In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [44]:

```
df = pd.read_csv('/home/joseph/Desktop/ml lab/lab1/dataset/Melbourne_housing_FULL.c
df.head(20)
```

Out[44]:

	Suburb	Address	Rooms	Type	Price	Method	SellerG	Date	Distance	Pos
0	Abbotsford	68 Studley St	2	h	NaN	SS	Jellis	3/09/2016	2.5	
1	Abbotsford	85 Turner St	2	h	1480000.0	S	Biggin	3/12/2016	2.5	
2	Abbotsford	25 Bloomburg St	2	h	1035000.0	S	Biggin	4/02/2016	2.5	
3	Abbotsford	18/659 Victoria St	3	u	NaN	VB	Rounds	4/02/2016	2.5	
4	Abbotsford	5 Charles St	3	h	1465000.0	SP	Biggin	4/03/2017	2.5	
5	Abbotsford	40 Federation La	3	h	850000.0	PI	Biggin	4/03/2017	2.5	
6	Abbotsford	55a Park St	4	h	1600000.0	VB	Nelson	4/06/2016	2.5	
7	Abbotsford	16 Maugie St	4	h	NaN	SN	Nelson	6/08/2016	2.5	
8	Abbotsford	53 Turner St	2	h	NaN	S	Biggin	6/08/2016	2.5	
9	Abbotsford	99 Turner St	2	h	NaN	S	Collins	6/08/2016	2.5	
10	Abbotsford	129 Charles St	2	h	941000.0	S	Jellis	7/05/2016	2.5	
11	Abbotsford	124 Yarra St	3	h	1876000.0	S	Nelson	7/05/2016	2.5	
12	Abbotsford	121/56 Nicholson St	2	u	NaN	PI	Biggin	7/11/2016	2.5	
13	Abbotsford	17 Raphael St	4	h	NaN	W	Biggin	7/11/2016	2.5	
14	Abbotsford	98 Charles St	2	h	1636000.0	S	Nelson	8/10/2016	2.5	
15	Abbotsford	217 Langridge St	3	h	1000000.0	S	Jellis	8/10/2016	2.5	
16	Abbotsford	18a Mollison St	2	t	745000.0	S	Jellis	8/10/2016	2.5	
17	Abbotsford	6/241 Nicholson St	1	u	300000.0	S	Biggin	8/10/2016	2.5	
18	Abbotsford	10 Valiant St	2	h	1097000.0	S	Biggin	8/10/2016	2.5	
19	Abbotsford	403/609 Victoria St	2	u	542000.0	S	Dingle	8/10/2016	2.5	

20 rows × 21 columns

Finding Unique Values

In [45]:

```
uniqueCounts = df.nunique();
print("Unique count across columns:")
print(uniqueCounts);
```

```
Unique count across columns:
Suburb             351
Address          34009
Rooms               12
Type                 3
Price             2871
Method               9
SellerG            388
Date                78
Distance           215
Postcode           211
Bedroom2            15
Bathroom            11
Car                 15
Landsize          1684
BuildingArea       740
YearBuilt          160
CouncilArea         33
Lattitude        13402
Longtitude       14524
Regionname           8
Propertycount      342
dtype: int64
```

Finding total number of null values

In [46]:

```
df.isnull().sum()
```

Out[46]:

```
Suburb              0
Address             0
Rooms               0
Type                0
Price            7610
Method              0
SellerG             0
Date                0
Distance            1
Postcode            1
Bedroom2         8217
Bathroom         8226
Car              8728
Landsize        11810
BuildingArea    21115
YearBuilt       19306
CouncilArea         3
Lattitude        7976
Longtitude       7976
Regionname          3
Propertycount       3
dtype: int64
```

Handling missing values using mean

In [47]:

```
df['Price'].fillna(value = df.Price.mean(), inplace = True)
df['Distance'].fillna(value = df.Distance.mean(), inplace = True)
df['Postcode'].fillna(value = df.Postcode.mean(), inplace = True)
df['Bedroom2'].fillna(value = df.Bedroom2.mean(), inplace = True)
df['Bathroom'].fillna(value = df.Bathroom.mean(), inplace = True)
df['Car'].fillna(value = df.Car.mean(), inplace = True)
df['Landsize'].fillna(value = df.Landsize.mean(), inplace = True)
df['Bedroom2'].fillna(value = df.Bedroom2.mean(), inplace = True)
df['YearBuilt'].fillna(value = df.YearBuilt.mean(), inplace = True)
df['Lattitude'].fillna(value = df.Lattitude.mean(), inplace = True)
df['Longtitude'].fillna(value = df.Longtitude.mean(), inplace = True)
df['Propertycount'].fillna(value = df.Propertycount.mean(), inplace = True)
df['BuildingArea'].fillna(value = df.BuildingArea.mean(), inplace = True)
```

