Big Data Predictive Analytics

Regression & Clustering

Report

The Manhattan real estate sales prices from August 2012 to August 2013 are documented in the Housing dataset. The dataset contains data on square footage, building type categories, neighbourhoods, selling prices, and other characteristics.

Our objective in this research is to create a model that can predict house selling prices based on a few relevant dataset attributes. The variables contain details regarding the BOROUGH, NEIGHBORHOOD, BUILDING CLASS CATEGORY, TAX CLASS AT PRESENT, BLOCK, LOT, EASE-MENT, BUILDING CLASS AT PRESENT, ADDRESS, APARTMENT NUMBER, ZIP CODE, RESIDENTIAL UNITS, COMMERCIAL UNITS, TOTAL UNITS, LAND SQUARE FEET, GROSS SQUARE FEET, YEAR BUILT, TAX CLASS AT TIME OF SALE, BUILDING CLASS AT TIME OF SALE, SALE PRICE, SALE DATE.

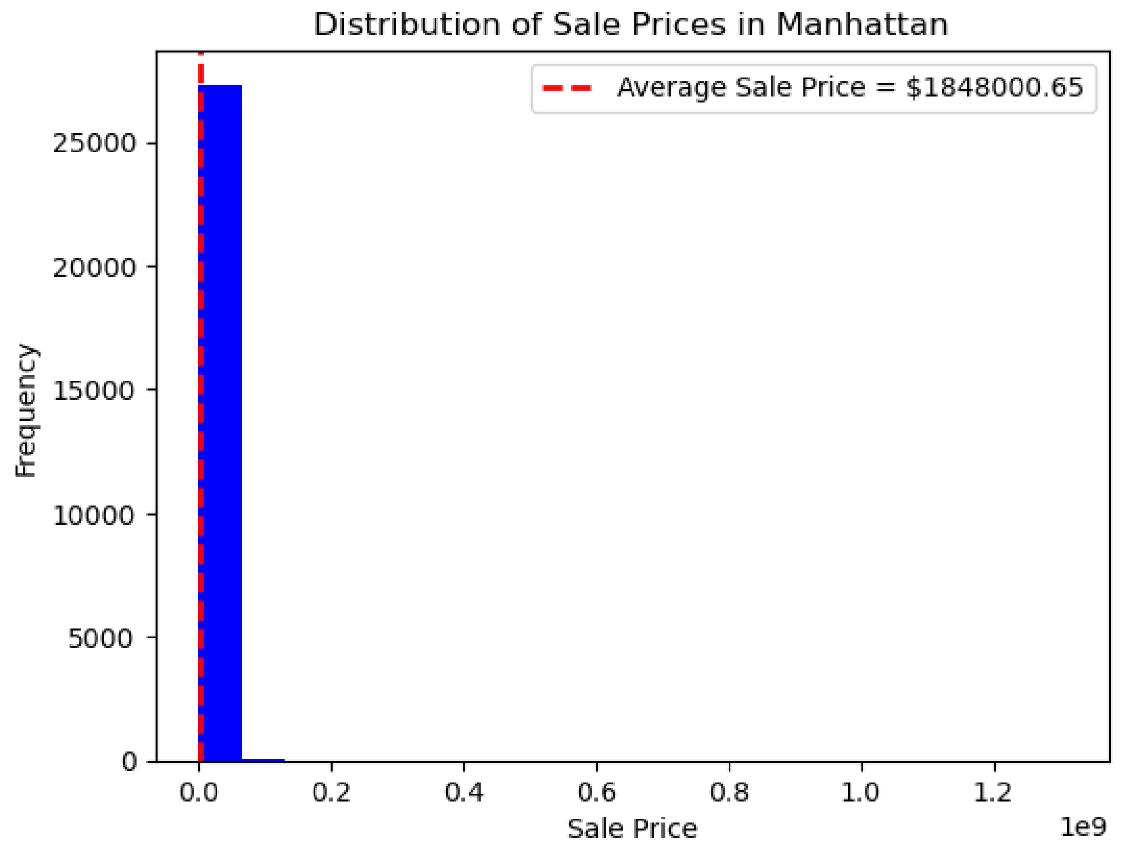
Phase 1: Data Cleaning & Transformation

First the relevant data is collected i.e., **pre-processing** the data. This involved preparing the data for analysis by cleaning it, dealing with missing values, and transforming it into an appropriate format. We also converted the 'SALE DATE' to datetime format and removed the '$' and ',' characters from the sale price and other numerical columns. Finally, we replaced zeros in prices, land squares, etc. with NaN values. To find any outliers or anomalies in the data that might have an impact on the success of the model, we also conducted exploratory data analysis.

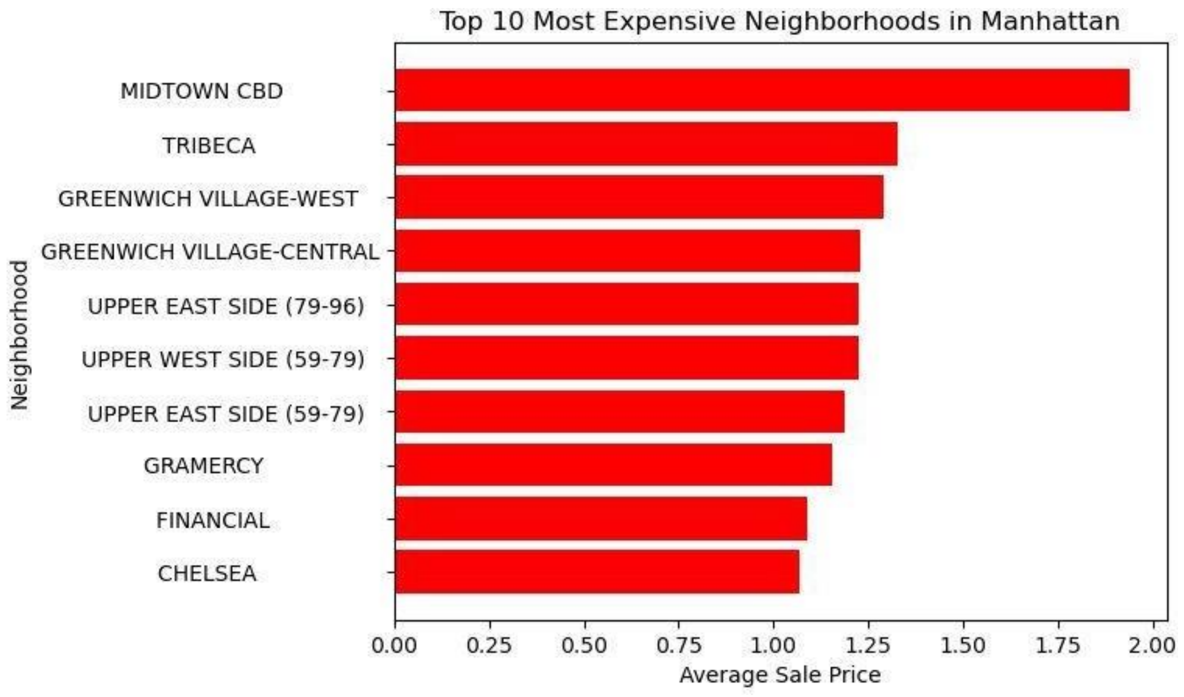
The resulting dataset had 27,395 rows and 21 columns.

