Airbnb Toronto Data Science Project

Team 4

Deliverable 2 Report : Data Exploration and Data cleaning

Sheridan College

Date: October 28th 2021

Table of Contents

TEAM 4-DELIVERABLE 2 : Data cleaning and Exploratory Data Analysis Report

Purpose of Deliverable 2

The goal of deliverable 2 of our project was to prepare a clean Toronto Airbnb dataset and provide data visualizations that would enable stakeholders to understand why essential steps characteristic of data cleaning; such as the removal of features and imputation took place. As noted in our earlier report, we believed that features in the dataset that are predictive of price of stay per night and review ratings score would enable the Airbnb company president and relevant stakeholders such as Airbnb hosts to develop business strategies such as marketing to emphasize on these features when providing Airbnb service to existing and potential customers. We expect a dramatic increase in Airbnb revenue based on the features to be presented in the results of this study. We utilized the programming language, Python for the steps required in data cleansing and Tableau (version 2021.3) for the Exploratory data analysis (EDA).

The Toronto Airbnb dataset presented a challenge in the data cleaning process because of four main reasons. First, we believe that there were too many variables/features (74, including listing ID) in the dataset. Upon closer examination, we identified features such as ‘listing_url’ that were not relevant to our business objective. We have also noted some variables that was redundant when comparing to other variables included in the dataset. We also removed variables if they contained more than 40% missing values. The complexity of the dataset is provided in Figure A (see Appendix) – where total number of 19,343 host listings included in our dataset is visually shown, imposed under a Toronto map.

Overview of the data cleaning process and relevance of EDA

Exploratory analysis (EDA) of the original dataset (uncleaned) was first conducted through the use of Tableau. Graphs and plots to identify relevant features and the relationships between the features were generated (refer to Part III of this report). Secondly, as is common with real world data, several features contained missing values and as result, imputation methods were implemented in Python to replace the missing values. Thirdly, outliers were identified in the dataset using both Tableau and Python. We detected both negative and positive skewness in several features of the dataset. The interquartile range (IQR) method was selected to remove the outliers.

Finally, using feature engineering, we created three features that we think is relevant to our business objective. One feature created in the dataset is 'former city' based on Toronto districts. There were more than 40 neighborhoods and we decided to consolidate each neighborhood to a city. In Python, we web scraped the Wikipedia page https://en.wikipedia.org/wiki/Toronto, under the section entitled "Neighborhoods" to collect all the districts and associated neighbourhoods in Toronto. Using this method, we were able to group the neighbourhoods by district. Python screenshot below displays a sample list of neighbourhoods grouped by Toronto district (i.e., 'former_city').

```
[5]: df2[['price','review_scores_rating','neighbourhood_cleansed','former_city']].sample(10)
```

[5]:

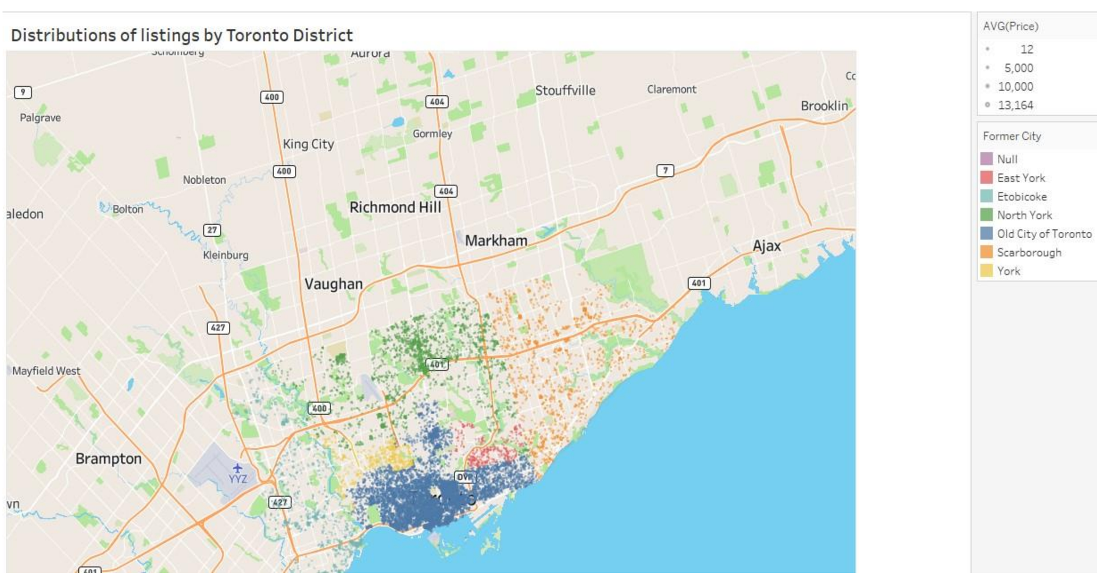
	price	review_scores_rating	neighbourhood_cleansed	former_city
13980	$250.00	NaN	Cabbagetown-South St.James Town	Old City of Toronto
13795	$75.00	60.0	Leaside-Bennington	East York
16291	$80.00	NaN	Lawrence Park South	Old City of Toronto
17805	$45.00	NaN	Dorset Park	Scarborough

Python was used to generate the cleaned dataset that would be used for (1) EDA in Tableau for visualization and (2) for implementation of the machine learning model. We used Tableau for both uncleaned and cleaned datasets to see the effect of cleaning the data. Below, we provide in detail the steps taken to clean our Airbnb dataset using Python (Part 1 & Part II) and the necessary EDA in Tableau for the uncleaned dataset (Part III).

Part I
Steps in the removal of features

To begin, our EDA using Tableau to characterize price variation by neighborhood or review ratings score by neighborhood was difficult to interpret as there are more than 70 neighbourhoods in Toronto. Using Python, we web scraped the Wikipedia page of the city of Toronto to collect all the districts and relevant neighborhoods. As a result, we were able to group the neighbourhoods by district. A merge function was implemented in Python to combine the web scraped table with our uncleaned dataset. Using this, a new feature called 'Former City' was added into the Toronto Airbnb dataset. The new feature 'Former City', consists of six Toronto districts namely Etobicoke, North York, East York, Old City of Toronto, York and Scarborough. Each of these Toronto districts have more than five neighbourhoods, with Old City of Toronto containing the most neighborhoods. For example, East York covers Broadview North, Danforth East York, Leaside-Bennington, O'Connor-Parkview, Old East-York, Thorncliffe Park and Woodbine-Lumsden. In *Figure B* below, you can see how listings can be viewed better by district rather than by neighborhood as shown with the uncleaned dataset in Tableau in *Figure A* (see appendix).

<u>Figure B. Listings by District in Toronto</u>



In *Table 1* in the Appendix, we have highlighted features in the original or uncleaned dataset in terms of the basis of removal (i.e., irrelevant, too many missing values and redundancy). There were 74 features (Id being a primary key) in the uncleaned dataset. Features were removed because of redundancy and irrelevance to our business objective and if data has more than 40% missing values. We have also highlighted features that had NaN values.

We used Python to reduce the 74 features to a set of features that were relevant to our machine learning model. We first did a correlation plot with the target variable being price (see *Figure 2* in Appendix). In general, there were few features correlated to price, highlighting the importance of data cleaning. Our second step in data cleaning involved removing features not relevant to our business objective.

Moreover, several features had missing values, as can be seen from *Table 2* in Appendix. For example, 'host_neighbourhood' had 15,552 non-null values, and as a result, 'host_neighbourhood' had 3,791 missing values (i.e., 19,343-15,552 = 3,791). We removed *id*, *'listing_url'*, *'scrape_id'*, *'last_scraped'*, *'name'*, *'description, neighbourhood_overview'*, *'picture_url'*, *'host_id'*, *'host_url'*, *'host_name'*, *'host_thumbnail_url'*. Textual information such as description and url weblinks variables were also removed. Features such as 'minimum_nights_avg_ntm' and 'maximum_nights_avg_ntm' were removed because of redundancy in the data. That is, both features had duplicate values.

