Week 1 : Data Import & Preparation

In [1]:
```
# Importing Libraries
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
# Importing module
import warnings
# Warnings filter.
warnings.filterwarnings('ignore')
# Import the necessary libraries
import plotly.offline as pyo
import plotly.graph_objs as go
# Set notebook mode to work in offline
pyo.init_notebook_mode()
```

In [2]:
```
train=pd.read_csv("train.csv")
test=pd.read_csv("test.csv")
```

Descriptive Analysis

In [3]:
```
train.head()
```

Out[3]:

	UID	BLOCKID	SUMLEVEL	COUNTYID	STATEID	state	state_ab	city	place	type	...	female_age_mean	female_age_median
0	267822	NaN	140	53	36	New York	NY	Hamilton	Hamilton	City	...	44.48629	45.33333
1	246444	NaN	140	141	18	Indiana	IN	South Bend	Roseland	City	...	36.48391	37.58333
2	245683	NaN	140	63	18	Indiana	IN	Danville	Danville	City	...	42.15810	42.83333
3	279653	NaN	140	127	72	Puerto Rico	PR	San Juan	Guaynabo	Urban	...	47.77526	50.58333
4	247218	NaN	140	161	20	Kansas	KS	Manhattan	Manhattan City	City	...	24.17693	21.58333

5 rows × 80 columns

In [4]:
```
test.head()
```

Out[4]:

	UID	BLOCKID	SUMLEVEL	COUNTYID	STATEID	state	state_ab	city	place	type	...	female_age_mean	female_age_me
0	255504	NaN	140	163	26	Michigan	MI	Detroit	Dearborn Heights City	CDP	...	34.78682	33.7
1	252676	NaN	140	1	23	Maine	ME	Auburn	Auburn City	City	...	44.23451	46.6
2	276314	NaN	140	15	42	Pennsylvania	PA	Pine City	Millerton	Borough	...	41.62426	44.5
3	248614	NaN	140	231	21	Kentucky	KY	Monticello	Monticello City	City	...	44.81200	48.0
4	286865	NaN	140	355	48	Texas	TX	Corpus Christi	Edroy	Town	...	40.66618	42.6

5 rows × 80 columns

In [5]:
```
train.describe()
```

Out[5]:

	UID	BLOCKID	SUMLEVEL	COUNTYID	STATEID	zip_code	area_code	lat	lng	ALand	...
count	27321.000000	0.0	27321.0	27321.000000	27321.000000	27321.000000	27321.000000	27321.000000	27321.000000	2.732100e+04	...
mean	257331.996303	NaN	140.0	85.646426	28.271806	50081.999524	596.507668	37.508813	-91.288394	1.295106e+08	...
std	21343.859725	NaN	0.0	98.333097	16.392846	29558.115660	232.497482	5.588268	16.343816	1.275531e+09	...
min	220342.000000	NaN	140.0	1.000000	1.000000	602.000000	201.000000	17.929085	-165.453872	4.113400e+04	...
25%	238816.000000	NaN	140.0	29.000000	13.000000	26554.000000	405.000000	33.899064	-97.816067	1.799408e+06	...
50%	257220.000000	NaN	140.0	63.000000	28.000000	47715.000000	614.000000	38.755183	-86.554374	4.866940e+06	...
75%	275818.000000	NaN	140.0	109.000000	42.000000	77093.000000	801.000000	41.380606	-79.782503	3.359820e+07	...
max	294334.000000	NaN	140.0	840.000000	72.000000	99925.000000	989.000000	67.074017	-65.379332	1.039510e+11	...

8 rows × 74 columns

In [6]:
```
test.describe()
```

Out[6]:

	UID	BLOCKID	SUMLEVEL	COUNTYID	STATEID	zip_code	area_code	lat	lng	ALand	...
count	11709.000000	0.0	11709.0	11709.000000	11709.000000	11709.000000	11709.000000	11709.000000	11709.000000	1.170900e+04	...
mean	257525.004783	NaN	140.0	85.710650	28.489196	50123.418396	593.598514	37.405491	-91.340229	1.095500e+08	...
std	21466.372658	NaN	0.0	99.304334	16.607262	29775.134038	232.074263	5.625904	16.407818	7.624940e+08	...
min	220336.000000	NaN	140.0	1.000000	1.000000	601.000000	201.000000	17.965835	-166.770979	8.299000e+03	...
25%	238819.000000	NaN	140.0	29.000000	13.000000	25570.000000	404.000000	33.919813	-97.816561	1.718660e+06	...
50%	257651.000000	NaN	140.0	61.000000	28.000000	47362.000000	612.000000	38.618093	-86.643344	4.835000e+06	...
75%	276300.000000	NaN	140.0	109.000000	42.000000	77406.000000	787.000000	41.232973	-79.697311	3.204540e+07	...
max	294333.000000	NaN	140.0	810.000000	72.000000	99929.000000	989.000000	64.804269	-65.695344	5.520166e+10	...

8 rows × 74 columns

In [7]:
```
train.columns
```

Out[7]:
```
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')
```

In [8]:
```
test.columns
```

Out[8]:
```
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')
```

In [9]:
```
# UID is unique userID value in the train and test dataset. So an index can be created from the UID feature
train.set_index(keys=['UID'],inplace=True)#Set the DataFrame index using existing columns.
test.set_index(keys=['UID'],inplace=True)
```

In [10]:
```
# Handling Missing value
train.isnull().sum()/len(train)*100
```

Out[10]:
```
BLOCKID        100.000000
SUMLEVEL         0.000000
COUNTYID         0.000000
STATEID          0.000000
state            0.000000
                   ...
pct_own          0.980930
married          0.699096
married_snp      0.699096
separated        0.699096
divorced         0.699096
Length: 79, dtype: float64
```

In [11]:
```
train=train.drop(['BLOCKID','SUMLEVEL'],axis=1)
```

In [12]:
```
test.isnull().sum()/len(test)*100
```

Out[12]:
```
BLOCKID        100.000000
SUMLEVEL         0.000000
COUNTYID         0.000000
STATEID          0.000000
state            0.000000
                   ...
pct_own          1.041934
married          0.717397
married_snp      0.717397
separated        0.717397
divorced         0.717397
Length: 79, dtype: float64
```

In [13]:
```
test=test.drop(['BLOCKID','SUMLEVEL'],axis=1)
```

In [14]:
```
# Imputing  missing values with mean
missing_train_cols=[]
for col in train.columns:
    if 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_2
0', 'rent_gt_25', 'rent_gt_30', 'rent_gt_35', 'rent_gt_40', 'rent_gt_50', 'hi_mean', 'hi_median', 'hi_stdev', 'hi_sampl
e_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_mea
n', '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_femal
e', 'male_age_mean', 'male_age_median', 'male_age_stdev', 'male_age_sample_weight', 'male_age_samples', 'female_age_mea
n', 'female_age_median', 'female_age_stdev', 'female_age_sample_weight', 'female_age_samples', 'pct_own', 'married', 'm
arried_snp', 'separated', 'divorced']
```

In [15]:
```
missing_test_cols=[]
for col in test.columns:
    if 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_2
0', 'rent_gt_25', 'rent_gt_30', 'rent_gt_35', 'rent_gt_40', 'rent_gt_50', 'hi_mean', 'hi_median', 'hi_stdev', 'hi_sampl
e_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_mea
n', '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_femal
e', 'male_age_mean', 'male_age_median', 'male_age_stdev', 'male_age_sample_weight', 'male_age_samples', 'female_age_mea
n', 'female_age_median', 'female_age_stdev', 'female_age_sample_weight', 'female_age_samples', 'pct_own', 'married', 'm
arried_snp', 'separated', 'divorced']
```

