PREDICTION AND CLUSTERING OF CONDIMINIUM SALES IN TORONTO USING SOCIO-ECONOMIC VARIABLES

MSCI 623: BIG DATA ANALYTICS

STUDENT 1: MUKUND VEMURI
UW ID #: 20328213

STUDENT 2: SUMIT SHARMA
UW ID #: 20767923

1. ABSTRACT

This report outlines the analysis conducted on a dataset scraped from a real estate MLS site. The motivation for the data collection was to predict condominium prices in the city of Toronto using the features of the condo and socio-economic variables pertaining to the neighbourhood the condo is located in. The idea behind this is that the socio-economic variables can explain the variance in sale price better and can provide interesting insight after performing clustering on it. The data for socio economic variables was obtained from the city of Toronto Open Data Initiative on wellbeing economics and safety. The data was then joined with the scraped data from the MLS site by using the neighbourhood name. The data was then filtered on condominiums and the sale price was log transformed to correct for positive skew. Pearson correlation was used to determine appropriate features. The selected features along with the others were used to perform a comparative regression analysis on the data. The resulting analysis showed buyers give more importance to the condos features than socio-economic indicators of the neighbourhood. The features that account for majority of the variance are yearly taxes, number of businesses in the neighbourhood, square footage of the condo, number of break and enters in the neighbourhood, parking places, distance to a park, number of social assistance recipients in the neighbourhood and distance to the nearest transit station. This shows that some socio-economic variables do account for the variance in the data. A clustering analysis was also performed and resulted in three clusters for different kinds of buyers and investors. The first cluster was for single, working professionals looking to be close to amenities. The second was for families looking to be close to downtown but require more space. The last cluster represented retirees and other low-income groups who live further away from major business areas and transit. Future work and improvements on the dataset were also investigated. The one significant improvement to be made would be to use socio-economic data that is not highly correlated and represent per capita data for only residential households. This should provide a more accurate picture of the forces accounting for the variance in the sale price.

2. INTRODUCTION

The Toronto real estate market is a popular and volatile market. In the last few years, the market has seen an increase in demand and has become one of the most expensive housing markets in the world along with Sydney, Vancouver, Hong Kong and San Francisco.

The real-estate market's popularity is aided by the large inventory and choice of real estate property available in the metropolitan. As with other metropolitans, one type of property that is popular in Toronto is Condominiums and the real-estate market has a significant volume of condominiums that are available and sold on the market.

An interesting aspect of the sale and purchase of such condominiums is how the different features of a condominium determine the sale price of it on the market. These include, but are not limited to - number of bedrooms, number of bathrooms, square footage, number of parking spaces, the distance from schools, transit, parks etc. We can take this idea further and consider how other aspects can affect the prices of condominiums. One such aspect is the neighbourhood the condominium is located in and how its socio-economic characteristics affect the price and duration of its listing. A simple example of such effect is that areas with more crime and fewer number of businesses or schools will take longer to sell and will sell for less than those that are socio-economically better.

Similar to other markets, sellers would like to maximize their profits and ensure that their property is sold on the market as soon as possible. While buyers would like to bid the lowest possible amount for a particular property and know after how many days on the market to bid for that amount. Thus, knowing the accurate value of sale price of such condominiums is of considerable value to both players in the market.

The ideas and value of predictions stated above provided an impetus for our analysis and hypotheses. Two hypotheses that were of interest for the study were as follows:

1. Condominium sale prices are affected by socio-economic variables of the neighbourhood the condo is located in. This effect can allow us to build a more accurate sale price prediction model for the sale of condo.

2. Condominiums sold in the market can be clustered by their socio-economic variables and there is uniformity or low standard deviation in the sale price of the houses within a socio-economic cluster.

Socio-economic variables pertain to the characteristics of a neighbourhood that the condo is located in. Some examples of the socio-economic variables are number of businesses in the neighbourhood, the debt-risk score of the neighbourhood, vehicle thefts, fires and alarms reported, distance to schools, distance to transit and distance to the city center.

While a buyer or seller may not think about these aspects when purchasing or selling condo, it is interesting to see how these "invisible forces" can affect the prices.

While additional variables do not always increase the accuracy of a model, we suspect that these socio-economic variables are a good choice of features that can additionally describe the variance in the sale price of a condo. Based on this, we hypothesize that we can come up with a more accurate model.

The descriptive analysis of clusters can also tell potential buyers and sellers about houses in socio-economic clusters that may be considered undesirable but have lower or higher sale prices.

3. RELATED WORK

Given that the real estate market is an important economic indicator for countries, and also a viable investment option for several people, several prior works have been conducted in this area in the past.

In [1], the author seeks to predict the loss in the value of the property prediction based on over and underestimation using the predictive mean. The author limits the study to only one-story houses in the Aspen neighbourhood of Calgary. While many of the feature variables are similar to our project, the author does not consider comparing economic, welfare or criminal activity in various neighbourhoods of the houses being assessed to develop a more accurate model.

In [2], the author makes use of fuzzy neural networks to determine the price of the houses. While there is use of multi-variate regression analysis as similar to ours, it is not considering the crime and economic characteristics of a neighbourhood. Additionally, the issue of overfitting caused by neural networks has not been discussed in this publication.

[3] performs a comparison of various methods used in the past for conducting prediction of housing prices. The author performs analysis using fuzzy-least-squares regression for the prediction of house prices. The publication also mentions the ability of multi-variate-regression analysis to outperform most of the other methods in price estimation in complexity and accuracy. While our study does not conduct a comparison of the different approaches and modelling techniques involved in predictive analytics, we are expecting to develop a fairly accurate model based on using the simple techniques of least-squares-multi-variate regression. The publication also uses different variables and classifies the location qualitatively when preparing the model.

In [4], the author conducts a comparison of different machine learning algorithms on real-estate data. The author makes use of decision trees, linear regression and support vector machines for doing a comparison of the types of algorithms that can be employed to conduct a prediction of real estate prices. The method of comparison used is Nonparametric Wilcoxon signed-rank to evaluate the difference between ensembles and original models. The authors results show that only one model is not enough for performing an accurate prediction, a hybrid approach, however, is optimal. While we will not be performing a comparative analysis of different models, we will be using both prediction and descriptive analysis (clustering) to determine key insights in our datasets. The comparison of these two approaches for analyzing data should provide new insight in this area.

In [5], the author makes use of a support-vector machine to perform a prediction of prices. In specific, the author makes use of particle-swarm-optimization to determine the parameters of SVM. The difference in approach is the sample data, which is for a city in China. Since the dataset corresponds to a different country, the model does not use the same feature variables. This is because different factors determine the price of a property in different countries. As pointed out by the author, the feature variables considered are land price, development costs, political aspects and so on. The socio-economic aspects are not considered in this study as in our project. Finally, the use of SVM is more suited for classification than prediction.

4. DATASET

4.1 ACQUIRING THE DATASET

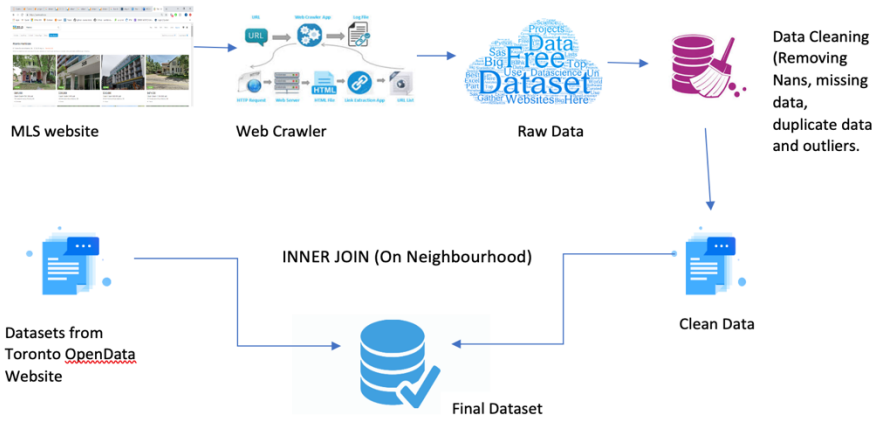


Figure 1: Overview of data scraping and dataset creation

The figure above shows an overview of the data collection procedure.

The dataset was obtained by scraping an MLS (Multiple-Listing Site) site for the information that was assumed to be pertinent to our analysis - price prediction using socio-economic variables. The scraping of the data allowed us to also document the neighbourhood of the house sold, along with the features of the house.

The socio-economic statistics for a neighbourhood were obtained from the City of Toronto's Open Data Initiative. This data was obtained as two sets - one for safety associated statistics (thefts, assaults etc.) and another for economic associated statistics (businesses, social assistance recipients etc.). These were joined together on the neighbourhood name to create one dataset for socio economic variables.

The above two datasets were joined on their Neighbourhood column. Where neighbourhood names did not match in either left or right datasets they were not considered in the final dataset (i.e. we performed only an Inner Join).

4.2 MISSING DATA

Where values for features desired could not be acquired/scraped, they were appropriately documented (as "error" or -1 for text and numeric data respectively) and later on removed from the dataset. Such data resulted as missing because relevant information was not provided, or a listing was used as click bait for advertising of a brokerage etc. Hence, the removal of such data did not affect the quality of the data.

The removal still left us with a dataset of ~3000 rows which we deemed sufficient for this analysis.

4.3 DATA SCHEMA AND SAMPLE VALUES

Table 1: Dataset schema and sample values (personally identifiable information removed)

Column Name	Description	Sample Value
city	City in which the property is located	Toronto
province	Province in which property is located	ON
neighbourhood	Neighbourhood area of the property	York University Heights
sale_price	Sale Price of the property (CAD$)	780000
list_price	List Price of the property (CAD$)	819000
bedrooms	No. of bedrooms included	4
bathrooms	Number of bathrooms included	5
square_feet	Size of the property in square feet	2500
days_on_market	Number of days the property was on market, before being sold	37
house_type	Type of the house	Semi-Detached
parking_places	Number of parking places	0
yearly_taxes	Yearly taxes paid on the property, in CAD$.	3828
school_distance	Distance from the nearest school in the neighbourhood in Km.	0.75
transit_distance	Distance from the nearest transit stop in Km.	5.26
medical_care_distance	Distance from the nearest medical care center in the neighbourhood in Km.	0.84
city_center_distance	Distance from the nearest major city center in Km.	10.75
park_distance	Distance from the nearest park in the neighbourhood in Km.	0.85
businesses	Number of businesses running in the neighbourhood	2643
child_care_spaces	The number of spaces for children from birth to senior kindergarten and the number of spaces for school age children grade 1 and up	156
debt_risk_score	TransUnion Canada index value that indicates the likelihood of missing three consecutive loan payments. Low-value scores (<707) indicate a High Risk of missing 3 consecutive loan payments; High-value scores (769+) indicate a low risk.	698
income_gini_coeff	Gini coefficients are measures of income inequality, or in other words how different are the incomes of the richest in relation to the poorest.	0.427
local_employment	Total local employment (jobs), persons aged 15+ years.	42885
social_assistance_recipients	Count of recipients (members) of aid qualifying for Ontario Works, Temporary Care (OW), Ontario Disability Support Program (ODSP) or Special Assistance (ODSP/OW) programs in 2011. Count is by Members, not Cases.	4720
break_enters	Number of break and enter incidents reported in the neighbourhood over one year	150
fires_and_alarms	Number of Fires and fire alarms incidents reported in the neighbourhood over one year	798
sexual_assaults	Number of sexual assaults reported in the neighbourhood over one year	49
thefts	Number of thefts reported in the neighbourhood over one year	16
vehicle_thefts	Number vehicle thefts reported in the neighbourhood over one year	122
total_major_crime_incidents	Total number of major crime incidents reported in the neighbourhood over one year	776

5. EXPLORATORY DATA ANALYSIS

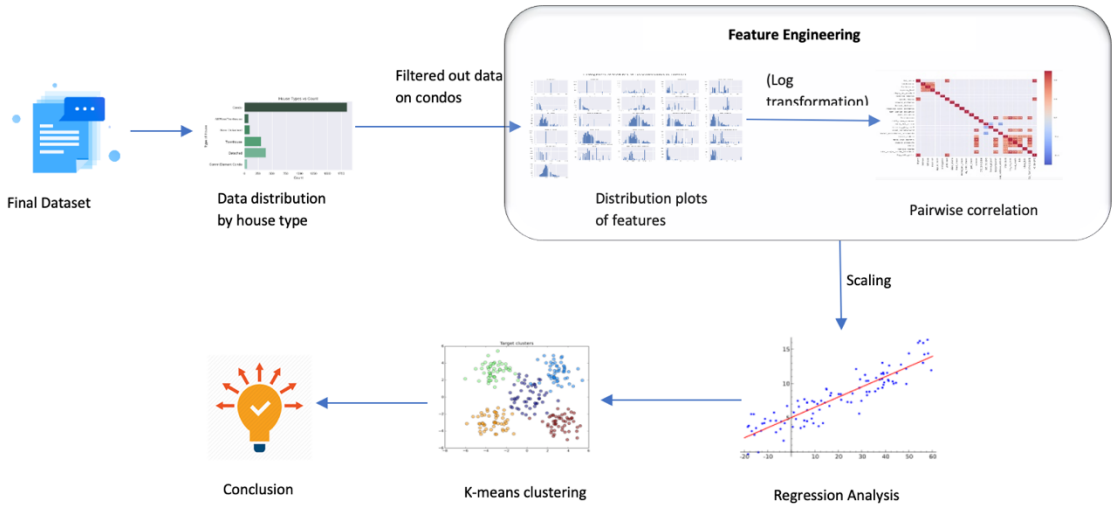


Figure 2: Flowchart of data analysis conducted on dataset

The figure above summarizes the overall analysis conducted for the resulting dataset. The sections that follow describe each step in the diagram in further detail.

5.1 ROW COUNT

Row Count of the dataset is 2779. This is the number of rows that were obtained by scraping the site.

5.2 VISUALIZING THE DATA

The following figure shows the type of each house in the dataset.

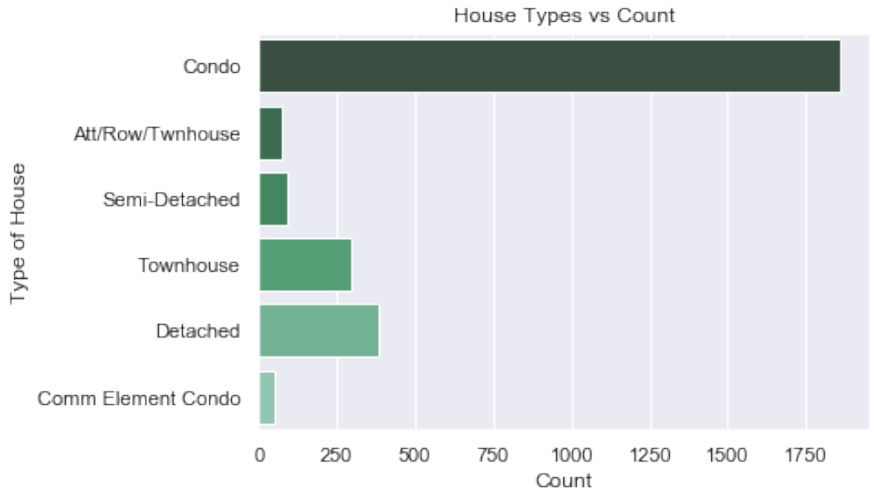


Figure 3: Distribution of type of houses in dataset

As can be seen from the figure above, Condominiums make up a large number of the house types in the dataset. This is expected of the Toronto housing market and also helps in validating our hypothesis which was focused on sale price of condominiums.

5.3 GEOMAPPING THE DATA

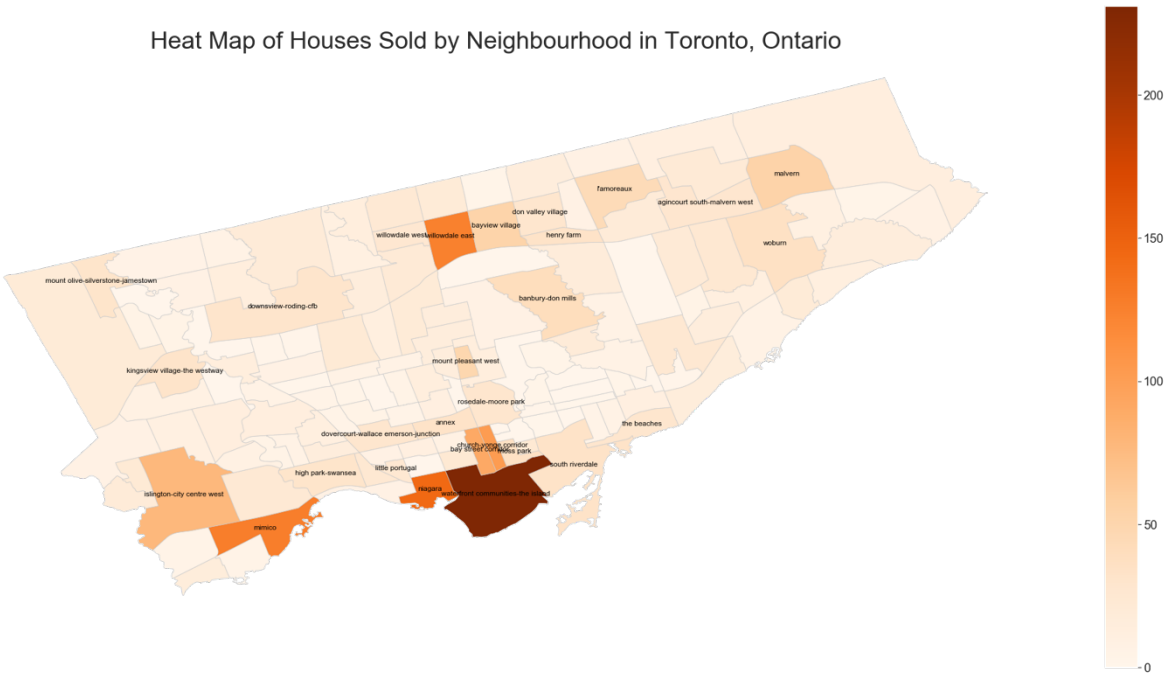


Figure 4: Heatmap of house sales in Toronto based on dataset under analysis.

As can be seen from the heat map above, the downtown core sees large sale activity. This is an expected result, considering the dataset consists of a large number of condominiums which are usually located in the downtown core. Further, the downtown core is a popular location to reside for new and old residents of the city. Finally, the Mimico and Willowdale-East neighbourhoods also see large house sale activity.

6. FEATURE ENGINEERING

6.1 REMOVING OUTLIERS

There were a majority of condominiums in the dataset. Given this observation, we used the data to make more accurate predictions on condominiums, as opposed to all other types of houses which had a small amount of data and may not have led to a very accurate model. This also helped further validate our hypothesis.

Additionally, the data had a few types that were labelled as 'Comm Element Condo' and 'Det Condo'. As these house types are similar to a condominium, we converted them to 'Condo' to get a slightly bigger set of data associated with just Condominiums. This transformation is shown in the table below.

Table 2: Conversion of Comm Element Condo and Det Condo to Condo. Table shows before and after house type counts

House Type	Original	After Transforming
Condo	1860	1915
Detached	385	385
Townhouse	298	298
Semi-Detached	94	94
Att/Row/Twnhouse	75	75
Comm Element Condo	51	-
Co-Ownership Apt	5	5
Det Condo	4	-
Triplex	2	2
Link	2	2
Duplex	1	1
Co-Op Apt	1	1
Multiplex	1	1

After transforming the data, we looked at the descriptive statistics of it to determine if the new dataset was suitable for analysis.

Table 3: Descriptive statistics of sale price of condominiums in dataset

count	1915
mean	627137.4
std	318574.9
min	47500
25%	460000
50%	560000
75%	690000
max	6500000

The average price of a condominium sold in Toronto is ~ CAD$ 630,000. This was an expected value given the current market conditions.

Additionally, seeing something like the sale price distribution would also help.

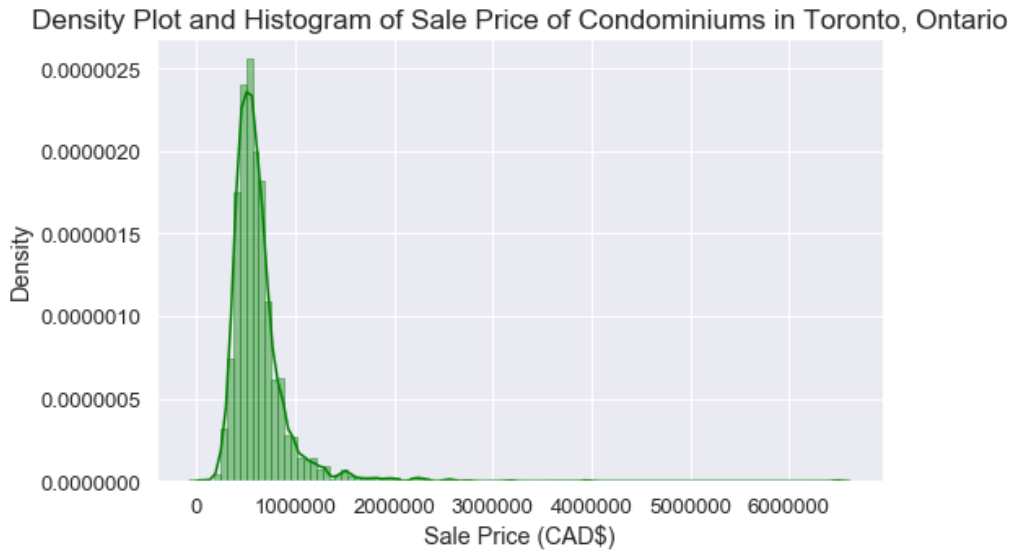


Figure 5: Density plot showing distribution of sale price of condominiums in dataset

The sale price was normally distributed below CAD 1.5 Million Dollars. This is much more than the average price we found before. Since this data is related to condominiums sold, it is indicative that there were some ultra-luxury condominiums that were also present in the dataset. The sale of such condos is not affected so much by the socio-economic features of the neighbourhood, but rather by the fancy of luxury house buyers and sellers.