Thirdly, we removed features for which there were no values. Fourthly, we removed features that had more than 40% missing values. These features were *'host_about', 'neighbourhood_group cleansed', 'bathrooms', 'calendar_updated', 'license'*. The 40% missing value threshold is what most businesses use for removal of missing values in datasets. Refer to the figure below for the output from Python, in terms of percentage of missing values being higher than 40% for five features.

```
Columns with more than 40% missing values:
 ['host_about' 'neighbourhood_group_cleansed' 'bathrooms'
 'calendar_updated' 'license']
```

Imputation of missing values in Categorical features

As a result of the above steps of data cleaning, we had 37 features in our dataset. There were nine categorical and 28 numeric features. Refer to *Table 3* in Appendix for profile of features with data type. To replace NaN values in the categorical features with data type of object, we imported the SimpleImputer from Sklearn Python library to impute missing values with most frequent. The SimpleImputer is a scikit-learn class. We implemented the ‘frequent’ strategy from SimpleImputer to impute the missing values. Host response time, host response rate and host acceptance rate were categorical features for which NaN values were imputed.

```
df.head(3)
```

	room_type	host_since	host_response_time	host_is_superhost	host_acceptance_rate	host_response_rate
0	Entire home/apt	2008-08-08	NaN	f	NaN	NaN
1	Entire home/apt	2011-06-07	NaN	f	NaN	NaN
2	Entire home/apt	2012-06-01	NaN	t	100%	NaN

```
df.head(3)
```

	room_type	host_since	host_response_time	host_is_superhost	host_acceptance_rate	host_response_rate
0	Entire home/apt	2008-08-08	within an hour	f	100%	100%
1	Entire home/apt	2011-06-07	within an hour	f	100%	100%
2	Entire home/apt	2012-06-01	within an hour	t	100%	100%

Outlier detection and transformation of data

As can be seen from the box plot and histogram distributions generated in Python and for the selected features in Tableau, displayed in *Figure 4* in appendix*;* most of the data contained within features was positively or negatively skewed. Several outliers were present within each feature. The outliers were noted from use of describe function and confirmed from the box plots and visualization of the distributions. For example, in the unclean or original dataset; for price, the maximum value is $13,164 but the average price is $141.28 and 75% are below $150. Similarly, for ‘minimum_nights’, the maximum value is 1125, but the average is 10 and 75% below 5. For both bedrooms and beds, mean was 1,39 and 1.63 respectively;

maximum being 16 and 17. Outliers were detected in the features: *‘price’*, *‘accommodates’*, *‘host_total_listings_count’*, *‘reviews per month’*, *‘bedrooms’*, *‘beds’*, *‘minimum_nights’*, *‘maximum_nights’*, *‘number of reviews’*, *‘number_of_review_ltm’*, *‘review_scores_accuracy’*, *‘review_scores_cleanliness’*, *‘review_scores_checkin’*, *‘review_scores_communication’*, *‘review_scores_value, reviews_per_month’*.

Three options for removing outliers were available to us, the Interquartile range (IQR) method, Winsorize and Log transform. Log transformation reduces the skewness of data and tries to make it normal. However, we had zero or negative values contained within some of our features. With winsorizing, any value of a variable above or below a percentile k on each side of the variables’ distribution is replaced with the value of the k-th percentile itself. For example, 90% winsorization means the replacement of the top 5% and bottom 5% of the data. The top 5% of the data is replaced by the value of the data at the 95th percentile and the value of the bottom 5% of the data is replaced by the value of the data at the 5th percentile. However, winsorization would not be feasible as it could not be applied properly to features in our data and would replace values in the dataset.

For the above reasons, we selected the IQR method. For the IQR method, the third quartile (75th percentile) and first quartile (25th percentile) is subtracted to get the IQR. Any numbers less than the First quartile subtracted from 1.5 x IQR is considered an outlier and removed; whereas any number greater than the third quartile subtracted from 1.5 x IQR is considered an outlier and removed. We applied the IQR method using Python, to all the numerical features in our dataset to remove outliers. Figure 5 in the appendix shows outliers in Price and Minimum nights and after removal of outliers, using the describe function. The data frame named ‘df4’ contained features such as price with outliers, whereas the data frame named ‘df’ contains features such as price without outliers.

Part II

Feature Engineering

Table 4 in the appendix represents the features in the cleaned dataset that were feature engineered. These features were ‘host_since_length’ (length of time the host has been a host of an Airbnb), ‘labels_host_acceptance_rate’ (categorical variable containing levels of host acceptance rate) and ‘labels_host_response_rate’ (categorical variable containing levels of host response rate).To extract some useful features from our dataset we converted two relevant features (host response rate and host acceptance rate) to categorical features, using binning and labels. We felt the response rate of the host would have a strong influence on both price and review ratings score. Host response rate refers to the host responding to the request of the client or customer before renting Airbnb, during the stay in Airbnb. The requests from clients could be on inquiries on the host listing location and characteristics of the rental space. Since, price fluctuated with neighbourhood or Toronto district, we felt there would be a relationship with the response rate of the host. If there was a higher price for rent, it was likely that the host was responding on time to the inquiries of the customer. A more direct relationship would be between review ratings score and response rate of host. We felt customers would give a higher score on review ratings if the host responded often to their inquiries before the transaction and during the rental period. Host response rate was formatted with % sign removed and was binned into 5 bins and labels were assigned as in very strict, strict, accepting and very accepting. For ‘labels_host_response_rate’, five bins were also created with labels ‘very slow’, ‘slow’, ‘fast’, ‘very fast’.

Another categorical feature that was feature engineered was ‘labels_host_acceptance rate’. The feature ‘host acceptance rate’ was defined as an ‘object’ variable in python, with numbers ranging from 100% to 50%, 30% and 0%. We felt that in locations of Toronto and for listings where hosts showed a low acceptance rate there would be a strong relationship with both price of rent and review ratings score. Perhaps with a high acceptance rate, the price of rent was low which may reflect the quality of the rental space. In terms of review ratings score, perhaps higher review ratings score was provided to hosts who had a high acceptance rate.

Finally, we thought that a factor that a customer would consider when choosing to rent an Airbnb in Toronto would be the length of time the host has been a host of an Airbnb. Perhaps the customer considered the experience of the host as measured by the number of years the individual has been an Airbnb host as an important factor in renting the host's Airbnb. We split the date into year under the column "Host_since_year" and month using Python and then subtracted the current year being 2021 from "Host_since_year" to get the new feature "Host_length". *Figure 6* in the Appendix displays a sample of the values for the new features.

Imputation of missing values in Numerical features

There were NaN values present in some of our numerical features. We have highlighted these features in *Table 1*. Using Python code, we imported the library called IterativeImputer to enable an iterative imputer from skitlearn to impute NaN values in the numerical features in the Airbnb dataset, with the prediction method. The Iterative Imputer models the missing values based on a prediction compared to the other features in the data set. i.e., known variables are used as a train set and used to predict the test (missing) rows. Essentially missing values are predicted. Moreover, there is a sequential imputation of each feature which allows prior imputed values to be used as part of a model in predicting subsequent features (Reference: Machine learning mastery website).The profile of the cleaned dataset with features is shown in *Table 4* of the Appendix.

Dummy coding categorical variables

In anticipation of the implementation of Machine learning algorithms as in Regression with price being the target variable and a possible implementation of clustering of the data, categorical features were dummy coded using the get_dummies function in Python Pandas. The complete set of features including features which are dummy coded are presented in *Table 4* in the Appendix.

