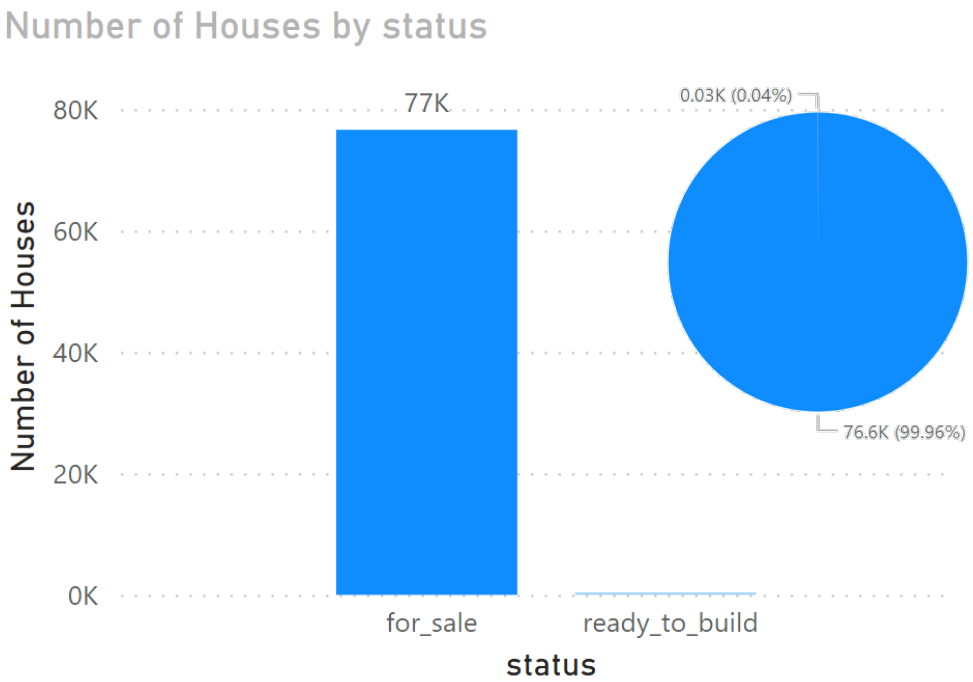
Cleaned Data

- My cleaned dataset represents 12.50% of the original dataset.

4

Missing Dates

39,536 properties sold having missing dates. While they were included in some relevant analyses, I dropped them from my time-series analysis.



House Status

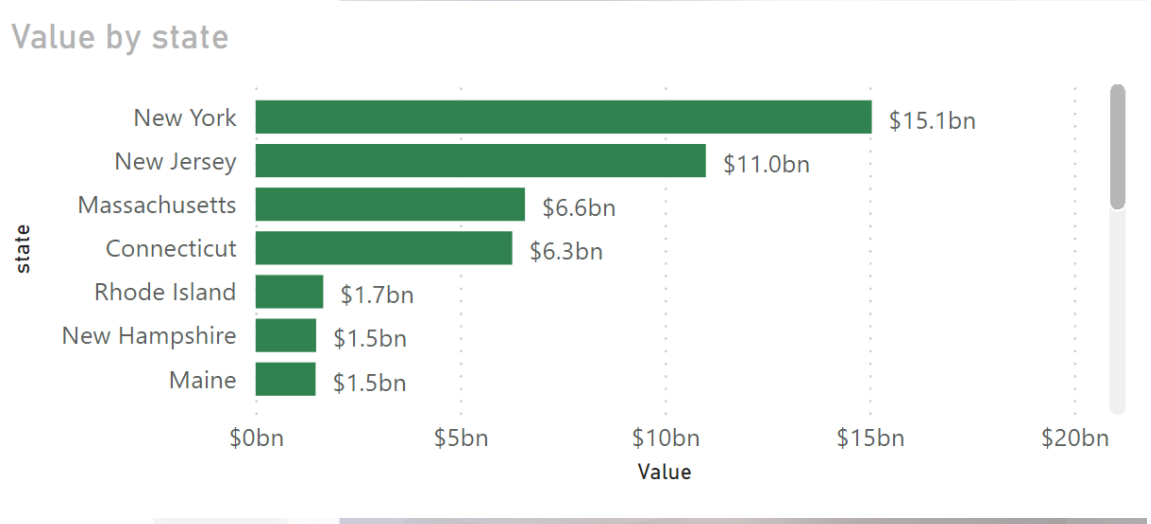
76,703 properties were shown as ready for sale representing 99.96%.