In a similar fashion, we see that the minimum price of the house sold is CAD 47,500. This is again much lower than the mean and does not indicate any further price determination by socio-economic characteristics of the neighbourhood.

Given the skew in the sale price, we took the log transformation of it and only considered the log of sale prices within 2 standard deviations of the mean sale price for getting a much more normally distributed outcome variable. This is shown in the figure below.

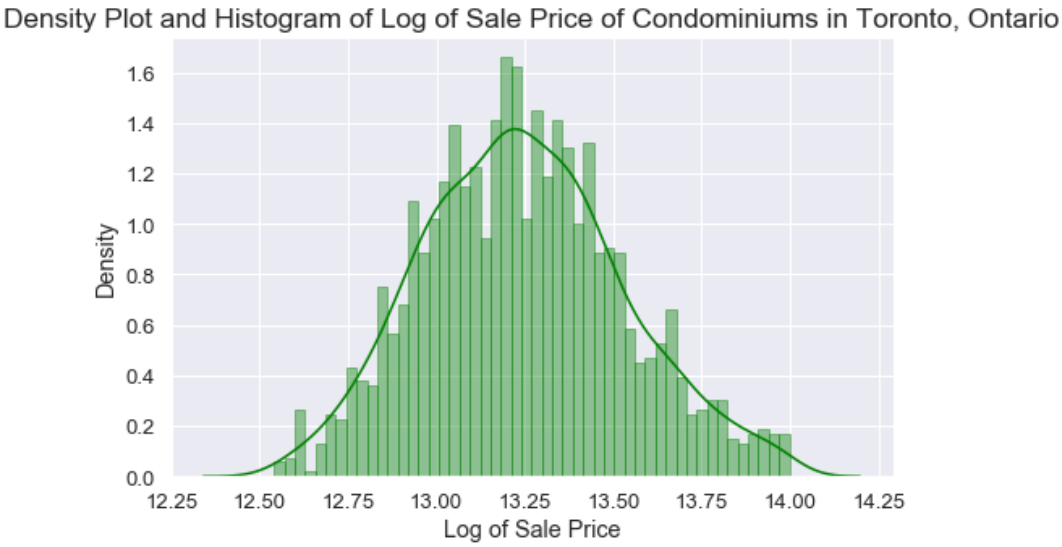


Figure 6: Distribution of condominium sale price after log transform of sale price

After normalizing the sale price, we were left with 1820 rows. We chose 2 standard deviations over 3, as the number of rows did not decrease by much but there was significant decrease in the skew of the data within 2 standard deviations. The final skew of sale price after transformation was 0.2 which was good for the analysis we wished to conduct.

The minimum price in this normalized dataset came out to be CAD$ 279,500. The maximum sale price was CAD$ 1.2 million. The Mean sale price was CAD 588,375. These values again made sense given the current market conditions and seemed reasonable to consider for further analysis. The percentiles were also more uniformly distributed. This is shown below.

Table 4: Descriptive statistics of sale price of condominiums in dataset after normalizing sale price

count	1820
mean	588375.977
std	173871.162
min	279500
25%	460000
50%	557250
75%	676000
max	1200000

It is important to note is that given our transformation, the new target variable was now the log of the sale price.

With the outliers removed from the sale price, we proceeded to observe the other feature variables and see if there were any outliers in them.

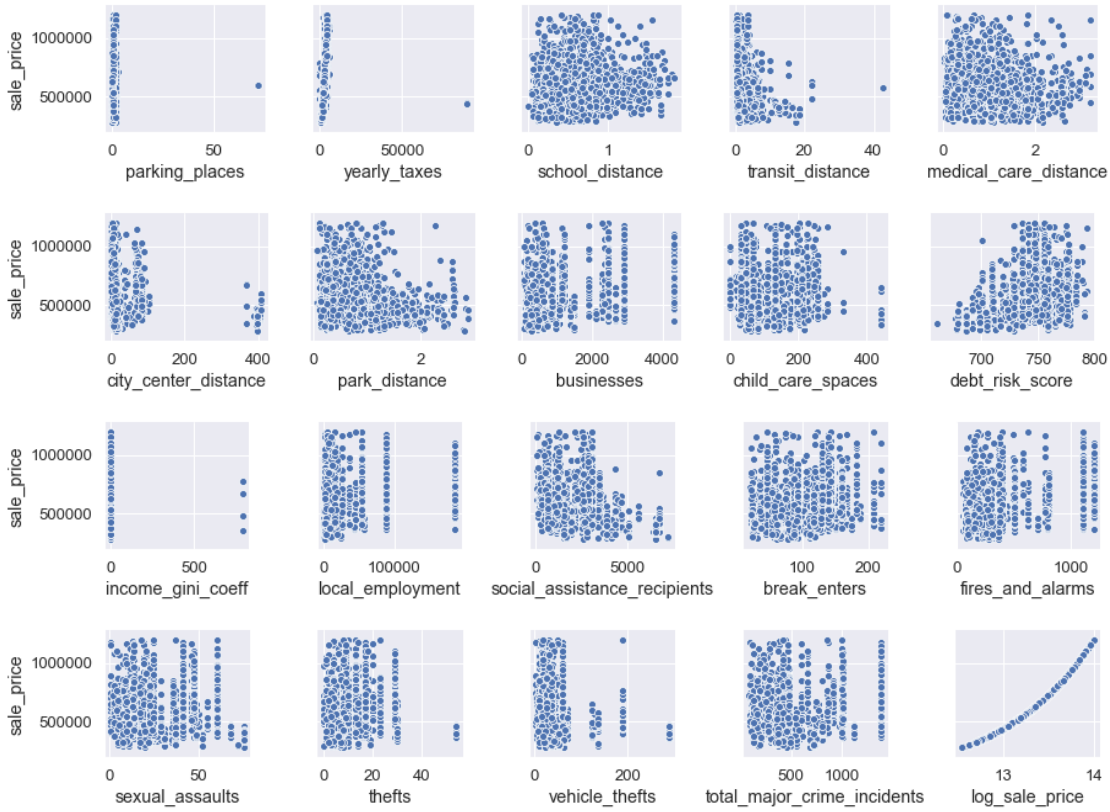


Figure 7: Pair-wise scatter plot of sale price vs all features. The plots show outliers in several of the features.

From the above plot of the feature variables against the sale price, it appears that there were outliers for parking places, yearly taxes, transit distance, city center distance, thefts and income Gini coefficient. These are summarized in the table below:

Table 5: List of outliers and their corresponding values in dataset

Column Name	Outlier Value
parking_places	20
yearly_taxes	25000
transit_distance	30
city_center_distance	200
thefts	40
income_gini_coeff	40

From the table above, it can be reasoned why these values are outliers. For instance, parking places being more than 3 in a house implies that the it is not a residential unit, but rather a commercial property. Further, having yearly taxes of CAD$ 25,000 does not make much sense for a residential unit. Based on the above table, we filtered out these values to get a more realistic dataset. The thefts and income Gini-coefficient values were deemed as outliers by looking at the graph and finding only a single point that represents those data values.

The resulting pair plots after removing the outliers were as shown below:

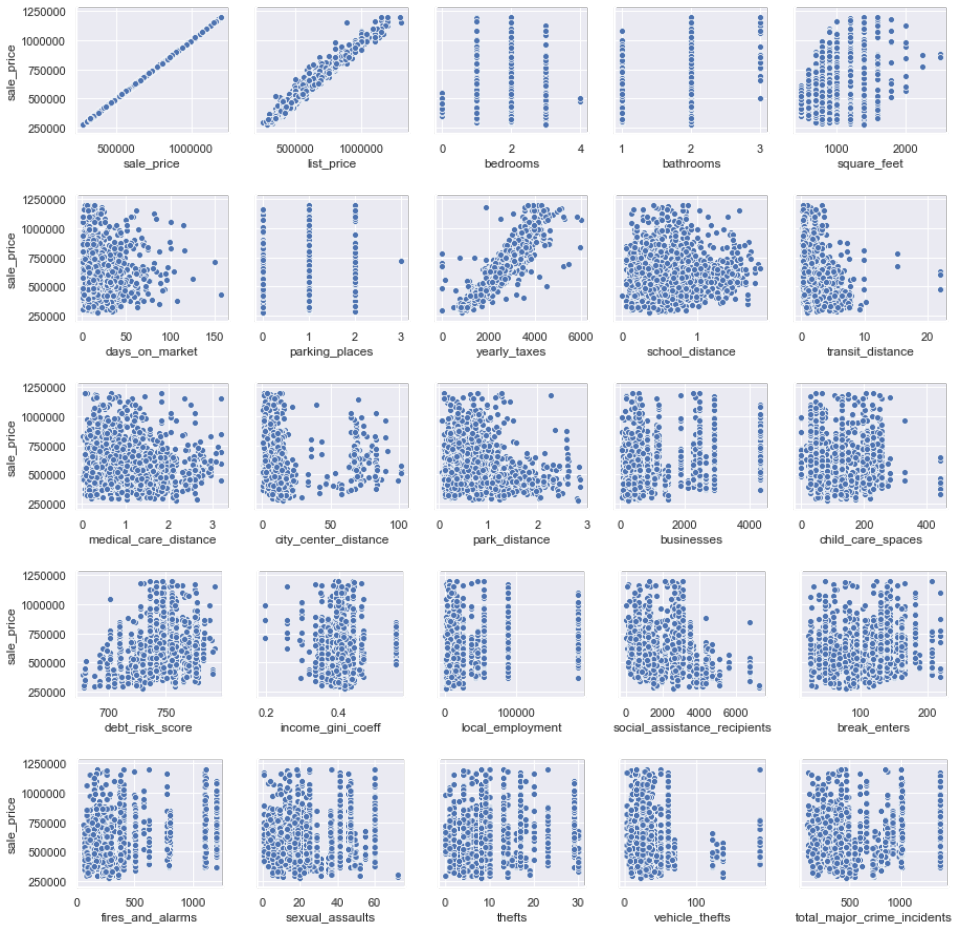


Figure 8: Pair-wise scatter plot of sale price vs all other features after removing outliers in all features.

We then decided to take a look at the histogram plots of each of the variables.

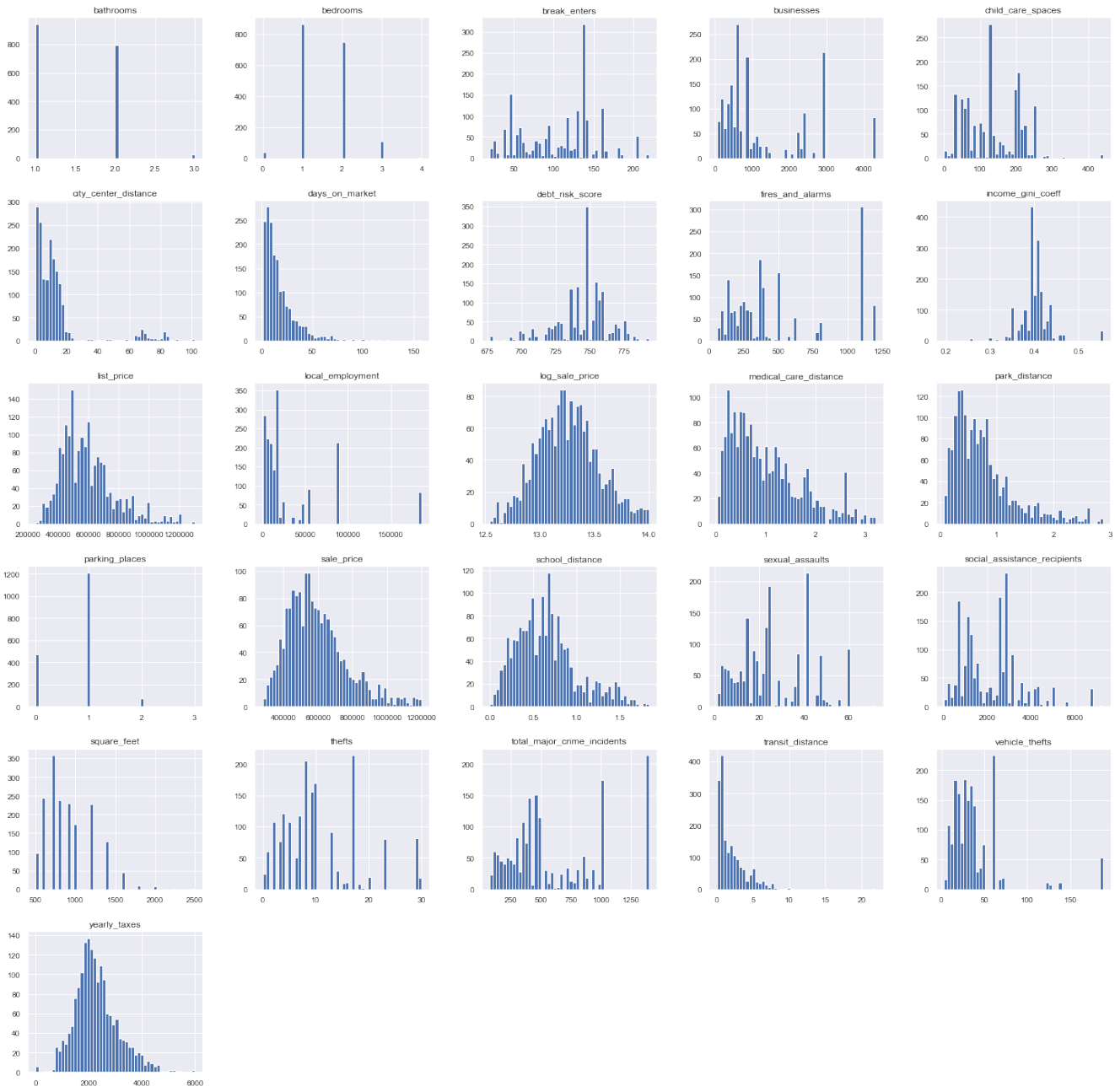


Figure 9: Histograms of all feature variables

From the histogram, it appears that yearly taxes, parks, medical care distance and school distance have the same distribution. Additionally, thefts and total_major_crime_incidents have a similar distribution. Finally, break and enters, income Gini-coefficient and the debt risk scores have similar histograms. This is indicative of some correlation among the features and the sale price as well.

After cleaning the dataset and removing the outliers, the following descriptive statistics resulted:

Table 6: Descriptive statistics of condominium sale price after removing outliers from all features in dataset

count	1760
mean	593363.89
std	173765.53
min	280000
25%	467125
50%	560000
75%	680000
max	1200000

After filtering the data, we see that the descriptive statistics did not change much. The resulting set appears reasonable for a further analysis.

6.2 PEARSON CORRELATION

Since we wanted to perform regression analysis on this data, it made sense to see the Pearson correlation and find those variables that are highly correlated with the outcome variable.

A table summarizing the pearson correlation values can be seen below.

Table 7: Pearson correlation values of feature variables against outcome variable (log of sale price)

Variable	Pearson Corr with log_sale_price
log_sale_price	1
list_price	0.96125
yearly_taxes	0.886556
bathrooms	0.373
fires_and_alarms	0.308304
debt_risk_score	0.271255
local_employment	0.265035
businesses	0.264677
square_feet	0.251013
total_major_crime_incidents	0.198109
bedrooms	0.170696
thefts	0.169478
sexual_assaults	0.131436
break_enters	0.125633
parking_places	0.109175
park_distance	-0.250178
social_assistance_recipients	-0.305269
transit_distance	-0.318992

Recall that the target variable was transformed to the log_sale_price.

We see that of the variables, yearly_taxes are strongly correlated with the log of the sale price.

The number of bathrooms, social assistance recipients and transit distance are weakly correlated.

Further, the debt risk score, local employment, business and park distance are also weakly correlated with their correlation scores being within the same range.

The signs of the correlation coefficients are also important. It is interesting to see that thefts, sexual assaults and break and enters have a positive sign. It indicates that there are more thefts, break and enters and sexual assaults in areas with higher home prices. This makes sense It would be more lucrative for thieves to steal in homes which cost more.

The distance from a park, transit and the number of social assistance recipients are negatively correlated, and this seems reasonable. The further you are from such amenities, the less value of the condominium.

Based on the correlation with the target variable we can see that most variables are only somewhat correlated with the log of the sale price. While this was not the best outcome of the pearson correlation calculation, we still considered these values for regression analysis. Our thinking here was that while individual variables may not explain the variance, it is possible for collective to.

6.3 PAIRWISE CORRELATION

In the previous section we determined the Pearson correlation coefficient against the target variable.

It also makes sense to calculate the pairwise correlation with each of the variables. This is useful in determining the features to select for the regression analysis. The pairwise correlations of highly correlated features are shown with different strengths of red and blue in the graph below.

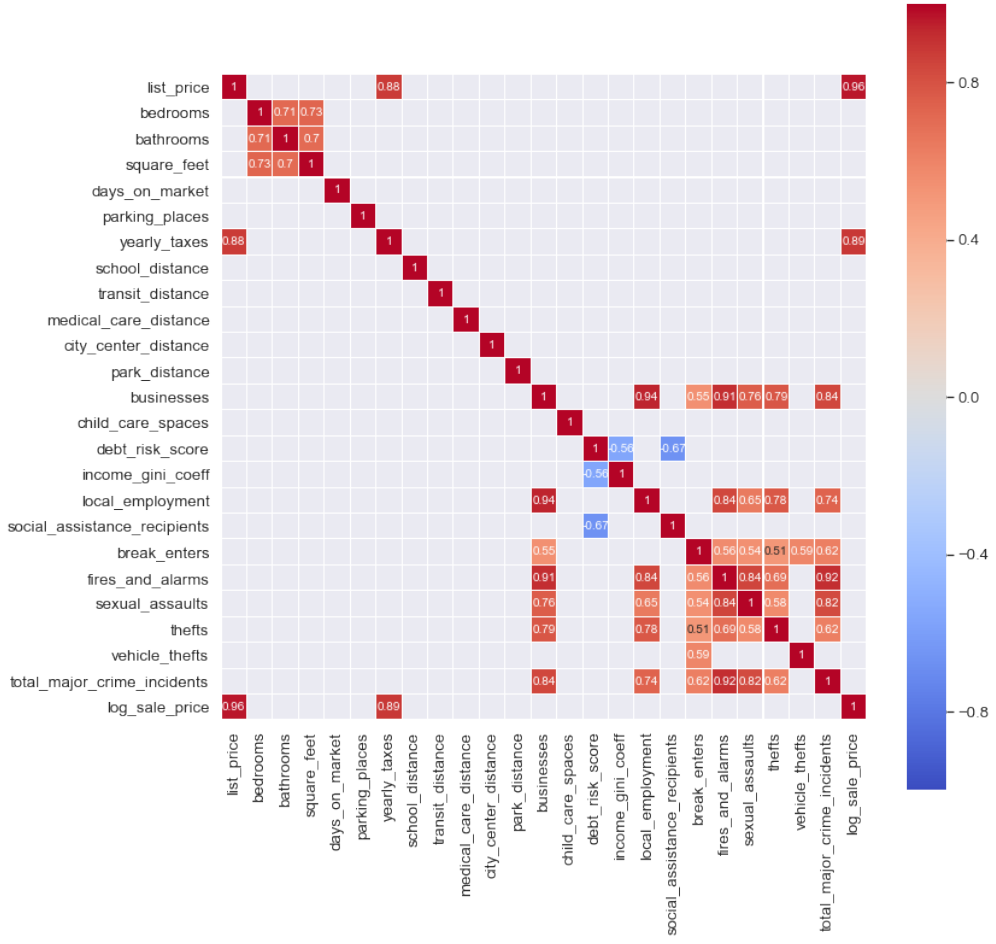


Figure 10: Heatmap of pairwise correlation of features in dataset. The plot shows those correlation values greater than 0.5.

From the above plot, we see that there are several of the variables that are correlated with each other.

Notable are the features related to the house namely, bedrooms, bathrooms and square feet. This is located in the top left corner of the chart.

Additionally, we see that there is strong correlation between the socio-economic variables. For instance, businesses and break and enters are highly correlated. Similarly, there is high correlation between fires and alarms, thefts, sexual assaults and businesses. Further, there is high correlation between total major crime incidents and fires and alarms, sexual assaults. There is also a positive correlation between social assistance recipients and debt risk score. This again seems like a reasonable correlation, as socio-economic trends tend to correlate with each other in cities.

7. REGRESSION ANALYSIS

Based on the above feature engineering, we performed regression analysis on the log of the sale price.

The feature variables that were correlated with the outcome variable but not correlated with each other were yearly_taxes, businesses, square_feet, break_enters, parking_places, park_distance, social_assistance_recipients and transit_distance. These were what we called the golden feature list.

All feature variables in the dataset were scaled and different combinations of the variables were used to perform a comparative regression analysis to verify our hypotheses. The results are summarized in the table below:

Table 8: Summary of regression analysis results for different combinations of features

Features	Golden Features	Only House Features	All Features	Only Socio-Economic Features
Adj. R-Sq.	0.79	0.80	0.83	0.29
Accuracy	0.80	0.79	0.82	0.27
Intercept	13.25	13.25	13.25	13.25

From the table above, we see that that the adjusted R-Squared and accuracy is lowest for the model using only the socio-economic variables. This makes sense, in that only the socio-economic variables cannot explain the variance in the dataset. Further, the model using the golden features and all features does not have a huge difference in the adjusted R-Squared and accuracy. This implies that the golden features account for most of the variance in the data. An interesting aspect of the analysis is the model that made use of only the house features in the dataset. These are features related to the house itself. Since the accuracy and adjusted R-squared values are same as other models, it is indicative that the buyers in the market do not consider the socio-economic variables when making a purchase.