Part III

Exploratory Data Analysis

Using Tableau, we conducted an exploratory data analysis of both the original and cleansed Airbnb datasets.

EDA of original dataset

Before conducting the explanatory analysis of the original dataset, we needed to assess which variables or features were most correlated with our target variable being price and the other target variable being review scores rating. Our correlation plot generated in Python shown in the Figure 2 of the Appendix, showed 'accommodates' (i.e., capacity limit for property) (r = .26), 'bed' (i.e., number of beds in property) (r = .22) and 'bedrooms' (i.e., number of bedrooms in property) (r = .24) to be the three features that were more correlated to 'price' than other features. What was also interesting was that 'review ratings score' had almost no relationship with 'price' (r = .02).

Our EDA of the original data also showed there were missing values or NaN values for most of the features and there were outliers for most of the features, even in the target variables as in Price or Review Ratings Score. Our EDA in Tableau showed why Price was most strongly correlated to 'accommodates', 'bed' and 'bedrooms'. The three figures shown in Figure 7 of the Appendix show scatter plots of 'price' with the features 'accommodates', 'beds' and 'bedrooms'.

However, correlation does not measure the relationship between numerical and categorical variables. As a result, several categorical features could relate to average price or review ratings score. One feature that could also relate to price or review ratings score was the location of the Airbnb listing. When we plotted in Tableau the listings by neighbourhood, a bird's eye view gave us the impression that in the district called "Old City Toronto" there were more listings compared to other districts such as Scarborough and York as can be seen from *Figure B* in the appendix, combined with the fact that price seemed higher for listings in location close to waterfront neighborhood in Old City of Toronto than areas further away from the waterfront neighborhood, such as North York and Etobicoke. As a result, our first business query was: Does

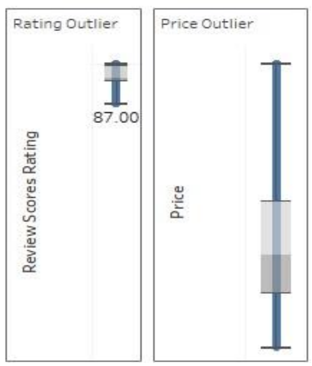
Average price and rating vary by the location of the Airbnb listing? As can be seen from the Figure below, price per day was much higher for listings in the Old City of Toronto ($158) compared to other districts. Lowest average price per listing was for listings in York ($92) and Scarborough ($96) which were located furthest from Old City of Toronto. Interestingly, review scores rating did not vary by location of listing, as shown from the figure below.

Our second business query was: Does average price and total host listings also vary by the room type of the property and by the district in Toronto? From our EDA of the unclean data, we found that Average price of listing in a district could also depend on the type of room (i.e., Entire home/apartment, Hotel room, Private room and Share room) and in the district in Toronto. As seen from the Figure 8 in the Appendix, there were more listings of Entire home/apartment in the Old City of Toronto, North York and Scarborough compared to other districts. Moreover, the Old City of Toronto seemed to have listings which offered a more diverse distribution of room type. Finally, the average price of Entire home/apartment was the highest in the Old City of Toronto and lower in York, Etobicoke, and East York.

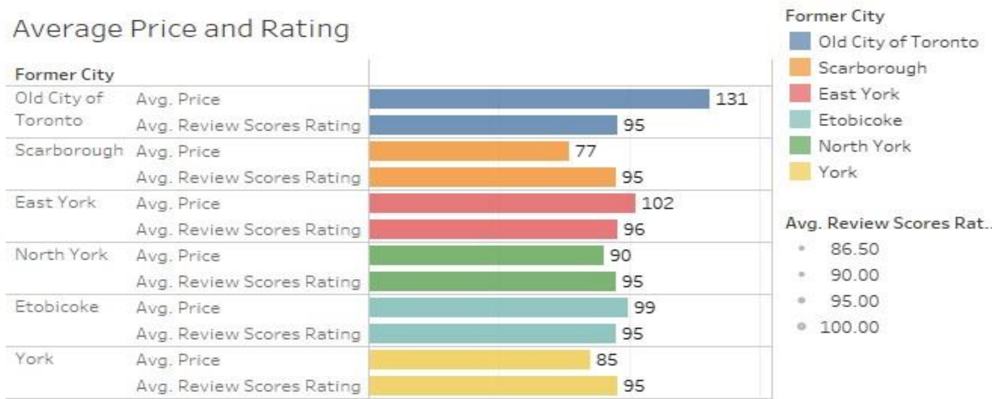
Figure 9 in the Appendix below displays the Tableau dashboard with the uncleansed data being the data source and shows Airbnb house/room ratings between August 2009 to September 2020 in the Greater Toronto Area. Using the cleansed data set, total listings count is at 19,339. Old City of Toronto holds the highest average price of $158 per night and highest average review score at 95. Outliers are shown in the rating and price outlier boxplots.

Visualization from Tableau of Clean data

We also did an EDA of the cleaned Airbnb dataset; however, the details of the EDA will be provided in deliverable 4 for which dashboards of the cleaned dataset will be presented. The main purpose of showing a brief EDA of the clean data in this report was to provide evidence that missing values and outliers in most of the features were removed using data cleaning processes such as imputation in Python. For example, there were no outliers for both review ratings score and price after the data cleaning process as evidenced by the figure below from visualization in Tableau.



Interestingly, the cleaned data did show that the general patterns present in the original dataset remained the same even after data cleaning. For example, the average price for a host Airbnb listing was still highest in Old City of Toronto compared to other districts of Toronto, as can be seen from the figure below.



We also did a visualization of the relationship between our two features: host response rate and host acceptance rate engineered using Python using a binning process. As can be seen from ***Figure 10*** in the Appendix, host acceptance rate does vary by district in Toronto. In addition, average price does seem to relate to host acceptance rate in certain districts of Toronto, but not other districts such as the Old City of Toronto.

Conclusion

In summary, our correlation plot did not show strong correlations between 'price' and other features in our Airbnb data set. However, when exploring the dataset, we noticed that price of rental per day fluctuated with location of the listing in Toronto. Moreover, there were several missing values and outliers in most of the features as noted from our EDA in Python and Tableau. We conducted a detailed data cleaning process using Python to remove irrelevant features, features with missing values more than the 40% threshold, impute missing values with Sklearn Simple and Iterative imputers, create new features and dummy code the cleaned data set. We also performed a brief EDA of the cleaned data exported from Python to confirm that missing values were inputed, outliers were removed and that there were some general patterns in the preserved such as fluctuation of price by location in Toronto.

In conclusion, we believe that we have prepared a clean Toronto Airbnb dataset and our use of EDA in Tableau and Python will enable stakeholders to understand why essential steps characteristic of data cleaning such as the removal of features and imputation took place. The next step as part of Deliverable

3 report will be to apply a suitable machine learning model such as regression to the clean dataset to identify features that predict price per day of a host listing. Importantly, we will determine using an appropriate machine learning model which features predict whether the customer rents an Airbnb listing for a given rental price.