In [16]:
```
# Missing cols are all numerical variables
for col in train.columns:
    if col in (missing_train_cols):
        train[col].replace(np.nan,train[col].mean(),inplace=True)
```

In [17]:
```
for col in test.columns:
    if col in (missing_test_cols):
        test[col].replace(np.nan,test[col].mean(),inplace=True)
```

In [18]:
```
train.isna().sum().sum()
```

Out[18]: 0

In [19]:
```
test.isna().sum().sum()
```

Out[19]: 0

Week 1 Exploratory Data Analysis

In [20]:
```
df = train[train['pct_own']>0.1]
df.shape
```

Out[20]: (26565, 77)

In [21]:
```
df = df.sort_values(by='second_mortgage',ascending=False)
```

In [22]:
```
pd.set_option('display.max_columns', None)
df.head()
```

Out[22]:

UID	COUNTYID	STATEID	state	state_ab	city	place	type	primary	zip_code	area_code	lat	lng	ALand
289712	147	51	Virginia	VA	Farmville	Farmville	Town	tract	23901	434	37.297357	-78.396452	413391.0
251185	27	25	Massachusetts	MA	Worcester	Worcester City	City	tract	1610	508	42.254262	-71.800347	797165.0
269323	81	36	New York	NY	Corona	Harbor Hills	City	tract	11368	718	40.751809	-73.853582	169666.0
251324	3	24	Maryland	MD	Glen Burnie	Glen Burnie	CDP	tract	21061	410	39.127273	-76.635265	1110282.0
235788	57	12	Florida	FL	Tampa	Egypt Lake-leto	City	tract	33614	813	28.029063	-82.495395	2050906.0

In [23]:
```
top_2500_second_mortgage_pctown_10 = df.head(2500)
top_2500_second_mortgage_pctown_10
```

Out[23]:

	COUNTYID	STATEID	state	state_ab	city	place	type	primary	zip_code	area_code	lat	lng	AL
UID													
289712	147	51	Virginia	VA	Farmville	Farmville	Town	tract	23901	434	37.297357	-78.396452	4133
251185	27	25	Massachusetts	MA	Worcester	Worcester City	City	tract	1610	508	42.254262	-71.800347	7971
269323	81	36	New York	NY	Corona	Harbor Hills	City	tract	11368	718	40.751809	-73.853582	1696
251324	3	24	Maryland	MD	Glen Burnie	Glen Burnie	CDP	tract	21061	410	39.127273	-76.635265	11102
235788	57	12	Florida	FL	Tampa	Egypt Lake-leto	City	tract	33614	813	28.029063	-82.495395	20509
...	...	...	...	...	...	...	...	...	...	...	...	...	
229021	67	6	California	CA	Carmichael	Carmichael	City	tract	95608	916	38.617256	-121.337317	24534
261444	183	37	North Carolina	NC	Raleigh	Raleigh City	Village	tract	27606	919	35.757135	-78.704288	40143
225977	37	6	California	CA	Marina Del Rey	Marina Del Rey	City	tract	90292	310	33.983204	-118.466139	9021
251433	5	24	Maryland	MD	Baltimore	Lochearn	CDP	tract	21208	410	39.353095	-76.733315	19135
230480	77	6	California	CA	Manteca	Manteca City	City	tract	95336	209	37.732143	-121.242902	997167

2500 rows × 77 columns

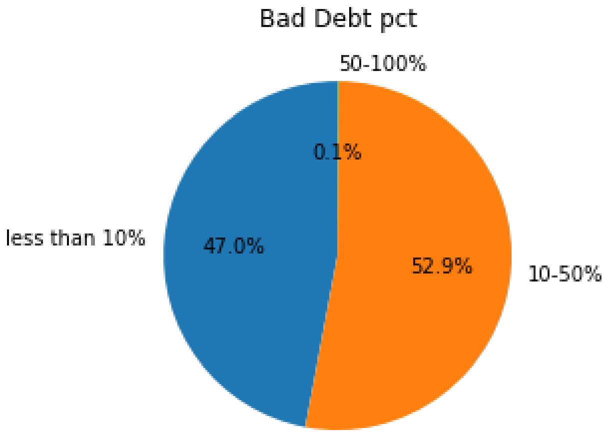
In [24]:
```
import plotly.express as px
import plotly.graph_objects as go
```

In [25]:
```
# Visualization 1 (Geo-Map):
fig = go.Figure(data=go.Scattergeo(
    lat = top_2500_second_mortgage_pctown_10['lat'],
    lon = top_2500_second_mortgage_pctown_10['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()
```

In [26]:
```
train['bad_debt']=train['second_mortgage']+train['home_equity']-train['home_equity_second_mortgage']
```

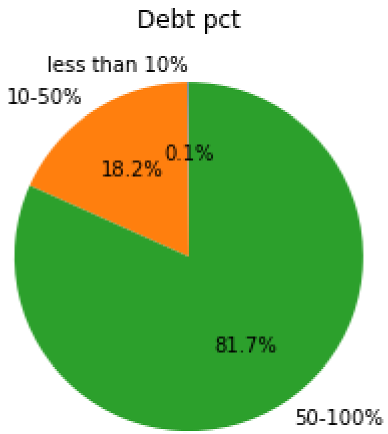
In [27]:
```
# Visualization 2:
train['bins_bad_debt'] = pd.cut(train['bad_debt'],bins=[0,0.1,.5,1], labels=["less than 10%","10-50%","50-100%"])
train.groupby(['bins_bad_debt']).size().plot(kind='pie',subplots=True,startangle=90, autopct='%1.1f%%')
plt.title('Bad Debt pct')
plt.ylabel("")

plt.show()
```



In [28]:
```
# Visualization 3:
train['bins_debt'] = pd.cut(train['debt'],bins=[0,0.1,.5,1], labels=["less than 10%","10-50%","50-100%"])
train.groupby(['bins_debt']).size().plot(kind='pie',subplots=True,startangle=90, autopct='%1.1f%%')
plt.title('Debt pct')
plt.ylabel("")

plt.show()
```

