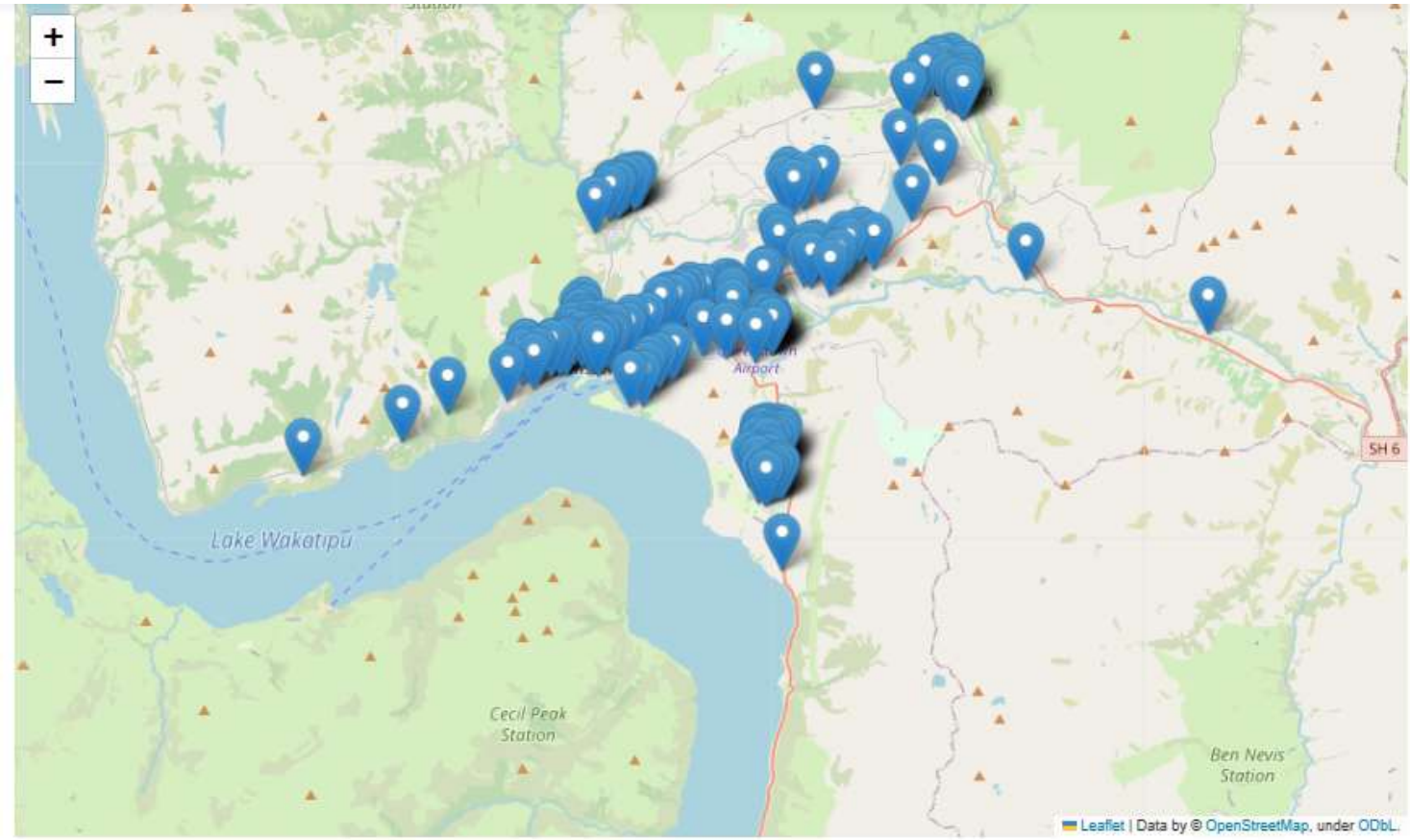
Predicting House Prices in the Queenstown-Lake Region

Questions from the Datasheets for Datasets paper, v7.

Sections:

- Motivation and Composition
- Collection process
- Preprocessing/cleaning
- Uses and Distribution
- Maintenance
- Conclusion



Motivation and Composition

1- For what purpose was the dataset created?

2 - Who created the dataset?

The database was created by Bruno Principi for educational purpose to apply newly acquired knowledge.

3 - What do the instances that comprise the dataset represent?

4 - How many instances are there in total?