References

https://www.kaggle.com/robinkongninglo/toronto-airbnb-dataset

https://machinelearningmastery.com/iterative-imputation-for-missing-values-in-machine-learning/

https://en.wikipedia.org/wiki/Toronto

Appendix

Figure A. Listings by Neighborhood in Toronto as displayed in Tableau

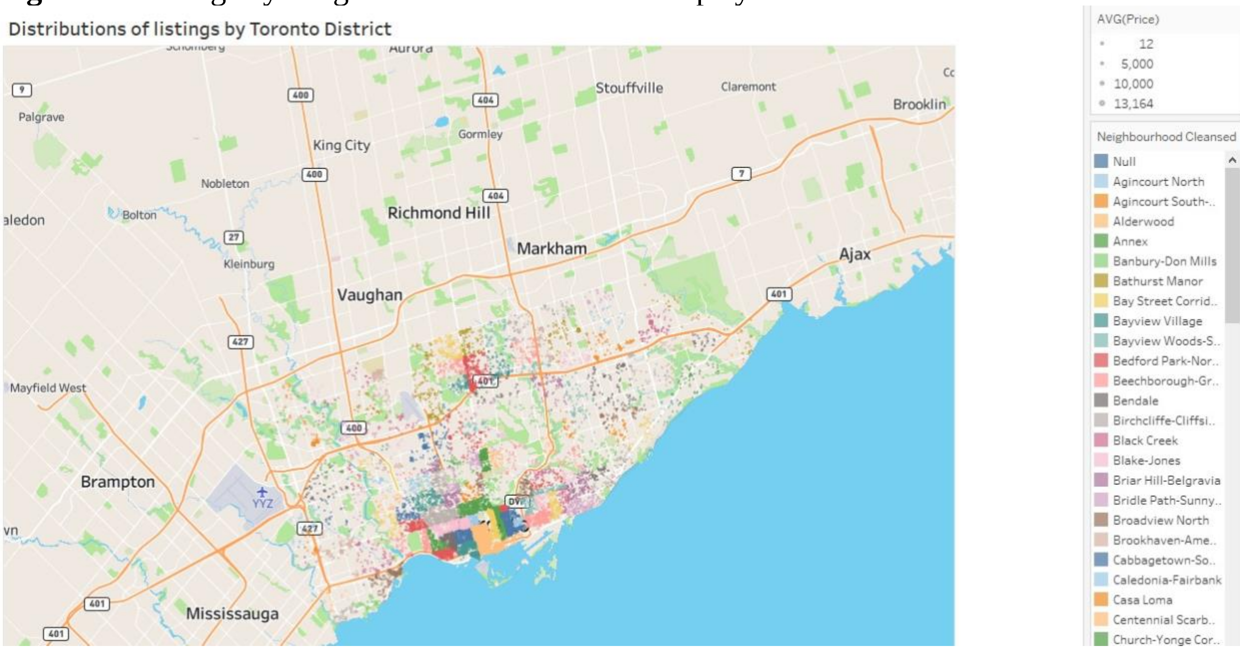


Figure B. Listings by District in Toronto

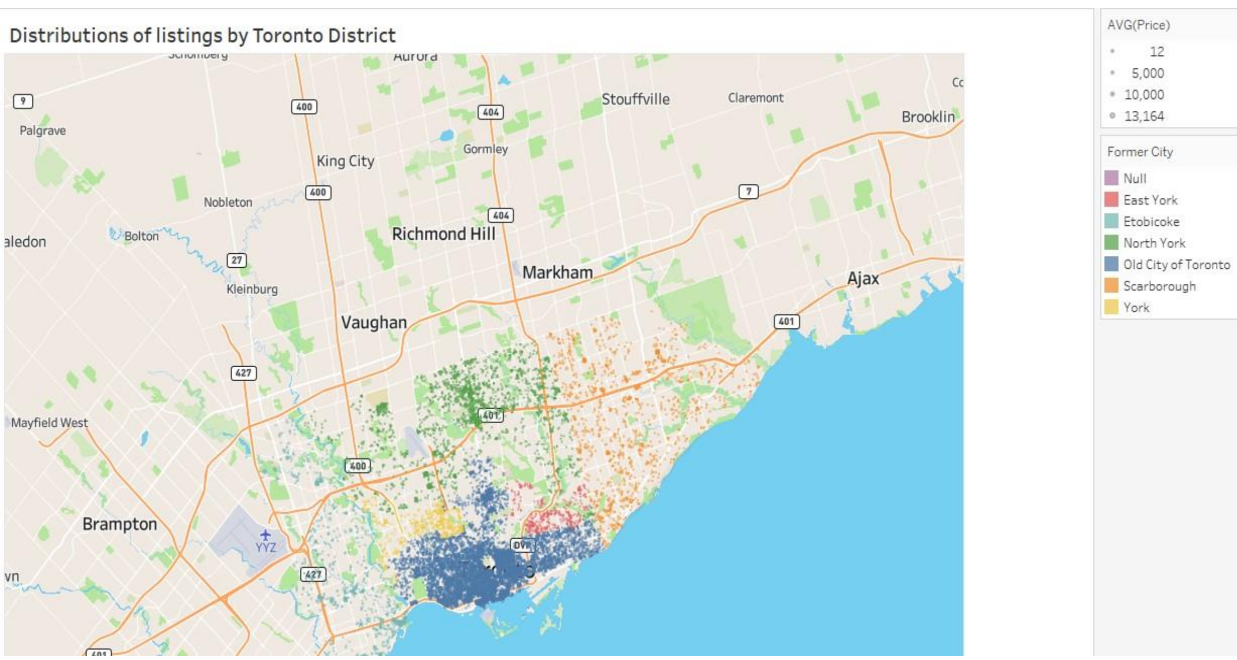


Table 1. Data dictionary showing the fields or features in the AirBnb Data Set with description and the type of feature is stated.

In total there were 73 features in the original dataset that were not cleaned. A colour scheme was implemented to show distribution of features in the unclean or original dataset that were irrelevant, had more than 40% missing values, were target features, contained redundant data or were relevant features.

Legend:
Orange-Irrelevant features
Blue-Features with more than 40% NaN values
Red-Target features
Dark Magenta-Redundant features
Purple-Relevant features

Number of Variables	Column Name	Description	Feature type(e.g., Numeric, String)
	id	Unique listing id (Primary Key)	Numeric
1	listing_url	Link to the rental property listing on Airbnb	String
2	scrape_id	Identifier for scraper	Numeric
3	last_scraped	Last date listing was scraped	Numeric
4	name	Title of Posting	String
5	description	Description of Posting	String
6	neighborhood_overview	Overview of neighborhood	String
7	picture_url	Link to the main vacation rental listing image on Airbnb	Image
8	host_id	Unique id for each host	Numeric
9	host_url	Link to the host profile on Airbnb	String
10	host_name	Host Name	String
11	host_since	Date an individual became a host	Numeric
12	host_location	Location of Listing	String
13	host_about	Description of host - relationship status, interests and hobbies	String

14	*host_response_time*	Time to respond to customer booking inquiry; ranges from 1hour to few days	Numeric
15	*host_response_rate*	Ranges from 0% to 100% for reply to booking inquiries	Numeric
16	*host_acceptance_rate*	Ranges from 0% to 100% response for acceptance of booking	Numeric
17	host_is_superhost	Is either t(true) or f(false)	Boolean
18	host_thumbnail_url	Host thumbnail URL	Image
19	host_picture_url	Host Picture URL	Image
20	host_neighbourhood	Description of neighborhood listing	String
21	host_listings_count	Current number of host listings	Numeric
22	host_total_listings_count	Total number of listings made by host	Numeric
23	host_verifications	Identifies how the host has completed the identity verification process	String
24	host_has_profile_pic	Host Profile Pic	Image
25	host_identity_verified	Identifies if the host has completed the verification process by indicating true or false	Boolean
26	neighbourhood	Specific location of Toronto area	String
27	neighbourhood_cleansed	Represents one of boroughs in Toronto in which a listing resides	String
28	neighbourhood_group_cleansed	No values (N/A)	N/A
29	latitude	The angular distance of a location or object north or south of the Earth's celestial equator	String
30	longitude	The angular distance of a location or object east or west of the meridian	String
31	property_type	Type of property (e.g., Entire house)	String
32	room_type	Specific type of room (e.g., Entire home/apt)	String
33	accommodates	How many people can stay (e.g., 6)	Numeric
34	bathrooms	No values, NA	N/A
35	bathrooms_text	Number of bathrooms in property	Numeric
36	bedrooms	Number of bedrooms in property	Numeric