Only 29 were ready to be built.

Value by State

As seen on the image on the right, by State, New York had the highest value at over $15.1B. Followed by New Jersey at $11.0B and Massachusetts at $6.6B. Connecticut was the 4th most valuable state, followed by Rhode Island and New Hampshire. South Carolina had the lowest value at $18,950.× The State of New York recorded 32.77% of the property value.

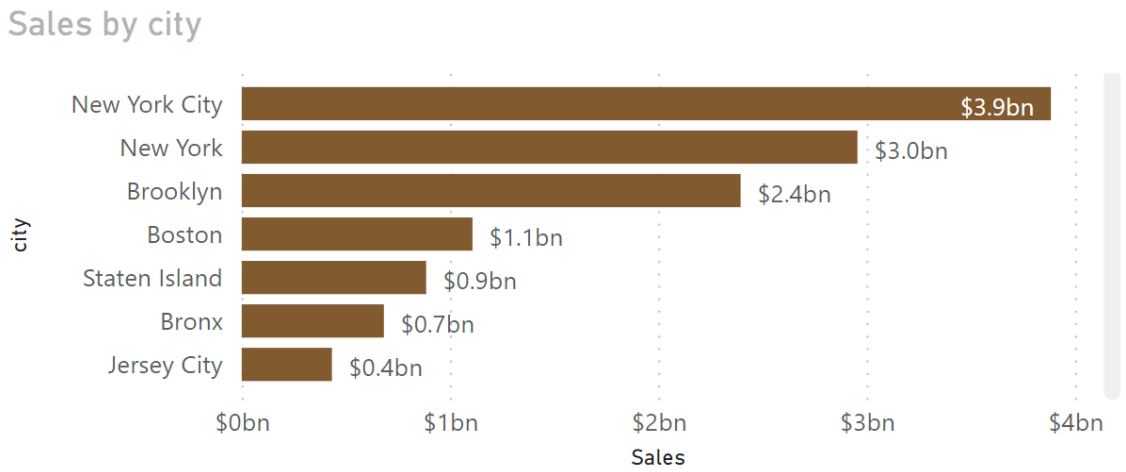
Properties ranged from $18,950 to $15,057,759 in state terms.



Value by City

From the previous slide, it makes sense to me that New York City in New York State had the most valuable property at $3.9bn, followed by Brooklyn and Boston. The valuation dropped into millions with Staten Island's properties starting at over $886m.

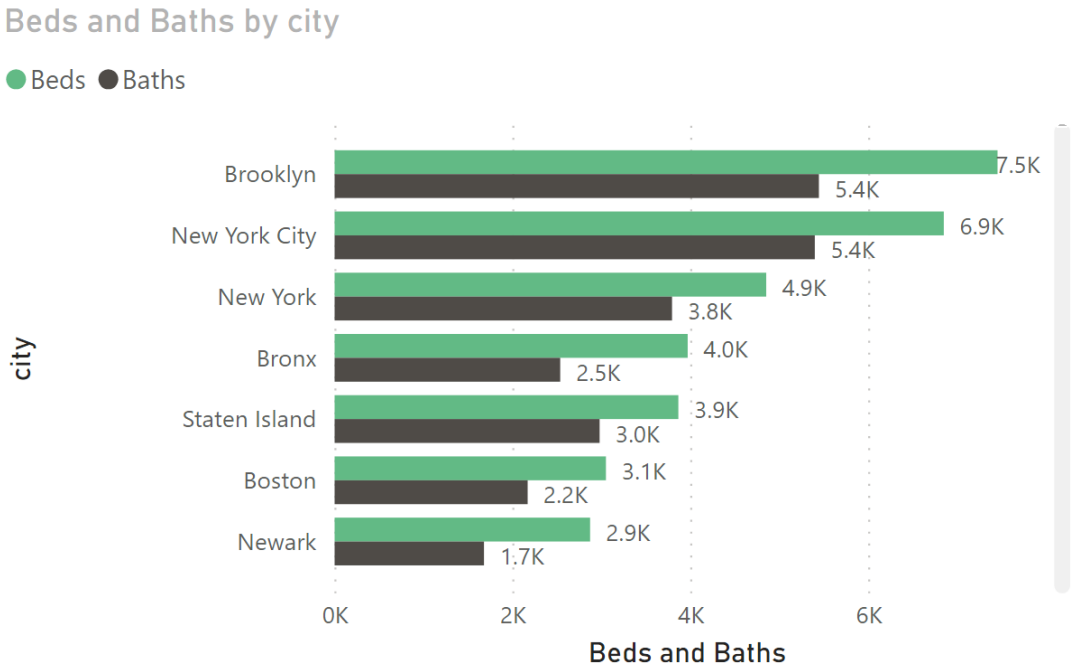
McCormick had the lowest property value at $18,950. New York City had 8.45% of the total valuation in all regions. Properties available to be sold ranged from $18,950 to $3,884,865,217.×

