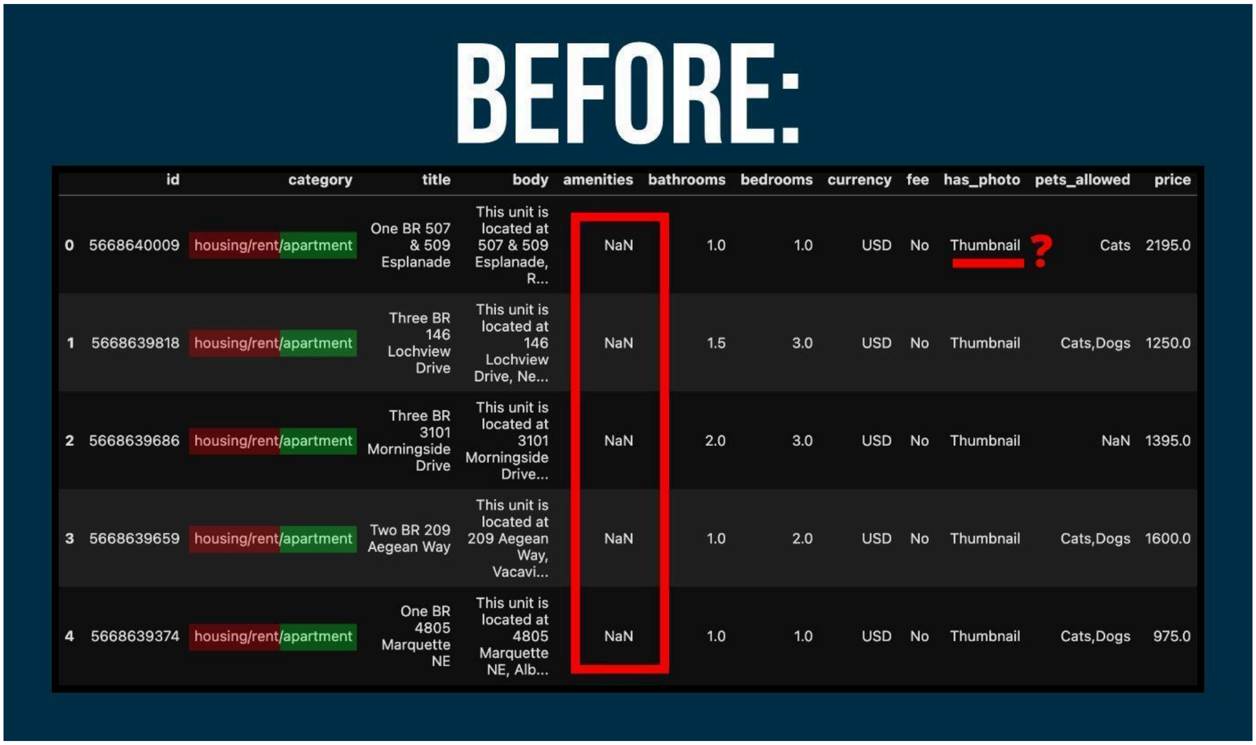
Intro

This analysis was done by Ali McCondichie, Kim Khue Nguyen, Leonardo Rodrigues Rodriguez, Seven George, and Tyler Beringer. We analyzed the Kaggle dataset named **Apartments for Rent Classified.** Our goal was to find the most important factors for customers when they are shopping for apartments.

Data Cleaning

The dataset contains 100,000 entries with information on things like price, bed/bath, and location. To better uncover insights about trends and patterns in rental pricing, we narrowed in on specific parts of the data.



BEFORE:

	id	category	title	body	amenities	bathrooms	bedrooms	currency	fee	has_photo	pets_allowed	price
0	5668640009	housing/rent/apartment	One BR 507 & 509 Esplanade	This unit is located at 507 & 509 Esplanade, R...	NaN	1.0	1.0	USD	No	Thumbnail ?	Cats	2195.0
1	5668639818	housing/rent/apartment	Three BR 146 Lochview Drive	This unit is located at 146 Lochview Drive, Ne...	NaN	1.5	3.0	USD	No	Thumbnail	Cats,Dogs	1250.0
2	5668639686	housing/rent/apartment	Three BR 3101 Morningside Drive	This unit is located at 3101 Morningside Drive...	NaN	2.0	3.0	USD	No	Thumbnail	NaN	1395.0
3	5668639659	housing/rent/apartment	Two BR 209 Aegean Way	This unit is located at 209 Aegean Way, Vacavi...	NaN	1.0	2.0	USD	No	Thumbnail	Cats,Dogs	1600.0
4	5668639374	housing/rent/apartment	One BR 4805 Marquette NE	This unit is located at 4805 Marquette NE, Alb...	NaN	1.0	1.0	USD	No	Thumbnail	Cats,Dogs	975.0