37	beds	Number of beds in property	Numeric
38	amenities	List of amenities such as shampoo available in property	String
39	*price*	Price of stay per night	Numeric
40	minimum_nights	Minimum nights can be booked by same individual	Numeric
41	maximum_nights	Maximum nights can be booked by same individual	Numeric
42	minimum_minimum_nights	Same values as Minimum nights	Numeric
43	maximum_minimum_nights	Same values as Maximum_nights	Numeric
44	minimum_maximum_nights	Same values as Maximum_nights	Numeric
45	maximum_maximum_nights	Same values as Maximum_nights	Numeric
46	minimum_nights_avg_ntm	Average minimum nights can be booked by same individual	Numeric
47	maximum_nights_avg_ntm	Average maximum nights can be booked by same individual	Numeric
48	calendar_updated	No values (N/A)	N/A
49	has_availability	True or False if listing is available	Boolean
50	availability_30	Availability of Property in 30 days	Numeric
51	availability_60	Availability of Property in 60 days	Numeric
52	availability_90	Availability of Property in 90 days	Numeric
53	availability_365	Availability of Property in 365 days	Numeric
54	calendar_last_scraped	Date last scraped	Numeric
55	number_of_reviews	Total number of reviews that a listing has received from customers	Numeric
56	number_of_reviews_ltm	The number of reviews that a listing has received last twelve month	Numeric
57	number_of_reviews_l30d	The number of reviews that a listing has received per 130 days	Numeric
58	first_review	Date of first review by customer	Numeric
59	last_review	Date of last review by customer	Numeric

60	***review_scores_rating***	***Customer-provided score rating (0% to 100%); A customer-provided review score attributed to a listing based on overall experience and satisfaction***	Numeric
61	review_scores_accuracy	Accuracy of review scores (0 to 10)	Numeric
62	review_scores_cleanliness	Cleanliness score (0 to 10)	Numeric
63	review_scores_checkin	Over-all check in score (0 to 10)	Numeric
64	review_scores_communication	Score on communication with host (0 to 10)	Numeric
65	***review_scores_location***	***Score on location based on factors such as nearby transportation, noise level (0 to 10)***	Numeric
66	review_scores_value	Over- all value/quality/experience (0 to 10)	Numeric
67	license	No values (N/A)	N/A
68	instant_bookable	True or False if customer can instantly book	BOolean
69	calculated_host_listings_count	Calculated number of listings by host	Numeric
70	calculated_host_listings_count_entire_homes	Number of host listings which are entire homes	Numeric
71	calculated_host_listings_count_private_rooms	Number of host listings which are private rooms	Numeric
72	calculated_host_listings_count_shared_rooms	Number of host listings which are shared homes	Numeric
73	reviews_per_month	Number of Customer reviews of the host accommodation or accommodations per month	Numeric

Figure 2. Correlations between numerical features and target variable being price

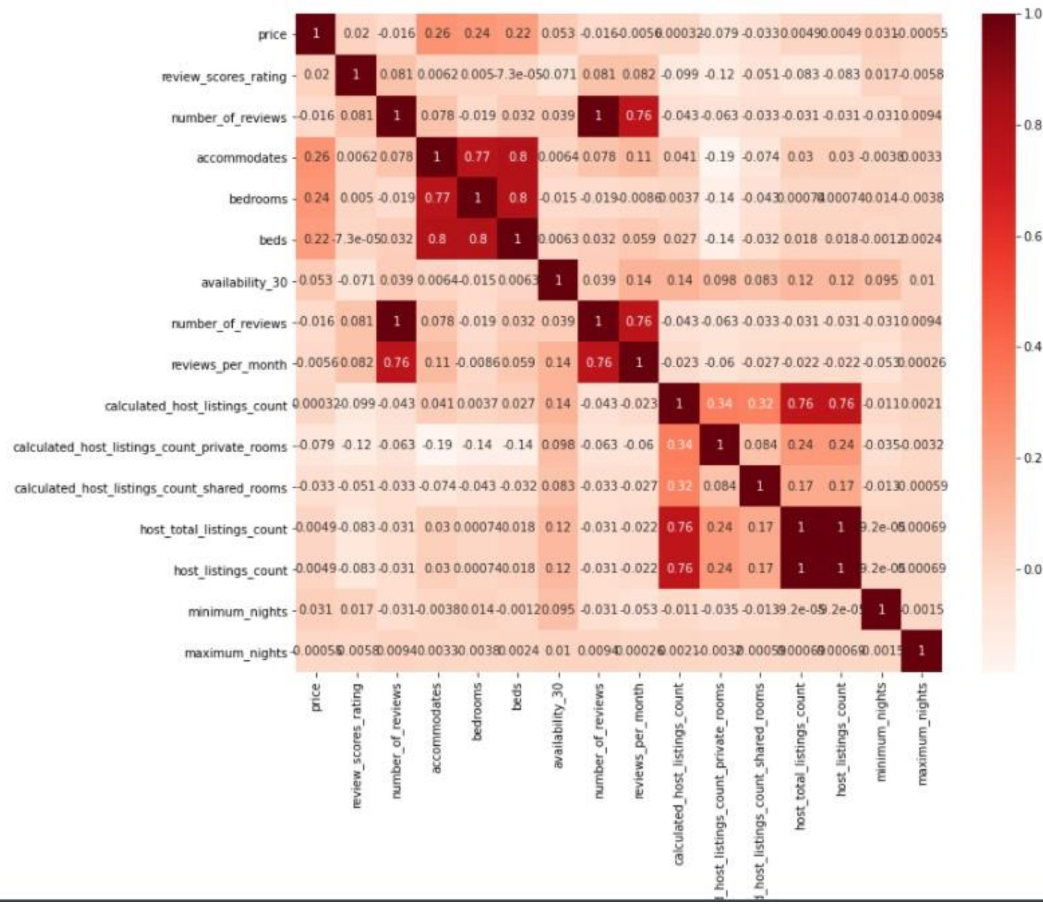


Table 2. Number of non-null values by each feature displayed in dataset feature profile displayed in output from python's dataframe.info() function. There were 74 features, 41 numerical(21 integer, 20 float) and 34 categorical (object) features. As can be seen missing values were present in most features. In addition, there were diversity of data types characterizing the features. For example, 'neighbourhood' is a categorical feature (i.e., Object), 'price' is listed as object due to being in currency format, review ratings score which is a numerical feature is of float data type and 'accomodates' which is a numerical feature is of integer data type.