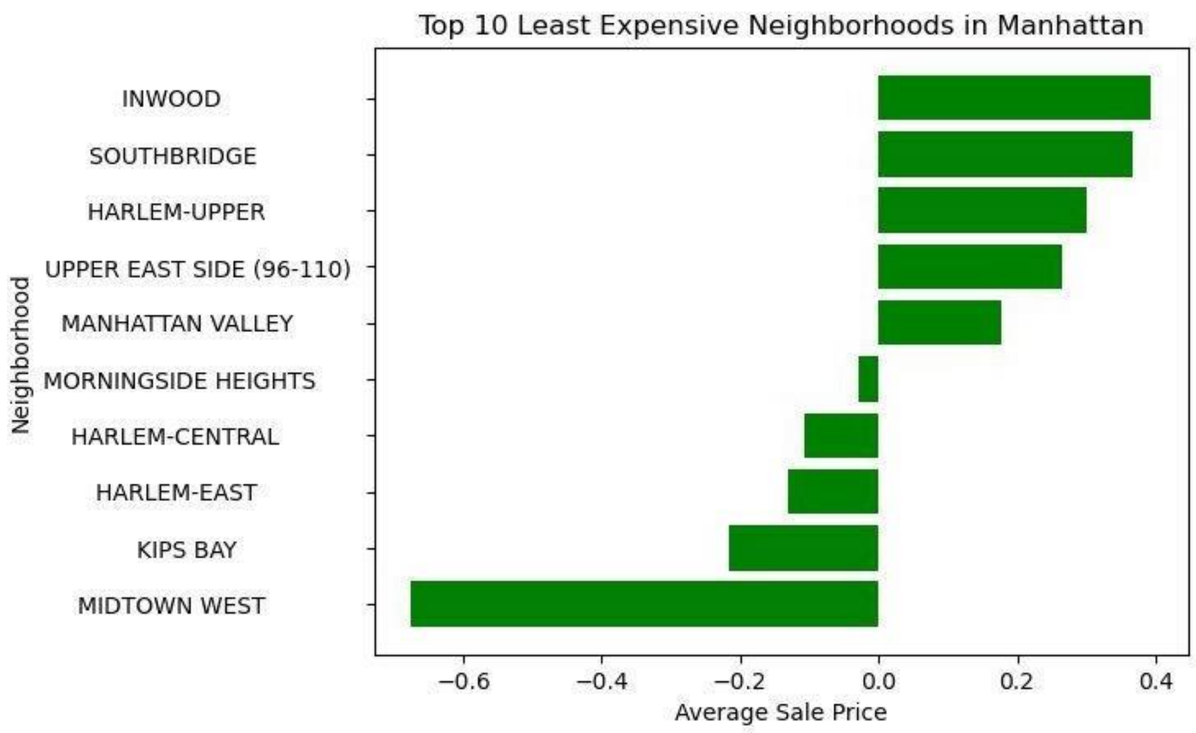
Phase 2: Data Exploratory

We examine the data and comprehend how the various factors are related before building a model. For analysing the data, we used a variety of techniques, including summary statistics, data visualization, and correlation analysis. Here are a few of the findings:

