AIML 1104 – TERM PROJECT

Group:

Khalid (Inactive)

Milan ??

Hoang Luong C0775283

Note: Khalid has been dropped out of the class due to visa rejection, so we are currently working with two people.

Handle Data

```
[4] df.shape
```

```
(10000, 22)
```

The current dataset initially has 10000 records and 22 features.

```
#4
print(df.columns.values)
```

```
['id' 'category' 'title' 'body' 'amenities' 'bathrooms' 'bedrooms'
 'currency' 'fee' 'has_photo' 'pets_allowed' 'price' 'price_display'
 'price_type' 'square_feet' 'address' 'cityname' 'state' 'latitude'
 'longitude' 'source' 'time']
```

These are the column names of our dataset.

```
#4
df.isnull().sum()
```

```
id                  0
category            0
title               0
body                0
amenities        3549
bathrooms          34
bedrooms            7
currency            0
fee                 0
has_photo           0
pets_allowed     1748
price               0
price_display       0
price_type          0
square_feet         0
address          3327
cityname           77
state              77
latitude           10
longitude          10
source              0
time                0
dtype: int64
```

Using df.isnull().sum on dataframe object df we could find all missing values. We have Amenities=3549, Bathrooms=33,Bedrooms, Pet_allowed=1745, address=3325, cityname=77, state=77, latitute=10,longitude=10.

```
#6
#Total no of rows missing
sum(df.isnull().any(axis=1))
```

```
6457
```

Total number of rows having missing values is 6457.

s	cityname	state	latitude	longitude	source	time
N	NaN	NaN	NaN	NaN	RentLingo	1577014825
N	NaN	NaN	NaN	NaN	RentLingo	1577358921

time column is currently in UNIX Timestamp (second). So we transform it into a more readable format (YY-MM-DD HH:MM:SS)

After using numpy to check the unique value of **category**, we have found that it only contains 3 unique values:

```
#9
import numpy as np
# Checking out the total unique categories in the dataset
np.unique(df['category'])

# Found that there are 3 unique categories ['housing/rent/apartment', 'housing/rent/home', 'housing/rent/short_term']
```

```
array(['housing/rent/apartment', 'housing/rent/home',
       'housing/rent/short_term'], dtype=object)
```

```
# checking that what are the occurences of each unique categories in the dataset
df['category'].value_counts()
```

```
housing/rent/apartment     9996
housing/rent/home             2
housing/rent/short_term       2
Name: category, dtype: int64
```

In addition, **home** and **short_term** only appeared 2 times each. So we decided to delete their records and this column.

```
#14
np.unique(df['currency'])
```

```
array(['USD'], dtype=object)
```

The **currency** feature only contains USD, so we decided to drop it.

body and **title** share the same information with **address**, **city**, **state**. So we deleted **body** and **title**.

price_display is just **price** with a dollar symbol before the number so we also decided to drop it.

id is meaning less without its related entities, so we deleted it.

fee only contains value “No” so we also deleted it.

```
del df['title']
del df['body']
del df['price_display']
del df['id']
def df['fee']
df.head()
```

```
df.head()
```

title	body	amenities	bathrooms	bedrooms	fee	has_photo	pets_allowed	price	price_display	price_type	square_feet	address	cityname	state
Studio apartment 2nd St NE, Uhland Terrace NE,...	This unit is located at second St NE, Uhland T...	NaN	NaN	0.0	No	Thumbnail	None	790	$790	Monthly	101	NaN	Washington	DC
Studio apartment 814 Schutte Road	This unit is located at 814 Schutte Road, Evan...	NaN	NaN	1.0	No	Thumbnail	None	425	$425	Monthly	106	814 Schutte Rd	Evansville	IN

We also dropped all NaN value from **latitude**, **longitude**, **state**, **cityname**.

```
#38
df=df.dropna(subset=['latitude'])
df=df.dropna(subset=['longitude'])
df=df.dropna(subset=['state'])
df=df.dropna(subset=['cityname'])
```

