Data Wrangling, I Perform the following operations using Python on any open-source dataset (e.g., data.csv)

1. Import all the required Python Libraries.
2. ocate an open-source data from L the web (e.g. https://www.kaggle.com (https://www.kaggle.com)). Provide a clear description of the data and its source (i.e., URL of the web site).
3. Load the Dataset into pandas' data frame.
4. Data Preprocessing: check for missing values in the data using pandas insult (), describe() function to get some initial statistics. Provide variable descriptions. Types of variables etc. Check the dimensions of the data frame.
5. Data Formatting and Data Normalization: Summarize the types of variables by checking the data types (i.e., character, numeric, integer, factor, and logical) of the variables in the data set. If variables are not in the correct data type, apply proper type conversions.
6. Turn categorical variables into quantitative variables in Python. In addition to the codes and outputs, explain every operation that you do in the above steps and explain everything that you do to import/read/scrape the data set.

In [1]:

```
import pandas as pd
import numpy as np
```

In [2]:

```
data = pd.read_csv("melb_data.csv")
```

In [3]:

```
data.describe()
```

Out[3]:

	Rooms	Price	Distance	Postcode	Bedroom2	Bathroom	
count	13580.000000	1.358000e+04	13580.000000	13580.000000	13580.000000	13580.000000	13
mean	2.937997	1.075684e+06	10.137776	3105.301915	2.914728	1.534242	
std	0.955748	6.393107e+05	5.868725	90.676964	0.965921	0.691712	
min	1.000000	8.500000e+04	0.000000	3000.000000	0.000000	0.000000	
25%	2.000000	6.500000e+05	6.100000	3044.000000	2.000000	1.000000	
50%	3.000000	9.030000e+05	9.200000	3084.000000	3.000000	1.000000	
75%	3.000000	1.330000e+06	13.000000	3148.000000	3.000000	2.000000	
max	10.000000	9.000000e+06	48.100000	3977.000000	20.000000	8.000000	

In [4]:

```
data.head()
```

Out[4]:

	Suburb	Address	Rooms	Type	Price	Method	SellerG	Date	Distance	Postc
0	Abbotsford	85 Turner St	2	h	1480000.0	S	Biggin	3/12/2016	2.5	30
1	Abbotsford	25 Bloomburg St	2	h	1035000.0	S	Biggin	4/02/2016	2.5	30
2	Abbotsford	5 Charles St	3	h	1465000.0	SP	Biggin	4/03/2017	2.5	30
3	Abbotsford	40 Federation La	3	h	850000.0	PI	Biggin	4/03/2017	2.5	30
4	Abbotsford	55a Park St	4	h	1600000.0	VB	Nelson	4/06/2016	2.5	30

5 rows × 21 columns

In [5]:

```
data.isnull()
```

Out[5]:

	Suburb	Address	Rooms	Type	Price	Method	SellerG	Date	Distance	Postcode	...
0	False	False	False	False	False	False	False	False	False	False	...
1	False	False	False	False	False	False	False	False	False	False	...
2	False	False	False	False	False	False	False	False	False	False	...
3	False	False	False	False	False	False	False	False	False	False	...
4	False	False	False	False	False	False	False	False	False	False	...
...	...	...	...	...	...	...	...	...	...	...	...
13575	False	False	False	False	False	False	False	False	False	False	...
13576	False	False	False	False	False	False	False	False	False	False	...
13577	False	False	False	False	False	False	False	False	False	False	...
13578	False	False	False	False	False	False	False	False	False	False	...
13579	False	False	False	False	False	False	False	False	False	False	...