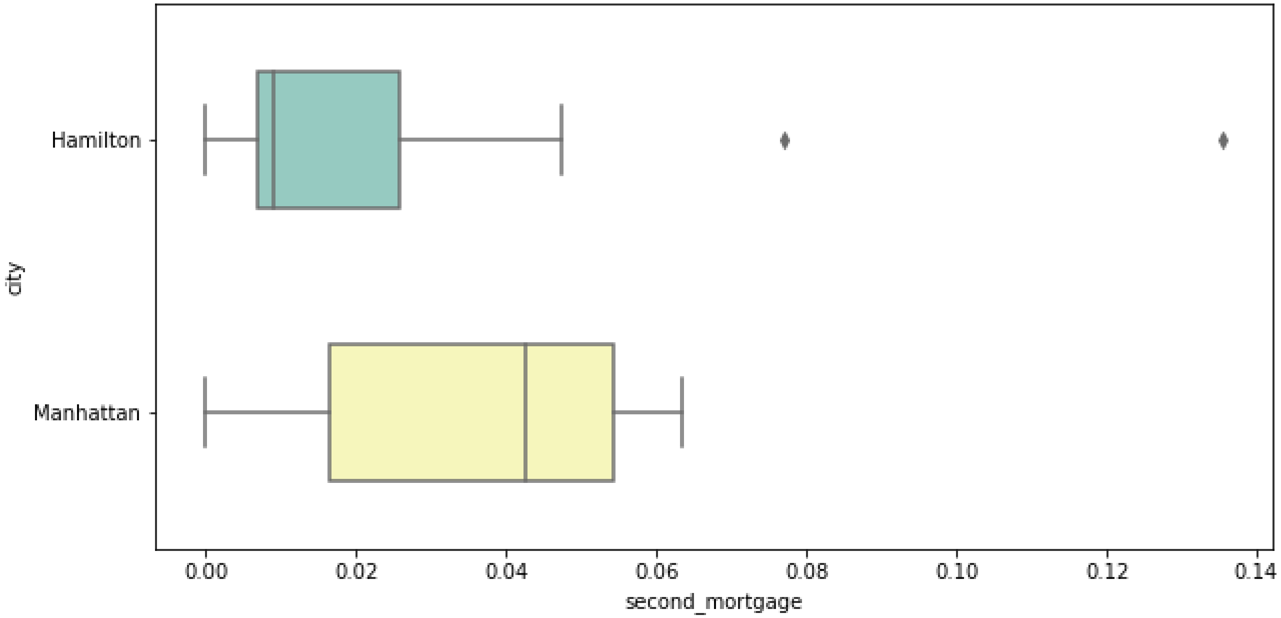


In [29]:
```
cols=['second_mortgage','home_equity','debt','bad_debt']
df_box_hamilton=train.loc[train['city'] == 'Hamilton']
df_box_manhattan=train.loc[train['city'] == 'Manhattan']
df_box_city=pd.concat([df_box_hamilton,df_box_manhattan])
df_box_city.head(4)
```

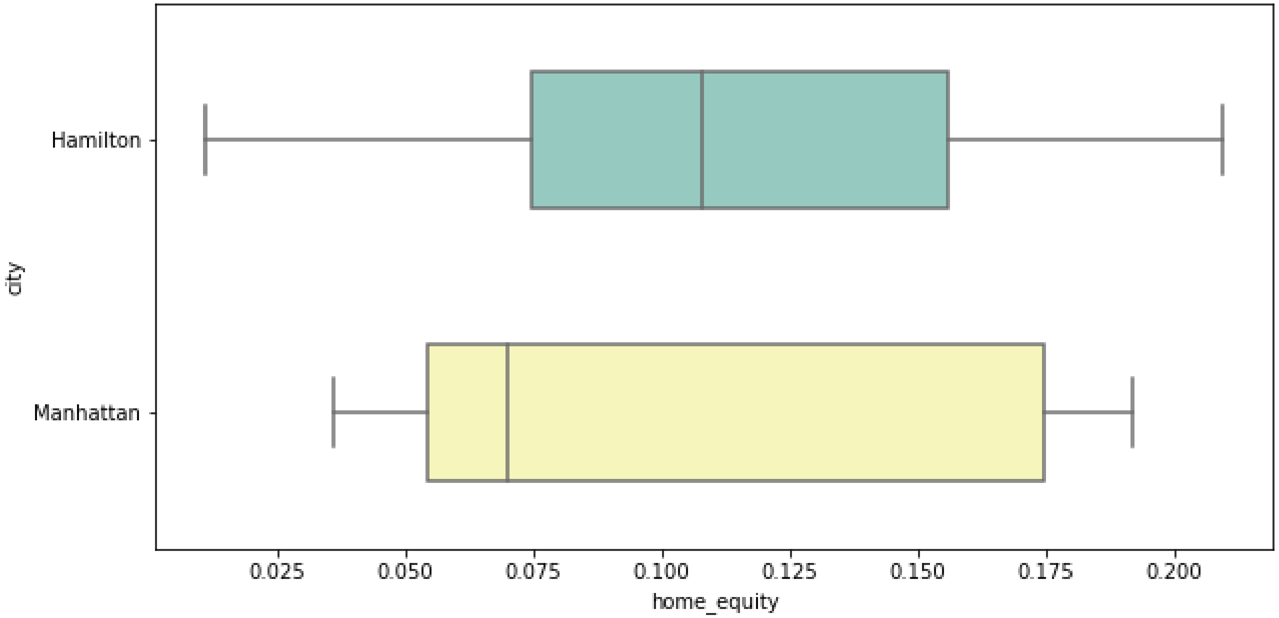
Out[29]:

	COUNTYID	STATEID	state	state_ab	city	place	type	primary	zip_code	area_code	lat	lng	ALand	A
UID														
267822	53	36	New York	NY	Hamilton	Hamilton	City	tract	13346	315	42.840812	-75.501524	202183361.0	16
263797	21	34	New Jersey	NJ	Hamilton	Yardville	City	tract	8610	609	40.206266	-74.675274	4623635.0	
270979	17	39	Ohio	OH	Hamilton	Hamilton City	Village	tract	45015	513	39.364028	-84.570717	3598447.0	1
259028	95	28	Mississippi	MS	Hamilton	Hamilton	CDP	tract	39746	662	33.759514	-88.377770	235934245.0	7

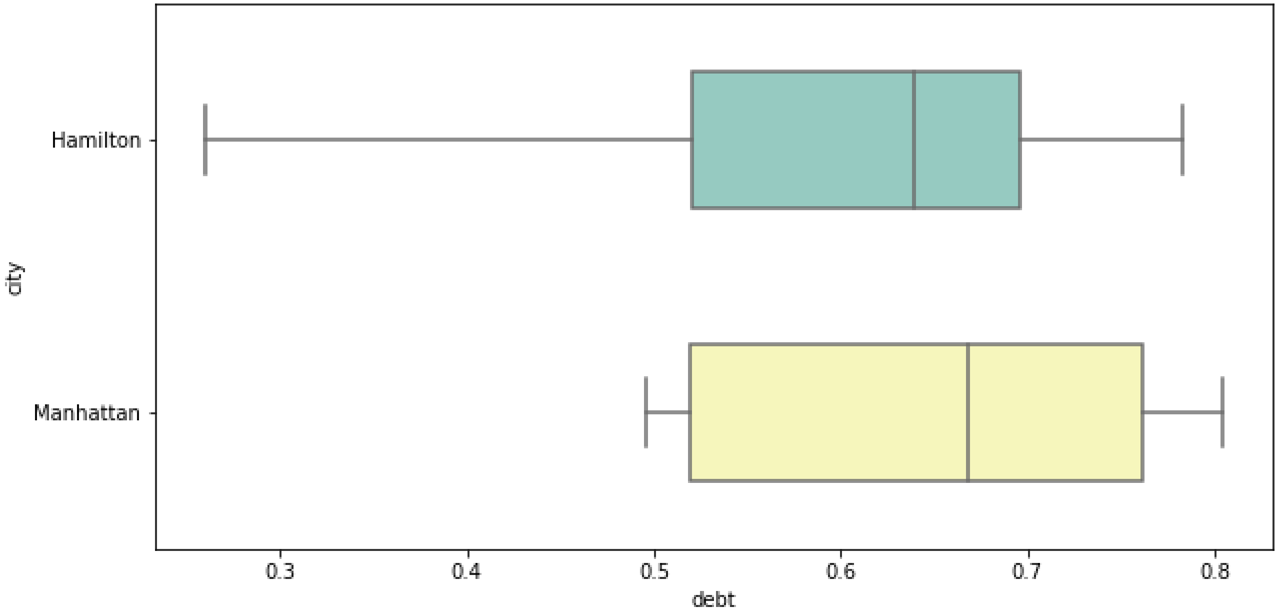
In [30]:
```
# Visualization 4:
plt.figure(figsize=(10,5))
sns.boxplot(data=df_box_city,x='second_mortgage', y='city',width=0.5,palette="Set3")
plt.show()
```



