Northeastern University
College of Professional Studies

Module 6 Final Project: Report

ALY6110: Data Management and Big Data

Name
Kavish Shah
Tanusha Erpula

1. Dimension of the Dataset

We are using One-dimensional datasets to give insights into Airbnb Listings in Melbourne, Victoria, Australia. The dimension of the dataset is 18,237*75

Listings include some of the names of the columns that are id, name, host_id, host_name, neighbor's group, neighborhood, latitude, longitude, room_type, price, minimum_nights, number of reviews, last_reviews, reviews_per_month, calculated_host_lisitngs, availability_365, number_of_reviews_Itm, and license.

2. Introduction

Data types

Numeric Values: - 44

Categorical Values: -31

• **Rationale of the Dataset:** To get a good analysis of the business side, we have taken a wanted successful company Airbnb (a rental system). The dataset of Airbnb contains a rich amount of information that can help recognize great deterministic features. We have considered the most popular city in Australia which have 74. interrelated columns that can provide meaningful

insights. We have all the main elements of Airbnb's business process from the host to customer reviews. Since this dataset is huge, we might add or reform our goals initially mentioned.

Tools Used: - Python Language, Tableau.

Data Cleaning

Checking for Null values:

```
airbnb_df.isna().sum()
```

```
id                                              0
listing_url                                     0
scrape_id                                       0
last_scraped                                    0
source                                          0
                                               ...
calculated_host_listings_count                  0
calculated_host_listings_count_entire_homes     0
calculated_host_listings_count_private_rooms    0
calculated_host_listings_count_shared_rooms     0
reviews_per_month                            3723
Length: 75, dtype: int64
```

The above command helps to check the null values in the dataset.

Checking for N/A values which are not zero:

```
df = pd.DataFrame(airbnb_df.isna().sum().reset_index())
```

```
df.loc[df[0] != 0]
```

Out[119]:

Index	0
5	name	2
6	description	426
7	neighborhood_overview	7160
11	host_name	2
12	host_since	2
13	host_location	4300
14	host_about	7826
15	host_response_time	7023
16	host_response_rate	7023
17	host_acceptance_rate	6240
18	host_is_superhost	11
19	host_thumbnail_url	2
20	host_picture_url	2
21	host_neighbourhood	9751
22	host_listings_count	2
23	host_total_listings_count	2
25	host_has_profile_pic	2
26	host_identity_verified	2
27	neighbourhood	7159
29	neighbourhood_group_cleansed	18236
35	bathrooms	18236
36	bathrooms_text	17
37	bedrooms	749
38	beds	224
45	calendar_updated	18236
59	first_review	3723
60	last_review	3723
61	review_scores_rating	3723
62	review_scores_accuracy	4005
63	review_scores_cleanliness	4002
64	review_scores_checkin	4008
65	review_scores_communication	4003
66	review_scores_location	4008
67	review_scores_value	4008
68	license	18236
74	reviews_per_month	3723

,

This table describes the values with N/A values without zeros.

Dropping unnecessary values:

```
airbnb_clean = airbnb_df.drop(['id','name','description','host_acceptance_rate','host_response_rate','neighborhood_overview',
```

```
pd.DataFrame(airbnb_clean.isna().sum().reset_index())
```

Out[121]:

	Index	0
0	host_location	4300
1	host_neighbourhood	9751
2	host_total_listings_count	2
3	neighbourhood	7159
4	neighbourhood_cleansed	0
5	latitude	0
6	longitude	0
7	property_type	0
8	room_type	0
9	accommodates	0
10	bathrooms_text	17
11	bedrooms	749
12	beds	224
13	price	0
14	minimum_nights	0
15	maximum_nights	0
16	minimum_minimum_nights	0
17	maximum_minimum_nights	0
18	minimum_maximum_nights	0
19	maximum_maximum_nights	0
20	minimum_nights_avg_ntm	0
21	maximum_nights_avg_ntm	0
22	has_availability	0
23	availability_30	0
24	availability_60	0
25	availability_90	0
26	availability_365	0
27	number_of_reviews	0
28	number_of_reviews_ltm	0
29	number_of_reviews_l30d	0
30	review_scores_rating	3723
31	review_scores_accuracy	4005
32	review_scores_cleanliness	4002
33	review_scores_checkin	4008
34	review_scores_communication	4003
35	review_scores_location	4008
36	review_scores_value	4008
37	instant_bookable	0
38	calculated_host_listings_count	0
39	calculated_host_listings_count_entire_homes	0
40	calculated_host_listings_count_private_rooms	0
41	calculated_host_listings_count_shared_rooms	0
42	reviews_per_month	3723

The above table shows the columns with unnecessary values.

```
airbnb_clean.columns
```

```
Out[122]: Index(['host_location', 'host_neighbourhood', 'host_total_listings_count',
                 'neighbourhood', 'neighbourhood_cleansed', 'latitude', 'longitude',
                 'property_type', 'room_type', 'accommodates', 'bathrooms_text',
                 'bedrooms', 'beds', 'price', 'minimum_nights', 'maximum_nights',
                 'minimum_minimum_nights', 'maximum_minimum_nights',
                 'minimum_maximum_nights', 'maximum_maximum_nights',
                 'minimum_nights_avg_ntm', 'maximum_nights_avg_ntm', 'has_availability',
                 'availability_30', 'availability_60', 'availability_90',
                 'availability_365', 'number_of_reviews', 'number_of_reviews_ltm',
                 'number_of_reviews_l30d', 'review_scores_rating',
                 'review_scores_accuracy', 'review_scores_cleanliness',
                 'review_scores_checkin', 'review_scores_communication',
                 'review_scores_location', 'review_scores_value', 'instant_bookable',
                 'calculated_host_listings_count',
                 'calculated_host_listings_count_entire_homes',
                 'calculated_host_listings_count_private_rooms',
                 'calculated_host_listings_count_shared_rooms', 'reviews_per_month'],
                dtype='object')
```

3. Questions to Investigate:

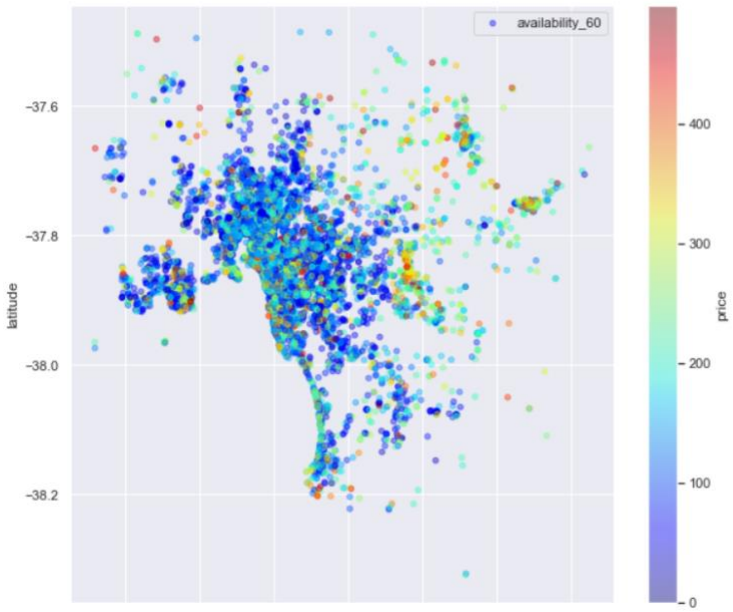
- What is the price distribution of Airbnb in different neighborhoods in Australia?
- What is the total number of rooms distributed in the neighborhood of Australia?
- What is the relationship between neighborhood & Ratings based on price & room type?
- What is the relationship between availability with the price of the property in the neighborhood of Australia?
- What is the total number of Airbnb listed according to the neighborhood of Australia?
- What is the Correlation plot to understand the relationship between each variable?

