Presentation of Findings

Matthew E. Heino

Data Analytics Graduate Capstone

Introduction

This executive summary is to discuss the outcomes of a research question that is proposed to answer a question that can be used to address a business issue. The business issue at hand is whether it is possible to develop a clustering model that can be used to determine the by location the price of the apartment. Is there a model that can be used to cluster the given apartment data using only a handful of features? This summary document will discuss the following key areas:

- The problem will include a discussion of the hypothesis.
- A summary of the data analysis process.
- A brief outline of the findings of the analysis.
- The limitations of the techniques and the tools used.
- Actions that can be taken based on the data analysis.
- Expected benefits of the study.

The Problem and the Hypothesis

The problem that was proposed to be solved was, "To what extent does the latitude and longitude of an apartment affect the price." This question wants to address the notion that areas of the United States have higher apartment prices compared to other areas. Is this price differential a feature of the location overall or is it based on the municipality or metropolitan location?   We look at location using only the location that is given using the latitude and longitude of the observations in the data.

This problem will not look at the location based on information like the city or the state. These features are often used to create models of this type. The clustering algorithm will only be given a handful of features to see if these features can aid in determining the price of the apartment.

The research question will help in the hypothesis for the problem the following:

- **Null hypothesis**: Using latitude and longitude as clustering features is it possible to produce a cluster that has silhouette score above 0.60.
- **Alternate hypothesis:** Using latitude and longitude as clustering features it is not possible to produce a cluster that has a silhouette score above 0.60.

The basic hypothesis is, "Is it possible to develop clusters that are adequately separated that can be of use?" This separation does not imply that the model will be able to be used in the manner prescribed earlier. It is possible the model will not cluster in a manner that makes it easy to determine the price of the model. The overall question is whether it is possible to create a clustering model that will cluster adequately so that a price can be arrived at.

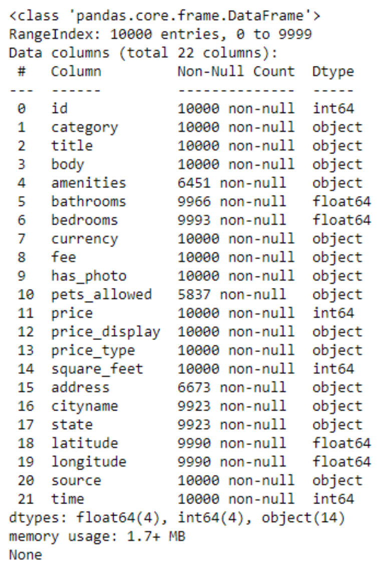
The Data Analysis Process

The data for this process began by procuring the appropriate data set for the question at hand. The data that was chosen came from a reputable repository. The repository is located at the University of California – Irvine. The name of the repository is the UC Irvine Machine Learning Repository.

The data set is composed of two different files. There is one that contains 10,000 observations and the other contains 100,000. The model for this problem will make use of the dataset that is composed of 10,000 observations. Whether using the larger or the smaller dataset there are 22 columns or features in the dataset. Not all these features will be used to create the model. The ones of interest are the longitude and the latitude. Other features will be cleaned to provide some utility when creating certain visualizations.

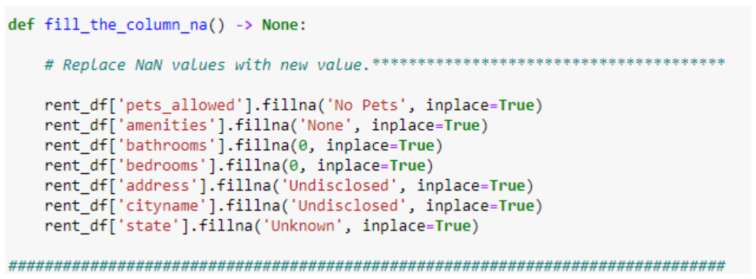
The data from the website was stored in a ZIP file that contained 7-zip files for the datasets. These were extracted using the **7-Zip File Manager**. The extracted data was saved to a directory to be used directly.

This data required some cleaning and preparation to get the data into a state that is suitable for use in the clustering model. Looking at the dataset some features lacked values that needed to be dealt with. The visual below shows the initial state of the data.



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 22 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   id             10000 non-null  int64
 1   category       10000 non-null  object
 2   title          10000 non-null  object
 3   body           10000 non-null  object
 4   amenities      6451 non-null   object
 5   bathrooms      9966 non-null   float64
 6   bedrooms       9993 non-null   float64
 7   currency       10000 non-null  object
 8   fee            10000 non-null  object
 9   has_photo      10000 non-null  object
 10  pets_allowed   5837 non-null   object
 11  price          10000 non-null  int64
 12  price_display  10000 non-null  object
 13  price_type     10000 non-null  object
 14  square_feet    10000 non-null  int64
 15  address        6673 non-null   object
 16  cityname       9923 non-null   object
 17  state          9923 non-null   object
 18  latitude       9990 non-null   float64
 19  longitude      9990 non-null   float64
 20  source         10000 non-null  object
 21  time           10000 non-null  int64
dtypes: float64(4), int64(4), object(14)
memory usage: 1.7+ MB
None
```

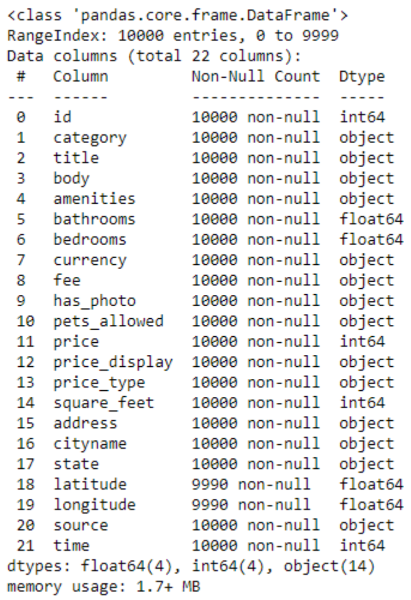
Please note not all the features or columns will be used to solve the problem that was discussed earlier in this document. As you can see there are 10,000 entries in this dataset. Numerous observations are missing values. These missing values will be imputed using the appropriate means. An example of how these values were imputed to make sure these values had logical values is shown below.



