****

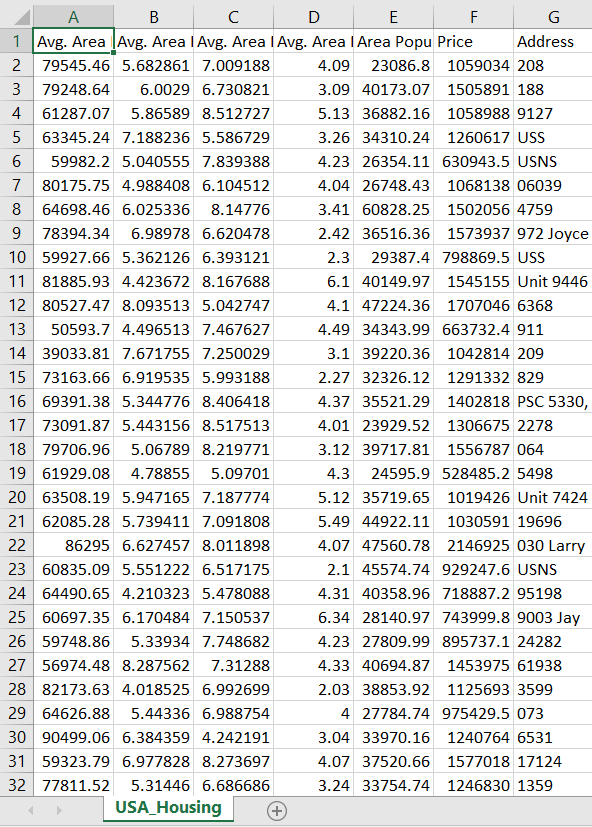
**Abstract:**

Predicting house prices using machine learning is a challenging but important task. Machine learning models can be used to learn from historical data and identify patterns that can be used to predict future house prices. This module will provide an overview of the machine learning process for predicting house prices, including data preparation, feature engineering, model selection, training, and evaluation.

**Module Outline :**

1. Introduction to Machine Learning for House Price Prediction
   * What is machine learning?
   * Regression models for predicting house prices
   * Challenges of predicting house prices
2. Data Preparation
   * Cleaning and formatting the data
   * Dealing with missing values
   * Handling categorical features
3. Feature Engineering
   * Creating new features from existing features
   * Scaling the features
4. Model Selection
   * Choosing a machine learning algorithm
   * Tuning the hyperparameters
5. Training and Evaluation
   * Training the model on the training data
   * Evaluating the model on the test data
6. Predicting House Prices
   * Using the trained model to predict house prices for new data

**Data Source:**

****

**Code:**

**import pandas as** **pd**

**import numpy as** **np**

**import** **pickle**

***#visualisation***

**import plotly.express as** **px**

**import matplotlib.pyplot as** **plt**

**%matplotlib inline**

**import seaborn as** **sns**

***#sns.set\_style('whitegrid')***

***# Model***

**from sklearn.svm import** **SVR**

**from sklearn.tree import** **DecisionTreeRegressor**

**from sklearn.ensemble import** **RandomForestRegressor**

**from sklearn.model\_selection import** **train\_test\_split**

**from sklearn.linear\_model import** **LinearRegression, SGDRegressor**

**from sklearn.linear\_model import** **ElasticNet**

***#perfomance***

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

**Importing Dataset**

**In [2]:**

**df = pd.read\_csv("/kaggle/input/usa-housing/USA\_Housing.csv")**

**EDA**

**In [3]:**

**df.head()**

**Out[3]:**

|  | **Avg. Area Income** | **Avg. Area House Age** | **Avg. Area Number of Rooms** | **Avg. Area Number of Bedrooms** | **Area Population** | **Price** | **Address** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **0** | **79545.458574** | **5.682861** | **7.009188** | **4.09** | **23086.800503** | **1.059034e+06** | **208 Michael Ferry Apt. 674\nLaurabury, NE 3701...** |
| **1** | **79248.642455** | **6.002900** | **6.730821** | **3.09** | **40173.072174** | **1.505891e+06** | **188 Johnson Views Suite 079\nLake Kathleen, CA...** |
| **2** | **61287.067179** | **5.865890** | **8.512727** | **5.13** | **36882.159400** | **1.058988e+06** | **9127 Elizabeth Stravenue\nDanieltown, WI 06482...** |
| **3** | **63345.240046** | **7.188236** | **5.586729** | **3.26** | **34310.242831** | **1.260617e+06** | **USS Barnett\nFPO AP 44820** |
| **4** | **59982.197226** | **5.040555** | **7.839388** | **4.23** | **26354.109472** | **6.309435e+05** | **USNS Raymond\nFPO AE 09386** |

**In [4]:**

**df.info()**

**<class 'pandas.core.frame.DataFrame'>**

**RangeIndex: 5000 entries, 0 to 4999**

**Data columns (total 7 columns):**

**# Column Non-Null Count Dtype**

**--- ------ -------------- -----**

**0 Avg. Area Income 5000 non-null float64**

**1 Avg. Area House Age 5000 non-null float64**

**2 Avg. Area Number of Rooms 5000 non-null float64**

**3 Avg. Area Number of Bedrooms 5000 non-null float64**

**4 Area Population 5000 non-null float64**

**5 Price 5000 non-null float64**

**6 Address 5000 non-null object**

**dtypes: float64(6), object(1)**

**memory usage: 273.6+ KB**

**Dropping the Address column**

**In [5]:**

**df.drop('Address',axis=1,inplace=True)**

**In [6]:**

**df.isna().sum()**

**Out[6]:**

**Avg. Area Income 0**

**Avg. Area House Age 0**

**Avg. Area Number of Rooms 0**

**Avg. Area Number of Bedrooms 0**

**Area Population 0**

**Price 0**

**dtype: int64**

**In [7]:**

**df.describe()**

**Out[7]:**

|  | **Avg. Area Income** | **Avg. Area House Age** | **Avg. Area Number of Rooms** | **Avg. Area Number of Bedrooms** | **Area Population** | **Price** |
| --- | --- | --- | --- | --- | --- | --- |
| **count** | **5000.000000** | **5000.000000** | **5000.000000** | **5000.000000** | **5000.000000** | **5.000000e+03** |
| **mean** | **68583.108984** | **5.977222** | **6.987792** | **3.981330** | **36163.516039** | **1.232073e+06** |
| **std** | **10657.991214** | **0.991456** | **1.005833** | **1.234137** | **9925.650114** | **3.531176e+05** |
| **min** | **17796.631190** | **2.644304** | **3.236194** | **2.000000** | **172.610686** | **1.593866e+04** |
| **25%** | **61480.562388** | **5.322283** | **6.299250** | **3.140000** | **29403.928702** | **9.975771e+05** |
| **50%** | **68804.286404** | **5.970429** | **7.002902** | **4.050000** | **36199.406689** | **1.232669e+06** |
| **75%** | **75783.338666** | **6.650808** | **7.665871** | **4.490000** | **42861.290769** | **1.471210e+06** |
| **max** | **107701.748378** | **9.519088** | **10.759588** | **6.500000** | **69621.713378** | **2.469066e+06** |