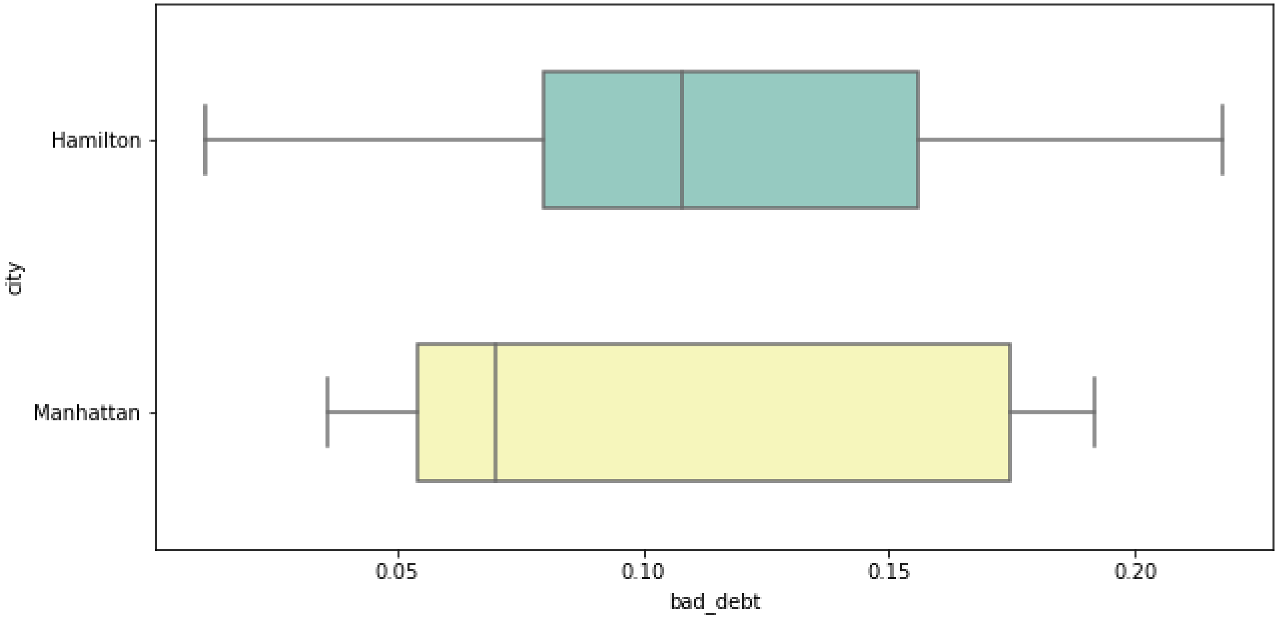
In [31]:
```
# Visualization 5:
plt.figure(figsize=(10,5))
sns.boxplot(data=df_box_city,x='home_equity', y='city',width=0.5,palette="Set3")
plt.show()
```



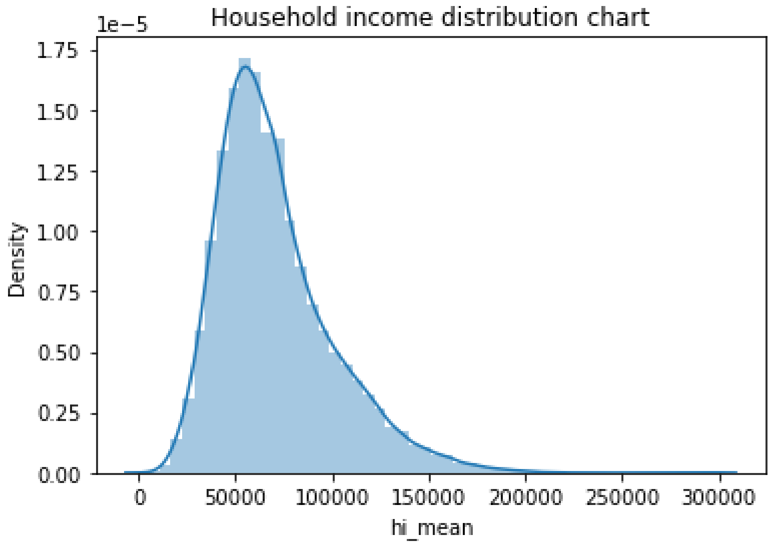
In [32]:
```
# Visualization 6:
plt.figure(figsize=(10,5))
sns.boxplot(data=df_box_city,x='debt', y='city',width=0.5,palette="Set3")
plt.show()
```



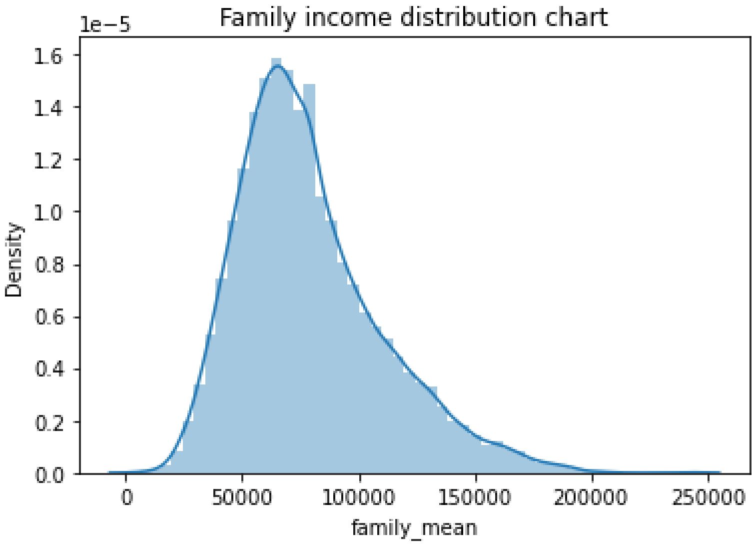
In [33]:
```
# Visualization 7:
plt.figure(figsize=(10,5))
sns.boxplot(data=df_box_city,x='bad_debt', y='city',width=0.5,palette="Set3")
plt.show()
```



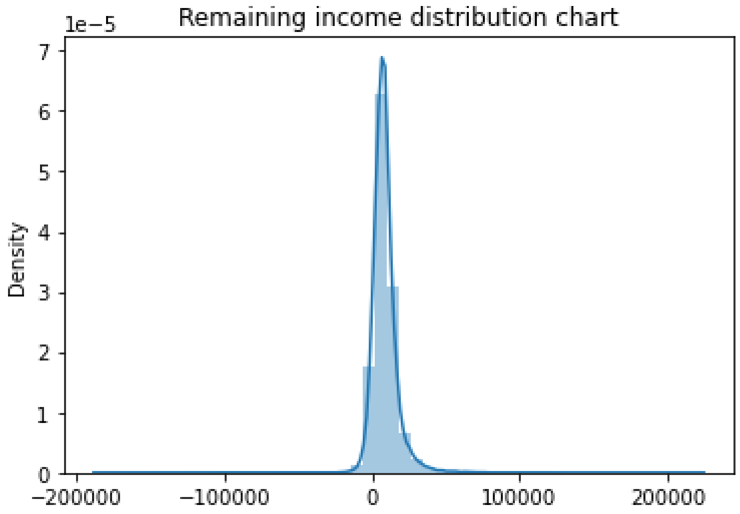
In [34]:
```
# Visualization 8:
sns.distplot(train['hi_mean'])
plt.title('Household income distribution chart')
plt.show()
```



In [35]:
```
# Visualization 9:
sns.distplot(train['family_mean'])
plt.title('Family income distribution chart')
plt.show()
```



