|  |  |
| --- | --- |
|  | Real Estate |
|  |  |
|  | Margil Shah  DATA SCIENCE CAPSTONE  6/1/21 |

Source Code

Table of Contents

[DESCRIPTION 2](#_Toc73466112)

[Problem Statement 2](#_Toc73466113)

[Dataset Description 2](#_Toc73466114)

[Solutions: 2](#_Toc73466115)

[Import plotting libraries 2](#_Toc73466116)

[Project Task: Week 1 3](#_Toc73466117)

[Data Import and Preparation: 3](#_Toc73466118)

[Exploratory Data Analysis (EDA): 16](#_Toc73466119)

[Project Task: Week 2 24](#_Toc73466120)

[Exploratory Data Analysis (EDA): 24](#_Toc73466121)

[Project Task: Week 3 33](#_Toc73466122)

[Data Pre-processing: 33](#_Toc73466123)

[Project Task: Week 4 36](#_Toc73466124)

[Data Modeling : 36](#_Toc73466125)

[Data Reporting: 42](#_Toc73466126)

# DESCRIPTION

## **Problem Statement**

* A banking institution requires actionable insights into mortgage-backed securities, geographic business investment, and real estate analysis.
* The mortgage bank would like to identify potential monthly mortgage expenses for each region based on monthly family income and rental of the real estate.
* A statistical model needs to be created to predict the potential demand in dollars amount of loan for each of the region in the USA. Also, there is a need to create a dashboard which would refresh periodically post data retrieval from the agencies.
* The dashboard must demonstrate relationships and trends for the key metrics as follows: number of loans, average rental income, monthly mortgage and owner’s cost, family income vs mortgage cost comparison across different regions. The metrics described here do not limit the dashboard to these few.

## **Dataset Description**

|  |  |
| --- | --- |
| **Variables** | **Description** |
| Second mortgage | Households with a second mortgage statistics |
| Home equity | Households with a home equity loan statistics |
| Debt | Households with any type of debt statistics |
| Mortgage Costs | Statistics regarding mortgage payments, home equity loans, utilities, and property taxes |
| Home Owner Costs | Sum of utilities, and property taxes statistics |
| Gross Rent | Contract rent plus the estimated average monthly cost of utility features |
| High school Graduation | High school graduation statistics |
| Population Demographics | Population demographics statistics |
| Age Demographics | Age demographic statistics |
| Household Income | Total income of people residing in the household |
| Family Income | Total income of people related to the householder |

# **Solutions:**

## **Import plotting libraries**

import time

import random

from math import \*

import operator

import pandas as pd

import numpy as np

# import plotting libraries

import matplotlib

import matplotlib.pyplot as plt

from pandas.plotting import scatter\_matrix

%matplotlib inline

import seaborn as sns

sns.set(style="white", color\_codes=True)

sns.set(font\_scale=1.5)

# **Project Task: Week 1**

## **Data Import and Preparation**:

1. Import data.

# 1. Import Data

df\_train=pd.read\_csv("train.csv")

df\_test=pd.read\_csv("test.csv")

df\_train.columns

Index(['UID', 'BLOCKID', 'SUMLEVEL', 'COUNTYID', 'STATEID', 'state',

       'state\_ab', 'city', 'place', 'type', 'primary', 'zip\_code', 'area\_code',

       'lat', 'lng', 'ALand', 'AWater', 'pop', 'male\_pop', 'female\_pop',

       'rent\_mean', 'rent\_median', 'rent\_stdev', 'rent\_sample\_weight',

       'rent\_samples', 'rent\_gt\_10', 'rent\_gt\_15', 'rent\_gt\_20', 'rent\_gt\_25',

       'rent\_gt\_30', 'rent\_gt\_35', 'rent\_gt\_40', 'rent\_gt\_50',

       'universe\_samples', 'used\_samples', 'hi\_mean', 'hi\_median', 'hi\_stdev',

       'hi\_sample\_weight', 'hi\_samples', 'family\_mean', 'family\_median',

       'family\_stdev', 'family\_sample\_weight', 'family\_samples',

       'hc\_mortgage\_mean', 'hc\_mortgage\_median', 'hc\_mortgage\_stdev',

       'hc\_mortgage\_sample\_weight', 'hc\_mortgage\_samples', 'hc\_mean',

       'hc\_median', 'hc\_stdev', 'hc\_samples', 'hc\_sample\_weight',

       'home\_equity\_second\_mortgage', 'second\_mortgage', 'home\_equity', 'debt',

       'second\_mortgage\_cdf', 'home\_equity\_cdf', 'debt\_cdf', 'hs\_degree',

       'hs\_degree\_male', 'hs\_degree\_female', 'male\_age\_mean',

       'male\_age\_median', 'male\_age\_stdev', 'male\_age\_sample\_weight',

       'male\_age\_samples', 'female\_age\_mean', 'female\_age\_median',

       'female\_age\_stdev', 'female\_age\_sample\_weight', 'female\_age\_samples',

       'pct\_own', 'married', 'married\_snp', 'separated', 'divorced'],

      dtype='object')

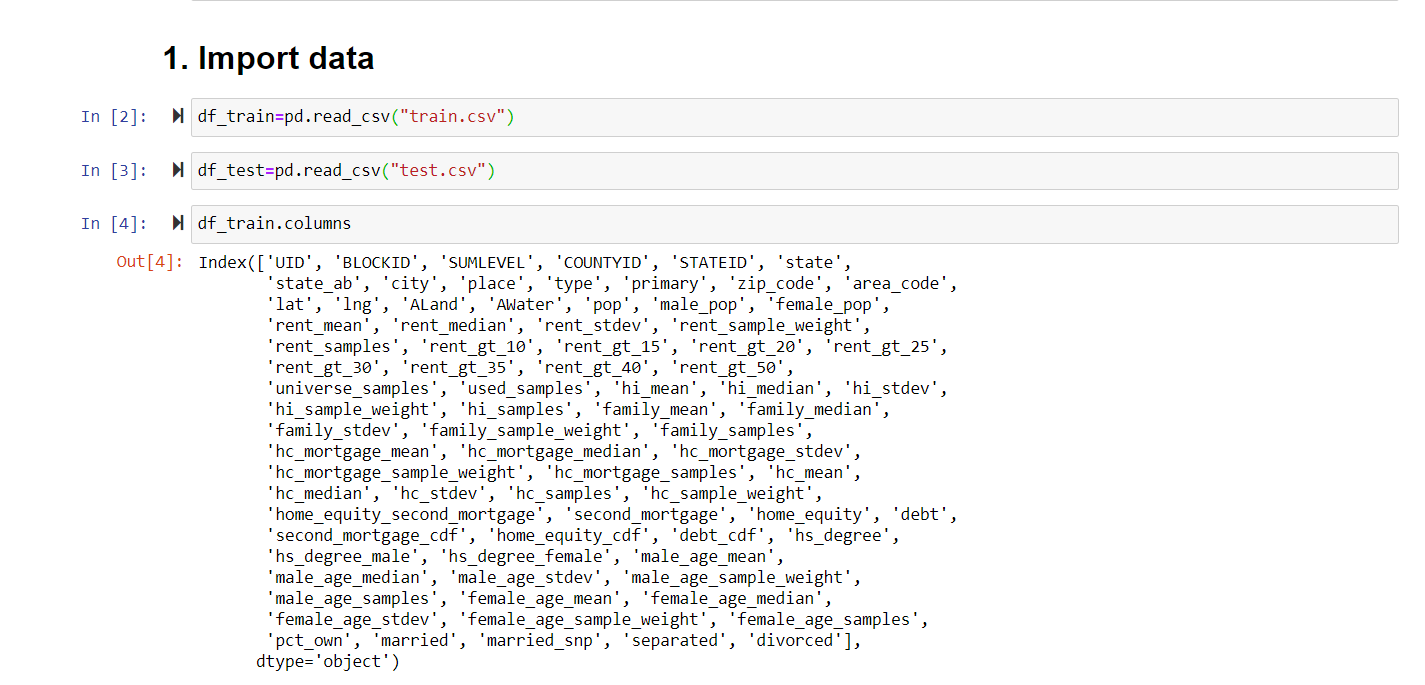


Figure 1 Import Data

df\_test.columns

Index(['UID', 'BLOCKID', 'SUMLEVEL', 'COUNTYID', 'STATEID', 'state',

       'state\_ab', 'city', 'place', 'type', 'primary', 'zip\_code', 'area\_code',

       'lat', 'lng', 'ALand', 'AWater', 'pop', 'male\_pop', 'female\_pop',

       'rent\_mean', 'rent\_median', 'rent\_stdev', 'rent\_sample\_weight',

       'rent\_samples', 'rent\_gt\_10', 'rent\_gt\_15', 'rent\_gt\_20', 'rent\_gt\_25',

       'rent\_gt\_30', 'rent\_gt\_35', 'rent\_gt\_40', 'rent\_gt\_50',

       'universe\_samples', 'used\_samples', 'hi\_mean', 'hi\_median', 'hi\_stdev',

       'hi\_sample\_weight', 'hi\_samples', 'family\_mean', 'family\_median',

       'family\_stdev', 'family\_sample\_weight', 'family\_samples',

       'hc\_mortgage\_mean', 'hc\_mortgage\_median', 'hc\_mortgage\_stdev',

       'hc\_mortgage\_sample\_weight', 'hc\_mortgage\_samples', 'hc\_mean',

       'hc\_median', 'hc\_stdev', 'hc\_samples', 'hc\_sample\_weight',

       'home\_equity\_second\_mortgage', 'second\_mortgage', 'home\_equity', 'debt',

       'second\_mortgage\_cdf', 'home\_equity\_cdf', 'debt\_cdf', 'hs\_degree',

       'hs\_degree\_male', 'hs\_degree\_female', 'male\_age\_mean',

       'male\_age\_median', 'male\_age\_stdev', 'male\_age\_sample\_weight',

       'male\_age\_samples', 'female\_age\_mean', 'female\_age\_median',

       'female\_age\_stdev', 'female\_age\_sample\_weight', 'female\_age\_samples',

       'pct\_own', 'married', 'married\_snp', 'separated', 'divorced'],

      dtype='object')

len(df\_train)

27321

len(df\_test)

11709

df\_train.head()

UID BLOCKID SUMLEVEL    COUNTYID    STATEID state   state\_ab    city    place   type    ... female\_age\_mean female\_age\_median   female\_age\_stdev    female\_age\_sample\_weight    female\_age\_samples  pct\_own married married\_snp separated   divorced

0   267822  NaN 140 53  36  New York    NY  Hamilton    Hamilton    City    ... 44.48629    45.33333    22.51276    685.33845   2618.0  0.79046 0.57851 0.01882 0.01240 0.08770

1   246444  NaN 140 141 18  Indiana IN  South Bend  Roseland    City    ... 36.48391    37.58333    23.43353    267.23367   1284.0  0.52483 0.34886 0.01426 0.01426 0.09030

2   245683  NaN 140 63  18  Indiana IN  Danville    Danville    City    ... 42.15810    42.83333    23.94119    707.01963   3238.0  0.85331 0.64745 0.02830 0.01607 0.10657

3   279653  NaN 140 127 72  Puerto Rico PR  San Juan    Guaynabo    Urban   ... 47.77526    50.58333    24.32015    362.20193   1559.0  0.65037 0.47257 0.02021 0.02021 0.10106

4   247218  NaN 140 161 20  Kansas  KS  Manhattan   Manhattan City  City    ... 24.17693    21.58333    11.10484    1854.48652  3051.0  0.13046 0.12356 0.00000 0.00000 0.03109

df\_test.head()

UID BLOCKID SUMLEVEL    COUNTYID    STATEID state   state\_ab    city    place   type    ... female\_age\_mean female\_age\_median   female\_age\_stdev    female\_age\_sample\_weight    female\_age\_samples  pct\_own married married\_snp separated   divorced

0   255504  NaN 140 163 26  Michigan    MI  Detroit Dearborn Heights City   CDP ... 34.78682    33.75000    21.58531    416.48097   1938.0  0.70252 0.28217 0.05910 0.03813 0.14299

1   252676  NaN 140 1   23  Maine   ME  Auburn  Auburn City City    ... 44.23451    46.66667    22.37036    532.03505   1950.0  0.85128 0.64221 0.02338 0.00000 0.13377

2   276314  NaN 140 15  42  Pennsylvania    PA  Pine City   Millerton   Borough ... 41.62426    44.50000    22.86213    453.11959   1879.0  0.81897 0.59961 0.01746 0.01358 0.10026

3   248614  NaN 140 231 21  Kentucky    KY  Monticello  Monticello City City    ... 44.81200    48.00000    21.03155    263.94320   1081.0  0.84609 0.56953 0.05492 0.04694 0.12489

4   286865  NaN 140 355 48  Texas   TX  Corpus Christi  Edroy   Town    ... 40.66618    42.66667    21.30900    709.90829   2956.0  0.79077 0.57620 0.01726 0.00588 0.16379

df\_train.describe()

UID BLOCKID SUMLEVEL    COUNTYID    STATEID zip\_code    area\_code   lat lng ALand   ... female\_age\_mean female\_age\_median   female\_age\_stdev    female\_age\_sample\_weight    female\_age\_samples  pct\_own married married\_snp separated   divorced

count   27321.000000    0.0 27321.0 27321.000000    27321.000000    27321.000000    27321.000000    27321.000000    27321.000000    2.732100e+04    ... 27115.000000    27115.000000    27115.000000    27115.000000    27115.000000    27053.000000    27130.000000    27130.000000    27130.000000    27130.000000

mean    257331.996303   NaN 140.0   85.646426   28.271806   50081.999524    596.507668  37.508813   -91.288394  1.295106e+08    ... 40.319803   40.355099   22.178745   544.238432  2208.761903 0.640434    0.508300    0.047537    0.019089    0.100248

std 21343.859725    NaN 0.0 98.333097   16.392846   29558.115660    232.497482  5.588268    16.343816   1.275531e+09    ... 5.886317    8.039585    2.540257    283.546896  1089.316999 0.226640    0.136860    0.037640    0.020796    0.049055

min 220342.000000   NaN 140.0   1.000000    1.000000    602.000000  201.000000  17.929085   -165.453872 4.113400e+04    ... 16.008330   13.250000   0.556780    0.664700    2.000000    0.000000    0.000000    0.000000    0.000000    0.000000