After finishing dropping all missing value records, we ended up with:

```
#39
df.isnull().sum()
```

```
amenities       3289
bathrooms          0
bedrooms           0
fee                0
has_photo          0
pets_allowed       0
price              0
price_type         0
square_feet        0
address         1674
cityname           0
state              0
latitude           0
longitude          0
source             0
time               0
dtype: int64
```

Measure of central tendency

```
[276] number_df = df[['bathrooms', 'bedrooms', 'price', 'square_feet']]
      string_df = df[['pets_allowed', 'cityname', 'state', 'source']]
      df.head()
```

We will split our dataset into 2 dataset, one contains only string and the other contains only number.

```
[277] string_df.describe()
```

	pets_allowed	cityname	state	source
count	8129	8129	8129	8129
unique	4	1436	51	10
top	Cats,Dogs	Austin	TX	RentLingo
freq	5169	522	1555	6793

```
[278] number_df.describe()
```

	bathrooms	bedrooms	price	square_feet
count	8,129	8,129	8,129	8,129
mean	1	2	1,461	938
std	1	1	909	537
min	1	0	200	107
25%	1	1	925	625
50%	1	2	1,250	788
75%	2	2	1,675	1,100
max	5	6	19,500	6,300

By using describe(), we can achieve some basic statistical details.

Below is the mode of string features:

```
1 string_df.mode()
```

	pets_allowed	cityname	state	source
0	Cats,Dogs	Austin	TX	RentLingo

Below is the measure of central tendency for number features:

```
Mean:
 bathrooms         1
bedrooms          2
price         1,461
square_feet     938
dtype: float64
Median:
 bathrooms         1
bedrooms          2
price         1,250
square_feet     788
dtype: float64
Mode:
    bathrooms  bedrooms  price  square_feet
0          1         1   1350          700
Standard deviation:
 bathrooms       1
bedrooms        1
price         909
square_feet   537
dtype: float64
IQR:
 bathrooms       1
bedrooms        1
price         750
square_feet   475
dtype: float64
```

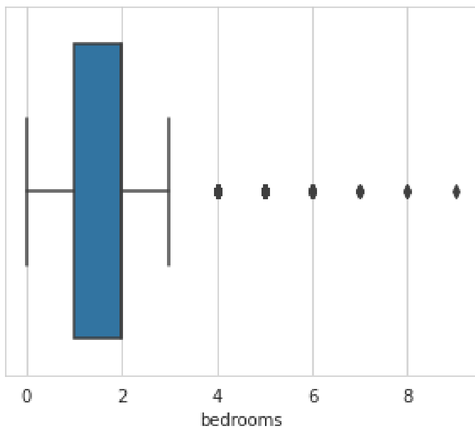
Data visualization

Outliers detection

Bedroom features:

```

sns.boxplot(x=df['bedrooms'])

```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f0a8c37e510>
```




```
#60

df['bedrooms'].value_counts()
```

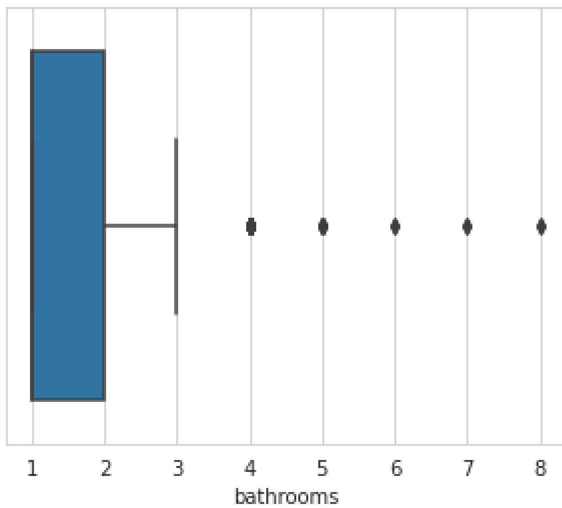
```
1.0    3858
2.0    2509
3.0    1107
4.0     385
0.0     175
5.0      84
6.0      15
7.0       3
8.0       2
9.0       1
Name: bedrooms, dtype: int64
```

From the above results it can be observed that bedrooms=7,8,9 has only less than 3 rows and it is an outlier so we will delete these rows.

Bathroom feature:

```
#53
import seaborn as sns
sns.boxplot(x=df['bathrooms'])


```

<matplotlib.axes._subplots.AxesSubplot at 0x7f0a8c326990>




```
df['bathrooms'].value_counts()
```

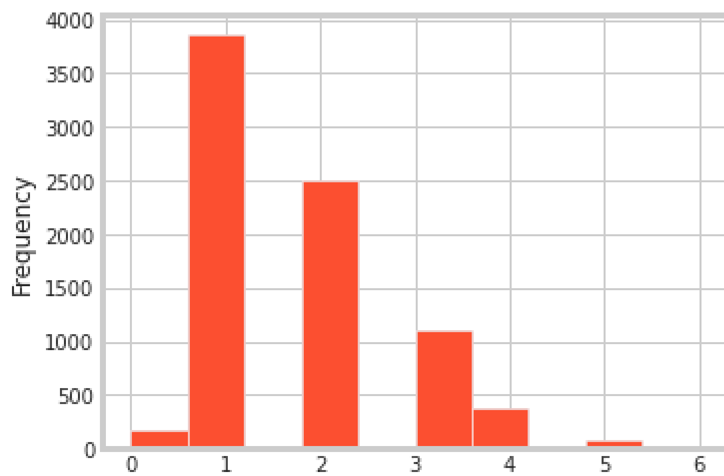
```
1.0    5586
2.0    2249
3.0     170
4.0     117
5.0       7
8.0       2
6.0       2
Name: bathrooms, dtype: int64
```

It can be seen from the above results that there is only 2 rows for 6 and 8 bathrooms each. So we can delete them to avoid outliers.

Frequency plotting of some features:

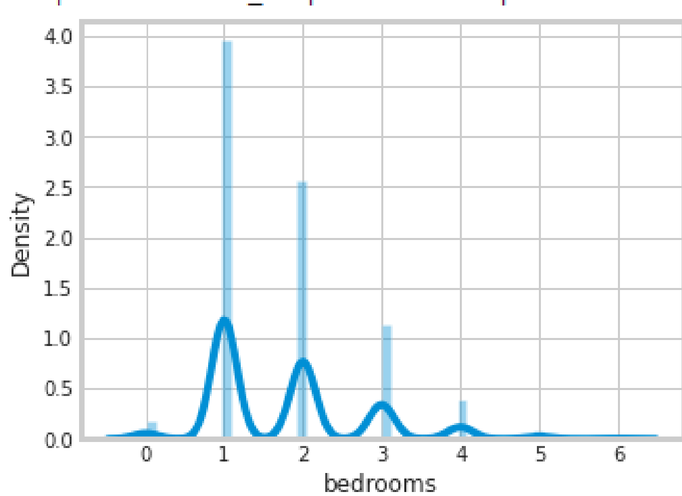
```python
#63
bedrooms.plot(kind='hist')
#17
plt.hist(bedrooms)
plt.plot()

```