<class 'pandas.core.frame.DataFrame'>

Int64Index: 19343 entries, 0 to 19342

Data columns (total 75 columns):

#	Column	Non-Null Count	Dtype
0	id	19343 non-null	int64
1	listing_url	19343 non-null	object
2	scrape_id	19343 non-null	int64
3	last_scraped	19343 non-null	object
4	name	19342 non-null	object
5	description	18623 non-null	object
6	neighborhood_overview	12364 non-null	object
7	picture_url	19343 non-null	object
8	host_id	19343 non-null	int64
9	host_url	19343 non-null	object
10	host_name	19339 non-null	object
11	host_since	19339 non-null	object
12	host_location	19329 non-null	object

13	**host_about**	**10958 non-null**	**object**
14	**host_response_time**	**11814 non-null**	**object**
15	**host_response_rate**	**11814 non-null**	**object**
16	**host_acceptance_rate**	**13672 non-null**	**object**
17	**host_is_superhost**	**19339 non-null**	**object**
18	**host_thumbnail_url**	**19339 non-null**	**object**
19	**host_picture_url**	**19339 non-null**	**object**
20	**host_neighbourhood**	**15552 non-null**	**object**
21	**host_listings_count**	**19339 non-null**	**float64**
22	**host_total_listings_count**	**19339 non-null**	**float64**
23	**host_verifications**	**19343 non-null**	**object**
24	**host_has_profile_pic**	**19339 non-null**	**object**
25	**host_identity_verified**	**19339 non-null**	**object**
26	**neighbourhood**	**12364 non-null**	**object**
27	**neighbourhood_cleansed**	**19343 non-null**	**object**
28	**neighbourhood_group_cleansed**	**0 non-null**	**float64**
29	**latitude**	**19343 non-null**	**float64**
30	**longitude**	**19343 non-null**	**float64**
31	**property_type**	**19343 non-null**	**object**
32	**room_type**	**19343 non-null**	**object**
33	**accommodates**	**19343 non-null**	**int64**

34	**bathrooms**	**0 non-null**	**float64**
35	**bathrooms_text**	**19330 non-null**	**object**
36	**bedrooms**	**17914 non-null**	**float64**
37	**beds**	**19147 non-null**	**float64**
38	**amenities**	**19343 non-null**	**object**
39	**price**	**19343 non-null**	**object**
40	**minimum_nights**	**19343 non-null**	**int64**
41	**maximum_nights**	**19343 non-null**	**int64**
42	**minimum_minimum_nights**	**19343 non-null**	**int64**
43	**maximum_minimum_nights**	**19343 non-null**	**int64**
44	**minimum_maximum_nights**	**19343 non-null**	**int64**
45	**maximum_maximum_nights**	**19343 non-null**	**int64**
46	**minimum_nights_avg_ntm**	**19343 non-null**	**float64**
47	**maximum_nights_avg_ntm**	**19343 non-null**	**float64**
48	**calendar_updated**	**0 non-null**	**float64**
49	**has_availability**	**19343 non-null**	**object**
50	**availability_30**	**19343 non-null**	**int64**
51	**availability_60**	**19343 non-null**	**int64**
52	**availability_90**	**19343 non-null**	**int64**
53	**availability_365**	**19343 non-null**	**int64**
54	**calendar_last_scraped**	**19343 non-null**	**object**

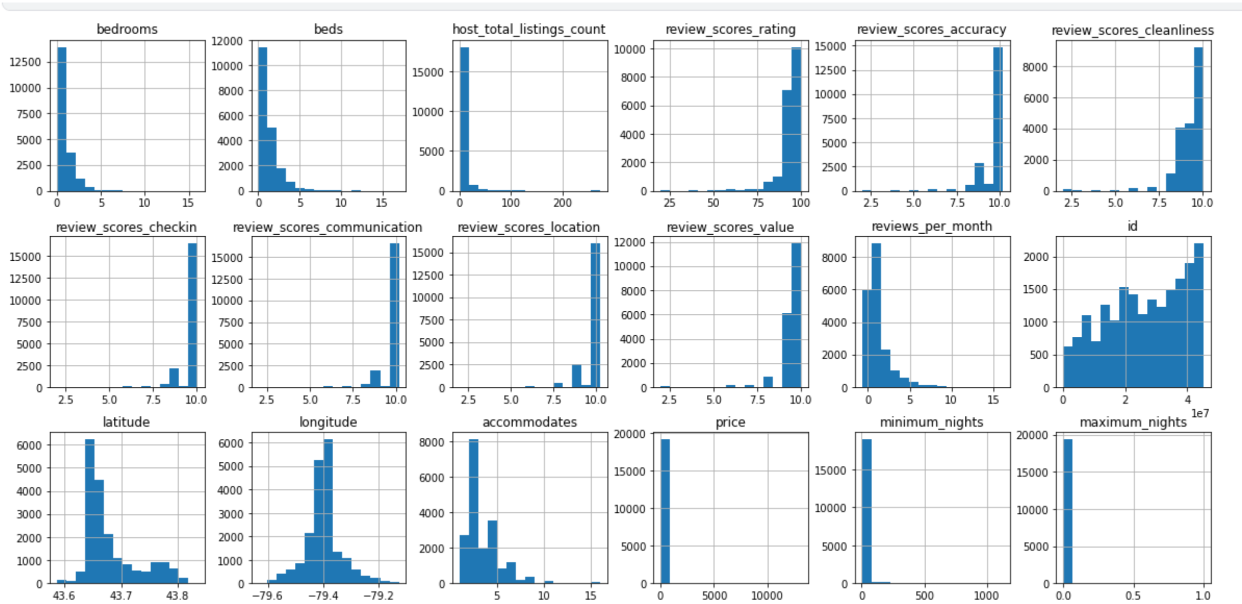
55	number_of_reviews	19343 non-null	int64
56	number_of_reviews_ltm	19343 non-null	int64
57	number_of_reviews_l30d	19343 non-null	int64
58	first_review	15278 non-null	object
59	last_review	15278 non-null	object
60	review_scores_rating	15010 non-null	float64
61	review_scores_accuracy	14976 non-null	float64
62	review_scores_cleanliness	14976 non-null	float64
63	review_scores_checkin	14974 non-null	float64
64	review_scores_communication	14978 non-null	float64
65	review_scores_location	14971 non-null	float64
66	review_scores_value	14972 non-null	float64
67	license	0 non-null	float64
68	instant_bookable	19343 non-null	object
69	calculated_host_listings_count	19343 non-null	int64
70	calculated_host_listings_count_entire_homes	19343 non-null	int64
71	calculated_host_listings_count_private_rooms	19343 non-null	int64
72	calculated_host_listings_count_shared_rooms	19343 non-null	int64
73	reviews_per_month	15278 non-null	float64
74	former_city	19343 non-null	object

dtypes: float64(20), int64(21), object(34)

Table 3. Feature dataset profile after imputation of missing values for numerical features

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 19343 entries, 0 to 19342
Data columns (total 37 columns):
 #   Column                                        Non-Null Count  Dtype
---  ------                                        --------------  -----
 0   host_total_listings_count                     19339 non-null  float64
 1   neighbourhood_cleansed                        19343 non-null  object
 2   latitude                                      19343 non-null  float64
 3   longitude                                     19343 non-null  float64
 4   accommodates                                  19343 non-null  int64
 5   bedrooms                                      17914 non-null  float64
 6   beds                                          19147 non-null  float64
 7   price                                         19343 non-null  float64
 8   minimum_nights                                19343 non-null  int64
 9   maximum_nights                                19343 non-null  int64
 10  availability_30                               19343 non-null  int64
 11  availability_60                               19343 non-null  int64
 12  availability_90                               19343 non-null  int64
 13  availability_365                              19343 non-null  int64
 14  number_of_reviews                             19343 non-null  int64
 15  number_of_reviews_ltm                         19343 non-null  int64
 16  number_of_reviews_l30d                        19343 non-null  int64
 17  review_scores_rating                          15010 non-null  float64
 18  review_scores_accuracy                        14976 non-null  float64
 19  review_scores_cleanliness                     14976 non-null  float64
 20  review_scores_checkin                         14974 non-null  float64
 21  review_scores_communication                   14978 non-null  float64
 22  review_scores_location                        14971 non-null  float64
 23  review_scores_value                           14972 non-null  float64
 24  instant_bookable                              19343 non-null  object
 25  calculated_host_listings_count                19343 non-null  int64
 26  calculated_host_listings_count_entire_homes   19343 non-null  int64
 27  calculated_host_listings_count_private_rooms  19343 non-null  int64
 28  calculated_host_listings_count_shared_rooms   19343 non-null  int64
 29  reviews_per_month                             15278 non-null  float64
 30  former_city                                   19343 non-null  object
 31  room_type                                     19343 non-null  object
 32  host_since                                    19343 non-null  object
 33  host_response_time                            19343 non-null  object
 34  host_is_superhost                             19343 non-null  object
 35  host_acceptance_rate                          19343 non-null  object
 36  host_response_rate                            19343 non-null  object
dtypes: float64(14), int64(14), object(9)
memory usage: 5.6+ MB
```

Figure 4. Outliers in numerical features as shown from Python and in Tableau



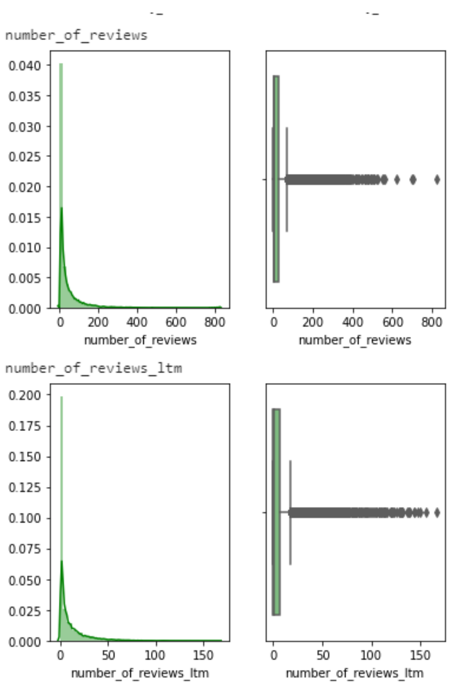


Figure 5. The dataframe named 'df4' contained features such as price with outliers, whereas the data frame named 'df' contains features such as price without outliers.