**In [8]:**

**df.shape**

**Out[8]:**

**(5000, 6)**

**In [9]:**

**df.var()**

**Out[9]:**

**Avg. Area Income 1.135928e+08**

**Avg. Area House Age 9.829854e-01**

**Avg. Area Number of Rooms 1.011700e+00**

**Avg. Area Number of Bedrooms 1.523095e+00**

**Area Population 9.851853e+07**

**Price 1.246921e+11**

**dtype: float64**

**In [10]:**

**df.kurt()**

**Out[10]:**

**Avg. Area Income 0.045574**

**Avg. Area House Age -0.083437**

**Avg. Area Number of Rooms -0.074652**

**Avg. Area Number of Bedrooms -0.701566**

**Area Population -0.006733**

**Price -0.054918**

**dtype: float64**

**In [11]:**

**df.skew()**

**Out[11]:**

**Avg. Area Income -0.033720**

**Avg. Area House Age -0.007214**

**Avg. Area Number of Rooms -0.040996**

**Avg. Area Number of Bedrooms 0.376240**

**Area Population 0.050650**

**Price -0.002718**

**dtype: float64**

**In [12]:**

**df.hist(bins=200,figsize=[20,10])**

**Out[12]:**

**array([[<AxesSubplot:title={'center':'Avg. Area Income'}>,**

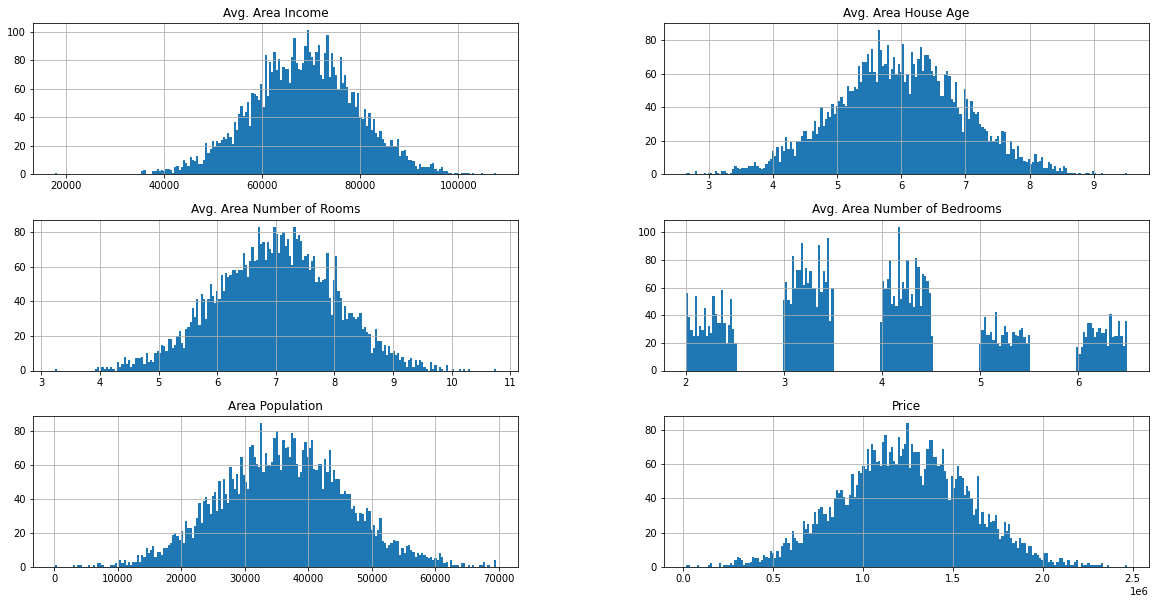
**<AxesSubplot:title={'center':'Avg. Area House Age'}>],**