There are 13 explanatory variables describing aspects of residential homes in Queenstown-Lake Region.

```
# The dataset contains 237 instances, 12 features, 1 label ('Price')
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 237 entries, 0 to 236
Data columns (total 13 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   Title         237 non-null    object 
 1   Full Address  237 non-null    object 
 2   Neighborhood  237 non-null    object 
 3   District      237 non-null    object 
 4   Latitude      237 non-null    float64
 5   Longitude     237 non-null    float64
 6   Street        237 non-null    object 
 7   Bedrooms      237 non-null    float64
 8   Bathrooms     237 non-null    float64
 9   Parking       237 non-null    float64
 10  Floor Area    237 non-null    float64
 11  Land Area     237 non-null    float64
 12  Price         237 non-null    object 
dtypes: float64(7), object(6)
memory usage: 25.9+ KB
```

5 - Does the dataset contain all possible instances or is it a sample (not necessarily random) of instances from a larger set?

The dataset contain all possible instances, in total there are 237 instances.

It is a small dataset since it was extracted with the following conditions:

Only houses or apartment for sale

Only published on the Trade Me website

Only for the Queenstown lake region.

Only data extracted on 04-18-2023

6 - What data does each instance consist of?

Title: title of the publication on the website

Full Address: contains street name, street number, neighborhood and district.

Neighborhood: name of the neighborhood.

District: name of the district.

Latitude: coordinate representing latitude.

Longitude: coordinate representing longitude.

Street: contains street number and street name.

Bedrooms: number of bedrooms.

Bathrooms: number of bathrooms.

Parking: number of parking place.

Floor Area: square meters of floor area.

Land Area: square meters of land area

Price: sale price.

7 - Is there a label or target associated with each instance?

‘Price’ - the property's sale price in New Zealand dollars. This is the target.

8 - Is any information missing from individual instances?

Some features have missing values that were filled with '0' or 'none' at the time the database was built. Then those values were replaced by making some kind of assumption.

Missing values were on:

- Bedrooms.
- Parking.
- Land Area.
- Neighborhood (there were wrong values).
- Floor Area
- Price

On the label ‘Price’, half of the instances doesn’t specified its value. This cause a huge reduction of the data available to train the model.

The instances without price value were used later to infer the price of a house based on the parameters learned by the best performance model.

9 - Are relationships between individual instances made explicit?

Yes, features related to the street, neighborhood, latitude and longitude are available.

10 - Are there recommended data splits (e.g., training, development/validation, testing)?

Considering the limitations given by the size of the database, it will be splitted 75% for training and 25% for testing.

11 - Are there any errors, sources of noise, or redundancies in the dataset?

- There were some errors detected on the feature ‘Neighborhood’.

12 - Does the dataset contain data that might be considered confidential?

Sensitive features related with street name, neighborhood, latitude and longitude could be considered confidential but they are necessary for the model to predict the sale price of the house. Furthermore, this information is also publicly accessible, just visiting the Trade Me website

13 - Does the dataset contain data that, if viewed directly, might be offensive, insulting, threatening, or might otherwise cause anxiety?

No, it doesn’t contain offensive data.

14 - Does the dataset relate to people?

No, it’s related with houses, its features and sale price.

Collection process

15 - How was the data associated with each instance acquired?

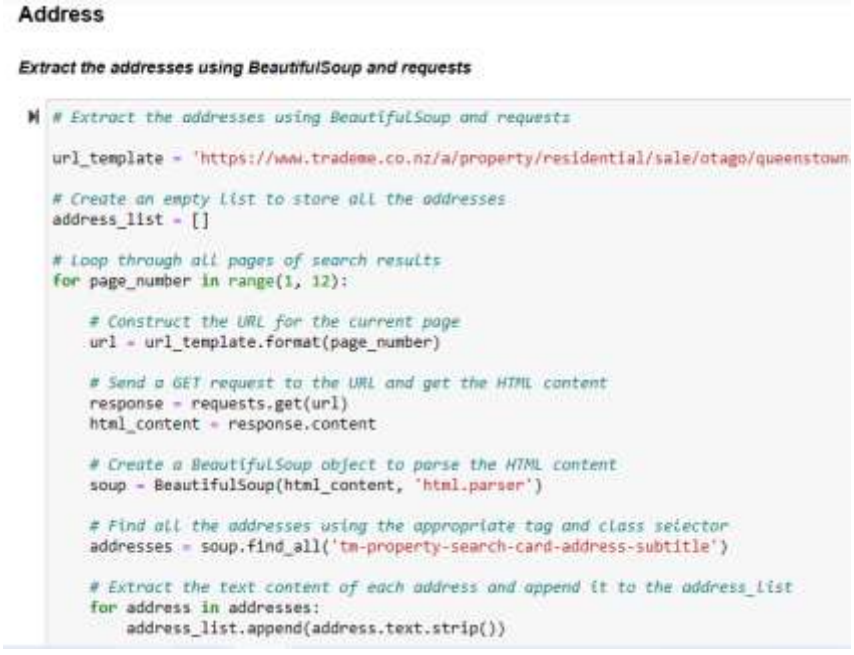
The data was acquired by web scraping using the Python libraries BeautifulSoup and requests. The script accessed a property search page on the Trademe.co.nz website and extracted the relevant information from each listing, such as the property address, bedrooms, bathrooms, parking, floor area, land area, and price. The address was used to extract the district and street information using regular expressions. Finally, the street information was used to extract the latitude and longitude coordinates using the Geopy library.