```
df4[['price', 'review_scores_rating', 'minimum_nights', 'host_total_listings_count']].describe()
```

	price	review_scores_rating	minimum_nights	host_total_listings_count
count	19343.000000	15010.000000	19343.000000	19339.00000
mean	141.278116	94.304930	9.988730	5.52283
std	290.664182	8.899373	37.022953	15.39491
min	12.000000	20.000000	1.000000	0.00000
25%	63.000000	93.000000	1.000000	1.00000
50%	100.000000	97.000000	2.000000	1.00000
75%	150.000000	100.000000	5.000000	4.00000
max	13164.000000	100.000000	1125.000000	272.00000

```
df[['price', 'review_scores_rating', 'minimum_nights', 'host_total_listings_count']].describe()
```

	price	review_scores_rating	minimum_nights	host_total_listings_count
count	19343.000000	19343.000000	19343.000000	19343.000000
mean	117.516891	95.257847	3.834876	2.742685
std	70.959498	4.043789	3.598508	2.770920
min	12.000000	86.500000	1.000000	0.000000
25%	63.000000	94.000000	1.000000	1.000000
50%	100.000000	95.000000	2.000000	1.000000
75%	150.000000	99.000000	5.000000	4.000000
max	280.500000	100.000000	11.000000	8.500000

Figure 6. Display of sample values in new features

```
df[['host_acceptance_rate','labels_host_acceptance_rate','host_response_rate','labels_h
```

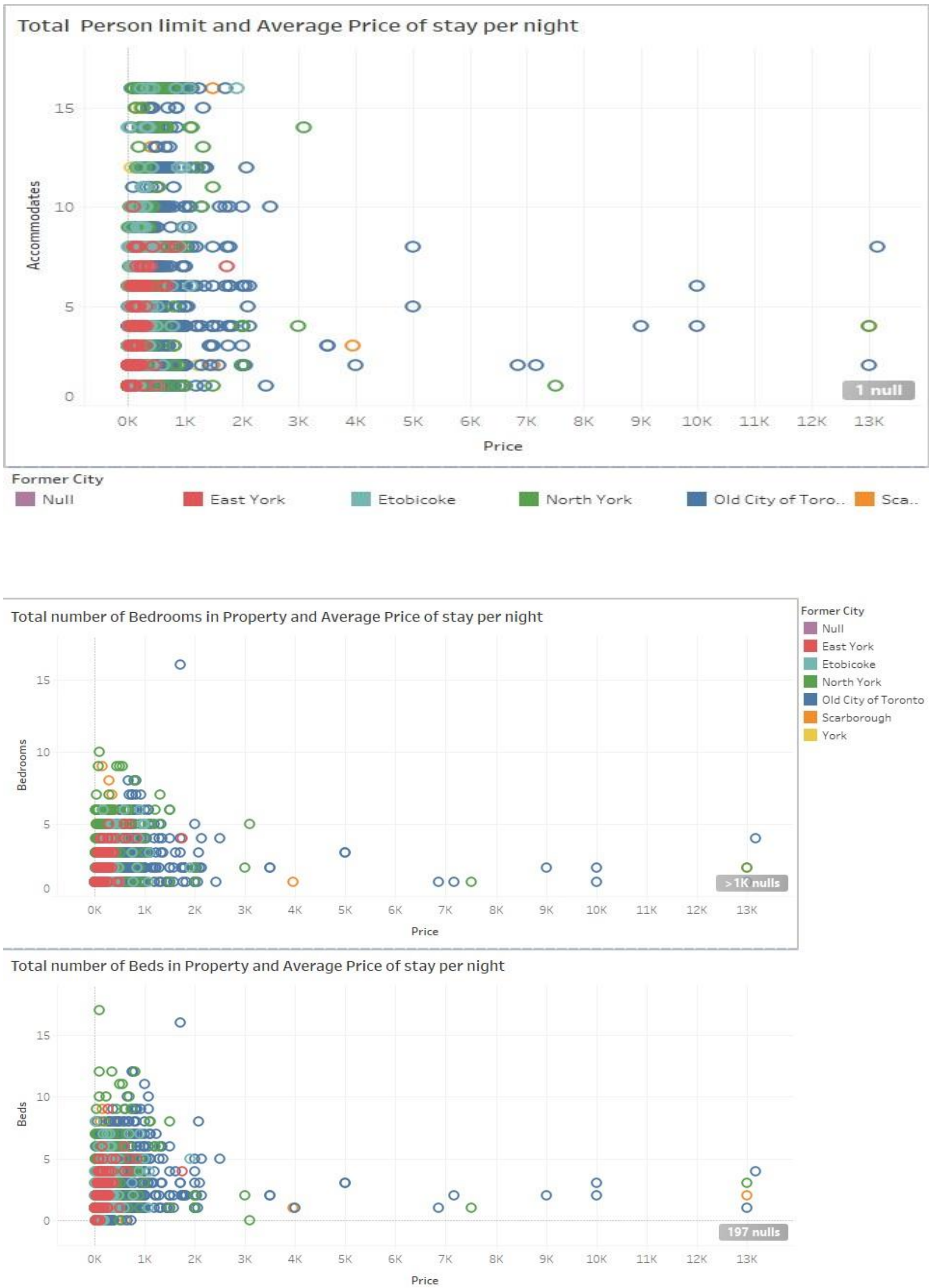
	host_acceptance_rate	labels_host_acceptance_rate	host_response_rate	labels_host_response_rate	host_since_year
0	100	Very Accepting	100	Very high	2008
1	100	Very Accepting	100	Very high	2011
2	100	Very Accepting	100	Very high	2012
3	100	Very Accepting	100	Very high	2012
4	100	Very Accepting	100	Very high	2012
...	...	...	...	...	...
19338	100	Very Accepting	100	Very high	2019
19339	100	Very Accepting	100	Very high	2020
19340	94	Very Accepting	100	Very high	2017
19341	75	Accepting	100	Very high	2019
19342	100	Very Accepting	100	Very high	2020

19343 rows × 5 columns

Table 4. Feature dataset profile of the Cleaned dataset that will be used for deliverables; Part 3 and Part 4

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 19343 entries, 0 to 19342
Data columns (total 41 columns):
 #   Column                                        Non-Null Count  Dtype
---  ------                                        --------------  -----
 0   bedrooms                                      19343 non-null  float64
 1   beds                                          19343 non-null  float64
 2   host_total_listings_count                     19343 non-null  float64
 3   review_scores_rating                          19343 non-null  float64
 4   review_scores_accuracy                        19343 non-null  float64
 5   review_scores_cleanliness                     19343 non-null  float64
 6   review_scores_checkin                         19343 non-null  float64
 7   review_scores_communication                   19343 non-null  float64
 8   review_scores_location                        19343 non-null  float64
 9   review_scores_value                           19343 non-null  float64
 10  reviews_per_month                             19343 non-null  float64
 11  neighbourhood_cleansed                        19343 non-null  object
 12  latitude                                      19343 non-null  float64
 13  longitude                                     19343 non-null  float64
 14  accommodates                                  19343 non-null  int64
 15  price                                         19343 non-null  float64
 16  minimum_nights                                19343 non-null  int64
 17  maximum_nights                                19343 non-null  int64
 18  availability_30                               19343 non-null  int64
 19  availability_60                               19343 non-null  int64
 20  availability_90                               19343 non-null  int64
 21  availability_365                              19343 non-null  int64
 22  number_of_reviews                             19343 non-null  int64
 23  number_of_reviews_ltm                         19343 non-null  int64
 24  number_of_reviews_l30d                        19343 non-null  int64
 25  instant_bookable                              19343 non-null  object
 26  calculated_host_listings_count                19343 non-null  int64
 27  calculated_host_listings_count_entire_homes   19343 non-null  int64
 28  calculated_host_listings_count_private_rooms  19343 non-null  int64
 29  calculated_host_listings_count_shared_rooms   19343 non-null  int64
 30  former_city                                   19343 non-null  object
 31  room_type                                     19343 non-null  object
 32  host_since                                    19343 non-null  object
 33  host_response_time                            19343 non-null  object
 34  host_is_superhost                             19343 non-null  object
 35  host_acceptance_rate                          19343 non-null  int64
 36  host_response_rate                            19343 non-null  int64
 37  labels_host_acceptance_rate                   19343 non-null  category
 38  labels_host_response_rate                     19343 non-null  category
 39  host_since_year                               19343 non-null  int64
 40  host_length                                   19343 non-null  int64
dtypes: category(2), float64(14), int64(18), object(7)
memory usage: 5.8+ MB
```