We noticed some problems with the dataset. Some of the columns were unnecessary, there were NaN values, and certain data was entered incorrectly.

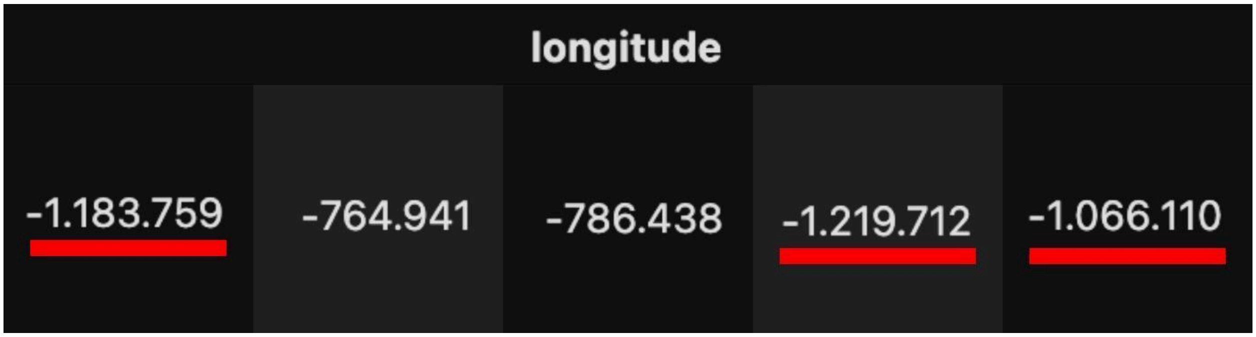
We decided which columns wouldn't be necessary:

```
UNWANTED_COLUMNS = ["amenities", "address", "pets_allowed", "body",
"has_photo", "source", "price_display", "title", "category"]
```

We dropped them:

```
df = df.drop(columns=UNWANTED_COLUMNS).dropna()
```

Now we could begin cleaning up the data within the columns!



longitude				
-1.183.759	-764.941	-786.438	-1.219.712	-1.066.110

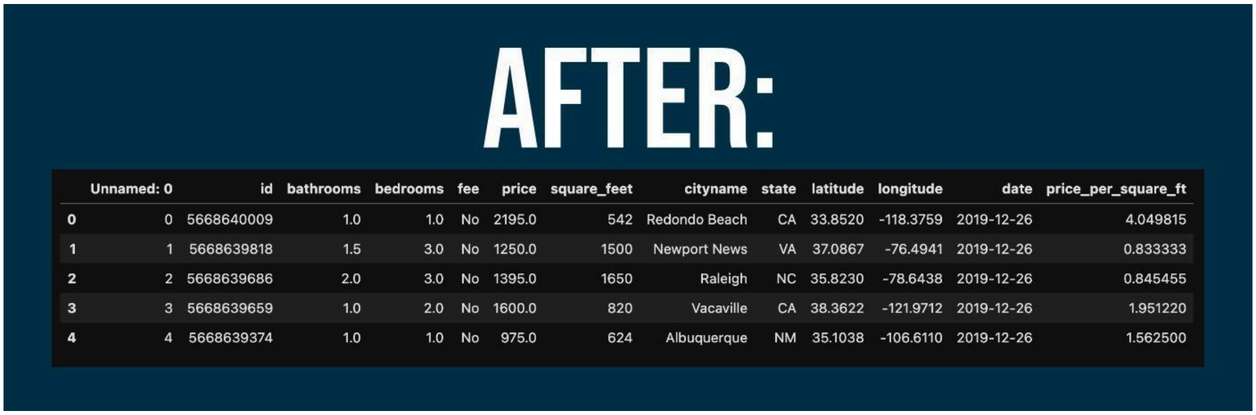
We noticed that some of the longitude values were formatted incorrectly (they had two periods instead of one). That was solved with a mapping like so:

```
df.longitude = df.longitude.map(clean_lon)
```

In essence, it went through each value in the longitude column and did the following:

```
longitude = longitude.replace(".", "", 1)
```

The first period was removed from all the longitudes and now they could be treated as numbers!