25% 238816.000000   NaN 140.0   29.000000   13.000000   26554.000000    405.000000  33.899064   -97.816067  1.799408e+06    ... 36.892050   34.916670   21.312135   355.995825  1471.000000 0.502780    0.425102    0.020810    0.004530    0.065800

50% 257220.000000   NaN 140.0   63.000000   28.000000   47715.000000    614.000000  38.755183   -86.554374  4.866940e+06    ... 40.373320   40.583330   22.514410   503.643890  2066.000000 0.690840    0.526665    0.038840    0.013460    0.095205

75% 275818.000000   NaN 140.0   109.000000  42.000000   77093.000000    801.000000  41.380606   -79.782503  3.359820e+07    ... 43.567120   45.416670   23.575260   680.275055  2772.000000 0.817460    0.605760    0.065100    0.027487    0.129000

max 294334.000000   NaN 140.0   840.000000  72.000000   99925.000000    989.000000  67.074018   -65.379332  1.039510e+11    ... 79.837390   82.250000   30.241270   6197.995200 27250.000000    1.000000    1.000000    0.714290    0.714290    1.000000

8 rows × 74 columns

df\_test.describe()

UID BLOCKID SUMLEVEL    COUNTYID    STATEID zip\_code    area\_code   lat lng ALand   ... female\_age\_mean female\_age\_median   female\_age\_stdev    female\_age\_sample\_weight    female\_age\_samples  pct\_own married married\_snp separated   divorced

count   11709.000000    0.0 11709.0 11709.000000    11709.000000    11709.000000    11709.000000    11709.000000    11709.000000    1.170900e+04    ... 11613.000000    11613.000000    11613.000000    11613.000000    11613.000000    11587.000000    11625.000000    11625.000000    11625.000000    11625.000000

mean    257525.004783   NaN 140.0   85.710650   28.489196   50123.418396    593.598514  37.405491   -91.340229  1.095500e+08    ... 40.111999   40.131864   22.148145   550.411243  2233.003186 0.634194    0.505632    0.047960    0.019346    0.099191

std 21466.372658    NaN 0.0 99.304334   16.607262   29775.134038    232.074263  5.625904    16.407818   7.624940e+08    ... 5.851192    7.972026    2.554907    280.992521  1072.017063 0.232232    0.139774    0.038693    0.021428    0.048525

min 220336.000000   NaN 140.0   1.000000    1.000000    601.000000  201.000000  17.965835   -166.770979 8.299000e+03    ... 15.360240   12.833330   0.737110    0.251910    3.000000    0.000000    0.000000    0.000000    0.000000    0.000000

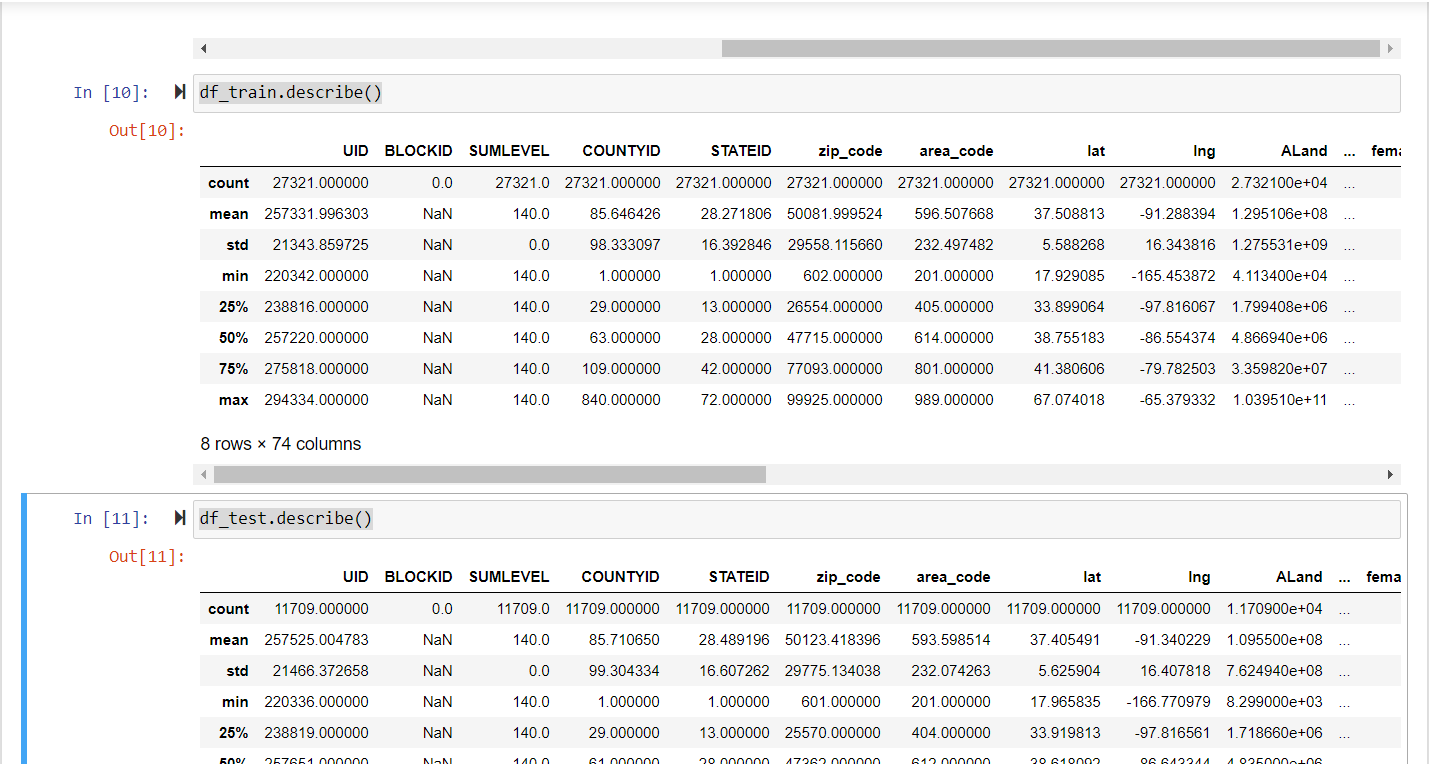
25% 238819.000000   NaN 140.0   29.000000   13.000000   25570.000000    404.000000  33.919813   -97.816561  1.718660e+06    ... 36.729210   34.750000   21.270920   363.225840  1499.000000 0.492500    0.422020    0.020890    0.004500    0.064590

50% 257651.000000   NaN 140.0   61.000000   28.000000   47362.000000    612.000000  38.618092   -86.643344  4.835000e+06    ... 40.196960   40.333330   22.472990   509.103610  2099.000000 0.687640    0.525270    0.038680    0.013870    0.094350

75% 276300.000000   NaN 140.0   109.000000  42.000000   77406.000000    787.000000  41.232973   -79.697311  3.204540e+07    ... 43.496490   45.333330   23.549450   685.883910  2800.000000 0.815235    0.605660    0.065340    0.027910    0.128400

max 294333.000000   NaN 140.0   810.000000  72.000000   99929.000000    989.000000  64.804269   -65.695344  5.520166e+10    ... 90.107940   90.166670   29.626680   4145.557870 15466.000000    1.000000    1.000000    0.714290    0.714290    0.362750

8 rows × 74 columns



df\_train.info()

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 27321 entries, 0 to 27320

Data columns (total 80 columns):

UID                            27321 non-null int64

BLOCKID                        0 non-null float64

SUMLEVEL                       27321 non-null int64

COUNTYID                       27321 non-null int64

STATEID                        27321 non-null int64

state                          27321 non-null object

state\_ab                       27321 non-null object

city                           27321 non-null object

place                          27321 non-null object

type                           27321 non-null object

primary                        27321 non-null object

zip\_code                       27321 non-null int64

area\_code                      27321 non-null int64

lat                            27321 non-null float64

lng                            27321 non-null float64

ALand                          27321 non-null float64

AWater                         27321 non-null int64

pop                            27321 non-null int64

male\_pop                       27321 non-null int64

female\_pop                     27321 non-null int64

rent\_mean                      27007 non-null float64

rent\_median                    27007 non-null float64

rent\_stdev                     27007 non-null float64

rent\_sample\_weight             27007 non-null float64

rent\_samples                   27007 non-null float64

rent\_gt\_10                     27007 non-null float64

rent\_gt\_15                     27007 non-null float64

rent\_gt\_20                     27007 non-null float64

rent\_gt\_25                     27007 non-null float64

rent\_gt\_30                     27007 non-null float64

rent\_gt\_35                     27007 non-null float64

rent\_gt\_40                     27007 non-null float64

rent\_gt\_50                     27007 non-null float64

universe\_samples               27321 non-null int64

used\_samples                   27321 non-null int64

hi\_mean                        27053 non-null float64

hi\_median                      27053 non-null float64

hi\_stdev                       27053 non-null float64

hi\_sample\_weight               27053 non-null float64

hi\_samples                     27053 non-null float64

family\_mean                    27023 non-null float64

family\_median                  27023 non-null float64

family\_stdev                   27023 non-null float64

family\_sample\_weight           27023 non-null float64

family\_samples                 27023 non-null float64

hc\_mortgage\_mean               26748 non-null float64

hc\_mortgage\_median             26748 non-null float64

hc\_mortgage\_stdev              26748 non-null float64

hc\_mortgage\_sample\_weight      26748 non-null float64

hc\_mortgage\_samples            26748 non-null float64

hc\_mean                        26721 non-null float64

hc\_median                      26721 non-null float64

hc\_stdev                       26721 non-null float64

hc\_samples                     26721 non-null float64

hc\_sample\_weight               26721 non-null float64

home\_equity\_second\_mortgage    26864 non-null float64

second\_mortgage                26864 non-null float64

home\_equity                    26864 non-null float64

debt                           26864 non-null float64

second\_mortgage\_cdf            26864 non-null float64

home\_equity\_cdf                26864 non-null float64

debt\_cdf                       26864 non-null float64

hs\_degree                      27131 non-null float64

hs\_degree\_male                 27121 non-null float64

hs\_degree\_female               27098 non-null float64

male\_age\_mean                  27132 non-null float64

male\_age\_median                27132 non-null float64

male\_age\_stdev                 27132 non-null float64

male\_age\_sample\_weight         27132 non-null float64

male\_age\_samples               27132 non-null float64

female\_age\_mean                27115 non-null float64

female\_age\_median              27115 non-null float64

female\_age\_stdev               27115 non-null float64

female\_age\_sample\_weight       27115 non-null float64

female\_age\_samples             27115 non-null float64

pct\_own                        27053 non-null float64

married                        27130 non-null float64

married\_snp                    27130 non-null float64

separated                      27130 non-null float64

divorced                       27130 non-null float64

dtypes: float64(62), int64(12), object(6)

memory usage: 16.7+ MB

df\_test.info()

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 11709 entries, 0 to 11708

Data columns (total 80 columns):

UID                            11709 non-null int64

BLOCKID                        0 non-null float64

SUMLEVEL                       11709 non-null int64

COUNTYID                       11709 non-null int64

STATEID                        11709 non-null int64

state                          11709 non-null object

state\_ab                       11709 non-null object

city                           11709 non-null object

place                          11709 non-null object

type                           11709 non-null object

primary                        11709 non-null object

zip\_code                       11709 non-null int64

area\_code                      11709 non-null int64

lat                            11709 non-null float64

lng                            11709 non-null float64

ALand                          11709 non-null int64

AWater                         11709 non-null int64

pop                            11709 non-null int64

male\_pop                       11709 non-null int64

female\_pop                     11709 non-null int64

rent\_mean                      11561 non-null float64

rent\_median                    11561 non-null float64

rent\_stdev                     11561 non-null float64

rent\_sample\_weight             11561 non-null float64

rent\_samples                   11561 non-null float64

rent\_gt\_10                     11560 non-null float64

rent\_gt\_15                     11560 non-null float64

rent\_gt\_20                     11560 non-null float64

rent\_gt\_25                     11560 non-null float64

rent\_gt\_30                     11560 non-null float64

rent\_gt\_35                     11560 non-null float64

rent\_gt\_40                     11560 non-null float64

rent\_gt\_50                     11560 non-null float64

universe\_samples               11709 non-null int64

used\_samples                   11709 non-null int64

hi\_mean                        11587 non-null float64

hi\_median                      11587 non-null float64

hi\_stdev                       11587 non-null float64

hi\_sample\_weight               11587 non-null float64

hi\_samples                     11587 non-null float64

family\_mean                    11573 non-null float64

family\_median                  11573 non-null float64

family\_stdev                   11573 non-null float64

family\_sample\_weight           11573 non-null float64

family\_samples                 11573 non-null float64

hc\_mortgage\_mean               11441 non-null float64

hc\_mortgage\_median             11441 non-null float64

hc\_mortgage\_stdev              11441 non-null float64

hc\_mortgage\_sample\_weight      11441 non-null float64

hc\_mortgage\_samples            11441 non-null float64

hc\_mean                        11419 non-null float64

hc\_median                      11419 non-null float64

hc\_stdev                       11419 non-null float64

hc\_samples                     11419 non-null float64

hc\_sample\_weight               11419 non-null float64

home\_equity\_second\_mortgage    11489 non-null float64

second\_mortgage                11489 non-null float64

home\_equity                    11489 non-null float64

debt                           11489 non-null float64

second\_mortgage\_cdf            11489 non-null float64

home\_equity\_cdf                11489 non-null float64

debt\_cdf                       11489 non-null float64

hs\_degree                      11624 non-null float64

hs\_degree\_male                 11620 non-null float64

hs\_degree\_female               11604 non-null float64

male\_age\_mean                  11625 non-null float64

male\_age\_median                11625 non-null float64

male\_age\_stdev                 11625 non-null float64

male\_age\_sample\_weight         11625 non-null float64

male\_age\_samples               11625 non-null float64

female\_age\_mean                11613 non-null float64

female\_age\_median              11613 non-null float64

female\_age\_stdev               11613 non-null float64

female\_age\_sample\_weight       11613 non-null float64

female\_age\_samples             11613 non-null float64

pct\_own                        11587 non-null float64

married                        11625 non-null float64

married\_snp                    11625 non-null float64

separated                      11625 non-null float64

divorced                       11625 non-null float64

dtypes: float64(61), int64(13), object(6)

memory usage: 7.1+ MB

1. Figure out the primary key and look for the requirement of indexing.

#UID is unique userID value in the train and test dataset. So an index can be created from the UID feature

df\_train.set\_index(keys=['UID'],inplace=True)#Set the DataFrame index using existing columns.