For the model built using the golden features, we can plot the residual distribution plot to analyze the goodness of fit.

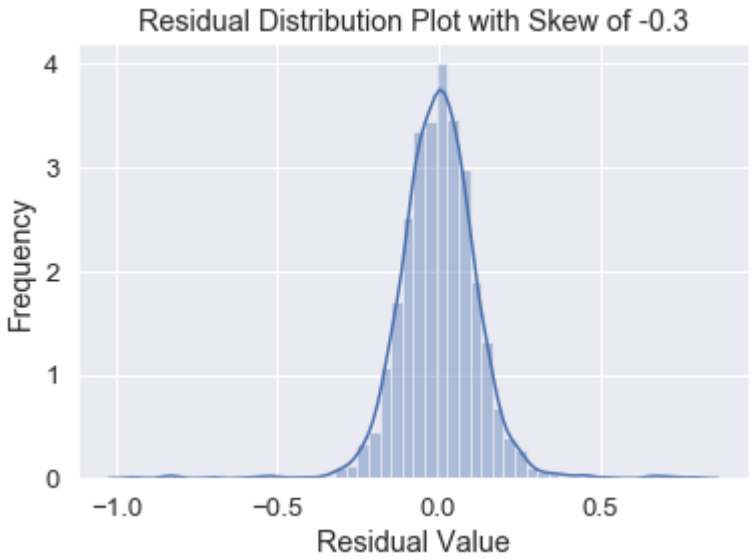


Figure 11: Residual distribution plot of model using selected features

From the plot above, we see that the residual distribution plot is normally distributed and has a skew of –0.3. This implies that the model overfits slightly but is still a fairly good fit for the regression model.

From the above models, we are able to determine that buyers and sellers on the market are not influenced by the socio-economic variables and that only a two of the socio-economic features account towards the sale price, namely, number of businesses and number of break and enters.

8. K-MEANS CLUSTERING

Using the above feature variables selected as our golden features list, a kmeans clustering analysis was conducted on the dataset.

The inertia for a point in a cluster is the sum of square distance from its nearest cluster center. The k-value for the clustering analysis was determined by plotting the inertia for different values of k and finding the elbow point in the plot. This is shown in the plot below.

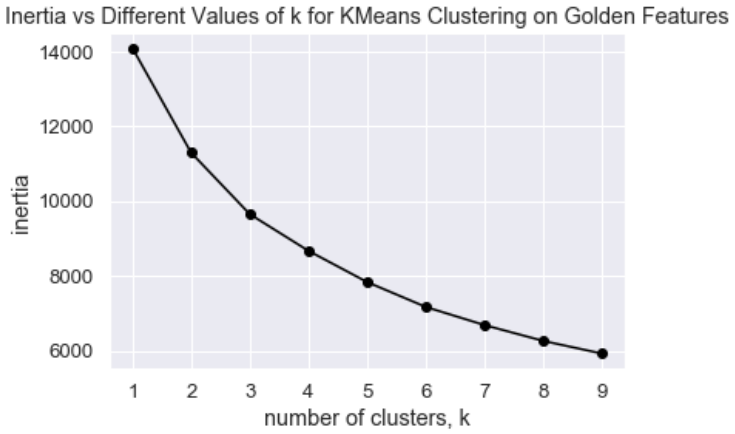


Figure 12: Plot of inertia vs different values of k. The plot shows the elbow point occurs as k=3

From the plot, we can see that the sum of square distances of points from the nearest cluster center does not decrease at a high rate after k=3. Thus, the optimal value for k to perform our k-means clustering is 3.

After creating the clusters, we can find some descriptive statistics associated with the cluster. This is shown in the table below.

Table 9: Count, Mean and standard deviation of sale price of each cluster resulting from k-means clustering

Cluster	Count	Sale Price (CAD$)	
		Mean	Standard Deviation
1	676	651,267.59	162,389.01
2	622	619,307.04	171,428.48
3	462	473,711.18	129,766.13

From the above table we see that each cluster is roughly the same size, with the biggest cluster being cluster 1. Cluster 1 also has the highest mean sale price. The standard deviation in price is the CAD$ 100K range for all three clusters, with cluster 3 having the highest standard deviation from the mean. These standard deviation values are high and implies all three clusters are fairly big covering different sale prices.

Figure 13: Plot showing variation of sale price for each cluster resulting from k-means clustering

The variation in the sale price across each of the clusters is also shown in the plot above. Again, it can be seen cluster 3 has most of its sale prices in the sub-CAD$400 category, while cluster 1 and 2 have a significant number of sale prices which can go up to CAD$ 1.2 Million.

We can also see how the features for each of these clusters compare against each other.

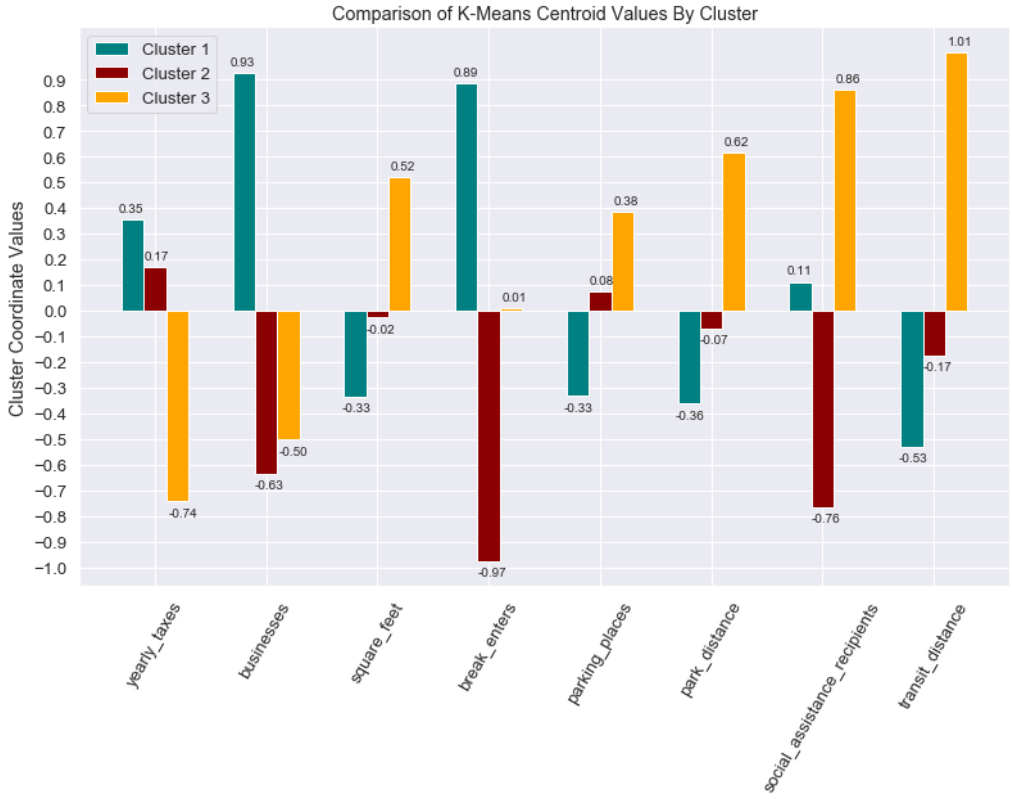


Figure 14: Comparison of centroid values by feature variables for each cluster

The above figure shows a comparison of cluster centroid coordinates for each of the cluster centroids. We can use this as a way to compare different features of each cluster. Cluster 1 has the highest yearly taxes and businesses, while having a high number of break and enters. It also has smallest square footage and is closer to parks and transit. This likely represents condominiums in the downtown core which is known for having smaller sized condominiums and sees higher foot traffic which accounts for the high number of businesses. Houses in such a cluster are likely to be of interest to single, working professionals who prefer to live closer to amenities and transit but don't require a large space.

Cluster 2 has the lowest number of businesses but has condominiums of size in between the other two clusters. It is also located further from parks and transit. This likely represents condominiums located just outside the downtown core areas in the city of Toronto. It also has the lowest number of social assistance recipients and has the second highest square footage of all three clusters. Cluster 2 likely represents the condominiums that would interest buyers with young families and higher than average incomes who have access to private transportation and can live further away from the downtown core.

Cluster 3 has the lowest amount of yearly taxes and fewer businesses than cluster 2 but the highest number of social assistance recipients. It also the cluster with houses located farthest away from the transit but has the houses with the highest square footage of the three clusters. This cluster likely represents people with lower income or people who are claiming Old-Age Security and are in the senior age bracket of the population. It likely represents empty-nesters or, retirees or people with low incomes who live further away from the main core areas of the city. From an investor's perspective, this cluster is lucrative for the lower yearly taxes, and the lower dollar per square-footage cost of purchase.

Investors looking to target young families for the sale and purchase of homes should look for houses in cluster 2. It is important to remember that cluster 2 also has the highest standard deviation so it is likely to find a higher or lower valued home that would attract higher and middle-income families that can afford to pay the higher sale price for proximity to downtown and larger space than the downtown core condominiums.

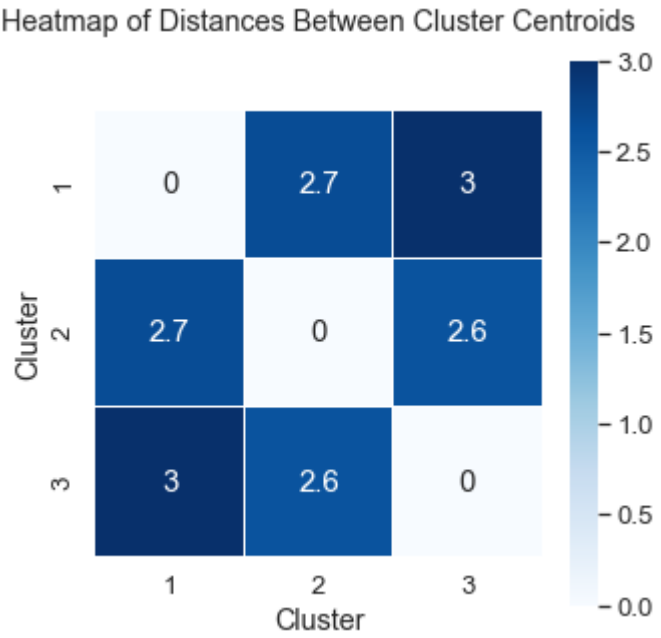


Figure 15: Heatmap of distances from cluster centroids. The figure all cluster centroids are fairly equidistant.

The above figure shows the distance between each of the cluster centroids. From the heat map we can determine that the cluster centers are roughly equidistant from each other. It also implies that cluster centers are not close and hence there is no significant overlap between the clusters. In other words, the condominiums in one cluster are not ambiguous and likely to be part of the cluster assigned, rather than being ambiguous and likely to be part of either one of two clusters.

We can also look at the clustering of sale price versus some of the socio-economic variables. In the figures below, we show the clustering along with the elbow point in the inertia plot for picking the k-value.

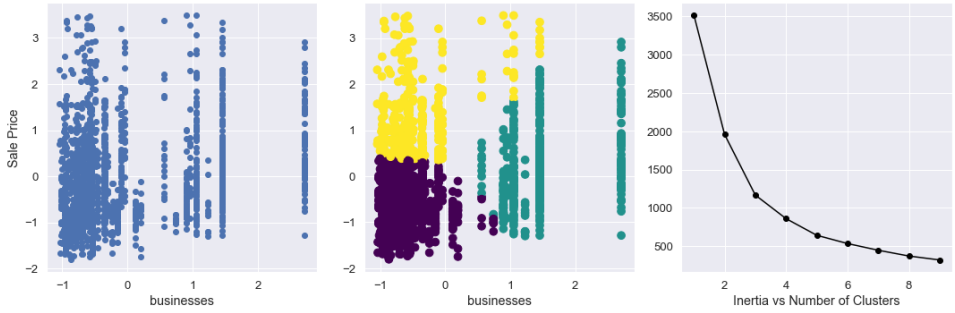


Figure 16: Clustering of sale price vs number of businesses. Plot also shows elbow point for optimal value of k.

For businesses, we see that there are three clusters as seen in the above plot. The first cluster in purple represents condominiums with low number of businesses and low sale price. There are also condos with high sale price and low number of businesses. The last cluster represents condos with high number of businesses and low sale price. This cluster is lucrative for investors who can expect these busy business areas to add further value through new essential services businesses in the future.

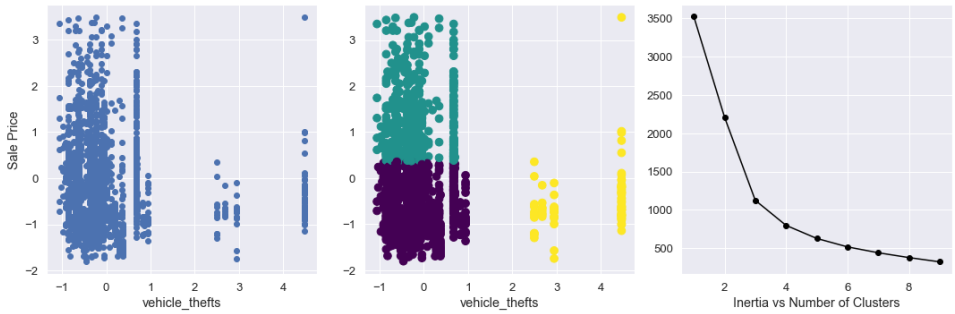


Figure 17: Clustering of sale price vs number of vehicle thefts. Plot also shows elbow point for optimal value of k.

A look at the cluster in of vehicle thefts in the above plots indicates that there is a cluster of condominiums which has a lower sale price but a high number of vehicle thefts. This is likely condominiums with no controlled parking entrance. It would be of interest to buyers looking to save money on the purchase of home but do not own private transportation and are not concerned with the theft of a vehicle.

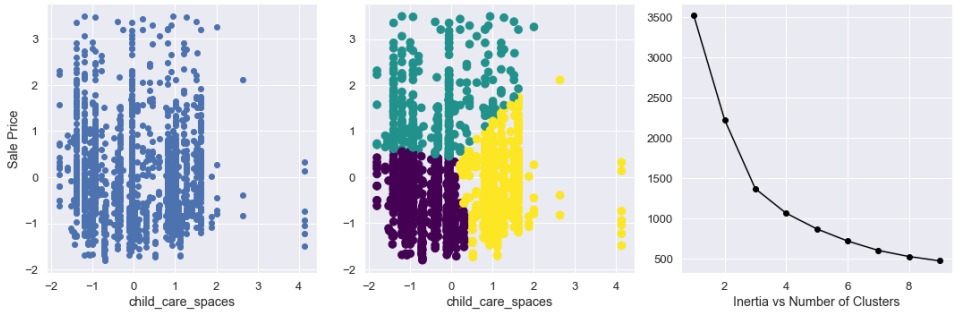


Figure 18: Clustering of sale price vs number of child care spaces. Plot also shows elbow point for optimal value of k.

Child care spaces are important for families when purchasing a condominium. The distribution and clustering of child care spaces is seen in the figure above. The most interesting cluster is highlighted in yellow which shows a moderate number of child care spaces with a low to moderate sale price. This cluster also has higher positive standard deviation of the number of child care spaces and families looking to purchase a condominium can consider this area. Families entering the market for the purchase of condominium and prioritizing child care as an amenity near the condo should ensure they are not in the boundary of any of the clusters, to avoid paying more than the mean sale price of yellow colored cluster.

9. FUTURE WORK AND IMPROVEMENTS

There are several areas for improvement in the analysis shown in this report. Firstly, the number of businesses in a neighbourhood is directly correlated with the number of break and enters. This indicates that the number of break and enters is not a measure of the number of break and enters in just condominiums but could indicate break and enters in commercial units as well. A more accurate measure would be the number of break and enters in residential units.

Secondly, several of the socio-economic variables show a count by neighbourhood. Unfortunately, these counts do not take into account the population density of the neighbourhood. For instance, the downtown core may have more social assistance recipients due to the higher population. More accurate and useful socio-economic variables would be those that give metrics in per-capita. This would account for differences in population density in neighbourhoods.

Thirdly, the socio-economic variables in this study are highly correlated amongst each other. This makes many of them un-useable in the regression analysis. While in this study, we used an already prepared dataset with these metrics, an improved

model would consist of metrics that are not correlated. For instance, number of families per household and income would be better uncorrelated data.

One variable that was not taken into consideration was age of the house. This was not obtained as the website that was scraped did not record this data. For a more accurate analysis of the data in the future, considering this variable would be useful.

Future areas of analysis include conducting a principal component analysis to reduce dimensionality and perform further analysis on the data. Several algorithms are available in the sklearn library that we did not implement here. These include gradient boosting, neural networks and support vector machines. These algorithms work differently with the data and are likely to give different accuracies based on the data. Further, data on this was only conducted for one time slice when the data was collected, I.e. in July 2019. A more thorough prediction of sale price requires a time-series analysis of the prices.

Finally, due to lack of time, we did not perform prediction on the number of days a condo is listed on the market. Given more time, it would be possible to conduct a similar numerical prediction for this.

10. CONCLUSION

Several interesting conclusions came out of this analysis. Firstly, there are a majority of condos that are bought and sold in the city of Toronto. Further, buyers and sellers give more importance to the features of the house than the socio-economic characteristics of the neighbourhood the house resides in. Of the features in our dataset, yearly_taxes, businesses, square_feet, break_enters, parking_places, park_distance, social_assistance_recipients and transit_distance explain a majority of the variance in the dataset. While this does not fully validate our initial hypothesis, it shows that some socio-economic aspects play a role in the price variance.

Clustering of our dataset using the aforementioned features results in 3 clusters which significantly minimize the inertia of the k-means model. The clusters can be classified as condominiums close to amenities and transit, suitable for single working professionals, condos located just outside the downtown core which are suitable for young families with higher than average income and access to private transportation, and a third cluster representing low income individual and senior citizens that is located away from the major business areas and transit. This is an important cluster for investors as it has lower yearly taxes and provides a lucrative investment option. There is a high standard deviation of sale price within each cluster. There also exist clusters for a large number of businesses and low sale price, low sale price and high number of vehicle thefts and high number of child care spaces and low sale price. These are respectively interesting investment options for investors, people with no private transportation, and families prioritizing child care spaces. While these observations do not fully validate our second hypothesis, they show how the data can be clustered with the golden features and a subset of the socio-economic variables.