Exploratory Data Analysis

Price distribution for avability_60 concerning the geographical locations

```
sub_6=airbnb_clean[airbnb_clean.price < 500]
viz_4=sub_6.plot(kind='scatter', x='longitude', y='latitude', label='availability_60', c='price',
cmap=plt.get_cmap('jet'), colorbar=True, alpha=0.4, figsize=(10,9))
viz_4.legend()
```

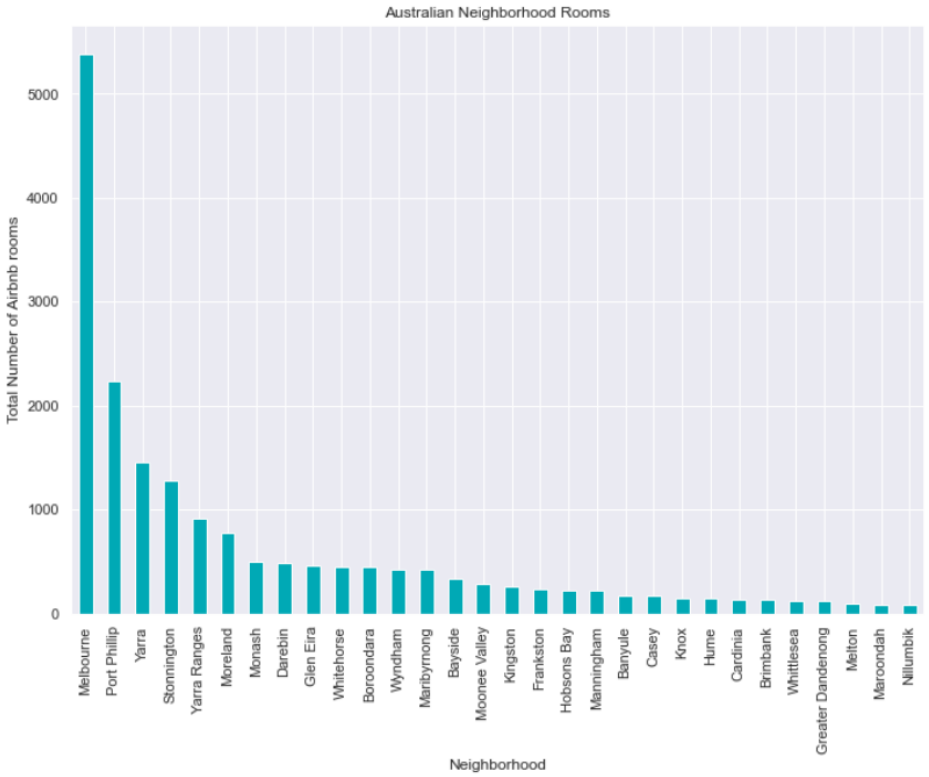
```
: <matplotlib.legend.Legend at 0x7fa624c73c10>
```



The above scatterplot depicts the price distribution of the properties according to the geographical locations in Australia.

Total number of rooms distributed over the Australian neighborhood

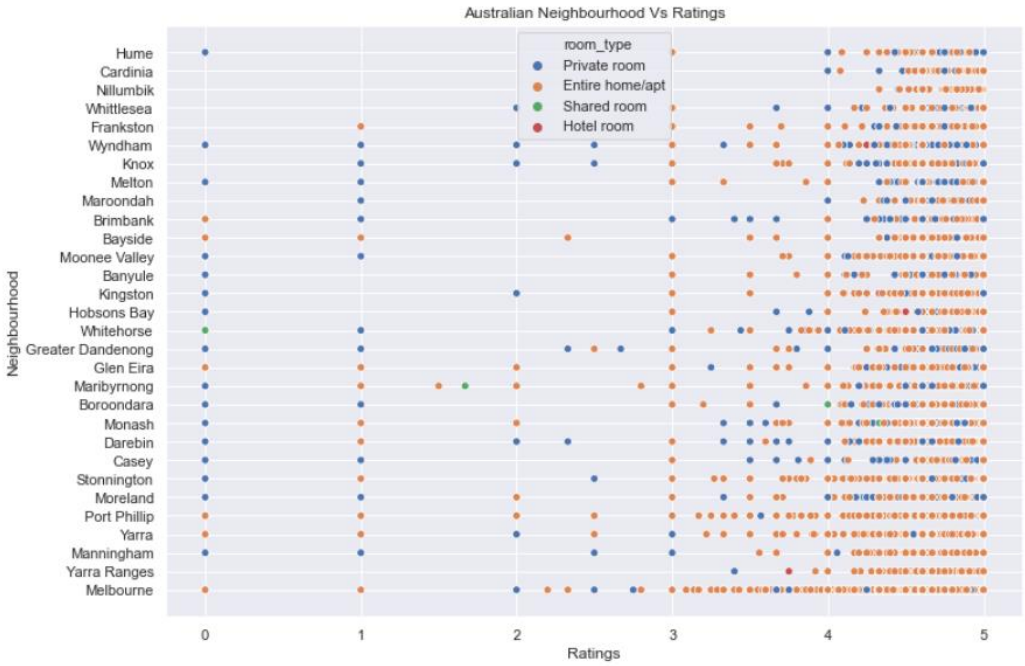
```
A=airbnb_clean.neighbourhood_cleansed.value_counts()
A.plot(kind='bar',color = "#00a9b5")
plt.title('Australian Neighborhood Rooms')
plt.ylabel('Total Number of Airbnb rooms')
plt.xlabel('Neighborhood');
```



The above bar graph explains the number of room distributions available over the Australian neighborhood.

Relationship between neighborhood & Ratings based on price & room type

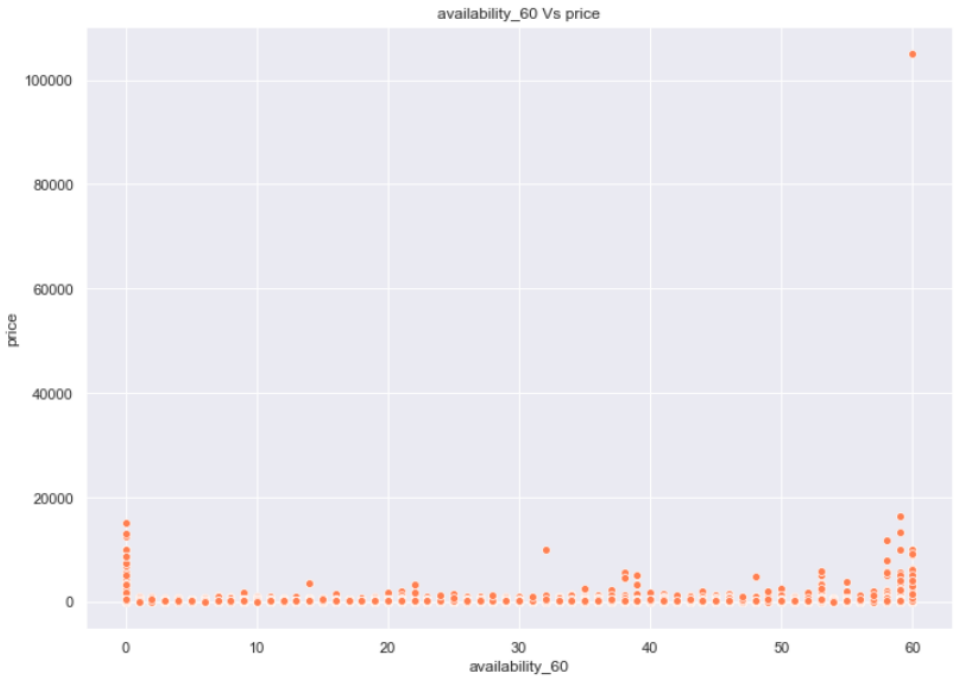
```
#Scatterplot showing the ratings of cambridge in different neighborhoods of different rooms.
#Australian Cities  Neighbourhood vs Rating
airbnb_clean[['neighbourhood_cleansed', 'review_scores_rating']]
sns.scatterplot(data=airbnb_clean, x='review_scores_rating', y='neighbourhood_cleansed',hue="room_type")
plt.ylabel('Neighbourhood')
plt.xlabel('Ratings')
sns.set(rc={'figure.figsize':(11.6,8.27)})
plt.title("Australian Neighbourhood Vs Ratings")
plt.show()
```



The scatterplot displays the relationship between ratings based on room type and ratings with the neighborhood in Australia.