df\_test.set\_index(keys=['UID'],inplace=True)

df\_train.head(2)

df\_test.head(2)

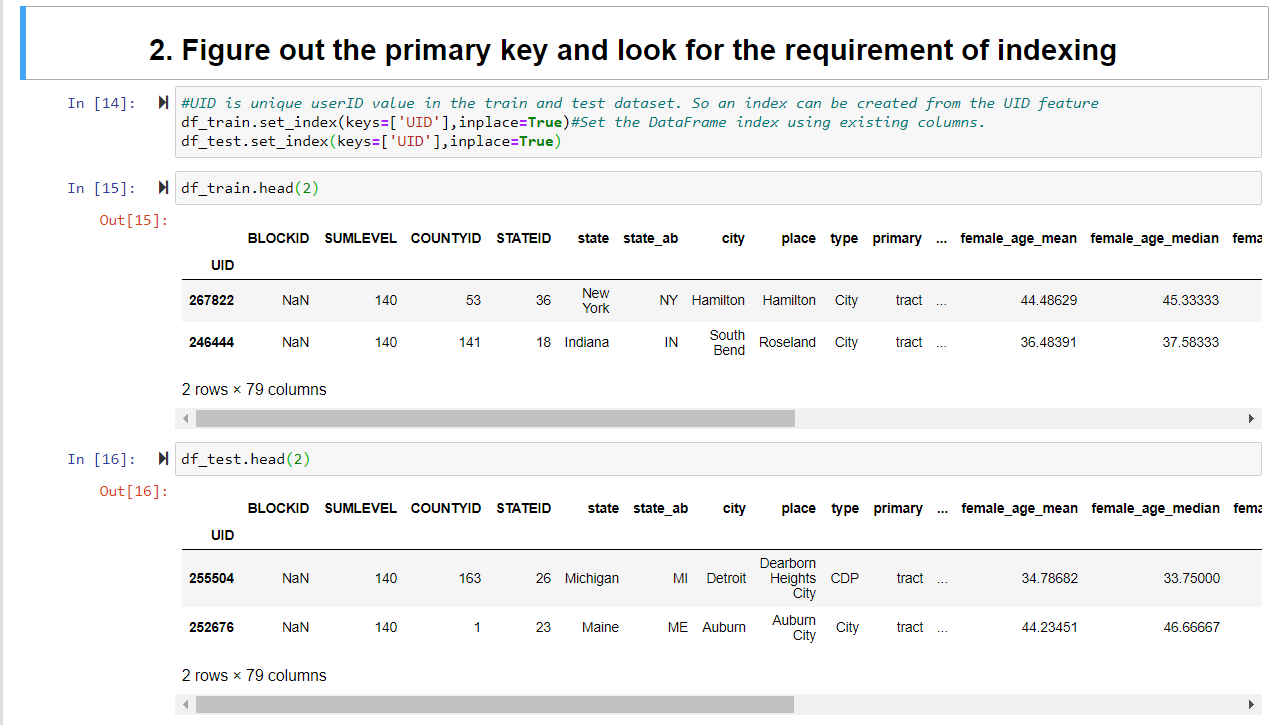


Figure 2 Figure out the primary key and look for the requirement of indexing

1. Gauge the fill rate of the variables and devise plans for missing value treatment. Please explain explicitly the reason for the treatment chosen for each variable.

#percantage of missing values in train set

missing\_list\_train=df\_train.isnull().sum() \*100/len(df\_train)

missing\_values\_df\_train=pd.DataFrame(missing\_list\_train,columns=['Percantage of missing values'])

missing\_values\_df\_train.sort\_values(by=['Percantage of missing values'],inplace=True,ascending=False)

missing\_values\_df\_train[missing\_values\_df\_train['Percantage of missing values'] >0][:10]

#BLOCKID can be dropped, since it is 100%missing values

#percantage of missing values in test set

missing\_list\_test=df\_test.isnull().sum() \*100/len(df\_train)

missing\_values\_df\_test=pd.DataFrame(missing\_list\_test,columns=['Percantage of missing values'])

missing\_values\_df\_test.sort\_values(by=['Percantage of missing values'],inplace=True,ascending=False)

missing\_values\_df\_test[missing\_values\_df\_test['Percantage of missing values'] >0][:10]

#BLOCKID can be dropped, since it is 43%missing values

df\_train .drop(columns=['BLOCKID','SUMLEVEL'],inplace=True) #SUMLEVEL doest not have any predictive power and no variance

Percantage of missing values

BLOCKID 100.000000

hc\_samples  2.196113

hc\_mean 2.196113

hc\_median   2.196113

hc\_stdev    2.196113

hc\_sample\_weight    2.196113

hc\_mortgage\_mean    2.097288

hc\_mortgage\_stdev   2.097288

hc\_mortgage\_sample\_weight   2.097288

hc\_mortgage\_samples 2.097288

df\_test .drop(columns=['BLOCKID','SUMLEVEL'],inplace=True) #SUMLEVEL doest not have any predictive power

Percantage of missing values

BLOCKID 42.857143

hc\_samples  1.061455

hc\_mean 1.061455

hc\_median   1.061455

hc\_stdev    1.061455

hc\_sample\_weight    1.061455

hc\_mortgage\_mean    0.980930

hc\_mortgage\_stdev   0.980930

hc\_mortgage\_sample\_weight   0.980930

hc\_mortgage\_samples 0.980930

# Imputing  missing values with mean

missing\_train\_cols=[]

for col in df\_train.columns:

    if df\_train[col].isna().sum() !=0:

         missing\_train\_cols.append(col)

print(missing\_train\_cols)

['rent\_mean', 'rent\_median', 'rent\_stdev', 'rent\_sample\_weight', 'rent\_samples', 'rent\_gt\_10', 'rent\_gt\_15', 'rent\_gt\_20', 'rent\_gt\_25', 'rent\_gt\_30', 'rent\_gt\_35', 'rent\_gt\_40', 'rent\_gt\_50', 'hi\_mean', 'hi\_median', 'hi\_stdev', 'hi\_sample\_weight', 'hi\_samples', 'family\_mean', 'family\_median', 'family\_stdev', 'family\_sample\_weight', 'family\_samples', 'hc\_mortgage\_mean', 'hc\_mortgage\_median', 'hc\_mortgage\_stdev', 'hc\_mortgage\_sample\_weight', 'hc\_mortgage\_samples', 'hc\_mean', 'hc\_median', 'hc\_stdev', 'hc\_samples', 'hc\_sample\_weight', 'home\_equity\_second\_mortgage', 'second\_mortgage', 'home\_equity', 'debt', 'second\_mortgage\_cdf', 'home\_equity\_cdf', 'debt\_cdf', 'hs\_degree', 'hs\_degree\_male', 'hs\_degree\_female', 'male\_age\_mean', 'male\_age\_median', 'male\_age\_stdev', 'male\_age\_sample\_weight', 'male\_age\_samples', 'female\_age\_mean', 'female\_age\_median', 'female\_age\_stdev', 'female\_age\_sample\_weight', 'female\_age\_samples', 'pct\_own', 'married', 'married\_snp', 'separated', 'divorced']

# Imputing  missing values with mean

missing\_test\_cols=[]

for col in df\_test.columns:

    if df\_test[col].isna().sum() !=0:

         missing\_test\_cols.append(col)

print(missing\_test\_cols)

['rent\_mean', 'rent\_median', 'rent\_stdev', 'rent\_sample\_weight', 'rent\_samples', 'rent\_gt\_10', 'rent\_gt\_15', 'rent\_gt\_20', 'rent\_gt\_25', 'rent\_gt\_30', 'rent\_gt\_35', 'rent\_gt\_40', 'rent\_gt\_50', 'hi\_mean', 'hi\_median', 'hi\_stdev', 'hi\_sample\_weight', 'hi\_samples', 'family\_mean', 'family\_median', 'family\_stdev', 'family\_sample\_weight', 'family\_samples', 'hc\_mortgage\_mean', 'hc\_mortgage\_median', 'hc\_mortgage\_stdev', 'hc\_mortgage\_sample\_weight', 'hc\_mortgage\_samples', 'hc\_mean', 'hc\_median', 'hc\_stdev', 'hc\_samples', 'hc\_sample\_weight', 'home\_equity\_second\_mortgage', 'second\_mortgage', 'home\_equity', 'debt', 'second\_mortgage\_cdf', 'home\_equity\_cdf', 'debt\_cdf', 'hs\_degree', 'hs\_degree\_male', 'hs\_degree\_female', 'male\_age\_mean', 'male\_age\_median', 'male\_age\_stdev', 'male\_age\_sample\_weight', 'male\_age\_samples', 'female\_age\_mean', 'female\_age\_median', 'female\_age\_stdev', 'female\_age\_sample\_weight', 'female\_age\_samples', 'pct\_own', 'married', 'married\_snp', 'separated', 'divorced']

# Missing cols are all numerical variables

for col in df\_train.columns:

    if col in (missing\_train\_cols):

        df\_train[col].replace(np.nan, df\_train[col].mean(),inplace=True)

# Missing cols are all numerical variables

for col in df\_test.columns:

    if col in (missing\_test\_cols):

        df\_test[col].replace(np.nan, df\_test[col].mean(),inplace=True)

df\_train.isna().sum().sum()

0

df\_test.isna().sum().sum()

0

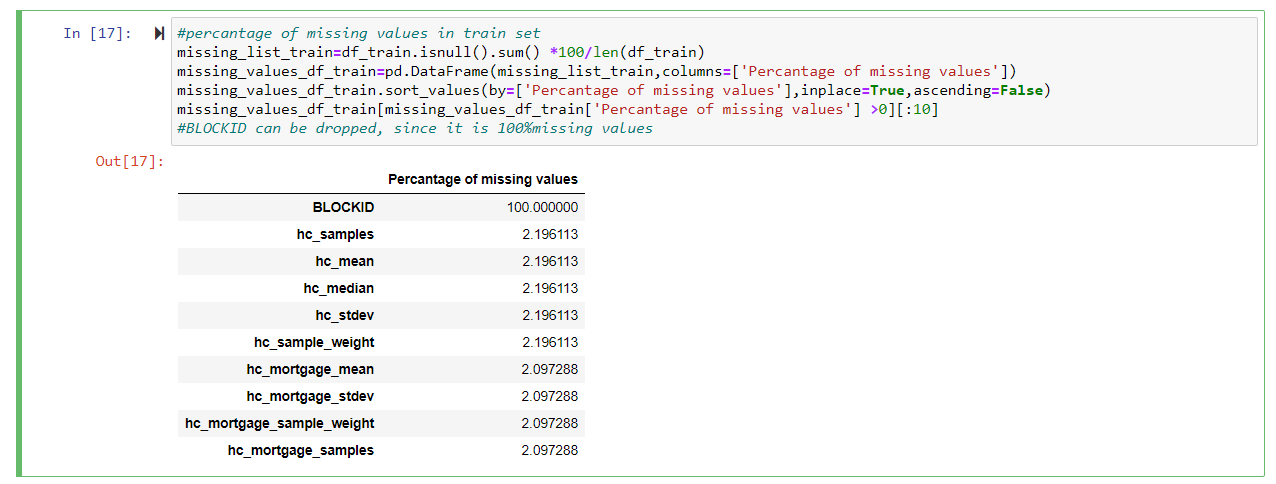


Figure 3 percantage of missing values in train set

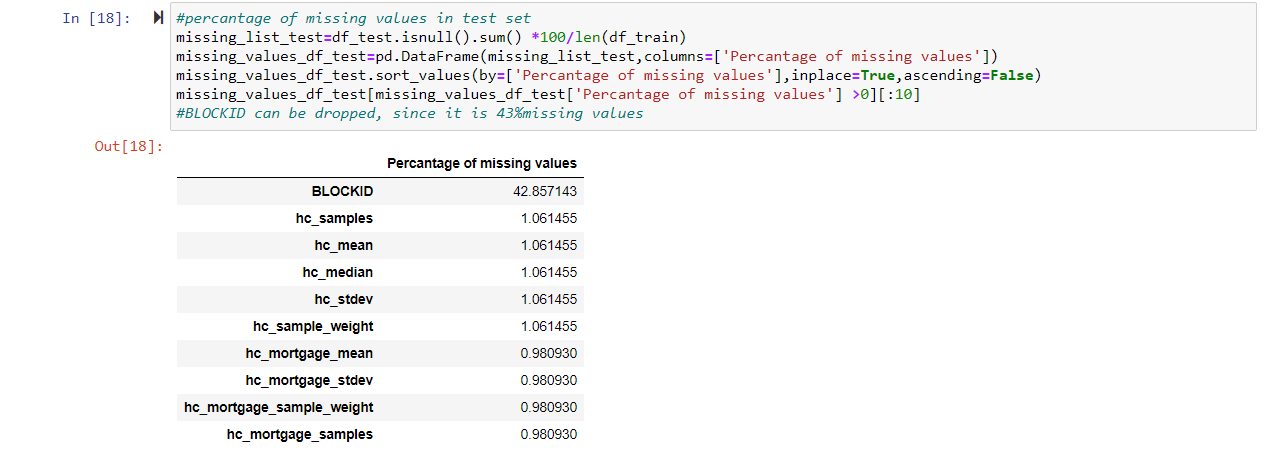


Figure 4 percantage of missing values in test set



Figure 5 Imputing missing values with mean

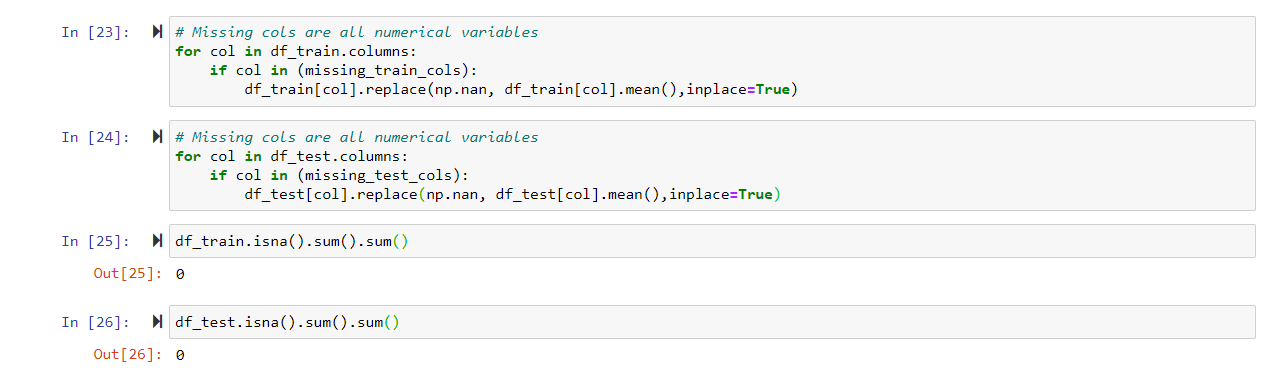


Figure 6 Missing cols are all numerical variables

## **Exploratory Data Analysis (EDA):**

1. Perform debt analysis. You may take the following steps:
   1. Explore the top 2,500 locations where the percentage of households with a second mortgage is the highest and percent ownership is above 10 percent. Visualize using geo-map. You may keep the upper limit for the percent of households with a second mortgage to 50 percent