**[<AxesSubplot:title={'center':'Avg. Area Number of Rooms'}>,**

**<AxesSubplot:title={'center':'Avg. Area Number of Bedrooms'}>],**

**[<AxesSubplot:title={'center':'Area Population'}>,**

**<AxesSubplot:title={'center':'Price'}>]], dtype=object)**

****

**Detecting Outliers using Boxplot**

**In [13]:**

**df.columns**

**Out[13]:**

**Index(['Avg. Area Income', 'Avg. Area House Age', 'Avg. Area Number of Rooms',**

**'Avg. Area Number of Bedrooms', 'Area Population', 'Price'],**

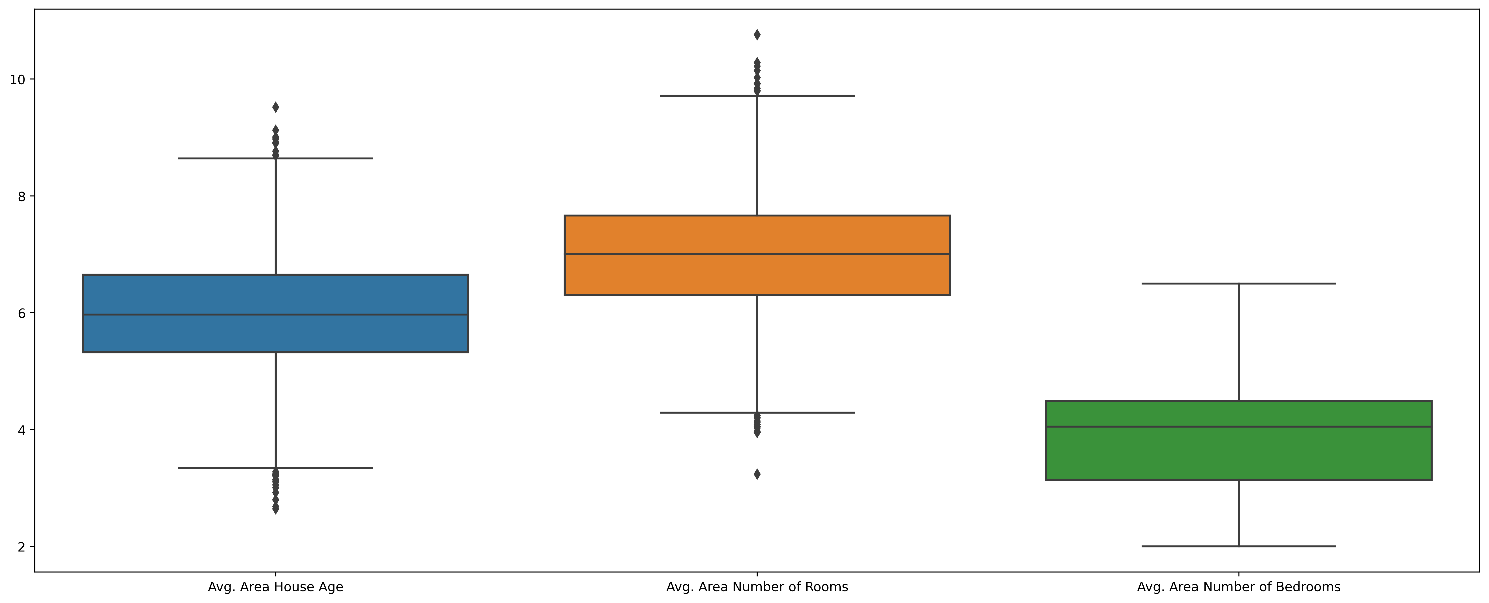
**dtype='object')**

**In [14]:**

**plt.figure(figsize=(20,8),dpi=400)**

**sns.boxplot(data=df[[ 'Avg. Area House Age', 'Avg. Area Number of Rooms','Avg. Area Number of Bedrooms']],orient='v')**