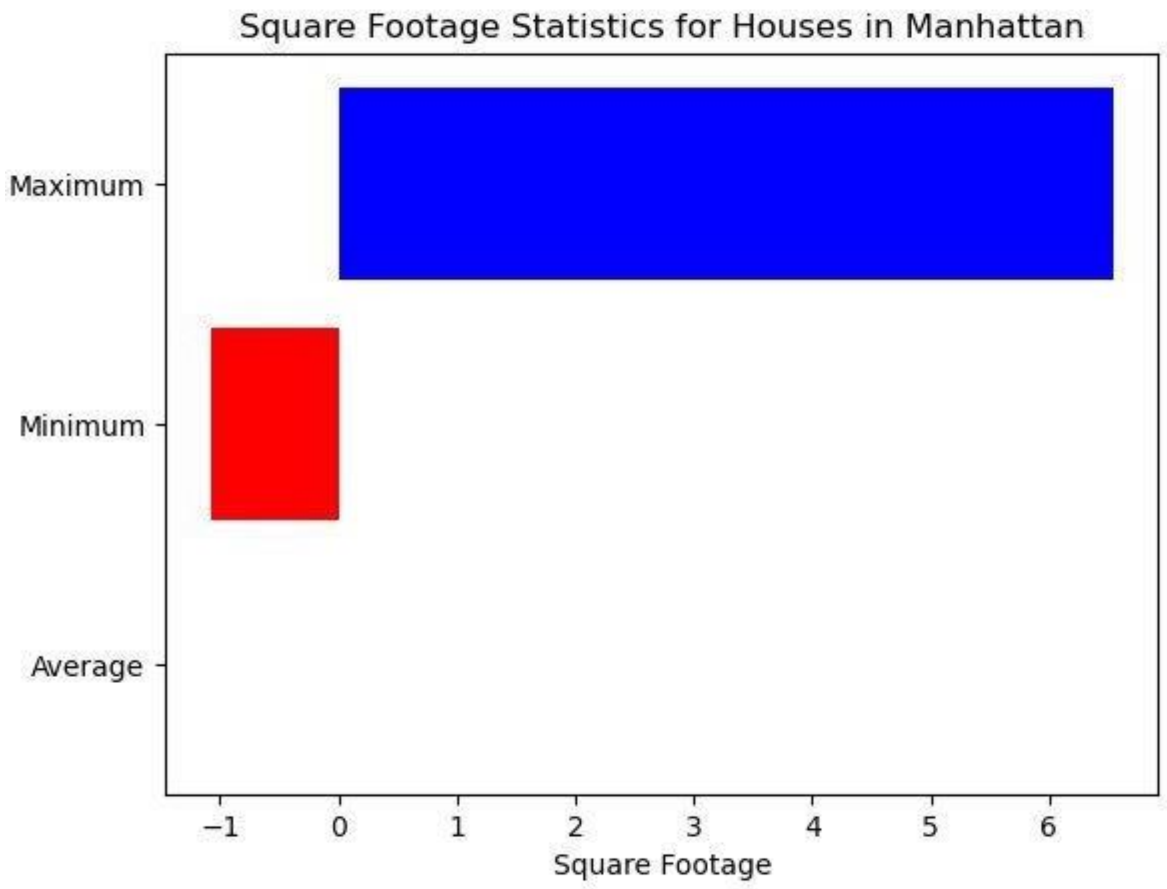


The average sale price of a house in Manhattan is $1848000.65, with a minimum of $0.00 an d a maximum of $1307965050.00.

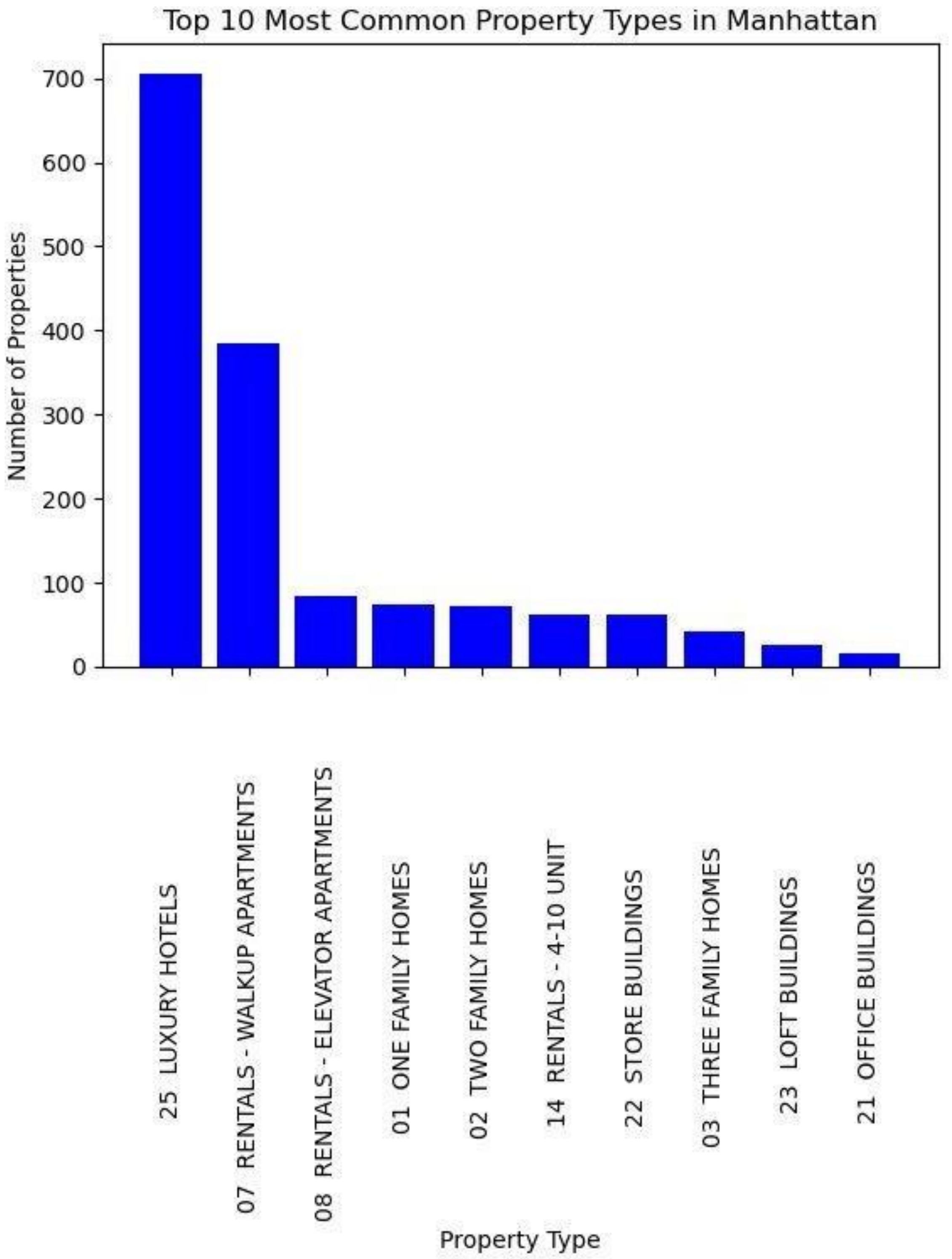




The most expensive neighbourhoods in Manhattan are Midtown CBD, while the least expensive neighbourhoods are Inwood in comparison to the average sales price value of the dataset.