from pandasql import sqldf

q1 = "select place,pct\_own,second\_mortgage,lat,lng from df\_train where pct\_own >0.10 and second\_mortgage <0.5 order by second\_mortgage DESC LIMIT 2500;"

pysqldf = lambda q: sqldf(q, globals())

df\_train\_location\_mort\_pct=pysqldf(q1)

df\_train\_location\_mort\_pct.head()

place   pct\_own second\_mortgage lat lng

0   Worcester City  0.20247 0.43363 42.254262   -71.800347

1   Harbor Hills    0.15618 0.31818 40.751809   -73.853582

2   Glen Burnie 0.22380 0.30212 39.127273   -76.635265

3   Egypt Lake-leto 0.11618 0.28972 28.029063   -82.495395

4   Lincolnwood 0.14228 0.28899 41.967289   -87.652434

import plotly.express as px

import plotly.graph\_objects as go

fig = go.Figure(data=go.Scattergeo(

    lat = df\_train\_location\_mort\_pct['lat'],

    lon = df\_train\_location\_mort\_pct['lng']),

    )

fig.update\_layout(

    geo=dict(

        scope = 'north america',

        showland = True,

        landcolor = "rgb(212, 212, 212)",

        subunitcolor = "rgb(255, 255, 255)",

        countrycolor = "rgb(255, 255, 255)",

        showlakes = True,

        lakecolor = "rgb(255, 255, 255)",

        showsubunits = True,

        showcountries = True,

        resolution = 50,

        projection = dict(

            type = 'conic conformal',

            rotation\_lon = -100

        ),

        lonaxis = dict(

            showgrid = True,

            gridwidth = 0.5,

            range= [ -140.0, -55.0 ],

            dtick = 5

        ),

        lataxis = dict (

            showgrid = True,

            gridwidth = 0.5,

            range= [ 20.0, 60.0 ],

            dtick = 5

        )

    ),

    title='Top 2,500 locations with second mortgage is the highest and percent ownership is above 10 percent')

fig.show()

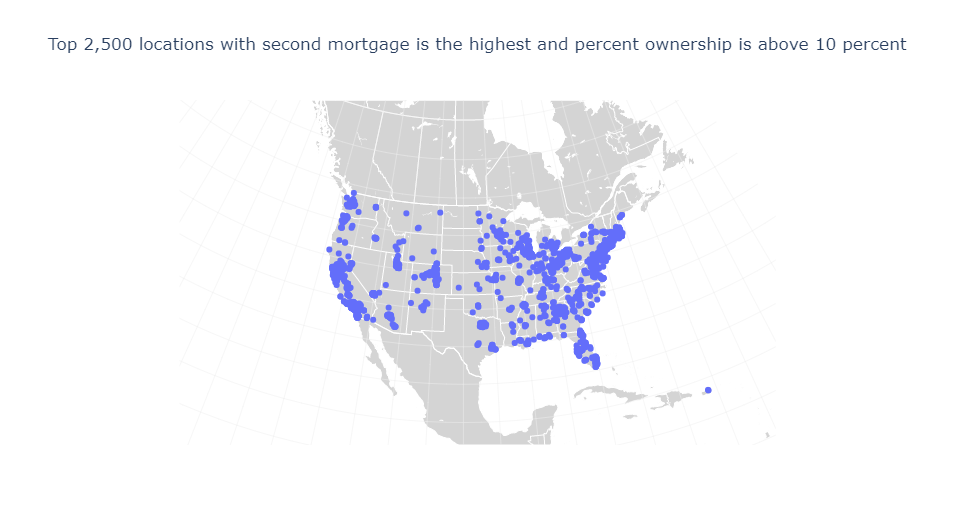


Figure 7 2500 locations

* 1. Use the following bad debt equation: Bad Debt = P (Second Mortgage ∩ Home Equity Loan) Bad Debt = second\_mortgage + home\_equity - home\_equity\_second\_mortgage

df\_train['bad\_debt']=df\_train['second\_mortgage']+df\_train['home\_equity']

* 1. Create pie charts to show overall debt and bad debt

df\_train['home\_equity\_second\_mortgage']

df\_train['bins'] = pd.cut(df\_train['bad\_debt'],bins=[0,0.10,1], labels=["less than 50%","50-100%"])

df\_train.groupby(['bins']).size().plot(kind='pie',subplots=True,startangle=90, autopct='%1.1f%%')

plt.axis('equal')

plt.show()

#df.plot.pie(subplots=True,figsize=(8, 3))

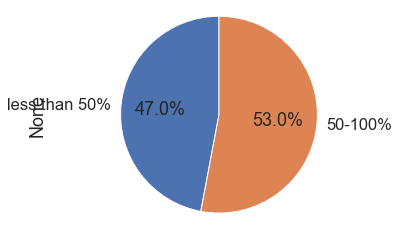


Figure 8 Overall Debt and bad Debt

* 1. Create Box and whisker plot and analyze the distribution for 2nd mortgage, home equity, good debt, and bad debt for different cities

cols=[]

df\_train.columns

Index(['COUNTYID', 'STATEID', 'state', 'state\_ab', 'city', 'place', 'type',

       'primary', 'zip\_code', 'area\_code', 'lat', 'lng', 'ALand', 'AWater',

       'pop', 'male\_pop', 'female\_pop', 'rent\_mean', 'rent\_median',

       'rent\_stdev', 'rent\_sample\_weight', 'rent\_samples', 'rent\_gt\_10',

       'rent\_gt\_15', 'rent\_gt\_20', 'rent\_gt\_25', 'rent\_gt\_30', 'rent\_gt\_35',

       'rent\_gt\_40', 'rent\_gt\_50', 'universe\_samples', 'used\_samples',

       'hi\_mean', 'hi\_median', 'hi\_stdev', 'hi\_sample\_weight', 'hi\_samples',

       'family\_mean', 'family\_median', 'family\_stdev', 'family\_sample\_weight',

       'family\_samples', 'hc\_mortgage\_mean', 'hc\_mortgage\_median',

       'hc\_mortgage\_stdev', 'hc\_mortgage\_sample\_weight', 'hc\_mortgage\_samples',

       'hc\_mean', 'hc\_median', 'hc\_stdev', 'hc\_samples', 'hc\_sample\_weight',

       'home\_equity\_second\_mortgage', 'second\_mortgage', 'home\_equity', 'debt',

       'second\_mortgage\_cdf', 'home\_equity\_cdf', 'debt\_cdf', 'hs\_degree',

       'hs\_degree\_male', 'hs\_degree\_female', 'male\_age\_mean',

       'male\_age\_median', 'male\_age\_stdev', 'male\_age\_sample\_weight',

       'male\_age\_samples', 'female\_age\_mean', 'female\_age\_median',

       'female\_age\_stdev', 'female\_age\_sample\_weight', 'female\_age\_samples',

       'pct\_own', 'married', 'married\_snp', 'separated', 'divorced',

       'bad\_debt', 'bins'],

      dtype='object')

#Taking Hamilton and Manhattan cities data

cols=['second\_mortgage','home\_equity','debt','bad\_debt']

df\_box\_hamilton=df\_train.loc[df\_train['city'] == 'Hamilton']

df\_box\_manhattan=df\_train.loc[df\_train['city'] == 'Manhattan']

df\_box\_city=pd.concat([df\_box\_hamilton,df\_box\_manhattan])

df\_box\_city.head(4)

COUNTYID    STATEID state   state\_ab    city    place   type    primary zip\_code    area\_code   ... female\_age\_stdev    female\_age\_sample\_weight    female\_age\_samples  pct\_own married married\_snp separated   divorced    bad\_debt    bins

UID

267822  53  36  New York    NY  Hamilton    Hamilton    City    tract   13346   315 ... 22.51276    685.33845   2618.0  0.79046 0.57851 0.01882 0.01240 0.08770 0.09408 less than 50%

263797  21  34  New Jersey  NJ  Hamilton    Yardville   City    tract   8610    609 ... 24.05831    732.58443   3124.0  0.64400 0.56377 0.01980 0.00990 0.04892 0.18071 50-100%

270979  17  39  Ohio    OH  Hamilton    Hamilton City   Village tract   45015   513 ... 22.66500    565.32725   2528.0  0.61278 0.47397 0.04419 0.02663 0.13741 0.15005 50-100%

259028  95  28  Mississippi MS  Hamilton    Hamilton    CDP tract   39746   662 ... 22.79602    483.01311   1954.0  0.83241 0.58678 0.01052 0.00000 0.11721 0.02130 less than 50%

4 rows × 79 columns

plt.figure(figsize=(10,5))

sns.boxplot(data=df\_box\_city,x='second\_mortgage', y='city',width=0.5,palette="Set3")

plt.show()

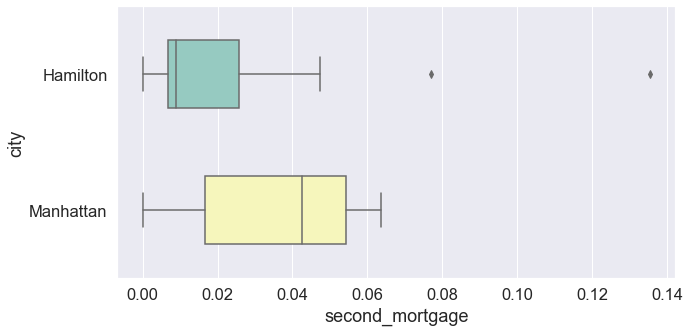


Figure 9 Second Mortgage

plt.figure(figsize=(10,5))

sns.boxplot(data=df\_box\_city,x='home\_equity', y='city',width=0.5,palette="Set3")

plt.show()

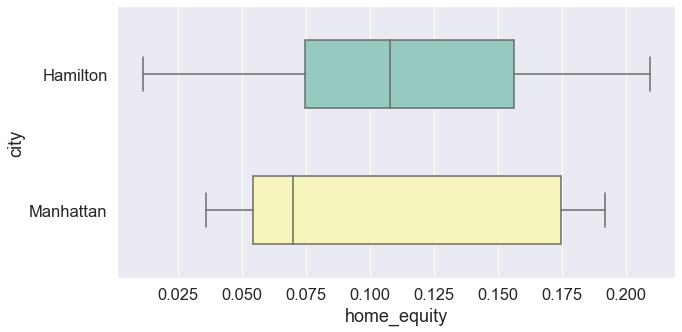


Figure 10 Home Equity

plt.figure(figsize=(10,5))

sns.boxplot(data=df\_box\_city,x='debt', y='city',width=0.5,palette="Set3")

plt.show()

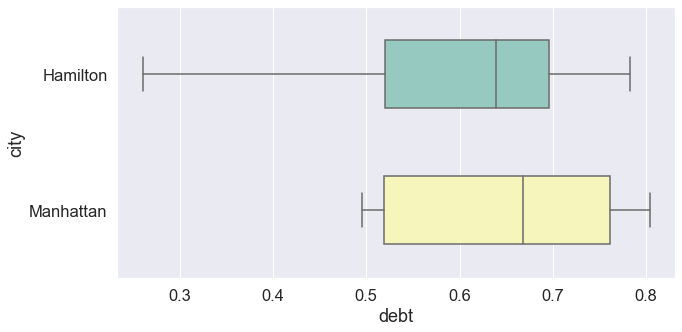


Figure 11 Debt

plt.figure(figsize=(10,5))

sns.boxplot(data=df\_box\_city,x='bad\_debt', y='city',width=0.5,palette="Set3")

plt.show()

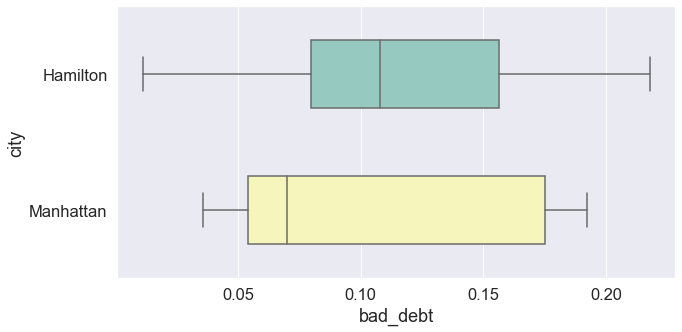


Figure 12 Bad Debt

Manhattan has higher metrics compared to Hamilton

* 1. Create a collated income distribution chart for family income, house hold income, and remaining income

 sns.distplot(df\_train['hi\_mean'])

plt.title('Household income distribution chart')

plt.show()

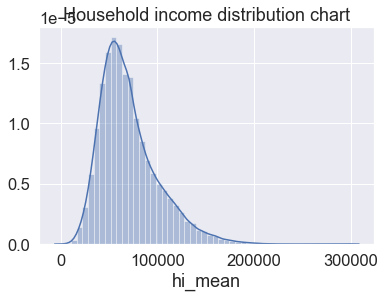


Figure 13 Hi Mean

sns.distplot(df\_train['family\_mean'])

plt.title('Family income distribution chart')

plt.show()