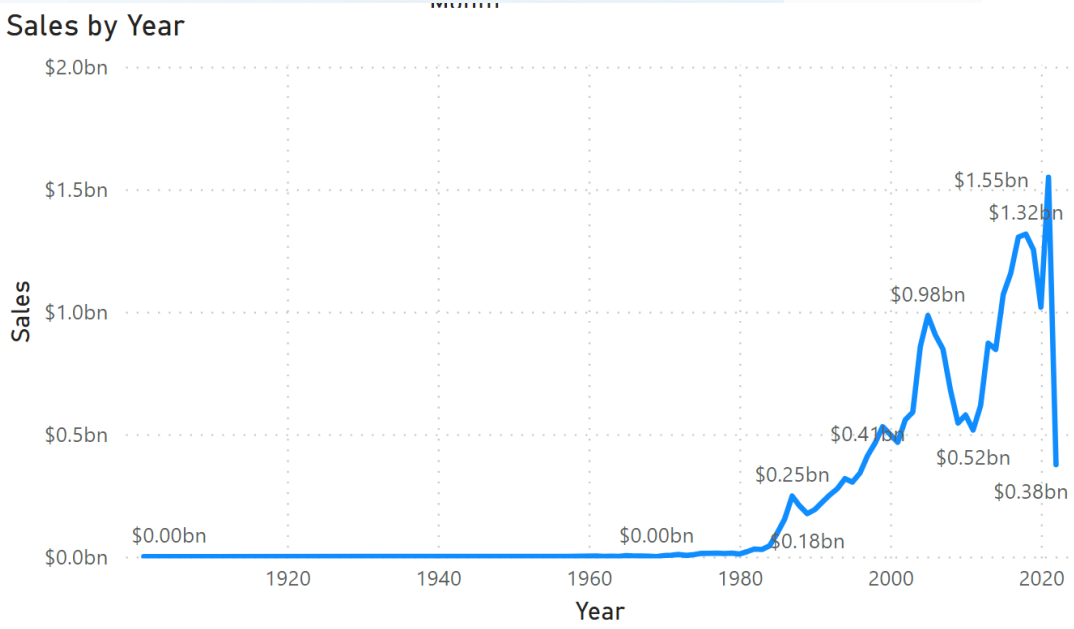


Beds and Baths

Though Brooklyn was not in the top 10 most valuable cities, its properties have the highest number of beds and baths. Followed by New York City.