**plt.show()**

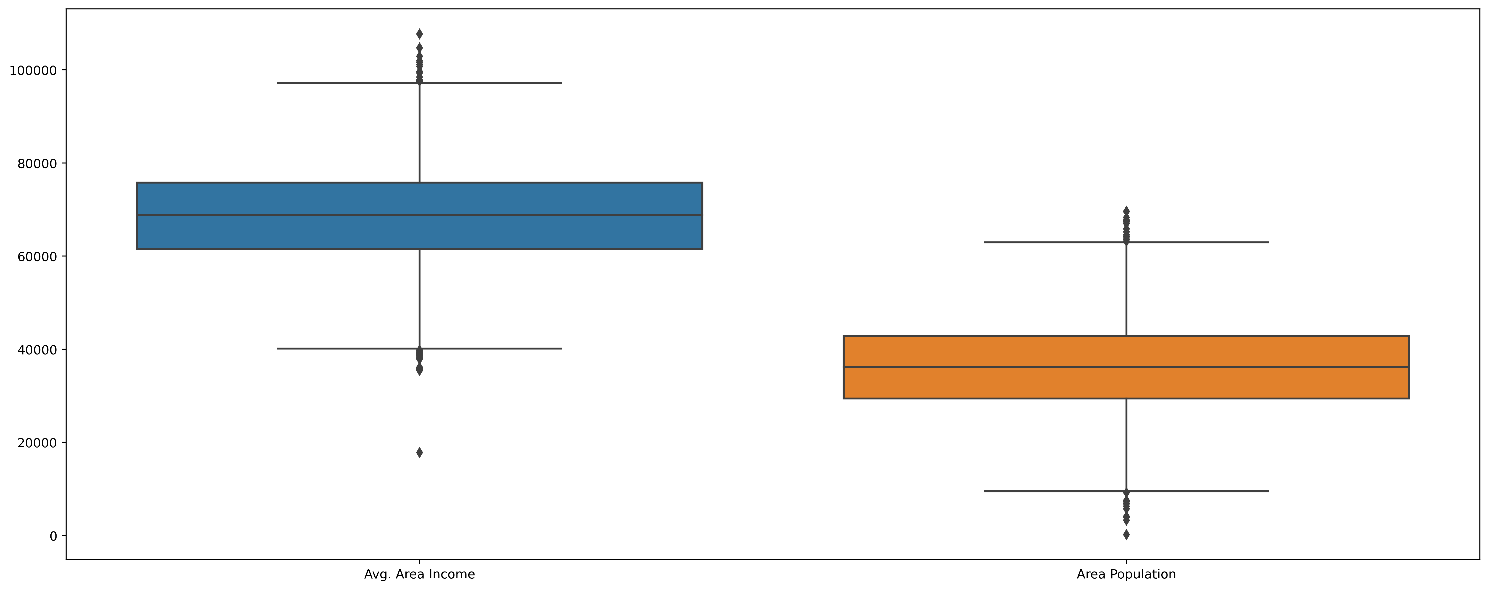
****

**In [15]:**

**plt.figure(figsize=(20,8),dpi=400)**

**sns.boxplot(data=df[[ 'Avg. Area Income', 'Area Population']],orient='v')**

**plt.show()**

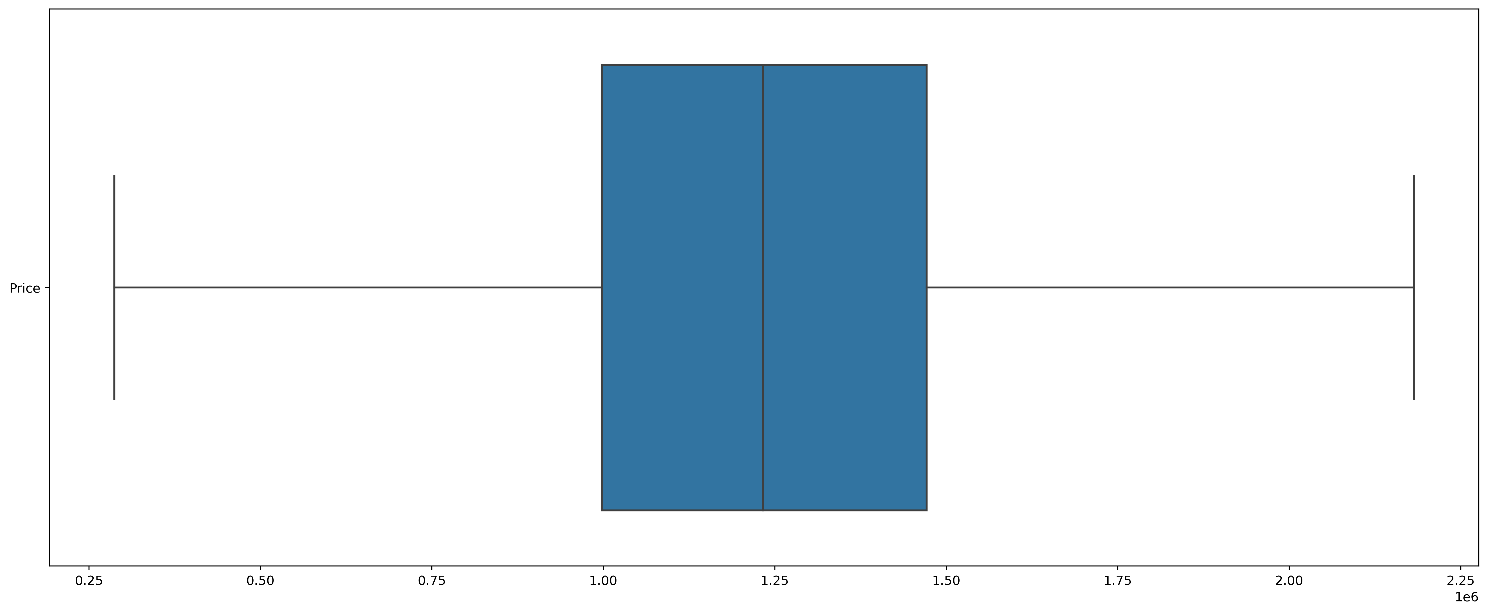
****

**In [23]:**

**plt.figure(figsize=(20,8),dpi=400)**

**sns.boxplot(data=df[[ 'Price']],orient='h')**

**plt.show()**

****

**Note Don't get confused about the outliers in the dataset as they are happenning due to exotic cars for special tasks with high performance so we will not be discarding them.**

**In [1]:**

***#pip install feature\_engine***

**In [22]:**

**from feature\_engine.outliers import** **Winsorizer**

**winsor = Winsorizer(capping\_method='iqr', *# choose IQR rule boundaries or gaussian for mean and std***

**tail='both', *# cap left, right or both tails***

**fold=1.5,**

**variables=['Avg. Area House Age'])**

**df['Avg. Area House Age'] = winsor.fit\_transform(df[['Avg. Area House Age']])**

**winsor = Winsorizer(capping\_method='iqr', *# choose IQR rule boundaries or gaussian for mean and std***

**tail='both', *# cap left, right or both tails***

**fold=1.5,**

**variables=['Avg. Area Number of Rooms'])**

**df['Avg. Area Number of Rooms'] = winsor.fit\_transform(df[['Avg. Area Number of Rooms']])**

**winsor = Winsorizer(capping\_method='iqr', *# choose IQR rule boundaries or gaussian for mean and std***

**tail='both', *# cap left, right or both tails***

**fold=1.5,**

**variables=['Avg. Area Income'])**

**df['Avg. Area Income'] = winsor.fit\_transform(df[['Avg. Area Income']])**

**winsor = Winsorizer(capping\_method='iqr', *# choose IQR rule boundaries or gaussian for mean and std***

**tail='both', *# cap left, right or both tails***

**fold=1.5,**

**variables=['Area Population'])**

**df['Area Population'] = winsor.fit\_transform(df[['Area Population']])**

**winsor = Winsorizer(capping\_method='iqr', *# choose IQR rule boundaries or gaussian for mean and std***

**tail='both', *# cap left, right or both tails***

**fold=1.5,**

**variables=['Price'])**

**df['Price'] = winsor.fit\_transform(df[['Price']])**

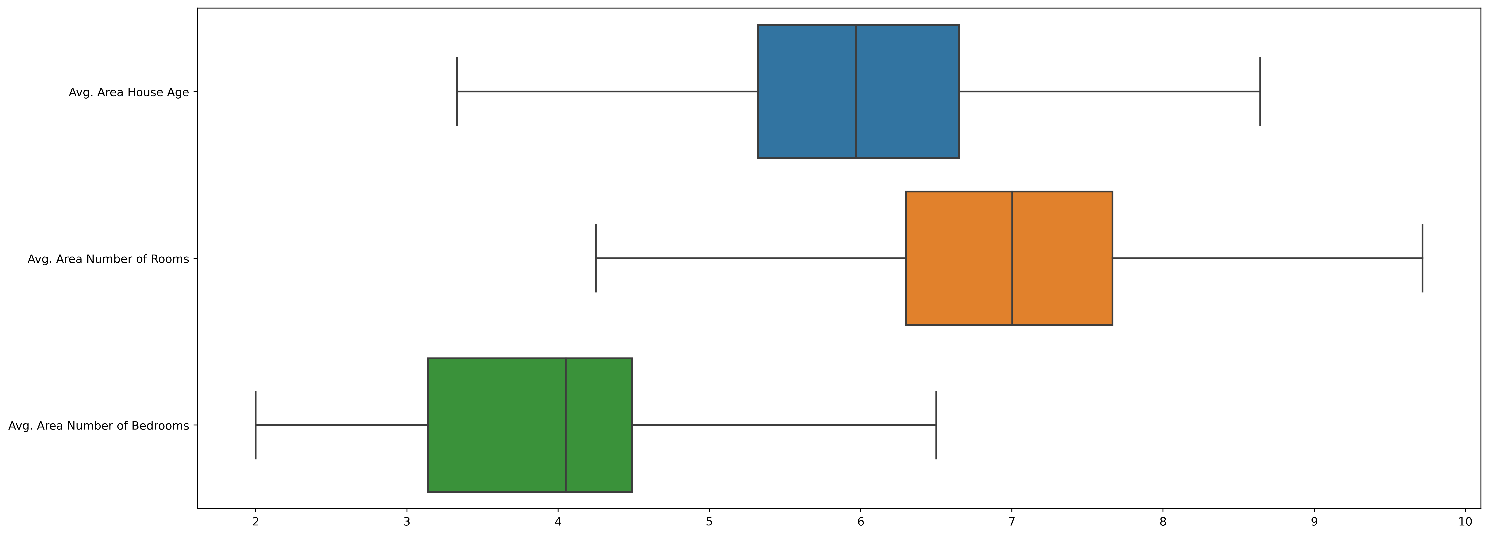
**Boxplot after dealing with Outliers**

**In [24]:**

**plt.figure(figsize=(20,8),dpi=400)**

**sns.boxplot(data=df[[ 'Avg. Area House Age', 'Avg. Area Number of Rooms','Avg. Area Number of Bedrooms']],orient='h')**