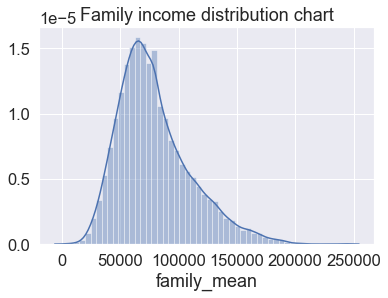


Figure 14 Family Mean

sns.distplot(df\_train['family\_mean']-df\_train['hi\_mean'])

plt.title('Remaining income distribution chart')

plt.show()

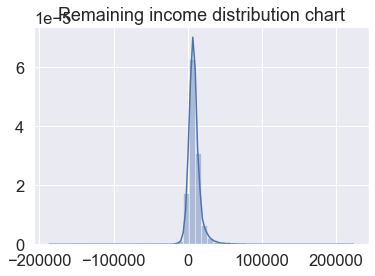


Figure 15 Remaining Income Distribution Chart

# **Project Task: Week 2**

## **Exploratory Data Analysis (EDA):**

1. Perform EDA and come out with insights into population density and age. You may have to derive new fields (make sure to weight averages for accurate measurements):

#plt.figure(figsize=(25,10))

fig,(ax1,ax2,ax3)=plt.subplots(3,1)

sns.distplot(df\_train['pop'],ax=ax1)

sns.distplot(df\_train['male\_pop'],ax=ax2)

sns.distplot(df\_train['female\_pop'],ax=ax3)

plt.subplots\_adjust(wspace=0.8,hspace=0.8)

plt.tight\_layout()

plt.show()

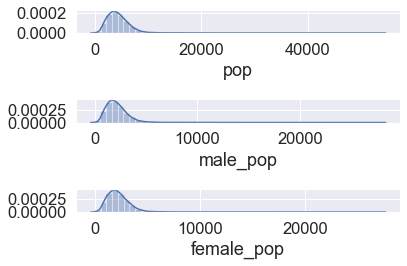


Figure 16 Pop, Male Pop, Family Pop

#plt.figure(figsize=(25,10))

fig,(ax1,ax2)=plt.subplots(2,1)

sns.distplot(df\_train['male\_age\_mean'],ax=ax1)

sns.distplot(df\_train['female\_age\_mean'],ax=ax2)

plt.subplots\_adjust(wspace=0.8,hspace=0.8)

plt.tight\_layout()

plt.show()

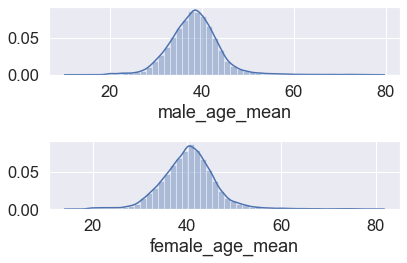


Figure 17 Male\_age\_mean, Female\_age\_mean

* 1. Use pop and ALand variables to create a new field called population density

df\_train['pop\_density']=df\_train['pop']/df\_train['ALand']

df\_test['pop\_density']=df\_test['pop']/df\_test['ALand']

sns.distplot(df\_train['pop\_density'])

plt.title('Population Density')

plt.show() # Very less density is noticed

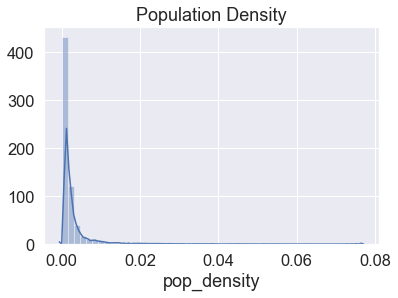


Figure 18 Pop Density

* 1. Use male\_age\_median, female\_age\_median, male\_pop, and female\_pop to create a new field called median age c) Visualize the findings using appropriate chart type

df\_train['age\_median']=(df\_train['male\_age\_median']+df\_train['female\_age\_median'])/2

df\_test['age\_median']=(df\_test['male\_age\_median']+df\_test['female\_age\_median'])/2

df\_train[['male\_age\_median','female\_age\_median','male\_pop','female\_pop','age\_median']].head()

male\_age\_median female\_age\_median   male\_pop    female\_pop  age\_median

UID

267822  44.00000    45.33333    2612    2618    44.666665

246444  32.00000    37.58333    1349    1284    34.791665

245683  40.83333    42.83333    3643    3238    41.833330

279653  48.91667    50.58333    1141    1559    49.750000

247218  22.41667    21.58333    2586    3051    22.000000

sns.distplot(df\_train['age\_median'])

plt.title('Median Age')

plt.show()

# Age of population is mostly between 20 and 60

# Majority are of age around 40

# Median age distribution has a gaussian distribution

# Some right skewness is noticed

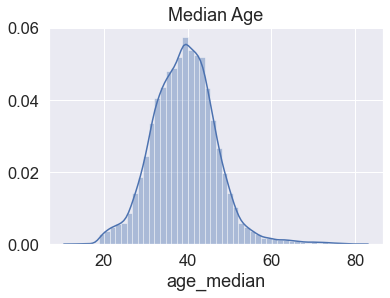


Figure 19 Age Median

sns.boxplot(df\_train['age\_median'])

plt.title('Population Density')

plt.show()

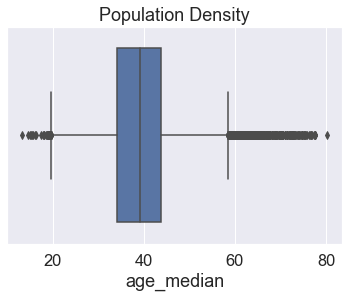


Figure 20 population Density

1. Create bins for population into a new variable by selecting appropriate class interval so that the number of categories don’t exceed 5 for the ease of analysis.

df\_train['pop'].describe()

count    27321.000000

mean      4316.032685

std       2169.226173

min          0.000000

25%       2885.000000

50%       4042.000000

75%       5430.000000

max      53812.000000

Name: pop, dtype: float64

df\_train['pop\_bins']=pd.cut(df\_train['pop'],bins=5,labels=['very low','low','medium','high','very high'])

df\_train[['pop','pop\_bins']]

pop    pop\_bins

UID

267822  5230    very low

246444  2633    very low

245683  6881    very low

279653  2700    very low

247218  5637    very low

... ... ...

279212  1847    very low

277856  4155    very low

233000  2829    very low

287425  11542   low

265371  3726    very low

27321 rows × 2 columns

df\_train['pop\_bins'].value\_counts()

very low     27058

low            246

medium           9

high             7

very high        1

Name: pop\_bins, dtype: int64

* 1. Analyze the married, separated, and divorced population for these population brackets

df\_train.groupby(by='pop\_bins')[['married','separated','divorced']].count()

married separated   divorced

pop\_bins

very low    27058   27058   27058

low 246 246 246

medium  9   9   9

high    7   7   7

very high   1   1   1

df\_train.groupby(by='pop\_bins')[['married','separated','divorced']].agg(["mean", "median"])

married separated   divorced

mean    median  mean    median  mean    median

pop\_bins

very low    0.507548    0.524680    0.019126    0.013650    0.100504    0.096020

low 0.584894    0.593135    0.015833    0.011195    0.075348    0.070045

medium  0.655737    0.618710    0.005003    0.004120    0.065927    0.064890

high    0.503359    0.335660    0.008141    0.002500    0.039030    0.010320

very high   0.734740    0.734740    0.004050    0.004050    0.030360    0.030360

1. Very high population group has more married people and less percantage of separated and divorced couples
2. Very low population groups, there are more divorced people
   1. Visualize using appropriate chart type

plt.figure(figsize=(10,5))

pop\_bin\_married=df\_train.groupby(by='pop\_bins')[['married','separated','divorced']].agg(["mean"])

pop\_bin\_married.plot(figsize=(20,8))

plt.legend(loc='best')

plt.show()

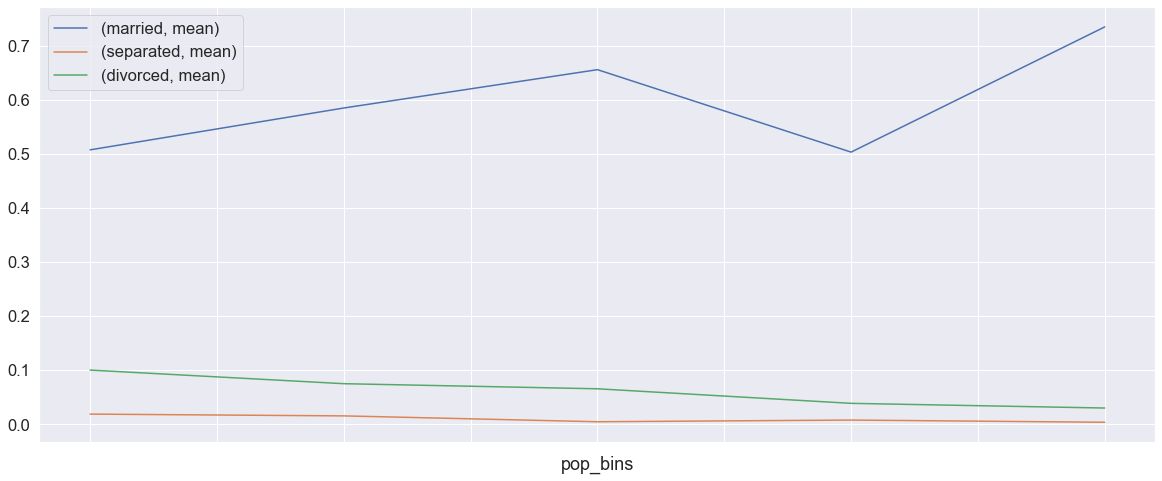


Figure 21 Pop Bins

1. Please detail your observations for rent as a percentage of income at an overall level, and for different states.

rent\_state\_mean=df\_train.groupby(by='state')['rent\_mean'].agg(["mean"])

rent\_state\_mean.head()

     mean

state

Alabama 774.004927

Alaska  1185.763570

Arizona 1097.753511

Arkansas    720.918575

California  1471.133857

income\_state\_mean=df\_train.groupby(by='state')['family\_mean'].agg(["mean"])

income\_state\_mean.head()

mean

state

Alabama 67030.064213

Alaska  92136.545109

Arizona 73328.238798

Arkansas    64765.377850

California  87655.470820

rent\_perc\_of\_income=rent\_state\_mean['mean']/income\_state\_mean['mean']

rent\_perc\_of\_income.head(10)

state

Alabama                 0.011547

Alaska                  0.012870

Arizona                 0.014970

Arkansas                0.011131

California              0.016783

Colorado                0.013529

Connecticut             0.012637

Delaware                0.012929

District of Columbia    0.013198

Florida                 0.015772

Name: mean, dtype: float64

#overall level rent as a percentage of income

sum(df\_train['rent\_mean'])/sum(df\_train['family\_mean'])

0.013358170721473864

1. Perform correlation analysis for all the relevant variables by creating a heatmap. Describe your findings.

df\_train.columns

Index(['COUNTYID', 'STATEID', 'state', 'state\_ab', 'city', 'place', 'type',

       'primary', 'zip\_code', 'area\_code', 'lat', 'lng', 'ALand', 'AWater',

       'pop', 'male\_pop', 'female\_pop', 'rent\_mean', 'rent\_median',

       'rent\_stdev', 'rent\_sample\_weight', 'rent\_samples', 'rent\_gt\_10',

       'rent\_gt\_15', 'rent\_gt\_20', 'rent\_gt\_25', 'rent\_gt\_30', 'rent\_gt\_35',

       'rent\_gt\_40', 'rent\_gt\_50', 'universe\_samples', 'used\_samples',

       'hi\_mean', 'hi\_median', 'hi\_stdev', 'hi\_sample\_weight', 'hi\_samples',

       'family\_mean', 'family\_median', 'family\_stdev', 'family\_sample\_weight',

       'family\_samples', 'hc\_mortgage\_mean', 'hc\_mortgage\_median',

       'hc\_mortgage\_stdev', 'hc\_mortgage\_sample\_weight', 'hc\_mortgage\_samples',

       'hc\_mean', 'hc\_median', 'hc\_stdev', 'hc\_samples', 'hc\_sample\_weight',

       'home\_equity\_second\_mortgage', 'second\_mortgage', 'home\_equity', 'debt',

       'second\_mortgage\_cdf', 'home\_equity\_cdf', 'debt\_cdf', 'hs\_degree',

       'hs\_degree\_male', 'hs\_degree\_female', 'male\_age\_mean',

       'male\_age\_median', 'male\_age\_stdev', 'male\_age\_sample\_weight',

       'male\_age\_samples', 'female\_age\_mean', 'female\_age\_median',

       'female\_age\_stdev', 'female\_age\_sample\_weight', 'female\_age\_samples',

       'pct\_own', 'married', 'married\_snp', 'separated', 'divorced',

       'bad\_debt', 'bins', 'pop\_density', 'age\_median', 'pop\_bins'],

      dtype='object')

cor=df\_train[['COUNTYID','STATEID','zip\_code','type','pop', 'family\_mean',

         'second\_mortgage', 'home\_equity', 'debt','hs\_degree',

           'age\_median','pct\_own', 'married','separated', 'divorced']].corr()

plt.figure(figsize=(20,10))

sns.heatmap(cor,annot=True,cmap='coolwarm')

plt.show()