In [48]:

```
df.isnull().sum()
```

Out[48]:

```
Suburb           0
Address          0
Rooms            0
Type             0
Price            0
Method           0
SellerG          0
Date             0
Distance         0
Postcode         0
Bedroom2         0
Bathroom         0
Car              0
Landsize         0
BuildingArea     0
YearBuilt        0
CouncilArea      3
Lattitude        0
Longtitude       0
Regionname       3
Propertycount    0
dtype: int64
```

In [49]:

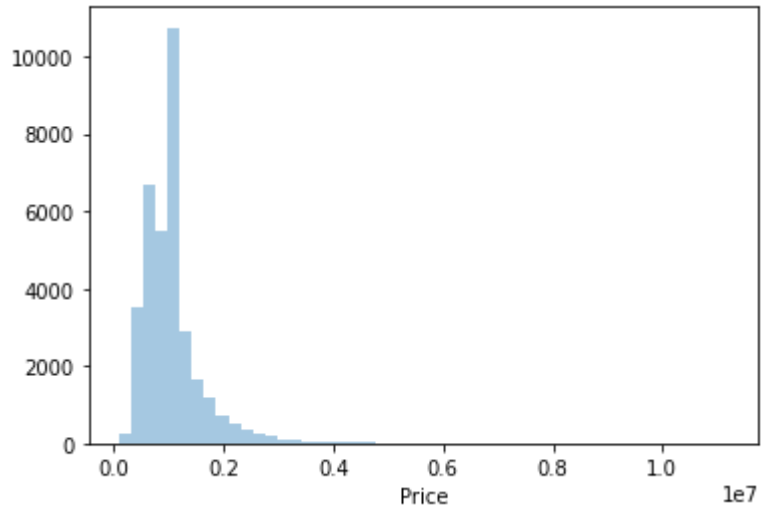
```
df = df.fillna(0)
```

Price before Scaling - Right skewed

In [50]:

```
import seaborn as sb
from matplotlib import pyplot as plt
sb.distplot(df['Price'],kde = False)
plt.show()
```

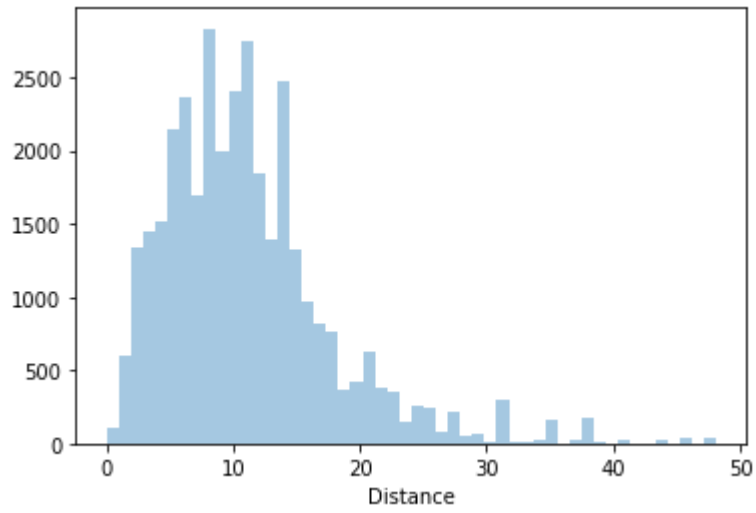
```
/home/joseph/Desktop/ml lab/lab1/mlenv/lib/python3.10/site-packages/se
aborn/distributions.py:2619: FutureWarning: `distplot` is a deprecated
function and will be removed in a future version. Please adapt your co
de to use either `displot` (a figure-level function with similar flexi
bility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```



Distance before Scaling - Right skewed

In [51]:

```
import seaborn as sb
from matplotlib import pyplot as plt
sb.distplot(df['Distance'],kde = False)
plt.show()
```



Standard Scaler

In [52]:

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
```

In [53]:

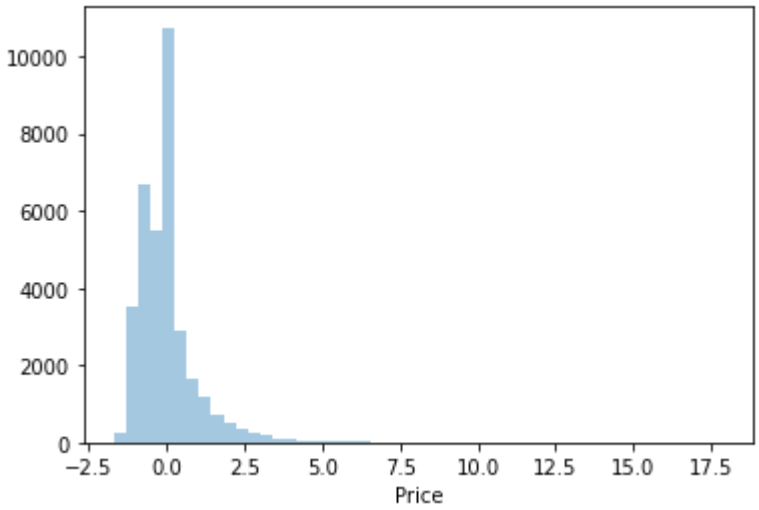
```
df[['Price','Distance','Landsize','Propertycount']] = scaler.fit_transform(df[['Pri
```

After Scaling

In [54]:

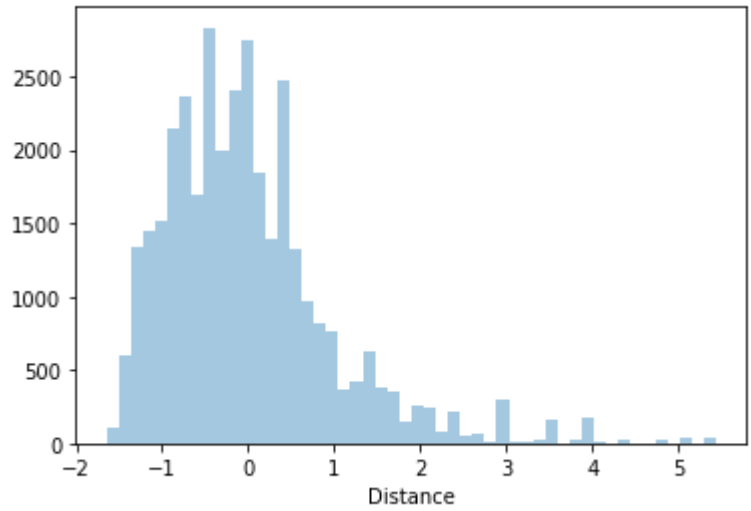
```
import seaborn as sb
from matplotlib import pyplot as plt
sb.distplot(df['Price'],kde = False)
plt.show()
```

```
/home/joseph/Desktop/ml lab/lab1/mlenv/lib/python3.10/site-packages/se
aborn/distributions.py:2619: FutureWarning: `distplot` is a deprecated
function and will be removed in a future version. Please adapt your co
de to use either `displot` (a figure-level function with similar flexi
bility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```



In [55]:

```
import seaborn as sb
from matplotlib import pyplot as plt
sb.distplot(df['Distance'],kde = False)
plt.show()
```



In [56]:

```
df.head(10)
```

Out[56]:

	Suburb	Address	Rooms	Type	Price	Method	SellerG	Date	Distance	Post
0	Abbotsford	68 Studley St	2	h	0.000000	SS	Jellis	3/09/2016	-1.279322	3
1	Abbotsford	85 Turner St	2	h	0.757901	S	Biggin	3/12/2016	-1.279322	3
2	Abbotsford	25 Bloomburg St	2	h	-0.026755	S	Biggin	4/02/2016	-1.279322	3
3	Abbotsford	18/659 Victoria St	3	u	0.000000	VB	Rounds	4/02/2016	-1.279322	3
4	Abbotsford	5 Charles St	3	h	0.731452	SP	Biggin	4/03/2017	-1.279322	3
5	Abbotsford	40 Federation La	3	h	-0.352960	PI	Biggin	4/03/2017	-1.279322	3
6	Abbotsford	55a Park St	4	h	0.969494	VB	Nelson	4/06/2016	-1.279322	3
7	Abbotsford	16 Maugie St	4	h	0.000000	SN	Nelson	6/08/2016	-1.279322	3
8	Abbotsford	53 Turner St	2	h	0.000000	S	Biggin	6/08/2016	-1.279322	3
9	Abbotsford	99 Turner St	2	h	0.000000	S	Collins	6/08/2016	-1.279322	3

10 rows × 21 columns

Hierarchical Clustering

In [57]:

```
from sklearn import preprocessing

label_encoder = preprocessing.LabelEncoder()
df['Type']= label_encoder.fit_transform(df['Type'])
```

In [58]:

```
df
```

Out[58]:

	Suburb	Address	Rooms	Type	Price	Method	SellerG	Date	Dis
0	Abbotsford	68 Studley St	2	0	0.000000	SS	Jellis	3/09/2016	-1.2
1	Abbotsford	85 Turner St	2	0	0.757901	S	Biggin	3/12/2016	-1.2
2	Abbotsford	25 Bloomburg St	2	0	-0.026755	S	Biggin	4/02/2016	-1.2
3	Abbotsford	18/659 Victoria St	3	2	0.000000	VB	Rounds	4/02/2016	-1.2
4	Abbotsford	5 Charles St	3	0	0.731452	SP	Biggin	4/03/2017	-1.2
...	...	...	...	...	...	...	...	...	
34852	Yarraville	13 Burns St	4	0	0.757901	PI	Jas	24/02/2018	-0.7
34853	Yarraville	29A Murray St	2	0	-0.285956	SP	Sweeney	24/02/2018	-0.7
34854	Yarraville	147A Severn St	2	1	-0.608634	S	Jas	24/02/2018	-0.7
34855	Yarraville	12/37 Stephen St	3	0	0.158389	SP	hockingstuart	24/02/2018	-0.7
34856	Yarraville	3 Tarrengower St	2	0	-0.053204	PI	RW	24/02/2018	-0.7

34857 rows × 21 columns

In [59]:

```
df = df.drop(['BuildingArea','YearBuilt','Bedroom2','Address','Postcode'], axis = 1
```

In [60]:

```
[df.Lattitude.isnull(), 'Lattitude']  = df.groupby('Suburb')['Lattitude'].transform(
[df.Longtitude.isnull(), 'Longtitude']  = df.groupby('Suburb')['Longtitude'].transfo
```

In [61]:

```
df.drop(['Suburb','SellerG'], axis=1,inplace=True)
```

In [62]:

```
df= pd.get_dummies(df, columns = ['Type','Method','CouncilArea','Regionname'])
```

In [63]:

```
df.drop(['Date'], axis=1,inplace=True)
```

In [64]:

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler
from scipy import stats
from sklearn.cluster import KMeans
from sklearn import metrics
from sklearn.preprocessing import scale
from sklearn.model_selection import train_test_split
from sklearn.linear_model import Ridge, RidgeCV, Lasso, LassoCV
from sklearn.metrics import mean_squared_error
%matplotlib inline
```

In [65]:

```
df_c=df
df_c.drop(['Price'], axis=1,inplace=True)
```

In [66]:

```
df_c.head()
```

Out[66]:

	Rooms	Distance	Bathroom	Car	Landsize	Lattitude	Longtitude	Propertycount	Type_0
0	2	-1.279322	1.0	1.0	-0.169196	-37.8014	144.9958	-0.802624	1
1	2	-1.279322	1.0	1.0	-0.141696	-37.7996	144.9984	-0.802624	1
2	2	-1.279322	1.0	0.0	-0.158341	-37.8079	144.9934	-0.802624	1
3	3	-1.279322	2.0	1.0	-0.214788	-37.8114	145.0116	-0.802624	0
4	3	-1.279322	2.0	0.0	-0.166301	-37.8093	144.9944	-0.802624	1

5 rows × 63 columns

In [67]:

```
#df_imp=df_c[['Rooms', 'Car']]
#df_imp=df_c[['Propertycount', 'Rooms']]

#df_imp=df_c[['Distance', 'Propertycount']]
df_imp=pd.DataFrame(df_c[['Lattitude','Longtitude']])
from sklearn import preprocessing

# x = df_imp #returns a numpy array
min_max_scaler = preprocessing.MinMaxScaler()
df_imp = min_max_scaler.fit_transform(df_imp)
```

In [41]:

```
df.head()
df = df.iloc[:1000, :]
df
```

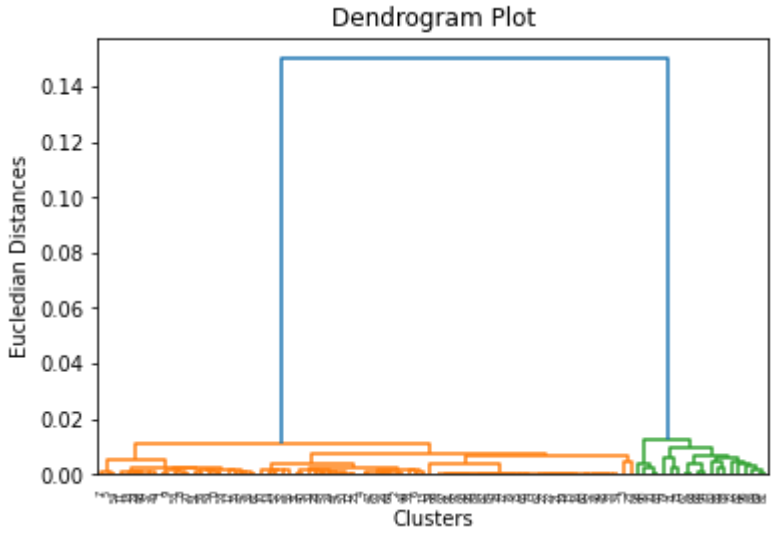
Out[41]:

	0	1
0	0.486148	0.518802
1	0.488397	0.521160
2	0.478025	0.516625
3	0.473651	0.533132
4	0.476276	0.517532
...	...	...
95	0.474608	0.524290
96	0.581870	0.404613
97	0.474608	0.524290
98	0.579121	0.420122
99	0.474608	0.524290

100 rows × 2 columns

In [42]:

```
import scipy.cluster.hierarchy as shc
dendro = shc.dendrogram(shc.linkage(df, method = "ward"))
plt.title("Dendrogram Plot")
plt.ylabel("Eucledian Distances")
plt.xlabel("Clusters")
plt.show()
```



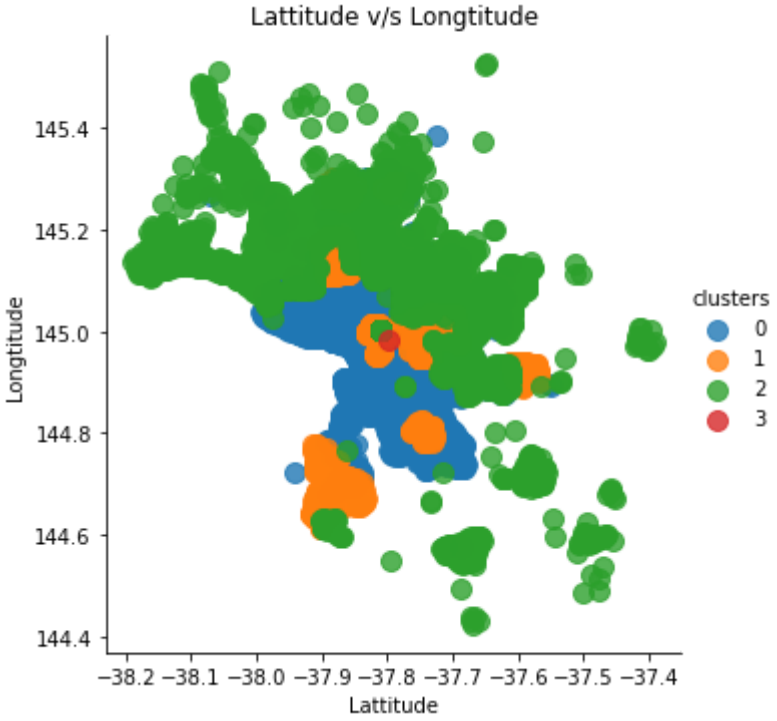
In [68]:

```
from sklearn.cluster import AgglomerativeClustering
kmeans = AgglomerativeClustering(n_clusters=4, affinity='euclidean', linkage='ward'
labels = kmeans.labels_
#Glue back to originaal data
df['clusters'] = labels
df2 = df.rename(columns = {0 : 'Lattitude', 1: 'Longtitude'})
```

In [69]:

```
sns.lmplot('Lattitude', 'Longtitude', data = df2, fit_reg=False,hue="clusters",  sc
plt.title('Lattitude v/s Longtitude')
plt.xlabel('Lattitude')
plt.ylabel('Longtitude')
plt.savefig('cluster_5.png')
plt.show()
```

```
/home/joseph/Desktop/ml lab/lab1/mlenv/lib/python3.10/site-packages/se
aborn/_decorators.py:36: FutureWarning: Pass the following variables a
s keyword args: x, y. From version 0.12, the only valid positional arg
ument will be `data`, and passing other arguments without an explicit
keyword will result in an error or misinterpretation.
  warnings.warn(
```



In [ ]: