Real Estate

Course-end Project 1

Description

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

Project Task: Week 1

Data Import and Preparation:

- Import data.
- Figure out the primary key and look for the requirement of indexing.
- 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.

Exploratory Data Analysis (EDA):

- Perform debt analysis. You may take the following steps:
- 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

Use the following bad debt equation:

```
Bad Debt = P (Second Mortgage ∩ Home Equity Loan)
Bad Debt = second_mortgage + home_equity - home_equity_second_mortgage
Create pie charts  to show overall debt and bad debt
```

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

- Create a collated income distribution chart for family income, house hold income, and remaining income
- 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):
- Use pop and ALand variables to create a new field called population density
- Use male_age_median, female_age_median, male_pop, and female_pop to create a new field called median age
- Visualize the findings using appropriate chart type
- 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.
- Analyze the married, separated, and divorced population for these population brackets
- Visualize using appropriate chart type
- Please detail your observations for rent as a percentage of income at an overall level, and for different states.
- Perform correlation analysis for all the relevant variables by creating a heatmap. Describe your findings.

Project Task: Week 2

Data Pre-processing:

- 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.
- 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
```

Data Modeling :

- 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.

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

 b) Run another model at State level. There are 52 states in USA.

 c) Keep below considerations while building a linear regression model:
```

- Variables should have significant impact on predicting Monthly mortgage and owner costs
- Utilize all predictor variable to start with initial hypothesis
- R square of 60 percent and above should be achieved
- Ensure Multi-collinearity does not exist in dependent variables
- Test if predicted variable is normally distributed

Data Reporting:

- Create a dashboard in tableau by choosing appropriate chart types and metrics useful for the business. The dashboard must entail the following:
- Box plot of distribution of average rent by type of place (village, urban, town, etc.).
- Pie charts to show overall debt and bad debt.
- 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.
- Heat map for correlation matrix.
- Pie chart to show the population distribution across different types of places (village, urban, town etc.).

Data Import and Preparation:

Question 1

Import data.

In [1]:
```
! pip install numpy pandas scikit-learn matplotlib seaborn plotly dash factor-analyzer
```

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: numpy in c:\users\aditya raj\appdata\local\packages\pythonsoftwarefoundation.python.3.12_qbz5n
2kfra8p0\localcache\local-packages\python312\site-packages (1.26.4)
Requirement already satisfied: pandas in c:\users\aditya raj\appdata\local\packages\pythonsoftwarefoundation.python.3.12_qbz5
n2kfra8p0\localcache\local-packages\python312\site-packages (2.2.1)
Requirement already satisfied: scikit-learn in c:\users\aditya raj\appdata\local\packages\pythonsoftwarefoundation.python.3.1
2_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (1.4.1.post1)
Requirement already satisfied: matplotlib in c:\users\aditya raj\appdata\local\packages\pythonsoftwarefoundation.python.3.12_
qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (3.8.3)
Requirement already satisfied: seaborn in c:\users\aditya raj\appdata\local\packages\pythonsoftwarefoundation.python.3.12_qbz
5n2kfra8p0\localcache\local-packages\python312\site-packages (0.13.2)
Requirement already satisfied: plotly in c:\users\aditya raj\appdata\local\packages\pythonsoftwarefoundation.python.3.12_qbz5
n2kfra8p0\localcache\local-packages\python312\site-packages (5.20.0)
Requirement already satisfied: dash in c:\users\aditya raj\appdata\local\packages\pythonsoftwarefoundation.python.3.12_qbz5n2
kfra8p0\localcache\local-packages\python312\site-packages (2.16.1)
Requirement already satisfied: factor-analyzer in c:\users\aditya raj\appdata\local\packages\pythonsoftwarefoundation.python.
3.12_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (0.5.1)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\users\aditya raj\appdata\local\packages\pythonsoftwarefoundation.
python.3.12_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (from pandas) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in c:\users\aditya raj\appdata\local\packages\pythonsoftwarefoundation.python.3.1
2_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (from pandas) (2024.1)
Requirement already satisfied: tzdata>=2022.7 in c:\users\aditya raj\appdata\local\packages\pythonsoftwarefoundation.python.
3.12_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (from pandas) (2024.1)
Requirement already satisfied: scipy>=1.6.0 in c:\users\aditya raj\appdata\local\packages\pythonsoftwarefoundation.python.3.1
2_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (from scikit-learn) (1.12.0)
Requirement already satisfied: joblib>=1.2.0 in c:\users\aditya raj\appdata\local\packages\pythonsoftwarefoundation.python.3.
12_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (from scikit-learn) (1.3.2)
Requirement already satisfied: threadpoolctl>=2.0.0 in c:\users\aditya raj\appdata\local\packages\pythonsoftwarefoundation.py
thon.3.12_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (from scikit-learn) (3.4.0)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\aditya raj\appdata\local\packages\pythonsoftwarefoundation.pytho
n.3.12_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (from matplotlib) (1.2.0)
Requirement already satisfied: cycler>=0.10 in c:\users\aditya raj\appdata\local\packages\pythonsoftwarefoundation.python.3.1
2_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\aditya raj\appdata\local\packages\pythonsoftwarefoundation.pytho
n.3.12_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (from matplotlib) (4.50.0)
Requirement already satisfied: kiwisolver>=1.3.1 in c:\users\aditya raj\appdata\local\packages\pythonsoftwarefoundation.pytho
n.3.12_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (from matplotlib) (1.4.5)
Requirement already satisfied: packaging>=20.0 in c:\users\aditya raj\appdata\local\packages\pythonsoftwarefoundation.python.
3.12_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (from matplotlib) (24.0)
Requirement already satisfied: pillow>=8 in c:\users\aditya raj\appdata\local\packages\pythonsoftwarefoundation.python.3.12_q
bz5n2kfra8p0\localcache\local-packages\python312\site-packages (from matplotlib) (10.2.0)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\aditya raj\appdata\local\packages\pythonsoftwarefoundation.pytho
n.3.12_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (from matplotlib) (3.1.2)
Requirement already satisfied: tenacity>=6.2.0 in c:\users\aditya raj\appdata\local\packages\pythonsoftwarefoundation.python.
3.12_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (from plotly) (8.2.3)
Requirement already satisfied: Flask<3.1,>=1.0.4 in c:\users\aditya raj\appdata\local\packages\pythonsoftwarefoundation.pytho
n.3.12_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (from dash) (3.0.2)
Requirement already satisfied: Werkzeug<3.1 in c:\users\aditya raj\appdata\local\packages\pythonsoftwarefoundation.python.3.1
2_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (from dash) (3.0.1)
Requirement already satisfied: dash-html-components==2.0.0 in c:\users\aditya raj\appdata\local\packages\pythonsoftwarefounda
tion.python.3.12_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (from dash) (2.0.0)
Requirement already satisfied: dash-core-components==2.0.0 in c:\users\aditya raj\appdata\local\packages\pythonsoftwarefounda
tion.python.3.12_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (from dash) (2.0.0)
Requirement already satisfied: dash-table==5.0.0 in c:\users\aditya raj\appdata\local\packages\pythonsoftwarefoundation.pytho
n.3.12_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (from dash) (5.0.0)
Requirement already satisfied: importlib-metadata in c:\users\aditya raj\appdata\local\packages\pythonsoftwarefoundation.pyth
on.3.12_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (from dash) (7.1.0)
Requirement already satisfied: typing-extensions>=4.1.1 in c:\users\aditya raj\appdata\local\packages\pythonsoftwarefoundatio
n.python.3.12_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (from dash) (4.10.0)
Requirement already satisfied: requests in c:\users\aditya raj\appdata\local\packages\pythonsoftwarefoundation.python.3.12_qb
z5n2kfra8p0\localcache\local-packages\python312\site-packages (from dash) (2.31.0)
Requirement already satisfied: retrying in c:\users\aditya raj\appdata\local\packages\pythonsoftwarefoundation.python.3.12_qb
z5n2kfra8p0\localcache\local-packages\python312\site-packages (from dash) (1.3.4)
Requirement already satisfied: nest-asyncio in c:\users\aditya raj\appdata\local\packages\pythonsoftwarefoundation.python.3.1
2_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (from dash) (1.6.0)
Requirement already satisfied: setuptools in c:\users\aditya raj\appdata\local\packages\pythonsoftwarefoundation.python.3.12_
qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (from dash) (69.2.0)
Requirement already satisfied: Jinja2>=3.1.2 in c:\users\aditya raj\appdata\local\packages\pythonsoftwarefoundation.python.3.
12_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (from Flask<3.1,>=1.0.4->dash) (3.1.3)
Requirement already satisfied: itsdangerous>=2.1.2 in c:\users\aditya raj\appdata\local\packages\pythonsoftwarefoundation.pyt
hon.3.12_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (from Flask<3.1,>=1.0.4->dash) (2.1.2)
Requirement already satisfied: click>=8.1.3 in c:\users\aditya raj\appdata\local\packages\pythonsoftwarefoundation.python.3.1
2_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (from Flask<3.1,>=1.0.4->dash) (8.1.7)
Requirement already satisfied: blinker>=1.6.2 in c:\users\aditya raj\appdata\local\packages\pythonsoftwarefoundation.python.
3.12_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (from Flask<3.1,>=1.0.4->dash) (1.7.0)
Requirement already satisfied: six>=1.5 in c:\users\aditya raj\appdata\local\packages\pythonsoftwarefoundation.python.3.12_qb
z5n2kfra8p0\localcache\local-packages\python312\site-packages (from python-dateutil>=2.8.2->pandas) (1.16.0)
Requirement already satisfied: MarkupSafe>=2.1.1 in c:\users\aditya raj\appdata\local\packages\pythonsoftwarefoundation.pytho
n.3.12_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (from Werkzeug<3.1->dash) (2.1.5)
```

```
Requirement already satisfied: zipp>=0.5 in c:\users\aditya raj\appdata\local\packages\pythonsoftwarefoundation.python.3.12_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (from importlib-metadata->dash) (3.18.1)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\aditya raj\appdata\local\packages\pythonsoftwarefoundation.python.3.12_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (from requests->dash) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in c:\users\aditya raj\appdata\local\packages\pythonsoftwarefoundation.python.3.12_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (from requests->dash) (3.6)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\aditya raj\appdata\local\packages\pythonsoftwarefoundation.python.3.12_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (from requests->dash) (2.2.1)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\aditya raj\appdata\local\packages\pythonsoftwarefoundation.python.3.12_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (from requests->dash) (2024.2.2)
Requirement already satisfied: colorama in c:\users\aditya raj\appdata\local\packages\pythonsoftwarefoundation.python.3.12_qbz5n2kfra8p0\localcache\local-packages\python312\site-packages (from click>=8.1.3->Flask<3.1,>=1.0.4->dash) (0.4.6)
```

In [2]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
from plotly.subplots import make_subplots
import plotly.graph_objects as go
import plotly.figure_factory as ff
from dash import Dash, html, dcc, callback, Output, Input
import geopandas as gpd
from factor_analyzer import FactorAnalyzer
from scipy import stats
import warnings
```

In [3]:
```
warnings.filterwarnings('ignore')
pd.set_option('display.max_rows', None)
sns.set_style('dark')
pd.set_option('display.max_columns', None)
sns.set(rc={'axes.facecolor':'black', 'figure.facecolor':'darkgrey'})
```

In [4]:
```
# Imports from Sklearn and other modelling packages
from sklearn.impute import KNNImputer
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
```

In [5]:
```
# Loading Data
df_train = pd.read_csv('train.csv'); print(df_train.shape)
df_test = pd.read_csv('test.csv'); print(df_test.shape)

#This flag will help us split the data back later
df_train['split']= 'Train'
df_test['split']= 'Test'


df = pd.concat([df_train, df_test], axis=0); print(df.shape)

df.head()
```

```
(27321, 80)
(11709, 80)
(39030, 81)
```

Out[5]:

	UID	BLOCKID	SUMLEVEL	COUNTYID	STATEID	state	state_ab	city	place	type	primary	zip_code	area_code	
0	267822	NaN	140	53	36	New York	NY	Hamilton	Hamilton	City	tract	13346	315	4
1	246444	NaN	140	141	18	Indiana	IN	South Bend	Roseland	City	tract	46616	574	4
2	245683	NaN	140	63	18	Indiana	IN	Danville	Danville	City	tract	46122	317	3
3	279653	NaN	140	127	72	Puerto Rico	PR	San Juan	Guaynabo	Urban	tract	927	787	1
4	247218	NaN	140	161	20	Kansas	KS	Manhattan	Manhattan City	City	tract	66502	785	3

Question 2

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

In [6]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 39030 entries, 0 to 11708
Data columns (total 81 columns):
 #   Column                        Non-Null Count  Dtype
---  ------                        --------------  -----
 0   UID                           39030 non-null  int64
 1   BLOCKID                       0 non-null      float64
 2   SUMLEVEL                      39030 non-null  int64
 3   COUNTYID                      39030 non-null  int64
 4   STATEID                       39030 non-null  int64
 5   state                         39030 non-null  object
 6   state_ab                      39030 non-null  object
 7   city                          39030 non-null  object
 8   place                         39030 non-null  object
 9   type                          39030 non-null  object
 10  primary                       39030 non-null  object
 11  zip_code                      39030 non-null  int64
 12  area_code                     39030 non-null  int64
 13  lat                           39030 non-null  float64
 14  lng                           39030 non-null  float64
 15  ALand                         39030 non-null  float64
 16  AWater                        39030 non-null  int64
 17  pop                           39030 non-null  int64
 18  male_pop                      39030 non-null  int64
 19  female_pop                    39030 non-null  int64
 20  rent_mean                     38568 non-null  float64
 21  rent_median                   38568 non-null  float64
 22  rent_stdev                    38568 non-null  float64
 23  rent_sample_weight            38568 non-null  float64
 24  rent_samples                  38568 non-null  float64
 25  rent_gt_10                    38567 non-null  float64
 26  rent_gt_15                    38567 non-null  float64
 27  rent_gt_20                    38567 non-null  float64
 28  rent_gt_25                    38567 non-null  float64
 29  rent_gt_30                    38567 non-null  float64
 30  rent_gt_35                    38567 non-null  float64
 31  rent_gt_40                    38567 non-null  float64
 32  rent_gt_50                    38567 non-null  float64
 33  universe_samples              39030 non-null  int64
 34  used_samples                  39030 non-null  int64
 35  hi_mean                       38640 non-null  float64
 36  hi_median                     38640 non-null  float64
 37  hi_stdev                      38640 non-null  float64
 38  hi_sample_weight              38640 non-null  float64
 39  hi_samples                    38640 non-null  float64
 40  family_mean                   38596 non-null  float64
 41  family_median                 38596 non-null  float64
 42  family_stdev                  38596 non-null  float64
 43  family_sample_weight          38596 non-null  float64
 44  family_samples                38596 non-null  float64
 45  hc_mortgage_mean              38189 non-null  float64
 46  hc_mortgage_median            38189 non-null  float64
 47  hc_mortgage_stdev             38189 non-null  float64
 48  hc_mortgage_sample_weight     38189 non-null  float64
 49  hc_mortgage_samples           38189 non-null  float64
 50  hc_mean                       38140 non-null  float64
 51  hc_median                     38140 non-null  float64
 52  hc_stdev                      38140 non-null  float64
 53  hc_samples                    38140 non-null  float64
 54  hc_sample_weight              38140 non-null  float64
 55  home_equity_second_mortgage   38353 non-null  float64
 56  second_mortgage               38353 non-null  float64
 57  home_equity                   38353 non-null  float64
 58  debt                          38353 non-null  float64
 59  second_mortgage_cdf           38353 non-null  float64
 60  home_equity_cdf               38353 non-null  float64
 61  debt_cdf                      38353 non-null  float64
 62  hs_degree                     38755 non-null  float64
 63  hs_degree_male                38741 non-null  float64
 64  hs_degree_female              38702 non-null  float64
 65  male_age_mean                 38757 non-null  float64
 66  male_age_median               38757 non-null  float64
 67  male_age_stdev                38757 non-null  float64
 68  male_age_sample_weight        38757 non-null  float64
 69  male_age_samples              38757 non-null  float64
 70  female_age_mean               38728 non-null  float64
 71  female_age_median             38728 non-null  float64
 72  female_age_stdev              38728 non-null  float64
 73  female_age_sample_weight      38728 non-null  float64
```

```
 74  female_age_samples           38728 non-null  float64
 75  pct_own                      38640 non-null  float64
 76  married                      38755 non-null  float64
 77  married_snp                  38755 non-null  float64
 78  separated                    38755 non-null  float64
 79  divorced                     38755 non-null  float64
 80  split                        39030 non-null  object
dtypes: float64(62), int64(12), object(7)
memory usage: 24.4+ MB
```

- Changing the naems of columns to lower-case

In [7]:
```
df.columns = df.columns.str.lower()
```

In [8]:
```
df.describe().transpose()
```

Out[8]:

	count	mean	std	min	25%	50%	75%	
uid	39030.0	2.573899e+05	2.138060e+04	220336.000000	2.388172e+05	2.573805e+05	2.759708e+05	2.94
blockid	0.0	NaN	NaN	NaN	NaN	NaN	NaN	
sumlevel	39030.0	1.400000e+02	0.000000e+00	140.000000	1.400000e+02	1.400000e+02	1.400000e+02	1.40
countyid	39030.0	8.566569e+01	9.862420e+01	1.000000	2.900000e+01	6.300000e+01	1.090000e+02	8.40
stateid	39030.0	2.833702e+01	1.645755e+01	1.000000	1.300000e+01	2.800000e+01	4.200000e+01	7.20
zip_code	39030.0	5.009443e+04	2.962301e+04	601.000000	2.626100e+04	4.763750e+04	7.732000e+04	9.99
area_code	39030.0	5.956349e+02	2.323714e+02	201.000000	4.050000e+02	6.140000e+02	8.010000e+02	9.89
lat	39030.0	3.747782e+01	5.599713e+00	17.929085	3.390548e+01	3.871221e+01	4.132777e+01	6.70
lng	39030.0	-9.130394e+01	1.636285e+01	-166.770979	-9.781644e+01	-8.658708e+01	-7.976499e+01	-6.53
aland	39030.0	1.235224e+08	1.146022e+09	8299.000000	1.777882e+06	4.854619e+06	3.320334e+07	1.03
awater	39030.0	6.112049e+06	2.010698e+08	0.000000	0.000000e+00	2.595650e+04	5.132392e+05	2.45
pop	39030.0	4.331385e+03	2.155202e+03	0.000000	2.899000e+03	4.065000e+03	5.442000e+03	5.38
male_pop	39030.0	2.132501e+03	1.106520e+03	0.000000	1.412000e+03	1.988000e+03	2.673000e+03	2.79
female_pop	39030.0	2.198884e+03	1.097316e+03	0.000000	1.463000e+03	2.067000e+03	2.773000e+03	2.72
rent_mean	38568.0	1.054833e+03	4.365636e+02	117.150000	7.426814e+02	9.529741e+02	1.259885e+03	3.96
rent_median	38568.0	1.007476e+03	4.431000e+02	104.000000	7.030000e+02	8.970000e+02	1.197000e+03	3.97
rent_stdev	38568.0	3.943633e+02	1.877907e+02	18.257420	2.632678e+02	3.471726e+02	4.756280e+02	1.72
rent_sample_weight	38568.0	2.985383e+02	2.749905e+02	0.343000	1.023907e+02	2.222738e+02	4.120453e+02	4.11
rent_samples	38568.0	5.526431e+02	4.655351e+02	3.000000	2.230000e+02	4.290000e+02	7.490000e+02	7.63
rent_gt_10	38567.0	9.577213e-01	6.331078e-02	0.000000	9.405900e-01	9.770500e-01	1.000000e+00	1.00
rent_gt_15	38567.0	8.673243e-01	1.090979e-01	0.000000	8.197400e-01	8.885000e-01	9.403100e-01	1.00
rent_gt_20	38567.0	7.403840e-01	1.434207e-01	0.000000	6.630550e-01	7.598800e-01	8.380800e-01	1.00
rent_gt_25	38567.0	6.133925e-01	1.606805e-01	0.000000	5.171350e-01	6.256700e-01	7.234200e-01	1.00
rent_gt_30	38567.0	5.003519e-01	1.645326e-01	0.000000	3.966900e-01	5.044700e-01	6.095150e-01	1.00
rent_gt_35	38567.0	4.116023e-01	1.605353e-01	0.000000	3.073900e-01	4.099400e-01	5.157350e-01	1.00
rent_gt_40	38567.0	3.458975e-01	1.534465e-01	0.000000	2.429400e-01	3.395000e-01	4.420000e-01	1.00
rent_gt_50	38567.0	2.547799e-01	1.377162e-01	0.000000	1.606400e-01	2.431500e-01	3.370750e-01	1.00
universe_samples	39030.0	5.786274e+02	4.695183e+02	0.000000	2.510000e+02	4.590000e+02	7.770000e+02	7.63
used_samples	39030.0	5.327799e+02	4.544985e+02	0.000000	2.110000e+02	4.130000e+02	7.250000e+02	7.33
hi_mean	38640.0	7.035984e+04	3.030312e+04	4999.846690	4.905325e+04	6.396553e+04	8.570023e+04	2.97
hi_median	38640.0	5.751498e+04	2.928903e+04	4790.000000	3.731525e+04	5.118800e+04	7.067925e+04	2.96
hi_stdev	38640.0	5.434974e+04	1.767257e+04	1825.741860	4.197626e+04	5.213996e+04	6.517940e+04	1.35
hi_sample_weight	38640.0	9.270302e+02	4.544973e+02	0.114260	6.033096e+02	8.672207e+02	1.184152e+03	1.09
hi_samples	38640.0	1.612883e+03	7.500159e+02	3.000000	1.099000e+03	1.522000e+03	2.022000e+03	2.03
family_mean	38596.0	7.889682e+04	3.156500e+04	5374.842520	5.662406e+04	7.285496e+04	9.591450e+04	2.42
family_median	38596.0	6.921084e+04	3.367063e+04	5278.000000	4.601725e+04	6.226850e+04	8.460075e+04	2.42
family_stdev	38596.0	5.063234e+04	1.427344e+04	1825.741860	4.074870e+04	4.958999e+04	6.039576e+04	1.11
family_sample_weight	38596.0	5.356586e+02	2.901442e+02	0.199960	3.332809e+02	4.924642e+02	6.868784e+02	6.90
family_samples	38596.0	1.066489e+03	5.579105e+02	3.000000	6.887500e+02	9.890000e+02	1.352000e+03	1.49
hc_mortgage_mean	38189.0	1.631830e+03	6.266921e+02	234.650000	1.156408e+03	1.461484e+03	1.985853e+03	4.46
hc_mortgage_median	38189.0	1.553907e+03	6.562238e+02	237.000000	1.067000e+03	1.372000e+03	1.879000e+03	4.47
hc_mortgage_stdev	38189.0	6.223144e+02	2.388920e+02	36.514840	4.395353e+02	5.884327e+02	7.879708e+02	1.81
hc_mortgage_sample_weight	38189.0	2.880717e+02	1.958908e+02	0.198400	1.478149e+02	2.540932e+02	3.872603e+02	4.22
hc_mortgage_samples	38189.0	6.709076e+02	4.635394e+02	1.000000	3.460000e+02	5.910000e+02	8.990000e+02	1.16

	count	mean	std	min	25%	50%	75%	
hc_mean	38140.0	5.400576e+02	2.228372e+02	53.594610	3.885522e+02	4.778551e+02	6.310456e+02	1.70
hc_median	38140.0	5.129899e+02	2.332398e+02	53.000000	3.600000e+02	4.480000e+02	5.990000e+02	1.70
hc_stdev	38140.0	2.184086e+02	9.195354e+01	18.257420	1.538986e+02	1.985915e+02	2.662663e+02	8.20
hc_samples	38140.0	3.701282e+02	2.504009e+02	2.000000	1.920000e+02	3.270000e+02	5.010000e+02	1.13
hc_sample_weight	38140.0	2.548620e+02	1.900167e+02	0.491230	1.202828e+02	2.126803e+02	3.432932e+02	7.10
home_equity_second_mortgage	38353.0	2.572316e-02	3.108756e-02	0.000000	5.020000e-03	1.859000e-02	3.704000e-02	1.00
second_mortgage	38353.0	3.001914e-02	3.398802e-02	0.000000	7.720000e-03	2.252000e-02	4.285000e-02	1.00
home_equity	38353.0	1.010637e-01	6.963801e-02	0.000000	4.938000e-02	9.461000e-02	1.436000e-01	1.00
debt	38353.0	6.299163e-01	1.566799e-01	0.000000	5.392900e-01	6.487900e-01	7.385900e-01	1.00
second_mortgage_cdf	38353.0	4.677382e-01	2.955378e-01	0.000000	2.481300e-01	4.190400e-01	5.538100e-01	1.00
home_equity_cdf	38353.0	4.768957e-01	2.564298e-01	0.000000	2.648800e-01	4.656200e-01	6.778500e-01	1.00
debt_cdf	38353.0	4.979522e-01	2.643914e-01	0.000000	2.788400e-01	4.907200e-01	7.173900e-01	1.00
hs_degree	38755.0	8.576950e-01	1.130291e-01	0.000000	8.063650e-01	8.883700e-01	9.398050e-01	1.00
hs_degree_male	38741.0	8.512397e-01	1.213130e-01	0.000000	7.934900e-01	8.830500e-01	9.408400e-01	1.00
hs_degree_female	38702.0	8.643533e-01	1.125550e-01	0.000000	8.168025e-01	8.953050e-01	9.447600e-01	1.00
male_age_mean	38757.0	3.828283e+01	5.596338e+00	12.145830	3.499516e+01	3.829111e+01	4.134406e+01	8.33
male_age_median	38757.0	3.800188e+01	7.851792e+00	9.750000	3.275000e+01	3.783333e+01	4.283333e+01	8.33
male_age_stdev	38757.0	2.147981e+01	2.553394e+00	0.737110	2.055694e+01	2.190030e+01	2.295085e+01	3.10
male_age_sample_weight	38757.0	5.377034e+02	3.079646e+02	0.745760	3.491365e+02	4.937908e+02	6.687151e+02	1.20
male_age_samples	38757.0	2.147522e+03	1.095788e+03	3.000000	1.425000e+03	1.995000e+03	2.679000e+03	2.79
female_age_mean	38728.0	4.025749e+01	5.876502e+00	15.360240	3.683268e+01	4.032055e+01	4.354850e+01	9.01
female_age_median	38728.0	4.028816e+01	8.019936e+00	12.833330	3.483333e+01	4.050000e+01	4.533333e+01	9.01
female_age_stdev	38728.0	2.216957e+01	2.544665e+00	0.556780	2.130053e+01	2.250221e+01	2.356625e+01	3.02
female_age_sample_weight	38728.0	5.460894e+02	2.827939e+02	0.251910	3.579785e+02	5.049821e+02	6.823097e+02	6.19
female_age_samples	38728.0	2.216031e+03	1.084201e+03	2.000000	1.480000e+03	2.078000e+03	2.781000e+03	2.72
pct_own	38640.0	6.385631e-01	2.283459e-01	0.000000	5.003375e-01	6.896700e-01	8.167825e-01	1.00
married	38755.0	5.074999e-01	1.377443e-01	0.000000	4.242950e-01	5.262100e-01	6.057150e-01	1.00
married_snp	38755.0	4.766375e-02	3.795871e-02	0.000000	2.084000e-02	3.879000e-02	6.518000e-02	7.14
separated	38755.0	1.916585e-02	2.098770e-02	0.000000	4.530000e-03	1.358000e-02	2.765000e-02	7.14
divorced	38755.0	9.993071e-02	4.889849e-02	0.000000	6.542000e-02	9.491000e-02	1.287900e-01	1.00

In [9]:
```
df['sumlevel'].unique()
```

Out[9]:
```
array([140], dtype=int64)
```

- Dropping the SUMLEVEL column as

In [10]:
```
df['blockid'].unique()
```

Out[10]:
```
array([nan])
```

- Dropping the BLOCKID column as there are no values in the feature

In [11]:
```
df.drop(['blockid', 'sumlevel'], axis = 1, inplace=True)
df.head()
```

Out[11]:

	uid	countyid	stateid	state	state_ab	city	place	type	primary	zip_code	area_code	lat	lng	
0	267822	53	36	New York	NY	Hamilton	Hamilton	City	tract	13346	315	42.840812	-75.501524	202
1	246444	141	18	Indiana	IN	South Bend	Roseland	City	tract	46616	574	41.701441	-86.266614	1
2	245683	63	18	Indiana	IN	Danville	Danville	City	tract	46122	317	39.792202	-86.515246	69
3	279653	127	72	Puerto Rico	PR	San Juan	Guaynabo	Urban	tract	927	787	18.396103	-66.104169	1
4	247218	161	20	Kansas	KS	Manhattan	Manhattan City	City	tract	66502	785	39.195573	-96.569366	2

In [12]:
```
print(df.duplicated().sum())
df.drop_duplicates(inplace= True)
print(df.duplicated().sum())
```

```
192
0
```

- Checking the UID for its suitablity of becoming the index

In [13]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 38838 entries, 0 to 11708
Data columns (total 79 columns):
 #   Column                         Non-Null Count  Dtype
---  ------                         --------------  -----
 0   uid                            38838 non-null  int64
 1   countyid                       38838 non-null  int64
 2   stateid                        38838 non-null  int64
 3   state                          38838 non-null  object
 4   state_ab                       38838 non-null  object
 5   city                           38838 non-null  object
 6   place                          38838 non-null  object
 7   type                           38838 non-null  object
 8   primary                        38838 non-null  object
 9   zip_code                       38838 non-null  int64
 10  area_code                      38838 non-null  int64
 11  lat                            38838 non-null  float64
 12  lng                            38838 non-null  float64
 13  aland                          38838 non-null  float64
 14  awater                         38838 non-null  int64
 15  pop                            38838 non-null  int64
 16  male_pop                       38838 non-null  int64
 17  female_pop                     38838 non-null  int64
 18  rent_mean                      38462 non-null  float64
 19  rent_median                    38462 non-null  float64
 20  rent_stdev                     38462 non-null  float64
 21  rent_sample_weight             38462 non-null  float64
 22  rent_samples                   38462 non-null  float64
 23  rent_gt_10                     38461 non-null  float64
 24  rent_gt_15                     38461 non-null  float64
 25  rent_gt_20                     38461 non-null  float64
 26  rent_gt_25                     38461 non-null  float64
 27  rent_gt_30                     38461 non-null  float64
 28  rent_gt_35                     38461 non-null  float64
 29  rent_gt_40                     38461 non-null  float64
 30  rent_gt_50                     38461 non-null  float64
 31  universe_samples               38838 non-null  int64
 32  used_samples                   38838 non-null  int64
 33  hi_mean                        38519 non-null  float64
 34  hi_median                      38519 non-null  float64
 35  hi_stdev                       38519 non-null  float64
 36  hi_sample_weight               38519 non-null  float64
 37  hi_samples                     38519 non-null  float64
 38  family_mean                    38483 non-null  float64
 39  family_median                  38483 non-null  float64
 40  family_stdev                   38483 non-null  float64
 41  family_sample_weight           38483 non-null  float64
 42  family_samples                 38483 non-null  float64
 43  hc_mortgage_mean               38151 non-null  float64
 44  hc_mortgage_median             38151 non-null  float64
 45  hc_mortgage_stdev              38151 non-null  float64
 46  hc_mortgage_sample_weight      38151 non-null  float64
 47  hc_mortgage_samples            38151 non-null  float64
 48  hc_mean                        38093 non-null  float64
 49  hc_median                      38093 non-null  float64
 50  hc_stdev                       38093 non-null  float64
 51  hc_samples                     38093 non-null  float64
 52  hc_sample_weight               38093 non-null  float64
 53  home_equity_second_mortgage    38274 non-null  float64
 54  second_mortgage                38274 non-null  float64
 55  home_equity                    38274 non-null  float64
 56  debt                           38274 non-null  float64
 57  second_mortgage_cdf            38274 non-null  float64
 58  home_equity_cdf                38274 non-null  float64
 59  debt_cdf                       38274 non-null  float64
 60  hs_degree                      38615 non-null  float64
 61  hs_degree_male                 38602 non-null  float64
 62  hs_degree_female               38571 non-null  float64
 63  male_age_mean                  38613 non-null  float64
 64  male_age_median                38613 non-null  float64
 65  male_age_stdev                 38613 non-null  float64
 66  male_age_sample_weight         38613 non-null  float64
 67  male_age_samples               38613 non-null  float64
 68  female_age_mean                38590 non-null  float64
 69  female_age_median              38590 non-null  float64
 70  female_age_stdev               38590 non-null  float64
 71  female_age_sample_weight       38590 non-null  float64
 72  female_age_samples             38590 non-null  float64
 73  pct_own                        38519 non-null  float64
```

```
 74  married                     38611 non-null  float64
 75  married_snp                 38611 non-null  float64
 76  separated                   38611 non-null  float64
 77  divorced                    38611 non-null  float64
 78  split                       38838 non-null  object
dtypes: float64(61), int64(11), object(7)
memory usage: 23.7+ MB
```

In [14]:
```
df.drop_duplicates('uid', inplace= True)
df.set_index('uid', inplace=True)
df.sort_index(inplace=True)
df.head(10)
```

Out[14]:

uid	countyid	stateid	state	state_ab	city	place	type	primary	zip_code	area_code	lat	lng	
220336	16	2	Alaska	AK	Unalaska	Unalaska City	City	tract	99685	907	53.621091	-166.770979	2.82318
220342	20	2	Alaska	AK	Eagle River	Anchorage	City	tract	99577	907	61.174250	-149.284329	5.09234
220343	20	2	Alaska	AK	Jber	Anchorage	City	tract	99505	907	61.284745	-149.653973	2.70593
220345	20	2	Alaska	AK	Anchorage	Point Mackenzie	City	tract	99501	907	61.229560	-149.893037	2.37151
220347	20	2	Alaska	AK	Anchorage	Anchorage	City	tract	99504	907	61.217082	-149.767214	1.97923
220348	20	2	Alaska	AK	Anchorage	Anchorage	City	tract	99504	907	61.217507	-149.744426	2.08977
220349	20	2	Alaska	AK	Anchorage	Anchorage	City	tract	99504	907	61.223372	-149.723327	2.93637
220350	20	2	Alaska	AK	Anchorage	Point Mackenzie	City	tract	99508	907	61.216701	-149.792744	2.49442
220352	20	2	Alaska	AK	Anchorage	Point Mackenzie	City	tract	99508	907	61.215067	-149.836271	4.46747
220353	20	2	Alaska	AK	Anchorage	Point Mackenzie	City	tract	99501	907	61.204383	-149.852526	1.14615

Question 3

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.

In [15]:
```
print(">>>Total number of columns with null values\n...", end = ' ')
print((df.isnull().sum() != 0).sum())
print(">>>Total number of Numerical columns with null values\n...", end = ' ')
print((df.select_dtypes('number').isnull().sum() != 0).sum())
print(">>>Total number of Object columns with null values\n...", end = ' ')
print((df.select_dtypes('object').isnull().sum() != 0).sum())
print(">>>Total number of missing values\n...", end = ' ')
print(df.isnull().sum().sum())
```

```
>>>Total number of columns with null values
... 58
>>>Total number of Numerical columns with null values
... 58
>>>Total number of Object columns with null values
... 0
>>>Total number of missing values
... 20328
```

- As the number of missing values are bigger, we will go *KNN Imputation* method

In [16]:
```
imputer = KNNImputer(n_neighbors= 1)
# Getting the columns of dataframe which have null values and imputing the values
for i in df.columns:
    if df[i].isna().sum() != 0:
        df[i] = imputer.fit_transform(pd.DataFrame(df[i]))


print(">>>Total number of columns with null values\n...", end = ' ')
print((df.isnull().sum() != 0).sum())
print(">>>Total number of Numerical columns with null values\n...", end = ' ')
```

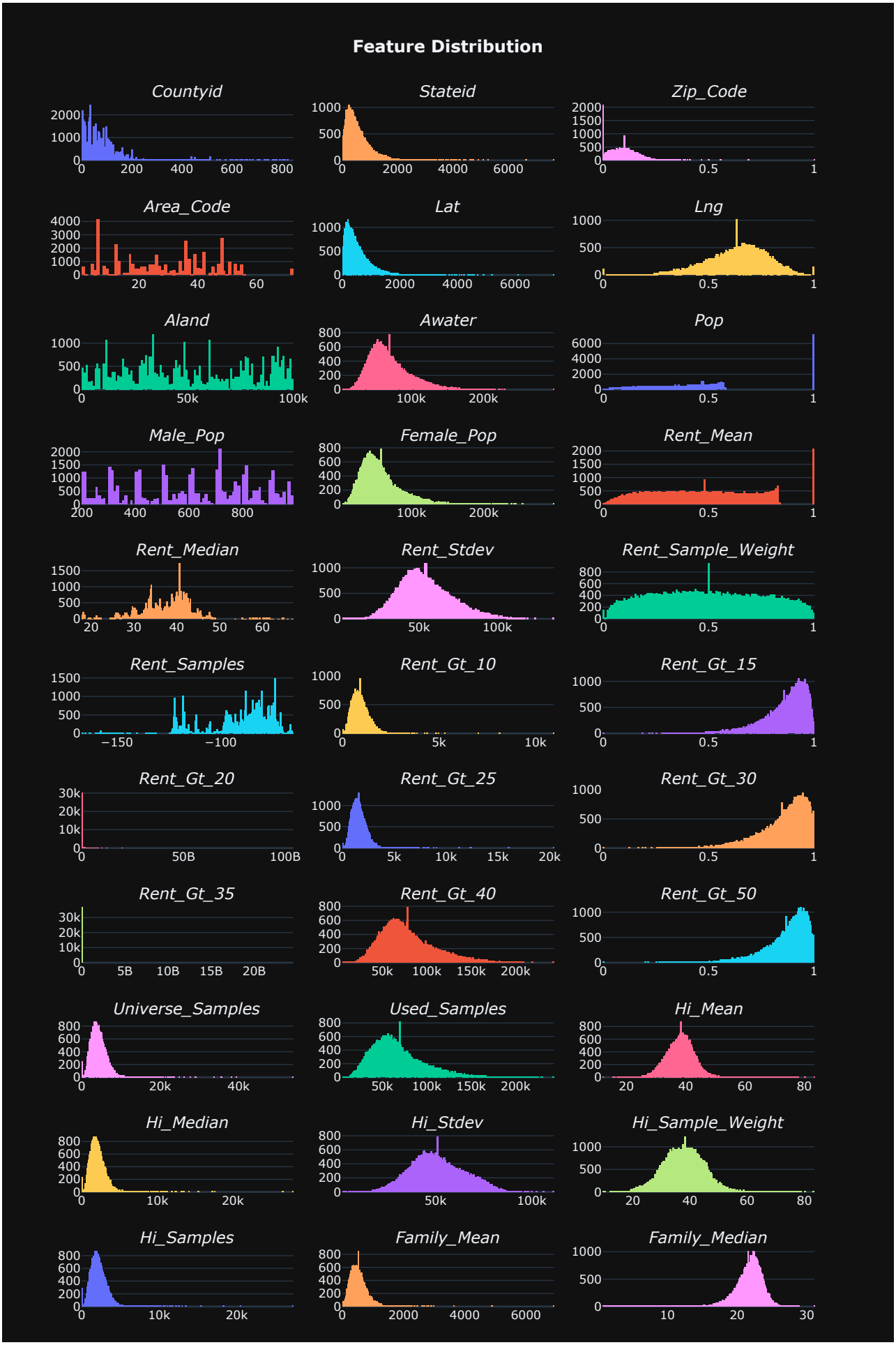
```
print((df.select_dtypes('number').isnull().sum() != 0).sum())
print(">>>Total number of Object columns with null values\n...", end = ' ')
print((df.select_dtypes('object').isnull().sum() != 0).sum())
print(">>>Total number of missing values\n...", end = ' ')
print(df.isnull().sum().sum())
print(">>>Shape of dataframe\n...", end = ' ')
print(df.shape)
```

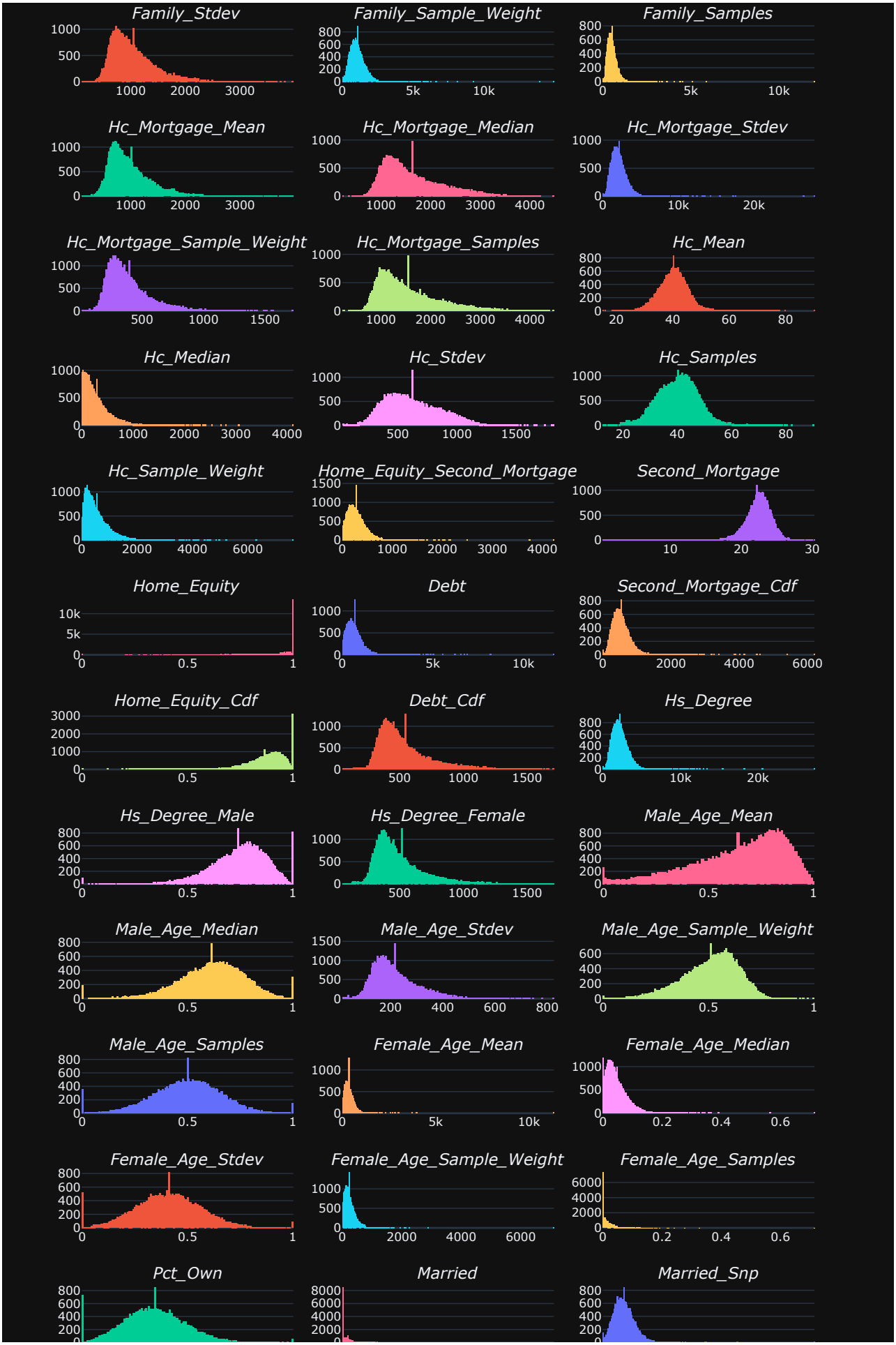
```
>>>Total number of columns with null values
... 0
>>>Total number of Numerical columns with null values
... 0
>>>Total number of Object columns with null values
... 0
>>>Total number of missing values
... 0
>>>Shape of dataframe
... (38715, 78)
```

Exploratory Data Analysis (EDA):

In [17]:
```
feature_list = df.select_dtypes('number').columns
```

In [18]:
```
l = len(feature_list)

row = l // 3 +1

fig = make_subplots(cols = 3,
                    rows = row,
                    subplot_titles=[f'<i>{i.title()}</i>' for i in feature_list])
for i in range(l):
    fig.add_trace(go.Histogram(x = df[feature_list[i]]),
                  col = i //  row +1,
                  row = i % row +1)

fig.update_layout(height = 2900,
                  width = 900,
                  title_text = '<b>Feature Distribution',
                  template = 'plotly_dark',
                  title_x = 0.5,
                  showlegend=False)


fig.show()
```


Feature Distribution
Countyid
Stateid
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

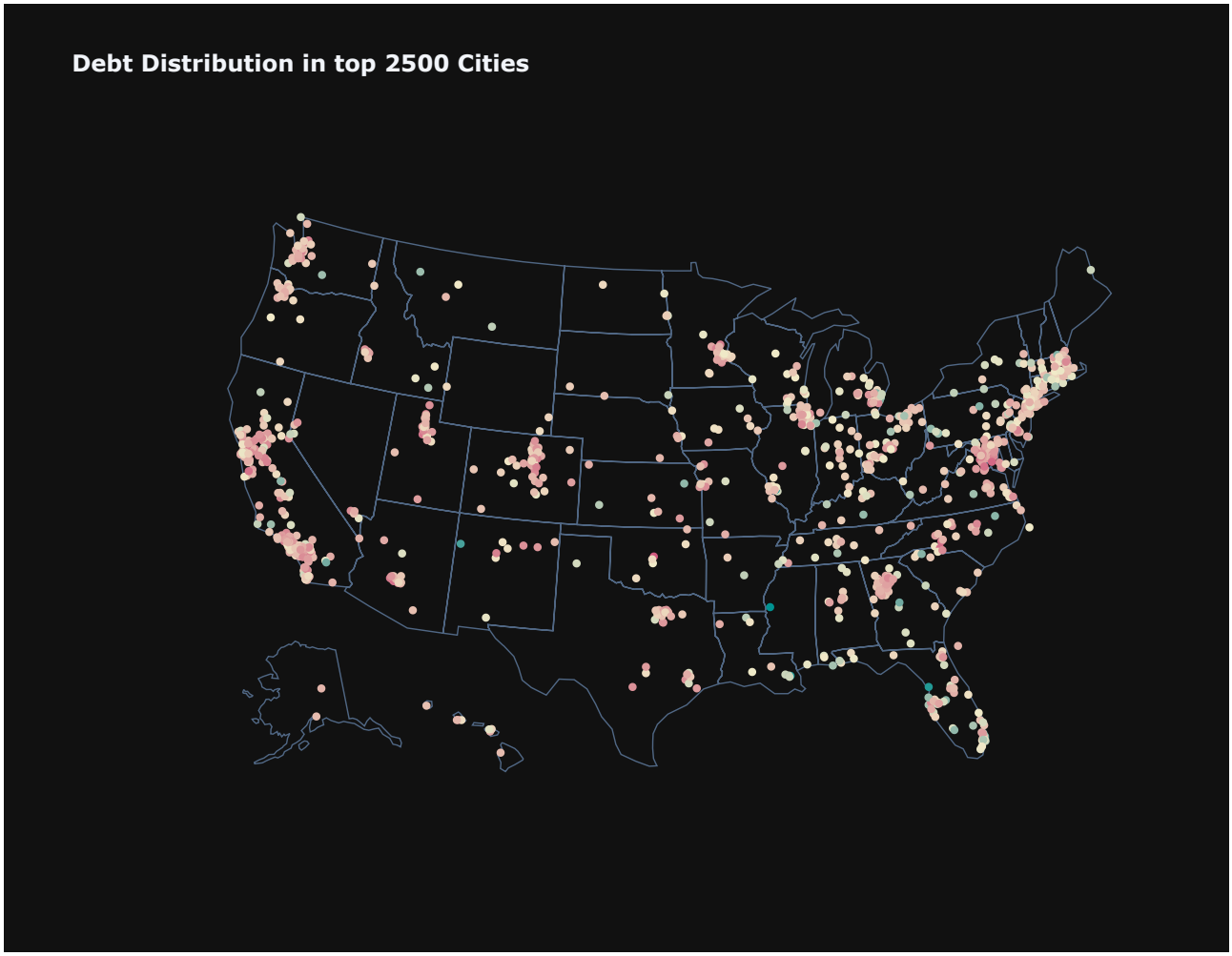
Question 4

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

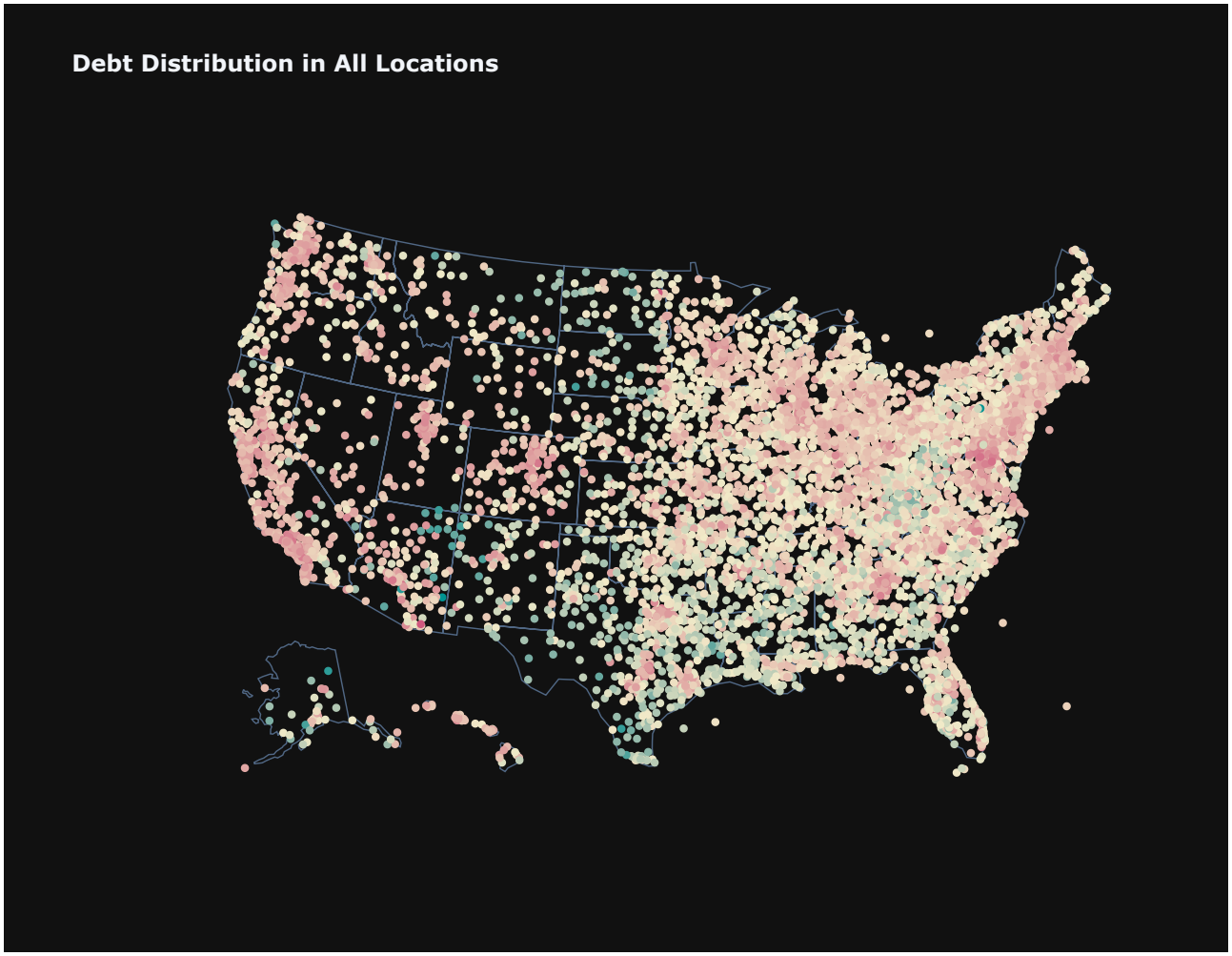
In [19]:
```
df_debt = df.sort_values(by = 'second_mortgage', ascending=False)
df_debt = df_debt[df_debt['pct_own']>0.1].iloc[0:2500,]
df_debt.shape
```

Out[19]: (2500, 78)

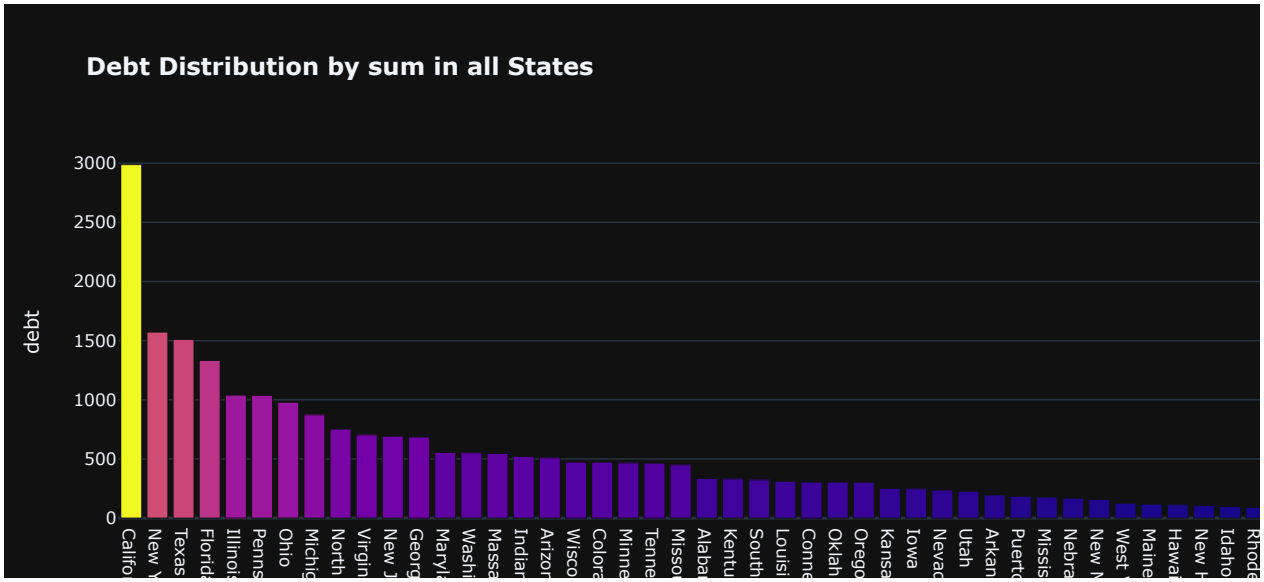
In [20]:
```
df_by_city_2500 = df_debt.select_dtypes('number').groupby(df['city']).mean()

debt_scatter = px.scatter_geo(data_frame=df_by_city_2500,
               lat = 'lat',
               lon= 'lng',
               color = 'debt',
               hover_name=df_by_city_2500.index,
               height = 700,
               width = 1000,
               template = 'plotly_dark',
               color_continuous_scale='tealrose',
               title = '<b>Debt Distribution in top 2500 Cities</b>',
               scope = 'usa')


debt_scatter.show()
```



In [21]:
```
df_by_city = df.select_dtypes('number').groupby(df['city']).mean()

debt_scatter = px.scatter_geo(data_frame=df_by_city,
               lat = 'lat',
               lon= 'lng',
               color = 'debt',
               hover_name=df_by_city.index,
               height = 700,
               width = 1000,
               template = 'plotly_dark',
               title = '<b>Debt Distribution in All Locations</b>',
               color_continuous_scale='tealrose',
               scope = 'usa')
debt_scatter.show()
```



In [22]:
```
df_by_state = df.groupby('state').sum().sort_values(by = 'debt', ascending=False)

debt_scatter = px.bar(data_frame=df_by_state,
                      y = 'debt',
                color = 'debt',
                hover_name=df_by_state.index,
                title = '<b>Debt Distribution by sum in all States</b>',
                template = 'plotly_dark')
debt_scatter.show()
```



Use the following bad debt equation:

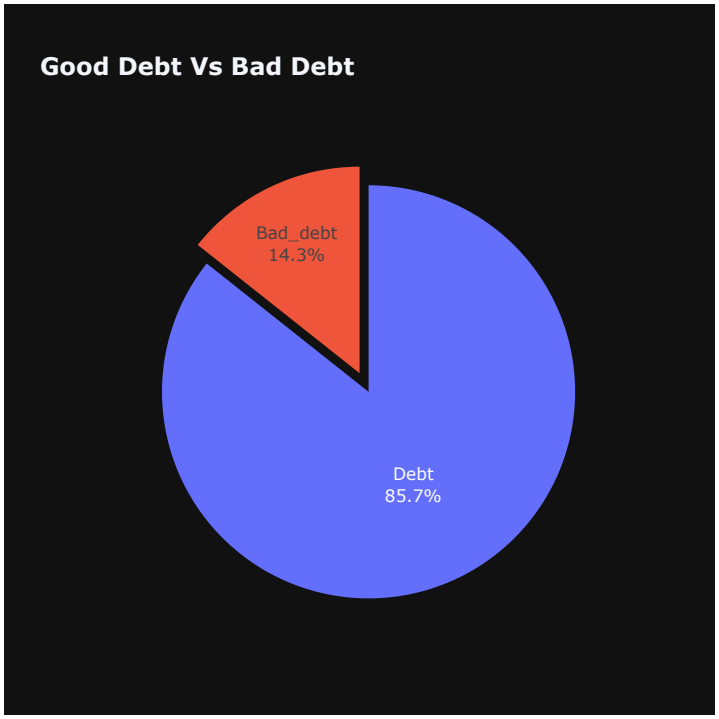
Bad Debt = P (Second Mortgage ∩ Home Equity Loan)
Bad Debt = second_mortgage + home_equity - home_equity_second_mortgage
Create pie charts to show overall debt and bad debt

In [23]:
```
df['bad_debt'] = df['second_mortgage'] + df['home_equity'] - df['home_equity_second_mortgage']

df.head()
```

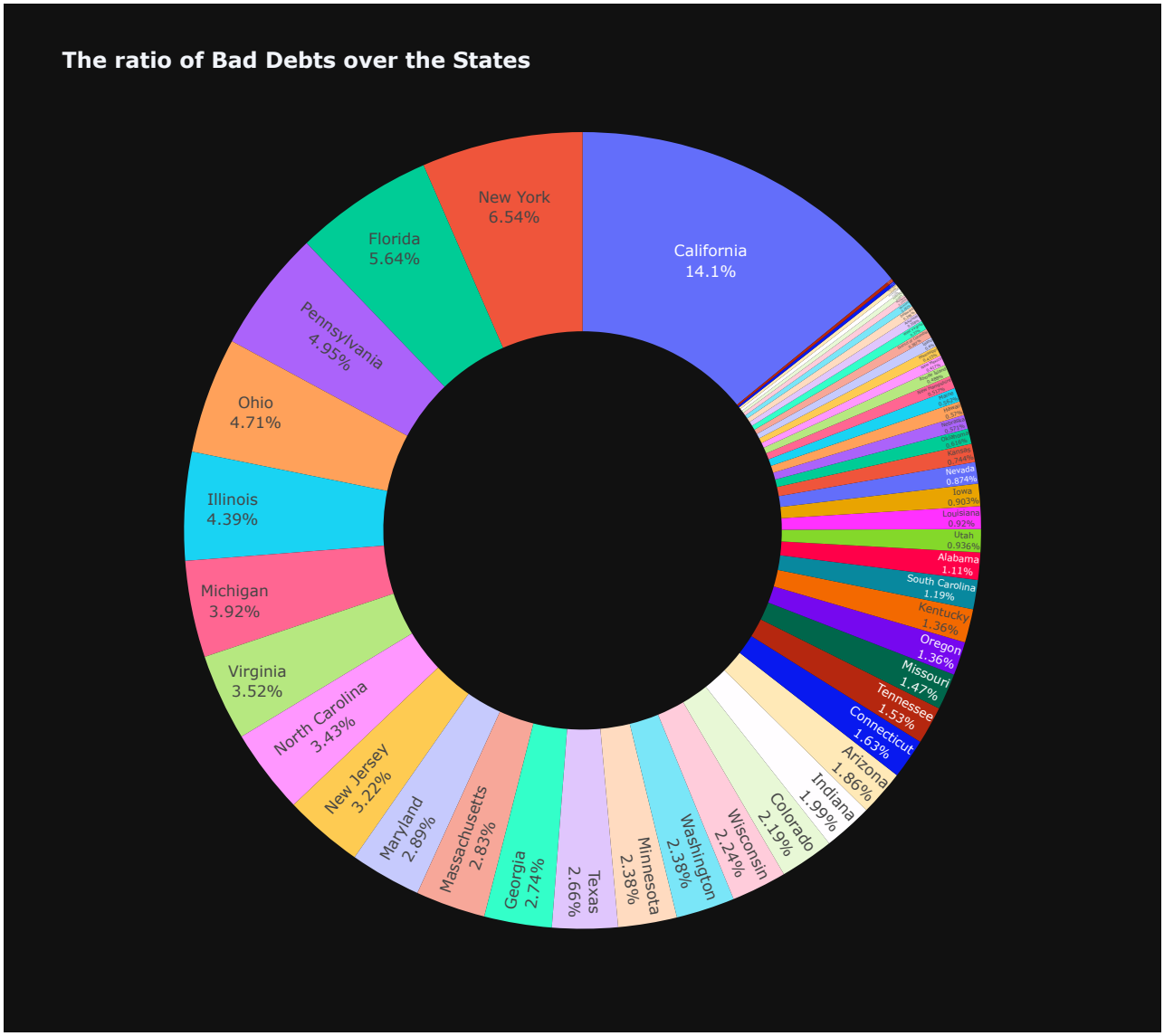
Out[23]:

uid	countyid	stateid	state	state_ab	city	place	type	primary	zip_code	area_code	lat	lng	
220336	16	2	Alaska	AK	Unalaska	Unalaska City	City	tract	99685	907	53.621091	-166.770979	2.82318
220342	20	2	Alaska	AK	Eagle River	Anchorage	City	tract	99577	907	61.174250	-149.284329	5.09234
220343	20	2	Alaska	AK	Jber	Anchorage	City	tract	99505	907	61.284745	-149.653973	2.70593
220345	20	2	Alaska	AK	Anchorage	Point Mackenzie	City	tract	99501	907	61.229560	-149.893037	2.37151
220347	20	2	Alaska	AK	Anchorage	Anchorage	City	tract	99504	907	61.217082	-149.767214	1.97923

In [24]:
```
labels = 'Debt', 'Bad_debt'
sizes = [df['debt'].mean()*100, df['bad_debt'].mean()*100]
df_debt_ratio = pd.DataFrame(sizes, index = labels)
#Plot
fig_debt = px.pie(data_frame=df_debt_ratio,
       values = 0,
       names = df_debt_ratio.index,
       height = 500,
       width = 500,
       title = '<b>Good Debt Vs Bad Debt</b>',
       template = 'plotly_dark',
       )
fig_debt.update_traces(textposition='inside', textinfo='percent+label', showlegend = False)
# Explode the pie chart
fig_debt.update_traces(pull=[0.1, 0, 0])
fig_debt.show()
```



In [25]:
```
pie_bad_debt = px.pie(data_frame = df,
                      values = 'bad_debt',
                      names = 'state',
                      height = 800,
                      width = 900,
                      hole = 0.5,
                      template = 'plotly_dark',
                      title = '<b>The ratio of Bad Debts over the States'
                      )
pie_bad_debt.update_traces(textposition='inside', textinfo='percent+label', showlegend = False)

pie_bad_debt.show()
```



In [26]:
```
gdf = gpd.GeoDataFrame(df_debt,
                       geometry=gpd.points_from_xy(x=df_debt.lng,
                                                   y=df_debt.lat)) # ignore the error as it is not really an error but a bug
gdf.head()
```

Out[26]:

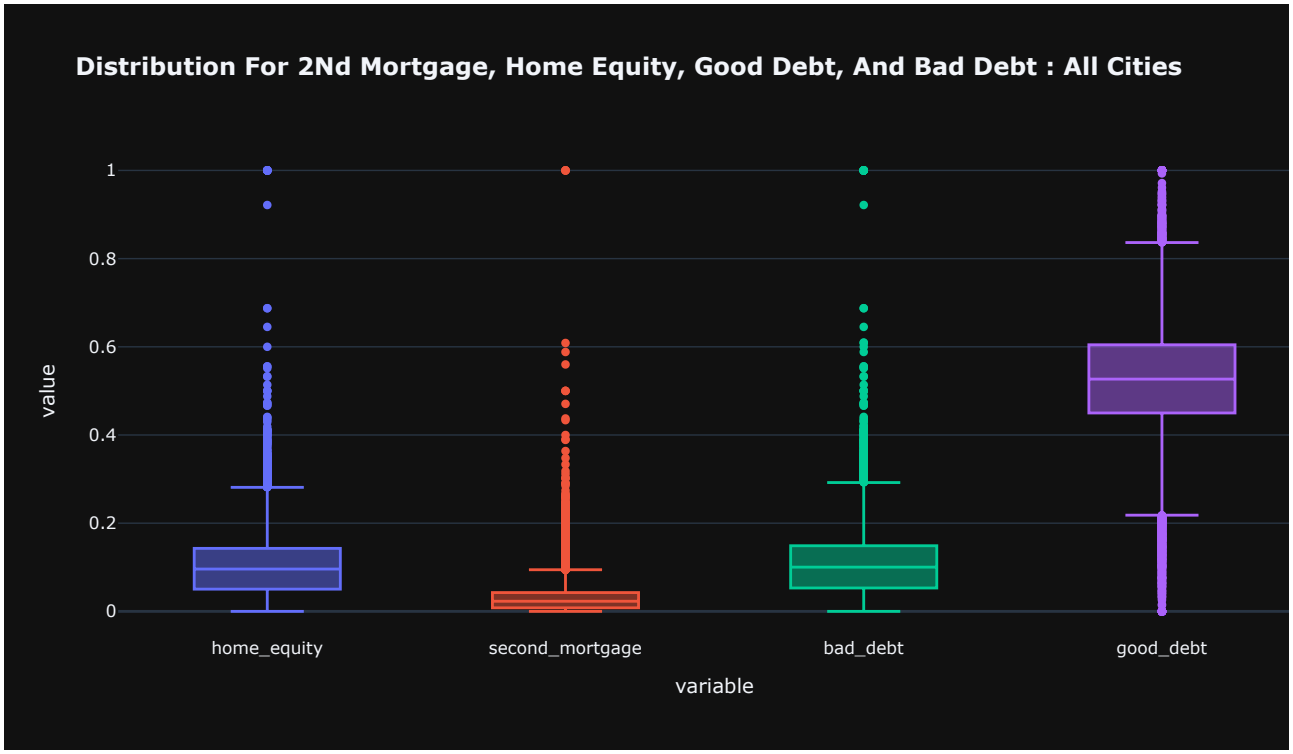
uid	countyid	stateid	state	state_ab	city	place	type	primary	zip_code	area_code	lat	lng	
289712	147	51	Virginia	VA	Farmville	Farmville	Town	tract	23901	434	37.297357	-78.396452	4
251185	27	25	Massachusetts	MA	Worcester	Worcester City	City	tract	1610	508	42.254262	-71.800347	7
274394	109	40	Oklahoma	OK	Edmond	Edmond City	CDP	tract	73003	405	35.655695	-97.471535	13
269323	81	36	New York	NY	Corona	Harbor Hills	City	tract	11368	718	40.751809	-73.853582	1
251324	3	24	Maryland	MD	Glen Burnie	Glen Burnie	CDP	tract	21061	410	39.127273	-76.635265	11

In [ ]:

Question 5

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

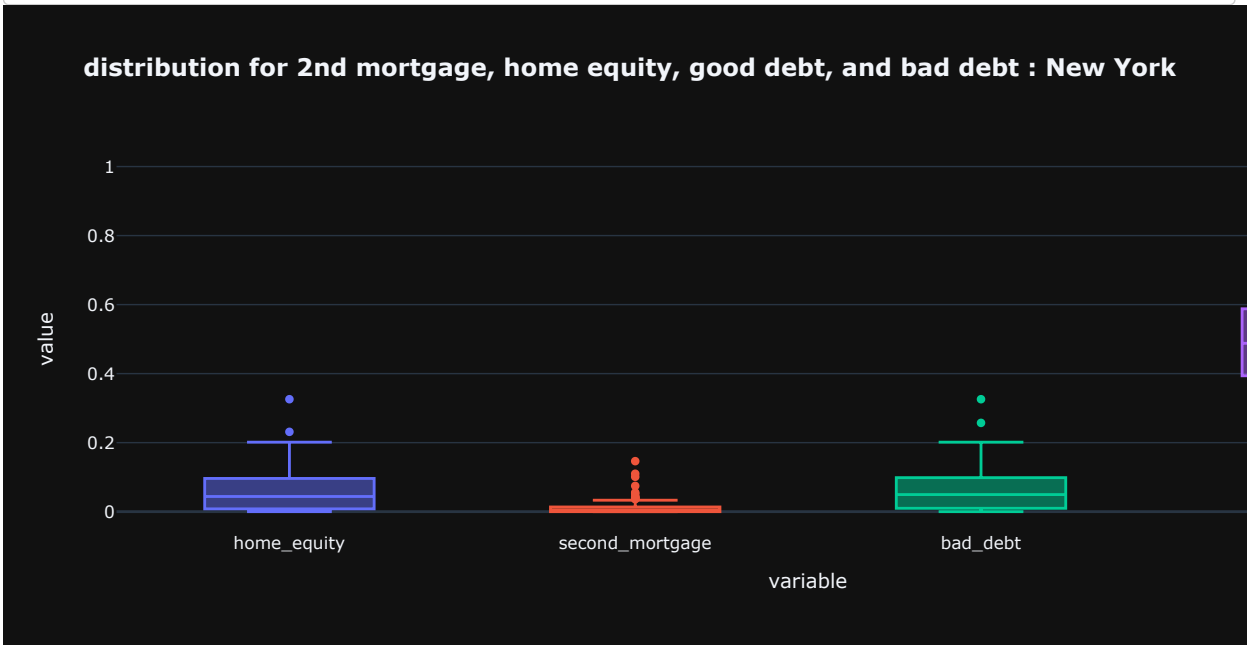
In [27]:
```
df['good_debt']=df['debt']-df['bad_debt']

all_cities = df[['home_equity','second_mortgage','bad_debt', 'good_debt']]

fig_dist = px.box(all_cities,
                  width = 1000,
                  template = 'plotly_dark',
                  color = 'variable',
                  title = '<b>distribution for 2nd mortgage, home equity, good debt, and bad debt : ALL CITIES</b>'.title())

fig_dist.update_traces(showlegend = False)

fig_dist.show()
```



In [28]:
```
app = Dash(__name__)

app.layout = html.Div([
    dcc.Dropdown(df.city.unique(), 'New York', id='dropdown-selection'),
    dcc.Graph(id='graph-content')
])

@callback(
    Output('graph-content', 'figure'),
    Input('dropdown-selection', 'value')
)
def update_graph(value):
    dff_value = df[df['city']==value]
    dff_value = dff_value[['home_equity','second_mortgage','bad_debt', 'good_debt']]

    dff_value_box = px.box(dff_value,
                  template = 'plotly_dark',
                  color = 'variable',
                  title = f'<b>distribution for 2nd mortgage, home equity, good debt, and bad debt : {value}</b>')
    dff_value_box.update_traces(showlegend = False)


    return dff_value_box
```

```
if __name__ == '__main__':
    app.run(debug=True)
```

New York

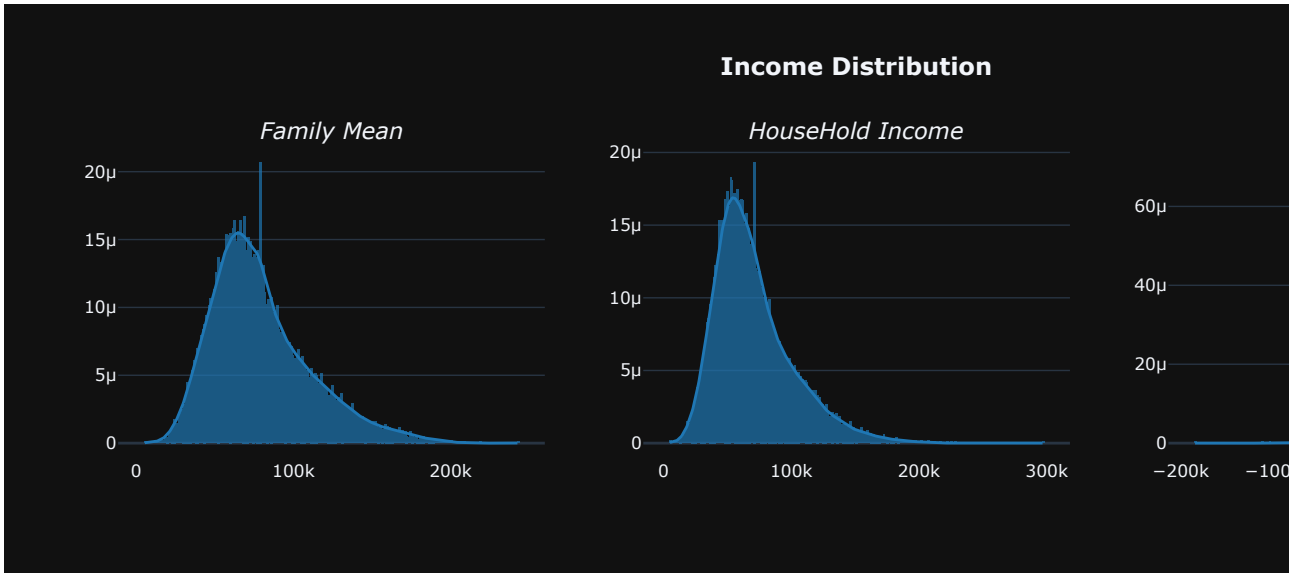


Question 6

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

In [29]:
```
title_list = ['Family Mean', 'HouseHold Income', 'Other Incomes']
row = 1

fig_income_dist = make_subplots(cols = 3,
                    rows = row,
                    subplot_titles=[f'<i>{i}</i>' for i in title_list])

dist_family_mean = ff.create_distplot([df['family_mean']],
                                      ['Family'],
                                      bin_size=1000)
for trace in dist_family_mean.select_traces():
    fig_income_dist.add_trace(trace, row=1, col=1)

dist_houshold_mean = ff.create_distplot([df['hi_mean']],
                                        ['HouseHold'],
                                        bin_size=1000)
for trace in dist_houshold_mean.select_traces():
    fig_income_dist.add_trace(trace, row=1, col=2)

dist_other_mean = ff.create_distplot([df['family_mean'] - df['hi_mean']],
                                     ['Family Other Incomes'],
                                     bin_size=1000)
for trace in dist_other_mean.select_traces():
    fig_income_dist.add_trace(trace, row=1, col=3)

fig_income_dist.update_layout(height = 400,
                width = 1200,
                title_text = '<b>Income Distribution',
                template = 'plotly_dark',
```

```
                title_x = 0.5,
                showlegend=False)


fig_income_dist.show()
```



Question 7

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)

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

In [30]:
```
# Getting the population density per square kilometer
df['population_density'] = (df['pop']/df['aland']) * 1000000
df.head()
```

Out[30]:

uid	countyid	stateid	state	state_ab	city	place	type	primary	zip_code	area_code	lat	lng	
220336	16	2	Alaska	AK	Unalaska	Unalaska City	City	tract	99685	907	53.621091	-166.770979	2.82318
220342	20	2	Alaska	AK	Eagle River	Anchorage	City	tract	99577	907	61.174250	-149.284329	5.09234
220343	20	2	Alaska	AK	Jber	Anchorage	City	tract	99505	907	61.284745	-149.653973	2.70593
220345	20	2	Alaska	AK	Anchorage	Point Mackenzie	City	tract	99501	907	61.229560	-149.893037	2.37151
220347	20	2	Alaska	AK	Anchorage	Anchorage	City	tract	99504	907	61.217082	-149.767214	1.97923

Use male_age_median, female_age_median, male_pop, and female_pop to create a new field called median age

In [31]:
```
df['median_age'] = ((
    df['male_age_median'] * df['male_pop']) + (
        df['female_age_median'] * df['female_pop'])) / (
            df['male_pop'] + df['female_pop'])
df.head()
```

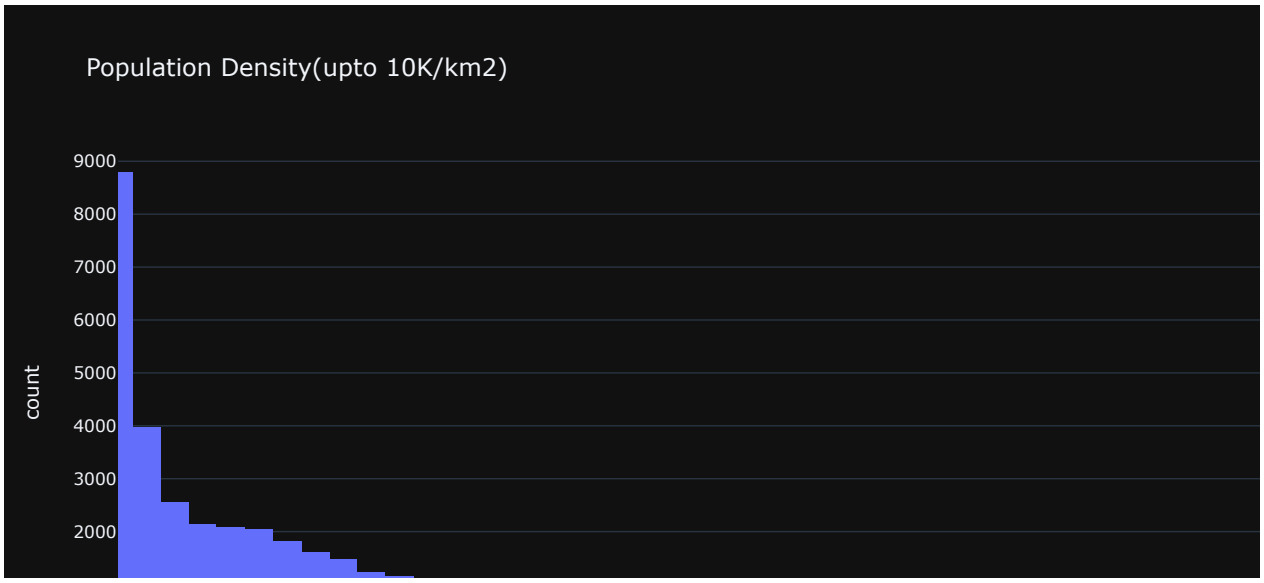
Out[31]:

uid	countyid	stateid	state	state_ab	city	place	type	primary	zip_code	area_code	lat	lng	
220336	16	2	Alaska	AK	Unalaska	Unalaska City	City	tract	99685	907	53.621091	-166.770979	2.82318
220342	20	2	Alaska	AK	Eagle River	Anchorage	City	tract	99577	907	61.174250	-149.284329	5.09234
220343	20	2	Alaska	AK	Jber	Anchorage	City	tract	99505	907	61.284745	-149.653973	2.70593
220345	20	2	Alaska	AK	Anchorage	Point Mackenzie	City	tract	99501	907	61.229560	-149.893037	2.37151
220347	20	2	Alaska	AK	Anchorage	Anchorage	City	tract	99504	907	61.217082	-149.767214	1.97923

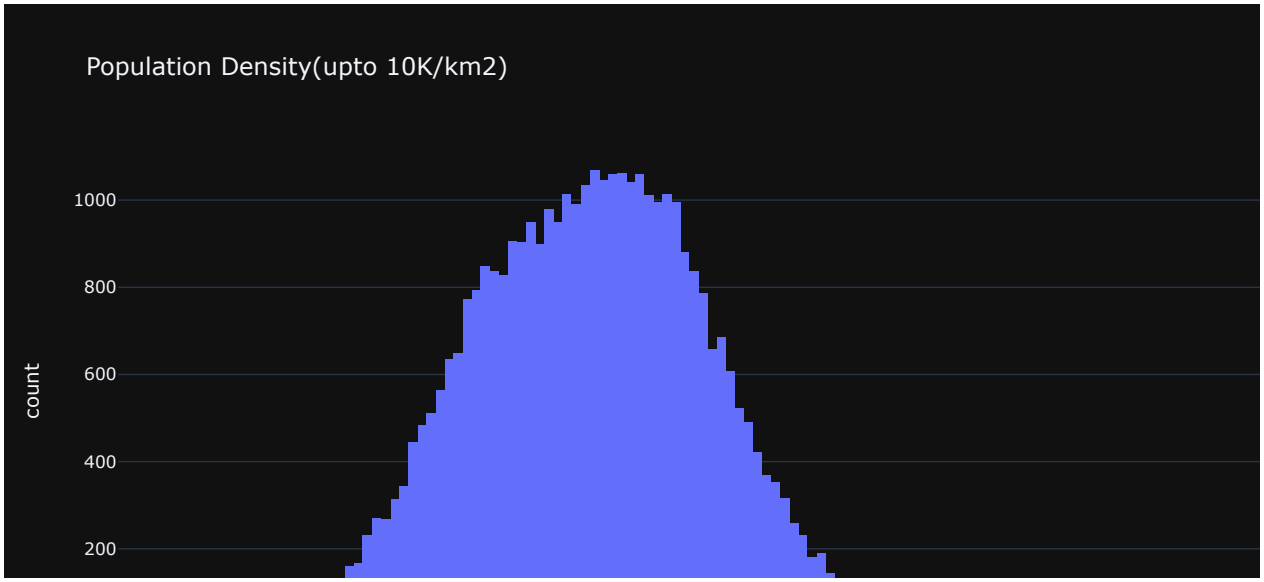
In [32]:
```
df.dropna(inplace = True)
```

Visualize the findings using appropriate chart type

In [33]:
```
px.histogram(data_frame=df,
             x = 'population_density',
             template = 'plotly_dark',
             range_x=[0, 10000],
             title = 'Population Density(upto 10K/km2)')
```



In [34]:
```
px.histogram(data_frame=df,
             x = 'median_age',
             template = 'plotly_dark',
             title = 'Population Density(upto 10K/km2)')
```



Question 8

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.

In [35]:
```
df['pop_bins'] = pd.cut(df['pop'],
                        bins=5,
                        labels=['very low',
                                'low',
                                'medium',
                                'high',
                                'very high'])
df['pop_bins'].value_counts()
```

Out[35]:
```
pop_bins
very low     38166
low            347
medium          12
high             4
very high        1
Name: count, dtype: int64
```

Analyze the married, separated, and divorced population for these population brackets

In [36]:
```
df.groupby(by = 'pop_bins')[['married',
                             'separated',
                             'divorced']].count()
```

Out[36]:

pop_bins	married	separated	divorced
very low	38166	38166	38166
low	347	347	347
medium	12	12	12
high	4	4	4
very high	1	1	1

Visualize using appropriate chart type

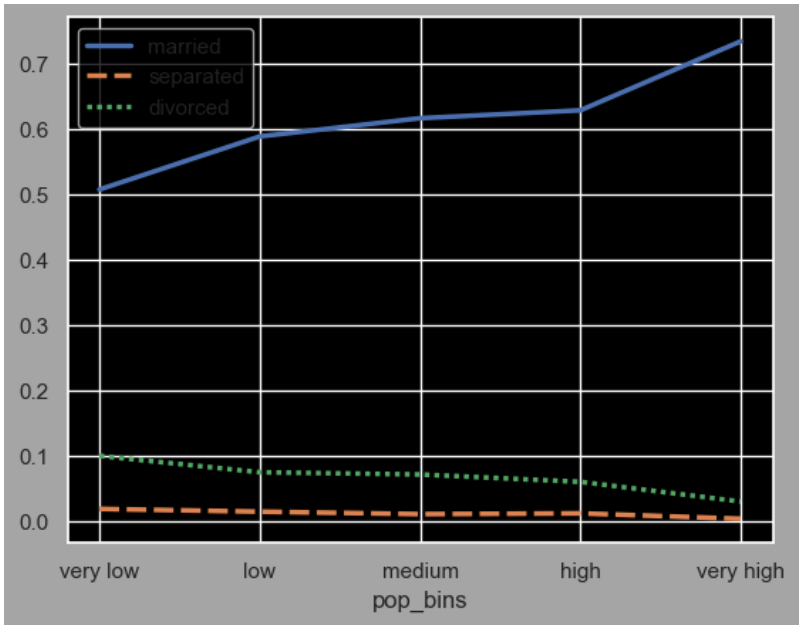
In [ ]:

```
```

In [37]:

```
pop_bin_married=df.groupby(by='pop_bins')[['married','separated','divorced']].mean()
sns.lineplot(pop_bin_married,
             linewidth=2.5
             )
```

Out[37]: <Axes: xlabel='pop_bins'>



Question 9

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

In [38]:

```
rent_state_mean = df.groupby(by='state')['rent_mean'].agg(["mean"])
income_state_mean=df.groupby(by='state')['family_mean'].agg(["mean"])
```

In [39]:

```
rent_perc_of_income=rent_state_mean['mean']/income_state_mean['mean']*100
rent_perc_of_income.head()
```

Out[39]:

```
state
Alabama       1.172225
Alaska        1.294831
Arizona       1.486834
Arkansas      1.115344
California    1.672457
Name: mean, dtype: float64
```

In [40]:

```
rent_perc_of_income=rent_state_mean['mean']/income_state_mean['mean']*100
rent_perc_of_income.head()
```

Out[40]:

```
state
Alabama       1.172225
Alaska        1.294831
Arizona       1.486834
Arkansas      1.115344
California    1.672457
Name: mean, dtype: float64
```

In [41]:

```
sum(df['rent_mean']) / sum(df['family_mean'])
```

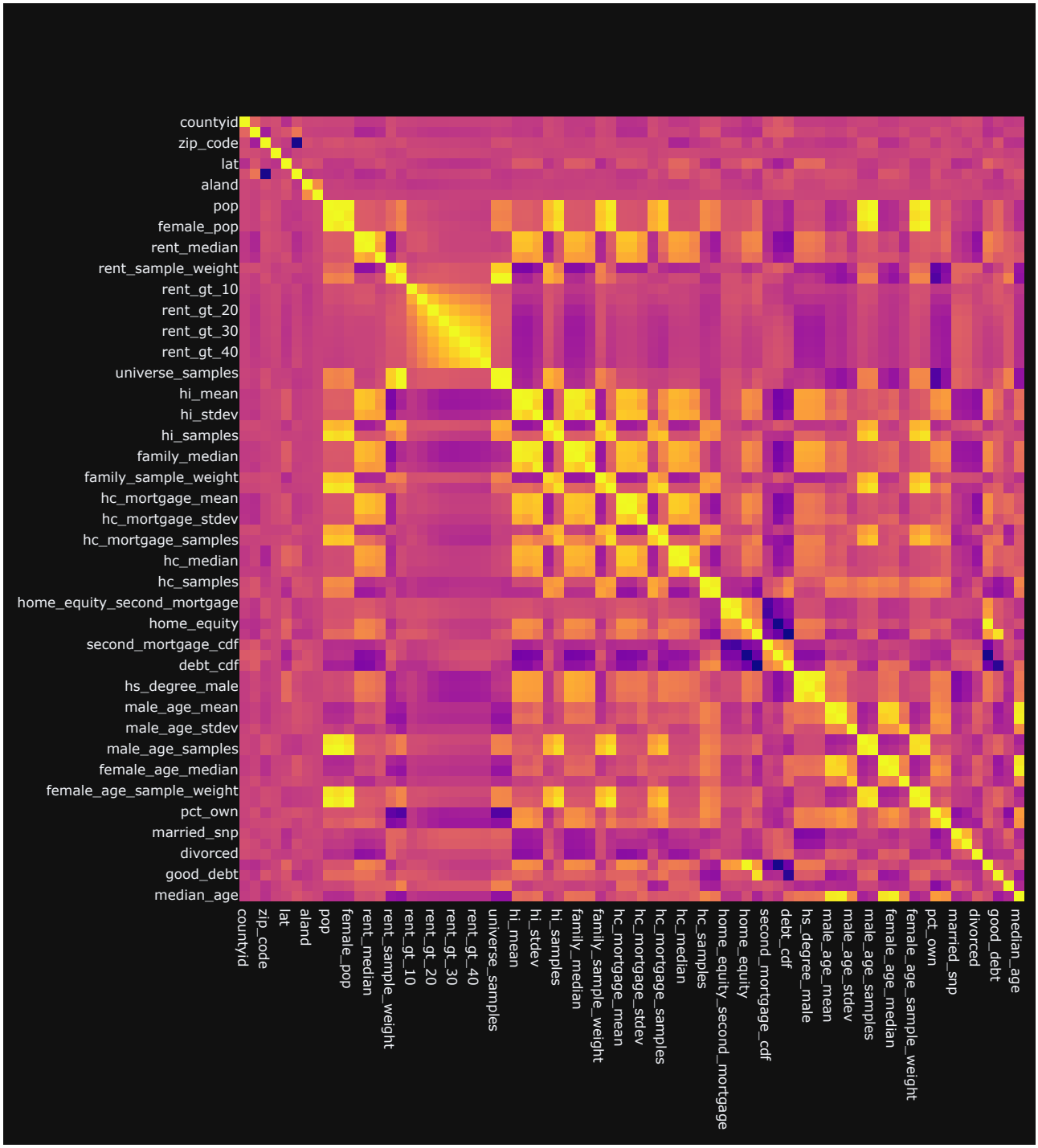
Out[41]:

```
0.01335408695373265
```

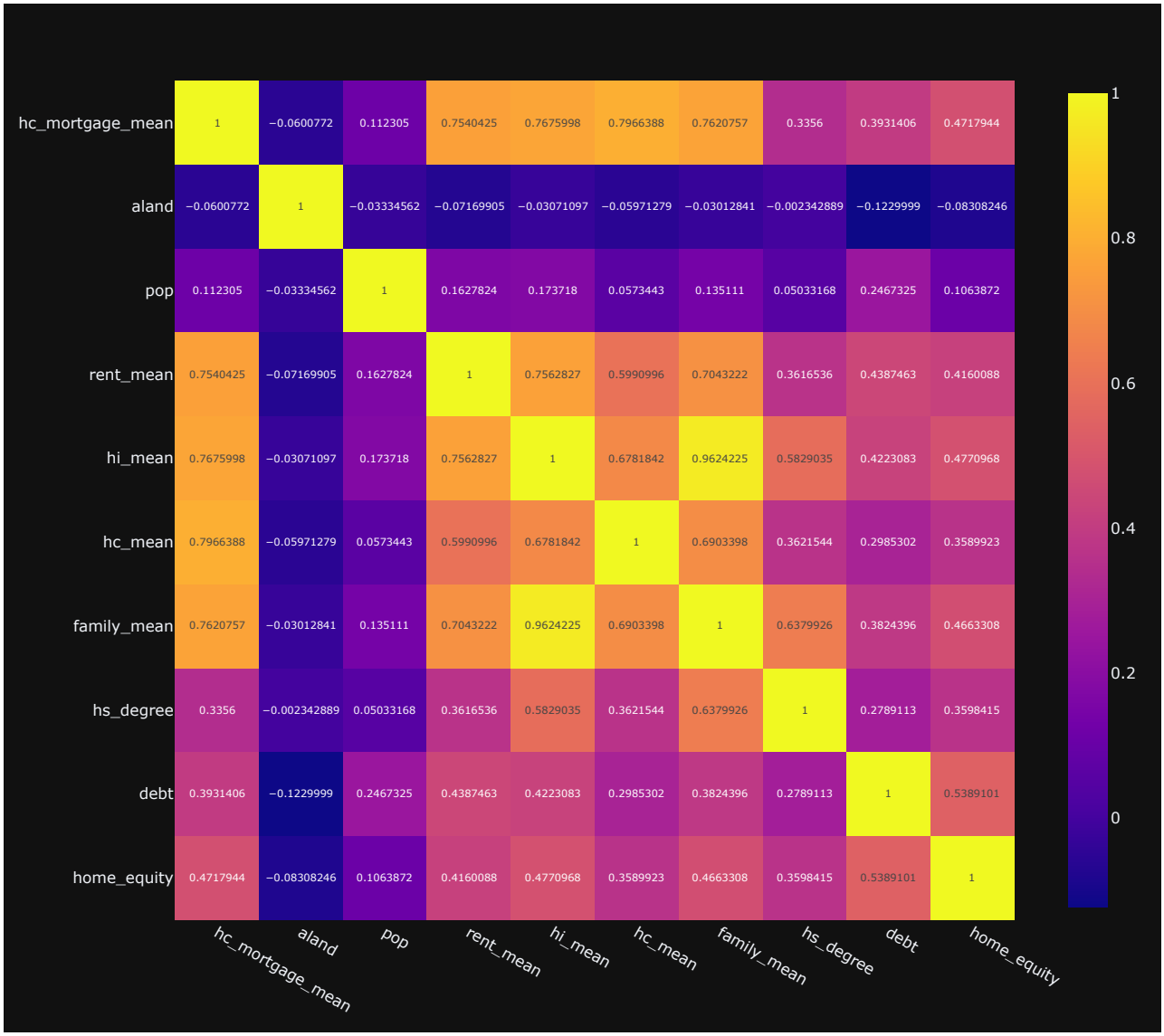
Question 10

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

In [42]:
```
px.imshow(df.select_dtypes('number').corr(),
          height = 1000,
          width = 1000,
          template = 'plotly_dark')
```



In [43]:
```
px.imshow(df[['hc_mortgage_mean', 'aland', 'pop', 'rent_mean', 'hi_mean', 'hc_mean', 'family_mean', 'hs_degree', 'debt', 'ho
          text_auto = True,
          template = 'plotly_dark',
          height = 800,
          width = 900)
```



Building Model

Data Pre-processing:

LabelEncoding

In [44]:
```
le = LabelEncoder()
```

In [45]:
```
# Fitting the Label Encoder
df['type'] = le.fit_transform(df['type'])
```

Question 11

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.

In [46]:
```
# Splitting the train test data back
train = df[df['split'] == 'Train']
test = df[df['split'] == 'Test']
```

In [47]:
```
train.head()
```

Out[47]:

uid	countyid	stateid	state	state_ab	city	place	type	primary	zip_code	area_code	lat	lng	a
220342	20	2	Alaska	AK	Eagle River	Anchorage	2	tract	99577	907	61.174250	-149.284329	5092348
220345	20	2	Alaska	AK	Anchorage	Point Mackenzie	2	tract	99501	907	61.229560	-149.893037	23715
220347	20	2	Alaska	AK	Anchorage	Anchorage	2	tract	99504	907	61.217082	-149.767214	19792
220348	20	2	Alaska	AK	Anchorage	Anchorage	2	tract	99504	907	61.217507	-149.744426	20897
220349	20	2	Alaska	AK	Anchorage	Anchorage	2	tract	99504	907	61.223372	-149.723327	29363

In [48]:
```
test.head()
```

Out[48]:

uid	countyid	stateid	state	state_ab	city	place	type	primary	zip_code	area_code	lat	lng
220336	16	2	Alaska	AK	Unalaska	Unalaska City	2	tract	99685	907	53.621091	-166.770979	2.82318
220343	20	2	Alaska	AK	Jber	Anchorage	2	tract	99505	907	61.284745	-149.653973	2.70593
220352	20	2	Alaska	AK	Anchorage	Point Mackenzie	2	tract	99508	907	61.215067	-149.836271	4.46747
220355	20	2	Alaska	AK	Anchorage	Point Mackenzie	2	tract	99501	907	61.217709	-149.891261	1.19691
220364	20	2	Alaska	AK	Anchorage	Anchorage	2	tract	99504	907	61.202363	-149.733801	2.43408

- 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 [49]:
```
fa = FactorAnalyzer(n_factors=5)
fa.fit_transform(train.select_dtypes(exclude= ('object','category')))
fa_results = fa.loadings_
```

In [50]:
```
fa_df = pd.DataFrame(fa_results)
fa_df.set_index(train.select_dtypes('number').columns, inplace= True)
fa_df.where(fa_df >=0.5)
```

Out[50]:

	0	1	2	3	4
countyid	NaN	NaN	NaN	NaN	NaN
stateid	NaN	NaN	NaN	NaN	NaN
type	NaN	NaN	NaN	NaN	NaN
zip_code	NaN	NaN	NaN	NaN	NaN
area_code	NaN	NaN	NaN	NaN	NaN
lat	NaN	NaN	NaN	NaN	NaN
lng	NaN	NaN	NaN	NaN	NaN
aland	NaN	NaN	NaN	NaN	NaN
awater	NaN	NaN	NaN	NaN	NaN
pop	NaN	0.961767	NaN	NaN	NaN
male_pop	NaN	0.926051	NaN	NaN	NaN
female_pop	NaN	0.954539	NaN	NaN	NaN
rent_mean	0.756421	NaN	NaN	NaN	NaN
rent_median	0.704784	NaN	NaN	NaN	NaN
rent_stdev	0.704324	NaN	NaN	NaN	NaN
rent_sample_weight	NaN	NaN	NaN	NaN	NaN
rent_samples	NaN	NaN	NaN	NaN	NaN
rent_gt_10	NaN	NaN	NaN	NaN	NaN
rent_gt_15	NaN	NaN	NaN	0.659146	NaN
rent_gt_20	NaN	NaN	NaN	0.816860	NaN
rent_gt_25	NaN	NaN	NaN	0.905071	NaN
rent_gt_30	NaN	NaN	NaN	0.933571	NaN
rent_gt_35	NaN	NaN	NaN	0.913893	NaN
rent_gt_40	NaN	NaN	NaN	0.869300	NaN
rent_gt_50	NaN	NaN	NaN	0.762257	NaN
universe_samples	NaN	NaN	NaN	NaN	NaN
used_samples	NaN	NaN	NaN	NaN	NaN
hi_mean	0.774085	NaN	NaN	NaN	NaN
hi_median	0.697474	NaN	NaN	NaN	NaN
hi_stdev	0.859444	NaN	NaN	NaN	NaN
hi_sample_weight	NaN	0.847529	NaN	NaN	NaN
hi_samples	NaN	0.958821	NaN	NaN	NaN
family_mean	0.823911	NaN	NaN	NaN	NaN
family_median	0.786580	NaN	NaN	NaN	NaN
family_stdev	0.814037	NaN	NaN	NaN	NaN
family_sample_weight	NaN	0.859784	NaN	NaN	NaN
family_samples	NaN	0.935016	NaN	NaN	NaN
hc_mortgage_mean	0.974647	NaN	NaN	NaN	NaN
hc_mortgage_median	0.954544	NaN	NaN	NaN	NaN
hc_mortgage_stdev	0.818570	NaN	NaN	NaN	NaN
hc_mortgage_sample_weight	NaN	0.718380	NaN	NaN	NaN
hc_mortgage_samples	NaN	0.730404	NaN	NaN	NaN
hc_mean	0.907873	NaN	NaN	NaN	NaN
hc_median	0.870011	NaN	NaN	NaN	NaN

	0	1	2	3	4
hc_stdev	0.760412	NaN	NaN	NaN	NaN
hc_samples	NaN	0.628482	0.595341	NaN	NaN
hc_sample_weight	NaN	0.575236	0.563933	NaN	NaN
home_equity_second_mortgage	NaN	NaN	NaN	NaN	NaN
second_mortgage	NaN	NaN	NaN	NaN	NaN
home_equity	NaN	NaN	NaN	NaN	NaN
debt	NaN	NaN	NaN	NaN	NaN
second_mortgage_cdf	NaN	NaN	NaN	NaN	0.626438
home_equity_cdf	NaN	NaN	NaN	NaN	0.589429
debt_cdf	NaN	NaN	NaN	NaN	0.717007
hs_degree	NaN	NaN	NaN	NaN	NaN
hs_degree_male	NaN	NaN	NaN	NaN	NaN
hs_degree_female	NaN	NaN	NaN	NaN	NaN
male_age_mean	NaN	NaN	0.734432	NaN	NaN
male_age_median	NaN	NaN	0.767031	NaN	NaN
male_age_stdev	NaN	NaN	0.563811	NaN	NaN
male_age_sample_weight	NaN	0.840713	NaN	NaN	NaN
male_age_samples	NaN	0.925881	NaN	NaN	NaN
female_age_mean	NaN	NaN	0.720388	NaN	NaN
female_age_median	NaN	NaN	0.791343	NaN	NaN
female_age_stdev	NaN	NaN	NaN	NaN	NaN
female_age_sample_weight	NaN	0.879196	NaN	NaN	NaN
female_age_samples	NaN	0.954892	NaN	NaN	NaN
pct_own	NaN	NaN	0.837123	NaN	NaN
married	NaN	NaN	0.577636	NaN	NaN
married_snp	NaN	NaN	NaN	NaN	NaN
separated	NaN	NaN	NaN	NaN	NaN
divorced	NaN	NaN	NaN	NaN	NaN
bad_debt	NaN	NaN	NaN	NaN	NaN
good_debt	NaN	NaN	NaN	NaN	NaN
population_density	NaN	NaN	NaN	NaN	NaN
median_age	NaN	NaN	0.821364	NaN	NaN

Data Modeling :

Build a linear Regression model to predict the total monthly expenditure for home mortgages loan.

In [51]:
```
feature_cols=['countyid', 'stateid', 'zip_code', 'type', 'pop', 'family_mean', 'second_mortgage', 'home_equity', 'debt','hs_
```

In [52]:
```
X_train = train[feature_cols]
y_train = train['hc_mortgage_mean']
```

In [53]:
```
X_test = test[feature_cols]
y_test = test['hc_mortgage_mean']
```

In [54]:
```
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.fit_transform(X_test)
```

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

In [55]:
```
lr = LinearRegression()
lr.fit(X_train, y_train)
```

Out[55]:
```
LinearRegression
LinearRegression()
```

In [56]:
```
y_pred= lr.predict(X_test)
```

In [57]:
```
r2_score_nation = r2_score(y_test, y_pred)

print('\n\n>>> R2 Score :')
print(r2_score_nation)
```

```
>>> R2 Score :
0.7293996111231789
```

In [58]:
```
# Uncomment the line Below to run the State Level Model
# r2_score_nation = 0
```

- Storing the Names of the States in dictionary

In [59]:
```
dict_state = {}
for i in train['stateid'].unique():
    dict_state[i] = df['state'].where(df['stateid']==i).unique()[1]
dict_state[2] = 'Alaska'
print(dict_state)
```

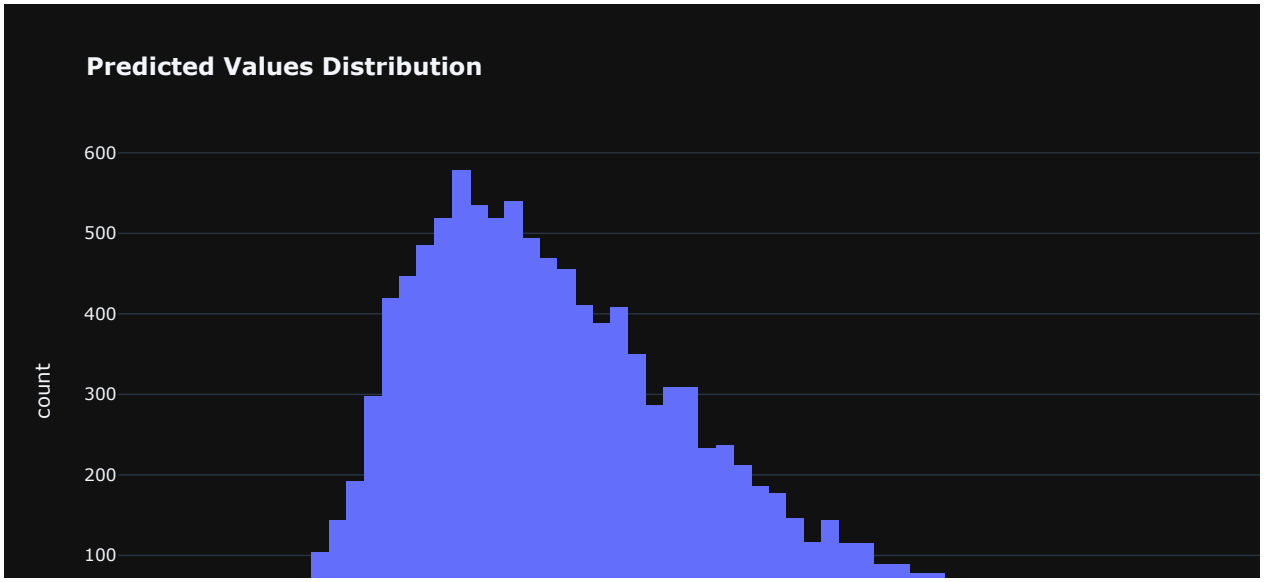
```
{2: 'Alaska', 1: 'Alabama', 5: 'Arkansas', 4: 'Arizona', 6: 'California', 8: 'Colorado', 9: 'Connecticut', 11: 'District of C
olumbia', 10: 'Delaware', 12: 'Florida', 13: 'Georgia', 15: 'Hawaii', 19: 'Iowa', 16: 'Idaho', 17: 'Illinois', 18: 'Indiana',
20: 'Kansas', 21: 'Kentucky', 22: 'Louisiana', 25: 'Massachusetts', 24: 'Maryland', 23: 'Maine', 26: 'Michigan', 27: 'Minneso
ta', 29: 'Missouri', 28: 'Mississippi', 30: 'Montana', 37: 'North Carolina', 38: 'North Dakota', 31: 'Nebraska', 33: 'New Ham
pshire', 34: 'New Jersey', 35: 'New Mexico', 32: 'Nevada', 36: 'New York', 39: 'Ohio', 40: 'Oklahoma', 41: 'Oregon', 42: 'Pen
nsylvania', 72: 'Puerto Rico', 44: 'Rhode Island', 45: 'South Carolina', 46: 'South Dakota', 47: 'Tennessee', 48: 'Texas', 4
9: 'Utah', 51: 'Virginia', 50: 'Vermont', 53: 'Washington', 55: 'Wisconsin', 54: 'West Virginia', 56: 'Wyoming'}
```

State Level Model

- Will run if the accuracy of the nation level model comes out to be below 0.6

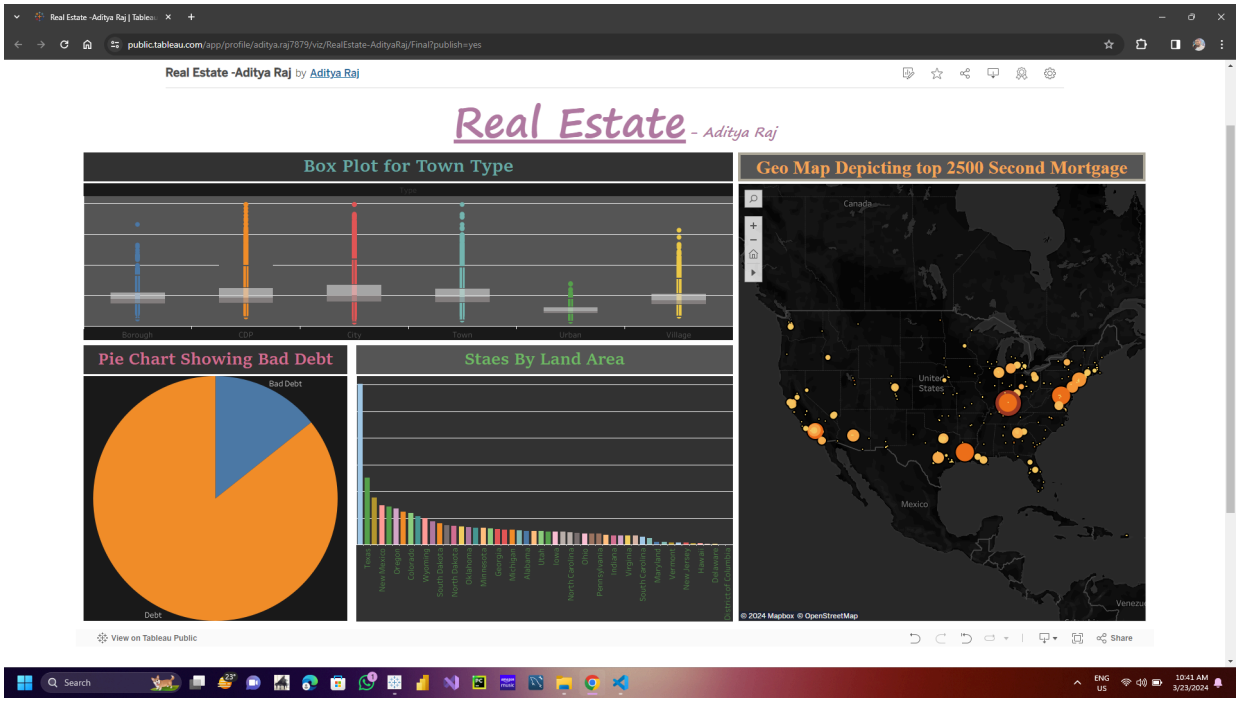
In [60]:
```
if r2_score_nation < 0.60:
    for i in train['stateid'].unique():
        print(">>>State ID-",i)

        X_train_nation = train[train['stateid'] == i][feature_cols]
        y_train_nation = train[train['stateid'] == i]['hc_mortgage_mean']

        X_test_nation = test[test['stateid'] == i][feature_cols]
        y_test_nation = test[test['stateid'] == i]['hc_mortgage_mean']


        lr.fit(X_train_nation,y_train_nation)
        y_pred_nation = lr.predict(X_test_nation)

        print(f">>>Overall R2 score of linear regression model for state, {dict_state[i]} : {r2_score(y_test_nation,y_pred_n
        print('*'*70)
```

In [61]:
```
fig_final = px.histogram(data_frame = y_pred,
                         template = 'plotly_dark',
                         title = '<b>Predicted Values Distribution</b>')
fig_final.update_traces(showlegend = False)
fig_final.show()
```



Data Reporting:

- Create a dashboard in tableau by choosing appropriate chart types and metrics useful for the business. The dashboard must entail the following:
- Box plot of distribution of average rent by type of place (village, urban, town, etc.).
- Pie charts to show overall debt and bad debt.
- 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.
- Heat map for correlation matrix.
- Pie chart to show the population distribution across different types of places (village, urban, town etc.).



https://public.tableau.com/shared/F7YGFTR65?:display_count=n&:origin=viz_share_link

Thank You for viewing the Real Estate Model Created by Aditya Raj