```
[]
```

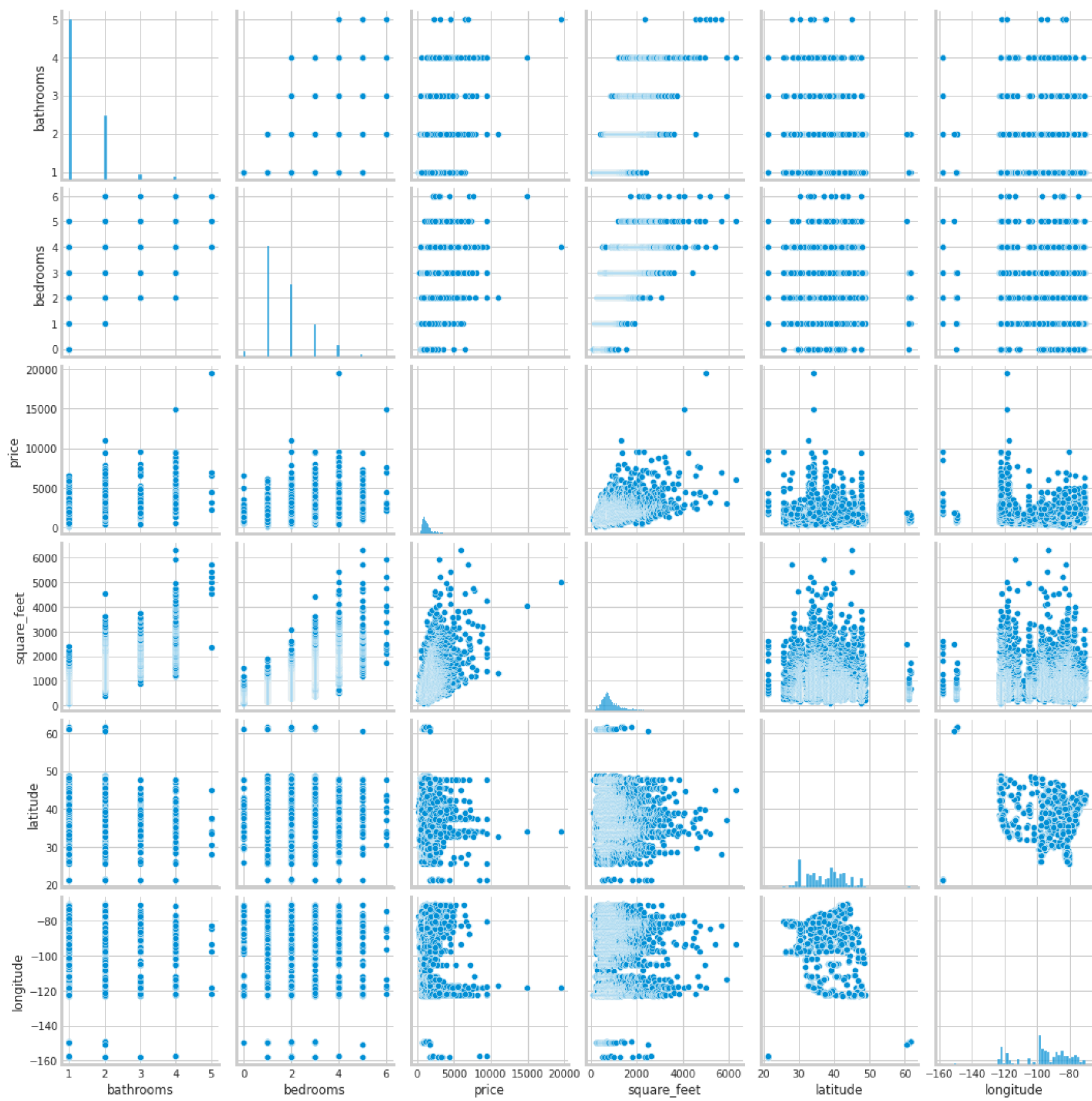



```python
#63
sb.distplot(bedrooms)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f883e754550>
```



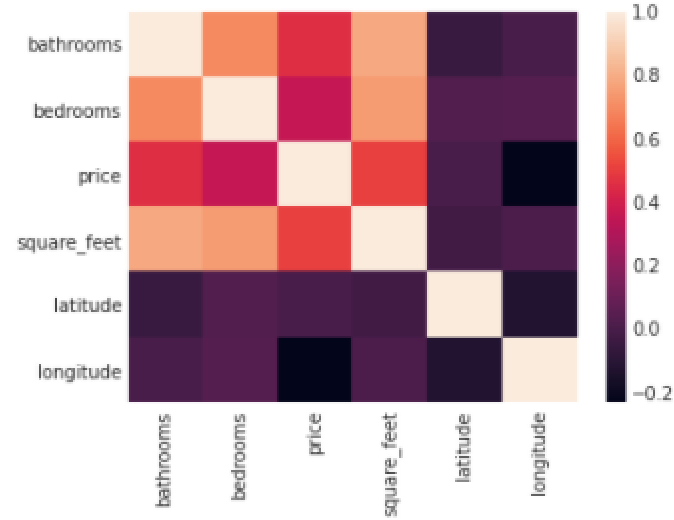
Pair plot:



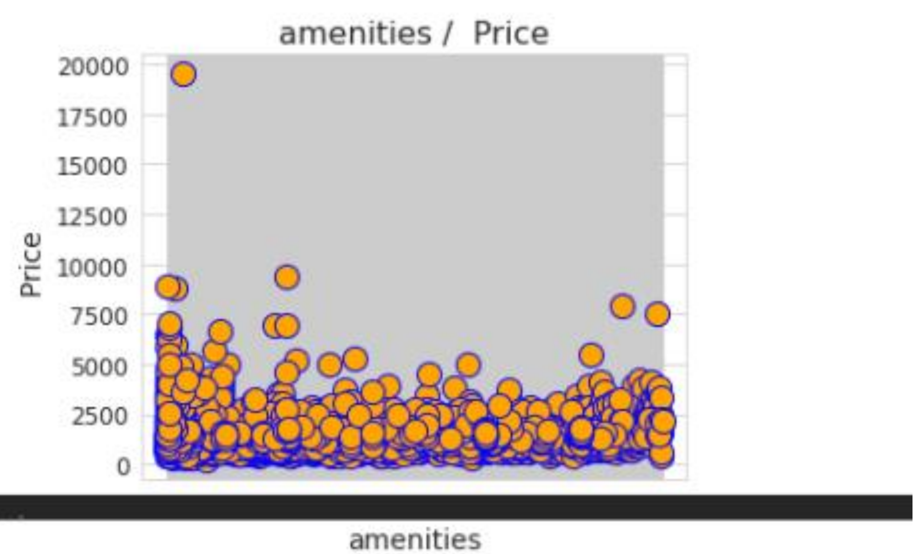
Dataframe correlation heatmap:

```
#65
df.corr()
sb.heatmap(df.corr())
```

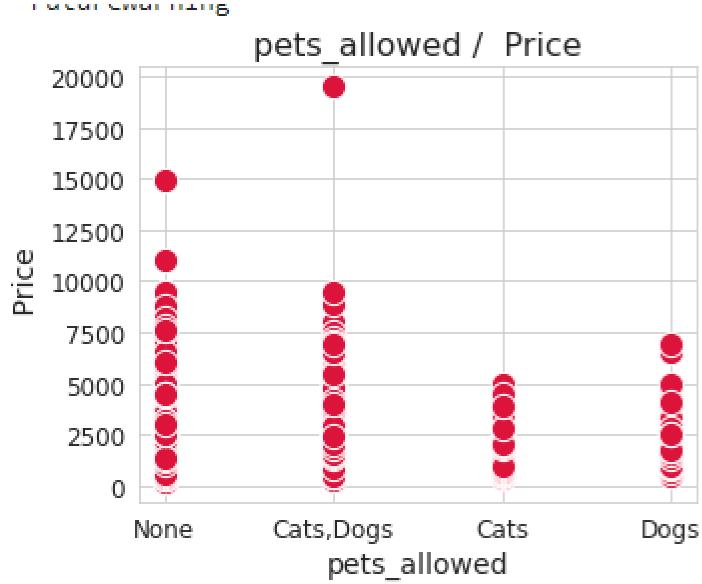
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f883eaa5150>
```

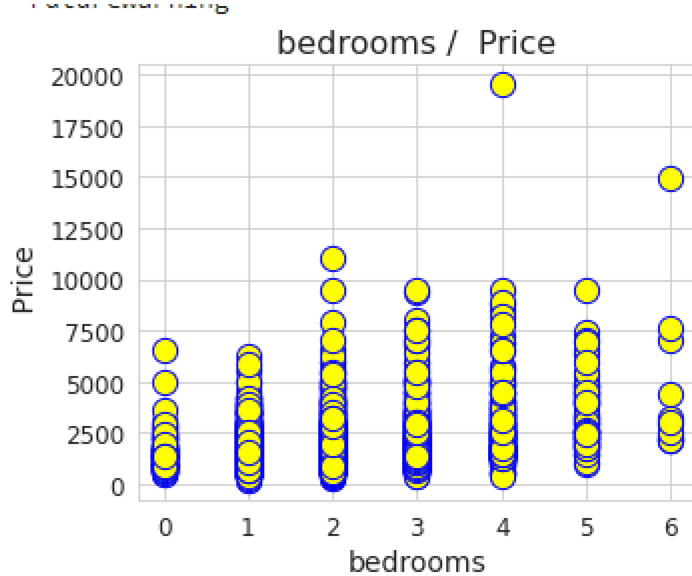


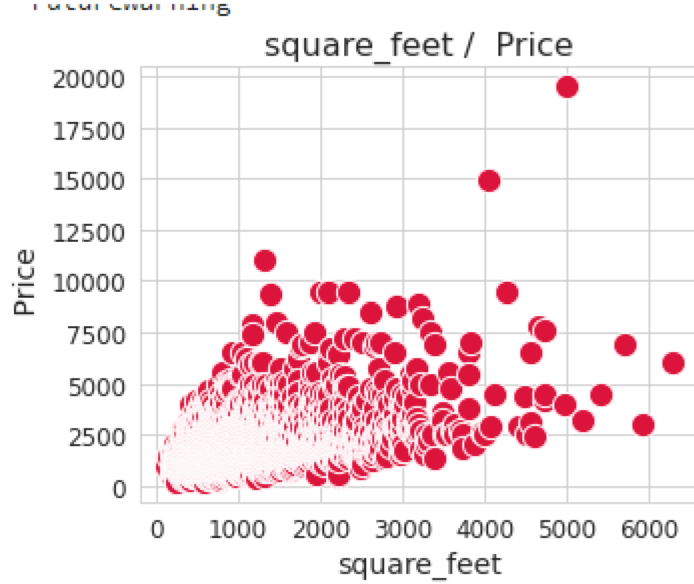
Scatter plot



Like heatmap, a scatter plot is also used to observe linear relations between two variables in a dataset. In a scatter plot, the dependent variable is marked on the x-axis and the independent variable is marked on the y-axis. In our case, the 'SalePrice' attribute is the dependent variable, and every other are the independent variables. It would be difficult to produce a plot for each variable, so we can define a function that takes only the dependent variable and returns a scatter plot for every independent variable present in a dataset.

