Model which predicts house prices in Melbourne using a given data set.

Importing Pandas library

```
In [1]: import pandas as pd
```

Get the dataset and inspect it

```
In [2]: melbourne_data = pd.read_csv("Melbourne Housing Dataset/melb_data.csv")
```

```
In [3]: print(melbourne_data.head())
```

```
       Suburb           Address  Rooms Type      Price Method SellerG  \
0  Abbotsford      85 Turner St      2    h  1480000.0      S  Biggin
1  Abbotsford   25 Bloomburg St      2    h  1035000.0      S  Biggin
2  Abbotsford      5 Charles St      3    h  1465000.0     SP  Biggin
3  Abbotsford  40 Federation La      3    h   850000.0     PI  Biggin
4  Abbotsford       55a Park St      4    h  1600000.0     VB  Nelson

        Date  Distance  Postcode  ...  Bathroom  Car  Landsize  BuildingArea  \
0  3/12/2016       2.5    3067.0  ...       1.0  1.0     202.0           NaN
1  4/02/2016       2.5    3067.0  ...       1.0  0.0     156.0          79.0
2  4/03/2017       2.5    3067.0  ...       2.0  0.0     134.0         150.0
3  4/03/2017       2.5    3067.0  ...       2.0  1.0      94.0           NaN
4  4/06/2016       2.5    3067.0  ...       1.0  2.0     120.0         142.0

   YearBuilt  CouncilArea Lattitude  Longtitude             Regionname  \
0        NaN        Yarra  -37.7996    144.9984  Northern Metropolitan
1     1900.0        Yarra  -37.8079    144.9934  Northern Metropolitan
2     1900.0        Yarra  -37.8093    144.9944  Northern Metropolitan
3        NaN        Yarra  -37.7969    144.9969  Northern Metropolitan
4     2014.0        Yarra  -37.8072    144.9941  Northern Metropolitan

  Propertycount
0        4019.0
1        4019.0
2        4019.0
3        4019.0
4        4019.0

[5 rows x 21 columns]
```

```
In [4]: print(melbourne_data.describe())
```

```
              Rooms         Price      Distance      Postcode      Bedroom2  \
count  13580.000000  1.358000e+04  13580.000000  13580.000000  13580.000000   
mean       2.937997  1.075684e+06     10.137776   3105.301915      2.914728   
std        0.955748  6.393107e+05      5.868725     90.676964      0.965921   
min        1.000000  8.500000e+04      0.000000   3000.000000      0.000000   
25%        2.000000  6.500000e+05      6.100000   3044.000000      2.000000   
50%        3.000000  9.030000e+05      9.200000   3084.000000      3.000000   
75%        3.000000  1.330000e+06     13.000000   3148.000000      3.000000   
max       10.000000  9.000000e+06     48.100000   3977.000000     20.000000   

           Bathroom           Car       Landsize  BuildingArea    YearBuilt  \
count  13580.000000  13518.000000   13580.000000   7130.000000  8205.000000   
mean       1.534242      1.610075     558.416127    151.967650  1964.684217   
std        0.691712      0.962634    3990.669241    541.014538    37.273762   
min        0.000000      0.000000       0.000000      0.000000  1196.000000   
25%        1.000000      1.000000     177.000000     93.000000  1940.000000   
50%        1.000000      2.000000     440.000000    126.000000  1970.000000   
75%        2.000000      2.000000     651.000000    174.000000  1999.000000   
max        8.000000     10.000000  433014.000000  44515.000000  2018.000000   

          Lattitude    Longtitude  Propertycount  
count  13580.000000  13580.000000   13580.000000  
mean     -37.809203    144.995216    7454.417378  
std        0.079260      0.103916    4378.581772  
min      -38.182550    144.431810     249.000000  
25%      -37.856822    144.929600    4380.000000  
50%      -37.802355    145.000100    6555.000000  
75%      -37.756400    145.058305   10331.000000  
max      -37.408530    145.526350   21650.000000  
```

In [5]:

```
print(melbourne_data.columns)
```

```
Index(['Suburb', 'Address', 'Rooms', 'Type', 'Price', 'Method', 'SellerG',
       'Date', 'Distance', 'Postcode', 'Bedroom2', 'Bathroom', 'Car',
       'Landsize', 'BuildingArea', 'YearBuilt', 'CouncilArea', 'Lattitude',
       'Longtitude', 'Regionname', 'Propertycount'],
      dtype='object')
```

When examining the dataset we can see that there are some columns with missing values. So, we will just drop those corresponding rows

In [6]:

```
melbourne_data = melbourne_data.dropna(axis = 0)
```

Select prediction target which is the house prices

In [7]:

```
y = melbourne_data.Price
print(y)
```

```
1        1035000.0
2        1465000.0
4        1600000.0
6        1876000.0
7        1636000.0
           ...    
12205     601000.0
12206    1050000.0
12207     385000.0
12209     560000.0
12212    2450000.0
Name: Price, Length: 6196, dtype: float64
```

Choose features - Here we consider the columns "Rooms", "Bathroom", "Landsize", "Lattitude", "Longtitude" as features.

In [8]:

```
melbourne_features = ["Rooms", "Bathroom", "Landsize", "Lattitude", "Longtitude"]
x = melbourne_data[melbourne_features]
```

In [9]:

```
print(x.describe())
```

```
             Rooms     Bathroom      Landsize    Lattitude   Longtitude
count  6196.000000  6196.000000   6196.000000  6196.000000  6196.000000
mean      2.931407     1.576340    471.006940   -37.807904   144.990201
std       0.971079     0.711362    897.449881     0.075850     0.099165
min       1.000000     1.000000      0.000000   -38.164920   144.542370
25%       2.000000     1.000000    152.000000   -37.855438   144.926198
50%       3.000000     1.000000    373.000000   -37.802250   144.995800
75%       4.000000     2.000000    628.000000   -37.758200   145.052700
max       8.000000     8.000000  37000.000000   -37.457090   145.526350
```

In [10]:

```
print(x.head())
```

```
   Rooms  Bathroom  Landsize  Lattitude  Longtitude
1      2       1.0     156.0   -37.8079    144.9934
2      3       2.0     134.0   -37.8093    144.9944
4      4       1.0     120.0   -37.8072    144.9941
6      3       2.0     245.0   -37.8024    144.9993
7      2       1.0     256.0   -37.8060    144.9954
```

Building the Model

In [11]:

```
# import scikit-learn library
from sklearn.tree import DecisionTreeRegressor
```

In [12]:

```
melbourne_model = DecisionTreeRegressor(random_state = 1)
```

In [13]:

```
# Fit the model
melbourne_model.fit(x, y)
```

Out[13]:

```
DecisionTreeRegressor(random_state=1)
```

Make predictions using the model

In [14]:

```
print("Making predictions for the following 5 houses.", "\n")
print(x.head())
```

```
Making predictions for the following 5 houses.

   Rooms  Bathroom  Landsize  Lattitude  Longtitude
1      2       1.0     156.0   -37.8079    144.9934
2      3       2.0     134.0   -37.8093    144.9944
4      4       1.0     120.0   -37.8072    144.9941
6      3       2.0     245.0   -37.8024    144.9993
7      2       1.0     256.0   -37.8060    144.9954
```

In [15]:

```
print("The predictions are:")
print(melbourne_model.predict(x.head()))
```

```
The predictions are:
[1035000. 1465000. 1600000. 1876000. 1636000.]
```

More check for accuracy of the model

```
In [16]: print(y.head())
```

```
1    1035000.0
2    1465000.0
4    1600000.0
6    1876000.0
7    1636000.0
Name: Price, dtype: float64
```

```
In [17]: print(melbourne_model.predict(x.head()))
```

```
[1035000. 1465000. 1600000. 1876000. 1636000.]
```

```
In [ ]:
```