Relationship between availability & price

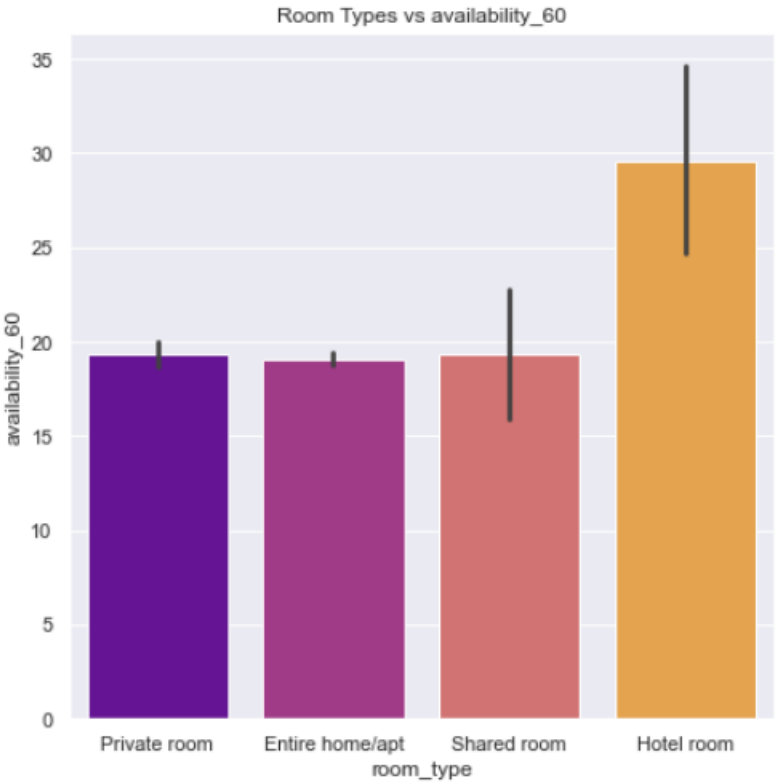
```
airbnb_clean[['availability_60', 'price']]

sns.scatterplot(data=airbnb_clean, x='availability_60', y='price', color ='coral')
plt.ylabel('price')
plt.xlabel('availability_60')
sns.set(rc={'figure.figsize':(11.7,8.27)})
plt.title("availability_60 Vs price")
plt.show()
```



The plot explains the relationship between the price of the property with the availability in the neighborhoods of Australia.

Relationship between room type & availability

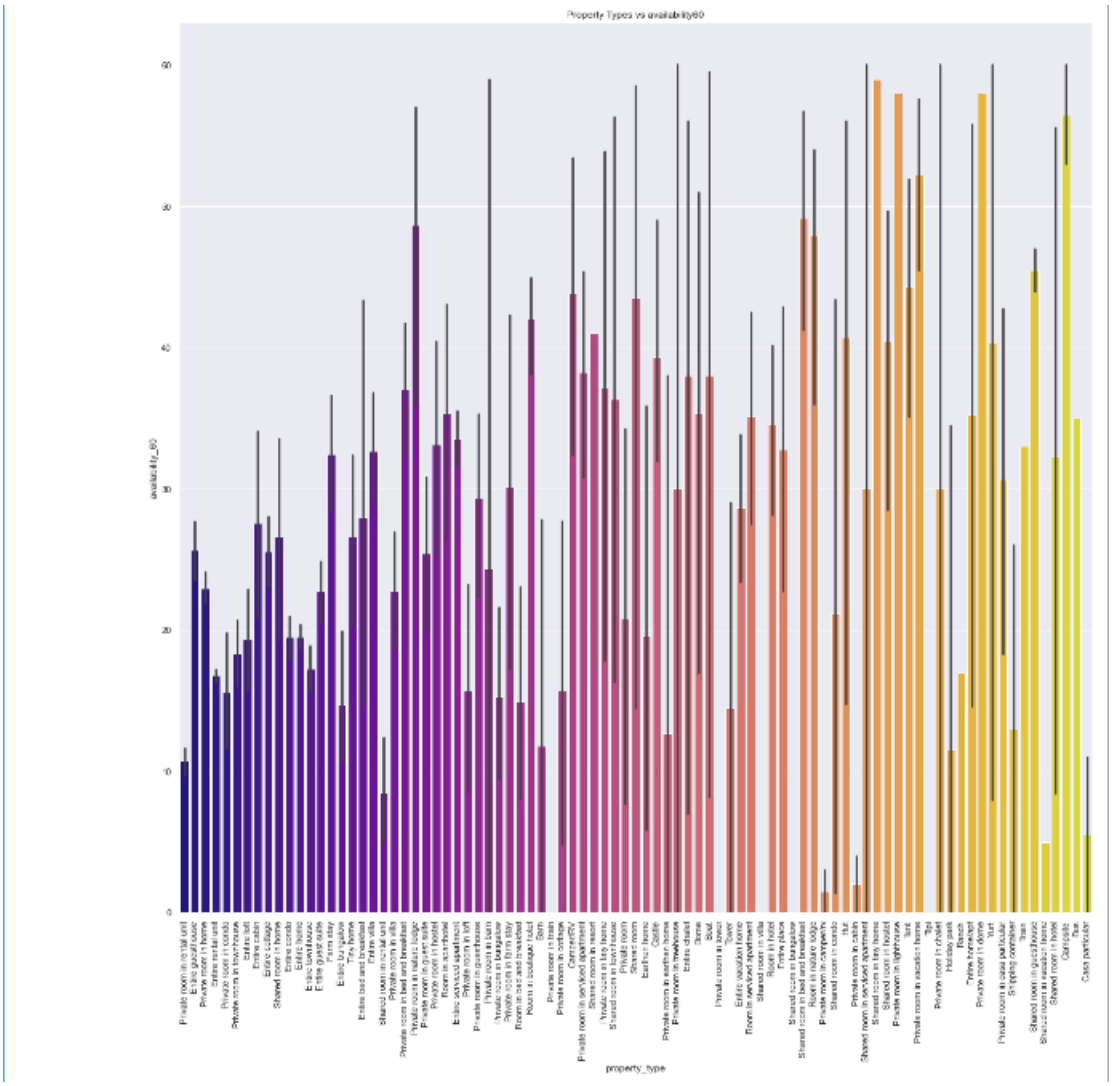
```
plt.figure(figsize=(7,7))
sns.barplot(x='room_type', y='availability_60', palette='plasma', data=airbnb_df)
plt.title('Room Types vs availability_60')
```



This bar graph explains the relationship between room type and availability in the neighborhoods of Australia.