Brooklyn made up of 3.02% of Beds.×





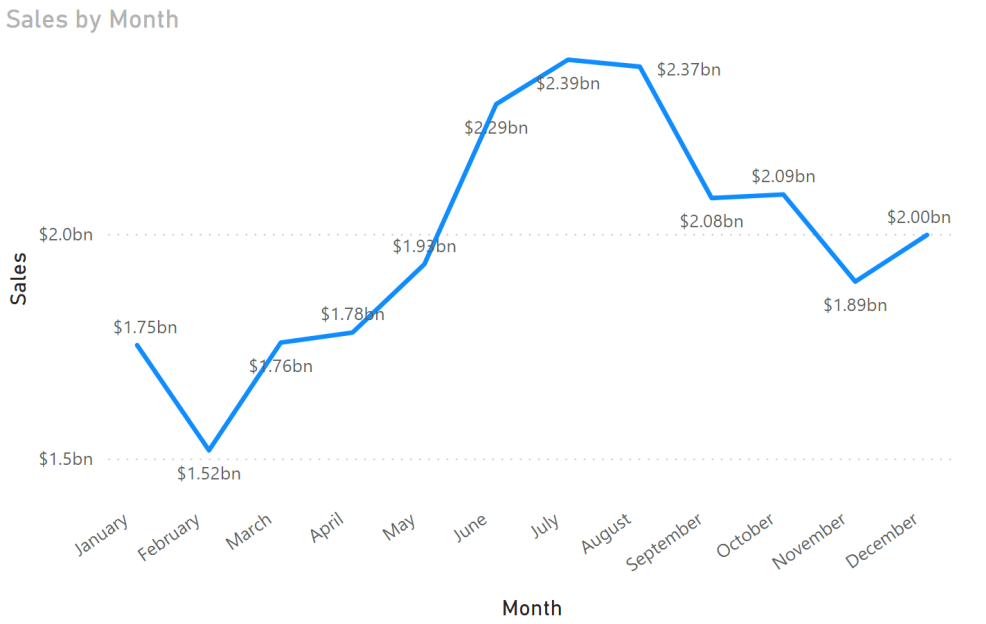
Yearly Sales

Properties have had an astronomical growth of 110,636.17% in sales value from 1901 to 2022.

The uptrend visibly starts in 1984 at $44m and continued to 2005 at $984m. Then it took a nose dive dropping to $516m in 2011.

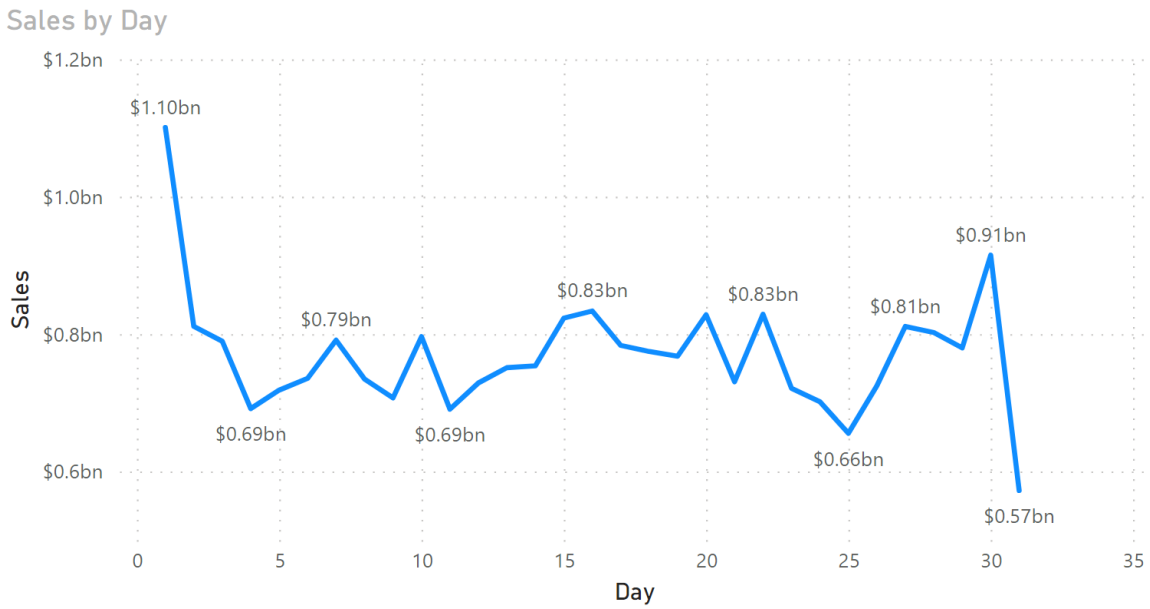
Sales skyrocketed strongly from 2011 through to recording $1.548B in 2021.