11. REFRENCES

1. Cain, M., & Janssen, C. (1994). Real estate price prediction under asymmetric loss. Retrieved from https://link.springer.com/article/10.1007/BF00773391
2. Jian-Guo Liu, X. Z. (2006). Application of Fuzzy Neural Network for Real Estate Prediction. Retrieved from https://link.springer.com/chapter/10.1007/11760191_173
3. Sarip, A. G., Hafez, M. B., & Daud, N. (n.d.). Application of Fuzzy Regression Model for Real Estate Price Prediction. Retrieved from https://ejournal.um.edu.my/index.php/MJCS/article/view/6889
4. Graczyk, M., Lasota, T., Trawiński, B., & Trawiński, K. (2010). Comparison of Bagging, Boosting and Stacking Ensembles Applied to Real Estate Appraisal. Retrieved from https://link.springer.com/chapter/10.1007/978-3-642-12101-2_35
5. Xibin Wanga, Junhao Wena,b, Yihao Zhanga, & Yubiao Wanga (2013). Real estate price forecasting based on SVM optimized by PSO. Retrieved from https://www.sciencedirect.com/science/article/pii/S0030402613012515

APPENDIX: SOURCE CODE

In [119]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import geopandas as gpd
from descartes import PolygonPatch
from scipy.stats import skew
pd.set_option('display.max_columns', 500)
pd.set_option('display.max_rows', 500)
sns.set(style="darkgrid")
import warnings
warnings.filterwarnings('ignore')
%matplotlib inline
```

A.1 SAMPLE DATA

In [120]:
```
print('')
print('')
data = pd.read_csv('forAnalysis/data.csv')
data.sample(5)
```

Out[120]:

	street	city	province	neighbourhood	sale_price	list_price	bedrooms	bat
1930	<removed for anonymity>	Toronto	ON	High Park-swansea	2225000	2298000	4	4
986	<removed for anonymity>	Toronto	ON	Niagara	851000	859999	3	2
289	<removed for anonymity>	Toronto	ON	Dovercourt-wallace Emerson-junction	900000	899900	3	2
2462	<removed for anonymity>	Toronto	ON	Leaside	406000	399000	1	1
2429	<removed for anonymity>	Toronto	ON	West Hill	448800	448800	3	2

A.2 ROW COUNT

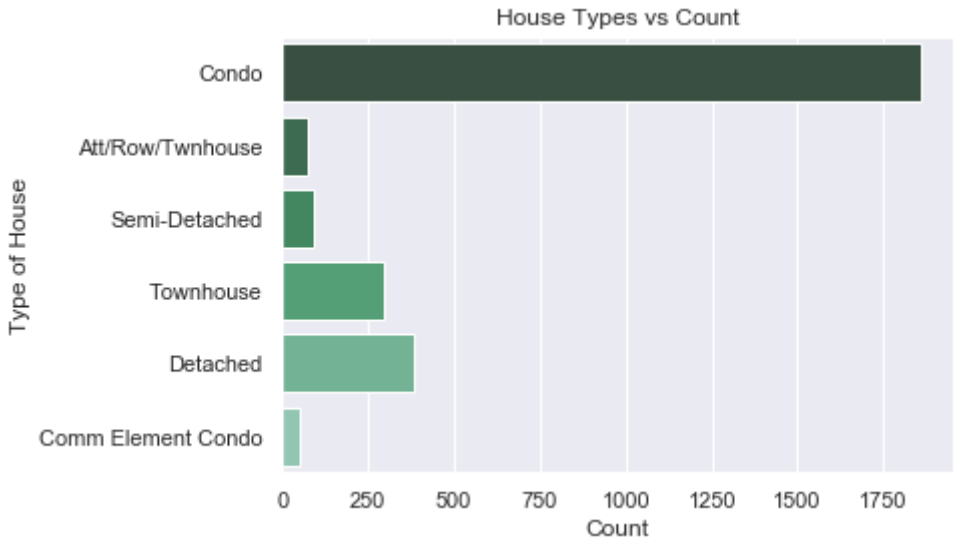
In [121]:
```
data.shape[0]
```

Out[121]: 2779

2779 rows were obtained by scraping the site.

A.3 DATA DISTRIBUTION

In [122]:
```
print('')
print('')
houseTypes = data.groupby("house_type").filter(lambda x: len(x) > 50)
fig = sns.countplot(y='house_type', data=houseTypes, palette='BuGn_d')
fig.set(xlabel='Count', ylabel='Type of House', title='House Types vs Co
unt')
plt.show(fig)
```



In [123]:
```
nb = 'forAnalysis/Neighbourhoods/Neighbourhoods.shp'
regions = gpd.read_file(nb)
regions['neighbourhood'] = regions['FIELD_7'].str.replace(' \(.+\)', '')
.str.lower()
```

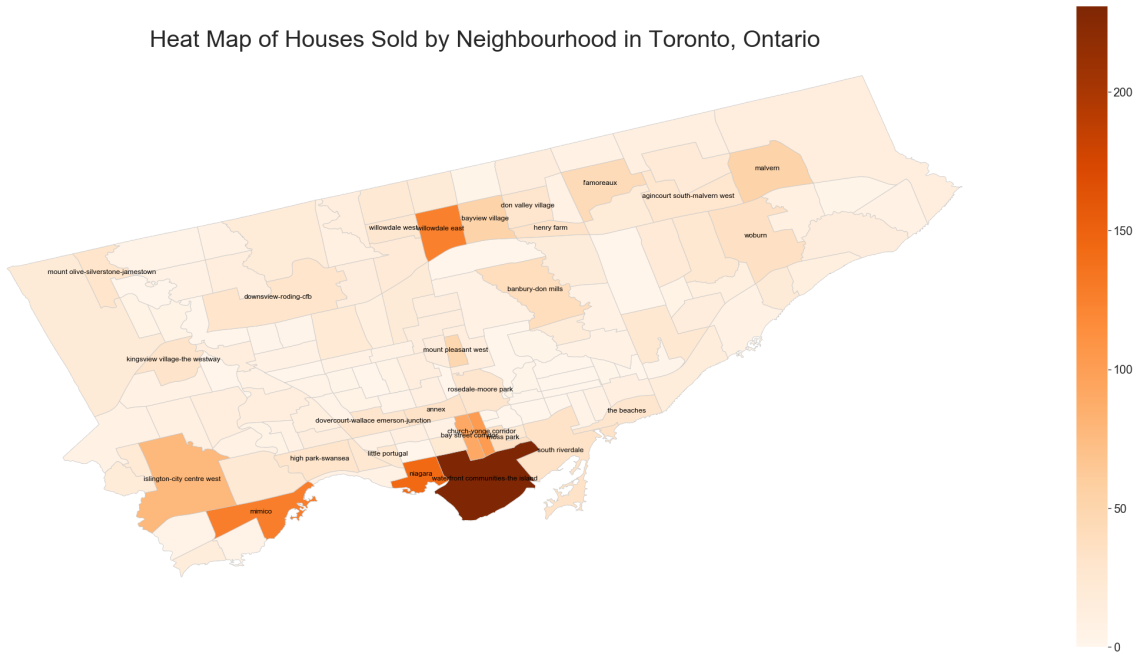
In [124]:
```
print('')
print('')
dataByNeighbourhood = data.groupby('neighbourhood').count()[['street']].
reset_index().replace('Waterfront Communities C1', 'Waterfront Communiti
es-The Island').replace('Rouge E11', 'Rouge')
dataByNeighbourhood.reset_index()
dataByNeighbourhood['neighbourhood'] = dataByNeighbourhood['neighbourhoo
d'].str.lower()
dataByNeighbourhood.sort_values('street', ascending=False).head(10)
```

Out[124]:

	neighbourhood	street
114	waterfront communities-the island	231
86	niagara	144
76	mimico	128
121	willowdale east	125
22	church-yonge corridor	102
6	bay street corridor	91
57	islington-city centre west	78
72	malvern	54
7	bayview village	53
82	mount pleasant west	49

A.4 GEO-MAPPING/HEAT MAP

In [125]:
```
# Join the dataset with the available mapping files.
# We will fill in the neighbourhoods which were not present in the datas
et with 0.
# this way we will still see the neighbourhood in the map, but no housin
g will be shown on it.
merged = regions.set_index('neighbourhood').join(dataByNeighbourhood.set
_index('neighbourhood'))
merged = merged.reset_index()
merged = merged.fillna(0)


# we are using the maximum and minimum house_type count values from the
 previous cell.
# setting additionally properties for the plot such as titles, turning o
f the axis for better visibility
# and setting the color scheme to look like a heat map.
vmin, vmax = 0, 231
fig, ax = plt.subplots(1, figsize=(40, 20))
ax.axis('off')
ax.set_title('Heat Map of Houses Sold by Neighbourhood in Toronto, Ontar
io', fontdict={'fontsize': '40', 'fontweight' : '3'})
color = 'Oranges'


# Create colorbar as a legend
# empty array for the data range
# add the colorbar to the figure
# set the color bar label text size
sm = plt.cm.ScalarMappable(cmap=color, norm=plt.Normalize(vmin=vmin, vma
x=vmax))
sm._A = []
cbar = fig.colorbar(sm)
cbar.ax.tick_params(labelsize=20)


# actually plot the map
# we will only annotate the plot for neighbourhoods with more than 25 ho
uses sold
merged.plot('street', cmap=color, linewidth=0.8, ax=ax, edgecolor='0.8',
figsize=(40,20))
for idx, row in merged.iterrows():
    if(row['street'] > 25):
        plt.annotate(s=row['neighbourhood'], xy=(row['FIELD_11'], row['F
IELD_12']),
                 horizontalalignment='center', fontsize='large', color=
'black', wrap=True)
plt.show()
```



A.5 CONVERTING AND FILTERING CONDOMINIUMS

In [126]:
```
print('')
print('')
data['house_type'].value_counts()
```

Out[126]:
```
Condo                1860
Detached              385
Townhouse             298
Semi-Detached          94
Att/Row/Twnhouse       75
Comm Element Condo     51
Co-Ownership Apt        5
Det Condo               4
Triplex                 2
Link                    2
Co-Op Apt               1
Duplex                  1
Multiplex               1
Name: house_type, dtype: int64
```

In [127]:
```
print('')
print('')
modified_data = data.replace('Comm Element Condo', 'Condo').replace('Det
Condo', 'Condo')
modified_data['house_type'].value_counts()
```

Out[127]:
```
Condo                1915
Detached              385
Townhouse             298
Semi-Detached          94
Att/Row/Twnhouse       75
Co-Ownership Apt        5
Triplex                 2
Link                    2
Co-Op Apt               1
Duplex                  1
Multiplex               1
Name: house_type, dtype: int64
```

In [128]:
```
print('')
print('')
condos_all_columns = modified_data[modified_data['house_type'] == 'Cond
o']
condos_all_columns.sample(5)
```

Out[128]:

	street	city	province	neighbourhood	sale_price	list_price	bedrooms	bat
2322	<removed for anonymity>	Toronto	ON	Church-yonge Corridor	600000	598000	1	1
213	<removed for anonymity>	Toronto	ON	Waterfront Communities C1	1175000	1149900	2	2
2079	<removed for anonymity>	Toronto	ON	Malvern	337000	299800	2	2
1908	<removed for anonymity>	Toronto	ON	Wychwood	570000	599000	1	1
587	<removed for anonymity>	Toronto	ON	Waterfront Communities C1	690000	699000	2	2

In [129]:
```
print('')
print('')
condos = condos_all_columns.drop(['street', 'city', 'province', 'neighbo
urhood', 'house_type','url'], axis=1)
condos.sample(5)
```

Out[129]:

	sale_price	list_price	bedrooms	bathrooms	square_feet	days_on_market	parking_
80	450000	450000	1	1	499	2	0
490	513000	524900	1	1	599	16	1
134	420000	429999	1	1	499	19	1
52	427000	439900	1	1	599	26	1
661	470000	475000	1	1	699	37	1

In [130]:
```
condos.shape
```

Out[130]: (1915, 25)

In [131]:
```
condos.isnull().values.any()
```

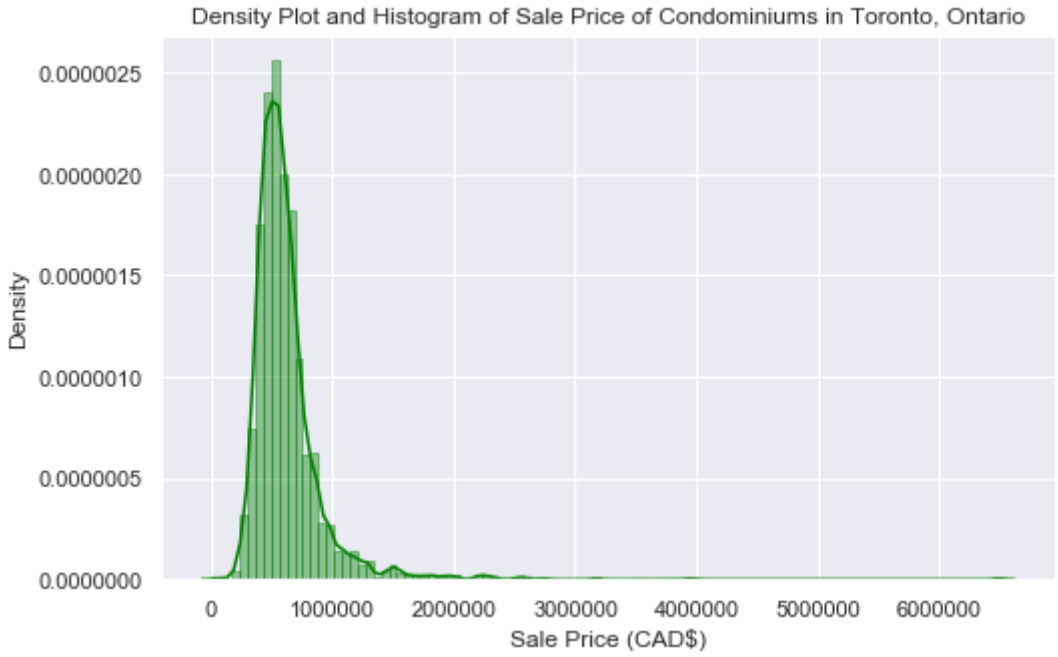
Out[131]: False

In [132]:
```
print('')
print('')
print(condos['sale_price'].describe().apply(lambda x: format(x, 'f')))
```

```
count       1915.000000
mean      627137.378068
std       318574.865134
min        47500.000000
25%       460000.000000
50%       560000.000000
75%       690000.000000
max      6500000.000000
Name: sale_price, dtype: object
```

A.6 DENSITY PLOT OF SALE PRICE

In [133]:
```
print('')
print('')
plt.figure(figsize=(8, 5))
fig = sns.distplot(condos['sale_price'], color='green', bins=100, hist_k
ws={'edgecolor':'green'});
plt.title('Density Plot and Histogram of Sale Price of Condominiums in T
oronto, Ontario', fontsize='large')
plt.ylabel('Density')
plt.xlabel('Sale Price (CAD$)')
plt.show(fig)
```

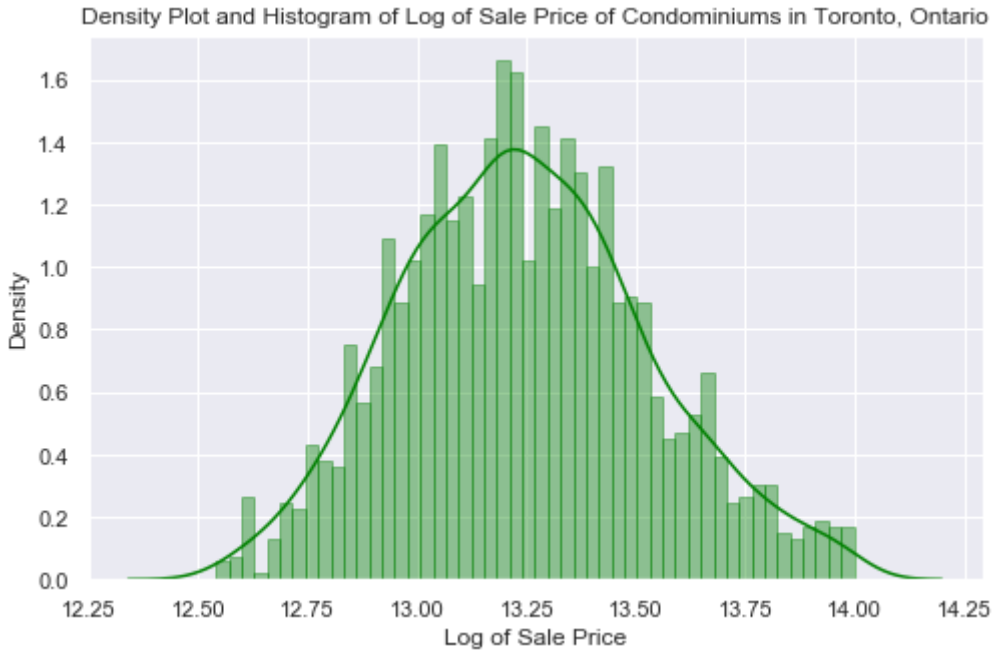


In [134]:
```
condos['sale_price'].skew()
```

Out[134]: 5.906281071445114

A.7 DENSITY PLOT OF SALE PRICE (NORMALIZED)

In [135]:
```
condos['log_sale_price'] = np.log(condos['sale_price'])
mean = condos['log_sale_price'].mean()
sd = condos['log_sale_price'].std()
condos_normalized = condos[np.abs(condos['log_sale_price'] - mean) < 2*s
d]

print('')
print('')
plt.figure(figsize=(8, 5))
fig = sns.distplot(condos_normalized['log_sale_price'], color='green', b
ins=50, hist_kws={'edgecolor':'green'});
plt.title('Density Plot and Histogram of Log of Sale Price of Condominiu
ms in Toronto, Ontario', fontsize='large')
plt.ylabel('Density')
plt.xlabel('Log of Sale Price')
plt.show(fig)
```



In [136]:
```
condos_normalized['log_sale_price'].skew()
```

Out[136]: 0.2151143258994082

In [137]:
```
condos_normalized.shape
```

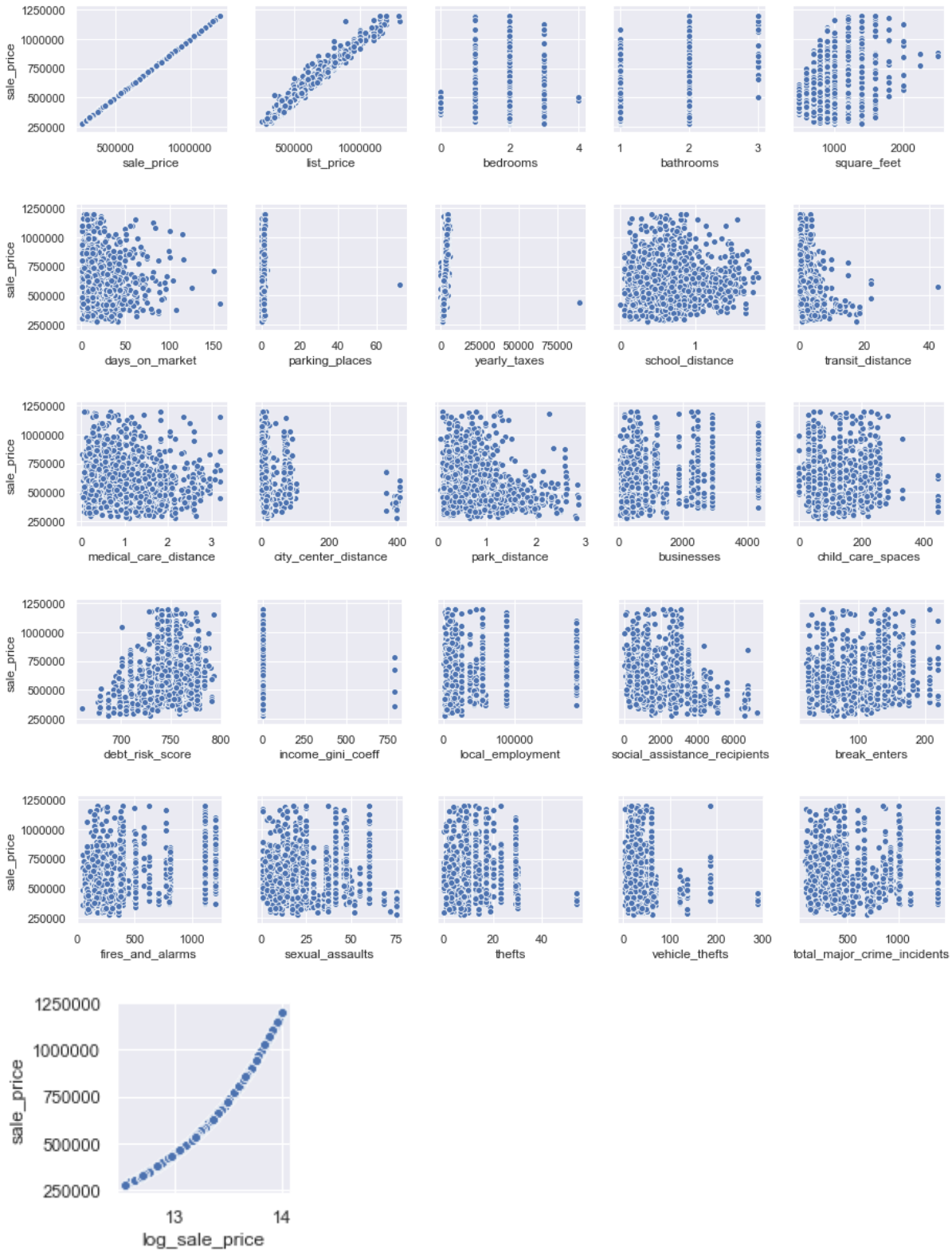
Out[137]: (1820, 26)

In [138]:
```
condos_normalized['sale_price'].describe().apply(lambda x: format(x,'f'
))
```

Out[138]:
```
count       1820.000000
mean      588375.977473
std       173871.162415
min       279500.000000
25%       460000.000000
50%       557250.000000
75%       676000.000000
max      1200000.000000
Name: sale_price, dtype: object
```

A.8 PAIR-WISE SCATTER PLOTS OF ALL FEATURES

In [139]:
```
print('')
print('')