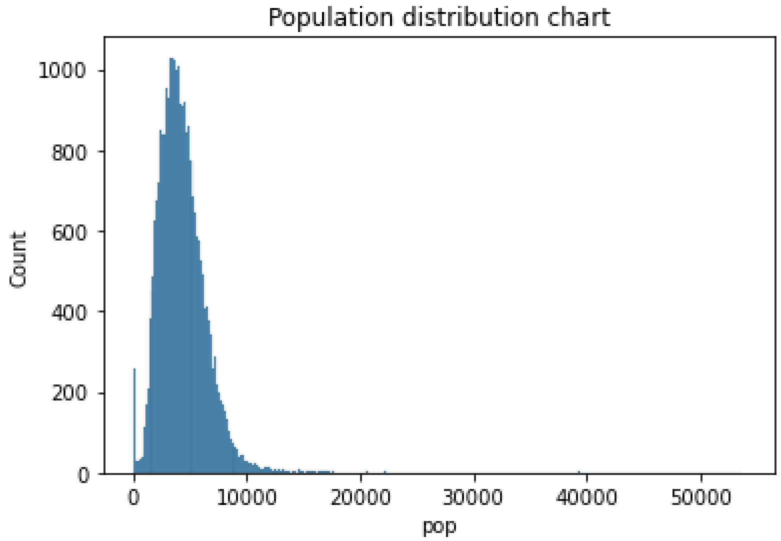
In [36]:
```
# Visualization 10:
sns.distplot(train['family_mean']-train['hi_mean'])
plt.title('Remaining income distribution chart')
plt.show()
```



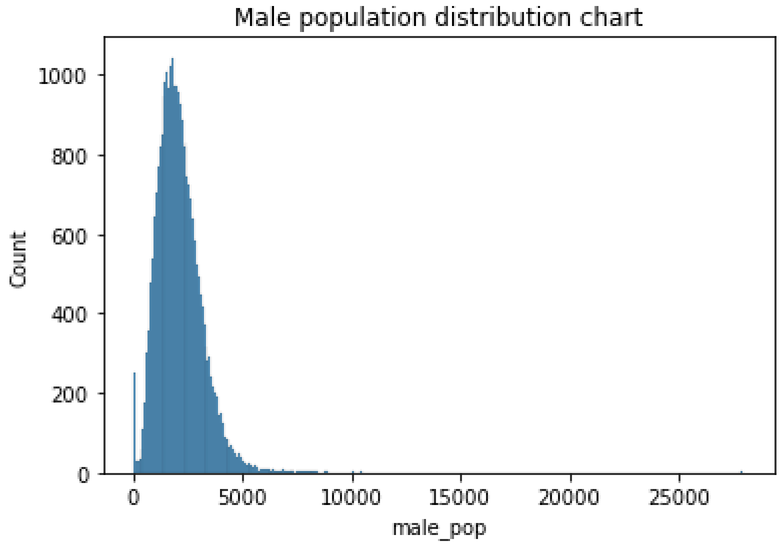
In [ ]:
```
```

Week 2 Exploratory Data Analysis:

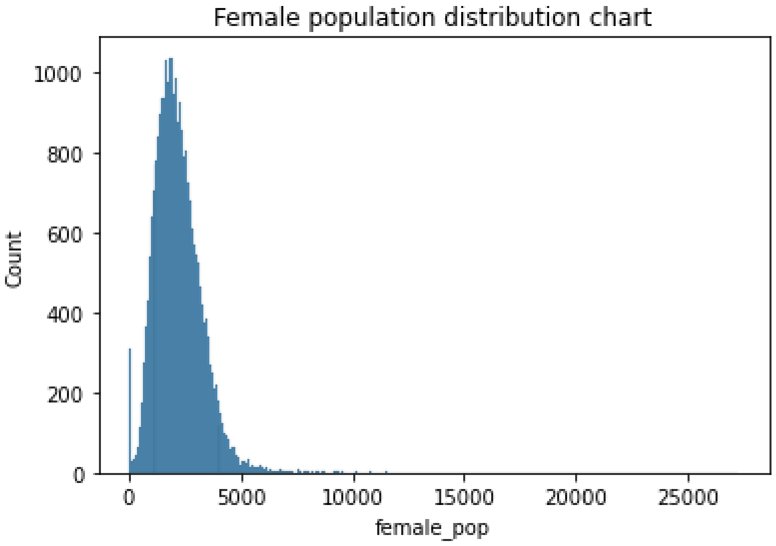
In [37]:
```
# Visualization 11:
sns.histplot(train['pop'])
plt.title('Population distribution chart')
plt.show()
```



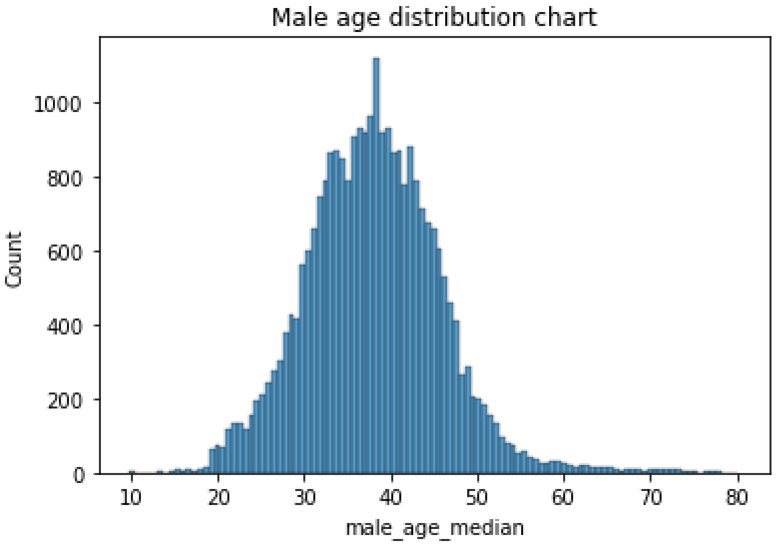
In [38]:
```
# Visualization 12:
sns.histplot(train['male_pop'])
plt.title('Male population distribution chart')
plt.show()
```



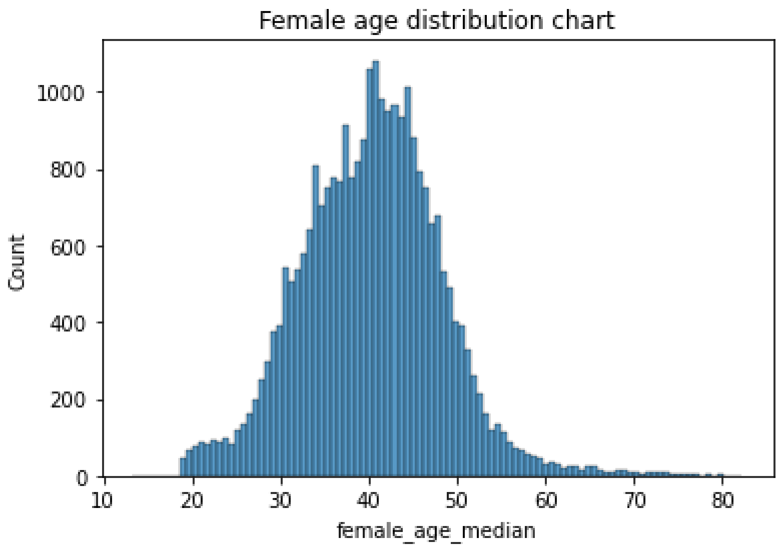
In [39]:
```
# Visualization 13:
sns.histplot(train['female_pop'])
plt.title('Female population distribution chart')
plt.show()
```



In [40]:
```
# Visualization 14:
sns.histplot(train['male_age_median'])
plt.title('Male age distribution chart')
plt.show()
```