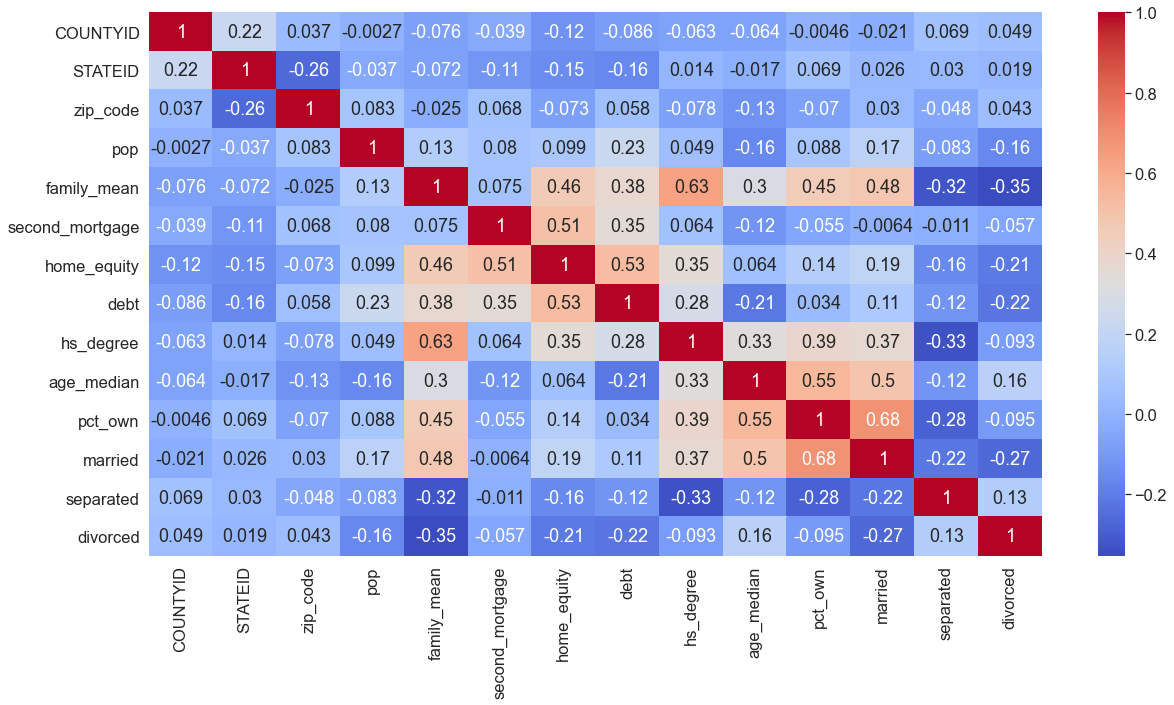


Figure 22 Color Warm Graph

1. High positive correaltion is noticed between pop, male\_pop and female\_pop
2. High positive correaltion is noticed between rent\_mean,hi\_mean, family\_mean,hc\_mean

# **Project Task: Week 3**

## **Data Pre-processing:**

1. The economic multivariate data has a significant number of measured variables. The goal is to find where the measured variables depend on a number of smaller unobserved common factors or latent variables.
2. Each variable is assumed to be dependent upon a linear combination of the common factors, and the coefficients are known as loadings. Each measured variable also includes a component due to independent random variability, known as “specific variance” because it is specific to one variable. Obtain the common factors and then plot the loadings. Use factor analysis to find latent variables in our dataset and gain insight into the linear relationships in the data. Following are the list of latent variables:
   * Highschool graduation rates
   * Median population age
   * Second mortgage statistics
   * Percent own
   * Bad debt expense

from sklearn.decomposition import FactorAnalysis

from factor\_analyzer import FactorAnalyzer

fa=FactorAnalyzer(n\_factors=5)

fa.fit\_transform(df\_train.select\_dtypes(exclude= ('object','category')))

fa.loadings\_

array([[-1.12589168e-01,  1.95646475e-02, -2.39331087e-02,

        -6.27632638e-02,  4.23474753e-02],

       [-1.10186763e-01,  1.33506222e-02,  2.79651240e-02,

        -1.49825865e-01,  1.10838809e-01],

       [-8.28678615e-02,  5.16372371e-02, -1.36451863e-01,

        -4.98918578e-02, -1.04024836e-01],

       [ 1.80961148e-02,  1.92013750e-02,  5.81329806e-03,

         2.64842730e-02, -6.12442435e-03],

       [ 9.02324721e-02, -9.72544319e-02, -6.54601343e-02,

        -1.33145904e-01, -1.48594605e-01],

       [-1.07335676e-02, -4.12376817e-02,  1.45853485e-01,

         8.80433269e-03,  1.08227570e-01],

       [-4.28796978e-02, -2.09780213e-02,  3.66726856e-02,

        -9.45597337e-02,  5.91380485e-02],

       [-2.44243078e-03, -1.53245408e-02, -2.68300811e-03,

        -4.52473017e-02,  2.37240639e-02],

       [ 7.92164317e-02,  9.57453306e-01, -8.71151628e-02,

        -6.59924032e-03, -3.97273193e-02],

       [ 7.39808192e-02,  9.18750508e-01, -1.08834838e-01,

        -2.79371593e-02, -3.93153650e-02],

       [ 8.06598882e-02,  9.47839205e-01, -6.08006510e-02,

         1.53627067e-02, -3.86977277e-02],

       [ 7.70052132e-01,  9.84675230e-03, -3.71249746e-02,

         1.14949043e-01, -1.23784684e-01],

       [ 7.18615886e-01,  6.24980329e-03, -4.59787405e-02,

         1.09109691e-01, -1.35301909e-01],

       [ 7.07647234e-01,  2.46625383e-02, -1.00860855e-02,

         1.04472483e-01,  7.72381182e-02],

       [-1.34545497e-01,  3.36809292e-01, -4.87894952e-01,

        -4.15446148e-02,  3.17608517e-01],

       [ 2.31079714e-01,  4.37729796e-01, -6.40209222e-01,

        -2.52311048e-02,  3.47216238e-01],

       [-4.52068115e-02,  3.51263839e-02,  3.07537010e-02,

         4.44793499e-01, -1.63273407e-01],

       [-2.50717063e-02,  1.70166786e-02,  4.57227294e-02,

         6.76083909e-01, -1.55256768e-01],

       [-3.90694433e-02, -1.67460881e-02,  8.13962719e-02,

         8.36389121e-01, -9.18259814e-02],

       [-5.14161938e-02, -3.57207142e-02,  1.10795163e-01,

         9.25123727e-01, -4.44866453e-02],

       [-6.08589974e-02, -4.41860619e-02,  1.35794016e-01,

         9.53019898e-01, -2.21548619e-02],

       [-4.57771131e-02, -5.25526131e-02,  1.41019879e-01,

         9.32702649e-01, -5.83303614e-07],

       [-4.19486024e-02, -5.90387643e-02,  1.28851779e-01,

         8.87316679e-01,  1.05894327e-02],

       [-2.47894610e-02, -7.29670562e-02,  9.41510472e-02,

         7.79023679e-01,  2.95352844e-02],

       [ 2.12258462e-01,  4.65992347e-01, -6.14495960e-01,

        -2.47660078e-02,  3.66644541e-01],

       [ 2.33057234e-01,  4.47057841e-01, -6.28263414e-01,

        -2.71547693e-02,  3.43419602e-01],

       [ 7.85157082e-01,  4.91249248e-02,  1.44540486e-01,

        -2.05217624e-01, -1.54523362e-01],

       [ 7.10324880e-01,  4.99730435e-02,  1.32239993e-01,

        -2.19171861e-01, -2.10505573e-01],

       [ 8.61780961e-01,  4.35044831e-02,  1.65839100e-01,

        -1.19850815e-01,  3.16733662e-02],

       [-2.23443275e-01,  8.46259563e-01, -4.61177203e-02,

         6.88599251e-02,  2.27742321e-01],

       [ 1.43837555e-01,  9.53197420e-01,  2.27887449e-02,

        -4.57890466e-02,  1.00796447e-01],

       [ 8.30286481e-01,  3.42025993e-02,  1.61106002e-01,

        -2.04570323e-01, -7.48710515e-02],

       [ 7.94476575e-01,  2.83818588e-02,  1.51219549e-01,

        -2.07681492e-01, -9.12497115e-02],

       [ 8.11481635e-01,  4.32314872e-02,  1.43645561e-01,

        -1.07778259e-01,  5.79540060e-02],

       [-3.37741910e-01,  8.64927632e-01,  3.58933714e-02,

         9.07183966e-02,  4.46327270e-02],

       [ 5.03572646e-02,  9.35515353e-01,  1.51475404e-01,

        -2.51501261e-02, -9.34471620e-02],

       [ 9.78242251e-01, -3.31490306e-02, -1.05261175e-01,

         4.50364249e-02,  7.37362095e-02],

       [ 9.59137197e-01, -3.90023025e-02, -1.20630340e-01,

         4.52591423e-02,  6.64877273e-02],

       [ 8.14087180e-01,  2.23057157e-03,  7.66518523e-02,

         2.02747442e-02,  1.27634825e-01],

       [-4.15353976e-01,  7.18339580e-01,  3.40068062e-01,

        -7.18402770e-02, -2.77950503e-01],

       [ 7.64912667e-02,  7.24900629e-01,  2.74193205e-01,

        -4.83952644e-02, -3.52988279e-01],

       [ 9.10390869e-01, -5.36541235e-02, -4.68641892e-02,

        -7.64183032e-04,  1.63870467e-01],

       [ 8.73011872e-01, -5.30302316e-02, -5.89943143e-02,

        -1.58989825e-03,  1.52417547e-01],

       [ 7.55087673e-01, -3.56133888e-03,  5.39542562e-02,

         4.24181456e-03,  2.58043485e-01],

       [-1.23469886e-01,  6.07438130e-01,  6.33039225e-01,

        -2.14798817e-02,  2.47973918e-01],

       [-3.42866894e-01,  5.59526285e-01,  5.88213014e-01,

        -2.51533511e-02,  2.18419887e-01],

       [-1.60867227e-01, -1.53062626e-02, -1.57026584e-01,

         1.09243758e-01, -6.61660849e-01],

       [-1.37306763e-01, -2.17250640e-02, -1.58408929e-01,

         1.25156192e-01, -6.71630804e-01],

       [ 2.45096179e-01, -2.54584602e-02, -2.66691424e-02,

         9.53148491e-02, -6.42510850e-01],

       [ 2.03988672e-01,  7.85172867e-02, -3.01656210e-01,

         2.28379435e-02, -6.29223308e-01],

       [ 1.08926097e-01, -6.34332391e-02, -3.36565249e-02,

        -9.49480501e-02,  6.81473864e-01],

       [-2.63787621e-01, -6.43281062e-03, -3.58792182e-02,

        -9.37962445e-02,  6.47817007e-01],

       [-2.15717047e-01, -7.36588959e-02,  3.50113235e-01,

        -1.95201601e-02,  6.36783767e-01],

       [ 3.94306145e-01,  6.09565687e-02,  2.55337866e-01,

        -2.20362098e-01, -1.84248083e-01],

       [ 4.07877887e-01,  6.27256518e-02,  2.23926911e-01,

        -2.10028736e-01, -1.71989226e-01],

       [ 3.53156876e-01,  5.36715662e-02,  2.69603574e-01,

        -2.16933222e-01, -1.80072075e-01],

       [ 2.33537257e-01, -4.91732977e-02,  8.14450780e-01,

         9.36688878e-02,  3.27131925e-01],

       [ 2.40298207e-01, -3.38140108e-02,  8.31496983e-01,

         7.52417611e-02,  2.46323609e-01],

       [-6.71839513e-02,  6.58504548e-02,  5.86207690e-01,

         8.72955237e-02,  9.12541356e-02],

       [ 5.59835543e-02,  8.17918702e-01, -1.78458350e-01,

        -1.55949440e-02, -3.34299742e-02],

       [ 7.16426395e-02,  9.23428543e-01, -1.07142695e-01,

        -2.78635384e-02, -4.35991121e-02],

       [ 1.92496948e-01, -4.75870397e-02,  8.03173211e-01,

         1.43492718e-01,  3.33862162e-01],

       [ 1.87644427e-01, -3.29941031e-02,  8.58024482e-01,

         1.31329950e-01,  2.55679717e-01],

       [-1.02263659e-01,  6.03984273e-02,  4.72982262e-01,

         7.36848412e-02,  1.12273909e-01],

       [ 6.14776642e-02,  8.77962748e-01, -1.50410286e-01,

         2.20991027e-02, -4.17158178e-02],

       [ 7.83728215e-02,  9.54508794e-01, -5.91095907e-02,

         1.64800924e-02, -4.32590992e-02],

       [-3.24381861e-02,  1.11167164e-01,  7.84467393e-01,

        -4.37718579e-02, -2.80931224e-01],

       [ 1.76682387e-01,  1.90494239e-01,  5.61405491e-01,

        -1.20746164e-01, -1.32570787e-01],

       [-6.37386686e-02, -7.03047914e-02, -2.68934059e-01,

         1.28589788e-01,  1.88507846e-01],

       [-1.56051274e-01, -7.08033933e-02, -1.45964499e-01,

         1.24253729e-01,  1.46293106e-01],

       [-3.56716296e-01, -5.29910744e-02,  1.47771604e-01,

         2.87196190e-02,  1.13159574e-01],

       [ 2.42173828e-01, -2.86199107e-02, -3.25958359e-02,

         1.05027812e-01, -6.55406062e-01],

       [ 3.50196763e-01, -1.05016417e-02, -3.95274130e-01,

         5.92876756e-02,  2.91651806e-01],

       [ 2.25671539e-01, -3.42672770e-02,  8.92876617e-01,

         1.12426808e-01,  2.67065195e-01]])

# **Project Task: Week 4**

## **Data Modeling :**

1. Build a linear Regression model to predict the total monthly expenditure for home mortgages loan. Please refer ‘deplotment\_RE.xlsx’. Column hc\_mortgage\_mean is predicted variable. This is the mean monthly mortgage and owner costs of specified geographical location. Note: Exclude loans from prediction model which have NaN (Not a Number) values for hc\_mortgage\_mean.