for i in range(0, len(condos_normalized.columns), 5):
    sns.pairplot(data=condos_normalized,
                 x_vars=condos_normalized.columns[i:i+5],
                 y_vars=['sale_price'])
```

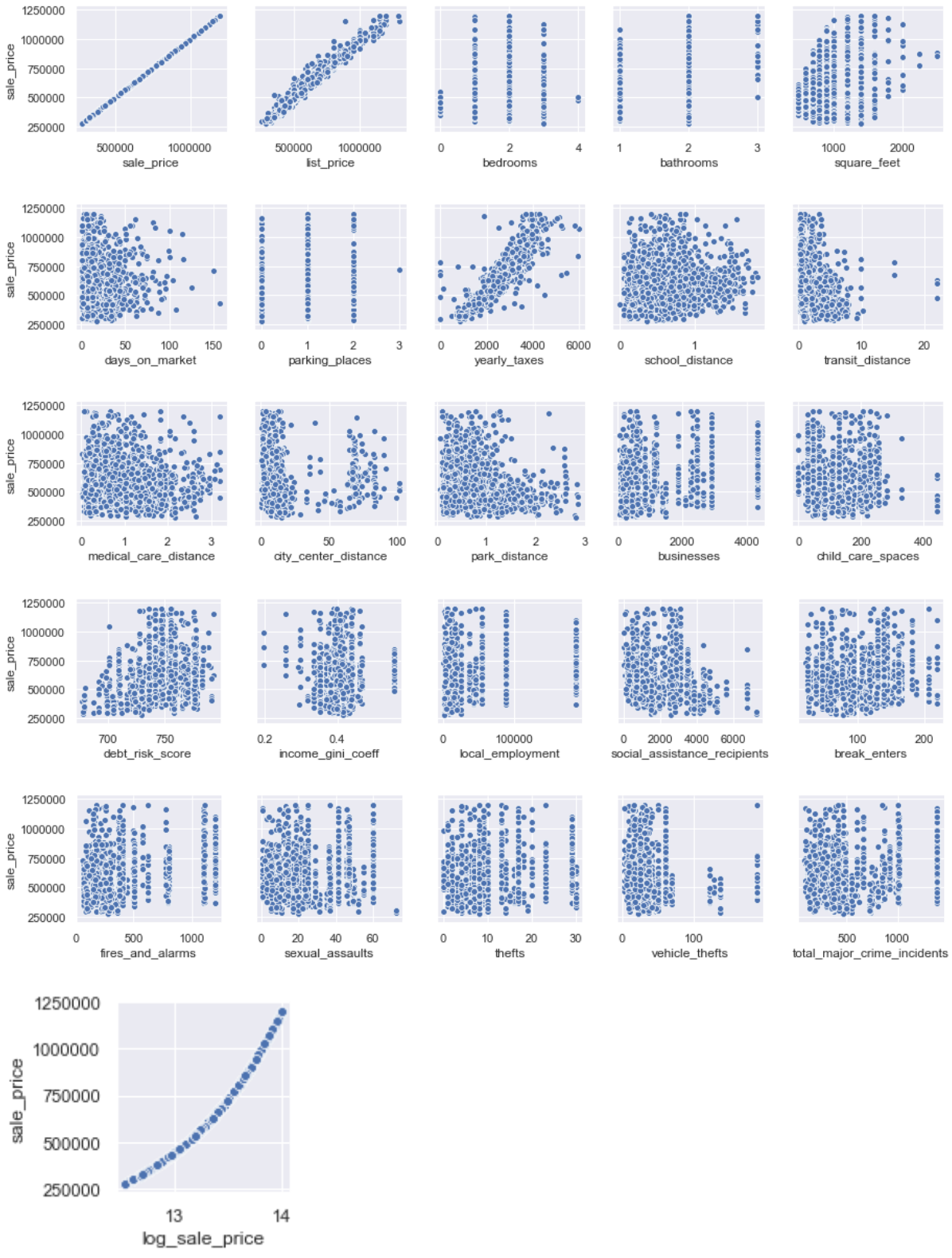


A.9 REMOVING OUTLIERS IN OTHER FEATURES AND REPLOTTING SCATTER PLOTS

In [140]:
```
condos_cleaned = condos_normalized[(condos_normalized['parking_places'] < 20 )
                                   & (condos_normalized['yearly_taxes'] < 25000)
                                   & (condos_normalized['transit_distance'] < 30)
                                   & (condos_normalized['city_center_distance'] < 200)
                                   & (condos_normalized['thefts'] < 40)
                                   & (condos_normalized['income_gini_coeff'] < 40 )
                                                                            ]
condos_cleaned.shape
```

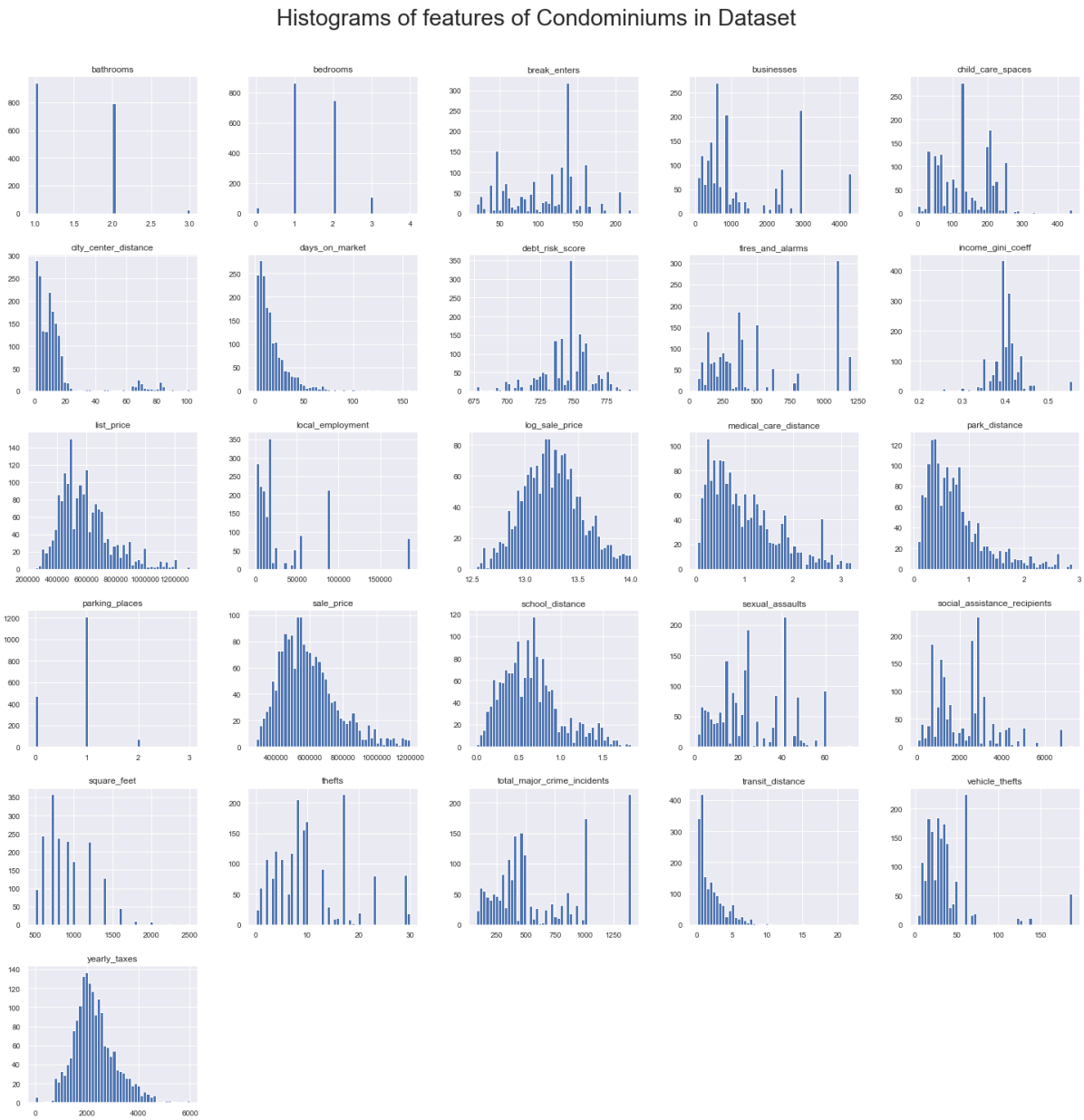
Out[140]: (1760, 26)

In [141]:
```
print('')
print('')

for i in range(0, len(condos_cleaned.columns), 5):
    sns.pairplot(data=condos_cleaned,
                 x_vars=condos_cleaned.columns[i:i+5],
                 y_vars=['sale_price'])
```



A.10 HISTOGRAM OF ALL FEATURES

In [142]:
```
print('')
print('')
condos_cleaned.hist(figsize=(25,25),bins=50, xlabelsize=10, ylabelsize=10)
plt.suptitle('Histograms of features of Condominiums in Dataset', y=0.93, fontsize=30)
plt.show()
```



In [143]:
```
condos_cleaned['sale_price'].describe().apply(lambda x: format(x,'f'))
```

Out[143]:
```
count       1760.000000
mean      593363.885227
std       173765.526225
min       280000.000000
25%       467125.000000
50%       560000.000000
75%       680000.000000
max      1200000.000000
Name: sale_price, dtype: object
```

In [144]:
```
condos_cleaned['log_sale_price'].skew()
```

Out[144]:
```
0.19544742161038794
```

A.11 PEARSON CORRELATION OF ALL FEATURES WITH OUTCOME

In [145]:
```
print('')
print('')
condos_corr_cleaned = condos_cleaned.corr()['log_sale_price'][1:]
golden_features_cleaned_list = condos_corr_cleaned[abs(condos_corr_clean
ed) > 0.1].sort_values(ascending=False)
print("There are {} strongly correlated values with Log Sale Price:\n{}"
.format(len(golden_features_cleaned_list), golden_features_cleaned_list
))
```

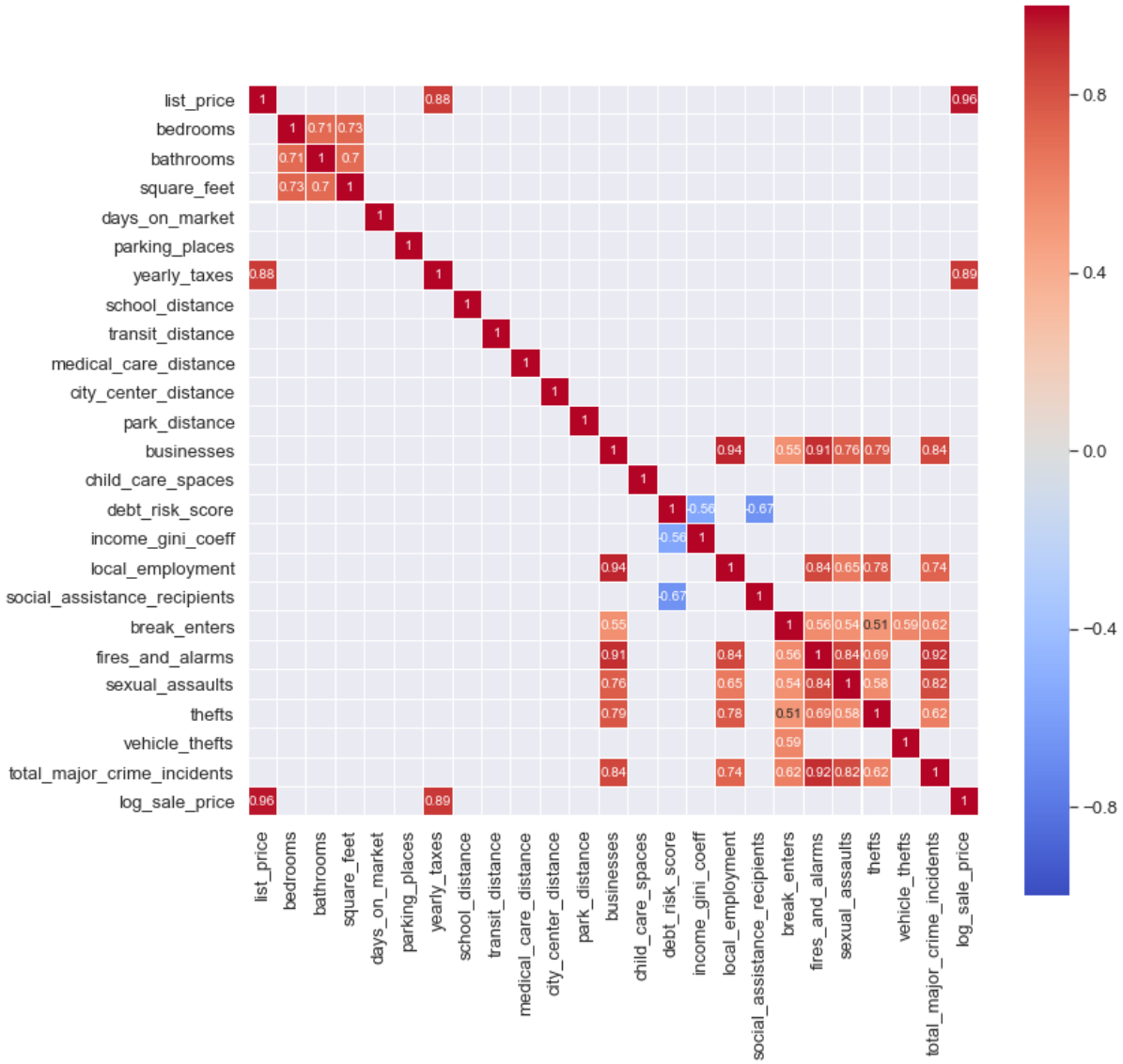
```
There are 18 strongly correlated values with Log Sale Price:
log_sale_price                  1.000000
list_price                      0.961250
yearly_taxes                    0.886556
bathrooms                       0.373000
fires_and_alarms                0.308304
debt_risk_score                 0.271255
local_employment                0.265035
businesses                      0.264677
square_feet                     0.251013
total_major_crime_incidents     0.198109
bedrooms                        0.170696
thefts                          0.169478
sexual_assaults                 0.131436
break_enters                    0.125633
parking_places                  0.109175
park_distance                  -0.250178
social_assistance_recipients   -0.305269
transit_distance               -0.318992
Name: log_sale_price, dtype: float64
```

In [146]:
```
golden_features_cleaned_list.index
```

Out[146]:
```
Index(['log_sale_price', 'list_price', 'yearly_taxes', 'bathrooms',
       'fires_and_alarms', 'debt_risk_score', 'local_employment', 'busi
nesses',
       'square_feet', 'total_major_crime_incidents', 'bedrooms', 'theft
s',
       'sexual_assaults', 'break_enters', 'parking_places', 'park_dista
nce',
       'social_assistance_recipients', 'transit_distance'],
      dtype='object')
```

A.12 PAIR-WISE PEARSON CORRELATION

In [147]:
```
cleaned_corr = condos_cleaned.drop('sale_price', axis = 1).corr()
plt.figure(figsize=(12, 12))
sns.set(font_scale=1.2)
sns.heatmap(cleaned_corr[(np.abs(cleaned_corr) >= 0.5) | (cleaned_corr ==
= 1)],
            cmap='coolwarm', vmax=1, vmin=-1, linewidths=0.1,
            annot=True, annot_kws={"size": 10}, square=True);
```



A.13 FEATURE SELECTION

In [148]:
```
golden_features_cleaned_list.index
```

Out[148]:
```
Index(['log_sale_price', 'list_price', 'yearly_taxes', 'bathrooms',
       'fires_and_alarms', 'debt_risk_score', 'local_employment', 'busi
nesses',
       'square_feet', 'total_major_crime_incidents', 'bedrooms', 'theft
s',
       'sexual_assaults', 'break_enters', 'parking_places', 'park_dista
nce',
       'social_assistance_recipients', 'transit_distance'],
      dtype='object')
```

In [149]:
```
golden_unstacked_corr = cleaned_corr.loc[golden_features_cleaned_list.in
dex, golden_features_cleaned_list.index].unstack()
golden_unstacked_corr[(golden_unstacked_corr.abs() > 0.5) & (golden_unst
acked_corr != 1)]
```

Out[149]:

log_sale_price	list_price	0.961250
	yearly_taxes	0.886556
list_price	log_sale_price	0.961250
	yearly_taxes	0.879152
yearly_taxes	log_sale_price	0.886556
	list_price	0.879152
bathrooms	square_feet	0.703789
	bedrooms	0.711302
fires_and_alarms	local_employment	0.840883
	businesses	0.914650
	total_major_crime_incidents	0.916897
	thefts	0.692334
	sexual_assaults	0.844394
	break_enters	0.563208
debt_risk_score	social_assistance_recipients	-0.667067
local_employment	fires_and_alarms	0.840883
	businesses	0.941023
	total_major_crime_incidents	0.737111
	thefts	0.779708
	sexual_assaults	0.645528
businesses	fires_and_alarms	0.914650
	local_employment	0.941023
	total_major_crime_incidents	0.842298
	thefts	0.787174
	sexual_assaults	0.761872
	break_enters	0.546692
square_feet	bathrooms	0.703789
	bedrooms	0.729675
total_major_crime_incidents	fires_and_alarms	0.916897
	local_employment	0.737111
	businesses	0.842298
	thefts	0.622084
	sexual_assaults	0.820557
	break_enters	0.621937
bedrooms	bathrooms	0.711302
	square_feet	0.729675
thefts	fires_and_alarms	0.692334
	local_employment	0.779708
	businesses	0.787174
	total_major_crime_incidents	0.622084
	sexual_assaults	0.577087
	break_enters	0.513636
sexual_assaults	fires_and_alarms	0.844394
	local_employment	0.645528
	businesses	0.761872
	total_major_crime_incidents	0.820557
	thefts	0.577087
	break_enters	0.539629
break_enters	fires_and_alarms	0.563208
	businesses	0.546692
	total_major_crime_incidents	0.621937
	thefts	0.513636
	sexual_assaults	0.539629
social_assistance_recipients	debt_risk_score	-0.667067

dtype: float64

A.14 REGRESSION ANALYSIS

In [150]:
```
from sklearn import datasets, linear_model
from sklearn.model_selection import cross_val_score
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import StandardScaler
import statsmodels.api as sm
```

In [151]:
```
##############################################
# First Model - selected features
##############################################


sc = StandardScaler()

X_original = condos_cleaned[['yearly_taxes', 'businesses',
       'square_feet', 'break_enters', 'parking_places', 'park_distance',
       'social_assistance_recipients', 'transit_distance']]

y = condos_cleaned['log_sale_price']
X = sc.fit_transform(X_original)
linearModel = linear_model.LinearRegression()
linear_score = cross_val_score(linearModel, X, y, cv = 10)

print('Linear model 1 accuracy: ', np.mean(linear_score))

model = linear_reg.fit(X,y)
print('R-Squared: \n', model.score(X,y))
print('Intercept: \n', model.intercept_)
print('Coefficients: \n', model.coef_)
scores = cross_val_score(linear_reg, X, y, cv=10)
print('Cross Validation Scores: \n', scores)
print('\nAverage Accuracy: {}'.format(np.mean(scores)))
x = sm.add_constant(X)
model = sm.OLS(y, x).fit()
predictions = model.predict(x)

# Print out the statistics
model.summary()
```

```
Linear model 1 accuracy:  0.7951379918974435
R-Squared:
 0.7997820048346587
Intercept:
 13.253426767235963
Coefficients:
 [ 0.22344751  0.01931482  0.02761301  0.00245046  0.007251   -0.021403
54
 -0.01866655 -0.00764285]
Cross Validation Scores:
 [0.86489429 0.75807808 0.84995199 0.75876826 0.72799924 0.80958705
 0.75601648 0.75083514 0.86875572 0.80649366]

Average Accuracy: 0.7951379918974435
```

Out[151]:

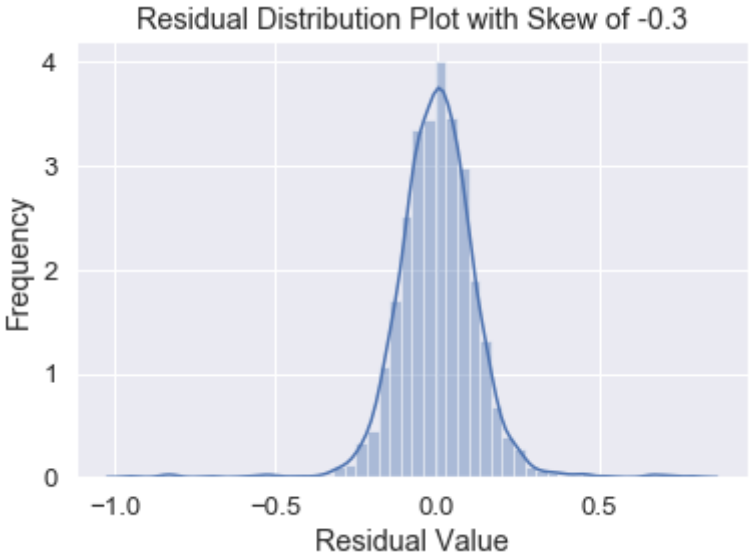
OLS Regression Results

Dep. Variable:	log_sale_price	**R-squared:**	0.800
Model:	OLS	**Adj. R-squared:**	0.799
Method:	Least Squares	**F-statistic:**	874.3
Date:	Sun, 28 Jul 2019	**Prob (F-statistic):**	0.00
Time:	12:32:39	**Log-Likelihood:**	1152.2
No. Observations:	1760	**AIC:**	-2286.
Df Residuals:	1751	**BIC:**	-2237.
Df Model:	8		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**
const	13.2534	0.003	4410.880	0.000	13.248	13.259
x1	0.2234	0.004	56.533	0.000	0.216	0.231
x2	0.0193	0.004	4.789	0.000	0.011	0.027
x3	0.0276	0.004	7.363	0.000	0.020	0.035
x4	0.0025	0.004	0.630	0.529	-0.005	0.010
x5	0.0073	0.003	2.131	0.033	0.001	0.014
x6	-0.0214	0.003	-6.682	0.000	-0.028	-0.015
x7	-0.0187	0.004	-5.202	0.000	-0.026	-0.012
x8	-0.0076	0.004	-2.148	0.032	-0.015	-0.001

Omnibus:	320.656	**Durbin-Watson:**	1.966
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	5489.182
Skew:	0.320	**Prob(JB):**	0.00
Kurtosis:	11.628	**Cond. No.**	2.60

In [152]:
```
residuals = predictions - y
sns.distplot(residuals)
plt.xlabel('Residual Value')
plt.ylabel('Frequency')
plt.title('Residual Distribution Plot with Skew of -0.3')
plt.show()
```



In [153]:
```
#######################################################################
### SEcond model - only socio economic variables
#######################################################################

X2_original = condos_cleaned.iloc[:,13:25]

X2 = sc.fit_transform(X2_original)

y2 = condos_cleaned['log_sale_price']

linearModel2 = linear_model.LinearRegression()
linear_score2 = cross_val_score(linearModel2, X2, y2, cv = 10)

print('Linear model 2 accuracy: ', np.mean(linear_score2))

model2 = linearModel2.fit(X2,y2)

print('R-Squared: \n', model2.score(X2,y2))
print('Intercept: \n', model2.intercept_)
print('Coefficients: \n', model2.coef_)
scores2 = cross_val_score(linear_reg, X2, y2, cv=10)
print('Cross Validation Scores: \n', scores2)
print('\nAverage Accuracy: {}'.format(np.mean(scores2)))
x2 = sm.add_constant(X2)
model2 = sm.OLS(y2, x2).fit()
predictions = model2.predict(x2)

model2.summary()
```

```
Linear model 2 accuracy:  0.278496567410697
R-Squared:
 0.2946912606777127
Intercept:
 13.253426767235963
Coefficients:
 [-0.0500798   0.03599302 -0.00487565  0.01299521 -0.03214592 -0.139282
31
  0.01485984  0.16811977 -0.0305812  -0.01419821 -0.03483877  0.0656379
2]
Cross Validation Scores:
 [0.29636009 0.18923254 0.35174084 0.29403391 0.30454744 0.07308851
 0.39438747 0.30448477 0.31671943 0.26037069]