```
def fill_the_column_na() -> None:

    # Replace NaN values with new value.**************************************

    rent_df['pets_allowed'].fillna('No Pets', inplace=True)
    rent_df['amenities'].fillna('None', inplace=True)
    rent_df['bathrooms'].fillna(0, inplace=True)
    rent_df['bedrooms'].fillna(0, inplace=True)
    rent_df['address'].fillna('Undisclosed', inplace=True)
    rent_df['cityname'].fillna('Undisclosed', inplace=True)
    rent_df['state'].fillna('Unknown', inplace=True)

##############################################################################
```

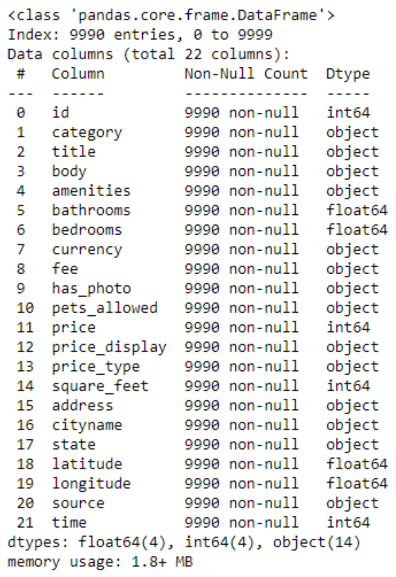
All missing values have been changed to a logical value for each of the features. After running the code above the state of the data is shown below.

Please note that there are still missing values for the two most important features in the dataset. There are 10 missing values for the latitude and the longitude. Since only 10 values are



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 22 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   id             10000 non-null  int64  
 1   category       10000 non-null  object 
 2   title          10000 non-null  object 
 3   body           10000 non-null  object 
 4   amenities      10000 non-null  object 
 5   bathrooms      10000 non-null  float64
 6   bedrooms       10000 non-null  float64
 7   currency       10000 non-null  object 
 8   fee            10000 non-null  object 
 9   has_photo      10000 non-null  object 
 10  pets_allowed   10000 non-null  object 
 11  price          10000 non-null  int64  
 12  price_display  10000 non-null  object 
 13  price_type     10000 non-null  object 
 14  square_feet    10000 non-null  int64  
 15  address        10000 non-null  object 
 16  cityname       10000 non-null  object 
 17  state          10000 non-null  object 
 18  latitude       9990 non-null   float64
 19  longitude      9990 non-null   float64
 20  source         10000 non-null  object 
 21  time           10000 non-null  int64  
dtypes: float64(4), int64(4), object(14)
memory usage: 1.7+ MB
```

missing from these features. The rows missing the data will be dropped from the dataset. This will result in a data frame that is now composed of 9990 rows or observations. This is still a suitable amount to begin to create the clustering model. The state of the data frame is shown below.



```
<class 'pandas.core.frame.DataFrame'>
Index: 9990 entries, 0 to 9999
Data columns (total 22 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   id             9990 non-null   int64  
 1   category       9990 non-null   object 
 2   title          9990 non-null   object 
 3   body           9990 non-null   object 
 4   amenities      9990 non-null   object 
 5   bathrooms      9990 non-null   float64
 6   bedrooms       9990 non-null   float64
 7   currency       9990 non-null   object 
 8   fee            9990 non-null   object 
 9   has_photo      9990 non-null   object 
 10  pets_allowed   9990 non-null   object 
 11  price          9990 non-null   int64  
 12  price_display  9990 non-null   object 
 13  price_type     9990 non-null   object 
 14  square_feet    9990 non-null   int64  
 15  address        9990 non-null   object 
 16  cityname       9990 non-null   object 
 17  state          9990 non-null   object 
 18  latitude       9990 non-null   float64
 19  longitude      9990 non-null   float64
 20  source         9990 non-null   object 
 21  time           9990 non-null   int64  
dtypes: float64(4), int64(4), object(14)
memory usage: 1.8+ MB
```

The data is now in a state to be used for modeling. To help get a feel for what the data looks like visually a quick scatterplot was created.

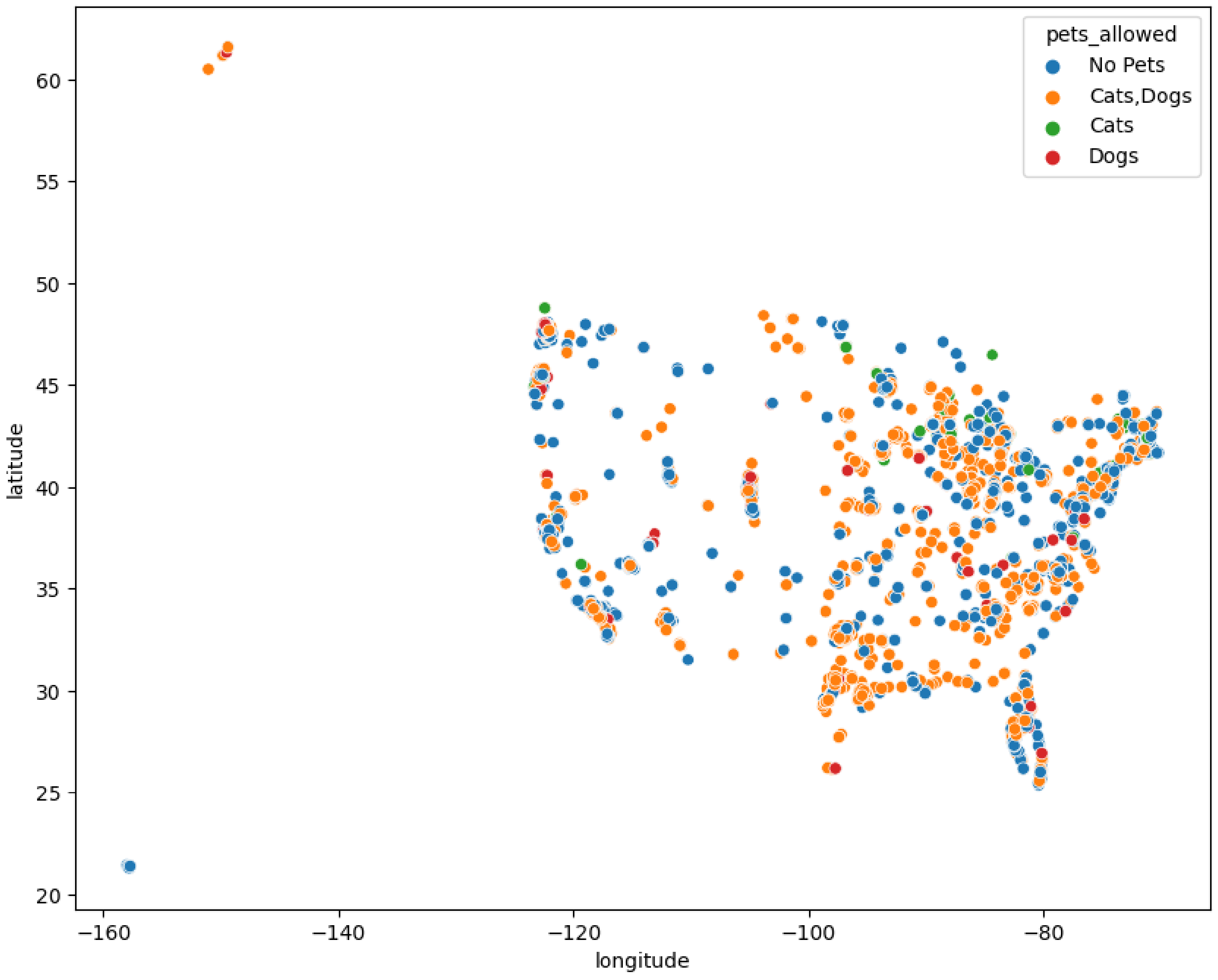
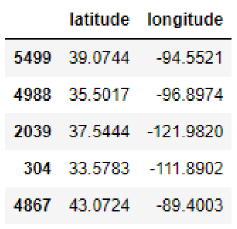


Image 1: The data visualized.

