Captsone Project

Presentation

Business Problem

Quick Presentation of the business objectives of this project

Data used

Presentation of the data used for this project

Methodology

Presentation of the model used, the data pre-processing & model evaluation

Results

Numbers and interpretation

Business Problem

Price Modeling

Do venues impact the price of real estate? If so, how?

For this project, I decided to model the relation between the price of an estate and the type of venues in its neighborhood. The objective is to determine whether or not some venues have a direct impact on the price, and if so how is the price affected by it.

Price Modeling

Target audience

Real Estate Investors

This problem may first and foremost generate the interest of potential real estate investors who want to estimate accurately the real value of an estate, as well as their ROI.

Homeowner

Future owners might be also mainly interested by those sort of studies, as they always struggle to negotiate the price of the estate they want to buy.

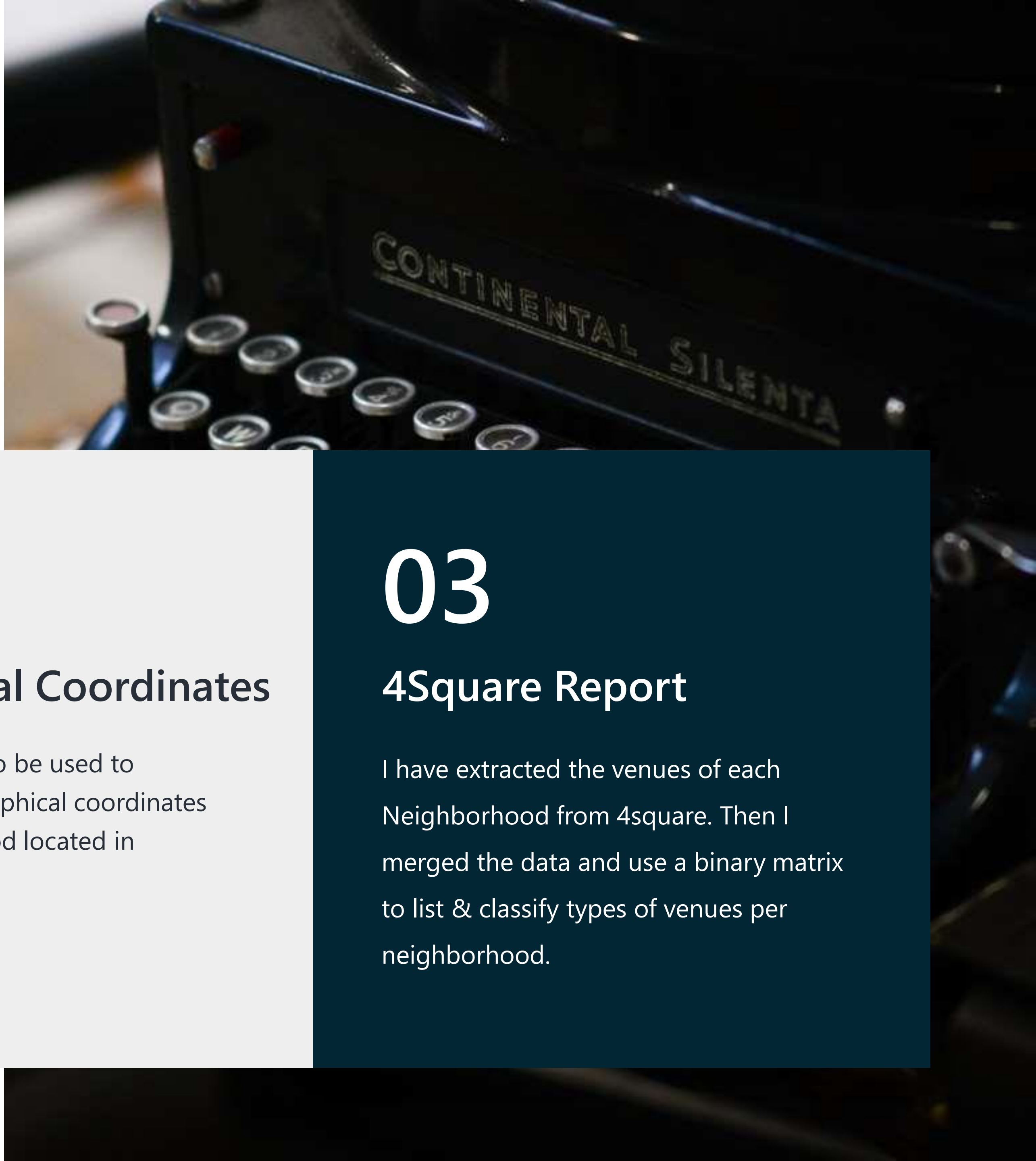
Municipalities

Policy makers may want to promote specific investments to impact the price of a given borough or neighborhood.

Data used


enter
return
shift
option
alt
command
Year 5
Year 6
60,000.00
50,000.00
40,000.00

Data Set



01

Property rolling sales

A data set that contains all the sales transactions of real estates that occured in Manhattan from 11-18 to 10-19.
I had to remove the rows with NaN values, transform the dtypes, group my data by neighborhood and calculate the average price per neighborhood per square meter

02

Geographical Coordinates

A second data had to be used to withdraw the geographical coordinates of each neighborhood located in Manhattan.