**plt.show()**

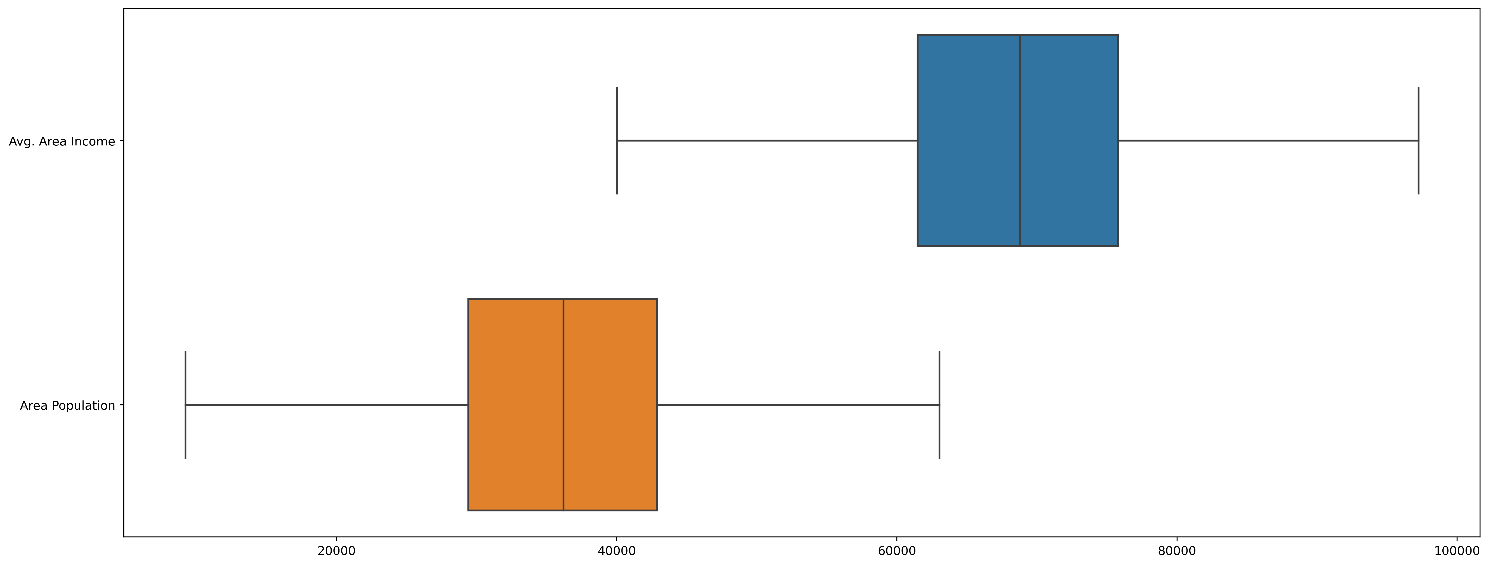
****

**In [25]:**

**plt.figure(figsize=(20,8),dpi=400)**

**sns.boxplot(data=df[[ 'Avg. Area Income', 'Area Population']],orient='h')**

**plt.show()**

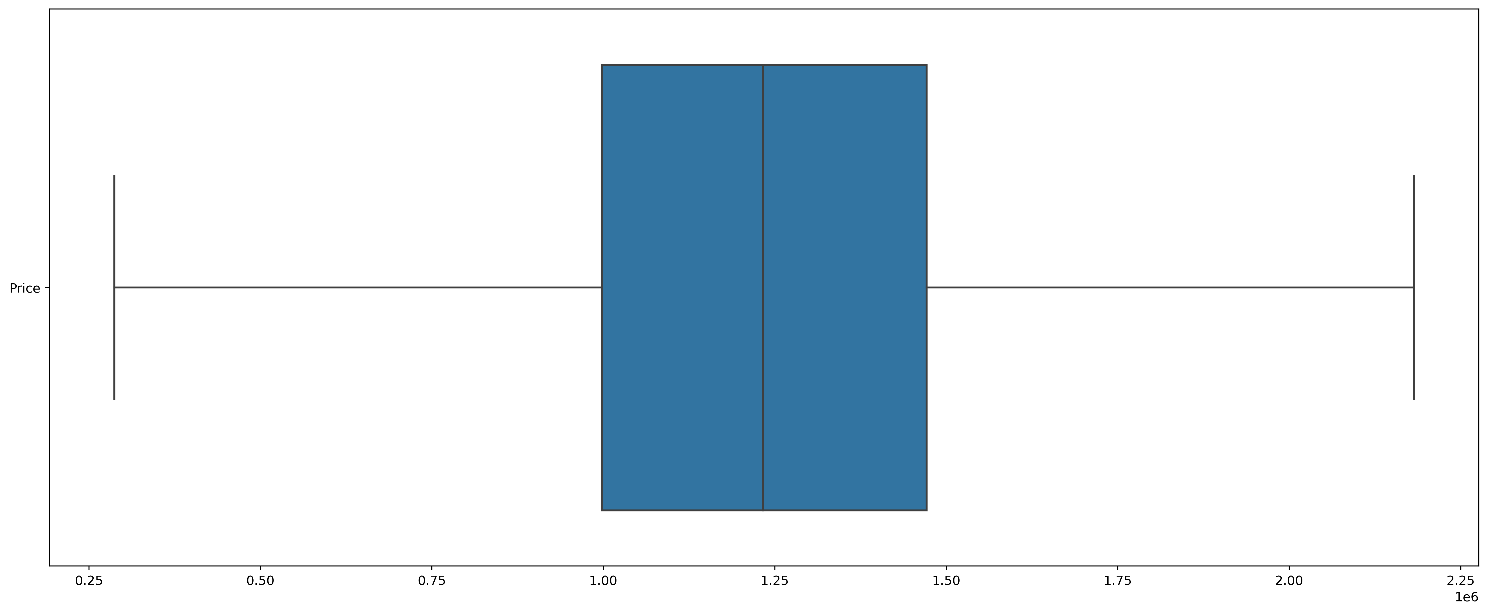
****

**In [26]:**

**plt.figure(figsize=(20,8),dpi=400)**

**sns.boxplot(data=df[[ 'Price']],orient='h')**

**plt.show()**

****

**In [28]:**

**df.to\_csv('without\_csv.csv')**

**Auto EDA and Visualisation**

**In [2]:**

***#pip install dataprep***

**In [27]:**

**from dataprep.eda import** **create\_report**

**create\_report(df)**

**bokeh.core.validation.check - ERROR - E-1019 (DUPLICATE\_FACTORS): FactorRange must specify a unique list of categorical factors for an axis: duplicate factors found: 'Avg. Area Num...oms'**

**Out[27]:**

[**DataPrep Report**](https://www.kaggleusercontent.com/kf/112565982/eyJhbGciOiJkaXIiLCJlbmMiOiJBMTI4Q0JDLUhTMjU2In0..rLwobED6pF9AcyIXKkfSVQ.L1LGlo0GdXx4jDncewM5H-HVe7T-mSqFa47LygF6TUbZYeI1vuV8cMwiWP6iS00myOLh7ehZfKWneIFBL5nLiSnObp2zBMtSRGt_tVmUgDOPaGU0TqfbzB7b4QBuYbp0CgLtHJMTvjFajRBnJjU3zgqo4Ak4ToHzyK-CzG5menGY47-tMqv3GmRsKTBimwfCtzJjSzfQHqO9eO0HPBgiFRnfn9-qWUcltsdXXfj0DNulEB2u6OfMnwk7u1P7-WGJF3oYO2Y8KaOfJqeE1rJfWOSoR41zocLaNT5TfGnlb-AiO4ReRWeuaGn02ehch-ohxodjJsa3frRjXlbTewznVGWdzmIFuq70Z-gGusVLR8nnkDsB2Ha-wYK2jD4BkzaKT4mJMIr4DUrb1eqW-ePT-hYvYTfEznZTHL3YHE1nMx9wP9dA67gRdaB0_55PFeEpjkfD8LGCNkQ4roL3CHx6Es07pm5UNqEcLmKcZ_Yn5zsl-TRnDLrYrBaJ5kpqXAnEtfO3EVHiVHfSoqqTiCwFEihC-0SSnEQsPZaqNZPBEzejZ21Q0zVJZbrb9SuiJ5LOcOHtSoGfBhYhgjYq8LMgJCcQHq4BCpBgEDTYjHDcusuUFfCSv8d0U9Or_dU8-2K7RrVG6-L6dUQ3i5pjLBkuIROlaHH2MYK8kPOmwYRti18.DOvMUP25ZaV2lRLeQ09B_w/__resultx__.html?sharingControls=true)[**Overview**](https://www.kaggleusercontent.com/kf/112565982/eyJhbGciOiJkaXIiLCJlbmMiOiJBMTI4Q0JDLUhTMjU2In0..rLwobED6pF9AcyIXKkfSVQ.L1LGlo0GdXx4jDncewM5H-HVe7T-mSqFa47LygF6TUbZYeI1vuV8cMwiWP6iS00myOLh7ehZfKWneIFBL5nLiSnObp2zBMtSRGt_tVmUgDOPaGU0TqfbzB7b4QBuYbp0CgLtHJMTvjFajRBnJjU3zgqo4Ak4ToHzyK-CzG5menGY47-tMqv3GmRsKTBimwfCtzJjSzfQHqO9eO0HPBgiFRnfn9-qWUcltsdXXfj0DNulEB2u6OfMnwk7u1P7-WGJF3oYO2Y8KaOfJqeE1rJfWOSoR41zocLaNT5TfGnlb-AiO4ReRWeuaGn02ehch-ohxodjJsa3frRjXlbTewznVGWdzmIFuq70Z-gGusVLR8nnkDsB2Ha-wYK2jD4BkzaKT4mJMIr4DUrb1eqW-ePT-hYvYTfEznZTHL3YHE1nMx9wP9dA67gRdaB0_55PFeEpjkfD8LGCNkQ4roL3CHx6Es07pm5UNqEcLmKcZ_Yn5zsl-TRnDLrYrBaJ5kpqXAnEtfO3EVHiVHfSoqqTiCwFEihC-0SSnEQsPZaqNZPBEzejZ21Q0zVJZbrb9SuiJ5LOcOHtSoGfBhYhgjYq8LMgJCcQHq4BCpBgEDTYjHDcusuUFfCSv8d0U9Or_dU8-2K7RrVG6-L6dUQ3i5pjLBkuIROlaHH2MYK8kPOmwYRti18.DOvMUP25ZaV2lRLeQ09B_w/__resultx__.html?sharingControls=true#overview)

[**Variables ≡**](https://www.kaggleusercontent.com/kf/112565982/eyJhbGciOiJkaXIiLCJlbmMiOiJBMTI4Q0JDLUhTMjU2In0..rLwobED6pF9AcyIXKkfSVQ.L1LGlo0GdXx4jDncewM5H-HVe7T-mSqFa47LygF6TUbZYeI1vuV8cMwiWP6iS00myOLh7ehZfKWneIFBL5nLiSnObp2zBMtSRGt_tVmUgDOPaGU0TqfbzB7b4QBuYbp0CgLtHJMTvjFajRBnJjU3zgqo4Ak4ToHzyK-CzG5menGY47-tMqv3GmRsKTBimwfCtzJjSzfQHqO9eO0HPBgiFRnfn9-qWUcltsdXXfj0DNulEB2u6OfMnwk7u1P7-WGJF3oYO2Y8KaOfJqeE1rJfWOSoR41zocLaNT5TfGnlb-AiO4ReRWeuaGn02ehch-ohxodjJsa3frRjXlbTewznVGWdzmIFuq70Z-gGusVLR8nnkDsB2Ha-wYK2jD4BkzaKT4mJMIr4DUrb1eqW-ePT-hYvYTfEznZTHL3YHE1nMx9wP9dA67gRdaB0_55PFeEpjkfD8LGCNkQ4roL3CHx6Es07pm5UNqEcLmKcZ_Yn5zsl-TRnDLrYrBaJ5kpqXAnEtfO3EVHiVHfSoqqTiCwFEihC-0SSnEQsPZaqNZPBEzejZ21Q0zVJZbrb9SuiJ5LOcOHtSoGfBhYhgjYq8LMgJCcQHq4BCpBgEDTYjHDcusuUFfCSv8d0U9Or_dU8-2K7RrVG6-L6dUQ3i5pjLBkuIROlaHH2MYK8kPOmwYRti18.DOvMUP25ZaV2lRLeQ09B_w/__resultx__.html?sharingControls=true#variables)

[**Interactions**](https://www.kaggleusercontent.com/kf/112565982/eyJhbGciOiJkaXIiLCJlbmMiOiJBMTI4Q0JDLUhTMjU2In0..rLwobED6pF9AcyIXKkfSVQ.L1LGlo0GdXx4jDncewM5H-HVe7T-mSqFa47LygF6TUbZYeI1vuV8cMwiWP6iS00myOLh7ehZfKWneIFBL5nLiSnObp2zBMtSRGt_tVmUgDOPaGU0TqfbzB7b4QBuYbp0CgLtHJMTvjFajRBnJjU3zgqo4Ak4ToHzyK-CzG5menGY47-tMqv3GmRsKTBimwfCtzJjSzfQHqO9eO0HPBgiFRnfn9-qWUcltsdXXfj0DNulEB2u6OfMnwk7u1P7-WGJF3oYO2Y8KaOfJqeE1rJfWOSoR41zocLaNT5TfGnlb-AiO4ReRWeuaGn02ehch-ohxodjJsa3frRjXlbTewznVGWdzmIFuq70Z-gGusVLR8nnkDsB2Ha-wYK2jD4BkzaKT4mJMIr4DUrb1eqW-ePT-hYvYTfEznZTHL3YHE1nMx9wP9dA67gRdaB0_55PFeEpjkfD8LGCNkQ4roL3CHx6Es07pm5UNqEcLmKcZ_Yn5zsl-TRnDLrYrBaJ5kpqXAnEtfO3EVHiVHfSoqqTiCwFEihC-0SSnEQsPZaqNZPBEzejZ21Q0zVJZbrb9SuiJ5LOcOHtSoGfBhYhgjYq8LMgJCcQHq4BCpBgEDTYjHDcusuUFfCSv8d0U9Or_dU8-2K7RrVG6-L6dUQ3i5pjLBkuIROlaHH2MYK8kPOmwYRti18.DOvMUP25ZaV2lRLeQ09B_w/__resultx__.html?sharingControls=true#interactions)[**Correlations**](https://www.kaggleusercontent.com/kf/112565982/eyJhbGciOiJkaXIiLCJlbmMiOiJBMTI4Q0JDLUhTMjU2In0..rLwobED6pF9AcyIXKkfSVQ.L1LGlo0GdXx4jDncewM5H-HVe7T-mSqFa47LygF6TUbZYeI1vuV8cMwiWP6iS00myOLh7ehZfKWneIFBL5nLiSnObp2zBMtSRGt_tVmUgDOPaGU0TqfbzB7b4QBuYbp0CgLtHJMTvjFajRBnJjU3zgqo4Ak4ToHzyK-CzG5menGY47-tMqv3GmRsKTBimwfCtzJjSzfQHqO9eO0HPBgiFRnfn9-qWUcltsdXXfj0DNulEB2u6OfMnwk7u1P7-WGJF3oYO2Y8KaOfJqeE1rJfWOSoR41zocLaNT5TfGnlb-AiO4ReRWeuaGn02ehch-ohxodjJsa3frRjXlbTewznVGWdzmIFuq70Z-gGusVLR8nnkDsB2Ha-wYK2jD4BkzaKT4mJMIr4DUrb1eqW-ePT-hYvYTfEznZTHL3YHE1nMx9wP9dA67gRdaB0_55PFeEpjkfD8LGCNkQ4roL3CHx6Es07pm5UNqEcLmKcZ_Yn5zsl-TRnDLrYrBaJ5kpqXAnEtfO3EVHiVHfSoqqTiCwFEihC-0SSnEQsPZaqNZPBEzejZ21Q0zVJZbrb9SuiJ5LOcOHtSoGfBhYhgjYq8LMgJCcQHq4BCpBgEDTYjHDcusuUFfCSv8d0U9Or_dU8-2K7RrVG6-L6dUQ3i5pjLBkuIROlaHH2MYK8kPOmwYRti18.DOvMUP25ZaV2lRLeQ09B_w/__resultx__.html?sharingControls=true#correlations)[**Missing Values**](https://www.kaggleusercontent.com/kf/112565982/eyJhbGciOiJkaXIiLCJlbmMiOiJBMTI4Q0JDLUhTMjU2In0..rLwobED6pF9AcyIXKkfSVQ.L1LGlo0GdXx4jDncewM5H-HVe7T-mSqFa47LygF6TUbZYeI1vuV8cMwiWP6iS00myOLh7ehZfKWneIFBL5nLiSnObp2zBMtSRGt_tVmUgDOPaGU0TqfbzB7b4QBuYbp0CgLtHJMTvjFajRBnJjU3zgqo4Ak4ToHzyK-CzG5menGY47-tMqv3GmRsKTBimwfCtzJjSzfQHqO9eO0HPBgiFRnfn9-qWUcltsdXXfj0DNulEB2u6OfMnwk7u1P7-WGJF3oYO2Y8KaOfJqeE1rJfWOSoR41zocLaNT5TfGnlb-AiO4ReRWeuaGn02ehch-ohxodjJsa3frRjXlbTewznVGWdzmIFuq70Z-gGusVLR8nnkDsB2Ha-wYK2jD4BkzaKT4mJMIr4DUrb1eqW-ePT-hYvYTfEznZTHL3YHE1nMx9wP9dA67gRdaB0_55PFeEpjkfD8LGCNkQ4roL3CHx6Es07pm5UNqEcLmKcZ_Yn5zsl-TRnDLrYrBaJ5kpqXAnEtfO3EVHiVHfSoqqTiCwFEihC-0SSnEQsPZaqNZPBEzejZ21Q0zVJZbrb9SuiJ5LOcOHtSoGfBhYhgjYq8LMgJCcQHq4BCpBgEDTYjHDcusuUFfCSv8d0U9Or_dU8-2K7RrVG6-L6dUQ3i5pjLBkuIROlaHH2MYK8kPOmwYRti18.DOvMUP25ZaV2lRLeQ09B_w/__resultx__.html?sharingControls=true#missing-values)

**Overview**

**Dataset Statistics**

|  |  |
| --- | --- |
| **Number of Variables** | **6** |
| **Number of Rows** | **5000** |
| **Missing Cells** | **0** |
| **Missing Cells (%)** | **0.0%** |
| **Duplicate Rows** | **0** |
| **Duplicate Rows (%)** | **0.0%** |
| **Total Size in Memory** | **234.5 KB** |
| **Average Row Size in Memory** | **48.0 B** |
| **Variable Types** | * **Numerical: 6** |

**Variables**

**Sort by              Feature order             Alphabetical             Amount missing             Approximate unique           Reverse order**

**Avg. Area Income**

**numerical**

**Show Details**

|  |  |
| --- | --- |
| **Approximate Distinct Count** | **4970** |
| **Approximate Unique (%)** | **99.4%** |
| **Missing** | **0** |
| **Missing (%)** | **0.0%** |
| **Infinite** | **0** |
| **Infinite (%)** | **0.0%** |
| **Memory Size** | **78.1 KB** |
| **Mean** | **68585.6362** |
| **Minimum** | **40026.398** |
| **Maximum** | **97237.5031** |
| **Zeros** | **0** |
| **Zeros (%)** | **0.0%** |
| **Negatives** | **0** |
| **Negatives (%)** | **0.0%** |

* **Avg. Area Income is skewed left (γ1 = -0.0236)**

**Avg. Area House Age**

**numerical**

**Show Details**

|  |  |
| --- | --- |
| **Approximate Distinct Count** | **4977** |
| **Approximate Unique (%)** | **99.5%** |
| **Missing** | **0** |
| **Missing (%)** | **0.0%** |
| **Infinite** | **0** |
| **Infinite (%)** | **0.0%** |
| **Memory Size** | **78.1 KB** |
| **Mean** | **5.9775** |
| **Minimum** | **3.3295** |
| **Maximum** | **8.6436** |
| **Zeros** | **0** |
| **Zeros (%)** | **0.0%** |
| **Negatives** | **0** |
| **Negatives (%)** | **0.0%** |

* **Avg. Area House Age is skewed left (γ1 = -0.0017)**

**Avg. Area Number of Rooms**

**numerical**

**Show Details**

|  |  |
| --- | --- |
| **Approximate Distinct Count** | **4978** |
| **Approximate Unique (%)** | **99.6%** |
| **Missing** | **0** |
| **Missing (%)** | **0.0%** |
| **Infinite** | **0** |
| **Infinite (%)** | **0.0%** |
| **Memory Size** | **78.1 KB** |
| **Mean** | **6.9876** |
| **Minimum** | **4.2493** |
| **Maximum** | **9.7158** |
| **Zeros** | **0** |
| **Zeros (%)** | **0.0%** |
| **Negatives** | **0** |
| **Negatives (%)** | **0.0%** |

* **Avg. Area Number of Rooms is skewed left (γ1 = -0.0458)**

**Avg. Area Number of Bedrooms**

**numerical**

**Show Details**

|  |  |
| --- | --- |
| **Approximate Distinct Count** | **255** |
| **Approximate Unique (%)** | **5.1%** |
| **Missing** | **0** |
| **Missing (%)** | **0.0%** |
| **Infinite** | **0** |
| **Infinite (%)** | **0.0%** |
| **Memory Size** | **78.1 KB** |
| **Mean** | **3.9813** |
| **Minimum** | **2** |
| **Maximum** | **6.5** |
| **Zeros** | **0** |
| **Zeros (%)** | **0.0%** |
| **Negatives** | **0** |
| **Negatives (%)** | **0.0%** |

* **Avg. Area Number of Bedrooms is skewed right (γ1 = 0.3761)**

**Area Population**

**numerical**

**Show Details**

|  |  |
| --- | --- |
| **Approximate Distinct Count** | **4972** |
| **Approximate Unique (%)** | **99.4%** |
| **Missing** | **0** |
| **Missing (%)** | **0.0%** |
| **Infinite** | **0** |
| **Infinite (%)** | **0.0%** |
| **Memory Size** | **78.1 KB** |
| **Mean** | **36159.0708** |
| **Minimum** | **9217.8856** |
| **Maximum** | **63047.3339** |
| **Zeros** | **0** |
| **Zeros (%)** | **0.0%** |
| **Negatives** | **0** |
| **Negatives (%)** | **0.0%** |

* **Area Population is skewed right (γ1 = 0.0412)**

**Price**

**numerical**

**Show Details**

|  |  |
| --- | --- |
| **Approximate Distinct Count** | **4967** |
| **Approximate Unique (%)** | **99.3%** |
| **Missing** | **0** |
| **Missing (%)** | **0.0%** |
| **Infinite** | **0** |
| **Infinite (%)** | **0.0%** |
| **Memory Size** | **78.1 KB** |
| **Mean** | **1.232e+06** |
| **Minimum** | **287127.5313** |
| **Maximum** | **2.1817e+06** |
| **Zeros** | **0** |
| **Zeros (%)** | **0.0%** |
| **Negatives** | **0** |
| **Negatives (%)** | **0.0%** |

* **Price is skewed left (γ1 = -0.0051)**

**Interactions**

**Correlations**

**PearsonSpearmanKendallTau**

**Missing Values**

**Bar ChartSpectrumHeat MapDendrogram**

**Report generated with**[**DataPrep**](https://dataprep.ai/)

**In [3]:**

***#pip install AutoViz***

**In [31]:**

**from autoviz.AutoViz\_Class import** **AutoViz\_Class**

**AV = AutoViz\_Class() *#instantiaze the AV***

**%matplotlib inline**

**filename = '/kaggle/working/without\_csv.csv'**

**sep = ","**

**dft = AV.AutoViz(**

**filename,**

**sep=****sep,**

**depVar="",**

**dfte=****None,**

**header=0,**

**verbose=2,**

**lowess=****False,**

**chart\_format="svg",**

**max\_rows\_analyzed=2000,**

**max\_cols\_analyzed=20,**

**)**

**Conclusion :**

This module has provided an overview of the machine learning process for predicting house prices. By following the steps outlined in this module, you can develop a machine learning model that can be used to predict house prices with reasonable accuracy.

Module Benefits

Upon completing this module, you will be able to:

* Understand the basics of machine learning and how it can be used to predict house prices
* Prepare data for machine learning modeling
* Engineer new features from existing features
* Select and tune a machine learning algorithm for predicting house prices
* Train and evaluate a machine learning model
* Use a trained machine learning model to predict house prices for new data

This module is beneficial for anyone who wants to learn how to predict house prices using machine learning. This includes real estate agents, mortgage lenders, investors, and anyone else who wants to make informed decisions about the housing market