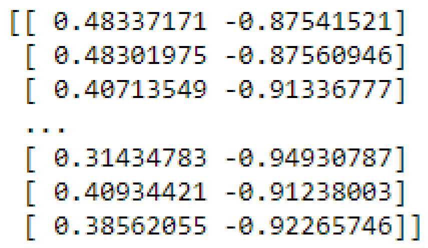
As you can see the data takes on roughly the shape of the United States. This shape will be advantageous when this analysis is discussed later.

The data was then split into training and testing sets. The training will have about 80% of the observations in it. The rest will be in the training. The training was not implemented in this attempt to develop the model. It was not required. An example of the training set is given below.



	latitude	longitude
5499	39.0744	-94.5521
4988	35.5017	-96.8974
2039	37.5444	-121.9820
304	33.5783	-111.8902
4867	43.0724	-89.4003

This data was normalized. It was normalized using a method found in the **sklearn preprocessing** module. Other methods could have been implemented to normalize the data but this seemed to be the most appropriate for the given the type of data and its method of storage. An example of the normalized data is shown below.

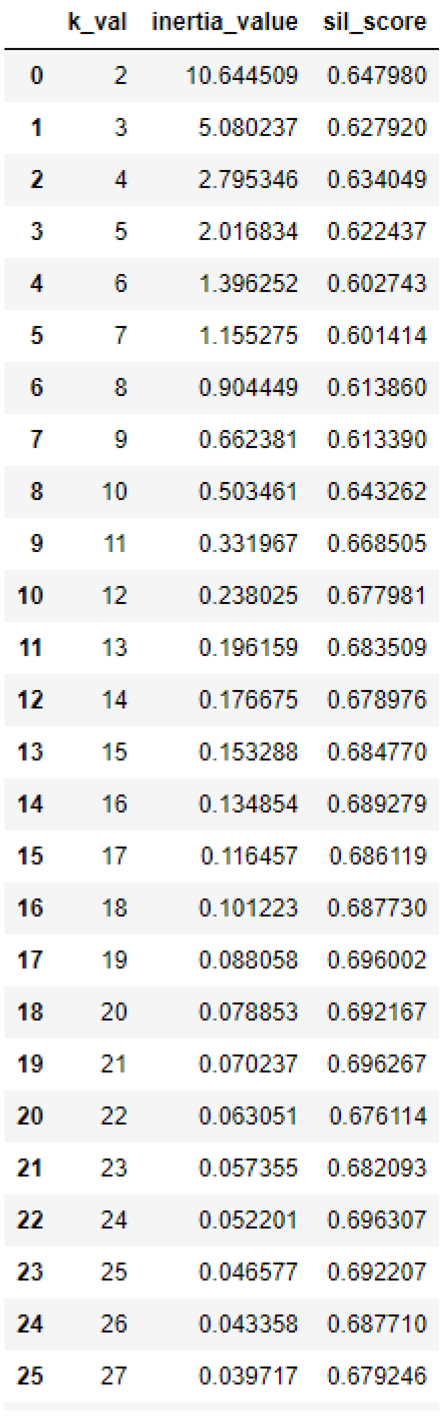
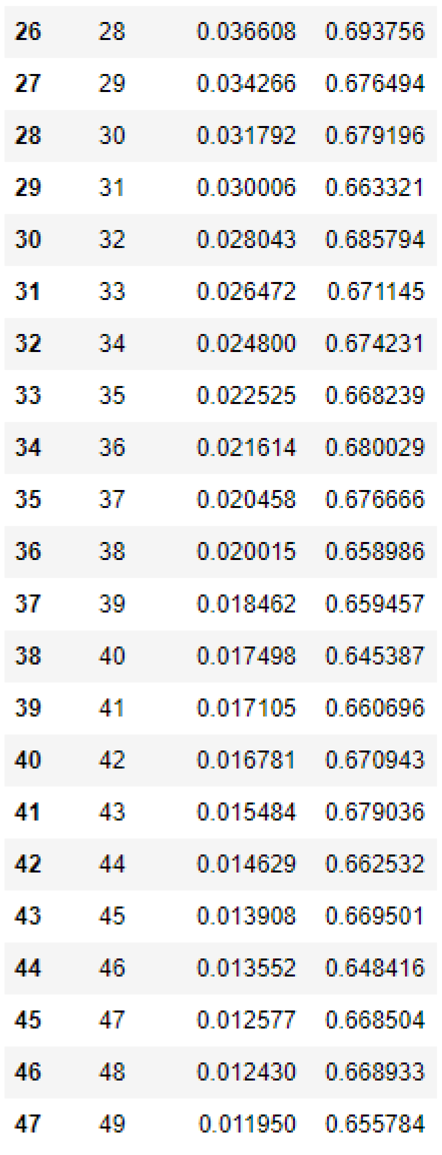


```
[[ 0.48337171 -0.87541521]
 [ 0.48301975 -0.87560946]
 [ 0.40713549 -0.91336777]
 ...
 [ 0.31434783 -0.94930787]
 [ 0.40934421 -0.91238003]
 [ 0.38562055 -0.92265746]]
```

With the completion of the analysis, it is now possible to begin the actual analysis. The first step will be to create the first model of the clustering algorithm. This model aims to create a model that has a silhouette score of at least 0.60. This score will dictate the number of clusters that will be utilized to create the final model. Other parameters were used to help optimize the model. The parameters were the following:

- The initializing strategy that was implemented for the model was using k-means++. This method was chosen because it will allow for the random selection of a data point to be used as the initial centroid of a cluster (Mayo, 2022).
- The algorithm that was chosen was the “elkan” algorithm. This algorithm was chosen because this will reduce the likelihood of redundant calculations being computed (VLFeat, n.d.).

The total number of clusters that were optimal for the desired silhouette score was 24. There were quite a few k-values used to see where the model could yield the desired silhouette score. The table below shows the values of k that were tried and the silhouette score along with the inertia.

	k_val	inertia_value	sil_score
0	2	10.644509	0.647980
1	3	5.080237	0.627920
2	4	2.795346	0.634049
3	5	2.016834	0.622437
4	6	1.396252	0.602743
5	7	1.155275	0.601414
6	8	0.904449	0.613860
7	9	0.662381	0.613390
8	10	0.503461	0.643262
9	11	0.331967	0.668505
10	12	0.238025	0.677981
11	13	0.196159	0.683509
12	14	0.176675	0.678976
13	15	0.153288	0.684770
14	16	0.134854	0.689279
15	17	0.116457	0.686119
16	18	0.101223	0.687730
17	19	0.088058	0.696002
18	20	0.078853	0.692167
19	21	0.070237	0.696267
20	22	0.063051	0.676114
21	23	0.057355	0.682093
22	24	0.052201	0.696307
23	25	0.046577	0.692207
24	26	0.043358	0.687710
25	27	0.039717	0.679246
26	28	0.036608	0.693756
27	29	0.034266	0.676494
28	30	0.031792	0.679196
29	31	0.030006	0.663321
30	32	0.028043	0.685794
31	33	0.026472	0.671145
32	34	0.024800	0.674231
33	35	0.022525	0.668239
34	36	0.021614	0.680029
35	37	0.020458	0.676666
36	38	0.020015	0.658986
37	39	0.018462	0.659457
38	40	0.017498	0.645387
39	41	0.017105	0.660696
40	42	0.016781	0.670943
41	43	0.015484	0.679036
42	44	0.014629	0.662532
43	45	0.013908	0.669501
44	46	0.013552	0.648416
45	47	0.012577	0.668504
46	48	0.012430	0.668933
47	49	0.011950	0.655784

The graphs below show the values plotted. There is a graph for the silhouette score and a graph for the inertia.