03

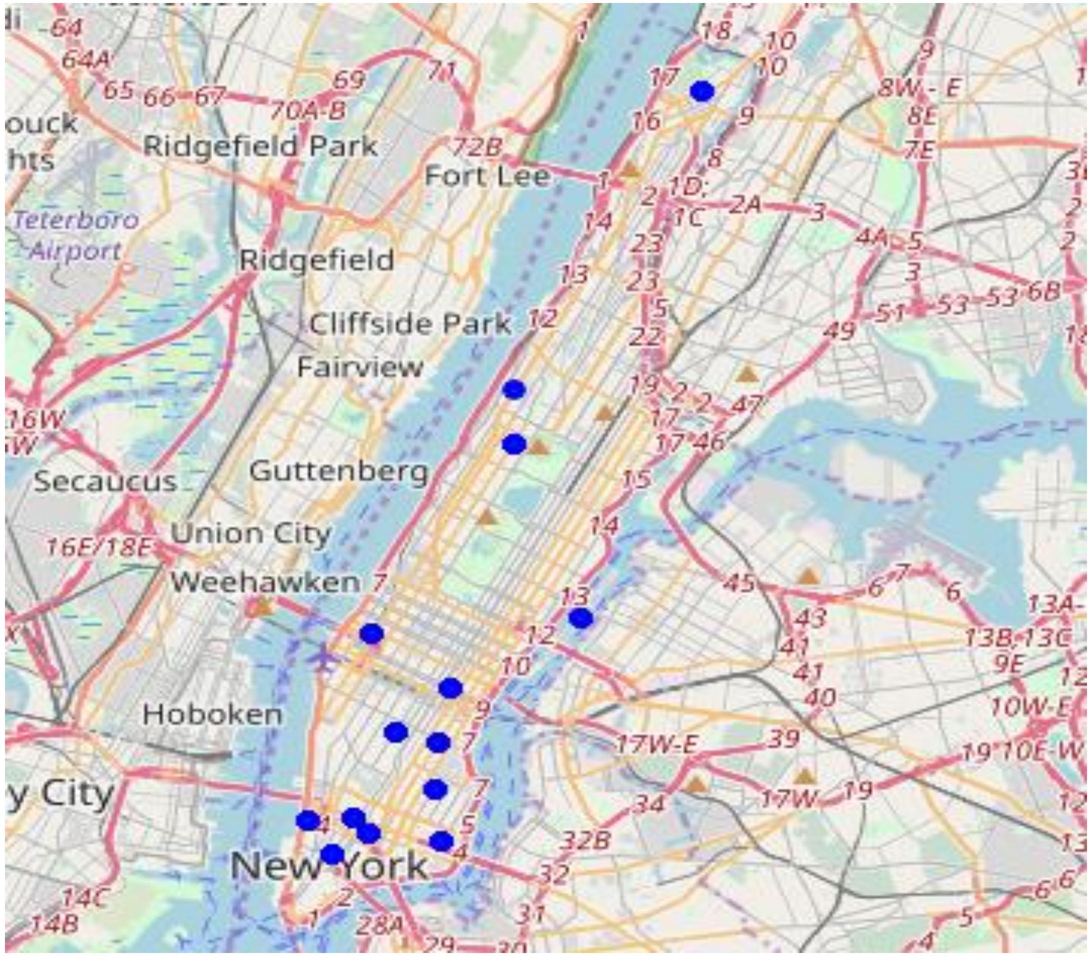
4Square Report

I have extracted the venues of each Neighborhood from 4square. Then I merged the data and use a binary matrix to list & classify types of venues per neighborhood.

My first two data set concatenated

NEIGHBORHOOD	GROSS_SQUARE_FEET	SALE_PRICE	Latitude	Longitude	Price/m2
CIVIC CENTER	4.423667	7.026654e+06	40.715229	-74.005415	2.056424e+06
CLINTON	437.652183	2.426260e+06	40.759101	-73.996119	5.123421e+05
EAST VILLAGE	225.606312	5.605654e+06	40.727847	-73.982226	1.017368e+06
FLATIRON	168.858676	1.542258e+07	40.739673	-73.990947	2.340082e+06
GRAMERCY	351.718578	4.411438e+06	40.737210	-73.981376	9.684232e+05
INWOOD	103.560471	2.679084e+06	40.867684	-73.921210	1.268207e+05
LITTLE ITALY	280.019705	5.945024e+06	40.719324	-73.997305	1.501599e+06
LOWER EAST SIDE	416.199264	3.293915e+06	40.717807	-73.980890	7.296687e+05
MANHATTAN VALLEY	472.416203	1.585142e+06	40.797307	-73.964286	1.921495e+05
MORNINGSIDE HEIGHTS	105.454000	3.123606e+07	40.808000	-73.963896	5.251334e+05
MURRAY HILL	392.100063	2.838923e+06	40.748303	-73.978332	7.952290e+05
ROOSEVELT ISLAND	467.908667	9.021032e+05	40.762160	-73.949168	6.001492e+05
SOHO	5557.267662	5.842216e+06	40.722184	-74.000657	1.343316e+06
TRIBECA	169.232918	6.742400e+06	40.721522	-74.010683	2.170659e+06

A map generated to visualize my neighborhoods



Data Science methodology

I will clean a bit my df by removing some useless columns (as I explained in my presentation, I will model a price function only based on a few variables)

```
df_data_0 = df_data_0.drop(["ADDRESS","ZIP CODE","YEAR BUILT","TAX CLASS AT TIME OF SALE", "BUILDING CLASS AT TIME OF SALE"], axis=1)
df_data_0.head()
```

Out[5]:

	NEIGHBORHOOD	GROSS_SQUARE_FEET	SALE_PRICE
0	ALPHABET CITY	3.68	3,200,000
1	ALPHABET CITY	5.2	6,100,000
2	ALPHABET CITY	3.6	6,300,000
3	ALPHABET CITY	7.989	1,950,000
4	ALPHABET CITY	17.478	14,000,000

In order to compute the quantities, I first need to transform the type (once into str to remove the comma; then a second time into float to apply some basic statistical functions)

```
df_data_0['SALE_PRICE'] = df_data_0.SALE_PRICE.astype(str)
df_data_0['GROSS_SQUARE_FEET'] = df_data_0.GROSS_SQUARE_FEET.astype(str)

a = df_data_0['SALE_PRICE'].tolist()
b = df_data_0['GROSS_SQUARE_FEET'].tolist()

a = [i.replace(",","") for i in a]
b = [i.replace(",","") for i in b]

df_data_0['SALE_PRICE'] = a
df_data_0['GROSS_SQUARE_FEET'] = b

df_data_0['SALE_PRICE'] = df_data_0.SALE_PRICE.astype(float)
df_data_0['GROSS_SQUARE_FEET'] = df_data_0.GROSS_SQUARE_FEET.astype(float)

df_data_0.head()
```

Create the new dataframe and display the top 10 venues for each neighborhood.

```
manhattan_grouped = manhattan_onehot.groupby('Neighborhood').mean().reset_index()
manhattan_grouped
```

[39]:

	Neighborhood	Art Gallery	Arts & Crafts Store	Bakery	Bar	Beer Bar	Beer Store	Bookstore	Burger Joint	Chinese Restaurant	Coffee Shop	Comedy Club	Cycle Studio	Dance Studio	Deli / Bodega	Dessert Shop	Dog Run	Falafel Restaurant	Farmers Market	Filipino Restaurant	Furniture / Home Store	Gourmet Shop	Greek Restaurant	Gym	Hawaiian Restaurant	Hostel	Hotel	Italian Restaurant	Japa Resta
0	CIVIC CENTER	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.2	0.0	0.0	0.0	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
1	CLINTON	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
2	EAST VILLAGE	0.0	0.0	0.0	0.0	0.0	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
3	FLATIRON	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.2	0.0	0.0	0.2	0.0	0.0	0.0	0.0	
4	GRAMERCY	0.0	0.0	0.0	0.0	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.2	0.0	0.0	0.0	0.0	0.0	0.0	
5	INWOOD	0.0	0.0	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.2	0.0	0.0	0.0	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
6	LITTLE ITALY	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
7	LOWER EAST SIDE	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.2	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
8	MANHATTAN VALLEY	0.0	0.0	0.0	0.2	0.0	0.0	0.0	0.0	0.0	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.2	0.0	0.2	
9	MORNINGSIDE HEIGHTS	0.0	0.0	0.0	0.0	0.0	0.0	0.2	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
10	MURRAY HILL	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.2	0.0	0.2	0.0	
11	ROOSEVELT ISLAND	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.2	0.0	0.0	0.0	0.2	0.0	0.0	0.0	0.0	0.0	
12	SOHO	0.0	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.2	0.0	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
13	TRIBECA	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.2	0.2	

→

Data Cleaning

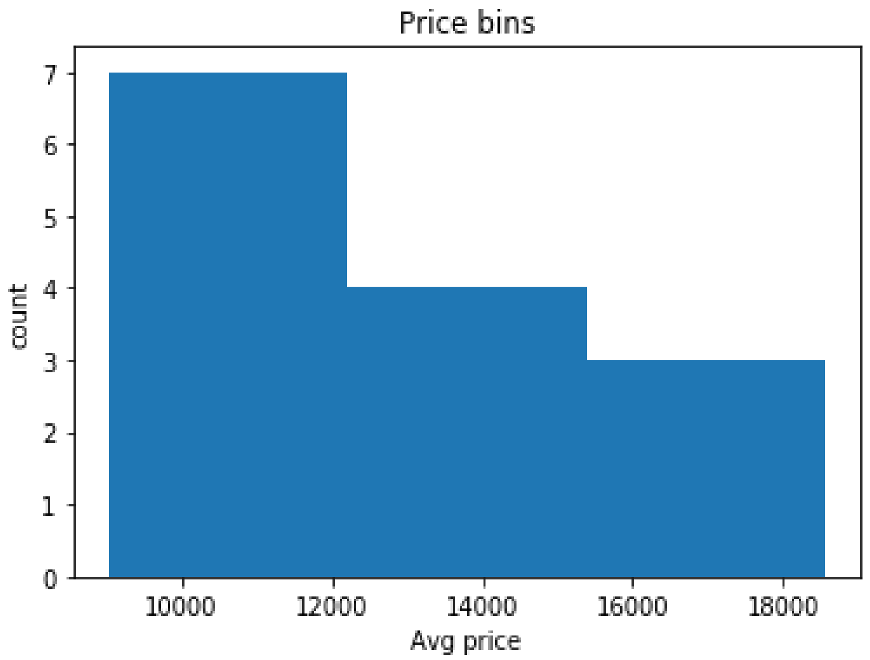
Data cleansing or data cleaning is the process of detecting and correcting (or removing) corrupt or inaccurate records from a record set, table, or database and refers to identifying incomplete, incorrect, inaccurate or irrelevant parts of the data and then replacing, modifying, or deleting the dirty or coarse data

→

Panda Library

In Python, we usually do this by dividing the sum of given numbers with the count of the number present. Python mean function can be used to calculate the mean/average of the given list of numbers. It returns the mean of the data set passed as parameters.

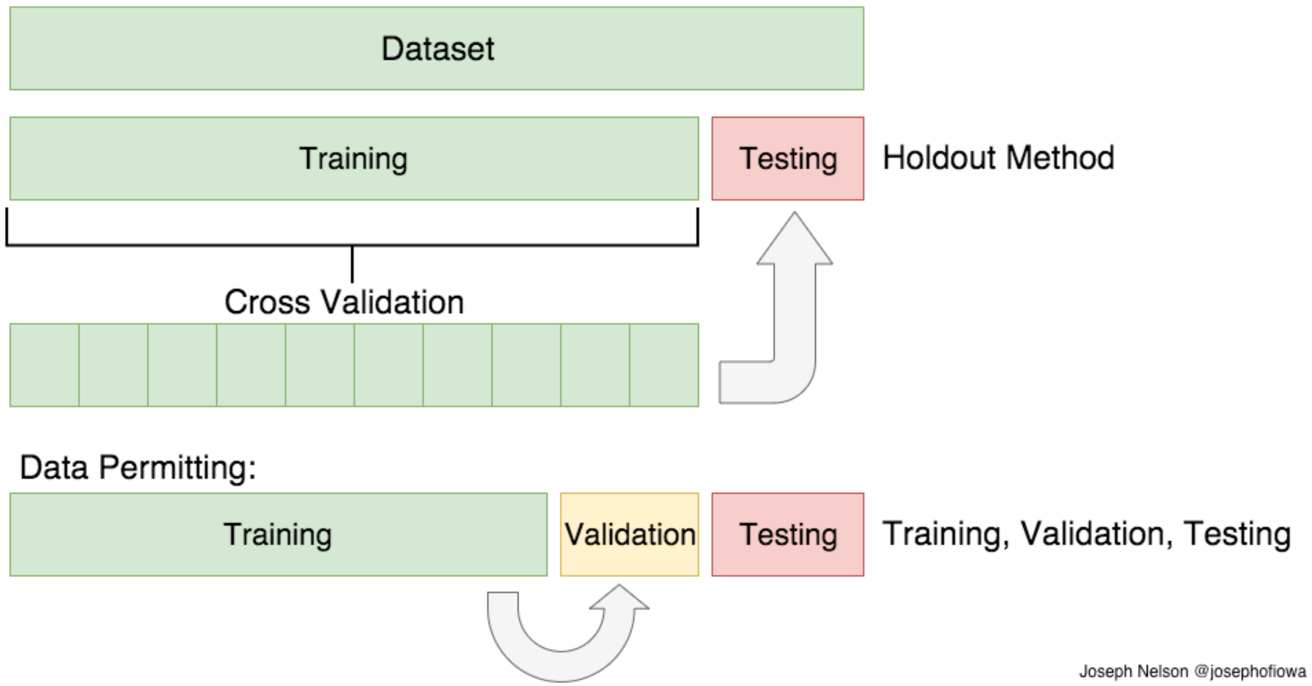
In the graph below we can see that I cut my continuous target data (avg price/m2) into 3 discrete labels: **Average Price**, **Above Average** and **High Price**



Binning

Data binning, which is also known as bucketing or discretization, is a technique used in data processing and statistics. Binning can be used for example, if there are more possible data points than observed data points.

	Price	Price-Categories
0	14256	Above Average
1	18567	High level
2	9657	Average level
3	10456	Average level
4	10250	Average level



Train-Test Split

Creation of a train and test dataset Train/Test Split involves splitting the dataset into training and testing sets respectively, which are mutually exclusive. After which, you train with the training set and test with the testing set.

This will provide a more accurate evaluation on out-of-sample accuracy because the testing dataset is not part of the dataset that have been used to train the data. It is more realistic for real world problems.

This means that we know the outcome of each data point in this dataset, making it great to test with! And since this data has not been used to train the model, the model has no knowledge of the outcome of these data points. So, in essence, it's truly an out-of-sample testing.

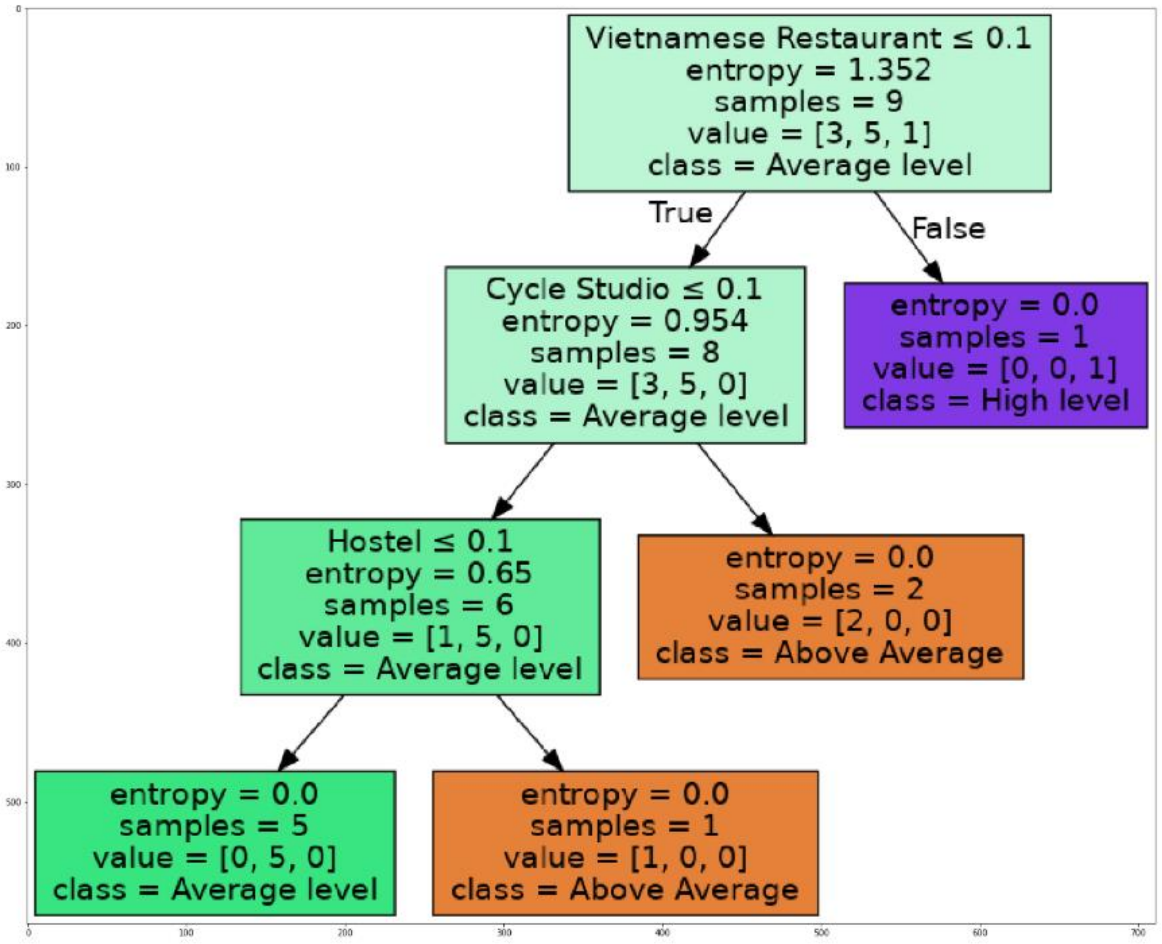
.

Modeling
results

Accuracy Test

```python
from sklearn import metrics
import matplotlib.pyplot as plt
print("DecisionTrees's Accuracy: ", metrics.accuracy_score(y_testset, predTree))
```

```
DecisionTrees's Accuracy:  0.6
```



Decision tree

A decision tree is a classification and prediction tool having a tree like structure, where each internal node denotes a test on an attribute, each branch represents an outcome of the test, and each leaf node (terminal node) holds a class label.

.