Figure 7.



Total Person limit and Average Price of stay per night
Accommodates
15
10
5
0
1 null
0K
1K
2K
3K
4K
5K
6K
7K
8K
9K
10K
11K
12K
13K
Price
Former City
Null
East York
Etobicoke
North York
Old City of Toro..
Sca..
Total number of Bedrooms in Property and Average Price of stay per night
Bedrooms
>1K nulls
Price
Former City
Null
East York
Etobicoke
North York
Old City of Toronto
Scarborough
York
Total number of Beds in Property and Average Price of stay per night
Beds
197 nulls
Price

Figure 8. Room type and Average price per day of listing by District in Toronto

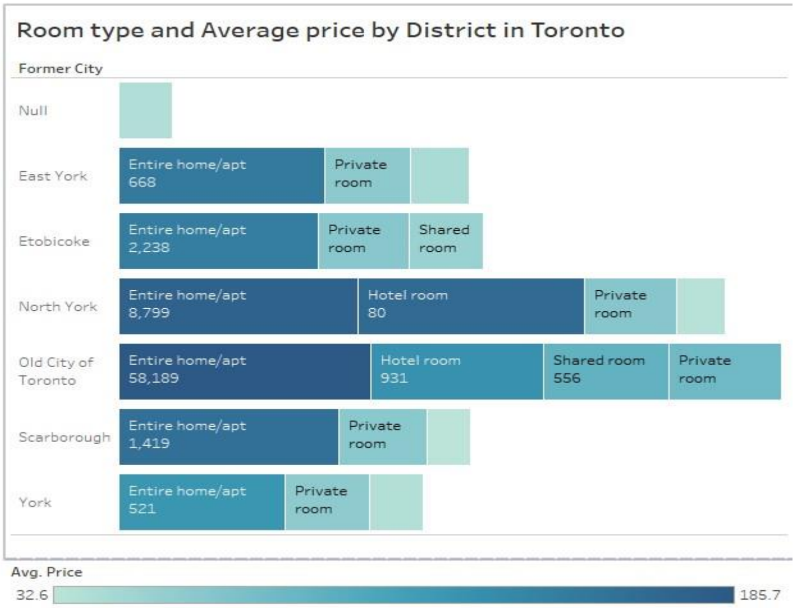


Figure 9. Tableau Dashboard with the uncleansed data being the data source

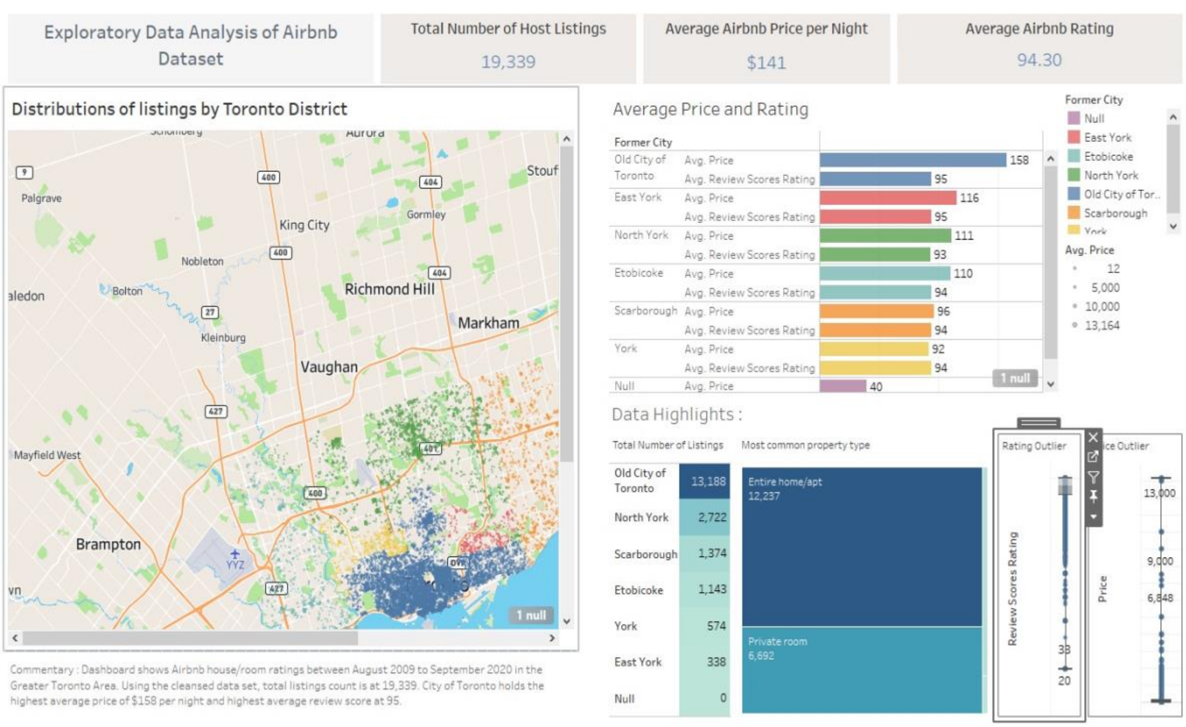


Figure 10. Average price and Host Acceptance rate by District in Toronto