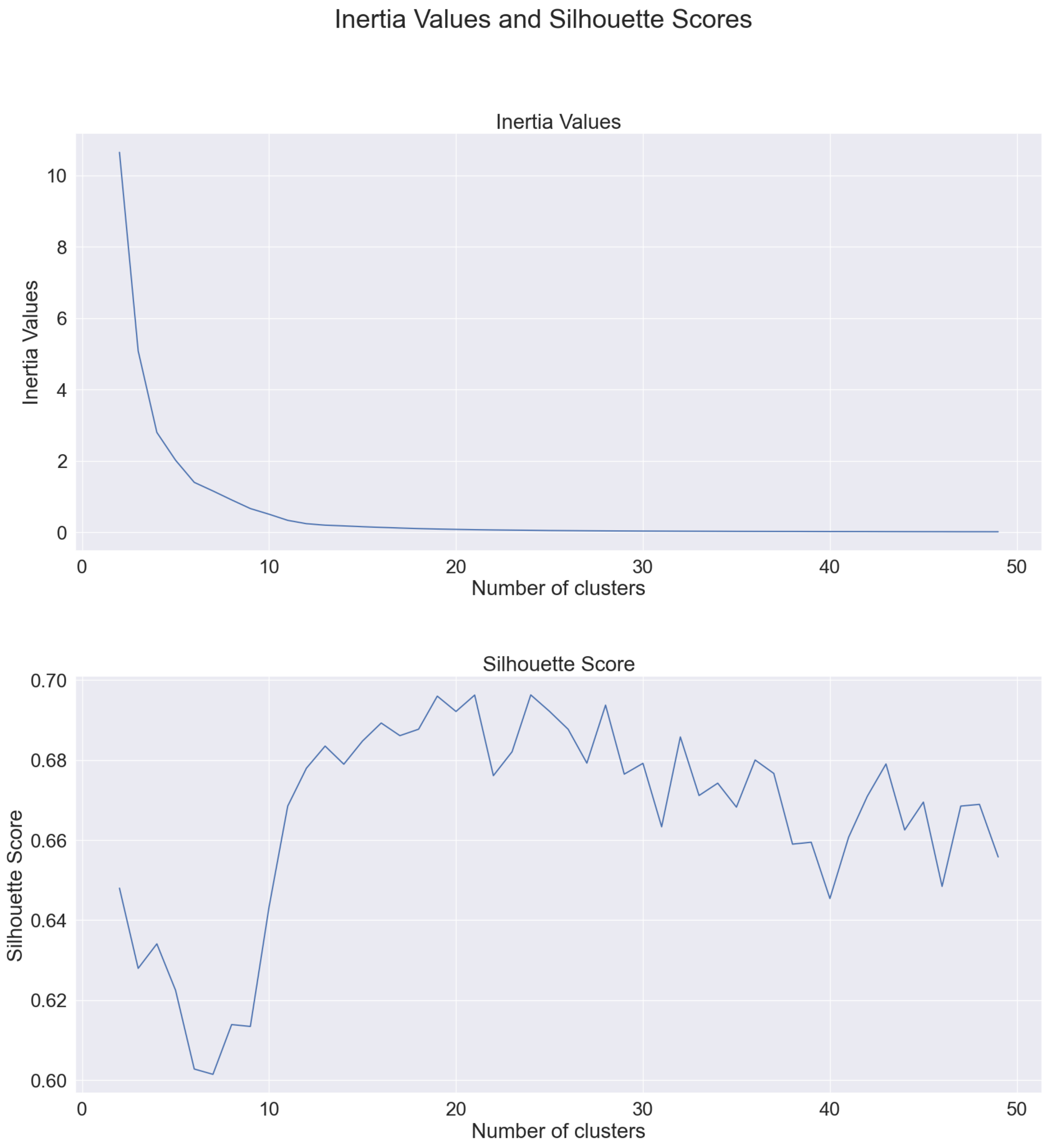
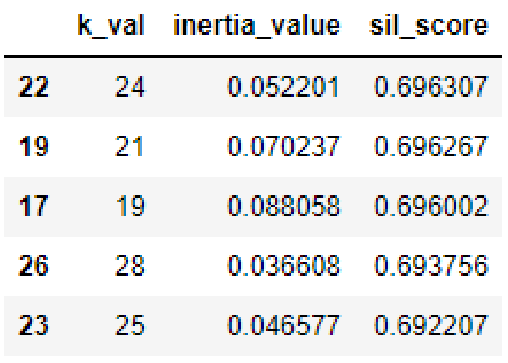


Image 2: The silhouette and inertia values.

Looking at these values by sorting them yielded the following table.



	k_val	inertia_value	sil_score
22	24	0.052201	0.696307
19	21	0.070237	0.696267
17	19	0.088058	0.696002
26	28	0.036608	0.693756
23	25	0.046577	0.692207

The model was created based on the scores that can be found in the first row of the table. The model yielded the following visual.