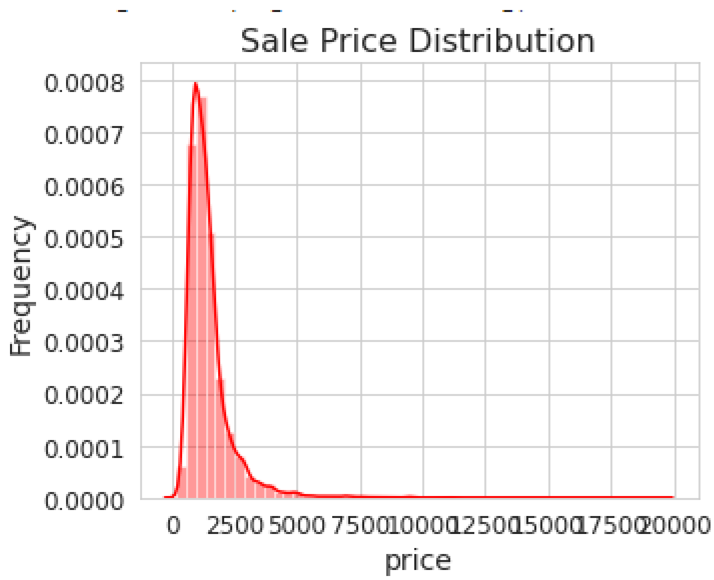






Distribution plot

Distribution plots are very useful to check how well a variable is distributed in the dataset. Let's now produce a distribution plot using the 'distplot' function to check the distribution of the 'SalePrice' variable in the dataset

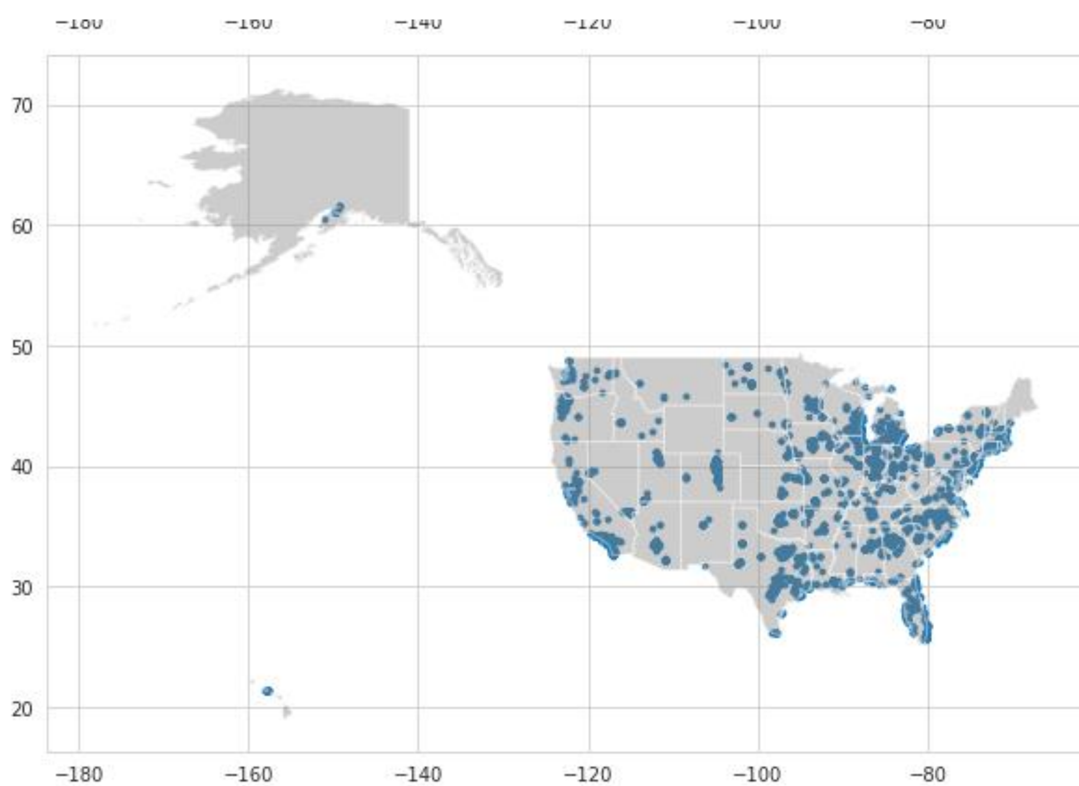


Exploratory Data Analysis

Using geopandas and geometry libs, we can visualize the location of records and do some exploratory data analysis on that.

It is required to have the USA map package to run this visualization. You can download it here: https://drive.google.com/drive/folders/1AzJ9dv0AdzyOOeshsupeUzBCz0SBveoW

After finishing downloading these files, create a /content folder and store these file in there.



We can try to use some regression model such as Facebook Prophet to predict the price in the future for 1 state. Below is the prediction for the price of accommodation in Texas.