Average Accuracy: 0.278496567410697
```

Out[153]:

OLS Regression Results

Dep. Variable:	log_sale_price	**R-squared:**	0.295
Model:	OLS	**Adj. R-squared:**	0.290
Method:	Least Squares	**F-statistic:**	60.83
Date:	Sun, 28 Jul 2019	**Prob (F-statistic):**	3.52e-123
Time:	12:32:40	**Log-Likelihood:**	44.087
No. Observations:	1760	**AIC:**	-62.17
Df Residuals:	1747	**BIC:**	8.977
Df Model:	12		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**
const	13.2534	0.006	2347.419	0.000	13.242	13.265
x1	-0.0501	0.029	-1.756	0.079	-0.106	0.006
x2	0.0360	0.008	4.724	0.000	0.021	0.051
x3	-0.0049	0.012	-0.405	0.685	-0.028	0.019
x4	0.0130	0.008	1.537	0.124	-0.004	0.030
x5	-0.0321	0.023	-1.387	0.166	-0.078	0.013
x6	-0.1393	0.011	-12.644	0.000	-0.161	-0.118
x7	0.0149	0.010	1.536	0.125	-0.004	0.034
x8	0.1681	0.027	6.282	0.000	0.116	0.221
x9	-0.0306	0.014	-2.265	0.024	-0.057	-0.004
x10	-0.0142	0.011	-1.321	0.187	-0.035	0.007
x11	-0.0348	0.009	-3.946	0.000	-0.052	-0.018
x12	0.0656	0.021	3.137	0.002	0.025	0.107

Omnibus:	81.353	**Durbin-Watson:**	2.000
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	92.213
Skew:	0.532	**Prob(JB):**	9.47e-21
Kurtosis:	3.353	**Cond. No.**	15.1

In [154]:
```
##############################################
# Third model all House Values
##############################################

X3_original = condos_cleaned.iloc[:,2:13]

y3 = condos_cleaned['log_sale_price']

X3 = sc.fit_transform(X3_original)

lasso3 = linear_model.Lasso()
linearModel3 = linear_model.LinearRegression()
linear_score3 = cross_val_score(linearModel3, X3, y3, cv = 10)

print('Linear model 3 accuracy: ', np.mean(linear_score3))

model3 = linearModel3.fit(X3,y3)

print('R-Squared: \n', model3.score(X3,y3))
print('Intercept: \n', model3.intercept_)
print('Coefficients: \n', model3.coef_)
scores3 = cross_val_score(linear_reg, X3, y3, cv=10)
print('Cross Validation Scores: \n', scores3)
print('\nAverage Accuracy: {}'.format(np.mean(scores3)))

x3 = sm.add_constant(X3)
model3 = sm.OLS(y3, x3).fit()
predictions = model3.predict(x3)

model3.summary()
```

```
Linear model 3 accuracy:  0.7995831791442237
R-Squared:
 0.8047944594949075
Intercept:
 13.253426767235963
Coefficients:
 [ 0.00178804  0.02592625  0.00616648 -0.01202886  0.00034881  0.229960
29
  0.01299522 -0.01622896  0.00292425 -0.01471657 -0.01973935]
Cross Validation Scores:
 [0.85690838 0.76175959 0.85996563 0.76518193 0.7338329  0.82443518
 0.76424512 0.74062718 0.8714891  0.81738678]

Average Accuracy: 0.7995831791442237
```

Out[154]:

OLS Regression Results

Dep. Variable:	log_sale_price	**R-squared:**	0.805
Model:	OLS	**Adj. R-squared:**	0.804
Method:	Least Squares	**F-statistic:**	655.2
Date:	Sun, 28 Jul 2019	**Prob (F-statistic):**	0.00
Time:	12:32:40	**Log-Likelihood:**	1174.5
No. Observations:	1760	**AIC:**	-2325.
Df Residuals:	1748	**BIC:**	-2259.
Df Model:	11		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**
const	13.2534	0.003	4463.324	0.000	13.248	13.259
x1	0.0018	0.005	0.368	0.713	-0.008	0.011
x2	0.0259	0.005	5.328	0.000	0.016	0.035
x3	0.0062	0.005	1.236	0.217	-0.004	0.016
x4	-0.0120	0.003	-3.992	0.000	-0.018	-0.006
x5	0.0003	0.003	0.105	0.917	-0.006	0.007
x6	0.2300	0.004	62.521	0.000	0.223	0.237
x7	0.0130	0.003	4.006	0.000	0.007	0.019
x8	-0.0162	0.004	-4.467	0.000	-0.023	-0.009
x9	0.0029	0.003	0.897	0.370	-0.003	0.009
x10	-0.0147	0.003	-4.919	0.000	-0.021	-0.009
x11	-0.0197	0.003	-6.258	0.000	-0.026	-0.014

Omnibus:	346.390	**Durbin-Watson:**	1.983
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	6279.784
Skew:	0.394	**Prob(JB):**	0.00
Kurtosis:	12.220	**Cond. No.**	3.53

In [155]:
```
##############################################
# Fourth model all features
##############################################

X4_original = condos_cleaned.iloc[:,2:25]

X4 = sc.fit_transform(X4_original)

y4 = condos_cleaned['log_sale_price']

linearModel4 = linear_model.LinearRegression()
linear_score4 = cross_val_score(linearModel4, X4, y4, cv = 10)

print('Linear model 4 accuracy: ', np.mean(linear_score4))


model4 = linearModel4.fit(X4,y4)

print('R-Squared: \n', model4.score(X4,y4))
print('Intercept: \n', model4.intercept_)
print('Coefficients: \n', model4.coef_)
scores4 = cross_val_score(linear_reg, X4, y4, cv=10)
print('Cross Validation Scores: \n', scores4)
print('\nAverage Accuracy: {}'.format(np.mean(scores4)))


x4 = sm.add_constant(X4)
model4 = sm.OLS(y4, x4).fit()
predictions = model4.predict(x4)

model4.summary()
```

```
Linear model 4 accuracy:  0.8236611625359462
R-Squared:
 0.831871064290193
Intercept:
 13.253426767235963
Coefficients:
 [ 0.00807902  0.02949302  0.0211603  -0.01223962  0.01105085  0.198647
6
  0.00998604 -0.01069755  0.01298931 -0.01113045 -0.01087425 -0.0243100
9
  0.00316682 -0.00901195  0.01476469 -0.02304404 -0.064435   -0.0044493
8
  0.05270837 -0.00349306  0.00278768 -0.01137757  0.05437557]
Cross Validation Scores:
 [0.86806366 0.79480259 0.85887877 0.78812348 0.77436878 0.83636191
 0.81733777 0.77552272 0.89709651 0.82605543]

Average Accuracy: 0.8236611625359462
```

Out[155]:

OLS Regression Results

Dep. Variable:	log_sale_price	**R-squared:**	0.832
Model:	OLS	**Adj. R-squared:**	0.830
Method:	Least Squares	**F-statistic:**	373.5
Date:	Sun, 28 Jul 2019	**Prob (F-statistic):**	0.00
Time:	12:32:40	**Log-Likelihood:**	1305.9
No. Observations:	1760	**AIC:**	-2564.
Df Residuals:	1736	**BIC:**	-2432.
Df Model:	23		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**
const	13.2534	0.003	4792.778	0.000	13.248	13.259
x1	0.0081	0.005	1.765	0.078	-0.001	0.017
x2	0.0295	0.005	6.455	0.000	0.021	0.038
x3	0.0212	0.005	4.427	0.000	0.012	0.031
x4	-0.0122	0.003	-4.347	0.000	-0.018	-0.007
x5	0.0111	0.003	3.413	0.001	0.005	0.017
x6	0.1986	0.004	49.094	0.000	0.191	0.207
x7	0.0100	0.003	3.096	0.002	0.004	0.016
x8	-0.0107	0.004	-3.014	0.003	-0.018	-0.004
x9	0.0130	0.003	3.992	0.000	0.007	0.019
x10	-0.0111	0.003	-3.952	0.000	-0.017	-0.006
x11	-0.0109	0.003	-3.285	0.001	-0.017	-0.004
x12	-0.0243	0.014	-1.699	0.089	-0.052	0.004
x13	0.0032	0.004	0.829	0.407	-0.004	0.011
x14	-0.0090	0.006	-1.503	0.133	-0.021	0.003
x15	0.0148	0.004	3.480	0.001	0.006	0.023
x16	-0.0230	0.012	-2.000	0.046	-0.046	-0.000
x17	-0.0644	0.006	-10.735	0.000	-0.076	-0.053
x18	-0.0044	0.005	-0.930	0.352	-0.014	0.005
x19	0.0527	0.014	3.852	0.000	0.026	0.080
x20	-0.0035	0.007	-0.514	0.607	-0.017	0.010

x21	0.0028	0.006	0.488	0.626	-0.008	0.014
x22	-0.0114	0.004	-2.552	0.011	-0.020	-0.003
x23	0.0544	0.011	5.177	0.000	0.034	0.075

Omnibus:	286.295	**Durbin-Watson:**	1.977
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	3884.627
Skew:	0.295	**Prob(JB):**	0.00
Kurtosis:	10.254	**Cond. No.**	16.3

A.15 KMEANS CLUSTERING

In [156]:
```
# condos_cleaned.iloc[:,2:25]
x = condos_cleaned.iloc[:,2:25]
x[1:5]
```

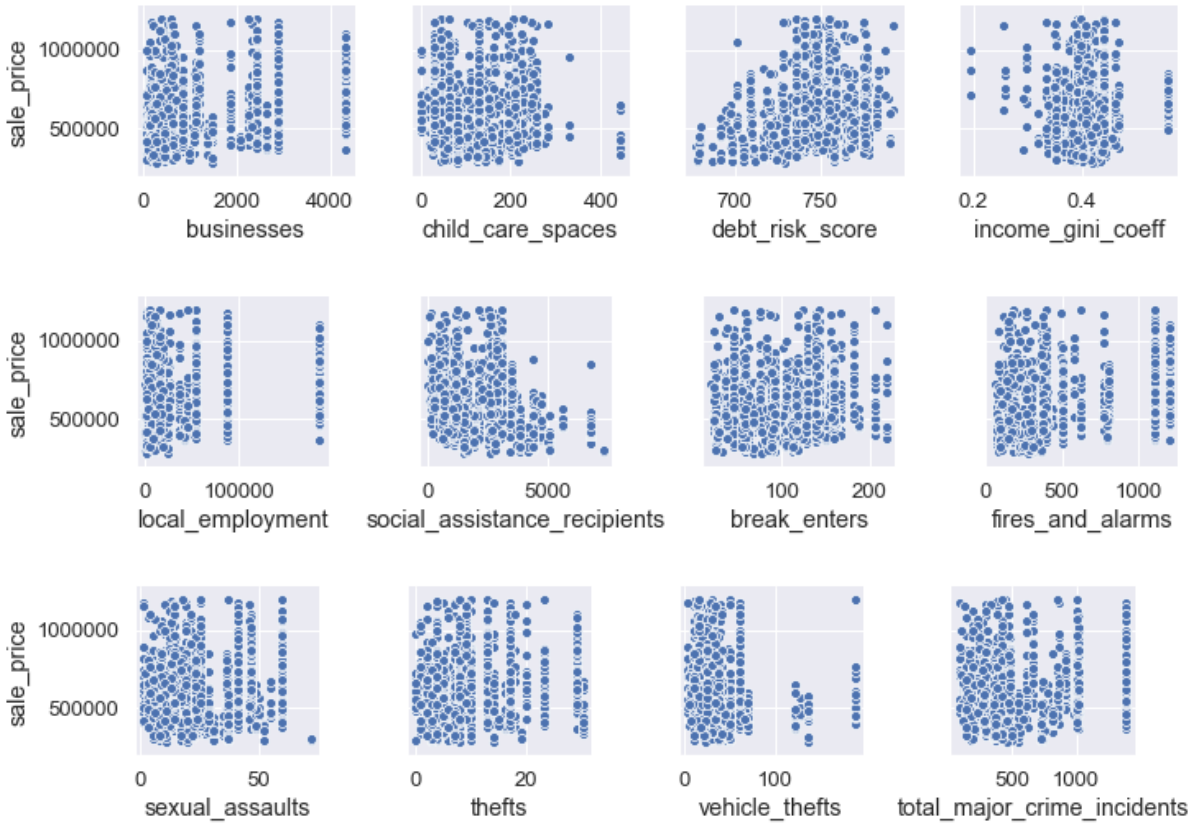
Out[156]:

	bedrooms	bathrooms	square_feet	days_on_market	parking_places	yearly_taxes	s
5	1	1	599	34	0	2586.00	0
6	1	1	599	32	1	1849.30	0
7	2	2	799	32	0	2184.95	0
10	2	2	1199	30	1	3952.84	0

In [157]:
```
x.shape
```

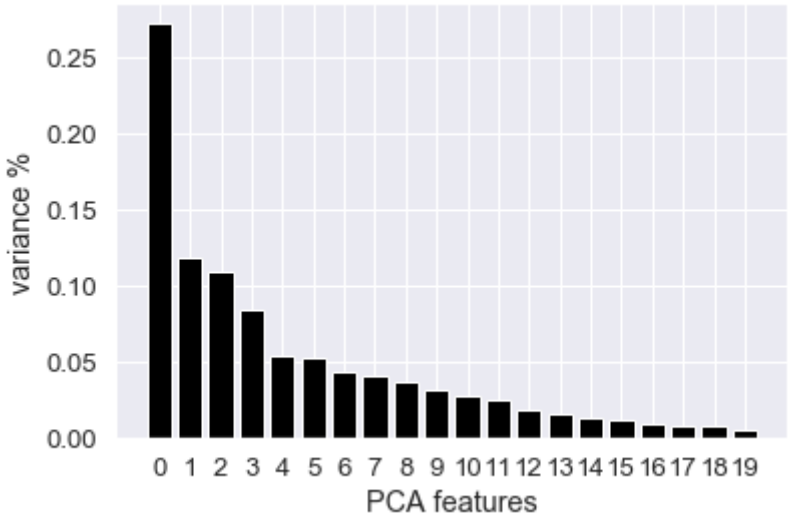
Out[157]: (1760, 23)

In [158]:
```
for i in range(13,25,4):
    sns.pairplot(data=condos_cleaned,
                 x_vars=condos_cleaned.columns[i:i+4],
                 y_vars=['sale_price'])
```



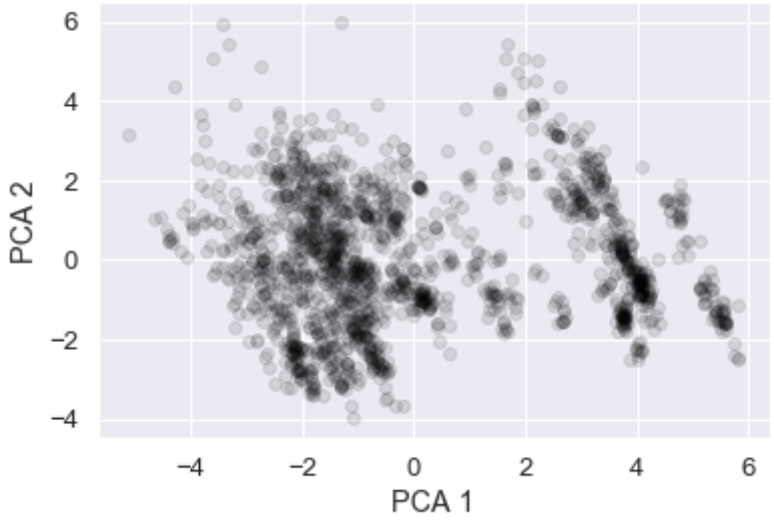
A.16 KMEANS CLUSTERING WITH PCA

In [159]:
```
from sklearn.decomposition import PCA
from sklearn.cluster import KMeans
from mpl_toolkits.mplot3d import Axes3D

from sklearn.preprocessing import StandardScaler

sc = StandardScaler()
x_scaled = sc.fit_transform(x)
pca = PCA(n_components=20)
principalComponents = pca.fit_transform(x_scaled)

features = range(pca.n_components_)
plt.bar(features, pca.explained_variance_ratio_, color='black')
plt.xlabel('PCA features')
plt.ylabel('variance %')
plt.xticks(features)

PCA_components = pd.DataFrame(principalComponents)
```



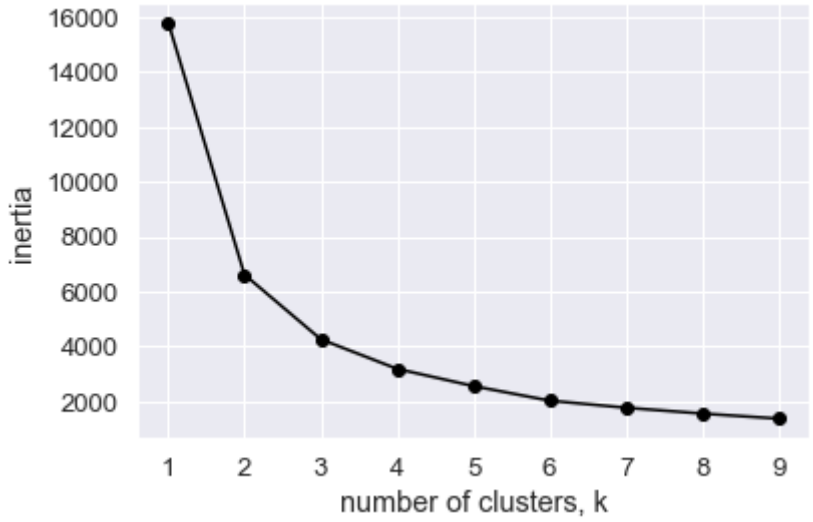
In [160]:
```
plt.scatter(PCA_components[0], PCA_components[1], alpha=.1, color='blac
k')
plt.xlabel('PCA 1')
plt.ylabel('PCA 2')
```

Out[160]: <matplotlib.text.Text at 0x1a240f2ef0>



In [161]:
```
pca = PCA(n_components=2)
x_pca = pca.fit_transform(x_scaled)
```

In [162]:
```
ks = range(1, 10)
inertias = []
for k in ks:
    # Create a KMeans instance with k clusters: model
    model = KMeans(n_clusters=k)