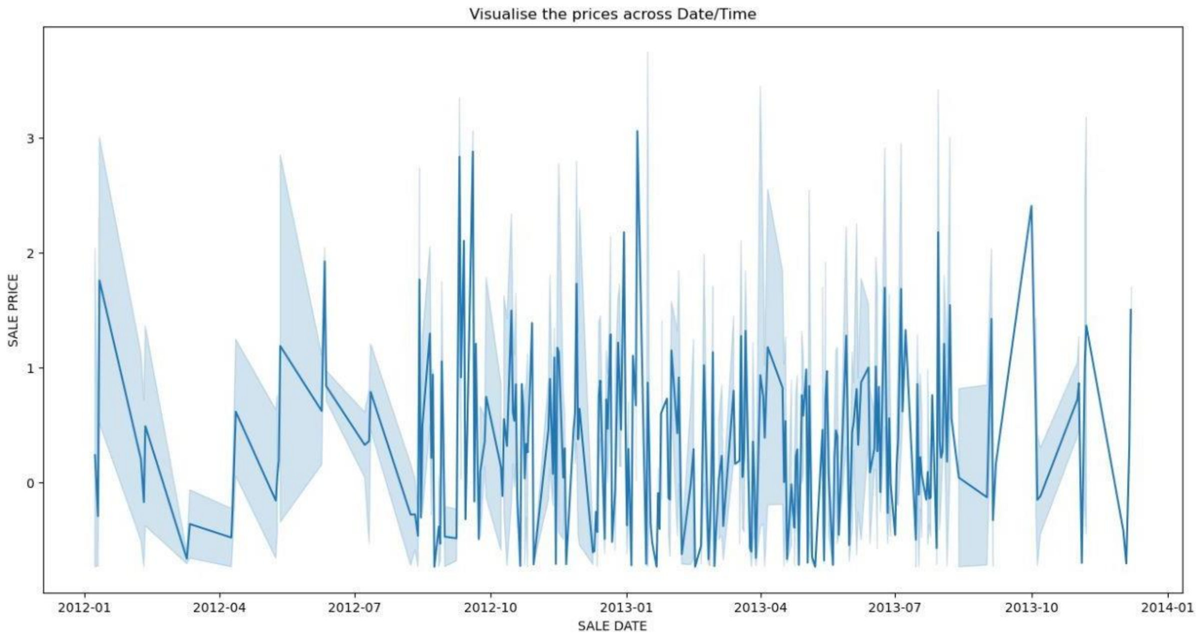


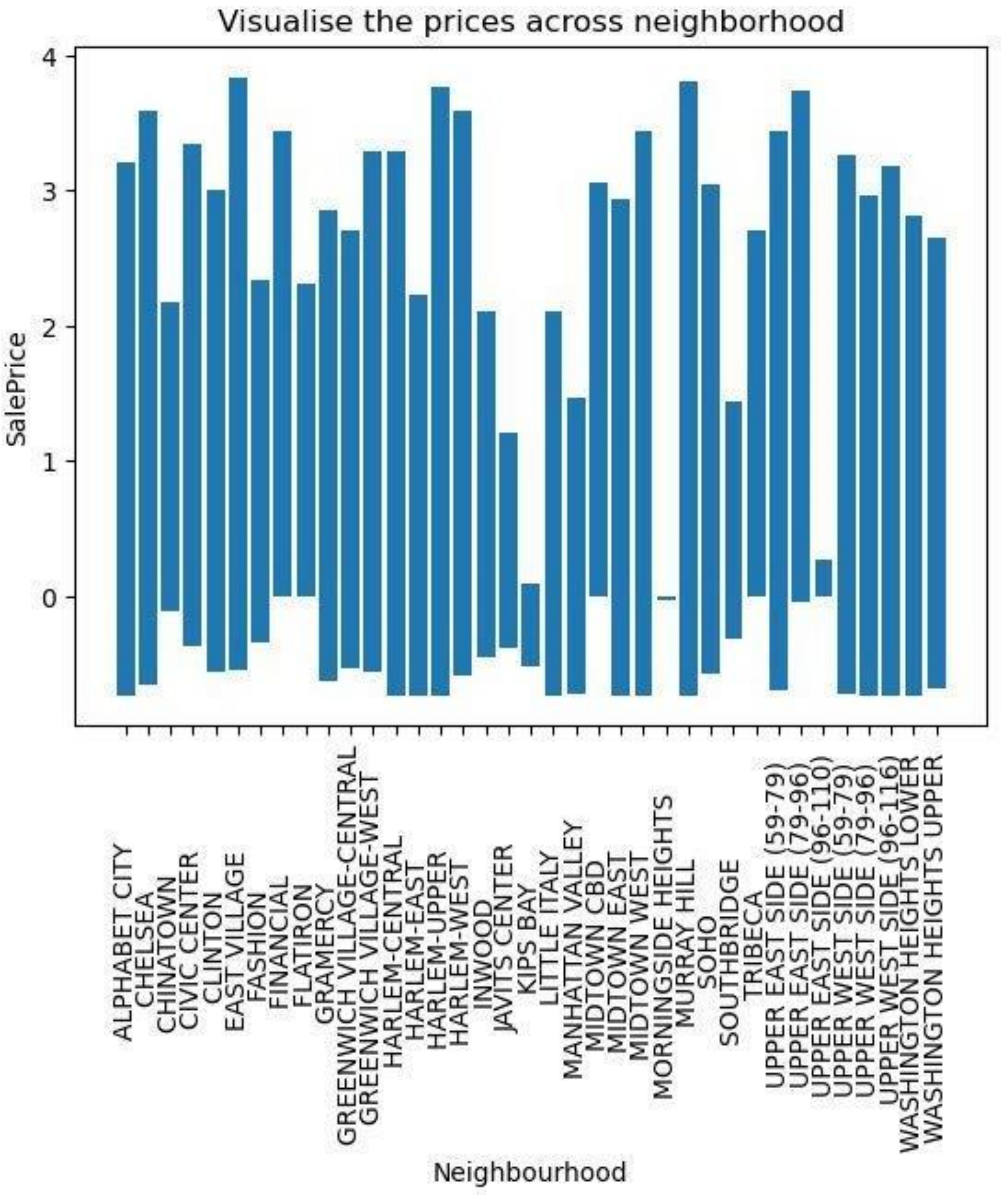
The average square footage of a house in Manhattan is 9572.0263 sqft, with a minimum of 0.0 sqft and a maximum of 1970736 sqft.