Monthly Sales



July records the highest sales at $2.38B, which is 57.28% more than February with the lowest sales at $1.52B.

Sales grow incrementally from February but plateau in June. They drop through November and make a comeback in December.



Daily Sales

Cumulatively, the 1st days of the months record the highest sales totalling $1.1B for me. It then nose dives to $691m on Day 4 for me.

From Day 4 through Day 22, it hovers around the $0.8B mark for me then drops to $655.48m on Day 25 for me.

Sales picked up for the next 3 days to $914.89m on Day 30 for me.

Conclusive Findings

It may be concluded from the time series that 2022 performed abysmally. But it is not so. The table exposes the dataset is limited. 2022 only had 561 records as against others which have over 2,500.

By slicing into 2022, I find no record for November and December while from June to October I had only 1 & 2 records.

Against all that, 2022 had a strong average of $669,154 beating 2021, 2022, 2019, and 38 other years.

Year	Number of Sales
2022	561
2021	2537
2020	1684
2019	2034
2018	2094
2017	1972
2016	1764
2015	1587
2014	1216
2013	1190
2012	902
Total	**76632**

Month	Number of Sales
January	254
February	187
March	90
April	14
May	11
June	1
July	1
September	1
October	2
Total	**561**

Year	Average Price
2022	$669,154.4153297683
2021	$610,447.4012613323
2020	$604,739.1152019002
2019	$616,256.7600786628
2018	$628,826.6852913084
2017	$661,617.8818458418
2016	$656,164.8429705215
2015	$674,643.8708254568
2014	$695,799.6694078947
2013	$732,178.4705882353
2012	$684,212.6607538803
Total	**$599,656.3539122038**

Thank You

References

Frost, J. (2023, July 7). Measures of variability: Range, interquartile range, variance, and standard deviation. Statistics By Jim. https://statisticsbyjim.com/basics/variability-range-interquartile-variance-standard-deviation/

Hayes, A. (n.d.). Descriptive statistics: Definition, Overview, types, example. Investopedia. https://www.investopedia.com/terms/d/descriptive_statistics.asp

Hoxha, V., Hoxha, D. & Hoxha, J. 2022, "Study of factors influencing apartment prices in Prishtina, Kosovo", International Journal of Housing Markets and Analysis, vol. 15, no. 5, pp. 1242-1258

James P. Curley & Tyler M. Milewski. (2020, August 1). PSY317L guidebook. 6 Descriptives. https://bookdown.org/curleyjp0/psy317l_guides5/descriptives.html

Krolage, C. (2022) The effect of real estate purchase subsidies on property prices - international tax and public finance, SpringerLink. Available at: https://link.springer.com/article/10.1007/s10797-022-09726-0#citeas (Accessed: 27 October 2023).

Marija Stanojcic mean, median, mode, variance & standard deviation. mean, median, mode, variance & standard deviation. (n.d.). https://www.csueastbay.edu/scaa/files/docs/student-handouts/marija-stanojcic-mean-median-mode-variance-standard-deviation.pdf

Mindrila, D., & Balentyne, P. (n.d.). Describing distributions with numbers - university of west georgia. https://www.westga.edu/academics/research/vrc/assets/docs/describing_distributions_with_numbers_notes.pdf

Mohd, T. et al. (1970) An overview of real estate modelling techniques for house price prediction, SpringerLink. Available at: https://link.springer.com/chapter/10.1007/978-981-15-3859-9_28 (Accessed: 27 October 2023).

Soetewey, A. (2020, January 22). Descriptive statistics in R. Stats and R. https://statsandr.com/blog/descriptive-statistics-in-r/

Wang, W.-C.; Chang, Y.-J.; Wang, H.-C. An Application of the Spatial Autocorrelation Method on the Change of Real Estate Prices in Taitung City. ISPRS Int. J. Geo-Inf. 2019

Y. Piao, A. Chen and Z. Shang, "Housing Price Prediction Based on CNN," 2019 9th International Conference on Information Science and Technology (ICIST), Hulunbuir, China, 2019, pp. 491-495, doi: 10.1109/ICIST.2019.8836731.

- Template from Canva