    # Fit model to samples
    model.fit(PCA_components.iloc[:,:2])

    # Append the inertia to the list of inertias
    inertias.append(model.inertia_)

plt.plot(ks, inertias, '-o', color='black')
plt.xlabel('number of clusters, k')
plt.ylabel('inertia')
plt.xticks(ks)
plt.show()
```



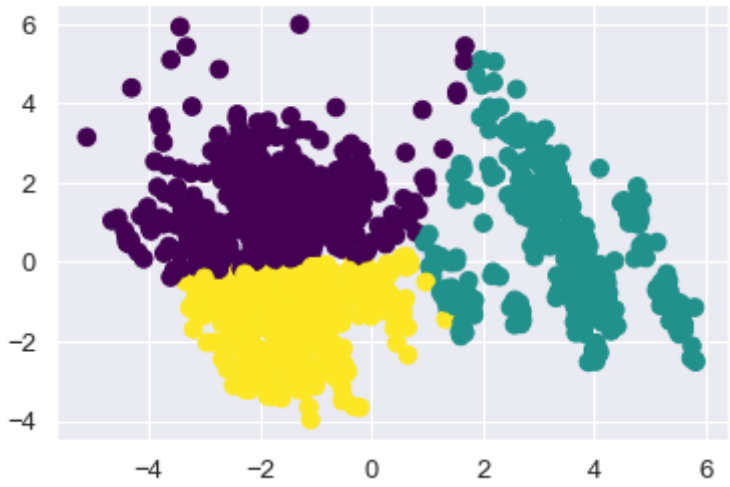
In [163]:
```
kmeans = KMeans(n_clusters=3, init='k-means++', random_state=170)
kmeans = kmeans.fit(PCA_components.iloc[:,:2])
kmeans
```

Out[163]:
```
KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
       n_clusters=3, n_init=10, n_jobs=None, precompute_distances='aut
o',
       random_state=170, tol=0.0001, verbose=0)
```

In [164]:
```
print("The cluster centroids are: \n", kmeans.cluster_centers_)
print("Cluster_label:\n", kmeans.labels_)
```

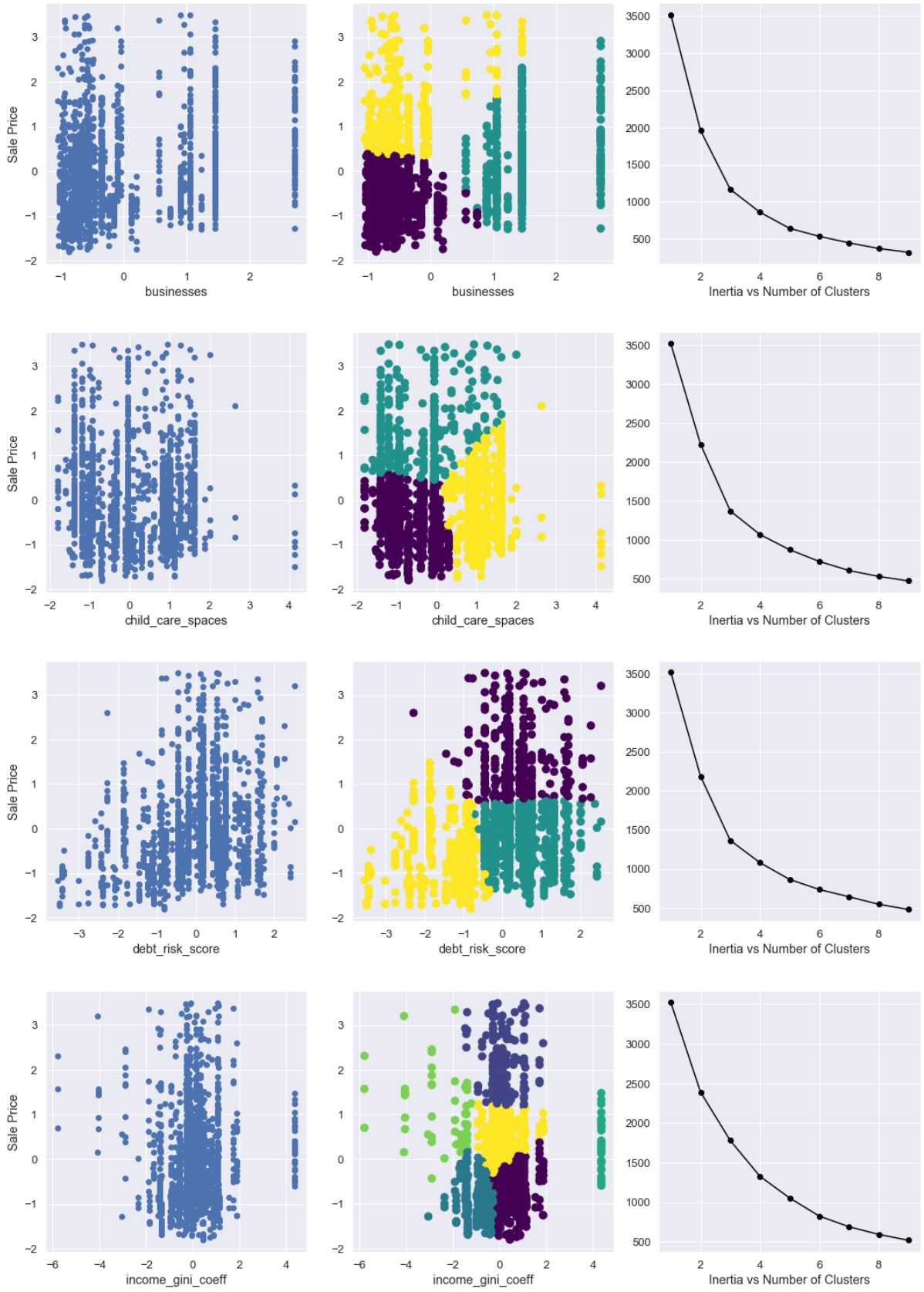
```
The cluster centroids are:
 [[-1.71827824  1.30489472]
 [ 3.50066172  0.21392529]
 [-1.24151665 -1.44093781]]
Cluster_label:
 [1 1 2 ... 1 0 2]
```

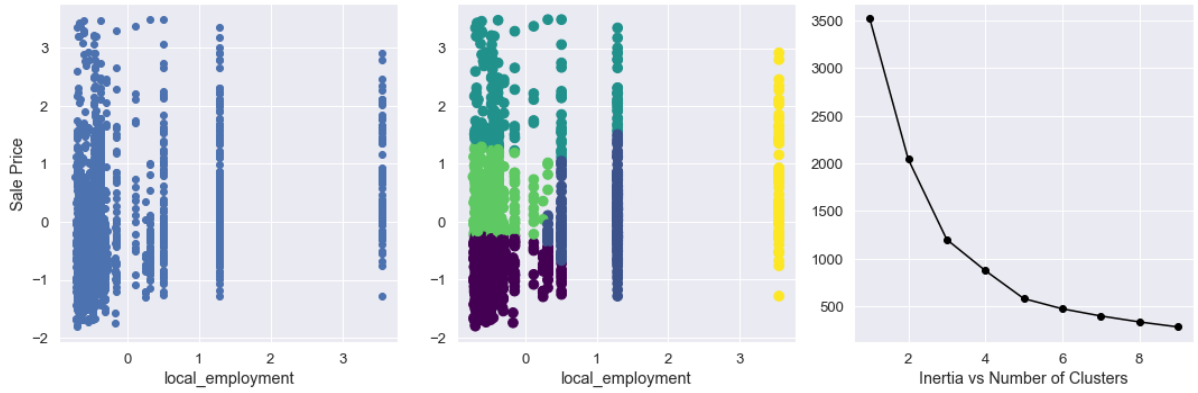
In [165]:
```
colors = ['blue','red']
plt.scatter(PCA_components[0],
            PCA_components[1],
            c=kmeans.labels_,
            cmap='viridis',
            s=75)
plt.show()
```

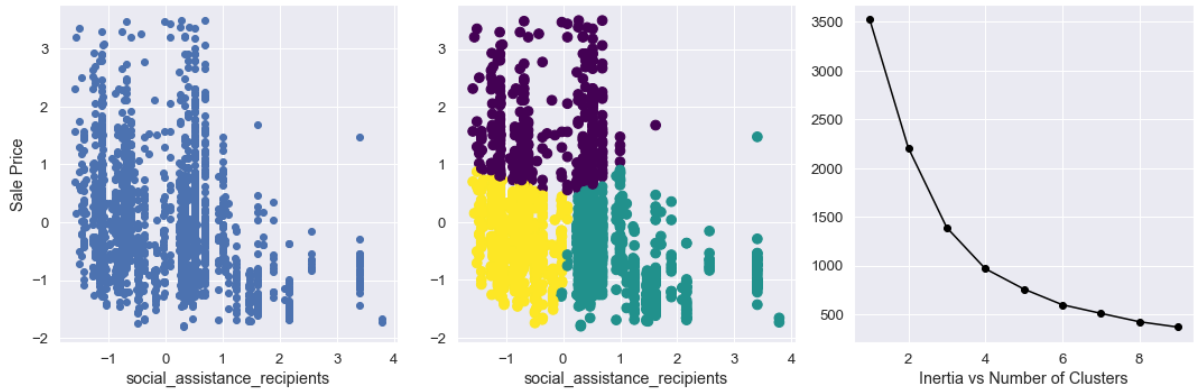


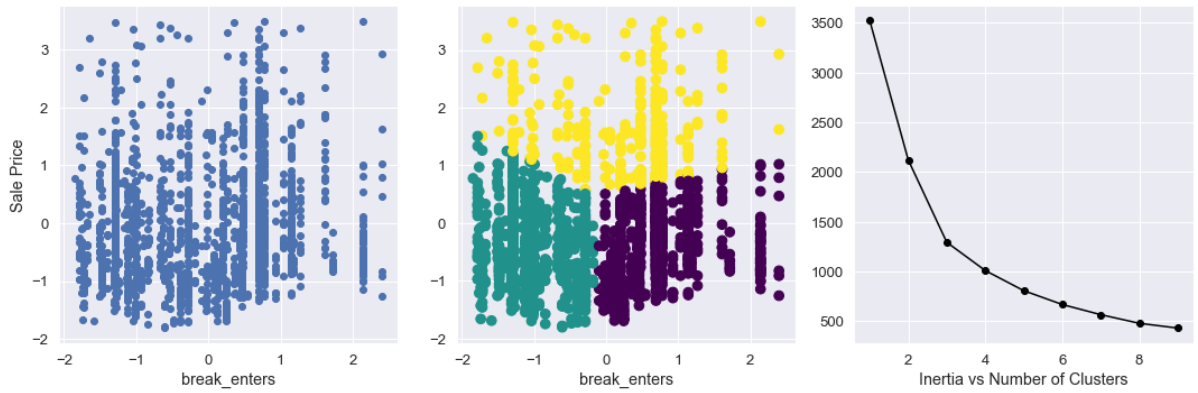
A.17 KMEANS CLUSTERING FOR ALL SOCIO-ECONOMIC VARIABLES VS SALE PRICE

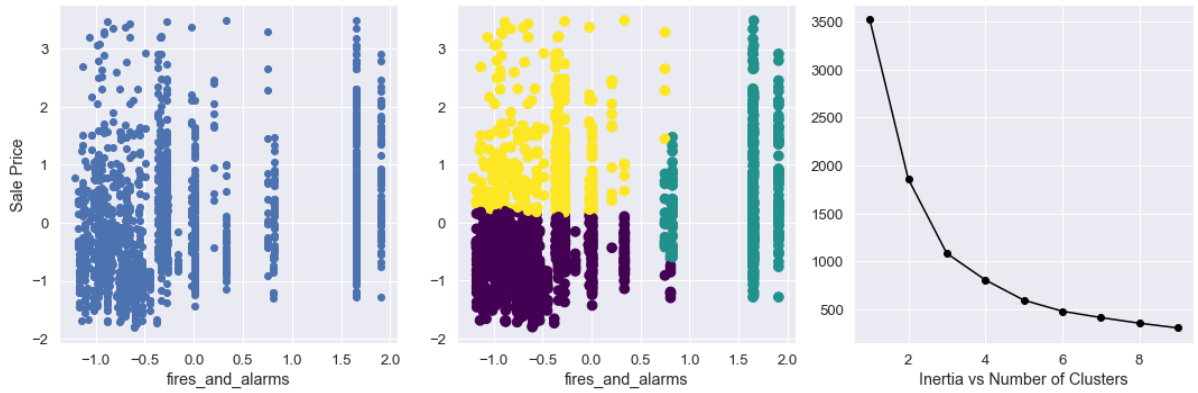
In [166]:
```
sev = condos_cleaned.iloc[:,13:25]
sev['sale_price'] = condos_cleaned['sale_price']

scaled_features = sc.fit_transform(sev)

sev_norm = pd.DataFrame(scaled_features, index=sev.index, columns=sev.co
lumns)

k_values = [3,3,3,6,5,3,3,3,3,3,3,3]
for i in range(0, 12):
    f, (ax1, ax2, ax3) = plt.subplots(1,3, figsize=(15, 5))

    f.tight_layout()

    ax1.scatter(sev_norm.iloc[:,i],sev_norm.iloc[:,12])
    ax1.set_ylabel('Sale Price')
    ax1.set_xlabel(sev_norm.columns[i])

    inertias = []
    ks = range(1, 10)
    x_norm = sev_norm.iloc[:,[i,12]]
    for k in ks:
        model = KMeans(n_clusters=k)
        model.fit(x_norm)
        inertias.append(model.inertia_)

    kmeans = KMeans(n_clusters=k_values[i], init='k-means++', random_sta
te=170)
    kmeans = kmeans.fit(x_norm)

    ax2.scatter(sev_norm.iloc[:, i],
            sev_norm.iloc[:, 12],
            c=kmeans.labels_,
            cmap='viridis',
            s=75)
    ax2.set_xlabel(sev_norm.columns[i])

    ax3.plot(ks, inertias, '-o', color='black')
    ax3.set_xlabel('Inertia vs Number of Clusters')

    plt.show()
```

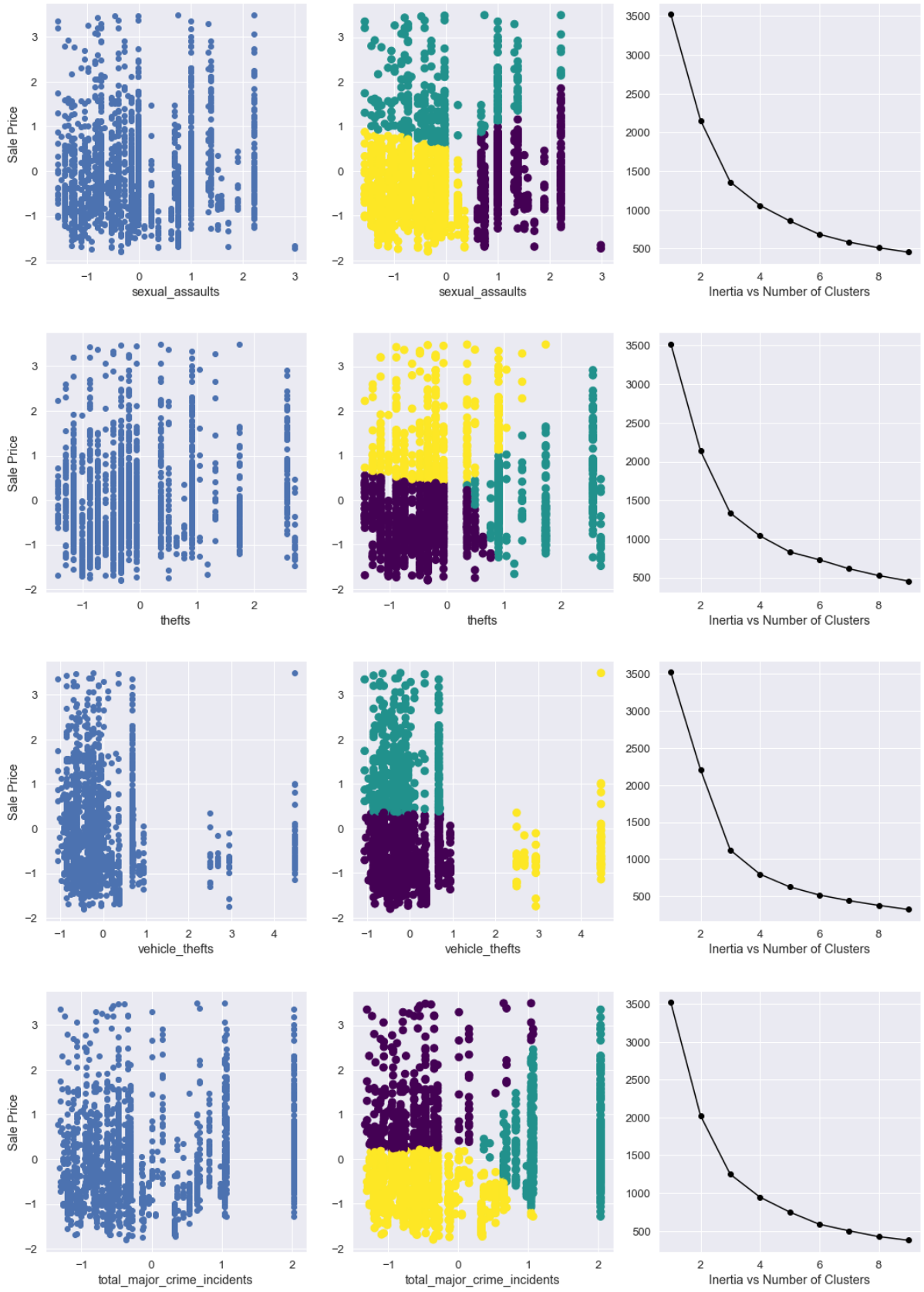

Sale Price
businesses
Inertia vs Number of Clusters
child_care_spaces
debt_risk_score
income_gini_coeff


Sale Price
local_employment
local_employment
Inertia vs Number of Clusters


Sale Price
social_assistance_recipients
social_assistance_recipients
Inertia vs Number of Clusters

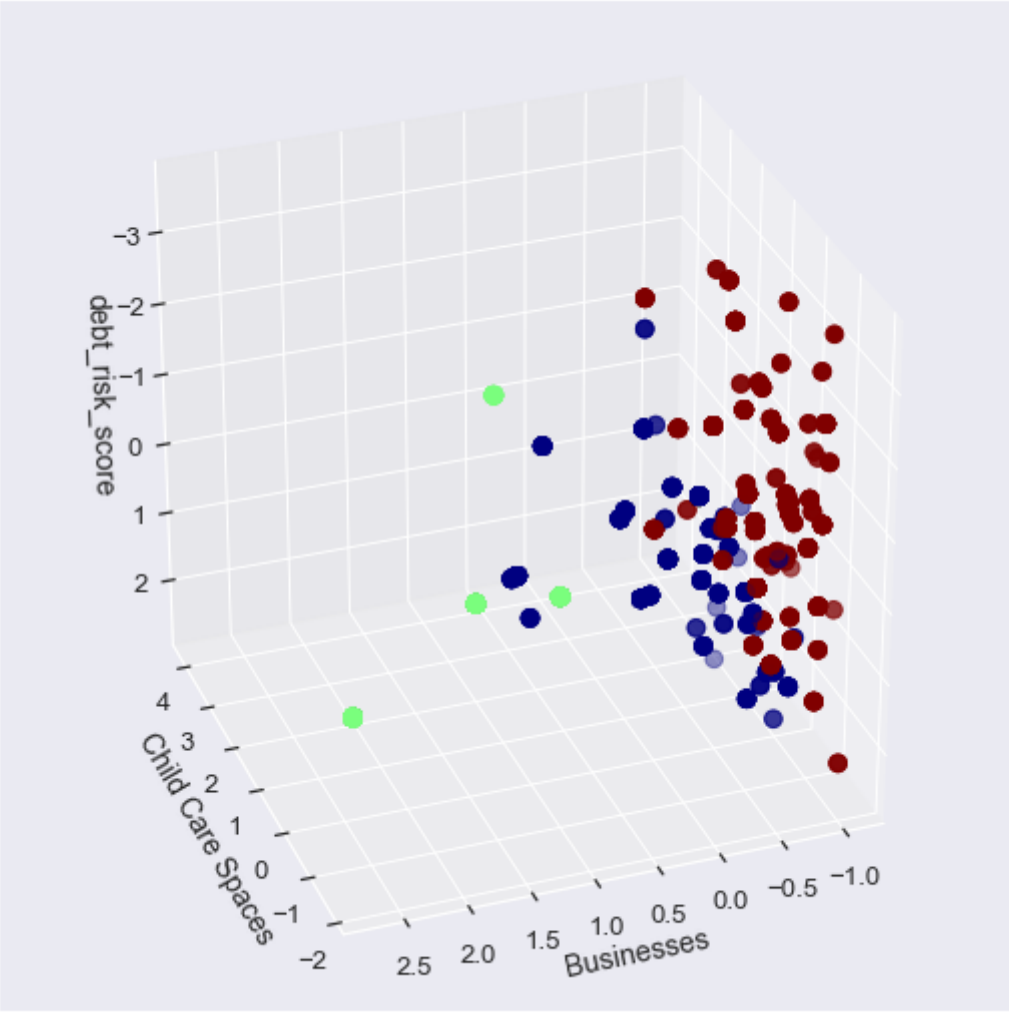

Sale Price
break_enters
break_enters
Inertia vs Number of Clusters


Sale Price
fires_and_alarms
fires_and_alarms
Inertia vs Number of Clusters

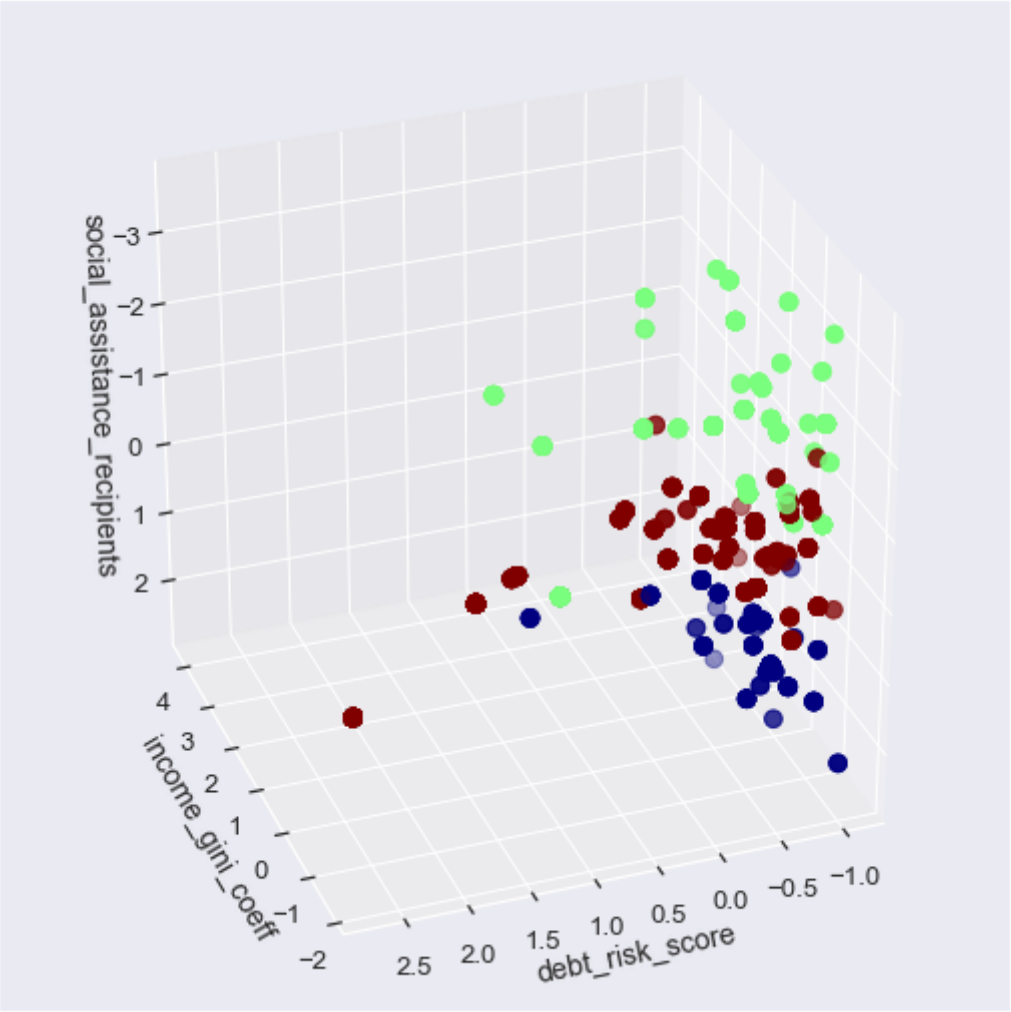

Sale Price
sexual_assaults
Inertia vs Number of Clusters
thefts
vehicle_thefts
total_major_crime_incidents

A.18 KMEANS CLUSTERING 3-D PLOTS FOR ALL SOCIO-ECONOMIC VARIABLES

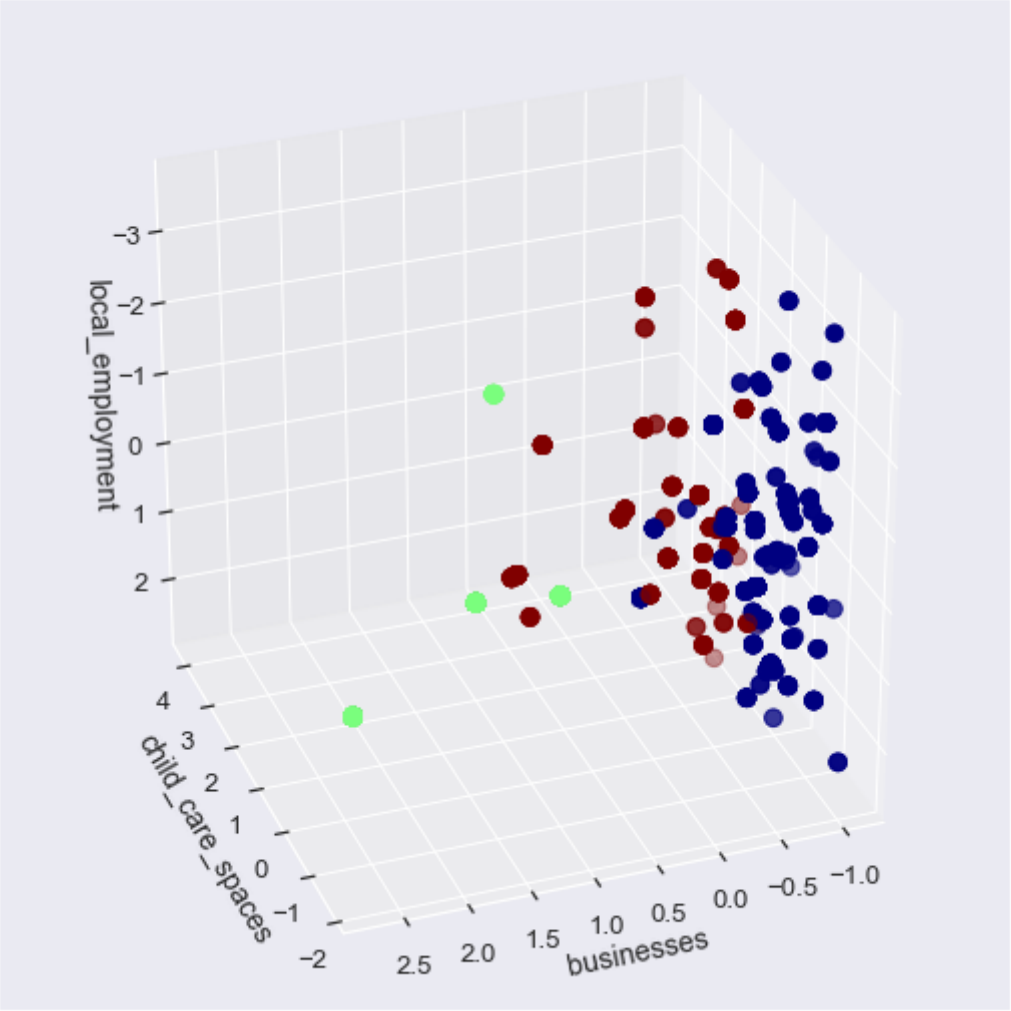
In [167]:
```
kmeans = KMeans(n_clusters=3, init='k-means++', random_state=170)
kmeans = kmeans.fit(sev_norm.iloc[:,0:3])