Bar plot denoting the availability based on property type

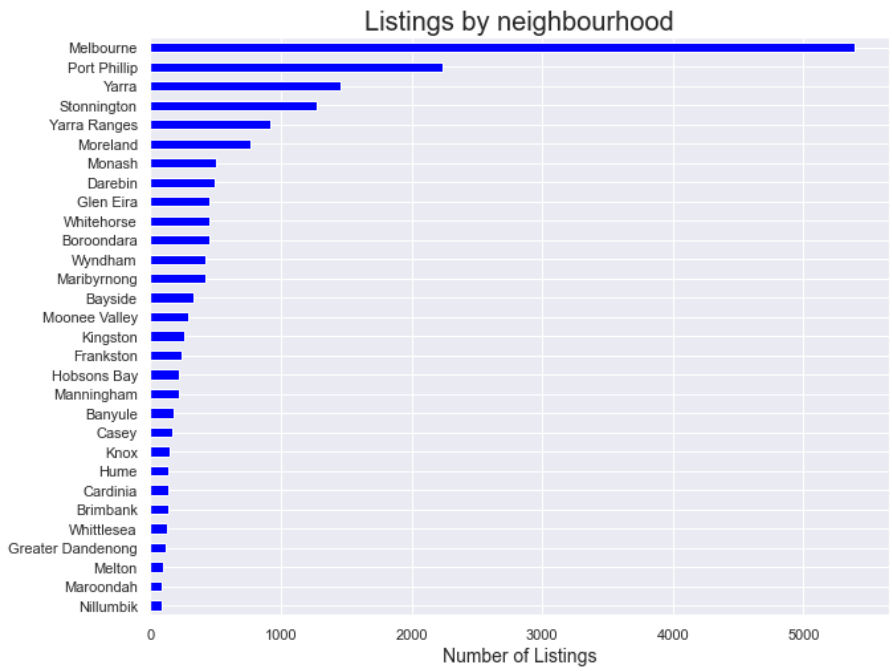
```
plt.figure(figsize=(20,20))
plot=sns.barplot(x='property_type', y='availability_60', palette='plasma', data=airbnb_df)
plt.title('Property Types vs availability60')
for item in plot.get_xticklabels():
    item.set_rotation(90)
```



The bar plot explains the availability of the property type in Australia.

Total number of Airbnb listed according to neighborhood

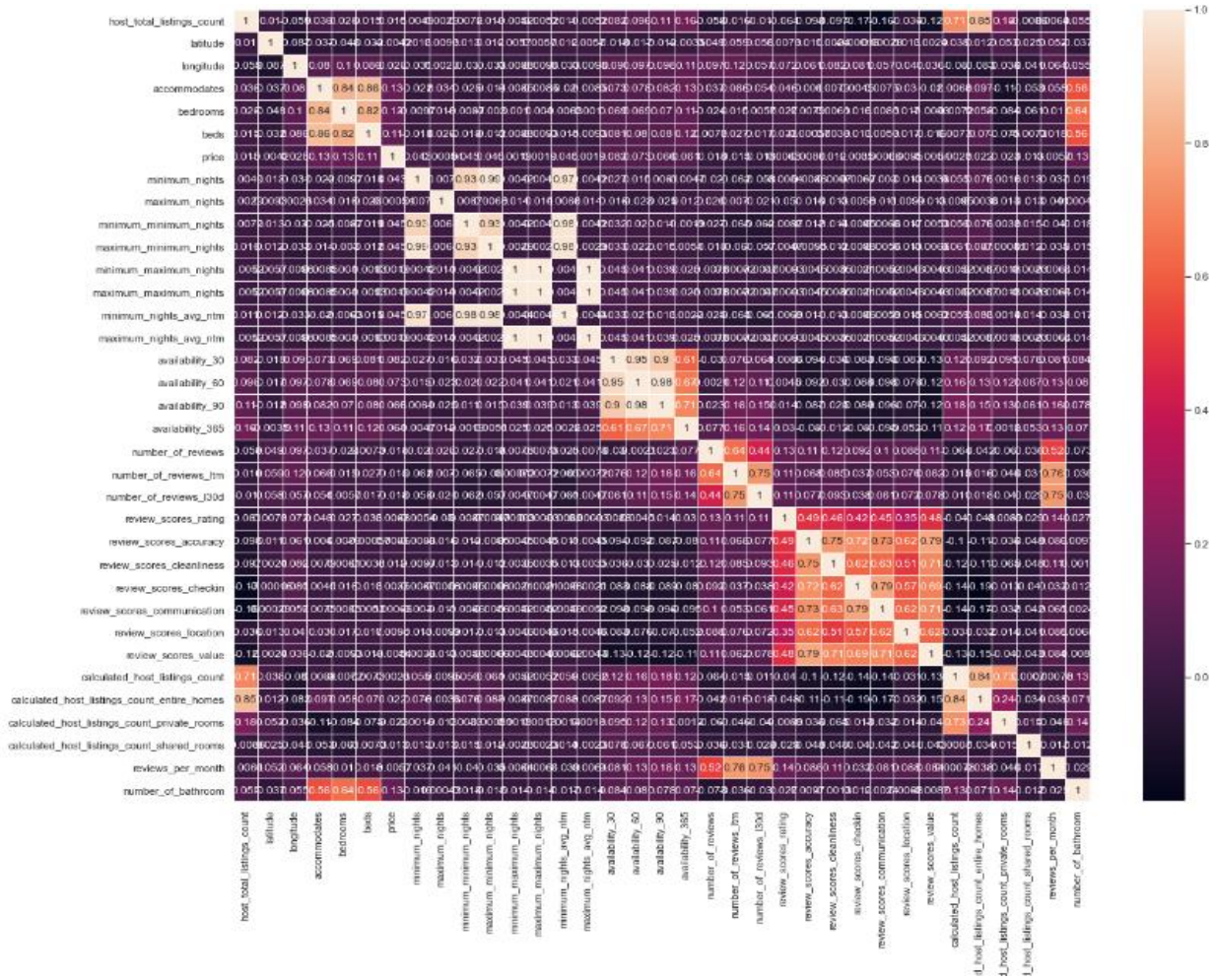
```
feq=airbnb_clean.neighbourhood_cleansed.value_counts().sort_values(ascending=True)
feq.plot.barh(figsize=(10, 8), color="blue")
plt.title("Listings by neighbourhood", fontsize=20)
plt.xlabel('Number of Listings', fontsize=14)
plt.savefig('nl.png', dpi=600, bbox_inches='tight')
plt.show()
```



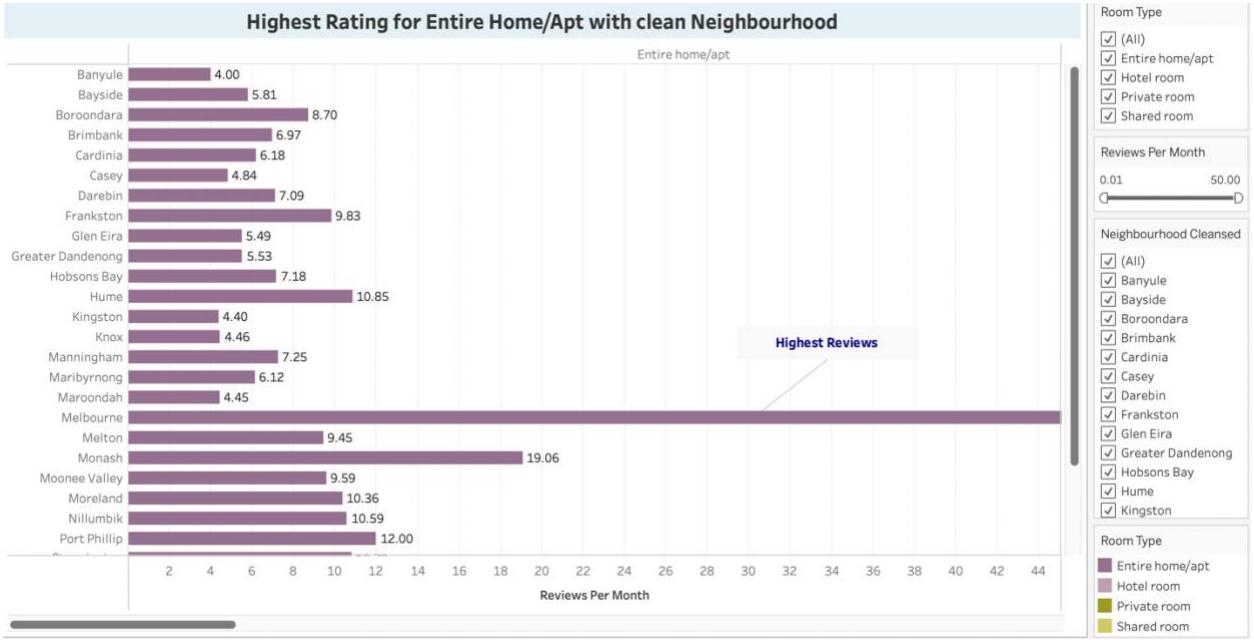
The bar plot displays the number of listings according to the neighborhood.

Correlation plot to understand the relationship between each variable

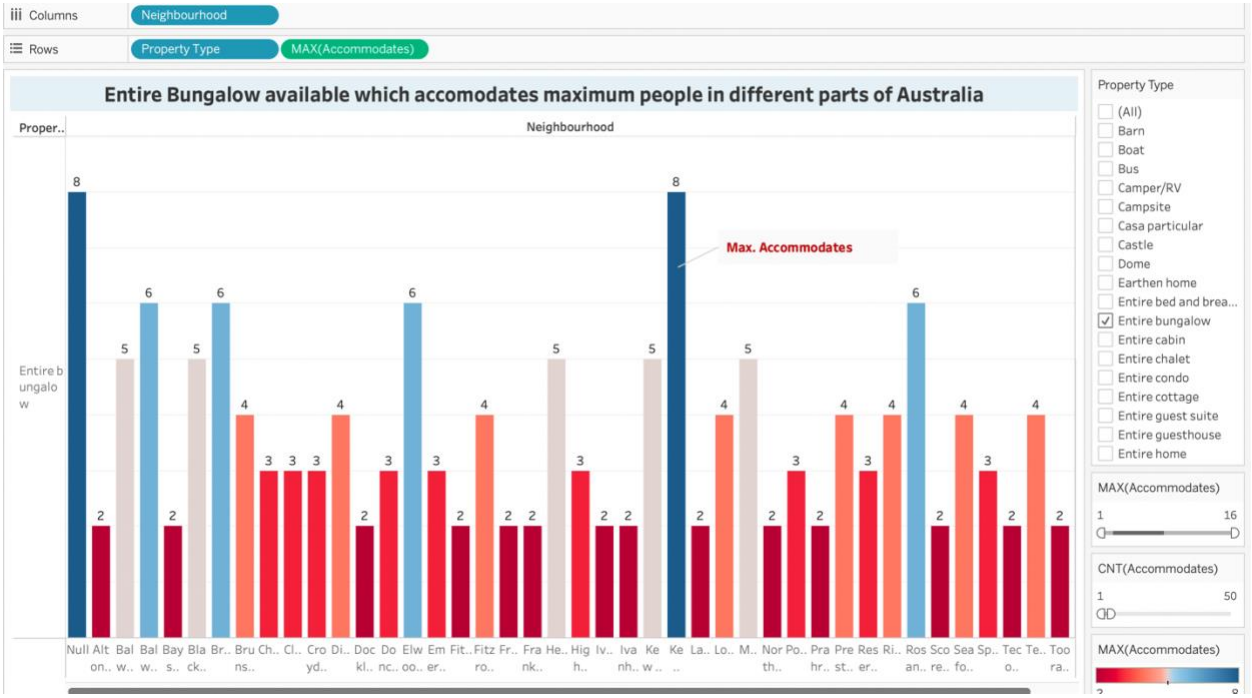
```
corr = airbnb_clean.corr()
mask = np.triu(np.ones_like(corr, dtype=bool))
fig, ax = plt.subplots(figsize=(20,16))
sns.heatmap(corr, annot=True, linewidths=.8, ax=ax)
plt.show()
```


host_total_listings_count
latitude
longitude
accommodates
bedrooms
beds
price
minimum_nights
maximum_nights
minimum_minimum_nights
maximum_minimum_nights
minimum_maximum_nights
maximum_maximum_nights
minimum_nights_avg_ntm
maximum_nights_avg_ntm
availability_30
availability_60
availability_90
availability_365
number_of_reviews
number_of_reviews_ltm
number_of_reviews_l30d
review_scores_rating
review_scores_accuracy
review_scores_cleanliness
review_scores_checkin
review_scores_communication
review_scores_location
review_scores_value
calculated_host_listings_count
calculated_host_listings_count_entire_homes
calculated_host_listings_count_private_rooms
calculated_host_listings_count_shared_rooms
reviews_per_month
number_of_bathroom
1.0
0.8
0.6
0.4
0.2
0.0

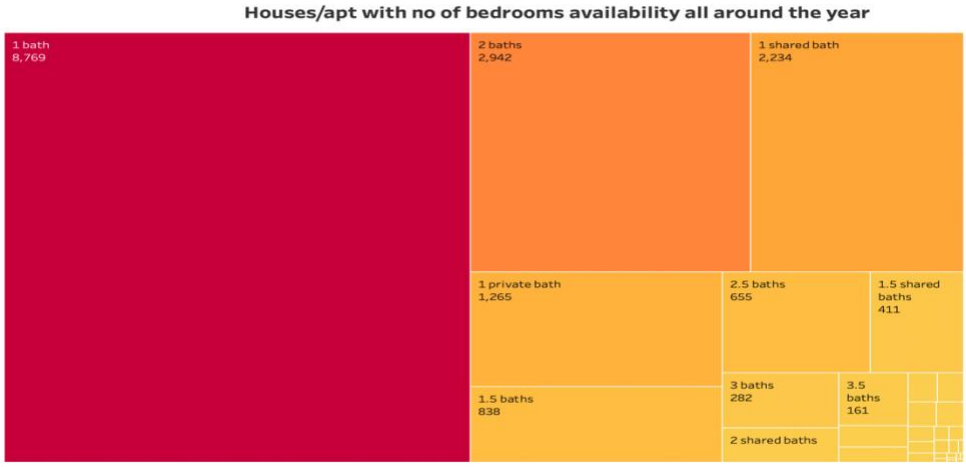
Analysis


Highest Rating for Entire Home/Apt with clean Neighbourhood
Entire home/apt
Banyule 4.00
Bayside 5.81
Boroondara 8.70
Brimbank 6.97
Cardinia 6.18
Casey 4.84
Darebin 7.09
Frankston 9.83
Glen Eira 5.49
Greater Dandenong 5.53
Hobsons Bay 7.18
Hume 10.85
Kingston 4.40
Knox 4.46
Manningham 7.25
Maribyrnong 6.12
Maroondah 4.45
Melbourne
Melton 9.45
Monash 19.06
Moonee Valley 9.59
Moreland 10.36
Nillumbik 10.59
Port Phillip 12.00
Highest Reviews
2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42 44
Reviews Per Month
Room Type
(All)
Entire home/apt
Hotel room
Private room
Shared room
Reviews Per Month
0.01
50.00
Neighbourhood Cleansed
(All)
Banyule
Bayside
Boroondara
Brimbank
Cardinia
Casey
Darebin
Frankston
Glen Eira
Greater Dandenong
Hobsons Bay
Hume
Kingston
Room Type
Entire home/apt
Hotel room
Private room
Shared room

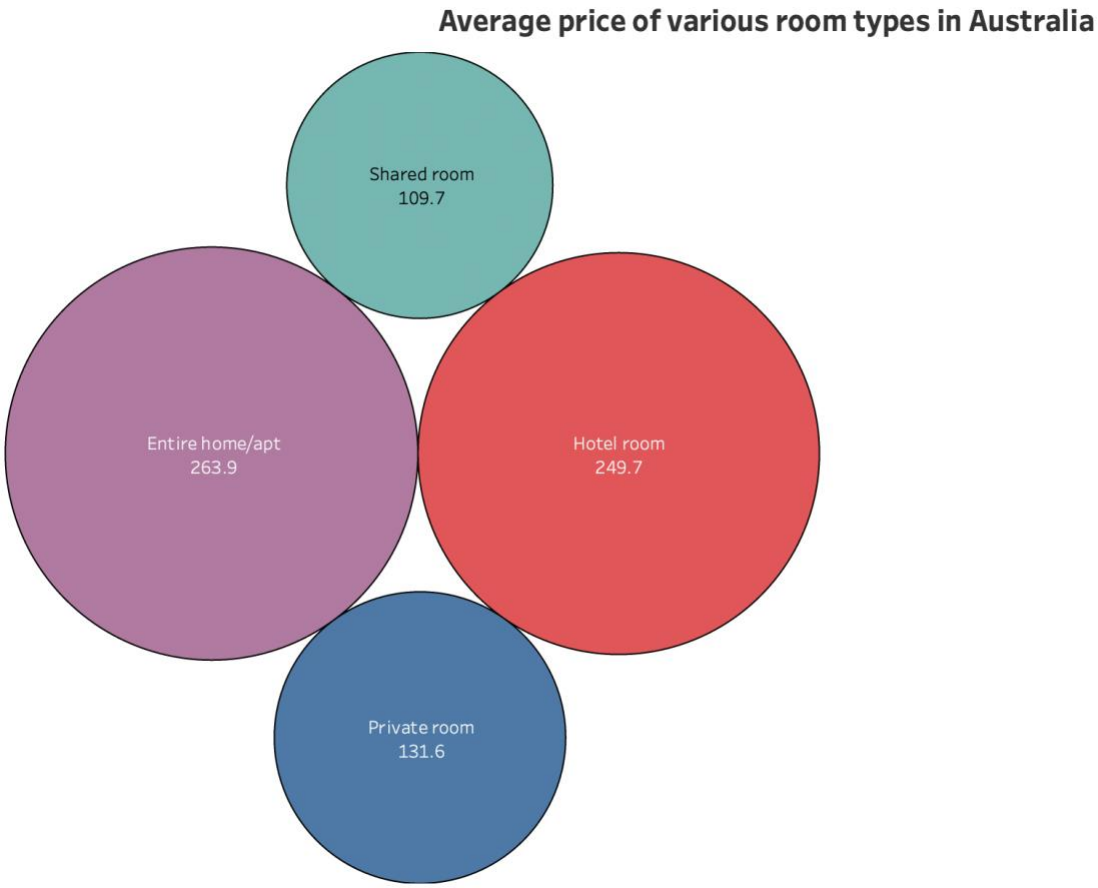
We have created a basic analysis using Tableau for the Airbnb dataset. I have used various measures and records to build multiple worksheets and create a dashboard. My dashboard consists of 5 worksheets and each worksheet displays a certain question. My first worksheet tries to answer the question 'Highest Rating for Entire Home/Apt with clean Neighborhood'. This worksheet shows the highest rating for a certain type of property with a clean vicinity. The highest rating is 49.27 for an entire home/apt property type in Melbourne.



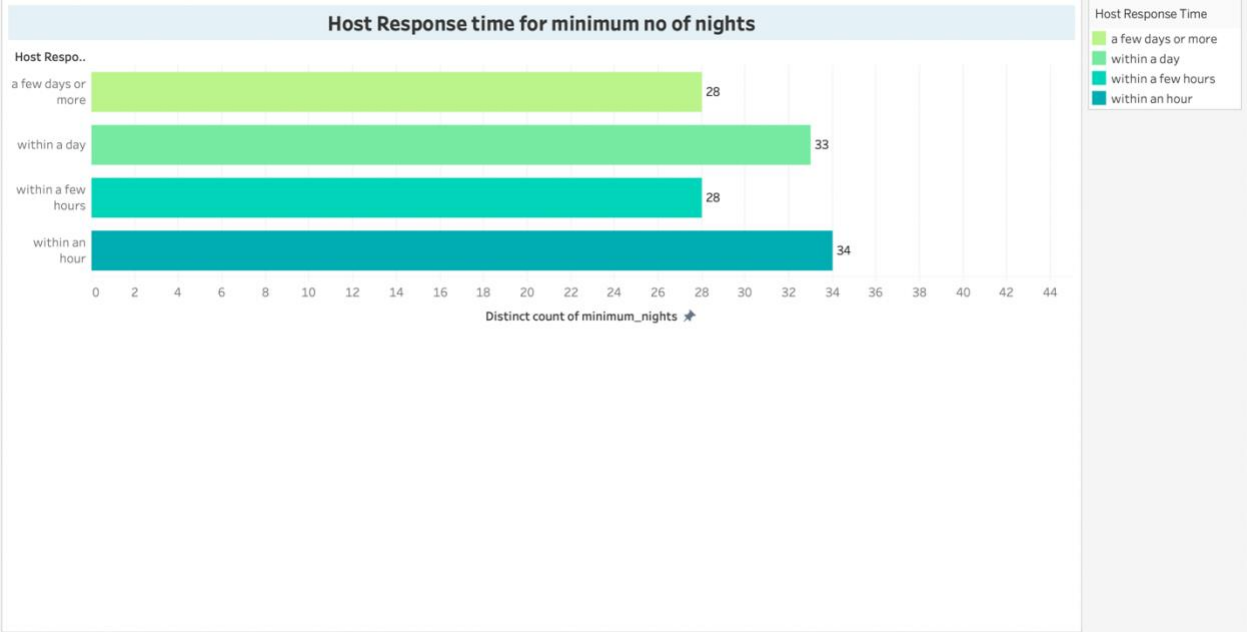
The second worksheet tries to answer the question 'Entire Bungalow available which accommodates a maximum number of people in different parts of Australia '. This worksheet gives the number of maximum occupants in Kew, Victoria, Australia. The maximum occupancy can be 8 in an entire bungalow house type.



The third worksheet answers the question ‘Houses/apt with no of bathrooms availability all around the year’. This worksheet shows that 1 bathroom type is the maximum number available throughout the year.

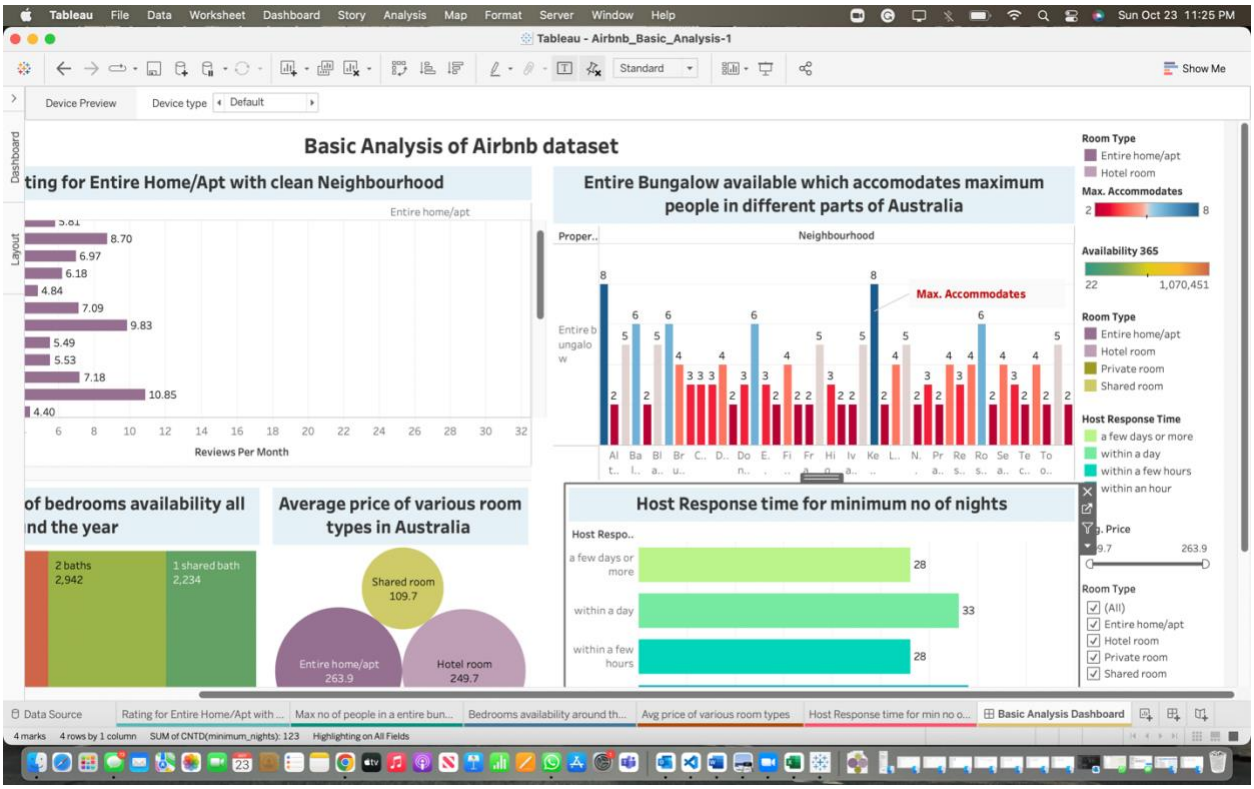


The fourth worksheet answers the question ‘Average price of various room types in Australia’. This sheet explains the prices of the various room types with their average price. This sheet concluded that the entire house/apt property type has the highest average price of $263.9 and the lowest average price is $109.7 for a shared room.



The fifth worksheet answers the question ‘Host Response time for a minimum no of nights. This worksheet works to display how long a host takes to respond depending on the days of occupancy of the rooms. This concludes that generally, hosts are taking few or more days to respond if the occupancy time is less compared to the host’s response time to maximum days of occupancy.

Dashboard



we created a dashboard titled ‘Basic analysis of Airbnb dataset’ with all the worksheets embedded in it. We have added annotations to the worksheets to highlight each worksheet’s conclusions.

Predictive Modeling

For prediction, we have taken the target variable as the availability of Airbnb for 60 days in a different neighborhood in Australia. After cleaning the dataset taken the remaining columns are taken as Features.

Splitting features and labels into X_features & y variables

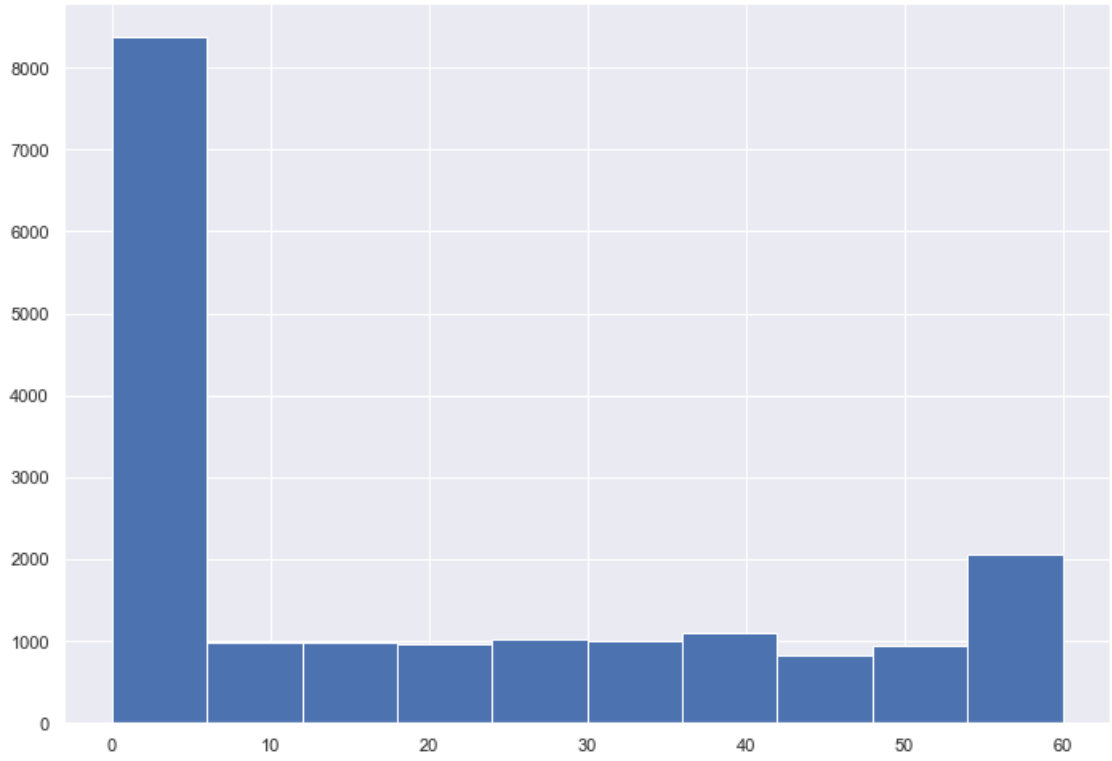
```
X_features = X_features_encode.drop(['availability_60'], axis = 1, inplace = False)
```

```
y = X_features_encode['availability_60']
```

Checking if data is normally distributed

```
y.hist()
```

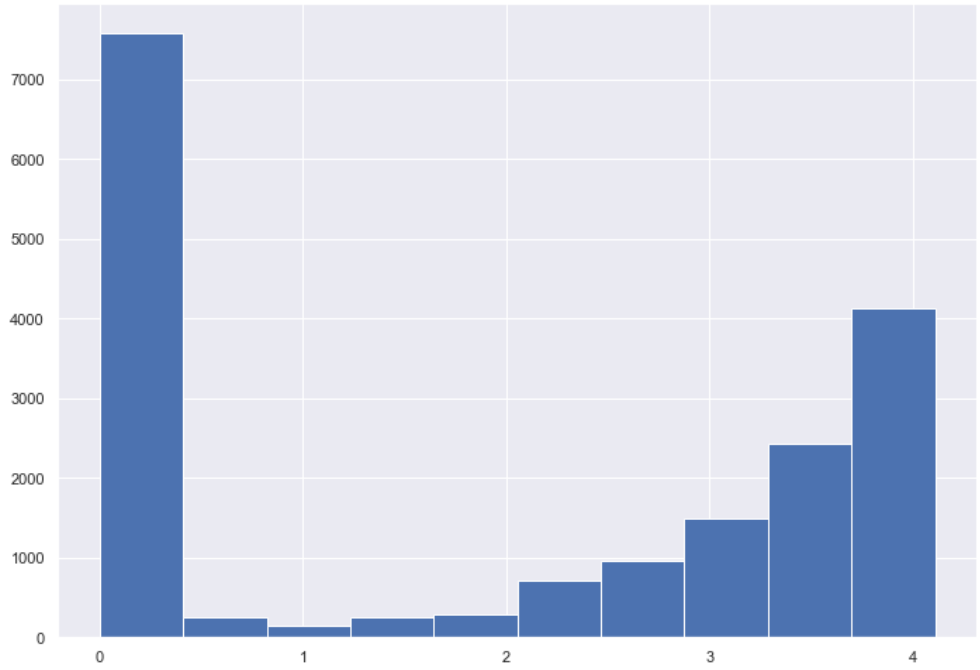
```
<AxesSubplot:>
```



Using a log to normalize the data, the graph is shown below:

```
y_log = np.log1p(y)
```

```
y_log.hist()
```



Decision Tree Regressor

It is possible to visualize choices and all of their potential outcomes, including outcomes, input costs, and utilities, using a decision-making tool named a decision tree.
The classification algorithms group includes the decision-tree algorithm. It works with output variables that are categorized and continuous.

```
from sklearn.tree import DecisionTreeRegressor
DTree=DecisionTreeRegressor(min_samples_leaf=.0001)
DTree.fit(X_train,y_train)

DTree_score = DTree.score(X_test, y_test)
print(DTree_score)
```

```
0.9651106817473044
```

```
y_pred = DTree.predict(X_test)
```

```
df=pd.DataFrame({'Actual':y_test, 'Predicted':y_pred})
df
```

Out[151]:

	Actual	Predicted
17703	3.332205	3.295837
6877	3.496508	3.663233
18216	0.000000	0.000000
4429	0.000000	0.000000
3679	4.043051	4.043051
...	...	...
11730	0.000000	0.000000
2105	0.000000	0.000000
18070	2.564949	3.133600
2618	4.060443	4.060443
16414	3.295837	2.838531

3648 rows × 2 columns

```
print('Mean Absolute Error:', metrics.mean_absolute_error(y_test, y_pred))
print('Mean Squared Error:', metrics.mean_squared_error(y_test, y_pred))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_test, y_pred)))
```

```
Mean Absolute Error: 0.11758216302865347
Mean Squared Error: 0.10453867750130548
Root Mean Squared Error: 0.3233244152570379
```

Ridge Regression

Any dataset that exhibits multicollinearity can be analyzed using the model's tuning technique known as ridge regression. This technique carries out L2 regularization. Estimated values differ much from real values when the problem of multicollinearity arises, least-squares are unbiased, and variances are significant.

```
ridge_reg = Ridge(alpha=10)

ridge_reg.fit(X_train, y_train)
pred = ridge_reg.predict(X_test)
```

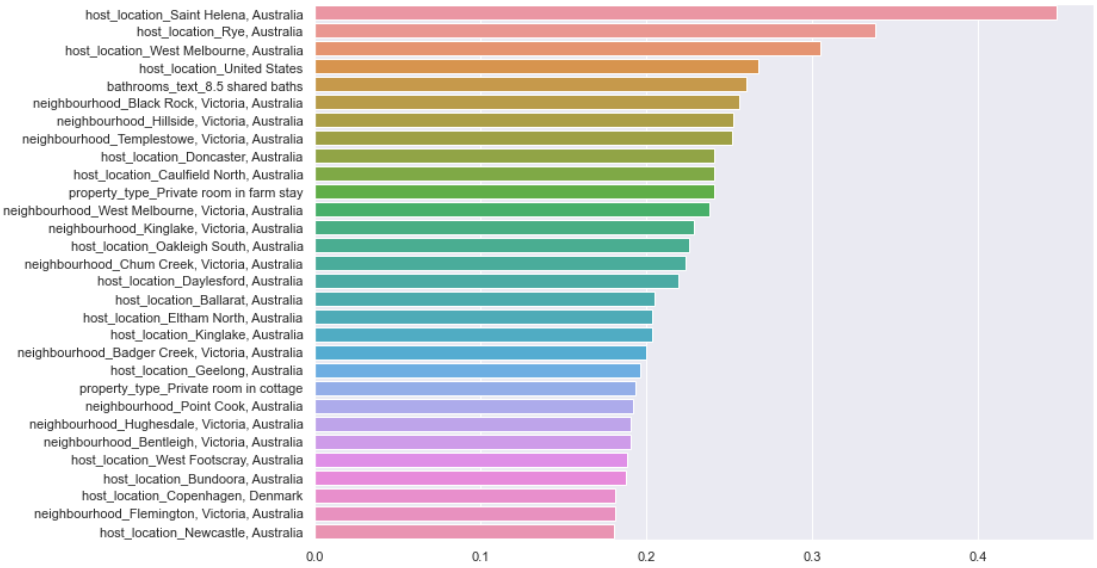
```
/Users/ishamore/opt/anaconda3/lib/python3.8/site-packages/sklearn/linear_model/_ridge.py:147: LinAlgWarning: Ill-conditioned
matrix (rcond=1.53819e-20): result may not be accurate.
  return linalg.solve(A, Xy, sym_pos=True,
```

```
pred = ridge_reg.predict(X_test)
```

```
mse = mean_squared_error(y_test, pred)
rmse = np.sqrt(mse)
Variance_score = r2_score(y_test,pred)

print('MSE: {0:.4f}, RMSE:{1:.4f}, Variance score:{2:.4f}'.format(mse,rmse,Variance_score))
```

```
MSE: 0.2765, RMSE:0.5258, Variance score:0.9077
```

```
coef = pd.Series(ridge_reg.coef_, index=X_features.columns)
coef_sort = coef.sort_values(ascending=False)[:30]
sns.barplot(x=coef_sort.values, y=coef_sort.index)
```



The above graph

Advantages and Disadvantages of Models

Advantages

- Decision trees take less work to prepare the data during pre-processing than other methods do.
- A decision tree does not require the normalization of data and scaling of data as well
- Additionally, the construction of a decision tree is not significantly impacted by missing values in the data.
- Technical teams and stakeholders can understand a decision tree model very quickly.

Disadvantages

- A slight change in the data can result in a big change in the decision tree's structure, which can lead to instability.
- Because of its intricacy and lengthier training period, decision tree training is relatively expensive.
- When compared to other algorithms, a decision tree's calculations can occasionally become significantly more complex.
- For the Decision Tree algorithm, regression applications and continuous value forecasts are insufficient.

Model Comparison:

Model	RMSE	MSE	Variance Score/Accuracy
Decision Tree Regressor	0.3233	0.1045	0.965
Random Forest Regressor	0.2436	0.0593	0.980
Ridge Regression	0.5258	0.2765	0.907

The above table explains the Random mean square error (RMSE), mean square error (MSE), and variance score. Among all the models Random Forest Regressor gives an accuracy of 98% whereas the decision tree regressor gives an accuracy of 96.5%.

Business Recommendation

By analyzing the dataset, it can be inferred that as the number of features in Airbnb are increasing the price is decreasing. Hence Airbnb can opt for a pricing model wherein the prices increase with an increase in the number of features.

Conclusion:

According to the above analysis, we can successfully conclude that the Random Forest Regressor worked the best and gave a variance score of 0.98 whereas the Ridge Regression model gave a variance score of 0.90. Hence the values predicted by Random Forest Regressor gave us close to accurate results. It is because the random forest uses many different samples in decision trees and that results in low variance. It can be concluded that the predicted value of the availability of Airbnb is close to the actual value.

Bibliography

Airbnb. (2022, October 16). *Inside Airbnb* . Retrieved from http://insideairbnb.com/get-the-data/

Create a Dashboard. Online Help. (n.d.). Retrieved October 15, 2022, from https://help.tableau.com/current/pro/desktop/en-us/dashboards_create.htm

Tableau Community Forums. (n.d.). Retrieved October 15, 2022, from https://community.tableau.com/s/idea/0874T000000HBAwQAO/detail

Sklearn.linear_model.Ridge. scikit. (n.d.). Retrieved October 29, 2022, from https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Ridge.html

Sklearn.tree.decisiontreeregressor. scikit. (n.d.). Retrieved October 29, 2022, from https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeRegressor.html