16 - What mechanisms or procedures were used to collect the data (e.g., hardware apparatus or sensor, manual human curation, software program, software API)?

The data was collected using a software program written in Python. The program used the BeautifulSoup and requests libraries to scrape data from the Trademe.co.nz website. Regular expressions were used to extract additional information from the property address, and the Geopy library was used to obtain the latitude and longitude coordinates of the property.

17 - Over what timeframe was the data collected?

The data was collected on a single day, April 18, 2023. The script was designed to extract data from the first 11 pages of search results for residential properties for sale in the Queenstown-Lakes district in Otago, New Zealand. However, it is possible that some properties were added or removed from the search results during the day.



Address

Extract the addresses using BeautifulSoup and requests

```
# Extract the addresses using BeautifulSoup and requests

url_template = 'https://www.trademe.co.nz/a/property/residential/sale/otago/queenstown

# Create an empty list to store all the addresses
address_list = []

# Loop through all pages of search results
for page_number in range(1, 12):

    # Construct the URL for the current page
    url = url_template.format(page_number)

    # Send a GET request to the URL and get the HTML content
    response = requests.get(url)
    html_content = response.content

    # Create a BeautifulSoup object to parse the HTML content
    soup = BeautifulSoup(html_content, 'html.parser')

    # Find all the addresses using the appropriate tag and class selector
    addresses = soup.find_all('tm-property-search-card-address-subtitle')

    # Extract the text content of each address and append it to the address_list
    for address in addresses:
        address_list.append(address.text.strip())
```

Data Cleaning

18 - Was any cleaning of the data done?

The following changes or imputations were done on the dataset:

- On the feature ‘Bedrooms’ the assumption was that a house must have at least 1 room. On instances with 0 rooms the value is replaced by 1.

```
In [37]: df[df['Bedrooms'] == 0]
```

Out[37]:

Index	Title	Full Address	Neighborhood	District	Latitud
6	SPECIAL ONE ONLY at $299,000	87-89 OBAN STREET, GLENORCHY, Glenorchy, Queen...	Glenorchy	Queenstown-Lakes	-44.85136

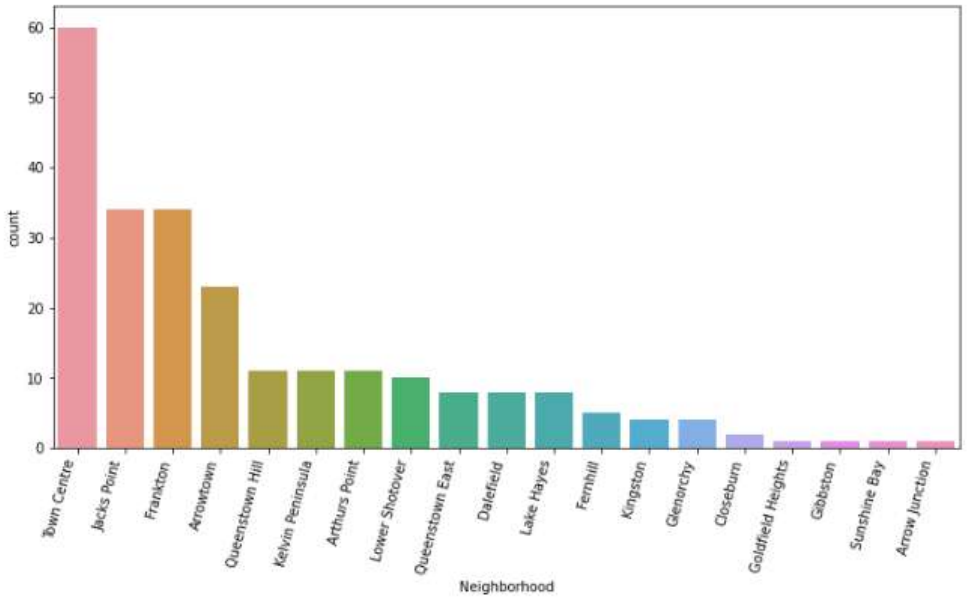
```
In [38]: df.loc[6,'Bedrooms'] = 1
```

```
In [39]: df.iloc[6]['Bedrooms']
```