fig = plt.figure(1, figsize=(7, 7))
ax = Axes3D(fig, elev=-150, azim=110)
ax.scatter(sev_norm.iloc[:,0],
           sev_norm.iloc[:,1],
           sev_norm.iloc[:,2],
           c=kmeans.labels_,
           cmap='jet_r',
           s=75)
ax.set_xlabel('Businesses')
ax.set_ylabel('Child Care Spaces')
ax.set_zlabel('debt_risk_score')
plt.show()
```



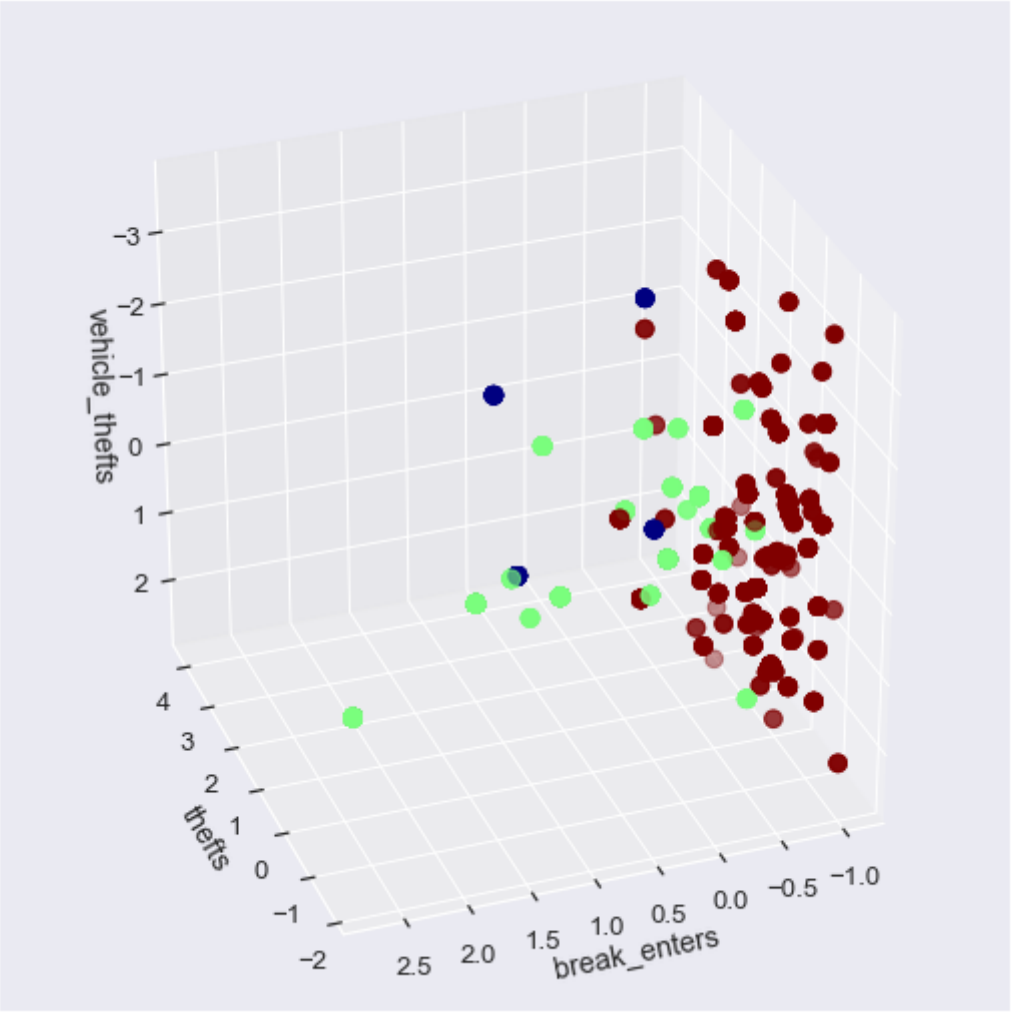
In [168]:
```
kmeans = KMeans(n_clusters=3, init='k-means++', random_state=170)
labels = ['debt_risk_score', 'income_gini_coeff', 'social_assistance_rec
ipients']
kmeans = kmeans.fit(sev_norm.loc[:,labels])


fig = plt.figure(1, figsize=(7, 7))
ax = Axes3D(fig, elev=-150, azim=110)
ax.scatter(sev_norm.iloc[:,0],
           sev_norm.iloc[:,1],
           sev_norm.iloc[:,2],
           c=kmeans.labels_,
           cmap='jet_r',
           s=75)
ax.set_xlabel(labels[0])
ax.set_ylabel(labels[1])
ax.set_zlabel(labels[2])
plt.show()
```



In [169]:
```
kmeans = KMeans(n_clusters=3, init='k-means++', random_state=170)
labels = ['businesses', 'child_care_spaces', 'local_employment']
kmeans = kmeans.fit(sev_norm.loc[:,labels])


fig = plt.figure(1, figsize=(7, 7))
ax = Axes3D(fig, elev=-150, azim=110)
ax.scatter(sev_norm.iloc[:,0],
           sev_norm.iloc[:,1],
           sev_norm.iloc[:,2],
           c=kmeans.labels_,
           cmap='jet_r',
           s=75)
ax.set_xlabel(labels[0])
ax.set_ylabel(labels[1])
ax.set_zlabel(labels[2])
plt.show()
```

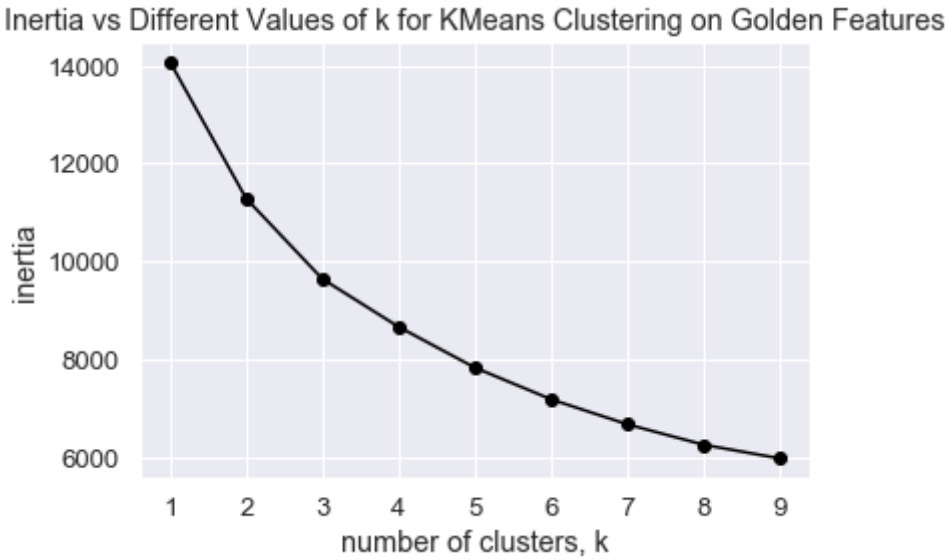


In [170]:
```
kmeans = KMeans(n_clusters=3, init='k-means++', random_state=170)
labels = ['break_enters', 'thefts', 'vehicle_thefts']
kmeans = kmeans.fit(sev_norm.loc[:,labels])


fig = plt.figure(1, figsize=(7, 7))
ax = Axes3D(fig, elev=-150, azim=110)
ax.scatter(sev_norm.iloc[:,0],
           sev_norm.iloc[:,1],
           sev_norm.iloc[:,2],
           c=kmeans.labels_,
           cmap='jet_r',
           s=75)
ax.set_xlabel(labels[0])
ax.set_ylabel(labels[1])
ax.set_zlabel(labels[2])
plt.show()
```



A.19 KMEANS FOR SELECTED FEATURES

In [171]:
```
k_data = condos_cleaned[['yearly_taxes', 'businesses',
       'square_feet', 'break_enters', 'parking_places', 'park_distance',
       'social_assistance_recipients', 'transit_distance']]

k_data_norm = sc.fit_transform(k_data)



ks = range(1, 10)
inertias = []
for k in ks:
    # Create a KMeans instance with k clusters: model
    model = KMeans(n_clusters=k)

    # Fit model to samples
    model.fit(k_data_norm)

    # Append the inertia to the list of inertias
    inertias.append(model.inertia_)

plt.plot(ks, inertias, '-o', color='black')
plt.title('Inertia vs Different Values of k for KMeans Clustering on Gol
den Features')
plt.xlabel('number of clusters, k')
plt.ylabel('inertia')
plt.xticks(ks)
plt.show()
```



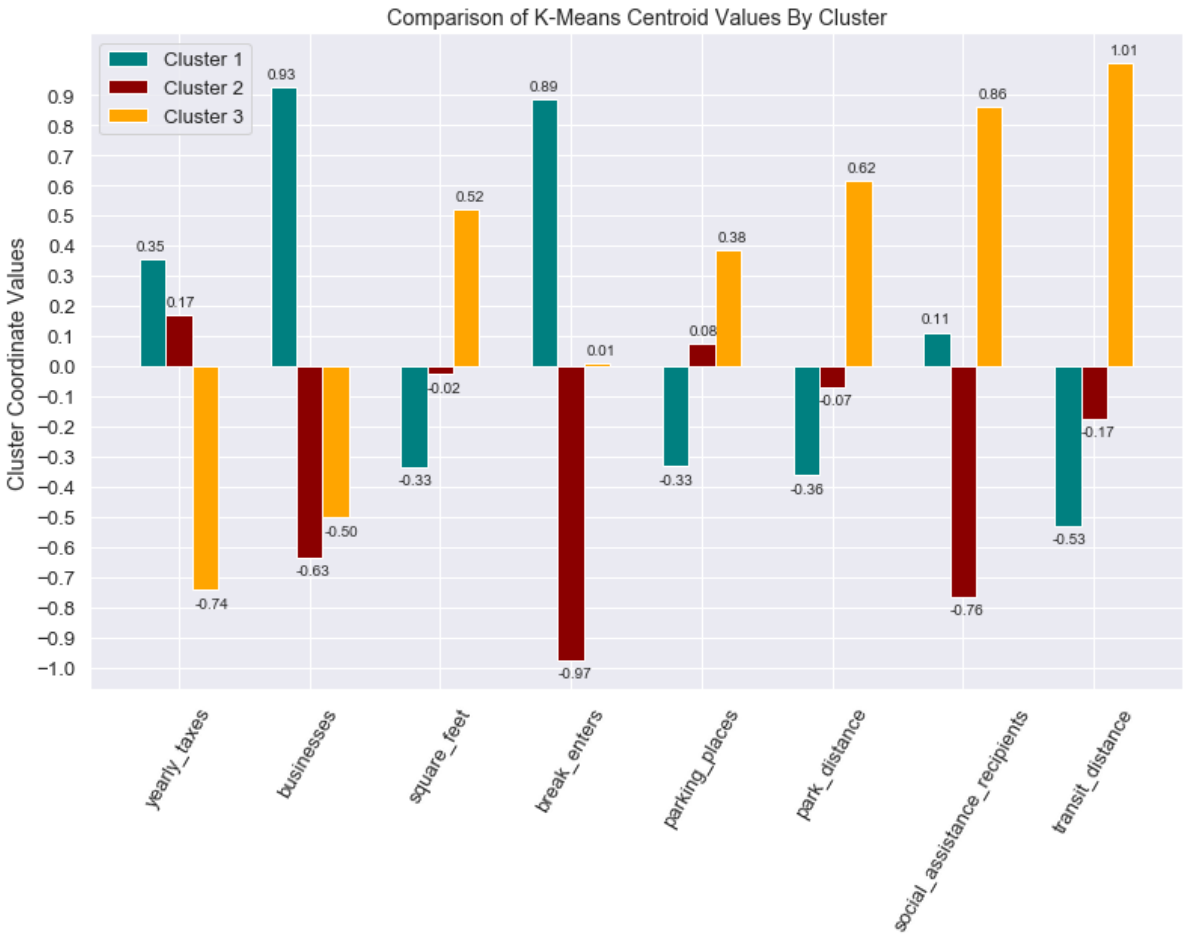
In [172]:
```
kmeans = KMeans(n_clusters=3, init='k-means++', random_state=170)
kmeans = kmeans.fit(k_data_norm)

print("The cluster centroids are: \n", kmeans.cluster_centers_)
print("Cluster_label:\n", kmeans.labels_)
```

```
The cluster centroids are:
 [[ 0.35332912  0.92694473 -0.33396801  0.88771086 -0.33224556 -0.35981
584
   0.11233273 -0.52940073]
 [ 0.16972456 -0.63407712 -0.02440088 -0.97458145  0.07524721 -0.068826
93
  -0.76466661 -0.17382368]
 [-0.74351931 -0.5029217   0.52033547  0.0110638   0.38416734  0.617660
97
   0.86160052  1.00608942]]
Cluster_label:
 [0 0 1 ... 0 1 2]
```

A.20 COMPARISON OF CLUSTER CENTROIDS

In [173]:
```
label = ['yearly_taxes', 'businesses',
       'square_feet', 'break_enters', 'parking_places', 'park_distance',
       'social_assistance_recipients', 'transit_distance']


plt.figure(figsize=(13, 8))

ind = np.arange(len(label))
width = 0.2

p1 = plt.bar(ind - 0.3, kmeans.cluster_centers_[0], width, color="teal",
align="edge")
p2 = plt.bar(ind, kmeans.cluster_centers_[1], width, color="darkred")
p3 = plt.bar(ind + 0.2, kmeans.cluster_centers_[2], width, color="orang
e")


plt.ylabel('Cluster Coordinate Values')
plt.xticks(ind, label, rotation=60)
plt.yticks(np.arange(-1, 1, 0.1))
# plt.legend(bbox_to_anchor=(0, 1), loc='upper right', ncol=1)
plt.legend((p1[0], p2[0], p3[0]), ('Cluster 1', 'Cluster 2', 'Cluster 3'
))

for v in ind:
    first = kmeans.cluster_centers_[0][v]
    second = kmeans.cluster_centers_[1][v]
    third = kmeans.cluster_centers_[2][v]
    padding = 0.03
    neg_padding = 0.06
    fs = 10
    plt.text(v-0.32, first - neg_padding if first < 0 else first + paddi
ng , "%.2f" % first, fontsize=fs)
    plt.text(v-0.1, second - neg_padding if second < 0 else second + pad
ding , "%.2f" % second, fontsize=fs)
    plt.text(v+0.12, third - neg_padding if third < 0 else third + paddi
ng , "%.2f" % third, fontsize=fs)

plt.title('Comparison of K-Means Centroid Values By Cluster')

plt.show()


## Alter:
# index = pd.Index(label)

# data = {
#     'Cluster 1': kmeans.cluster_centers_[0],
#     'Cluster 2': kmeans.cluster_centers_[1],
#     'Cluster 3': kmeans.cluster_centers_[2],
# }

# df = pd.DataFrame(data, index=index)
# ax = df.plot(kind='bar', stacked=True, figsize=(13, 8), color=['teal',
'darkred', 'orange'], width=0.3)
# # ax.set_ylabel('foo')
```

```
# # plt.savefig('stacked.png')
# plt.yticks(np.arange(-1, 1, 0.1))

# plt.show()
```



A.21 CLUSTER PRICE VARIATIONS

In [174]:
```
# Apply cluster labels to original dataframe
condos_cleaned['cluster'] = kmeans.labels_
kmeans_condos = condos_cleaned
kmeans_condos['cluster'] = kmeans_condos['cluster'] + 1
```

In [175]:
```
kmeans_condos.groupby('cluster')[['sale_price']].count()
```

Out[175]:

	sale_price
cluster	
1	676
2	622
3	462

In [176]:
```
kmeans_condos.groupby('cluster')[['sale_price']].mean()
```

Out[176]:

	sale_price
cluster	
1	651267.590237
2	619307.041801
3	473711.183983

In [177]:
```
kmeans_condos.groupby('cluster')[['sale_price']].std()
```

Out[177]:

	sale_price
cluster	
1	162389.010164
2	171428.480550
3	129766.139238

In [178]:
```
kmeans_condos['y'] = 0
kmeans_condos['color']='darkred'
kmeans_condos.loc[kmeans_condos.cluster == 1, 'y'] = 1
kmeans_condos.loc[kmeans_condos.cluster == 2, 'y'] = 0
kmeans_condos.loc[kmeans_condos.cluster == 3, 'y'] = -1

kmeans_condos.loc[kmeans_condos.cluster == 1, 'color'] = "teal"
kmeans_condos.loc[kmeans_condos.cluster == 2, 'color'] = "darkred"
kmeans_condos.loc[kmeans_condos.cluster == 3, 'color'] = "orange"

# markers = {1: "s", 2: "X", 3:"o"}
plt.figure(figsize=(10, 5))
arks=['x','d','o']
for i in range(1,4):
    x = kmeans_condos.loc[kmeans_condos.cluster == i,'sale_price']
    y = kmeans_condos.loc[kmeans_condos.cluster == i,'y']
    color = kmeans_condos.loc[kmeans_condos.cluster == i,'color']
    plt.scatter(x, y,  s=30, c=color, marker=marks[i-1])
plt.yticks([1,0,-1], ['Cluster 1','Cluster 2','Cluster 3'])
plt.xlabel('Sale Price (CAD$)')
plt.legend(('Cluster 1', 'Cluster 2', 'Cluster 3'))
plt.title('Variation in Sale Price Across Clusters')
plt.show()
```

A.22 DISTANCES BETWEEN CLUSTER CENTROIDS

In [179]:
```
# calculating distances between centroids

import math

distances = [[0,0,0],[0,0,0],[0,0,0]]
for i in range(0,3):
    for j in range(0,3):
        if i != j:
            sum_values = 0
            for k in range(0,8):
                sum_values += ( kmeans.cluster_centers_[i][k] - kmeans.c
luster_centers_[j][k] )**2
            distance = math.sqrt(sum_values)
            distances[i][j]=distance
distances
```

Out[179]:
```
[[0, 2.6796925273706806, 3.0222143511279795],
 [2.6796925273706806, 0, 2.5928916972929277],
 [3.0222143511279795, 2.5928916972929277, 0]]
```

In [180]:
```
plt.figure(figsize=(5, 5))
sns.set(font_scale=1.2)
sns.heatmap(distances,
            cmap='Blues', vmax=3, vmin=0, linewidths=0.1,
            annot=True, annot_kws={"size": 15}, square=True,
           xticklabels=[1,2,3], yticklabels=[1,2,3] );
plt.title('Heatmap of Distances Between Cluster Centroids', y=1.15)
plt.xlabel('Cluster')
plt.ylabel('Cluster')
```

Out[180]:
```
<matplotlib.text.Text at 0x1a269224e0>
```