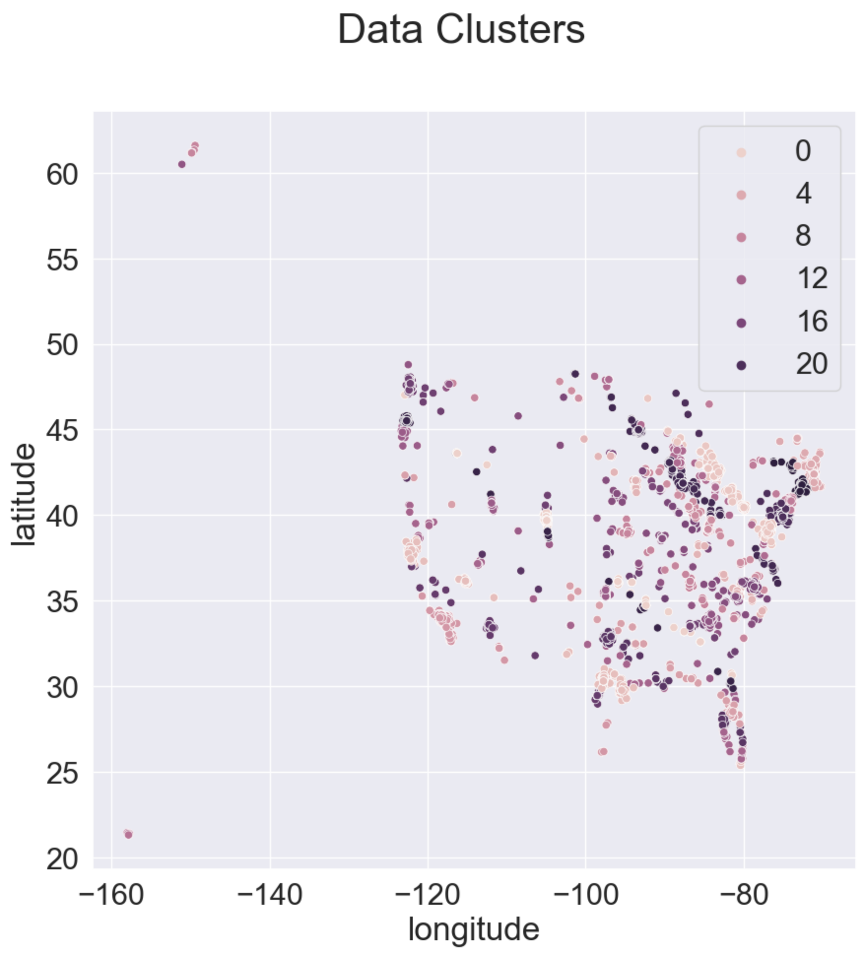
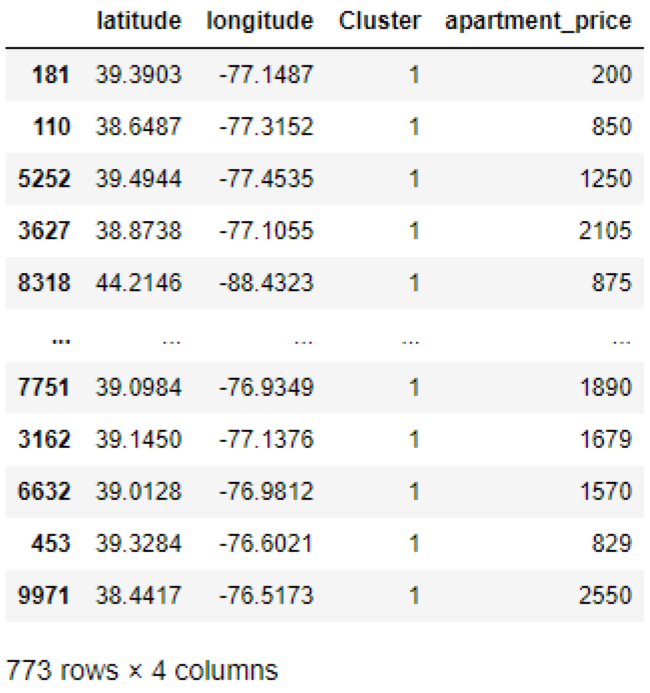


Image 3: The clusters of the final model.

There are a few things to take note of in this visual. The data that was used to create this graph was not the normalized data. This was to make sure that the data still represented the rough outline of the United States. An example of the data that was used is shown below.



	latitude	longitude	Cluster	apartment_price
181	39.3903	-77.1487	1	200
110	38.6487	-77.3152	1	850
5252	39.4944	-77.4535	1	1250
3627	38.8738	-77.1055	1	2105
8318	44.2146	-88.4323	1	875
...	...	...	...	...
7751	39.0984	-76.9349	1	1890
3162	39.1450	-77.1376	1	1679
6632	39.0128	-76.9812	1	1570
453	39.3284	-76.6021	1	829
9971	38.4417	-76.5173	1	2550

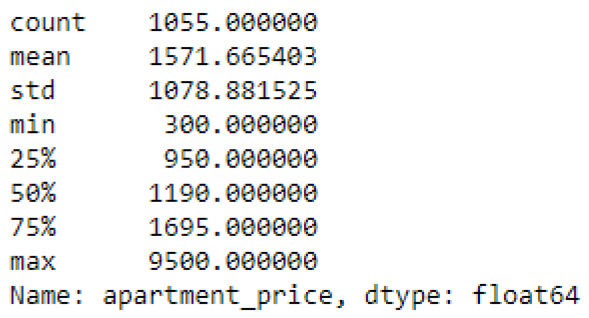
773 rows × 4 columns

After the creation of the model was created it was necessary to look at the statistics for some of the clusters. There will be only five clusters looked at. These clusters will be the five largest or the ones that contain the most observations.

Outline of the Findings

Once the five largest clusters were found. These clusters underwent some statistical analysis. This analysis looked at the mean, min, max, and other statistics that can be ascertained by Python's describe function. There are boxplots for each of these clusters in the Appendix which is found at the end of this document.

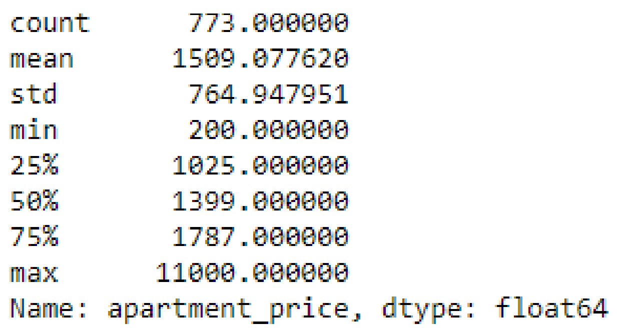
The first cluster exhibited a very large standard deviation in the prices of the apartments. The cluster had a standard deviation of ~$1078.88 and a wide range of values in the cluster. The minimum for the cluster was $300 and the maximum was $9,500. The other statistics can be seen in the table below.



```
count    1055.000000
mean     1571.665403
std      1078.881525
min       300.000000
25%       950.000000
50%      1190.000000
75%      1695.000000
max      9500.000000
Name: apartment_price, dtype: float64
```

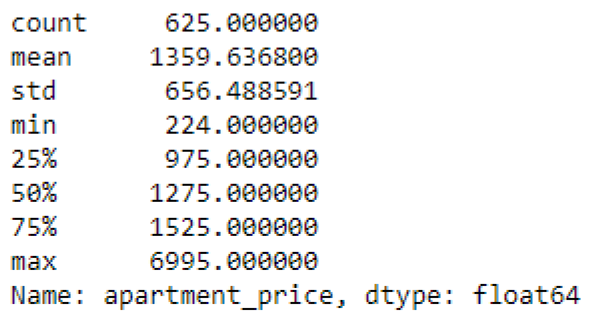
The next cluster it was found to have a very similar mean to the first cluster discussed, but the standard deviation was not as severe. This could be interpreted that the clusters are more closely related to the price. The standard deviation was ~$764.95. This cluster did have a max that was quite high relative to the other data in the cluster. The maximum was $11,000. Also in this cluster, the minimum was low. It was only $200.