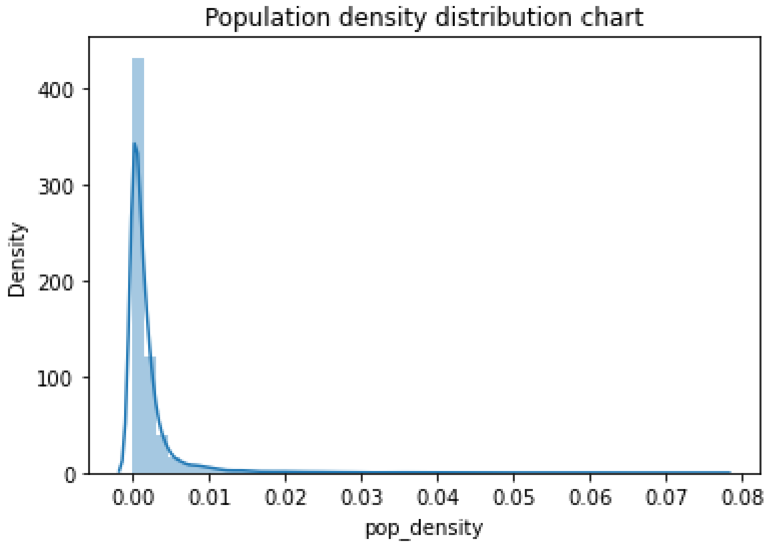
In [41]:
```
# Visualization 15:
sns.histplot(train['female_age_median'])
plt.title('Female age distribution chart')
plt.show()
```



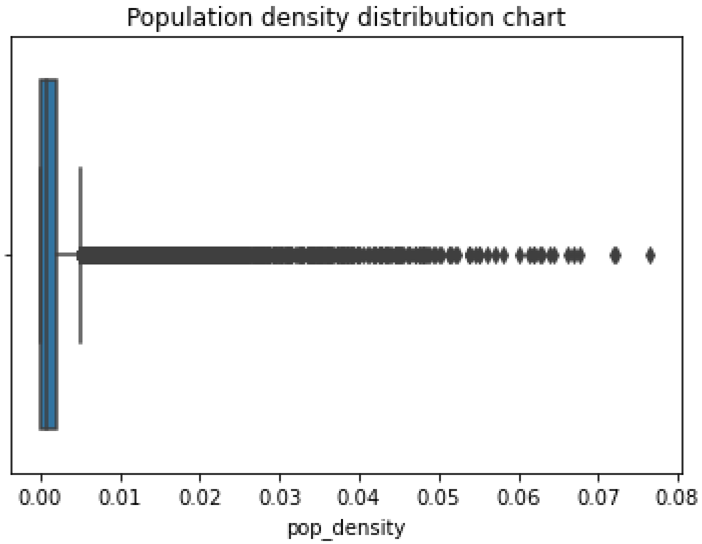
In [42]:
```
train["pop_density"]=train["pop"]/train["ALand"]
```

In [43]:
```
test["pop_density"]=test["pop"]/test["ALand"]
```

In [44]:
```
# Visualization 16:
sns.distplot(train['pop_density'])
plt.title('Population density distribution chart')
plt.show()
```



In [45]:
```
# Visualization 17:
sns.boxplot(train['pop_density'])
plt.title('Population density distribution chart')
plt.show()
```



In [46]:
```
train["median_age"]=(train["male_age_median"]+train["female_age_median"])/2
```

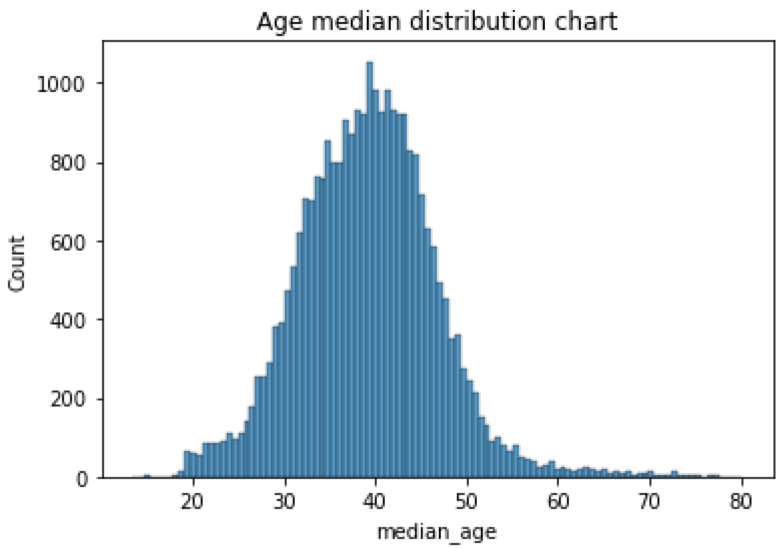
In [47]:
```
test["median_age"]=(test["male_age_median"]+test["female_age_median"])/2
```

In [48]:
```
train[['male_age_median','female_age_median','male_pop','female_pop','median_age']].head()
```

Out[48]:

UID	male_age_median	female_age_median	male_pop	female_pop	median_age
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

In [49]:
```
# Visualization 18:
sns.histplot(train['median_age'])
plt.title('Age median distribution chart')
plt.show()
```



In [50]:
```
train["pop"].describe()
```

Out[50]:
```
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
```

In [51]:
```
train['pop_bins']=pd.cut(train['pop'],bins=5,labels=['very low','low','medium','high','very high'])
```

In [52]:
```
train[['pop','pop_bins']]
```

Out[52]:

UID	pop	pop_bins
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

In [53]:
```
train['pop_bins'].value_counts()
```

Out[53]:
```
very low     27058
low            246
medium           9
high             7
very high        1
Name: pop_bins, dtype: int64
```

In [54]:
```
train.groupby(by='pop_bins')[['married','separated','divorced']].count()
```

Out[54]:

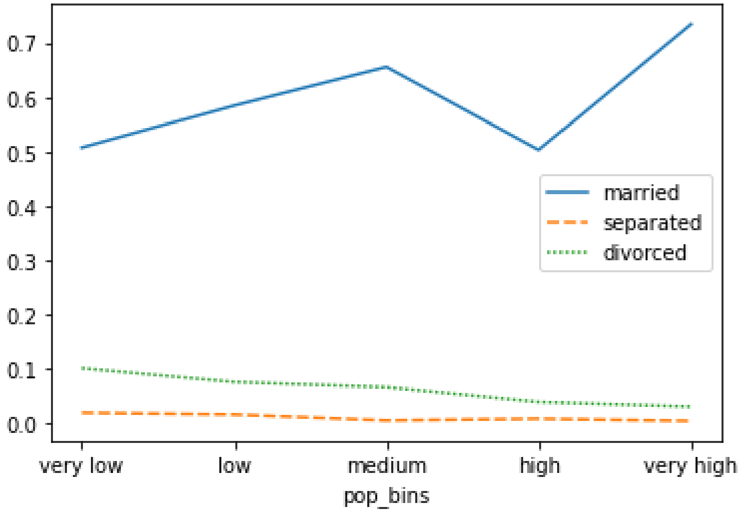
pop_bins	married	separated	divorced
very low	27058	27058	27058
low	246	246	246
medium	9	9	9
high	7	7	7
very high	1	1	1

In [55]:
```
train.groupby(by='pop_bins')[['married','separated','divorced']].agg(["mean", "median"])
```

Out[55]:

	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

In [56]:
```
# Visualization 19:
pop_bin_married=train.groupby(by='pop_bins')[['married','separated','divorced']].agg(["mean"])
sns.lineplot(data=pop_bin_married)
plt.show()
```