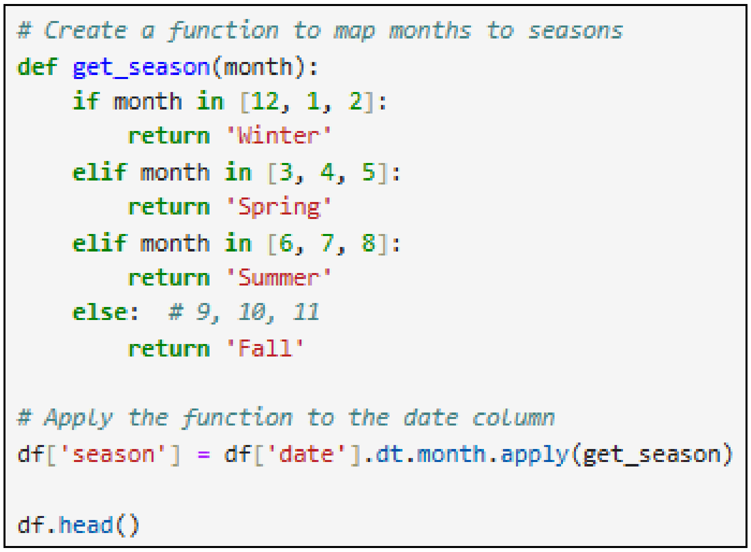
AFTER:

	Unnamed: 0	id	bathrooms	bedrooms	fee	price	square_feet	cityname	state	latitude	longitude	date	price_per_square_ft
0	0	5668640009	1.0	1.0	No	2195.0	542	Redondo Beach	CA	33.8520	-118.3759	2019-12-26	4.049815
1	1	5668639818	1.5	3.0	No	1250.0	1500	Newport News	VA	37.0867	-76.4941	2019-12-26	0.833333
2	2	5668639686	2.0	3.0	No	1395.0	1650	Raleigh	NC	35.8230	-78.6438	2019-12-26	0.845455
3	3	5668639659	1.0	2.0	No	1600.0	820	Vacaville	CA	38.3622	-121.9712	2019-12-26	1.951220
4	4	5668639374	1.0	1.0	No	975.0	624	Albuquerque	NM	35.1038	-106.6110	2019-12-26	1.562500

By the time we concluded the data cleaning process, we had a quality dataset that was ready for use. It was time to start answering questions.

Question 1: Are there any trends in apartment prices throughout the year?

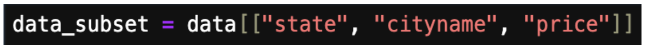
If you're considering the prices of apartments in the USA, it might be useful to know if there are any seasonal trends and how they are impacting things. To answer this question, we included some feature engineering by creating a new column called seasons where listings were grouped according to the month they were posted.



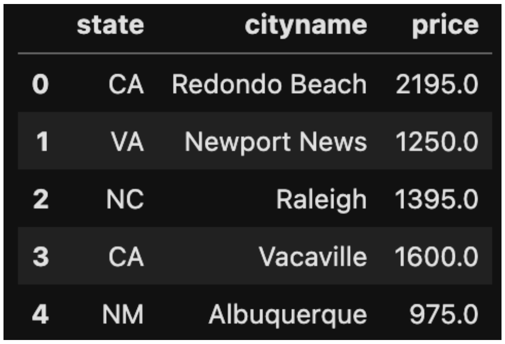
```
# Create a function to map months to seasons
def get_season(month):
    if month in [12, 1, 2]:
        return 'Winter'
    elif month in [3, 4, 5]:
        return 'Spring'
    elif month in [6, 7, 8]:
        return 'Summer'
    else:  # 9, 10, 11
        return 'Fall'

# Apply the function to the date column
df['season'] = df['date'].dt.month.apply(get_season)

df.head()
```

Question 2: How do apartment prices vary between states and cities?