What implications could this spread of values imply? Without further analysis of what is actually in the cluster. It is hard to speculate as to what the inclusion of these outliers had on the cluster and how it was created. This should be inspected in the future to see what the exact composition of the cluster is. The table shows the other statistics that were given for the cluster.



```
count      773.000000
mean      1509.077620
std        764.947951
min        200.000000
25%       1025.000000
50%       1399.000000
75%       1787.000000
max      11000.000000
Name: apartment_price, dtype: float64
```

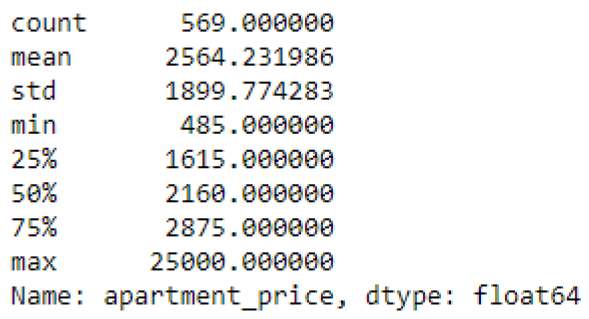
In the third cluster, the standard deviation was even better. It was ~$656.49. With a minimum of $224 and a maximum of $6995.00. Nothing else can be deduced from the statistics of the cluster. The table shows the rest of the statistics.



```
count     625.000000
mean     1359.636800
std       656.488591
min       224.000000
25%       975.000000
50%      1275.000000
75%      1525.000000
max      6995.000000
Name: apartment_price, dtype: float64
```

The fourth cluster is one of the most interesting. It had the most deviation in terms of the price of the apartment. The standard deviation was ~$1899.77. This was the highest of the clusters that were examined. This cluster is the cluster with the highest maximum apartment price. The maximum was $25,000.

This cluster would be worth looking into to see how this value is skewing the statistics for the cluster. This value may be enough to alter how the cluster was formed. It might be beneficial to look at where this cluster is located. This will be looked at later in the document. The statistics for the cluster are shown below.



```
count      569.000000
mean      2564.231986
std       1899.774283
min        485.000000
25%       1615.000000
50%       2160.000000
75%       2875.000000
max      25000.000000
Name: apartment_price, dtype: float64
```

In the last cluster, it had the following statistics.

```
count      497.000000
mean      1277.038229
std        521.655496
min        390.000000
25%        930.000000
50%       1200.000000
75%       1515.000000
max       6900.000000
Name: apartment_price, dtype: float64
```

This cluster had the lowest standard deviation out of the ones that were looked at. It had a maximum that was comparable to the third cluster.

Looking at the clusters in relation to each other we get the following scatterplot.

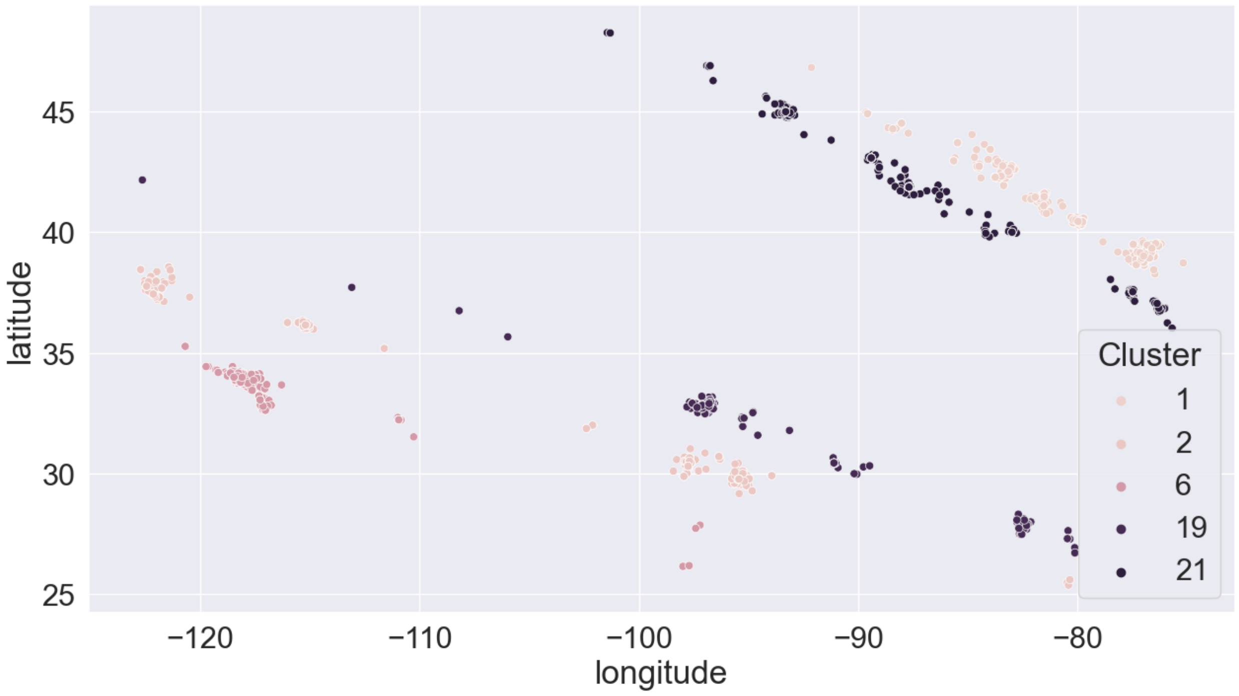


Image 4: Scatter plot of all the clusters.

Seeing the five largest clusters in this manner we can see that there are distinct bands for the clusters. These clusters are not or may not be spherical. This could mean the K-means model is not effective in using these features that were the basis of the model. Another model may be to create a more appropriate model for cluster data based on the desired features.

To explore the clusters a little further. It might be beneficial to look at the clusters as they relate to where they are located in the continental United States. Using the **geopandas** library it is possible to visualize these clusters in this manner. The map below shows the clusters and their position in the United States.