In [57]:
```
rent_state_mean=train.groupby(by='state')['rent_mean'].agg(["mean"])
rent_state_mean.head()
```

Out[57]:

	mean
state	
Alabama	774.004927
Alaska	1185.763570
Arizona	1097.753511
Arkansas	720.918575
California	1471.133857

In [58]:
```
income_state_mean=train.groupby(by='state')['family_mean'].agg(["mean"])
income_state_mean.head()
```

Out[58]:

state	mean
Alabama	67030.064213
Alaska	92136.545109
Arizona	73328.238798
Arkansas	64765.377850
California	87655.470820

In [59]:
```
rent_perc_of_income=rent_state_mean['mean']/income_state_mean['mean']
rent_perc_of_income.head(10)
```

Out[59]:
```
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
```

In [60]:
```
#overall level rent as a percentage of income
sum(train['rent_mean'])/sum(train['family_mean'])
```

Out[60]:
```
0.013358170721473864
```

In [61]:

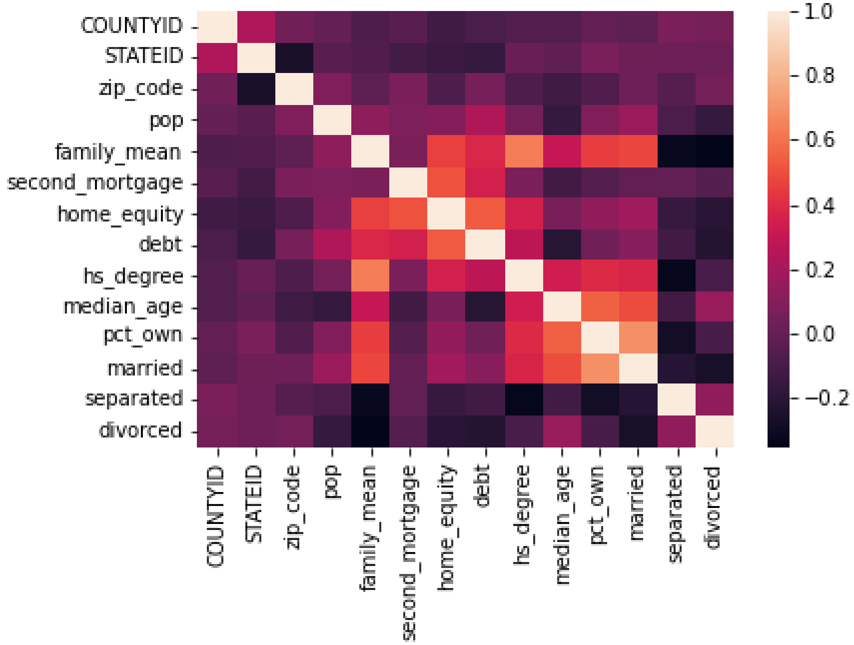
```
#Correlation analysis and heatmap
train[["COUNTYID","STATEID","zip_code", "type","pop","family_mean",'second_mortgage', 'home_equity', 'debt','hs_degree',
```

Out[61]:

	COUNTYID	STATEID	zip_code	pop	family_mean	second_mortgage	home_equity	debt	hs_degree	median_age	pc
COUNTYID	1.000000	0.224549	0.036527	-0.002662	-0.075688	-0.039283	-0.123939	-0.086231	-0.062703	-0.063521	-0.0
STATEID	0.224549	1.000000	-0.261465	-0.036599	-0.071612	-0.112512	-0.145301	-0.160532	0.014132	-0.017172	0.0
zip_code	0.036527	-0.261465	1.000000	0.083058	-0.024658	0.067693	-0.073191	0.057775	-0.077672	-0.126150	-0.0
pop	-0.002662	-0.036599	0.083058	1.000000	0.128173	0.079675	0.099352	0.231013	0.049238	-0.162499	0.0
family_mean	-0.075688	-0.071612	-0.024658	0.128173	1.000000	0.074703	0.458973	0.378871	0.634493	0.300215	0.4
second_mortgage	-0.039283	-0.112512	0.067693	0.079675	0.074703	1.000000	0.510460	0.351298	0.064412	-0.116616	-0.0
home_equity	-0.123939	-0.145301	-0.073191	0.099352	0.458973	0.510460	1.000000	0.532062	0.354566	0.063776	0.1
debt	-0.086231	-0.160532	0.057775	0.231013	0.378871	0.351298	0.532062	1.000000	0.279957	-0.213281	0.0
hs_degree	-0.062703	0.014132	-0.077672	0.049238	0.634493	0.064412	0.354566	0.279957	1.000000	0.334228	0.3
median_age	-0.063521	-0.017172	-0.126150	-0.162499	0.300215	-0.116616	0.063776	-0.213281	0.334228	1.000000	0.5
pct_own	-0.004632	0.069314	-0.069965	0.088457	0.450961	-0.054530	0.140941	0.034207	0.390815	0.546692	1.0
married	-0.021428	0.025763	0.030217	0.167656	0.480095	-0.006438	0.189763	0.108496	0.370706	0.495153	0.6
separated	0.069059	0.030409	-0.048023	-0.083182	-0.323433	-0.010731	-0.155198	-0.119073	-0.333321	-0.116763	-0.2
divorced	0.048850	0.018748	0.043310	-0.160931	-0.353274	-0.056991	-0.207202	-0.222350	-0.092984	0.164205	-0.0

In [62]:
```
# Visualization 20:
sns.heatmap(train[["COUNTYID","STATEID","zip_code", "type","pop","family_mean",'second_mortgage', 'home_equity', 'debt',
```

Out[62]:
```
<AxesSubplot:>
```



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

In [63]:
```
from sklearn.decomposition import FactorAnalysis
```

In [64]:
```
fa = FactorAnalysis(n_components=5,random_state=11)
```

In [65]:
```
train_transformed = fa.fit_transform(train.select_dtypes(exclude=('object','category')))
```

In [66]:
```
train_transformed.shape
```

Out[66]:
```
(27321, 5)
```

In [67]:
```
train_transformed
```

Out[67]:
```
array([[ 0.05640687, -0.05073008,  1.25002287, -0.32623122,  0.1814258 ],
       [-0.10015645,  0.01442735,  0.11011385, -0.95809505,  0.58805725],
       [-0.04710979, -0.0094559 ,  0.13106345,  0.45168299,  0.90055   ],
       ...,
       [ 0.93167634, -0.37995383, -0.96907522,  0.41947921,  0.30372189],
       [-0.08682288,  0.00848632, -0.88563901,  3.03163033,  1.15593996],
       [-0.09529886,  0.01164864, -1.3315217 , -0.69048311, -0.11200756]])
```

In [68]:
```
x_train = pd.read_csv('train.csv')
x_test = pd.read_csv('test.csv')
```

In [69]:
```
x_train.drop(['BLOCKID','SUMLEVEL'],axis=1,inplace=True)
```

In [70]:
```
x_train.dropna(axis=0,inplace=True)
x_train.head()
```

Out[70]:

	UID	COUNTYID	STATEID	state	state_ab	city	place	type	primary	zip_code	area_code	lat	lng	ALand
0	267822	53	36	New York	NY	Hamilton	Hamilton	City	tract	13346	315	42.840812	-75.501524	202183361.0
1	246444	141	18	Indiana	IN	South Bend	Roseland	City	tract	46616	574	41.701441	-86.266614	1560828.0
2	245683	63	18	Indiana	IN	Danville	Danville	City	tract	46122	317	39.792202	-86.515246	69561595.0
3	279653	127	72	Puerto Rico	PR	San Juan	Guaynabo	Urban	tract	927	787	18.396103	-66.104169	1105793.0
4	247218	161	20	Kansas	KS	Manhattan	Manhattan City	City	tract	66502	785	39.195573	-96.569366	2554403.0