If you're considering the prices of apartments in the USA, it might be useful to know how cost varies between states and cities as you decide on a location.

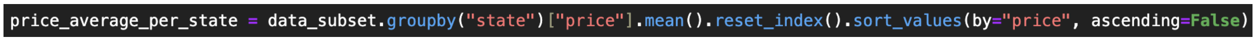


```
data_subset = data[["state", "cityname", "price"]]
```



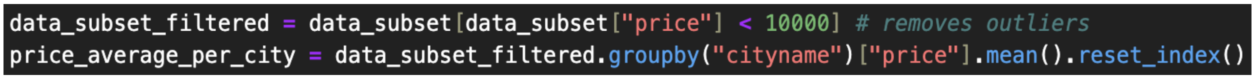
	state	cityname	price
0	CA	Redondo Beach	2195.0
1	VA	Newport News	1250.0
2	NC	Raleigh	1395.0
3	CA	Vacaville	1600.0
4	NM	Albuquerque	975.0

For this, we created a subset of our dataset to narrow things down. Now we would just be working with the following columns: state, cityname, and price.



```
price_average_per_state = data_subset.groupby("state")["price"].mean().reset_index().sort_values(by="price", ascending=False)
```

Getting the average price of apartments in each state allowed us to graph the most expensive and least expensive states in the country.



```
data_subset_filtered = data_subset[data_subset["price"] < 10000] # removes outliers
price_average_per_city = data_subset_filtered.groupby("cityname")["price"].mean().reset_index()
```

```
most_expensive_cities = price_average_per_city.sort_values(by="price", ascending=False).head(10)
```

```
least_expensive_cities = price_average_per_city.sort_values(by="price").head(10)
```

Getting the average price of apartments in each city allowed us to graph the most expensive and least expensive cities in the country. There were over 2,000 cities in our dataset! We narrowed it down to the top 10 most expensive and top 10 least expensive cities for practical purposes.

Question 3: What is more valuable to renters, having more bedrooms and bathrooms or having more square footage?

If you're considering the prices of apartments in the USA, it might be useful to know if people prefer having more bedrooms/bathrooms or if they just like having more space. Also, what might they pay to get it?

The key columns for this were price, bathrooms, bedrooms, and square_feet. The columns useful for location were cityname and state. This resulted in a total row count of 99,004 which required some processing.

Starting:

```
CLEAN_DATA_PATH = "../Project_draft/clean_data.csv"
df = pd.read_csv(CLEAN_DATA_PATH, low_memory=False)
FACILITY_COLUMN_NAMES = ["bathrooms", "bedrooms", "square_feet"]
```

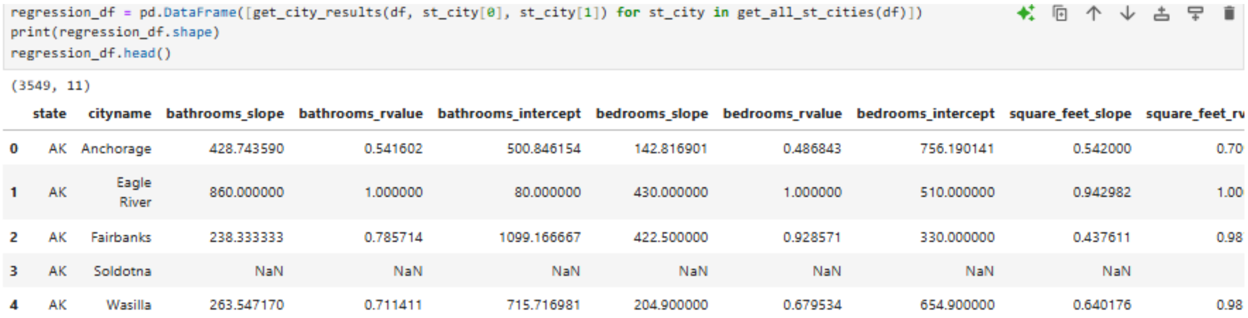
Filtering by city and state:

```
def filter_city(df, state, city):
   city_mask_s = df["cityname"] == city
   state_mask_s = df["state"] == state
   city_df = df[city_mask_s & state_mask_s]
   return city_df
```

Null and inconsistent values were then removed.

Linear regression was performed for each feature against price, slope, and r-value. The slope measured the rate of price change per unit of the feature while the r-value measured the correlation strength between the feature and the price.

Sometimes, the same city names could be found in multiple states. So it became clear that it's important to pay attention to both city name and state. For example, IL, MA, MI, MO, NJ, OH, PA, OR, TN, VA also have the same city name, Springfield. Therefore, we need to filter both city and state together.

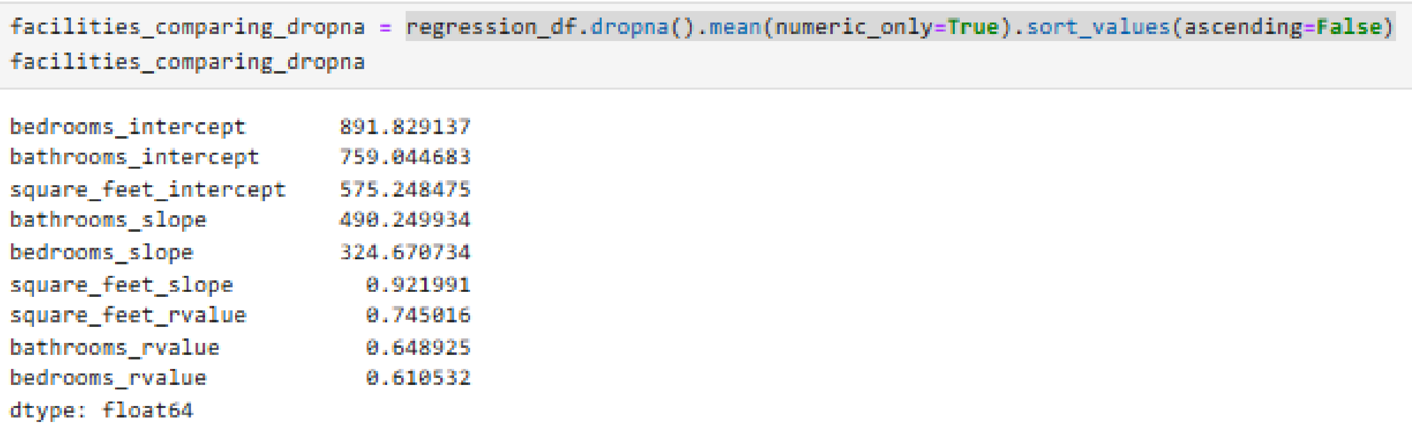


```
regression_df = pd.DataFrame([get_city_results(df, st_city[0], st_city[1]) for st_city in get_all_st_cities(df)])
print(regression_df.shape)
regression_df.head()
```

```
(3549, 11)
```

	state	cityname	bathrooms_slope	bathrooms_rvalue	bathrooms_intercept	bedrooms_slope	bedrooms_rvalue	bedrooms_intercept	square_feet_slope	square_feet_rv
0	AK	Anchorage	428.743590	0.541602	500.846154	142.816901	0.486843	756.190141	0.542000	0.70
1	AK	Eagle River	860.000000	1.000000	80.000000	430.000000	1.000000	510.000000	0.942982	1.00
2	AK	Fairbanks	238.333333	0.785714	1099.166667	422.500000	0.928571	330.000000	0.437611	0.98
3	AK	Soldotna	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
4	AK	Wasilla	263.547170	0.711411	715.716981	204.900000	0.679534	654.900000	0.640176	0.98

The DataFrame we would be using.



```
facilities_comparing_dropna = regression_df.dropna().mean(numeric_only=True).sort_values(ascending=False)
facilities_comparing_dropna
```

```
bedrooms_intercept       891.829137
bathrooms_intercept      759.044683
square_feet_intercept    575.248475
bathrooms_slope          490.249934
bedrooms_slope           324.670734
square_feet_slope          0.921991
square_feet_rvalue         0.745016
bathrooms_rvalue           0.648925
bedrooms_rvalue            0.610532
dtype: float64
```

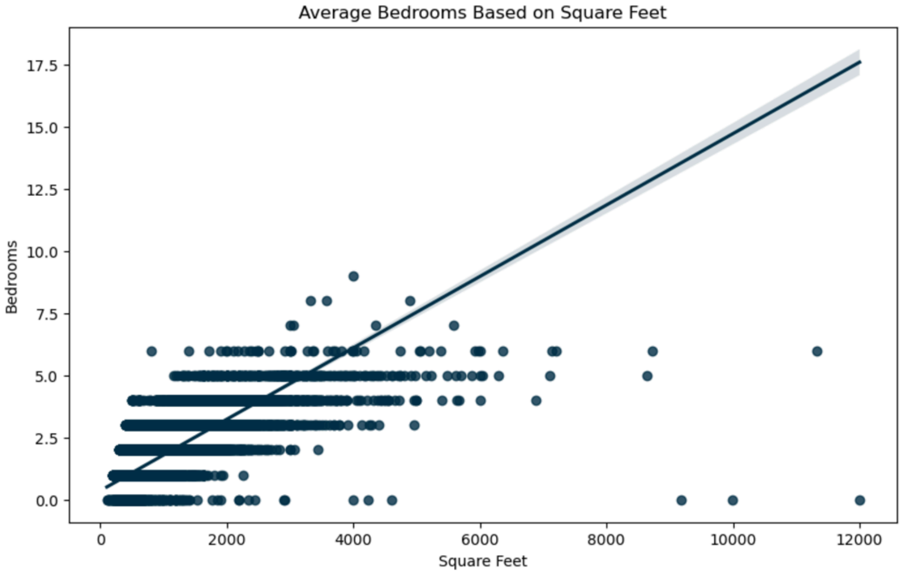
A comparison from the DataFrame.

We found that all three variables (bedrooms, bathrooms, and square feet) have a positive relationship with apartment price. As any of these variables increase, the price tends to increase as well. Among the three, square feet had the highest explanatory power regarding price, followed by bathrooms and then bedrooms.

Regression

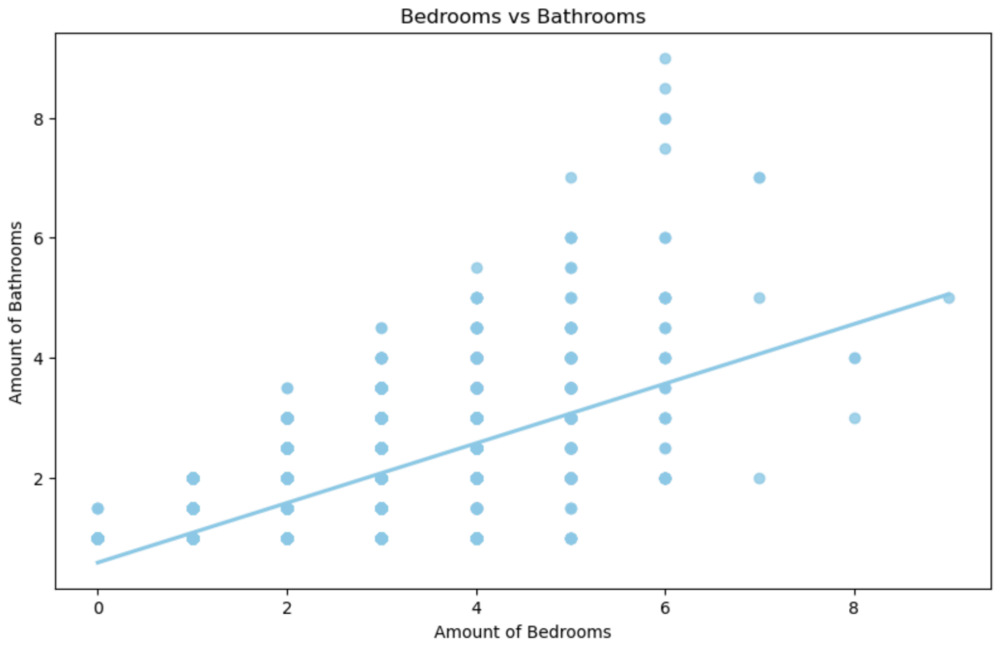
What kind of regressions appear in our data set? And what stories do they tell?
The questions answered earlier in the project analyzed relationships between price and multiple other variables such as square feet, amount of bedrooms and bathrooms, time of year, etc. But what other variables in our data have noticeable or interesting regressional relationships? When looking for an apartment to rent, two important

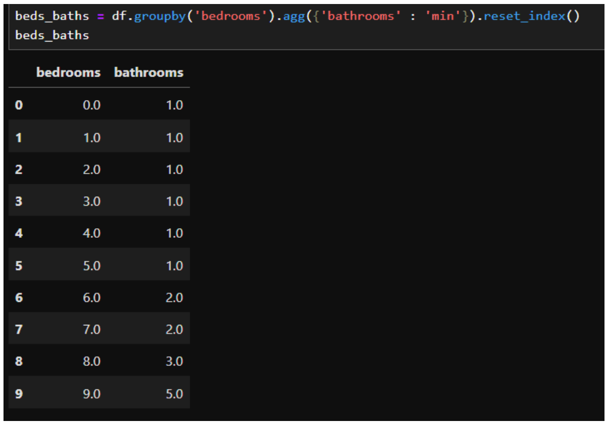
aspects may be the amount of bedrooms and the amount of square footage available, so it might be good to know what the relationship between these variables is.



We can see that there is a consistent positive regression between square feet and amount of bedrooms. We can also see there is a slight confidence interval on our regression line. This is because with our wide range of data there are bound to be some extreme outliers. In the specific example above, note that somewhere in the US you can get a rather large studio apartment with no bedrooms at all. This is one exception outlier to the general conclusion that as square footage increases, the amount of bedrooms also increases.

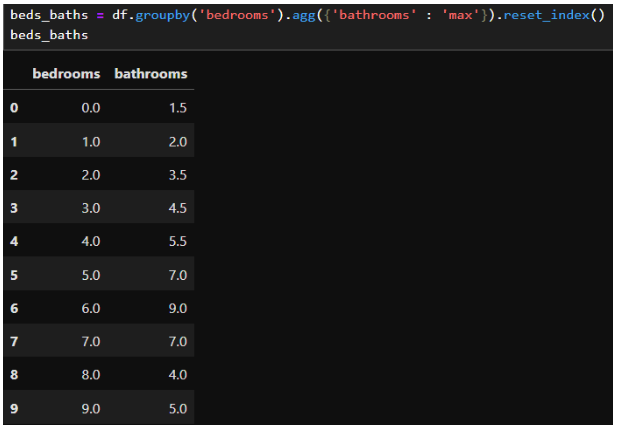
Another interesting regression is the relationship between the amount of bedrooms and amount of bathrooms in apartments. There is a strong positive regression overall. What stands out in the chart below is that depending on where you rent a six bedroom apartment in the US, you could have anywhere from 2 to 9 bathrooms. The raw data can be grouped by amount of bedrooms and aggregated with the minimum and maximum amount of bathrooms to confirm how wide of a range there is.





```
beds_baths = df.groupby('bedrooms').agg({'bathrooms' : 'min'}).reset_index()
beds_baths
```

	bedrooms	bathrooms
0	0.0	1.0
1	1.0	1.0
2	2.0	1.0
3	3.0	1.0
4	4.0	1.0
5	5.0	1.0
6	6.0	2.0
7	7.0	2.0
8	8.0	3.0
9	9.0	5.0



```
beds_baths = df.groupby('bedrooms').agg({'bathrooms' : 'max'}).reset_index()
beds_baths
```

	bedrooms	bathrooms
0	0.0	1.5
1	1.0	2.0
2	2.0	3.5
3	3.0	4.5
4	4.0	5.5
5	5.0	7.0
6	6.0	9.0
7	7.0	7.0
8	8.0	4.0
9	9.0	5.0

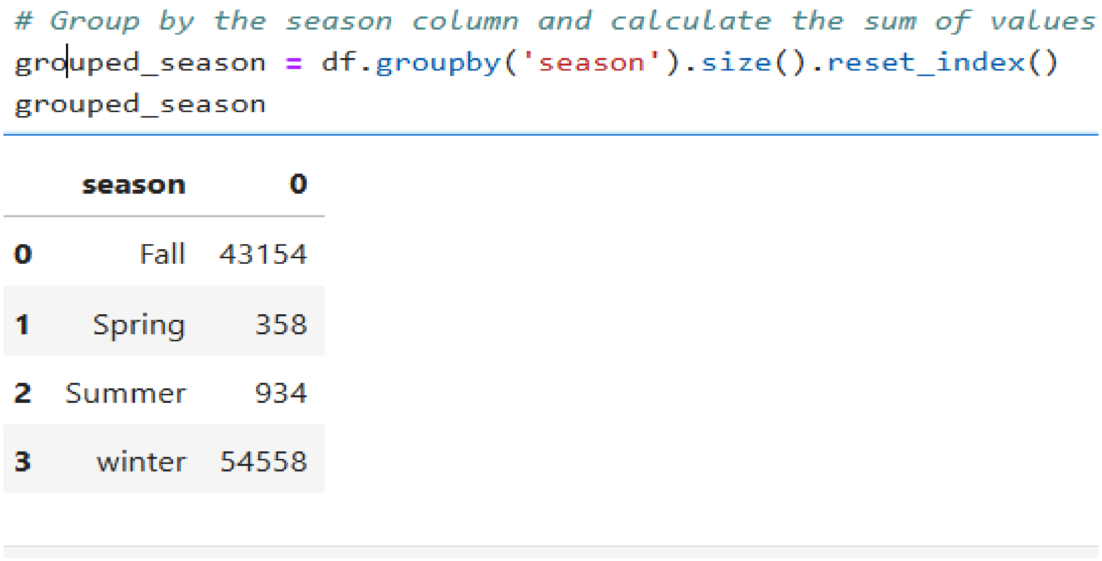
Biases & Limitations

When analyzing and visualizing the impact of features such as "bathrooms," "bedrooms," and "square_feet" on apartment prices, there are several limitations and potential biases to consider:

Limitations:

1. **Timeline:**
    - Old data since 2019

- Unequal rows of each season. For instance: When we groupby the seasons, the Fall and the Winter have most of the value, while the Spring has only 358 rows and the Summer has only 934 rows. This leads to unequal data for comparison.



```
# Group by the season column and calculate the sum of values
grouped_season = df.groupby('season').size().reset_index()
grouped_season
```

	season	0
0	Fall	43154
1	Spring	358
2	Summer	934
3	winter	54558

2. **Data Quality and Completeness**:
    - **Missing Data**: If some entries for bathrooms, bedrooms, square_feet, or price are missing, this can skew the results. ⇒ We cleaned the data thoroughly, handled missing values, and removed outliers appropriately.
    - **Outliers**: Extremely high or low values in the dataset (e.g., luxury apartments or very small spaces) can disproportionately influence regression and correlation analyses.
    - **Simplistic Linear Relationships:** Linear regression assumes a linear relationship between features and price, but real-world relationships may be non-linear or influenced by external factors (e.g., proximity to amenities, local economy). This might oversimplify the true impact of features. ⇒ Consider interaction terms (e.g., square_feet × groupby city/state) in our regression analysis.
3. **Geographical Context**
    - Differences in real estate markets across cities and states can result in variations that are not captured when combining data from multiple locations.
    - For example, the impact of "square_feet" in a small city might differ significantly from a metropolitan area.
4. **Temporal Factors**:

    - Real estate prices fluctuate due to market conditions, seasons, or economic trends. If the dataset spans several years or months without accounting for time, the results may not be relevant to current trends.

Bias:

1. **Selection Bias**:
    - If the dataset only includes properties from specific regions, price ranges, the analysis may not generalize to all apartments. For instance, if the data is mostly from urban areas, the findings may not apply to suburban or rural markets.
2. **Feature Bias**:
    - The analysis focuses solely on "bathrooms," "bedrooms," and "square_feet," but other important factors like **parking availability, school district**, **age of the building**, **amenities**, and **neighborhood safety** can also significantly affect prices. Without these, the model might be incomplete or provide misleading conclusions.
3. **Correlation vs. Causation**:
    - High correlation does not imply causation. For instance, larger apartments might correlate with higher prices, but the cause could also be location or market demand, not just the size itself.
4. **Regional Economic Bias**:
    - Local economic conditions or housing policies can significantly influence prices. For example, rent control in certain cities might cap prices regardless of the number of features.

Conclusion

This was an interesting analysis which revealed many insights about apartment prices in the USA while giving us the chance to utilize our technical skills.