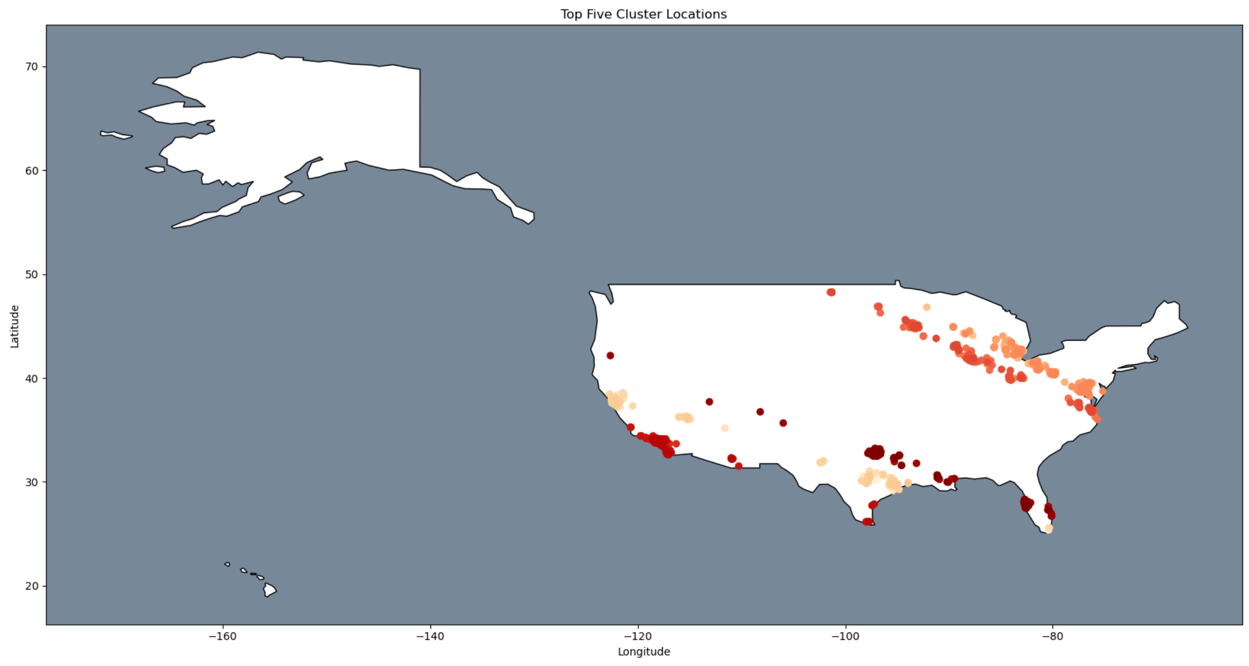


Image 5: Geopandas plot of the five clusters.

The legend does not appear[1], the clusters are the following (starting from bottom left):

- 6 – the red color
- 2 – is the light orange
- 19 – is the burnt red color
- 21 – dark orange
- 1 – medium orange

We can now see the location of the cluster that had the highest value for apartment rent. This is cluster 6 (red color). We can see that this cluster is located in southern California. The excessive price seems to make more sense when the cluster is visualized in this manner.

[1] The code to produce this legend would not work.

Limitations of the Techniques and Tools

Using the features that were used to implement the model imposed a few limitations on the model. Including the outliers in the model may have spread clusters in a manner that probably would not have happened if these outliers were cleaned from the data in the initial stages of the data preparation.

While there were 10,000 samples in the dataset. It might be beneficial to try to use the larger dataset for the creation of the model. It would allow the model to see the more possible variance in the observations of the data. There is one caveat to know that simply utilizing the larger dataset does not necessarily mean that the data will cluster better. This data will more than likely exhibit the same type of banding that was exhibited in the smaller dataset.

The clusters themselves show a large standard deviation. It is safe to assume that this is true for the other clusters that were not analyzed. Remedies for this will discussed in a subsequent section of the document.

The Python libraries that were used performed as expected and this was expected. The only K-means while it was able to create clusters that met the 0.60 silhouette score. The model itself would be pressed to adequately cluster the price of the apartment. There was too much leeway in the prices that were observed in the clusters that were created.

Summary of Proposed Actions

The model that was created while it met the objective of creating a model that adhered to a silhouette value of at least 0.60. It achieved a score of 0.69. It would be prudent to develop another model that will better group the data based on the given parameters. While it will group the data using the current features. It is not possible to use this model to figure out the price of the apartment based on the latitude and the longitude of the apartment.

This model does a good job of grouping if the data is to be used regarding its geospatial location. Using the grouped data along with the geopandas seems to confirm this assumption.

It might be beneficial to look into using other parameters other than the ones that were chosen to be used in this initial model. It might be beneficial to use things like square footage or other features that are included in the dataset. Using the other features could yield better and more reliable clusters. These newer clusters can be better defined with less variance among the observations within the clusters.

Also, it is possible to remove the outliers that were included and see what this exclusion has on the model. It is supposed that this removal of the outliers will have little or no effect on the model and the clusters that it created.

In regards to the model, it may be possible to find another clustering algorithm that may be better suited for the type of data that was utilized in the model. There are other clustering algorithms and with a little research, it may be possible that there is a better model for the data that was used here.

Benefits of the Study

While the model did not work as intended. This model did prove that it is possible to create a clustering model that met the hypothesis that was suggested earlier. This benefit was partially negated by the fact that clusters did not conform to the traditional notion of "spherical" clusters.   This model will excel and the benefits will be reaped if the model is used to cluster data based on the location of the data. This will be the best use of the model and something that can be useful to look at the data based on the latitude and longitude of the data.

If there limitations of the model that were discussed in the previous section are addressed it is possible to get this model to perform in a manner that will more adequately answer the question that was proposed at the beginning of the document.

References

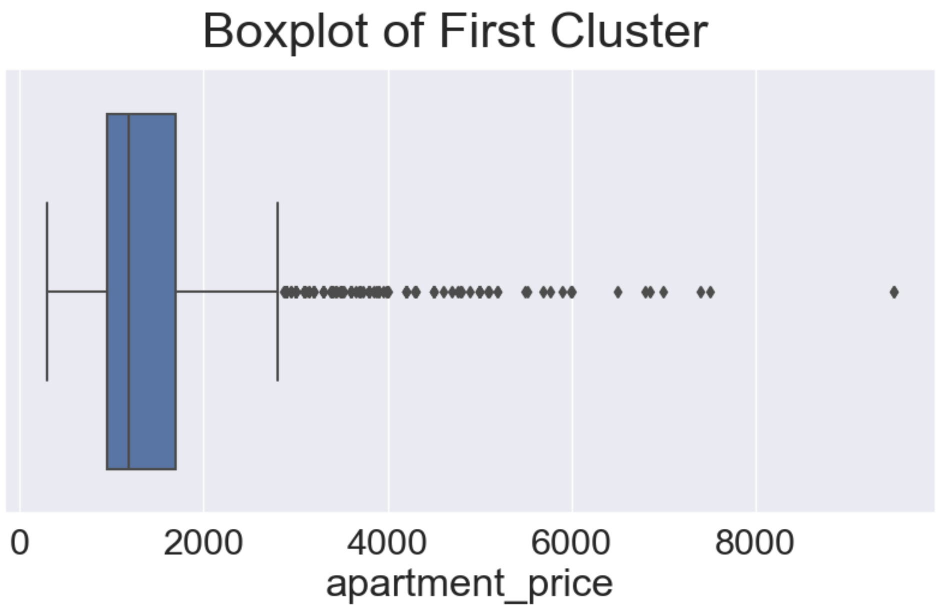
In-text citations:

Mayo, M. (2022, May 13). *Centroid Initialization Methods for k-means Clustering – KDnuggets*. KDnuggets. Retrieved February 18, 2024, from https://www.kdnuggets.com/2020/06/centroid-initialization-k-means-clustering.html