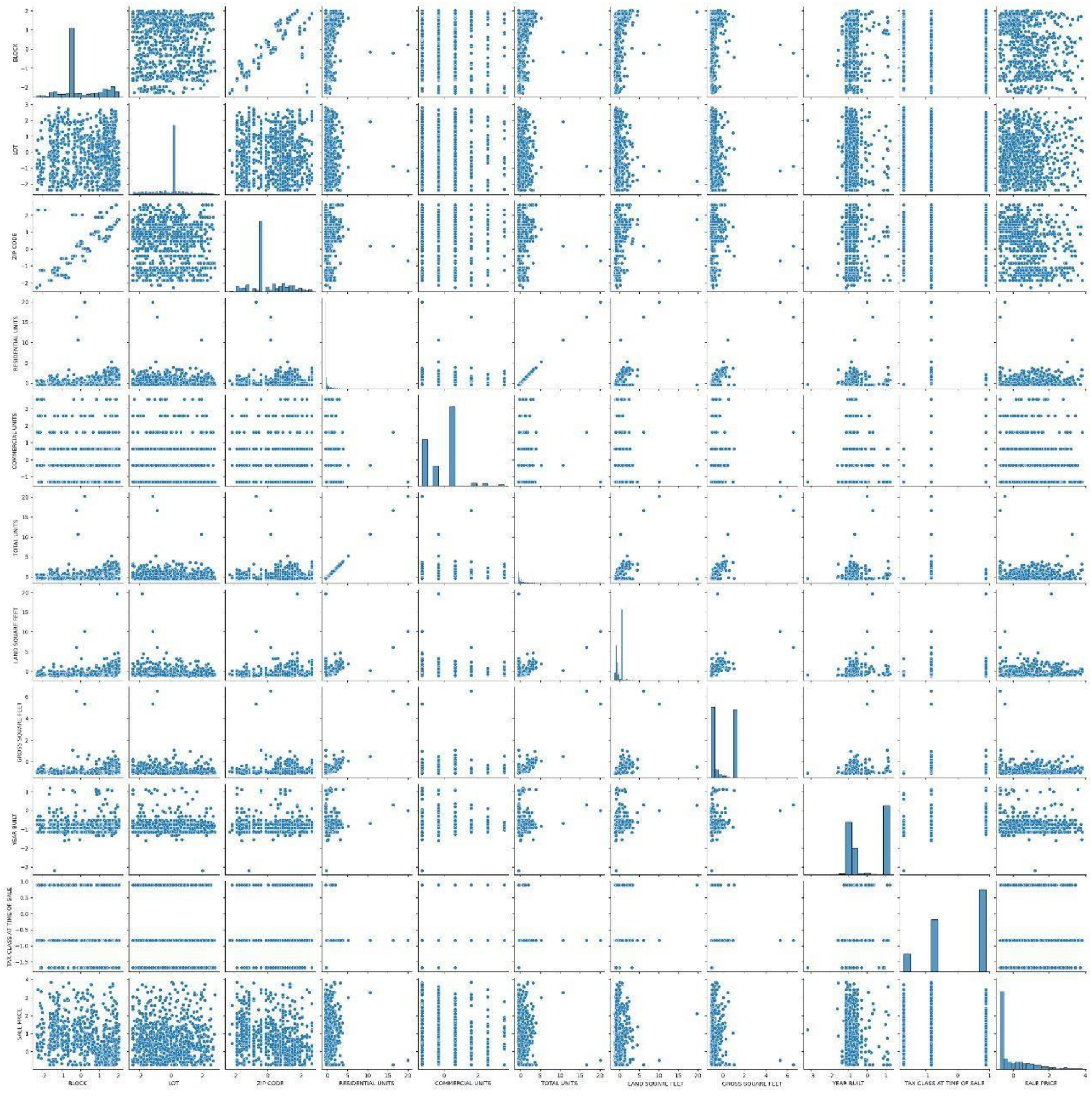


The most common property types in Manhattan are LUXURY HOTELS and RENTALS - EL EVATOR APARTMENTS

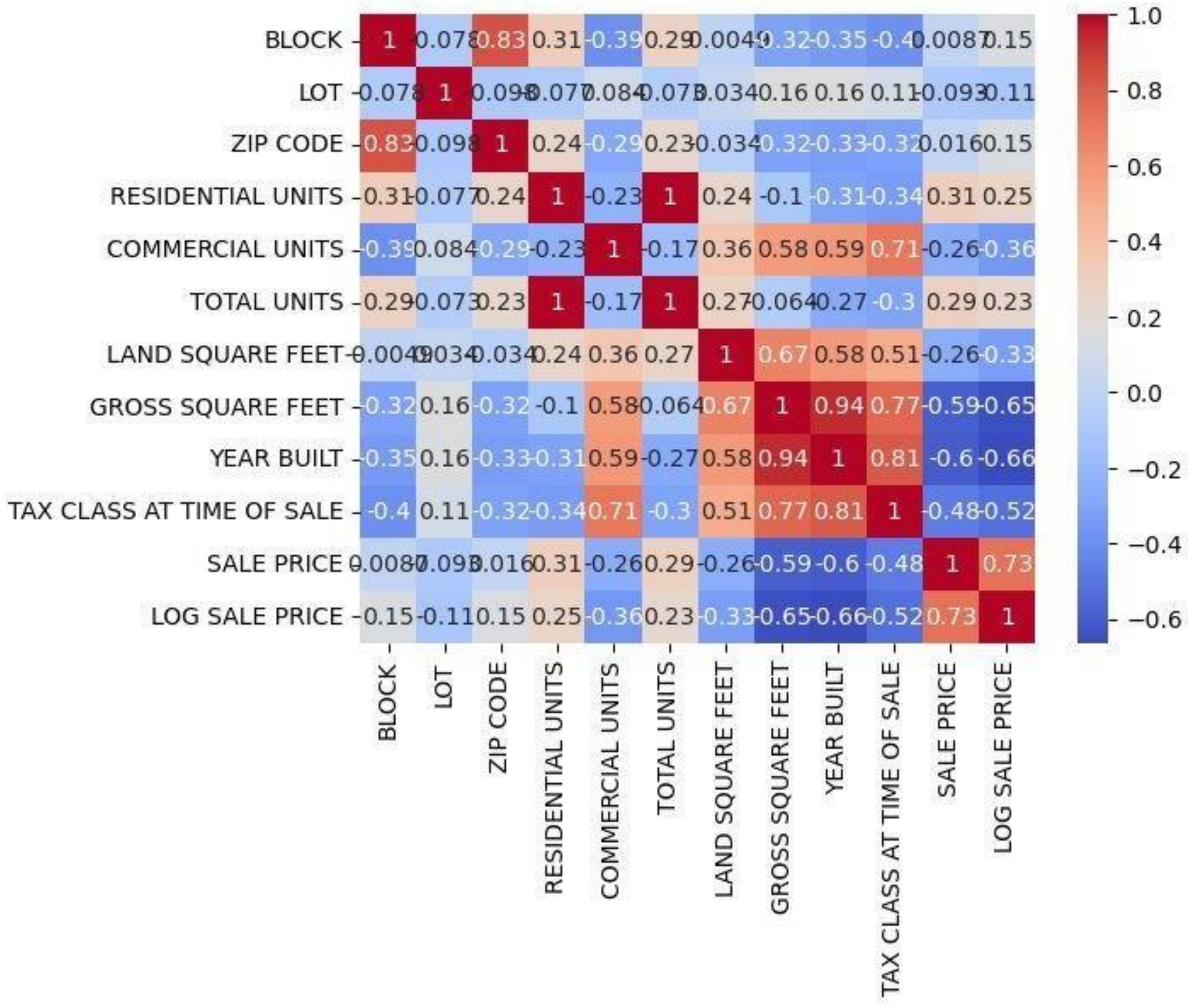
The below two bar graphs represents the comparison of Neighbourhood & Time across Sale Price respectively.







The above the scatter matrix plot determines the relationship among the features.



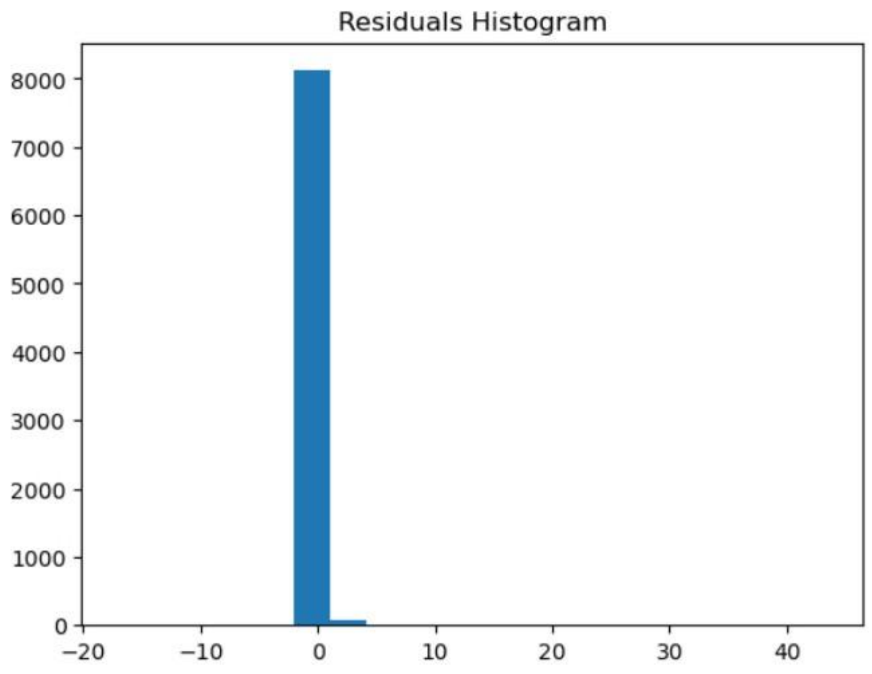
The observations from the above correlation matrix states that GROSS SQUARE FEET, YEAR BUILT, TAX CLASS AT TIME OF SALE, LOG SALE PRICE possess a strong correlation with the SALE PRICE. It also displays the features like BLOCK, LOT, ZIP CODE has weak correlations with sale price.

Phase 3: Building Model

Predicting the selling prices of housing is essential in the real estate sector for several reasons. Making well-informed decisions about buying or selling real estate is one of the main benefits. The process of developing a model that can forecast housing selling prices based on some pertinent dataset features will be covered in this phase.

In this phase we first select the predictors and target variable for our linear regression model. We split the data into training & testing sets (70-30 split) using **train_test_split()** function from scikit-learn. We built the linear regression model using the **LinearRegression()** class from scikit-learn and fit it on the training data using the **fit()** method.

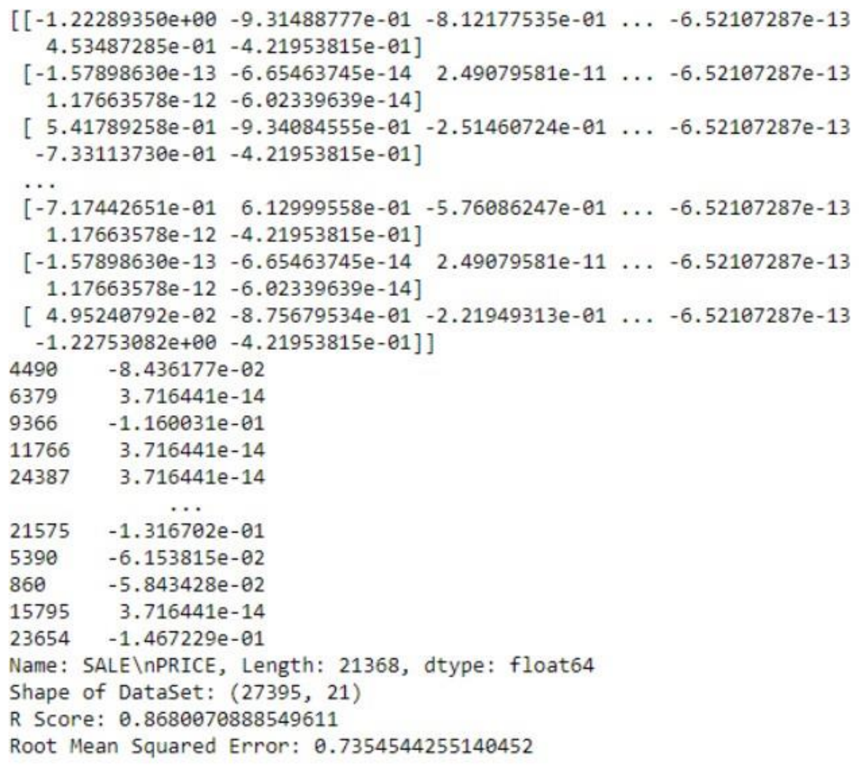
We evaluate the model on the testing data by calculating the Root Mean Square using **mean_squared_error()**. We also evaluate the model using cross- validation by calculating the average RMSE across 5 folds using the **cross_val_score().** And finally, we plot the histogram of residuals by subtracting the predicted the values from the actual values.



The results showed that our model had an R-squared value of 0.163 on the trained dataset, indicating that it can explain 16.3% of the variance in the sale prices. While these results indicate that our model has some predictive power, they also indicate that there is a lot of variability in the data that is not captured by our model. The Mean cross-validation score resulted is 0.01879.

Phase 4: Model Optimization & K-Means clustering.

After assessing the model's performance, we now improve it by altering the model's complexity and hyperparameters to get higher performance. For this we use the RandomForestRegression model. But this time we impute the missing values instead of dropping those. Similar steps followed previously for LinearRegression model are used here.



```
[[-1.22289350e+00 -9.31488777e-01 -8.12177535e-01 ... -6.52107287e-13
   4.53487285e-01 -4.21953815e-01]
 [-1.57898630e-13 -6.65463745e-14  2.49079581e-11 ... -6.52107287e-13
   1.17663578e-12 -6.02339639e-14]
 [ 5.41789258e-01 -9.34084555e-01 -2.51460724e-01 ... -6.52107287e-13
  -7.33113730e-01 -4.21953815e-01]
 ...
 [-7.17442651e-01  6.12999558e-01 -5.76086247e-01 ... -6.52107287e-13
   1.17663578e-12 -4.21953815e-01]
 [-1.57898630e-13 -6.65463745e-14  2.49079581e-11 ... -6.52107287e-13
   1.17663578e-12 -6.02339639e-14]
 [ 4.95240792e-02 -8.75679534e-01 -2.21949313e-01 ... -6.52107287e-13
  -1.22753082e+00 -4.21953815e-01]]
4490    -8.436177e-02
6379     3.716441e-14
9366    -1.160031e-01
11766    3.716441e-14
24387    3.716441e-14
             ...
21575   -1.316702e-01
5390    -6.153815e-02
860     -5.843428e-02
15795    3.716441e-14
23654   -1.467229e-01
Name: SALE\nPRICE, Length: 21368, dtype: float64
Shape of DataSet: (27395, 21)
R Score: 0.8680070888549611
Root Mean Squared Error: 0.7354544255140452
```

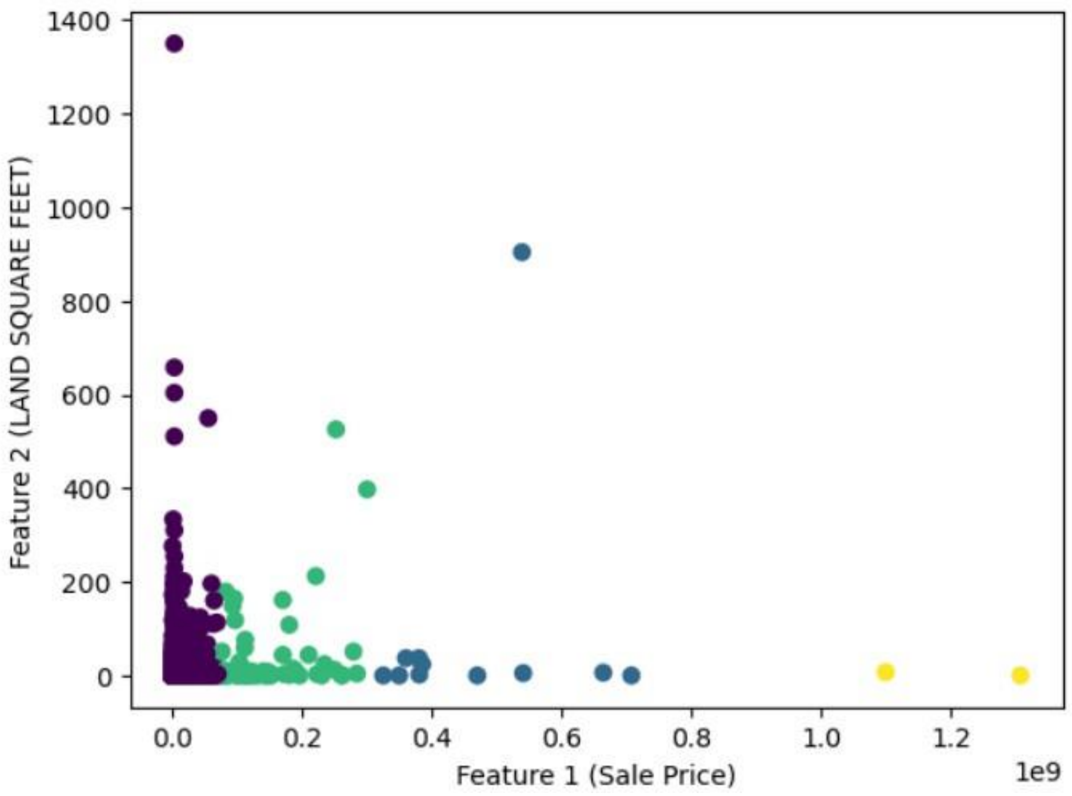
The improvised model using resulted RandomForestRegression in **R Score value of 86.8% & RMSE value by 73.54%.**

We proposed a nontrivial collection of housing clusters using a K-Means technique, which enhance the performance of the regression model by suggesting clusters-based (or local) regression models. The ideal number of clusters was determined using the elbow approach. The elbow plot indicates that **four clusters** are the ideal number. Each home was then allocated to one of the 4 clusters using the K-Means technique, which we then applied to the dataset. We initialize the K-Means model with the chosen number of clusters, the k-means++ initialization method, 300 maximum iterations, 10 initializations, and a random state of 0. We calculate the silhouette score using the silhouette_score() method from sklearn.metrics