In [71]:
```
x_train.drop_duplicates(inplace=True)
```

In [72]:
```
x_train.shape
```

Out[72]: (26585, 78)

In [73]:
```
x_test.head()
```

Out[73]:

	UID	BLOCKID	SUMLEVEL	COUNTYID	STATEID	state	state_ab	city	place	type	primary	zip_code	area_code	
0	255504	NaN	140	163	26	Michigan	MI	Detroit	Dearborn Heights City	CDP	tract	48239	313	42.346
1	252676	NaN	140	1	23	Maine	ME	Auburn	Auburn City	City	tract	4210	207	44.100
2	276314	NaN	140	15	42	Pennsylvania	PA	Pine City	Millerton	Borough	tract	14871	607	41.948
3	248614	NaN	140	231	21	Kentucky	KY	Monticello	Monticello City	City	tract	42633	606	36.746
4	286865	NaN	140	355	48	Texas	TX	Corpus Christi	Edroy	Town	tract	78410	361	27.882

In [74]:
```
x_test.shape
```

Out[74]: (11709, 80)

In [75]:
```
x_test.drop(['BLOCKID','SUMLEVEL'],axis=1,inplace=True)
```

In [76]:
```
x_test.isna().sum()
```

Out[76]:
```
UID              0
COUNTYID         0
STATEID          0
state            0
state_ab         0
              ...
pct_own        122
married         84
married_snp     84
separated       84
divorced        84
Length: 78, dtype: int64
```

In [77]:
```
x_test.dropna(axis=0,inplace=True)
```

In [78]:
```
x_test.drop_duplicates(inplace=True)
```

In [79]:
```
x_test.shape
```

Out[79]:
```
(11355, 78)
```

In [80]:
```
imp_feature = x_train.select_dtypes(exclude=('object','category'))
```

In [81]:
```
imp_feature.head()
```

Out[81]:

	UID	COUNTYID	STATEID	zip_code	area_code	lat	lng	ALand	AWater	pop	male_pop	female_pop	rent_mean	rent_
0	267822	53	36	13346	315	42.840812	-75.501524	202183361.0	1699120	5230	2612	2618	769.38638	
1	246444	141	18	46616	574	41.701441	-86.266614	1560828.0	100363	2633	1349	1284	804.87924	
2	245683	63	18	46122	317	39.792202	-86.515246	69561595.0	284193	6881	3643	3238	742.77365	
3	279653	127	72	927	787	18.396103	-66.104169	1105793.0	0	2700	1141	1559	803.42018	
4	247218	161	20	66502	785	39.195573	-96.569366	2554403.0	0	5637	2586	3051	938.56493	

In [82]:
```
imp_feature.shape
```

Out[82]:
```
(26585, 72)
```

In [83]:
```
to_drop = ['UID','COUNTYID', 'STATEID', 'zip_code', 'area_code', 'lat', 'lng']
```

In [84]:
```
for col in imp_feature.columns:
    if col in to_drop:
        imp_feature.drop(col,axis=1,inplace=True)
```

In [85]:
```
imp_feature.head()
```

Out[85]:

	ALand	AWater	pop	male_pop	female_pop	rent_mean	rent_median	rent_stdev	rent_sample_weight	rent_samples	rent_gt_10	rent_gt_1
0	202183361.0	1699120	5230	2612	2618	769.38638	784.0	232.63967	272.34441	362.0	0.86761	0.7915
1	1560828.0	100363	2633	1349	1284	804.87924	848.0	253.46747	312.58622	513.0	0.97410	0.9322
2	69561595.0	284193	6881	3643	3238	742.77365	703.0	323.39011	291.85520	378.0	0.95238	0.8862
3	1105793.0	0	2700	1141	1559	803.42018	782.0	297.39258	259.30316	368.0	0.94693	0.8715
4	2554403.0	0	5637	2586	3051	938.56493	881.0	392.44096	1005.42886	1704.0	0.99286	0.9824

In [86]:
```
x_train_features = imp_feature[['pop','rent_median','hi_median','family_median','hc_mean','second_mortgage','home_equity
```

In [87]:
```
x_train_features.head()
```

Out[87]:

	pop	rent_median	hi_median	family_median	hc_mean	second_mortgage	home_equity	debt	hs_degree	pct_own	married	separated	divo
0	5230	784.0	48120.0	53245.0	570.01530	0.02077	0.08919	0.52963	0.89288	0.79046	0.57851	0.01240	0.0
1	2633	848.0	35186.0	43023.0	351.98293	0.02222	0.04274	0.60855	0.90487	0.52483	0.34886	0.01426	0.0
2	6881	703.0	74964.0	85395.0	556.45986	0.00000	0.09512	0.73484	0.94288	0.85331	0.64745	0.01607	0.1
3	2700	782.0	37845.0	44399.0	288.04047	0.01086	0.01086	0.52714	0.91500	0.65037	0.47257	0.02021	0.1
4	5637	881.0	22497.0	50272.0	443.68855	0.05426	0.05426	0.51938	1.00000	0.13046	0.12356	0.00000	0.0

In [88]:
```
x_train_features.shape
```

Out[88]:
```
(26585, 13)
```

In [89]:
```
y_train = imp_feature['hc_mortgage_mean']
```

In [90]:
```
x_test_feature = x_test[['pop','rent_median','hi_median','family_median','hc_mean','second_mortgage','home_equity','debt
```

In [91]:
```
from sklearn.linear_model import LinearRegression
le = LinearRegression()
```

In [92]:
```
le.fit(x_train_features,y_train)
```

Out[92]: LinearRegression()

In [93]:
```
y_pred = le.predict(x_test_feature)
```

In [94]:
```
y_test = x_test['hc_mortgage_mean']
```

In [95]:
```
from sklearn.metrics import r2_score,mean_squared_error
```

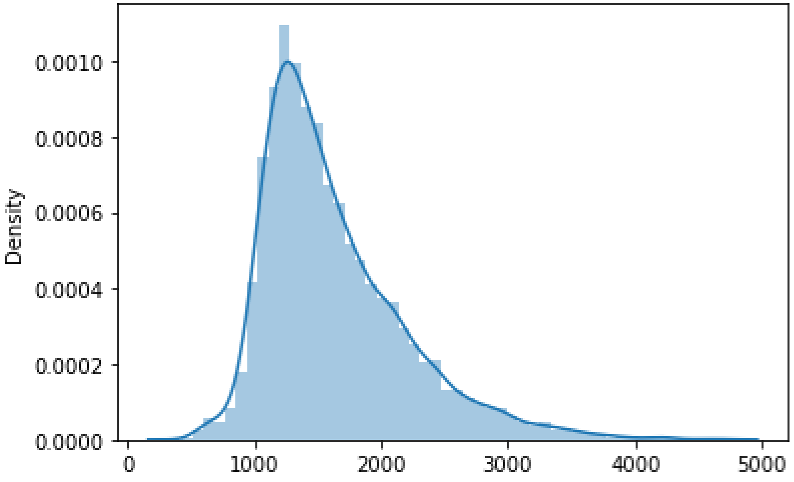
In [96]:
```
r2_score(y_test,y_pred)
```

Out[96]: 0.8073813546881963

In [97]:
```
np.sqrt(mean_squared_error(y_test,y_pred))
```

Out[97]: 277.04518388580743

In [98]:
```
# Visualization 21:
sns.distplot(y_pred)
plt.show()
```



In [ ]: