MSA-ASSIGNMENT-PHASE1 notebook file - Yatai Tian

July 8, 2020

[1]:
```python
import json
import sys
import time
sys.path.append('/home/nbuser/library/')

import pandas as pd
import requests
import matplotlib.pyplot as plt
import seaborn as sns

df = pd.read_csv('Dataset for Assignment.csv')
```

```
First, we read in the data set into our work space. Let's check the first few rows␣
↪and see what we're dealing with.
```

[2]:
```python
df.head()
```

[2]:
```
   Bedrooms  Bathrooms                                     Address Land area  \
0         5        3.0  106 Lawrence Crescent Hill Park, Auckland       714
1         5        3.0             8 Corsica Way Karaka, Auckland       564
2         6        4.0      243 Harbourside Drive Karaka, Auckland       626
3         2        1.0  2/30 Hardington Street Onehunga, Auckland        65
4         3        1.0      59 Israel Avenue Clover Park, Auckland       601

        CV   Latitude   Longitude      SA1  0-19 years  20-29 years  \
0   960000 -37.012920  174.904069  7009770          48           27
1  1250000 -37.063672  174.922912  7009991          42           18
2  1250000 -37.063580  174.924044  7009991          42           18
3   740000 -36.912996  174.787425  7007871          42            6
4   630000 -36.979037  174.892612  7008902          93           27

   30-39 years  40-49 years  50-59 years  60+ years      Suburbs
0           24           21           24         21     Manurewa
1           12           21           15         30       Karaka
2           12           21           15         30       Karaka
3           21           21           12         15     Onehunga
4           33           30           21         33  Clover Park
```

```
This is our function for API calling and we store this into our data frame by␣
↪adding a column called 'Population'.
```

[3]:
```python
def population(lat, lon):
    url = 'https://koordinates.com/services/query/v1/vector.json'
    params = {
        'key': '4a7d61ba2b634a08a297cd2a9f5d582d',
        'layer': '104612',
        'x': lon,
        'y': lat
    }
    response = requests.get(url, params = params)

    pop = response.
↪json()['vectorQuery']['layers']['104612']['features'][0]['properties']['C18_CURPop']
    return pop
```

[4]:
```python
df['Population'] = df.apply(lambda x: population(x['Latitude'], x['Longitude']),␣
↪axis = 1)
```

```
We now get the depreciation index for every SA1 location and merge it with our␣
↪data frame.
```

[5]:
```python
depriv_df = pd.read_excel('otago730395.xlsx')
```

[6]:
```python
merge_df = pd.merge(df, depriv_df[['SA12018_code', 'NZDep2018']], left_on =␣
↪'SA1', right_on = 'SA12018_code')
```

[7]:
```python
merge_df.head()
```

[7]:
```
   Bedrooms  Bathrooms                                    Address Land area  \
0         5        3.0  106 Lawrence Crescent Hill Park, Auckland       714
1         5        3.0               8 Corsica Way Karaka, Auckland       564
2         6        4.0      243 Harbourside Drive Karaka, Auckland       626
3         2        1.0  2/30 Hardington Street Onehunga, Auckland        65
4         3        1.0      59 Israel Avenue Clover Park, Auckland       601

        CV   Latitude   Longitude      SA1  0-19 years  20-29 years  \
0   960000 -37.012920  174.904069  7009770          48           27
1  1250000 -37.063672  174.922912  7009991          42           18
2  1250000 -37.063580  174.924044  7009991          42           18
3   740000 -36.912996  174.787425  7007871          42            6
4   630000 -36.979037  174.892612  7008902          93           27

   30-39 years  40-49 years  50-59 years  60+ years      Suburbs  Population  \
0           24           21           24         21     Manurewa         174
1           12           21           15         30       Karaka         129
2           12           21           15         30       Karaka         129
3           21           21           12         15     Onehunga         120
```

```
4          33          30          21          33  Clover Park          231

   SA12018_code  NZDep2018
0       7009770        6.0
1       7009991        1.0
2       7009991        1.0
3       7007871        2.0
4       7008902        9.0
```

Let us check if there any null values, see the types of each variable and check the data with the describe() function.

[8]:
```
merge_df.isnull().values.any()
```

[8]:
```
True
```

[9]:
```
merge_df.dtypes
```

[9]:
```
Bedrooms           int64
Bathrooms        float64
Address           object
Land area         object
CV                 int64
Latitude         float64
Longitude        float64
SA1                int64
0-19 years         int64
20-29 years        int64
30-39 years        int64
40-49 years        int64
50-59 years        int64
60+ years          int64
Suburbs           object
Population         int64
SA12018_code       int64
NZDep2018        float64
dtype: object
```

[10]:
```
merge_df.describe()
```

[10]:
```
          Bedrooms    Bathrooms            CV     Latitude    Longitude  \
count  1051.000000  1049.000000  1.051000e+03  1051.000000  1051.000000
mean      3.777355     2.073403  1.387521e+06   -36.893715   174.799325
std       1.169412     0.992985  1.182939e+06     0.130100     0.119538
min       1.000000     1.000000  2.700000e+05   -37.265021   174.317078
25%       3.000000     1.000000  7.800000e+05   -36.950565   174.720779
50%       4.000000     2.000000  1.080000e+06   -36.893132   174.798575
75%       4.000000     3.000000  1.600000e+06   -36.855789   174.880944
max      17.000000     8.000000  1.800000e+07   -36.177655   175.492424
```

```
                SA1    0-19 years   20-29 years   30-39 years   40-49 years  \
count  1.051000e+03  1051.000000  1051.000000  1051.000000  1051.000000
mean   7.006319e+06    47.549001    28.963844    27.042816    24.125595
std    2.591262e+03    24.692205    21.037441    17.975408    10.942770
min    7.001130e+06     0.000000     0.000000     0.000000     0.000000
25%    7.004416e+06    33.000000    15.000000    15.000000    18.000000
50%    7.006325e+06    45.000000    24.000000    24.000000    24.000000
75%    7.008384e+06    57.000000    36.000000    33.000000    30.000000
max    7.011028e+06   201.000000   270.000000   177.000000   114.000000

       50-59 years   60+ years   Population  SA12018_code    NZDep2018
count  1051.000000  1051.000000  1051.000000  1.051000e+03  1051.000000
mean     22.615604    29.360609   179.914367  7.006319e+06     5.063749
std      10.210578    21.805031    71.059280  2.591262e+03     2.913471
min       0.000000     0.000000     3.000000  7.001130e+06     1.000000
25%      15.000000    18.000000   138.000000  7.004416e+06     2.000000
50%      21.000000    27.000000   174.000000  7.006325e+06     5.000000
75%      27.000000    36.000000   210.000000  7.008384e+06     8.000000
max      90.000000   483.000000   789.000000  7.011028e+06    10.000000
```

```
Since, land area was an 'object', we wanted to convert this to a float so we␣
 ↪remove the letters and characters that are not numbers and convert this to a␣
 ↪float.
```

[11]:
```
merge_df['Land area'] = merge_df['Land area'].str.extract('(\d+)').astype(float)
```

[12]:
```
merge_df.isnull().sum()
#two missing values in bathrooms and one in suburb
#since our data has over 1000 values, I have decided to drop the three rows with␣
 ↪NULL values
```

[12]:
```
Bedrooms         0
Bathrooms        2
Address          0
Land area        0
CV               0
Latitude         0
Longitude        0
SA1              0
0-19 years       0
20-29 years      0
30-39 years      0
40-49 years      0
50-59 years      0
60+ years        0
Suburbs          1
Population       0
SA12018_code     0
```

```
NZDep2018       0
dtype: int64
```

[13]:
```
house_df = merge_df.dropna()
final_df = house_df.drop('SA12018_code', axis = 1)
```

[14]:
```
final_df.isnull().sum()
```

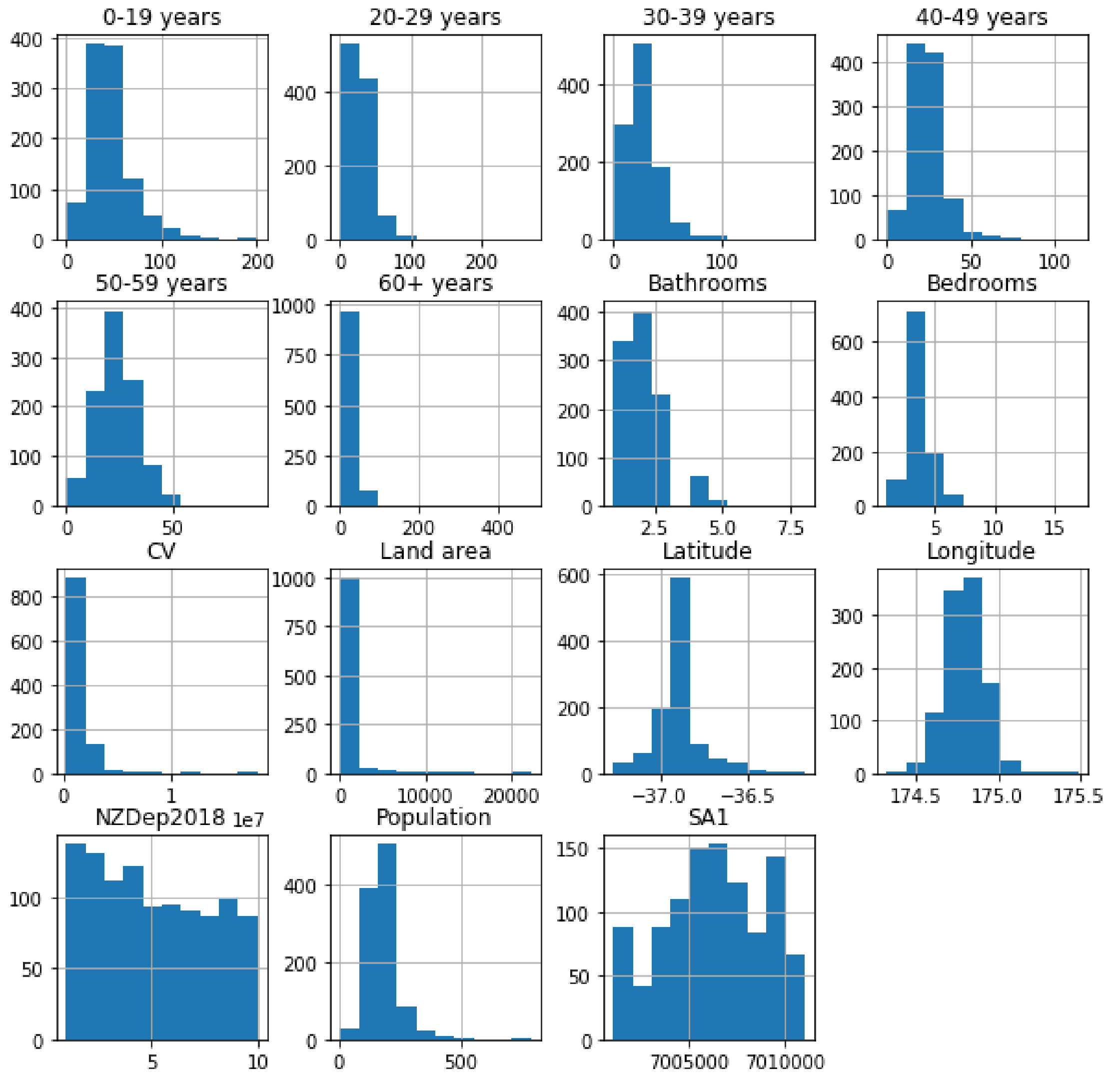
[14]:
```
Bedrooms       0
Bathrooms      0
Address        0
Land area      0
CV             0
Latitude       0
Longitude      0
SA1            0
0-19 years     0
20-29 years    0
30-39 years    0
40-49 years    0
50-59 years    0
60+ years      0
Suburbs        0
Population     0
NZDep2018      0
dtype: int64
```

Seaborn plots for every numeric variable for an idea on what we're working with. Noticeably, CV is heavily left skewed.

[15]:
```
final_df.hist(figsize=(10,10))
```

[15]:
```
array([[<matplotlib.axes._subplots.AxesSubplot object at 0x7fcadc71ee48>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7fcadb2fbf28>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7fcadb6a4390>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7fcadbb2a908>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x7fcadbbd2e80>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7fcadbeda438>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7fcadb26a9b0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7fcadba7af60>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x7fcadba7af98>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7fcadb7d8a58>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7fcadbba2fd0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7fcadb730588>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x7fcadb292b00>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7fcadbaa80b8>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7fcadaf9f630>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7fcadaab4ba8>]],
      dtype=object)
```

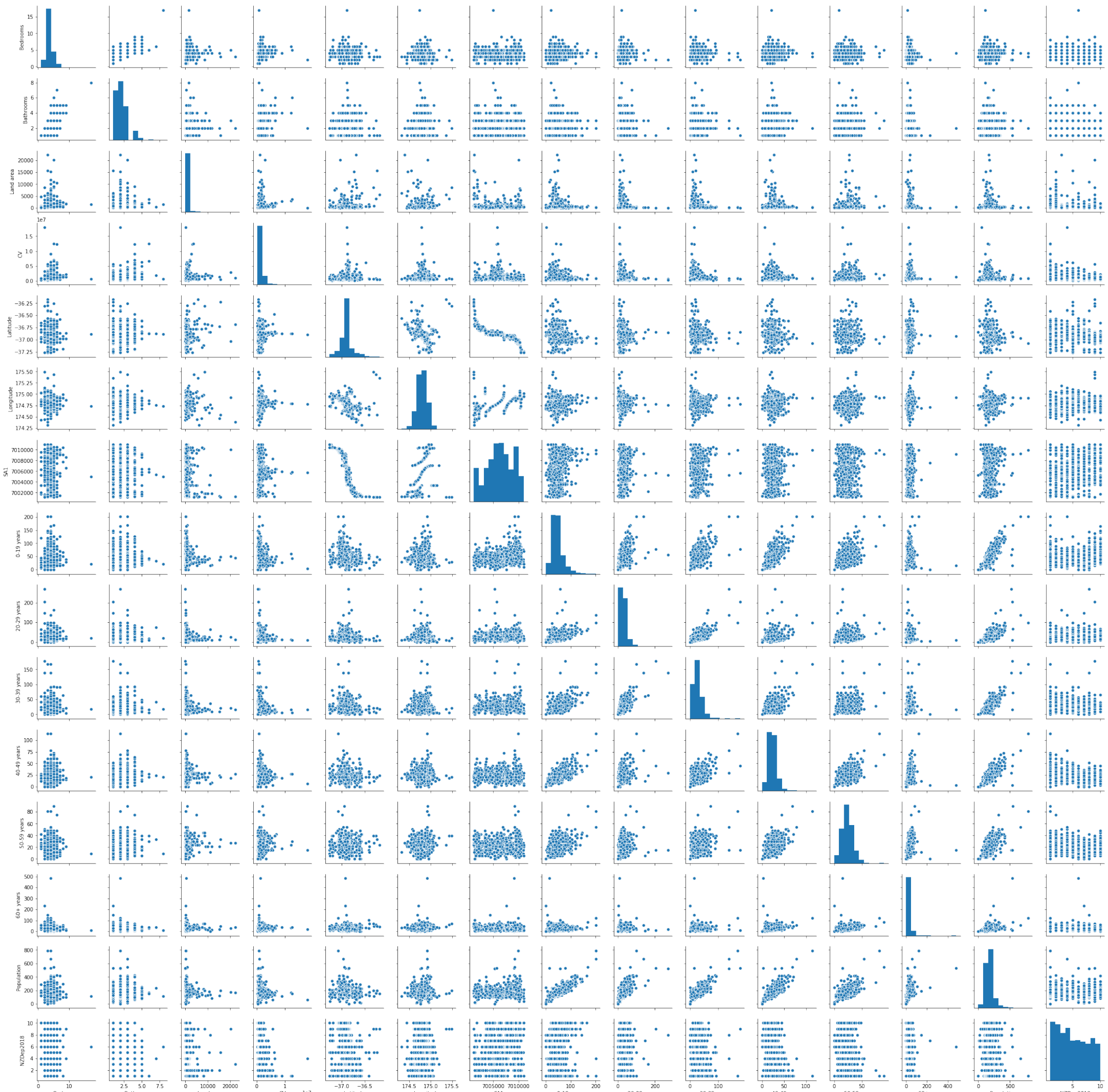
A very mess pairs plot which we will clean up in R.



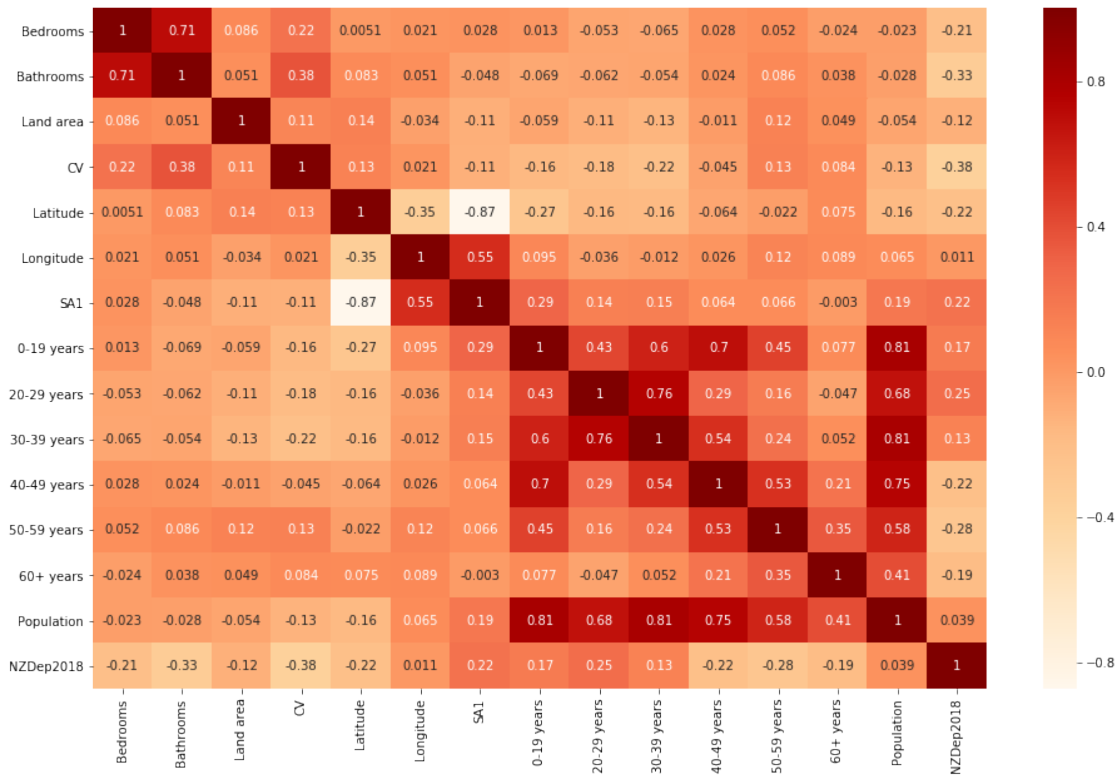
[16]:
```
sns.pairplot(final_df, height = 2.0)
```

Finally, a heat map of correlation which will be anaylsed in depth in R.

[16]: <seaborn.axisgrid.PairGrid at 0x7fcadab01198>



[17]:
```python
ax, fig = plt.subplots(figsize=(16,10))
correlation_matrix = final_df.corr()
sns.heatmap(correlation_matrix, annot = True, cmap = 'OrRd')
plt.show()
```

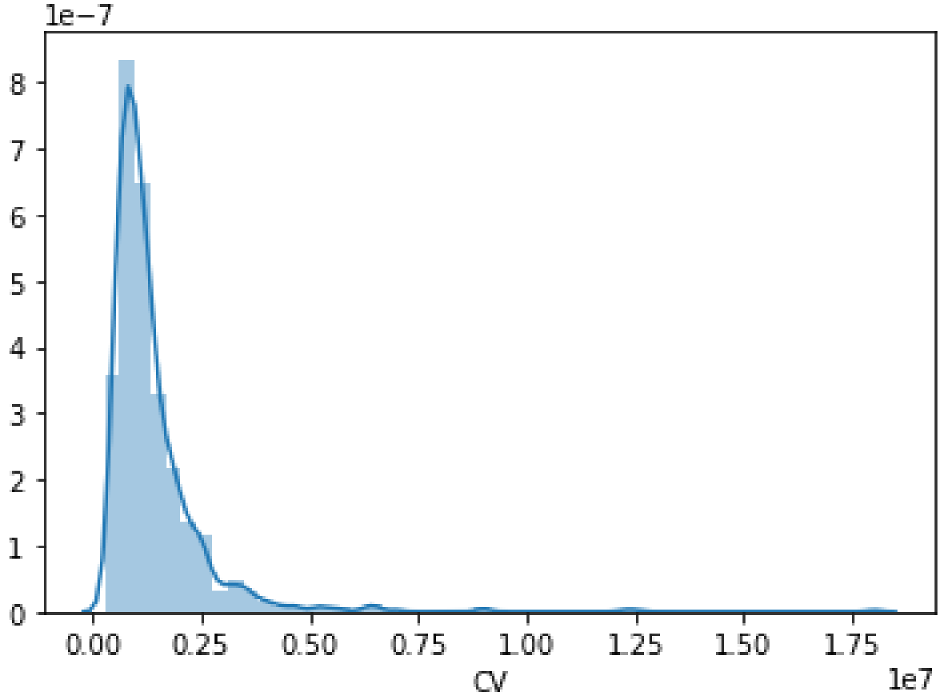


[18]:
```
#final_df.to_csv ('final_house.csv', header=True)
```

[19]:
```
sns.distplot(final_df['CV'])
#skewed CV so we should log(CV) and use median
```

```
CV is highly left skewed so we will either log(CV) or change the distribution in R
↪further on. I have exported the data set so we can now import it in R.
```

[19]: `<matplotlib.axes._subplots.AxesSubplot at 0x7fcab18ff7f0>`


1e−7
8
7
6
5
4
3
2
1
0
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
CV
1e7