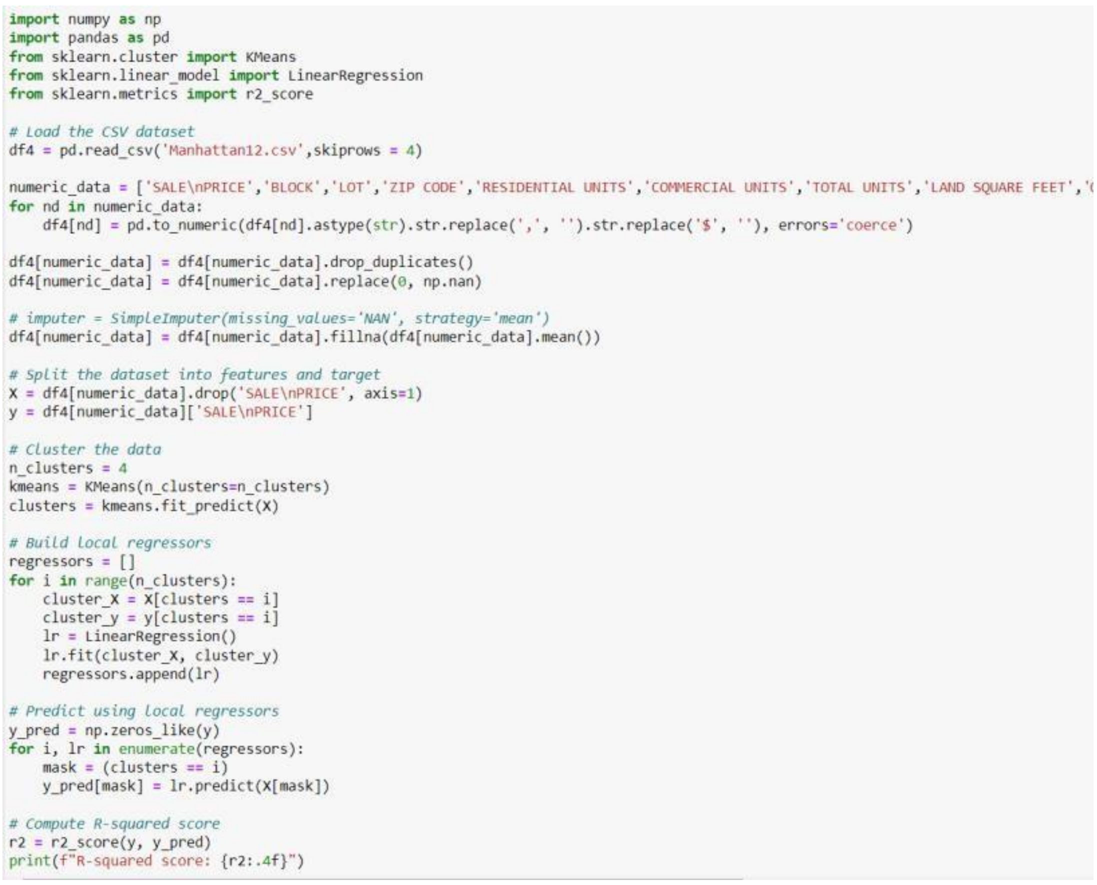
We visualize the clusters using a scatter plot with feature 1 on the x-axis, feature 2 on the y-axis, and the predicted clusters as the colour using the scatter() method from pyplot.

To justify the clustering, we examine the silhouette score. A higher silhouette score indicates better clustering, where the average distance between points in each cluster is low and the average distance between points in different clusters is high. We can also visually inspect the scatter plot to see if the clusters are well-separated and distinct from each other.

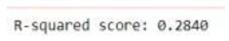
```
The average silhouette score is:  0.9832483645346444
```



We also built up a local regressor based on our previous cluster we created. The data is clustered using K-means, each cluster is assigned with the local linear regressor which predicts the target variable. Finally, it computes the R-squared score to evaluate the performance of the local regressors.



```python
import numpy as np
import pandas as pd
from sklearn.cluster import KMeans
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score

# Load the CSV dataset
df4 = pd.read_csv('Manhattan12.csv',skiprows = 4)

numeric_data = ['SALE\nPRICE','BLOCK','LOT','ZIP CODE','RESIDENTIAL UNITS','COMMERCIAL UNITS','TOTAL UNITS','LAND SQUARE FEET','
for nd in numeric_data:
    df4[nd] = pd.to_numeric(df4[nd].astype(str).str.replace(',', '').str.replace('$', ''), errors='coerce')

df4[numeric_data] = df4[numeric_data].drop_duplicates()
df4[numeric_data] = df4[numeric_data].replace(0, np.nan)

# imputer = SimpleImputer(missing_values='NAN', strategy='mean')
df4[numeric_data] = df4[numeric_data].fillna(df4[numeric_data].mean())

# Split the dataset into features and target
X = df4[numeric_data].drop('SALE\nPRICE', axis=1)
y = df4[numeric_data]['SALE\nPRICE']

# Cluster the data
n_clusters = 4
kmeans = KMeans(n_clusters=n_clusters)
clusters = kmeans.fit_predict(X)

# Build local regressors
regressors = []
for i in range(n_clusters):
    cluster_X = X[clusters == i]
    cluster_y = y[clusters == i]
    lr = LinearRegression()
    lr.fit(cluster_X, cluster_y)
    regressors.append(lr)

# Predict using local regressors
y_pred = np.zeros_like(y)
for i, lr in enumerate(regressors):
    mask = (clusters == i)
    y_pred[mask] = lr.predict(X[mask])

# Compute R-squared score
r2 = r2_score(y, y_pred)
print(f"R-squared score: {r2:.4f}")
```



```
R-squared score: 0.2840
```

Conclusion:

In the conclusion, we have explored the Manhattan housing dataset and built a linear regression model that predict the sale price of a property based on its features. We have also used the RandomForest as an improvised predictive model, the model has an R-squared value of 0.868. We have also performed clustering analysis to group similar properties based on their features and built local regressors for each cluster. Our results suggest that clustering the data can improve the predictive power of our model by capturing the variability in the data that is not captured by a global model.