Housing Rent for the major Canadian Cities

Susiette Adams, Student ID 303594

July 29 2019

Abstract

This project was assigned by York University 1050 Advanced Analytics Capstone Course. The goal for this project is to find insights from the housing market using analytical tools.

Scope

Here are the boundaries of the project:

- Upon completion the results will outline the cities that are most livable / cost effective for renter's base on the locations.
- Evaluating factors such as household income and market indicators will be used to determine if there are correlations between the costs of the housing in the top Canadian cities.
- Completion of this project will be by August 27th 2019.

Research questions

- What are the top 10 cities with the highest and lowest average income?
- What cities have the highest and lowest rent cost?
- What are the Market Indicators that affects rents cost in these cities?

What is needed to be done to run the codes.

Python programming language was used to do perform the analysis. To be able to run the codes and see the outputs you will need to have Python installed on your computer. All libraries that is needed to run the codes have been inputted within the codes.

Introduction and overall approach

Below you won't be able to see the outputs of the codes because the file was opened in a WordPad document, but you can click on the hyperlink that will take you to the .py python file. There were 3 data sets that were used this analysis and the same approach was used to perform and will be used to complete the feature engineering process and the Exploratory Data Analysis. In the datasets information on all Canada's provinces were provided along with their cities, but only the cities we will be focusing on.

- Loading libraries
- Data Preparation/ loading the data
- Viewing the data contents
- Checking the shape and size of the dataset
- Checking for missing values
- Checking the mean and median
- Checking the correlations of the data
- Checking that data for positive and negative features
- Check the data for the top 10 and bottom 10 cities for apartment prices.
- Create a histogram with the numeric variables
- Create scatterplots to check if the data is good for regression
- Check data for outliers and us box plots to visualize the outlier
- Check the index values

```
#!/usr/bin/env python
# coding: utf-8

# In[1]:


# Loading packages
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import os
get_ipython().run_line_magic('matplotlib', 'inline')


# In[88]:


# Importing the Renter Household data
rent= pd.read_csv(r"C:\\Users\\susie\\Documents\\Data Science\\Capstone
data\\Average rent for 2 bedrooms.csv"
                                ,encoding="latin")
mydata.info()


# In[89]:


# Taking a peek at the data the rent.head function inly returns the first five rows
because there are 26 columns we are unable to see all of them.
print(rent.head())
```

In[90]:

# The .to_string() method give a much better view of the data contents and I have selecting 10 rows to view
print(rent.head(10).to_string())

In[92]:

Checking the number of rows and columns
rent.shape

In[93]:

# The df.info() method is great for seeing all the columns in the dataset and it will also give you a quick glance at null values and the data types float, object for my features.
print(rent.info())

As you can see above all features have 45 total entries and 45 non-null entries there are no missing entries in the data.

In[91]:

# We have to sure we set the data up in order to be used in regression so I am just double checking for null values and this confirms that there are none.
print(rent.isnull().sum().to_string())

With no missing values this makes it easier to clean the data.

In[227]:

Location is one of the most important factors in real estate and the â€˜Citiesâ€™ provides insight into this.
We use the .groupby() method to do this so we can take a quick glance at the Cities mean prices
rent.groupby('City', as_index=True)['1992', '1993', '1994', '1995', '1996', '1997', '1998', '1999',

```
        '2000', '2001', '2002', '2003', '2004', '2005', '2006', '2007', '2008',
        '2009', '2010', '2011', '2012', '2013', '2014', '2015', '2016'].mean()


# We can see the output above and itâ€™s quickly apparent that cities is a major
factor in prices.

# Here are the results for median. Note that the values for most Cities prices stayed
the say
# come down with median, which provides the insight that weâ€™re more likely to see
outliers on the â€˜high endâ€™ than the low end which is typical in real estate.

# In[229]:


# We can use this in the groupby function and then divide by the mean.
rent.groupby('City', as_index=True)[].mean()/ rent.groupby('City',
as_index=True)['2007', '2008',
        '2009', '2010', '2011', '2012', '2013', '2014', '2015', '2016].mean()


# This ends up being a pretty insightful variable as we can see at least some of the
difference in mean prices area result of the year 2013.

# In[100]:


# Using Seaborn heatmap to analyze correlation in the data
explore = rent[['City', '1992', '1993', '1994', '1995', '1996', '1997', '1998', '1999',
        '2000', '2001', '2002', '2003', '2004', '2005', '2006', '2007', '2008',
        '2009', '2010', '2011', '2012', '2013', '2014', '2015', '2016']]
explore


# In[102]:


# Plotting the correlation data
corr = explore.corr()
plt.figure(figsize = (10,12))
sns.heatmap(corr,
                xticklabels=corr.columns.values,
                yticklabels=corr.columns.values)


# I used rent.columns to get all the column names with â€˜Cityâ€™.
```

In[166]:

```
# Plotting the correlation data to get a better visual on the graph
f,ax = plt.subplots(figsize=(10,10))
sns.heatmap(rent.corr(), annot = True,linewidths=.4, fmt='.1f', ax=ax)
plt.show()
```

In[114]:

```
# what Iâ€™m doing here is using rent.corr() to get the correlations. I am then turning
that information into a dictionary with .to_dict(),
# while .items() is turning the correlations into key, value pairs necessary for a
dictionary. Weâ€™re correlating by â€˜2014â€™.
# We use the lambda expression to order things properly. Finally, we use .sorted()
method to sort the values, with the argument â€˜reverse=Trueâ€™,
# so that the highest correlation shows up at the top.
corr_list = sorted(rent.corr().to_dict()['2014'].items(), key=lambda x: x[1], reverse=True)
corr_list
```

What this allows us to do is quickly see which features have the most significant (positive or negative) correlations, and also see which features are unlikely to be that relevant. For this dataset, I find that 2014, overall quality, 2013 and 2015 are the most correlated features with City.

```
# # The rent dataset includes data for all of Canada's provinces and major cities
# For my analysis I will only be focusing on the top cities.
```

In[213]:

```
# Using the n.largest I will search the data to find the highest prices for 2 bedroom
apartments.
lrg = rent.nlargest(10, ['1992', '1993', '1994', '1995', '1996', '1997', '1998', '1999',
            '2000', '2001', '2002', '2003', '2004', '2005', '2006', '2007', '2008',
            '2009', '2010', '2011', '2012', '2013', '2014', '2015', '2016'])
lrg
```

In the ouput above there are 8 cities and 2 provinces British Columbia and Ontario which are not relavant for my analysis so I will only forcus on the 8 cities.

In[231]:

```
# histogram of the average rent in 2014 for the 8 most expensive cities
sns.distplot(lrg['2014']);


# In[218]:


# The numeric variables in the dataset
numerical = [
      '1992', '1993', '1994', '1995', '1996',
          '1997', '1998', '1999', '2000', '2001', '2002', '2003', '2004', '2005',
          '2006', '2007', '2008', '2009', '2010', '2011', '2012', '2013', '2014',
          '2015', '2016'
]


# In[221]:


# Histograms of the nueric variables using only information for the 8 most expensive
cities
lrg[numerical].hist(bins=15, figsize=(25, 10), layout=(8, 4));


# In[262]:


# Here is a clearer view of 2016 rent prices
sns.countplot(lrg['2016']);


# In[233]:


## Scatter Plot of the average rent in 2013, 2014 and 2015 using the filtered data for
the top 8 cites
lrg.plot(kind='scatter',x = '2013', y = '2014', alpha=0.5, color='b')
plt.xlabel('2013')
plt.ylabel('2014')
plt.title('Average rent in 2013 to 2013')
lrg.plot(kind='scatter',x = '2014', y = '2015', alpha=0.5, color='r')
plt.xlabel('2014')
plt.ylabel('2015')
plt.title('Average rent in 2014 to 2015')
```

In[ ]:

#I choose these 3 years to plot because they have the strongest correlation in the dataset.
With these 2 plots it gives us a shape of the data and tells us it's good for regression.

In[222]:

The .nsmallest was used t get the cities with the lowest prices for 2 bedroom apartments
```
sml = rent.nsmallest(10, ['1992', '1993', '1994', '1995', '1996', '1997', '1998', '1999',
            '2000', '2001', '2002', '2003', '2004', '2005', '2006', '2007', '2008',
            '2009', '2010', '2011', '2012', '2013', '2014', '2015', '2016'])
sml
```

In[ ]:

In the output above .

In[253]:

```
# Detect Outliers
sns.boxplot(data=sml)
```

In[ ]:

There seem to be few outliers in the data which we would have to take a closer look at.

In[252]:

```
# Looking for outliers
sns.boxplot(data=lrg)
```

In[ ]:

There seem to be few outliers in the data which we would have to take a closer look at.

In[256]:

sns.boxplot(lrg['2016'])

In[ ]:

There seem to be few outliers in the data which we would have to take a closer look at.

In[237]:

Check the index values
rent.index.values

In[137]:

# Loading the income after tax dataset
income= pd.read_csv(r"C:\\Users\\susie\\Documents\\Data Science\\Capstone data\\Average after tax income.csv"
                    ,encoding="latin1")
income.info()

In[139]:

# Taking a peek at the data, by using the df.head() functiononly returns the first five rows because there are 13 columns we are unable to see all of them.
print(income.head())

In[141]:

The .to_string() method give a much better view of the data contents and I have selecting 10 rows to view

print(income.head(20).to_string())

In[140]:

Checking the shape of the data which outputs 40 rows and 13 columns
income.shape

In[142]:

# The df.info() method is great for seeing all the columns in the dataset and it will also give you a quick glance at null values and the data types float, object for my features.
print(income.info())

As you can see above all features have 40 total entries and 40 non-null entries we have no missing entries.

In[143]:

For our data types Cities column is an object and all other columns are int which is what we want to see. There isn't much cleaning to be done with ut data
income.dtypes

In[144]:

# We have to make sure we set the data up in order to be used in regression so I am double check for null values.
print(rent.isnull().sum().to_string())

The above code is to sum up the null values and they all out put zero

In[145]:

```
# Describing the entire DataFrame it only shows results for columns with numeric
datatypes.
income.describe()


# In[152]:


income.columns


# In[156]:


# Cities provides insight into this averge income and seem to have a big factor on
income rates.
# We use the .groupby() method to do this so we can take a quick glance at the Cities
mean income'
income.groupby('Cities', as_index=True)['2006', '2007', '2008', '2009', '2010', '2011',
'2012',
       '2013', '2014', '2015', '2016', '2017'].mean()


# We can see the output above and itâ€™s quickly apparent that Cities is a major
factor on the salary.

# In[162]:


# Using Seaborn heatmap to analyze correlation in the data

explore1 = income[['2006', '2007', '2008', '2009', '2010', '2011', '2012',
       '2013', '2014', '2015', '2016', '2017']]
explore1


# In[263]:


# With this we get the visual of our heat map
corr = explore1.corr()
plt.figure(figsize = (10,10))
sns.heatmap(corr,
            xticklabels=corr.columns.values,
            yticklabels=corr.columns.values)
```

In[ ]:

From the visual above it seem like 2006, 2012 and 2016 has the strongest correlation in the income dataset

In[264]:

Here is a better visual on the correlation graph
f,ax = plt.subplots(figsize=(10,10))
sns.heatmap(income.corr(), annot = True,linewidths=.4, fmt='.1f', ax=ax)
plt.show()

In[265]:

# what Iâ€™m doing is using rent.corr() to get the correlations. I am then turning that information into a dictionary with .to_dict(),
# while .items() is turning the correlations into key, value pairs necessary for a dictionary. Weâ€™re correlating by â€˜2014â€™.
# We use the lambda expression to order things properly. Finally, we use .sorted() method to sort the values, with the argument â€˜reverse=Trueâ€™,
# so that the highest correlation shows up at the top.
corr_list1 = sorted(income.corr().to_dict()['2006'].items(), key=lambda x: x[1], reverse=True)
corr_list1

What this allows us to do is quickly see which features have the most significant (positive or negative) correlations, and also see which features are unlikely to be that relevant.

The rent dataset includes data for all of Canada's provinces and major citiesÂ¶
For my analysis I will only be focusing on the top cities.

In[271]:

income.columns

In[272]:

```
# Using the n.largest I will search the data to find cities with the highest income.
high = income.nlargest(10, ['2006', '2007', '2008', '2009', '2010', '2011', '2012',
          '2013', '2014', '2015', '2016', '2017'])
high


# In the output above there are 9 cities with the highest income bracket and 1
province which is not relevant to our analysis because we are only focusing on the
cities.

# In[273]:


# histogram of the average income in 2016 for the 9 most expensive cities
sns.distplot(high['2016']);


# In[171]:


# The numeric variables in the dataset
numerical1 = ['2006', '2007', '2008', '2009', '2010', '2011', '2012',
          '2013', '2014', '2015', '2016', '2017']


# In[285]:


# Histograms of the nueric variables
high[numerical1].hist(bins=15, figsize=(15, 8), layout=(8, 3));


# In[287]:


# Here is a clearer view of 2017 income range
sns.countplot(high['2017']);


# In[288]:


## Scatter Plot of the average rent in 2006, 2007 and 2017 using the filtered data for
the top 9 cites
lrg.plot(kind='scatter',x = '2006', y = '2007', alpha=0.5, color='b')
plt.xlabel('2013')
plt.ylabel('2014')
```

```
plt.title('Average rent in 2006 to 2007')
lrg.plot(kind='scatter',x = '2014', y = '2015', alpha=0.5, color='r')
plt.xlabel('2007')
plt.ylabel('2017')
plt.title('Average income in 2007 to 2017')


# I choose these 3 years to plot because they amongs the strongest correlation in the
dataset.
# With these 2 plots it gives us a shape of the data and tells us it's good for
regression.

# In[289]:


# Using the n.largest I will search the data to find cities with the highest income.
low = income.nsmallest(10, ['2006', '2007', '2008', '2009', '2010', '2011', '2012',
           '2013', '2014', '2015', '2016', '2017'])
low


# In the output above there are 7 cities with the lowest prices and 3 provinces which
will have to be removed later because we are only interested in the cities they are New
Brunswick, Prince Edward Island and Manitoba.

# In[290]:


# Looking for outliers in the data
sns.boxplot(data=high)


# In[ ]:


# There seem to be few outliers in the data which we would have to take a closer look
at.


# In[291]:


# Looking for outliers
sns.boxplot(data=low)


# In[ ]:
```

There seem to be few outliers in the data which we would have to take a closer look at.

In[138]:

Checking the data index values
income.index.values

In[139]:

Checking the column names
income.columns

In[184]:

# Loading the monthly debt payments
market= pd.read_csv(r"C:\\Users\\susie\\Documents\\Data Science\\Capstone data\\Housing market indicators.csv"
                    ,encoding="latin1")
market.info()

In[185]:

# Taking a peek at the data the rent.head function only returns the first five rows because there are 26 columns we are unable to see all of them.
print(rent.head())

In[186]:

# The .to_string() method give a much better view of the data contents and I have selecting 10 rows to view
print(market.head(10).to_string())

In[187]:

```
# Checking the number of rows and columns
market.shape


# In[188]:


# The df.info() method is great for seeing all the columns in the dataset and it will also give you a quick glance at null values and the data types    float, object for my features.
print(market.info())


# As you can see above all features have 36 total entries and only the Indicators column has no non-null entries there are fer non null entries in all the other columns.

# In[189]:


# We have to sure we set the data up in order to be used in regression weâ€™ll also need to deal with the null values.
# lets take a look at the null values
print(market.isnull().sum().to_string())


# As you can see only one of the column has 0 null 1900 and 1980 has the most null values.

# In[190]:


def object_vcs_and_no_nulls(market):
    for i in market:
        if market[i].dtype == 'O':
            if market[i].isnull().sum() == 0:
                print(market[i].value_counts())
                print("Number of Null Values: " + str(market[i].isnull().sum()))
                print("Percentage of Nulls = " + str(np.round((market[i].isnull().sum() / 14.60), 2)) + "%")
                print("\n")

object_vcs_and_no_nulls(market)


# In[ ]:
```

#Before we transform, we should explore these variables a bit. For some features, there is a clear order
#(e.g. a â€˜qualityâ€™ variable that has â€˜Excellentâ€™, â€˜Goodâ€™, â€˜Fairâ€™ and â€˜Poorâ€™ as the possible values).

In[295]:

```
market.groupby('Indicators', as_index=True)['1990', '1991', '1992', '1993', '1994', '1995', '1996',
        '1997', '1998', '1999', '2000', '2001', '2002', '2003', '2004', '2005',
        '2006', '2007', '2008', '2009', '2010', '2011', '2012', '2013', '2014',
        '2015', '2016'].mean()
```

In[192]:

```
market.columns
```

In[304]:

```
# Wr use the .groupby() method to do this so we can take a quick glance at the Cities mean and median prices
market.groupby('Indicators', as_index=True)['1991', '1992', '1993', '1994', '1995', '1996',
        '1997', '1998', '1999', '2000', '2001', '2002', '2003', '2004', '2005',
        '2006', '2007', '2008', '2009', '2010', '2011', '2012', '2013', '2014',
        '2015', '2016'].mean()
```

In[ ]:

We can see the output above and itâ€™s quickly apparent that indicators is a major factor in the values.

In[302]:

```
# We are also going to use median, as well, since there might be some outliers that
skew the data.
market.groupby('Indicators', as_index=True)['1991', '1992', '1993', '1994', '1995',
'1996',
        '1997', '1998', '1999', '2000', '2001', '2002', '2003', '2004', '2005',
        '2006', '2007', '2008', '2009', '2010', '2011', '2012', '2013', '2014',
        '2015', '2016'].median()


# In[305]:


# We can use this in the groupby function and then divide by the mean.
market.groupby('Indicators', as_index=True)['2015'].mean()/
market.groupby('Indicators', as_index=True)['2016'].mean()


# In[198]:


## I used income columns to get all the column names and spread the features out
for our heat map
explore2 = market[['1990', '1991', '1992', '1993', '1994', '1995', '1996',
        '1997', '1998', '1999', '2000', '2001', '2002', '2003', '2004', '2005',
        '2006', '2007', '2008', '2009', '2010', '2011', '2012', '2013', '2014',
        '2015', '2016']]
explore2


# In[306]:


# With this we get the visual of our heat map
corr = explore2.corr()
plt.figure(figsize = (10,12))
sns.heatmap(corr,
                xticklabels=corr.columns.values,
                yticklabels=corr.columns.values)


# In[307]:


# Plotting the correlation data to get a better visual on the graph
f,ax = plt.subplots(figsize=(10,10))
sns.heatmap(explore2.corr(), annot = True,linewidths=.4, fmt='.1f', ax=ax)
plt.show()
```

In[308]:

```
# what Iâ€™m doing here is using rent.corr() to get the correlations. I am then turning
that information into a dictionary with .to_dict(),
# while .items() is turning the correlations into key, value pairs necessary for a
dictionary. Weâ€™re correlating by â€˜2015â€™.
# We use the lambda expression to order things properly. Finally, we use .sorted()
method to sort the values, with the argument â€˜reverse=Trueâ€™,
# so that the highest correlation shows up at the top.
corr_list2 = sorted(explore2.corr().to_dict()['2015'].items(), key=lambda x: x[1],
reverse=True)
corr_list2
```

In[ ]:

```
# What this allows us to do is quickly see which features have the most significant
(positive or negative) correlations,
#and also see which features are unlikely to be that relevant. For this dataset, I find
that 2015, overall quality, 2012 and 2001 seem to be the most correlated features.
```

In[310]:

```
# Using the n.largest I will search the data to find the top 10 indicators that affects
houssing cost.
top = market.nlargest(10, ['1991', '1992', '1993', '1994', '1995', '1996',
          '1997', '1998', '1999', '2000', '2001', '2002', '2003', '2004', '2005',
          '2006', '2007', '2008', '2009', '2010', '2011', '2012', '2013', '2014',
          '2015', '2016'])
top
```

In[311]:

```
# histogram of the top 10 housing indicators
sns.distplot(lrg['2016']);
```

In[313]:

Scatter Plot of the top 10 housing indicators
```
top.plot(kind='scatter',x = '2013', y = '2014', alpha=0.5, color='b')
plt.xlabel('2013')
plt.ylabel('2014')
plt.title('Top 10 housing indicators 2013 to 2014')
top.plot(kind='scatter',x = '2014', y = '2015', alpha=0.5, color='r')
plt.xlabel('2014')
plt.ylabel('2015')
plt.title('Top 10 housing indicators 2014 to 2015')
```

In[ ]:

With these 2 plots it gives us a shape of the data and tells us it's good for regression.

In[315]:

Using the n.smallest I will search the data to find the top 10 indicators that affects houssing cost.
```
low = market.nsmallest(10, ['1991', '1992', '1993', '1994', '1995', '1996',
            '1997', '1998', '1999', '2000', '2001', '2002', '2003', '2004', '2005',
            '2006', '2007', '2008', '2009', '2010', '2011', '2012', '2013', '2014',
            '2015', '2016'])
low
```

In[316]:

Detect Outliers
```
sns.boxplot(data=top)
```

In[317]:

Detect Outliers
```
sns.boxplot(data=low)
```

In[ ]:

There seem to be few outliers in the data which we would have to take a closer look at.

Observation and recommendation

Base on the exploratory analysis a decision will be made to assess whether the research questions are achievable contrarily if we should review the research questions.

Research question #1

What are the top 10 cities with the highest and lowest average income?

There are 7 cities with the lowest income and 3 provinces they are New Brunswick, Prince Edward Island and Manitoba, but we are only interested in the cities. There are 9 cities with the highest income bracket and 1 province which is not relevant to our analysis because we are only focusing on the cities. We will take some time and exclude these unwanted provinces form our selections.

Research question #2

What cities have the highest and lowest rent cost?

For this question 8 cities were in the top 10 and 2 provinces British Columbia and Ontario which are not relevant for the analysis so we will only focus on the 8 cities. There are 7 cities with the lowest prices and 3 provinces which. This should be achievable to answer the research question by editing the top number. We will have to take some time to exclude the provinces from our selections.

Research question #3

What are the Market Indicators that affects rents cost in these cities?

For this question the top 10 market indicators that affects rent and house prices are residential building permits, completion total, start by intended market, multiple, single detach home owner free hold, apartment, rental, population and row percentage change. The bottom 10 market indicators are real disposal income, rental vacancy, new housing price index, employment rate, rental accommodation cost, consumer price index, unemployment rate, labor force participation, and bachelor. This information shows that we can answer the research question.

Steps to be taken

Our next step is to complete the data cleanup process and assign values in the missing data for the market indicator dataset. Then remove outliers to prepare the datasets for modelling.