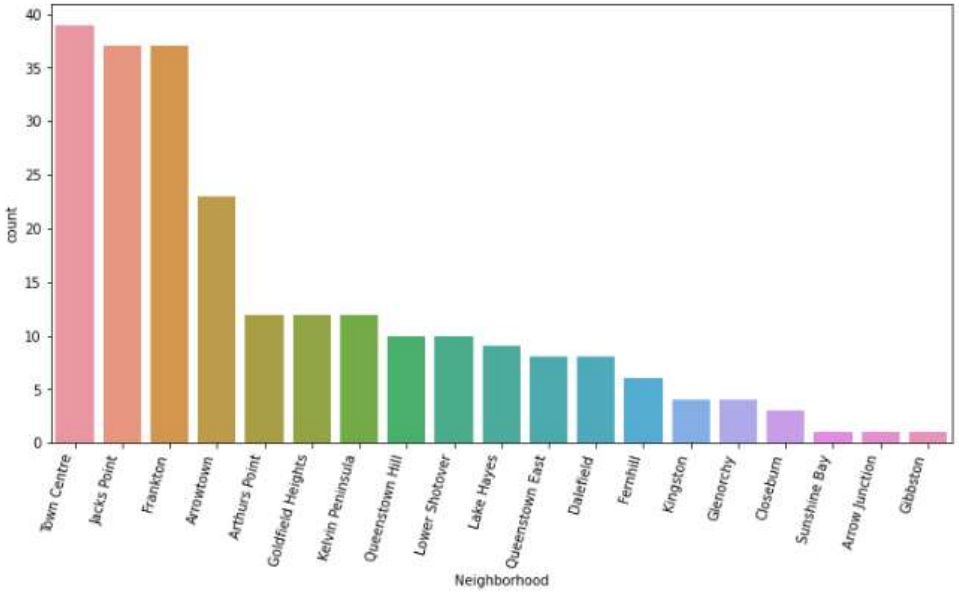
- To correct the 'Neighborhood' names where a house is located, first a polygon of each neighborhood's coordinates is created using Google Earth and the library shapely.geometry , then it is checked if the location of the house match with the neighborhood's name assigned to it, and finally it is corrected on the dataset if appropriate. After that the location of the houses is checked by drawing a map using the Folium library.

polygons of each neighborhood

Without correction



Corrected



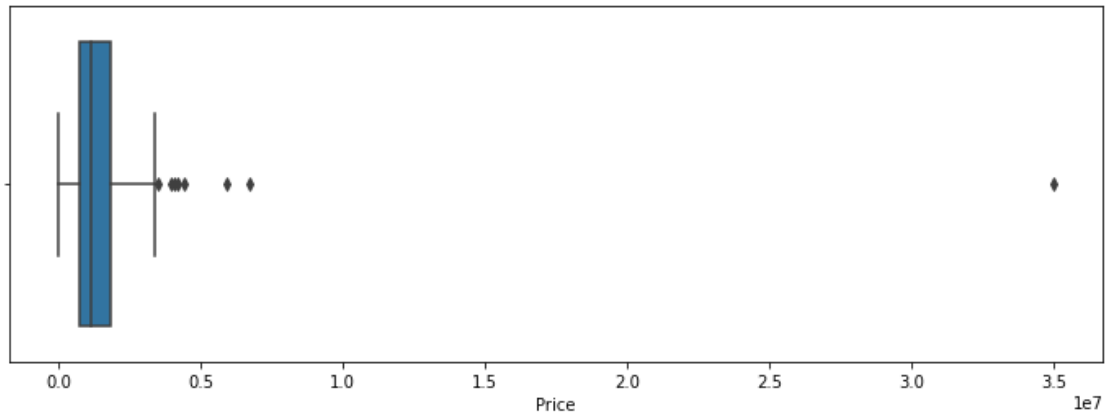
- On the label ‘Price’ were deleted o changed some outliers data points. For this, the relationship between 'Price' and 'Floor Area' was considered ('Floor Area' was the feature most correlated with price).

  - The instance at index 0 was deleted since it is referring to architectural plans that explain the low price.

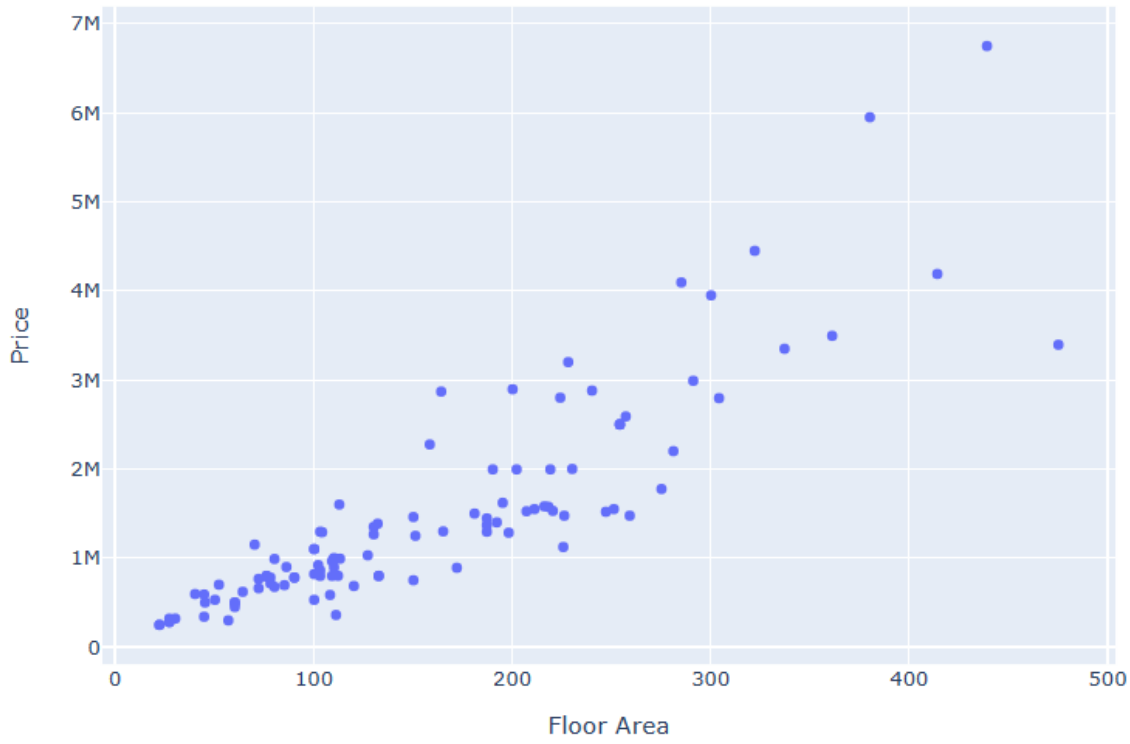
  - The instance at index 1 it is referring to a share sale. The price reflects a quarter of the true value, therefore to avoid delete the instance, the price will be multiplied by 4.

  - The instance at index 236 was deleted since it refers to a house that has characteristics of a boutique hotel, not a regular house.

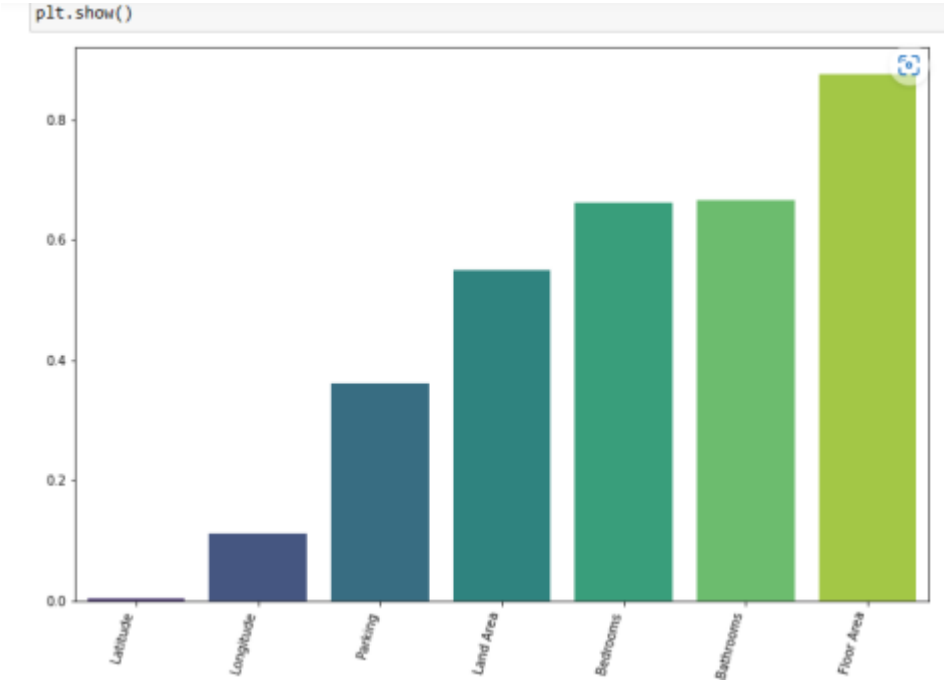
Boxplot ‘Price’



Scatterplot ‘Price’/ ‘Floor Area’

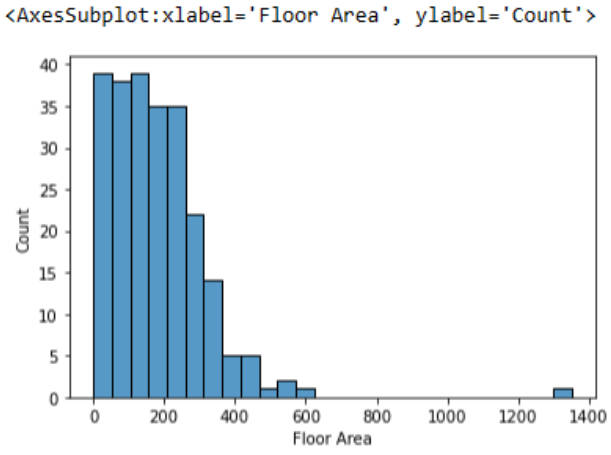


Correlations features / ‘Price’

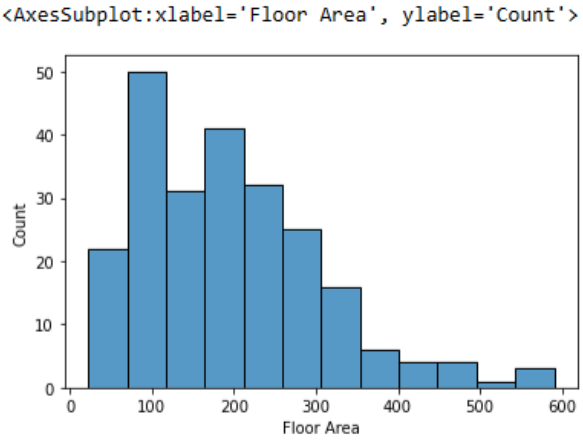


- On the feature 'Floor Area' the missing values were predicted using the Sklearn predictor IterativeImputer. Link: sklearn.impute.IterativeImputer — scikit-learn 1.2.2 documentation.

Distribution ‘Floor Area’ (m2) before



Distribution ‘Floor Area’ (m2) after



- The instances where the 'Price' label had missing values were deleted from the data set used to train the models.
- The features that presumably do not contribute to predicting the price were deleted. Those are: 'Title', 'Full Address', 'District', 'Street'.

```
# Assumption: Dropping features that doesn't add useful information to predict 'Price'
df_2.drop(['Title','Full Address','District','Street'],axis=1,inplace=True)
```

- One hot encoding the categorical features using OneHotEncoder (OneHotEncoder is a sklearn transformer that encode categorical features as a one-hot numeric array). Link: sklearn.preprocessing.OneHotEncoder — scikit-learn 1.2.2 documentation.

[195]:
```
# Create an instance of OneHotEncoder
onehot_encoder = OneHotEncoder(sparse=False, handle_unknown='ignore' )

# Store in a variable the numerical features of the dataframe
df_num = df_2.select_dtypes(exclude='object')

# Store in a variable the categorical features of the dataframe
df_obj = df_2.select_dtypes(include='object')

# Store in a variable the output of one hot encoding on categorical features
df_categorical = pd.DataFrame(onehot_encoder.fit_transform(df_obj),
                              columns=onehot_encoder.get_feature_names(),
                              index=df_obj.index)
# Concatenate the numeric and object dataframes
df_fi = pd.concat([df_num,df_categorical],axis=1)
# return df_onehot
```

- Standardization was applied on train and test set to fit some of the models created. Standardization of a dataset is a common requirement for many machine learning estimators: they might behave badly if the individual features do not more or less look like standard normally distributed data (e.g. Gaussian with 0 mean and unit variance).Centering and scaling happen independently on each feature by computing the relevant statistics on the samples in the training set. Mean and standard deviation are then stored to be used on later data using transform (Sklearn documentation).

  Link: 6.3. Preprocessing data — scikit-learn 1.2.2 documentation

StandardScaler

[204]:
```
# Create an instance of StandardScaler
stdr = StandardScaler()
```

[205]:
```
# Scale the features
X_train_scaled = stdr.fit_transform(X_train)
X_test_scaled = stdr.transform(X_test)
```

19 - Was the “raw” data saved in addition to the cleaned data (e.g., to support unanticipated future uses)?

Yes using the method .copy()

Uses and Distribution

20 - Has the dataset been used for any tasks already?

No, The dataset was created only for educational purposes.

21 - Is there a repository that links to any or all papers or systems that use the dataset?

GitHub Link:

22 - Is there anything about the composition of the dataset or the way it was collected and cleaned that might impact future uses?

There is a lot of room to improve the dataset, from collecting more data from different sources to creating new features using feature engineering.

23 - Are there tasks for which the dataset should not be used?

The dataset was created only for educational purposes.

24 - Will the dataset be distributed to third parties outside of the entity (e.g., company, institution, organization) on behalf of which the dataset was created?

The database is publicly accessible on the GitHub link provided.

25 - How will the dataset will be distributed (e.g., zip file, website, GitHub)?

Once the data cleaning process is done, the dataset will be uploaded to GitHub.

26 - How can the owner/curator/manager of the dataset be contacted (e.g., email address)?

The owner could be contacted through GitHub o by email.

27 - Will the dataset be updated (e.g., to correct labeling errors, add new instances, delete instances)?

Probably not.

28 - If others want to extend/augment/build on/contribute to the dataset, is there a mechanism for them to do so?
Yes, the dataset will be publicly available on the GitHub repository linked.

Models performance:

Since this is a regression problem where, given a set of features, the goal is to predict the sale price of a house. To achieve that goal 3 estimators will be tested, these are: LinearRegression, ElasticNet, RandomForestRegressor.

The metric used to evaluate the performance was **Root Mean Squared Error (RMSE).**

Models performance:

LinearRegression **RMSE = 364.461**

ElasticNet **RMSE = 392.684**

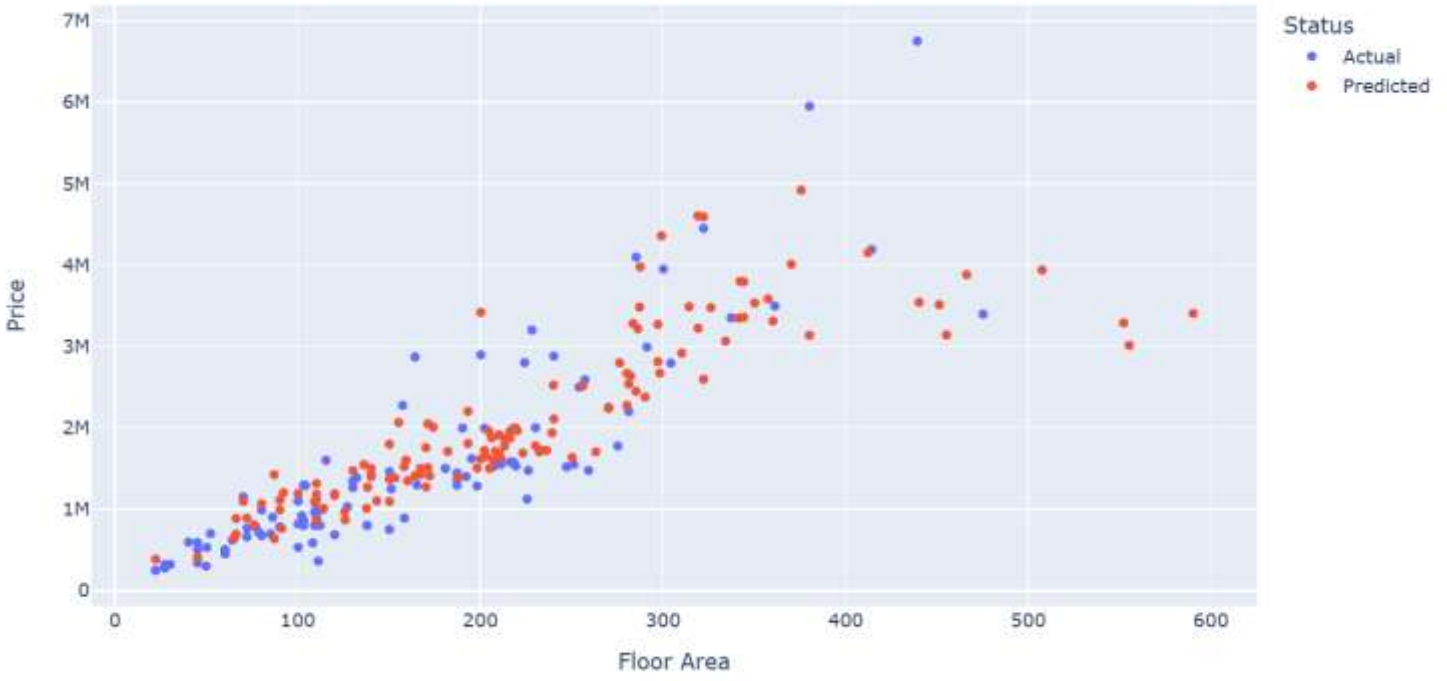
RandomForestRegressor **RMSE = 310.569**

Conclusion:

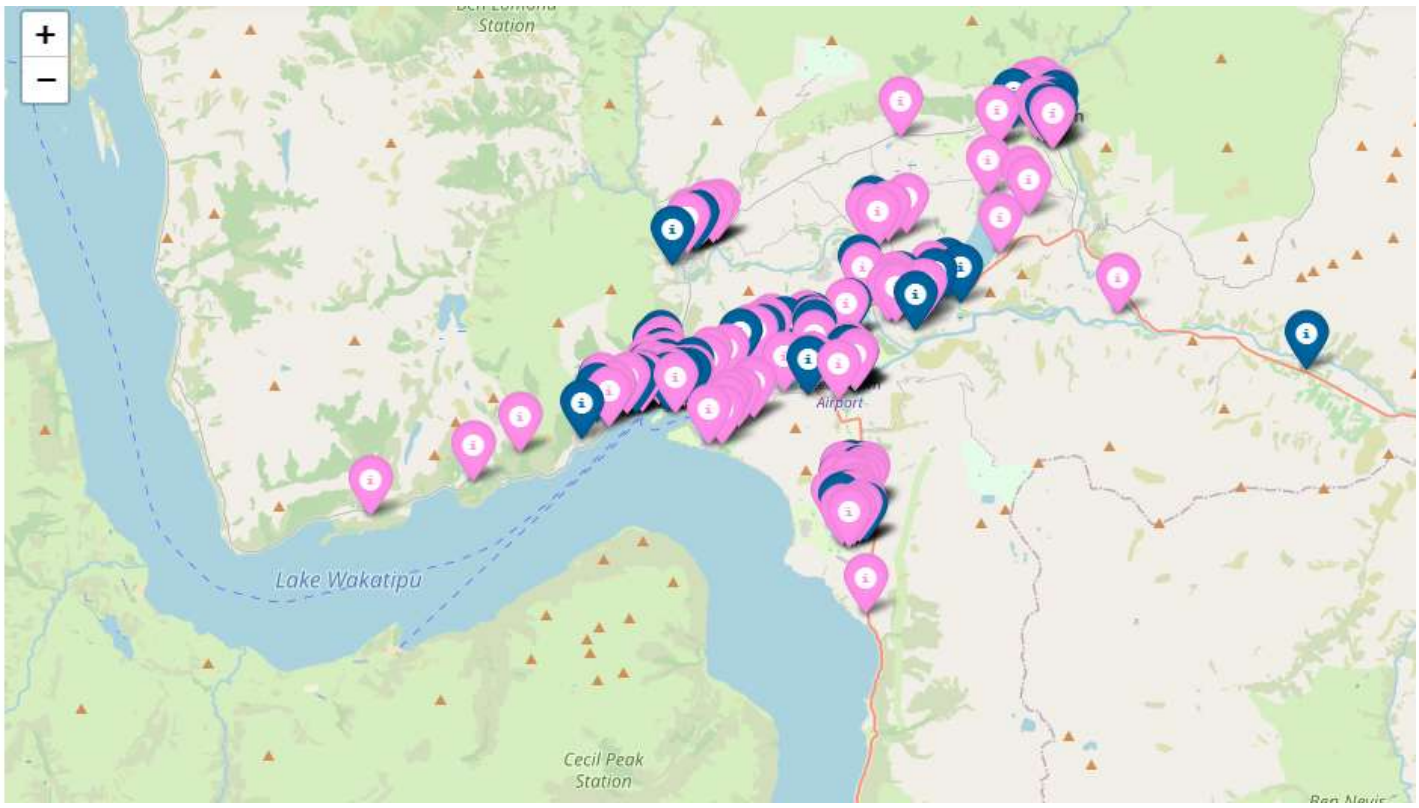
The best performance on the training set was achieved by RandomForestRegressor estimator.

Comparing the mean sale price of a house, around NZD $1,450,000, the RMSE value indicates that the model has an average error of around 20% of the mean price. This is not ideal, but considering the limitations of the data set, it is not bad. The models performance could be improved collecting more data, performing feature engineering or applying different techniques and algorithms.

Relation between 'Floor Area' / the predicted and actual values on 'Price'



The maps below show the houses with predicted price (pink marker) and the houses used to train the model (dark blue marker)




```
# Display the map object
display(mapa)
```