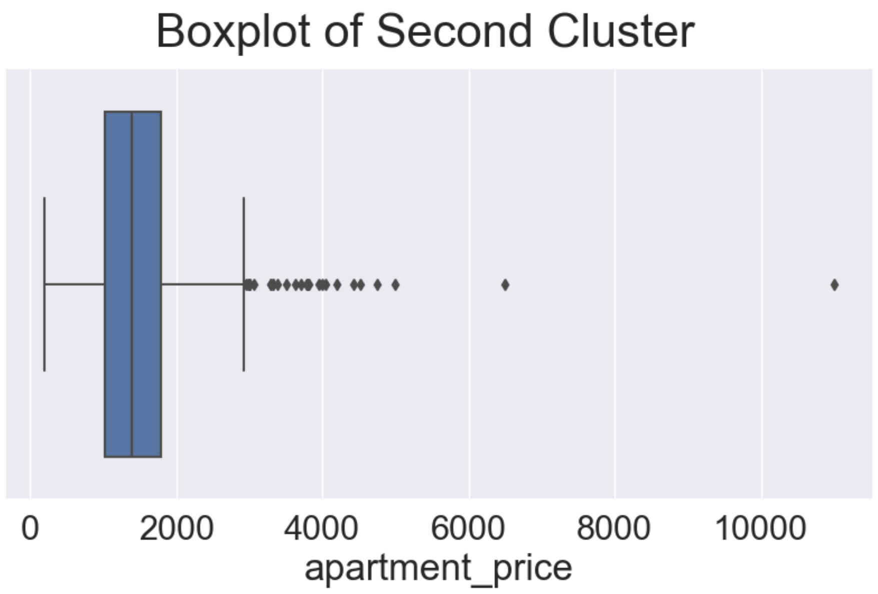
VLFeat. (n.d.). *VLFEAT – Documentation > C API*. Retrieved February 18, 2024, from https://www.vlfeat.org/api/kmeans-fundamentals.html

Appendix

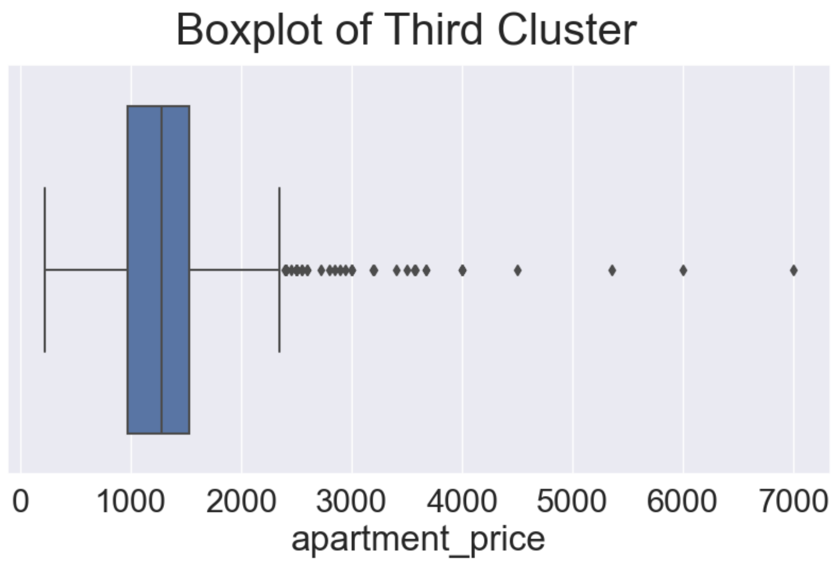
Boxplots and other plots will be found in this section of the document.



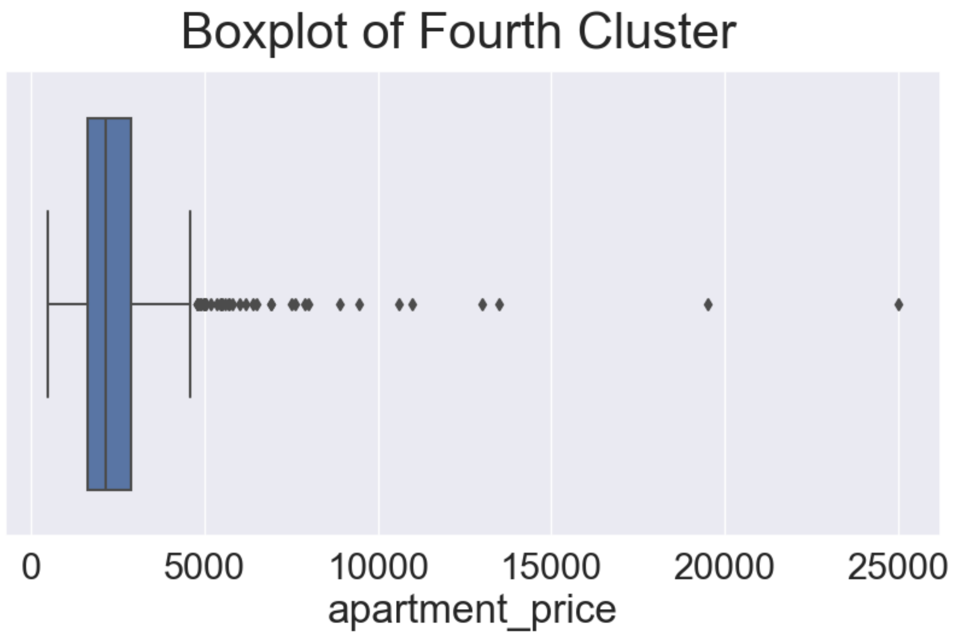
Boxplot 1: Boxplot of the first cluster.



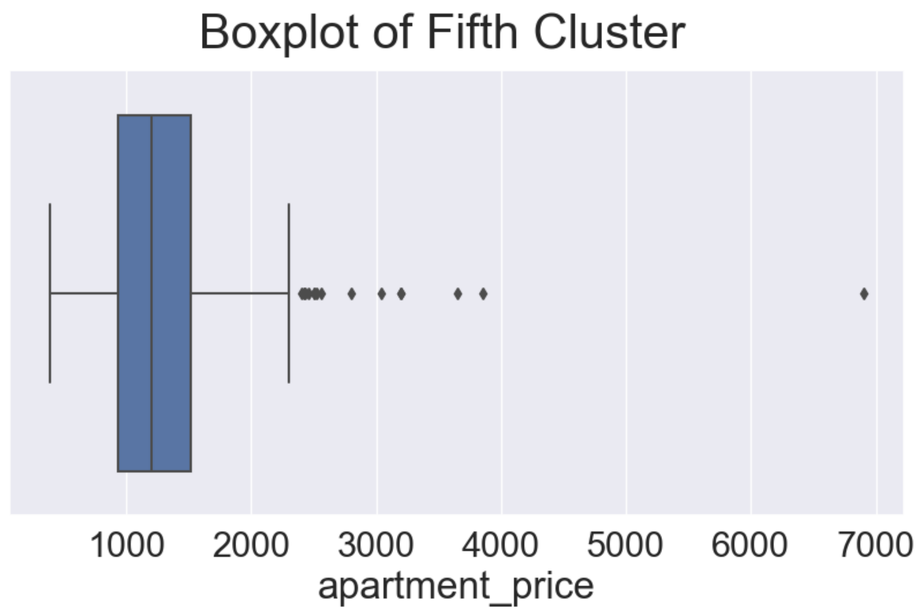
Boxplot 2: Boxplot of the second cluster.



Boxplot 3: Boxplot of the third cluster.



Boxplot 4: Boxplot of the fourth cluster.



Boxplot 5: Boxplot of the fifth cluster.