13580 rows × 21 columns

In [6]:

```
data.isnull().sum()
```

Out[6]:

```
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                62
Landsize            0
BuildingArea     6450
YearBuilt        5375
CouncilArea      1369
Lattitude           0
Longtitude          0
Regionname          0
Propertycount       0
dtype: int64
```

In [7]:

```
data.shape
```

Out[7]:

```
(13580, 21)
```

In [8]:

```
data.dtypes
```

Out[8]:

```
Suburb           object
Address          object
Rooms             int64
Type             object
Price           float64
Method           object
SellerG          object
Date             object
Distance        float64
Postcode        float64
Bedroom2        float64
Bathroom        float64
Car             float64
Landsize        float64
BuildingArea    float64
YearBuilt       float64
CouncilArea      object
Lattitude       float64
Longtitude      float64
Regionname       object
Propertycount   float64
dtype: object
```

In [9]:

```
data = data[data['YearBuilt'].notna()]
```

In [12]:

```
data['BuildingArea']=data['BuildingArea'].fillna(data['BuildingArea'].mean())

data['Car'].fillna(data.Car.mode()[0], inplace=True)

data['YearBuilt']=data['YearBuilt'].fillna(data['YearBuilt'].median())

data['CouncilArea'].fillna(value="new type", inplace=True)
```

In [13]:

```
data["YearBuilt"]= data["YearBuilt"].astype(int)
data['Date'] = data["Date"].astype("datetime64")
data["Postcode"]= data["Postcode"].astype('int64')
data["Bedroom2"]= data["Bedroom2"].astype('int64')
data["Bathroom"]= data["Bathroom"].astype('int64')
data["Car"]= data["Car"].astype('int64')
```

In [14]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 8205 entries, 1 to 13579
Data columns (total 21 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Suburb         8205 non-null   object
 1   Address        8205 non-null   object
 2   Rooms          8205 non-null   int64
 3   Type           8205 non-null   object
 4   Price          8205 non-null   float64
 5   Method         8205 non-null   object
 6   SellerG        8205 non-null   object
 7   Date           8205 non-null   datetime64[ns]
 8   Distance       8205 non-null   float64
 9   Postcode       8205 non-null   int64
 10  Bedroom2       8205 non-null   int64
 11  Bathroom       8205 non-null   int64
 12  Car            8205 non-null   int64
 13  Landsize       8205 non-null   float64
 14  BuildingArea   8205 non-null   float64
 15  YearBuilt      8205 non-null   int32
 16  CouncilArea    8205 non-null   object
 17  Lattitude      8205 non-null   float64
 18  Longtitude     8205 non-null   float64
 19  Regionname     8205 non-null   object
 20  Propertycount  8205 non-null   float64
dtypes: datetime64[ns](1), float64(7), int32(1), int64(5), object(7)
memory usage: 1.3+ MB
```

In [15]:

```
data.isnull().sum()
```

Out[15]:

```
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      0
Lattitude        0
Longtitude       0
Regionname       0
Propertycount    0
dtype: int64
```

In [16]:

```
from sklearn.preprocessing import LabelEncoder
```

In [17]:

```
encoder = LabelEncoder()
data['Type'] = encoder.fit_transform(data['Type'])
```

In [19]:

```
data.Method.unique()
```

Out[19]:

```
array(['S', 'SP', 'VB', 'PI', 'SA'], dtype=object)
```

In [21]:

```
data['Method'] = encoder.fit_transform(data['Method'])
```

In [23]:

```
data.SellerG.unique()
```

Out[23]:

```
array(['Biggin', 'Nelson', 'Jellis', 'LITTLE', 'Kay', 'Beller', 'Collins',
       'Marshall', 'Brad', 'Maddison', 'Barry', 'Rendina', 'Harcourts',
       'hockingstuart', 'Buxton', 'Greg', 'RT', 'Cayzer', 'Brace',
       'Miles', 'Love', 'McGrath', 'Barlow', 'Village', 'Sweeney',
       'Burnham', 'Williams', 'Compton', 'FN', 'Jas', 'Raine&Horne',
       'Hunter', 'Hodges', 'Ray', 'Woodards', 'Raine', 'Walshe',
       'Alexkarbon', 'McDonald', 'Stockdale', 'Fletchers', 'Noel', 'Tim',
       'Purplebricks', 'Moonee', 'Edward', 'Gary', 'Chisholm', 'Philip',
       'RW', 'Ascend', 'Christopher', 'Mandy', 'Fletchers/One', 'Assisi',
       'One', 'Bayside', 'C21', 'First', 'Matthew', 'Nick', 'Lindellas',
       'Allens', 'Bells', 'Trimson', 'YPA', 'GL', "Tiernan's", 'J', 'HAR',
       'Dingle', 'Chambers', 'Peter', 'Grantham',
       'hockingstuart/Advantage', 'Gunn&Co', "O'Donoghues", 'Ross',
       'Weast', 'Century', 'Kelly', 'Property', 'Thomson',
       "Private/Tiernan's", 'Australian', 'Anderson', 'Rodney',
       "Abercromby's", 'Castran', 'Bekdon', 'Harrington', 'iTRAK',
       'Nicholson', 'Re', 'RE', 'Parkes', 'Vic', 'Holland', 'Scott',
       'Pride', 'Owen', 'Morleys', 'Wilson', 'Buxton/Advantage', 'Frank',
       'Pagan', 'Paul', 'Red', 'Caine', 'Naison', 'Jason', 'Eview',
       'Melbourne', "D'Aprano", 'Wood', 'Haughton', 'William',
       'Buckingham', 'Domain', 'Nardella', 'Walsh', 'Sweeney/Advantage',
       'Direct', 'Besser', 'Johnston', 'Redina', 'Clairmont', 'Galldon',
       'MICM', "O'Brien", 'Buxton/Find', 'W.B.', 'New', 'Considine',
       "Sotheby's", 'Geoff', 'Darren', 'Whiting', 'Morrison', 'VICPROP',
       'Charlton', 'Douglas', 'Prof.', 'Homes', 'Zahn', 'Mason', 'Dixon',
       'Luxe', 'Prowse', 'Ken', 'iOne', 'hockingstuart/Village', 'JMRE',
       'Crane', 'ASL', 'Oak', 'Reed', 'Oriental', 'Rosin', 'Hooper',
       'R&H', 'Hall', 'Ham', 'WHITEFOX', 'buyMyplace', 'LJ', 'Hoskins',
       'Iconek', 'PRDNationwide', 'Only', 'Obrien', 'Reliance', 'Lucas',
       'Millership', 'iSell', 'Rounds', 'Appleby', '@Realty', 'Jim',
       'Max', 'Real', 'iProperty', 'Triwest', 'Hayeswinckle', 'Schroeder',
       'Del', 'VICProp', 'REMAX', 'Victory', 'Smart', 'Mindacom', 'Ryder',
       'Carter', 'S&L', 'Weda', 'U', 'Win', 'Leyton', 'Prime', 'Veitch',
       'Peake', 'Sell', 'Ristic', 'Ash', 'Upper', 'TRUE', 'Leading',
       'Bullen', 'Aquire', 'Westside', 'Gardiner', 'Langwell', 'Kaye',
       'Bowman', 'Weston', 'Leeburn', 'McLennan', 'McNaughton', 'Daniel',
       'The', 'Follett', 'LLC', 'Garvey', 'Joseph', 'Luxton', 'SN',
       'Rexhepi', 'Point'], dtype=object)
```

In [25]:

```
data['SellerG']=encoder.fit_transform(data['SellerG'])
```

In [26]:

```
data['Regionname'] = encoder.fit_transform(data['Regionname'])
```

In [27]:

```
data['CouncilArea'] = encoder.fit_transform(data['CouncilArea'])
```

In [28]:

```
data.head()
```

Out[28]:

	Suburb	Address	Rooms	Type	Price	Method	SellerG	Date	Distance	Postcode
1	Abbotsford	25 Bloomburg St	2	0	1035000.0	1	19	2016-04-02	2.5	3067
2	Abbotsford	5 Charles St	3	0	1465000.0	3	19	2017-04-03	2.5	3067
4	Abbotsford	55a Park St	4	0	1600000.0	4	128	2016-04-06	2.5	3067
6	Abbotsford	124 Yarra St	3	0	1876000.0	1	128	2016-07-05	2.5	3067
7	Abbotsford	98 Charles St	2	0	1636000.0	1	128	2016-08-10	2.5	3067

5 rows × 21 columns