df\_train.columns

Index(['COUNTYID', 'STATEID', 'state', 'state\_ab', 'city', 'place', 'type',

       'primary', 'zip\_code', 'area\_code', 'lat', 'lng', 'ALand', 'AWater',

       'pop', 'male\_pop', 'female\_pop', 'rent\_mean', 'rent\_median',

       'rent\_stdev', 'rent\_sample\_weight', 'rent\_samples', 'rent\_gt\_10',

       'rent\_gt\_15', 'rent\_gt\_20', 'rent\_gt\_25', 'rent\_gt\_30', 'rent\_gt\_35',

       'rent\_gt\_40', 'rent\_gt\_50', 'universe\_samples', 'used\_samples',

       'hi\_mean', 'hi\_median', 'hi\_stdev', 'hi\_sample\_weight', 'hi\_samples',

       'family\_mean', 'family\_median', 'family\_stdev', 'family\_sample\_weight',

       'family\_samples', 'hc\_mortgage\_mean', 'hc\_mortgage\_median',

       'hc\_mortgage\_stdev', 'hc\_mortgage\_sample\_weight', 'hc\_mortgage\_samples',

       'hc\_mean', 'hc\_median', 'hc\_stdev', 'hc\_samples', 'hc\_sample\_weight',

       'home\_equity\_second\_mortgage', 'second\_mortgage', 'home\_equity', 'debt',

       'second\_mortgage\_cdf', 'home\_equity\_cdf', 'debt\_cdf', 'hs\_degree',

       'hs\_degree\_male', 'hs\_degree\_female', 'male\_age\_mean',

       'male\_age\_median', 'male\_age\_stdev', 'male\_age\_sample\_weight',

       'male\_age\_samples', 'female\_age\_mean', 'female\_age\_median',

       'female\_age\_stdev', 'female\_age\_sample\_weight', 'female\_age\_samples',

       'pct\_own', 'married', 'married\_snp', 'separated', 'divorced',

       'bad\_debt', 'bins', 'pop\_density', 'age\_median', 'pop\_bins'],

      dtype='object')

df\_train['type'].unique()

type\_dict={'type':{'City':1,

                   'Urban':2,

                   'Town':3,

                   'CDP':4,

                   'Village':5,

                   'Borough':6}

          }

df\_train.replace(type\_dict,inplace=True)

df\_train['type'].unique()

array([1, 2, 3, 4, 5, 6], dtype=int64)

df\_test.replace(type\_dict,inplace=True)

df\_test['type'].unique()

array([4, 1, 6, 3, 5, 2], dtype=int64)

feature\_cols=['COUNTYID','STATEID','zip\_code','type','pop', 'family\_mean',

         'second\_mortgage', 'home\_equity', 'debt','hs\_degree',

           'age\_median','pct\_own', 'married','separated', 'divorced']

x\_train=df\_train[feature\_cols]

y\_train=df\_train['hc\_mortgage\_mean']

x\_test=df\_test[feature\_cols]

y\_test=df\_test['hc\_mortgage\_mean']

from sklearn.preprocessing import StandardScaler

from sklearn.linear\_model import LinearRegression

from sklearn.metrics import r2\_score, mean\_absolute\_error,mean\_squared\_error,accuracy\_score

x\_train.head()

COUNTYID    STATEID zip\_code    type    pop family\_mean second\_mortgage home\_equity debt    hs\_degree   age\_median  pct\_own married separated   divorced

UID

267822  53  36  13346   1   5230    67994.14790 0.02077 0.08919 0.52963 0.89288 44.666665   0.79046 0.57851 0.01240 0.08770

246444  141 18  46616   1   2633    50670.10337 0.02222 0.04274 0.60855 0.90487 34.791665   0.52483 0.34886 0.01426 0.09030

245683  63  18  46122   1   6881    95262.51431 0.00000 0.09512 0.73484 0.94288 41.833330   0.85331 0.64745 0.01607 0.10657

279653  127 72  927 2   2700    56401.68133 0.01086 0.01086 0.52714 0.91500 49.750000   0.65037 0.47257 0.02021 0.10106

247218  161 20  66502   1   5637    54053.42396 0.05426 0.05426 0.51938 1.00000 22.000000   0.13046 0.12356 0.00000 0.03109

sc=StandardScaler()

x\_train\_scaled=sc.fit\_transform(x\_train)

x\_test\_scaled=sc.fit\_transform(x\_test)

* 1. Run a model at a Nation level. If the accuracy levels and R square are not satisfactory proceed to below step.

linereg=LinearRegression()

linereg.fit(x\_train\_scaled,y\_train)

LinearRegression()

y\_pred=linereg.predict(x\_test\_scaled)

print("Overall R2 score of linear regression model", r2\_score(y\_test,y\_pred))

print("Overall RMSE of linear regression model", np.sqrt(mean\_squared\_error(y\_test,y\_pred)))

Overall R2 score of linear regression model 0.7348210754610929

Overall RMSE of linear regression model 323.1018894984635

The Accuracy and R2 score are good, but still will investigate the model performance at state level

* 1. Run another model at State level. There are 52 states in USA.

state=df\_train['STATEID'].unique()

state[0:5]

#Picking a few iDs 20,1,45,6

array([36, 18, 72, 20,  1], dtype=int64)

for i in [20,1,45]:

    print("State ID-",i)

    x\_train\_nation=df\_train[df\_train['COUNTYID']==i][feature\_cols]

    y\_train\_nation=df\_train[df\_train['COUNTYID']==i]['hc\_mortgage\_mean']

    x\_test\_nation=df\_test[df\_test['COUNTYID']==i][feature\_cols]

    y\_test\_nation=df\_test[df\_test['COUNTYID']==i]['hc\_mortgage\_mean']

    x\_train\_scaled\_nation=sc.fit\_transform(x\_train\_nation)

    x\_test\_scaled\_nation=sc.fit\_transform(x\_test\_nation)

    linereg.fit(x\_train\_scaled\_nation,y\_train\_nation)

    y\_pred\_nation=linereg.predict(x\_test\_scaled\_nation)

    print("Overall R2 score of linear regression model for state,",i,":-" ,r2\_score(y\_test\_nation,y\_pred\_nation))

    print("Overall RMSE of linear regression model for state,",i,":-" ,np.sqrt(mean\_squared\_error(y\_test\_nation,y\_pred\_nation)))

    print("\n")

# To check the residuals

residuals=y\_test-y\_pred

residuals

UID

255504    281.969088

252676    -69.935775

276314    190.761969

248614   -157.290627

286865     -9.887017

             ...

238088    -67.541646

242811    -41.578757

250127   -127.427569

241096   -330.820475

287763    217.760642

Name: hc\_mortgage\_mean, Length: 11709, dtype: float64

plt.hist(residuals) # Normal distribution of residuals

(array([6.000e+00, 3.000e+00, 2.900e+01, 7.670e+02, 7.823e+03, 2.716e+03,

        3.010e+02, 4.900e+01, 1.200e+01, 3.000e+00]),

 array([-2515.04284233, -1982.92661329, -1450.81038425,  -918.69415521,

         -386.57792617,   145.53830287,   677.65453191,  1209.77076095,

         1741.88698999,  2274.00321903,  2806.11944807]),

 <a list of 10 Patch objects>)

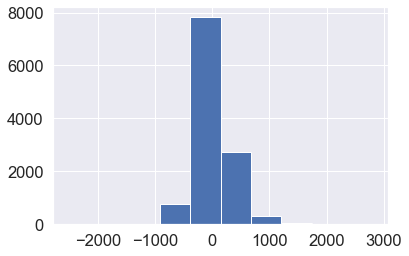


Figure 23 Normal Distribution of Residuals

 sns.distplot(residuals)

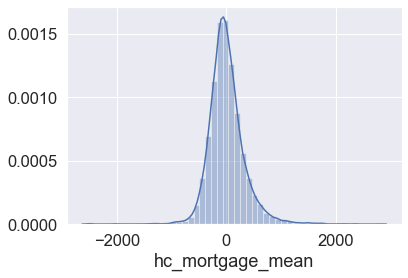


Figure 24 HC Mortgage Mean

plt.scatter(residuals,y\_pred) # Same variance and residuals does not have correlation with predictor

# Independance of residuals

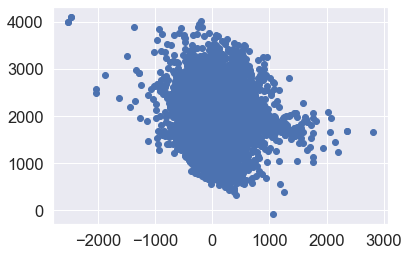


Figure 25 Independance of residuals

* 1. Keep below considerations while building a linear regression model. Data Modeling :
     1. Variables should have significant impact on predicting Monthly mortgage and owner costs
     2. Utilize all predictor variable to start with initial hypothesis
     3. R square of 60 percent and above should be achieved
     4. Ensure Multi-collinearity does not exist in dependent variables
     5. Test if predicted variable is normally distributed

## **Data Reporting:**

1. Create a dashboard in tableau by choosing appropriate chart types and metrics useful for the business. The dashboard must entail the following:
   1. Box plot of distribution of average rent by type of place (village, urban, town, etc.).

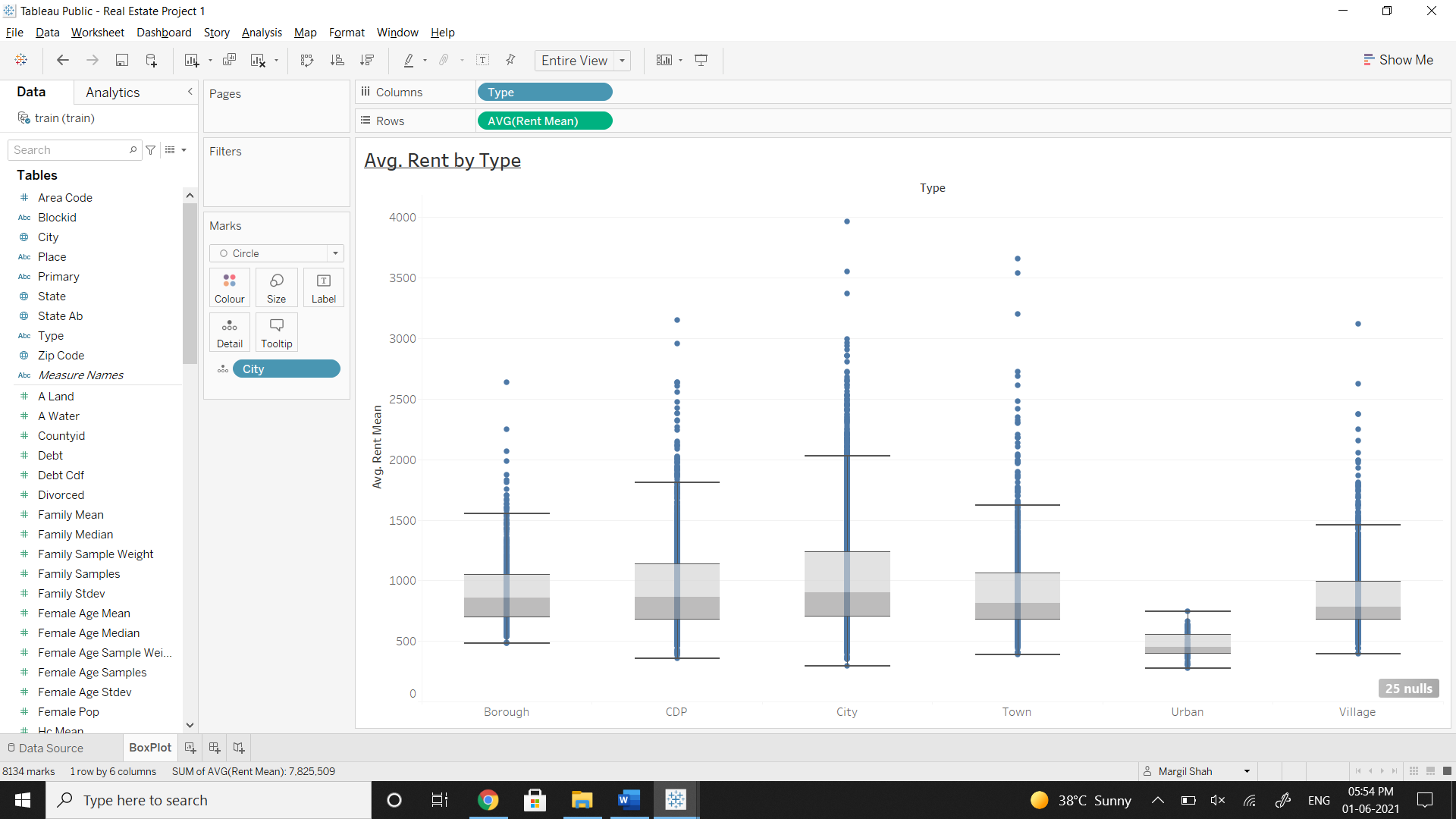


Figure 26 Box Plot Analysis

* 1. Pie charts to show overall debt and bad debt.

Calculated Field:

Bad Debt = ([Second Mortgage] + [Home Equity]) -[Home Equity Second Mortgage]

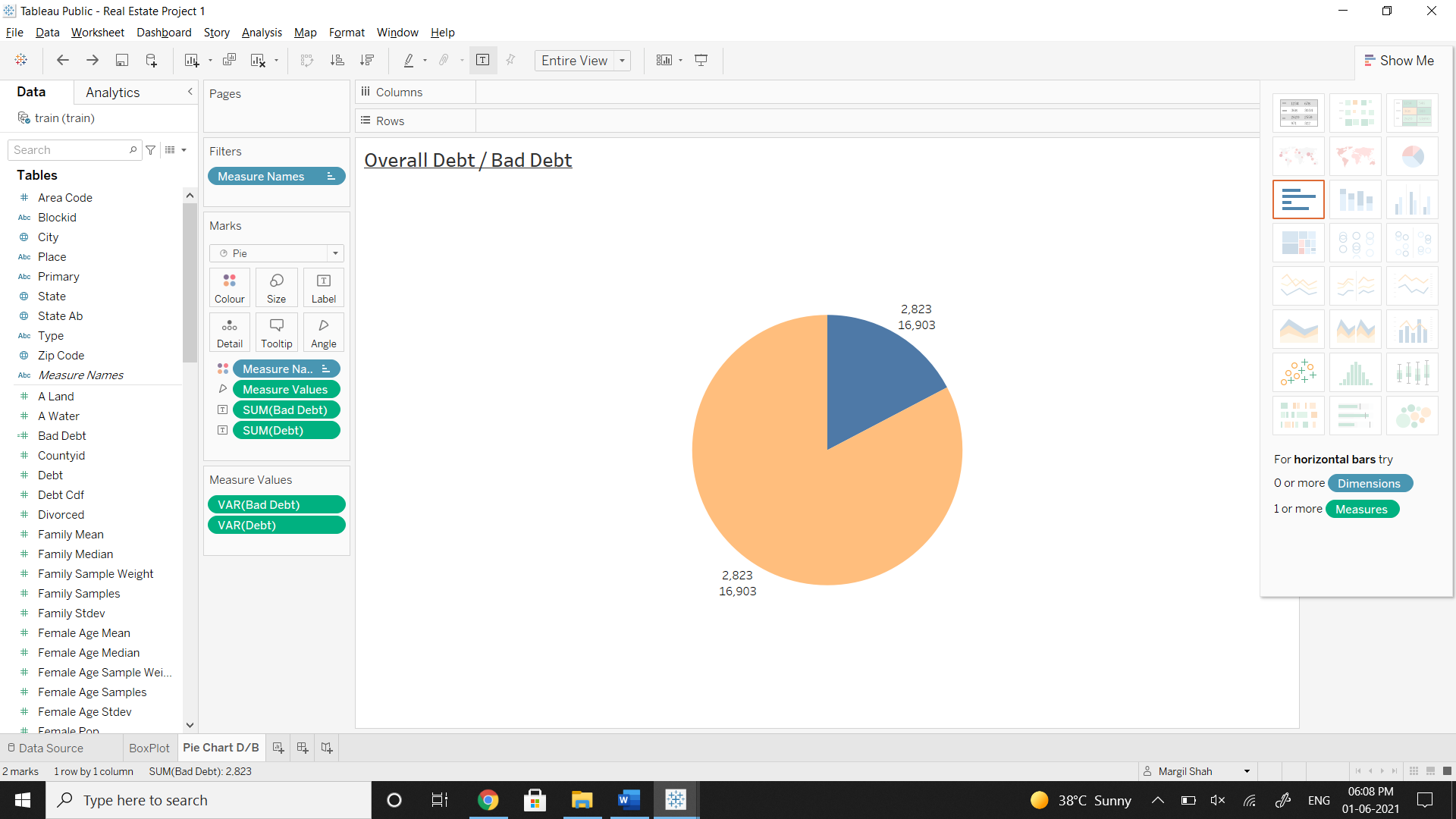


Figure 27 Pie Charts to show overall debt and bad debt

* 1. Explore the top 2,500 locations where the percentage of households with a second mortgage is the highest and percent ownership is above 10 percent. Visualize using geo-map.

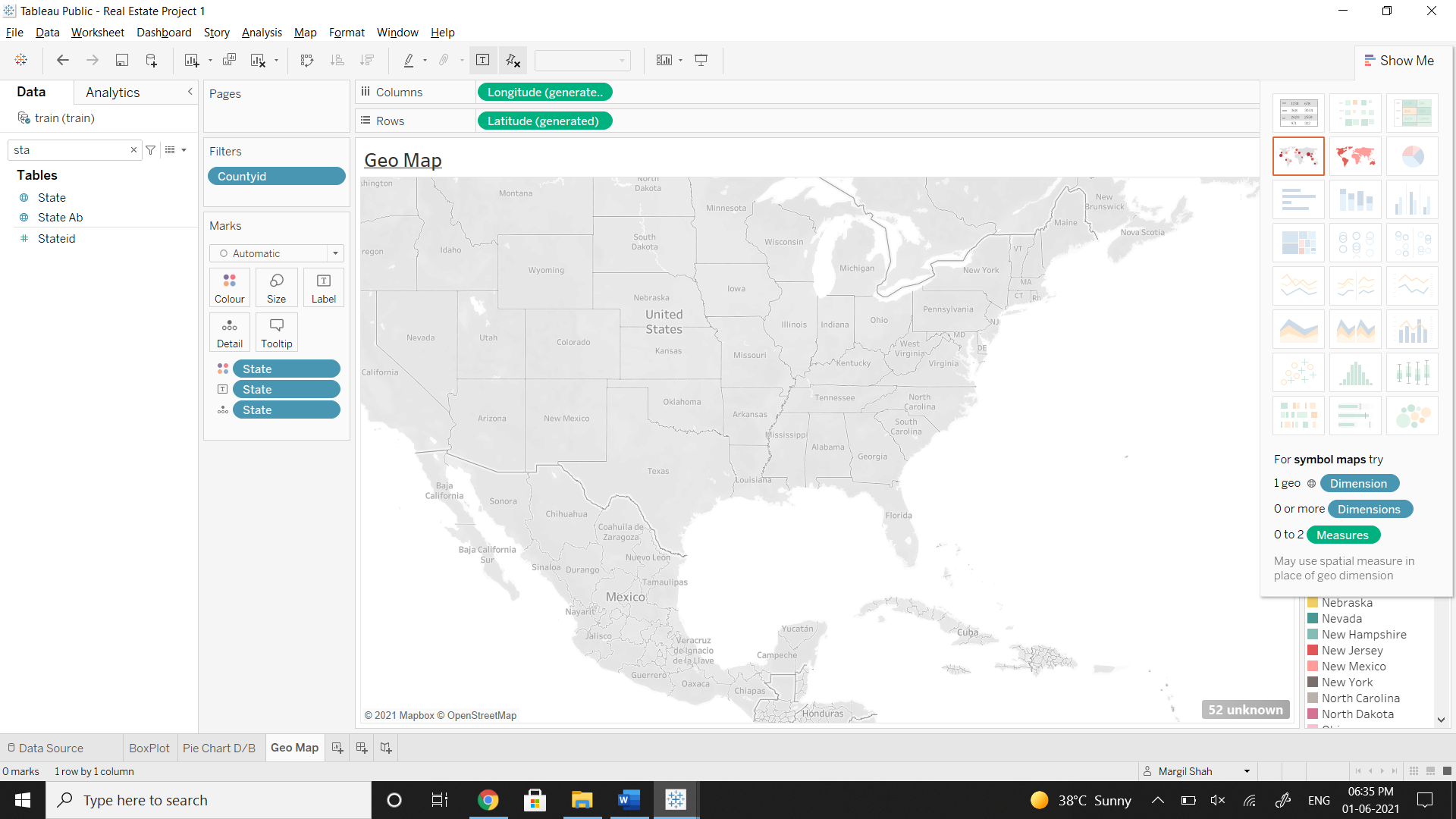


Figure 28 Visualize Geo Map

* 1. Heat map for correlation matrix.

**Calculated Field**

Corr = CORR([Family Mean],[Rent Mean])

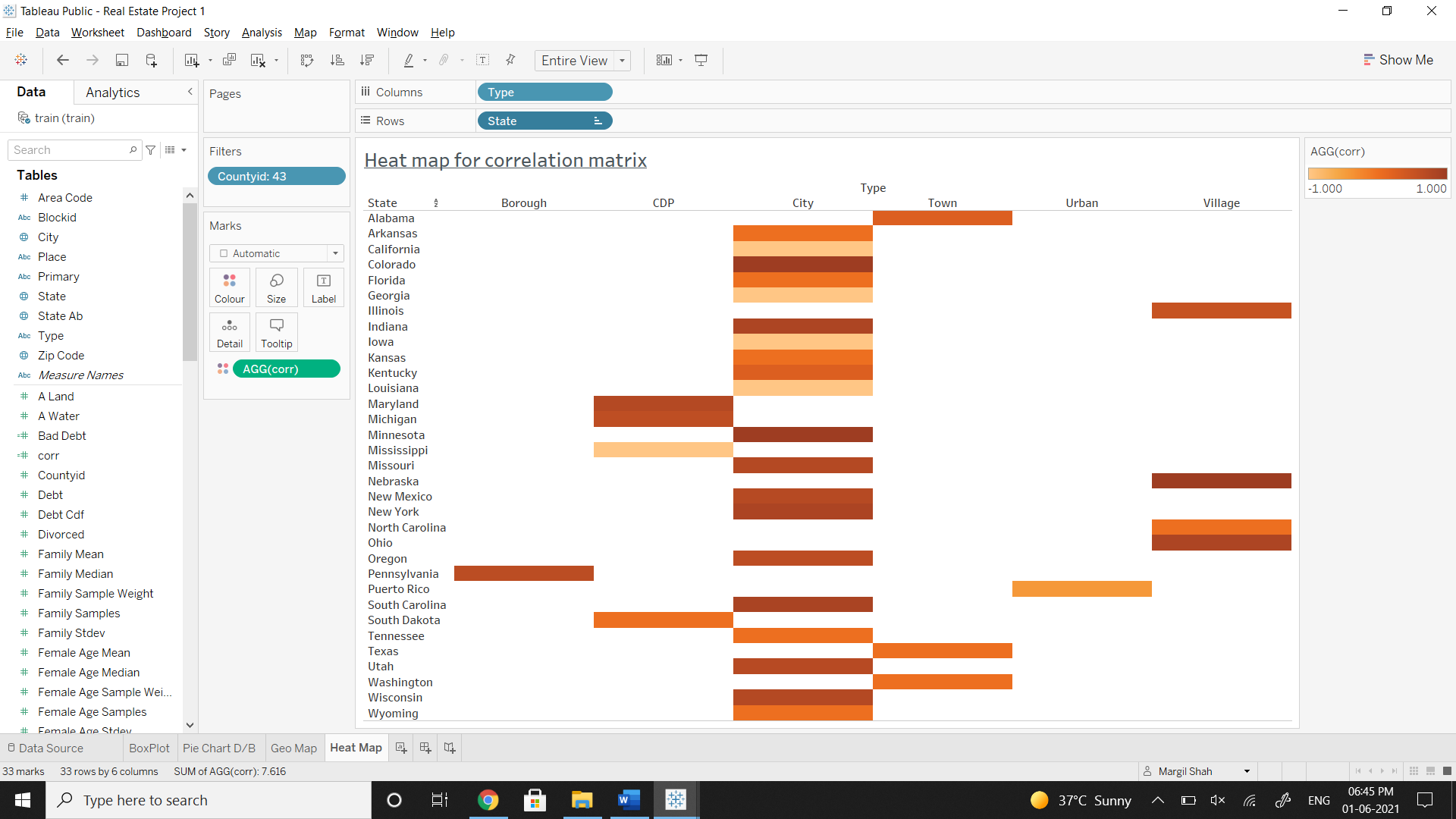


Figure 29 Heat Map for Correlation Matrix

* 1. Pie chart to show the population distribution across different types of places (village, urban, town etc.)

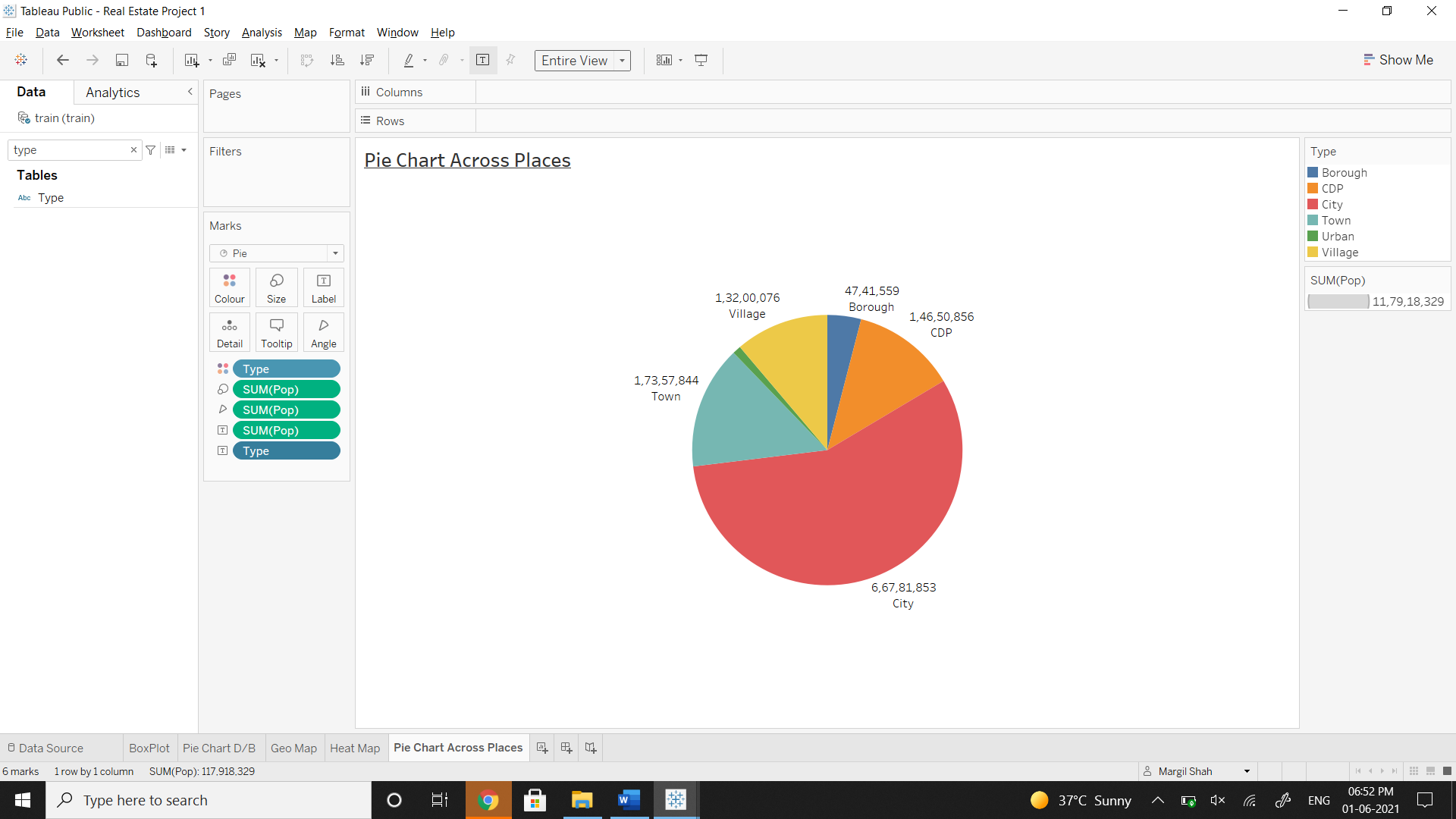


Figure 30 Pie chart to show the population distribution across different types of places (village, urban, town etc.)