**Group Project – California Houses**

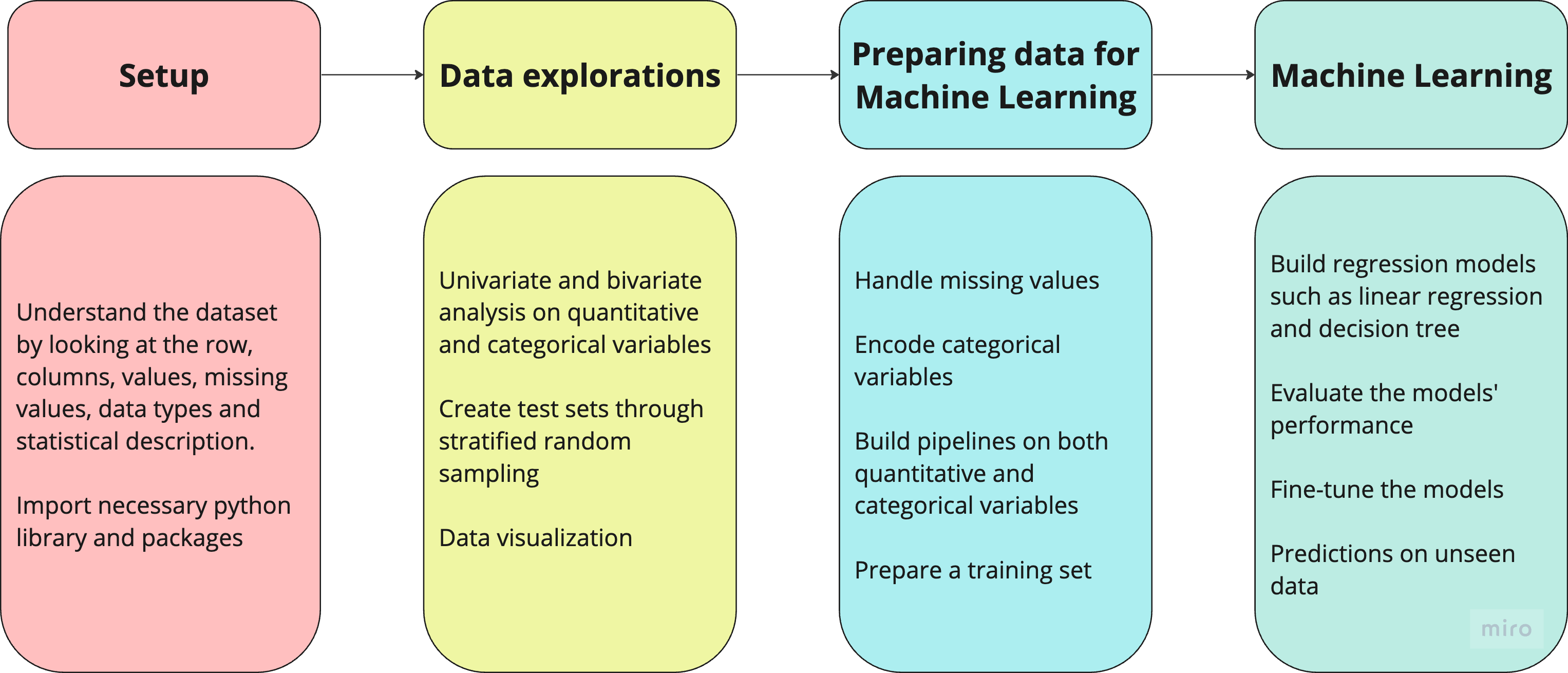
**Course: Introduction of Machine Learning**

**By Group 7 (Ata Avlar, Qiqi Bao, Gautier Hubert, King Man Siu, Romain Thomas)**

**Date: 15 Oct 2023**

**Project Overview**

The project consists of a dataset about the statistics of houses in each Californian districts. The requirements of this project are to conduct data analysis and build models for machine learning, using the dataset. We aim to find out the variables that correlates with the Median House Value and make predictions. In this project, we follow the workflow below:



**Part 0 – Setup**

Below is the summary of what we have done in this part.

* Import necessary libraries and set up working directory
* Understand the dataset, such as data types, missing values, number of rows and columns, statistical description and number of null values.
* Create a categorical variable, Closest City, indicating the closest CA city and drop the distance to each city. Results: (Los Angeles: 9,823, San Francisco: 5,054, San Jose: 3,764, San Diego: 1,999)
* Define the new data frame for the coming sections.

*\*For the attributes and its details, please refer to the notebook.*

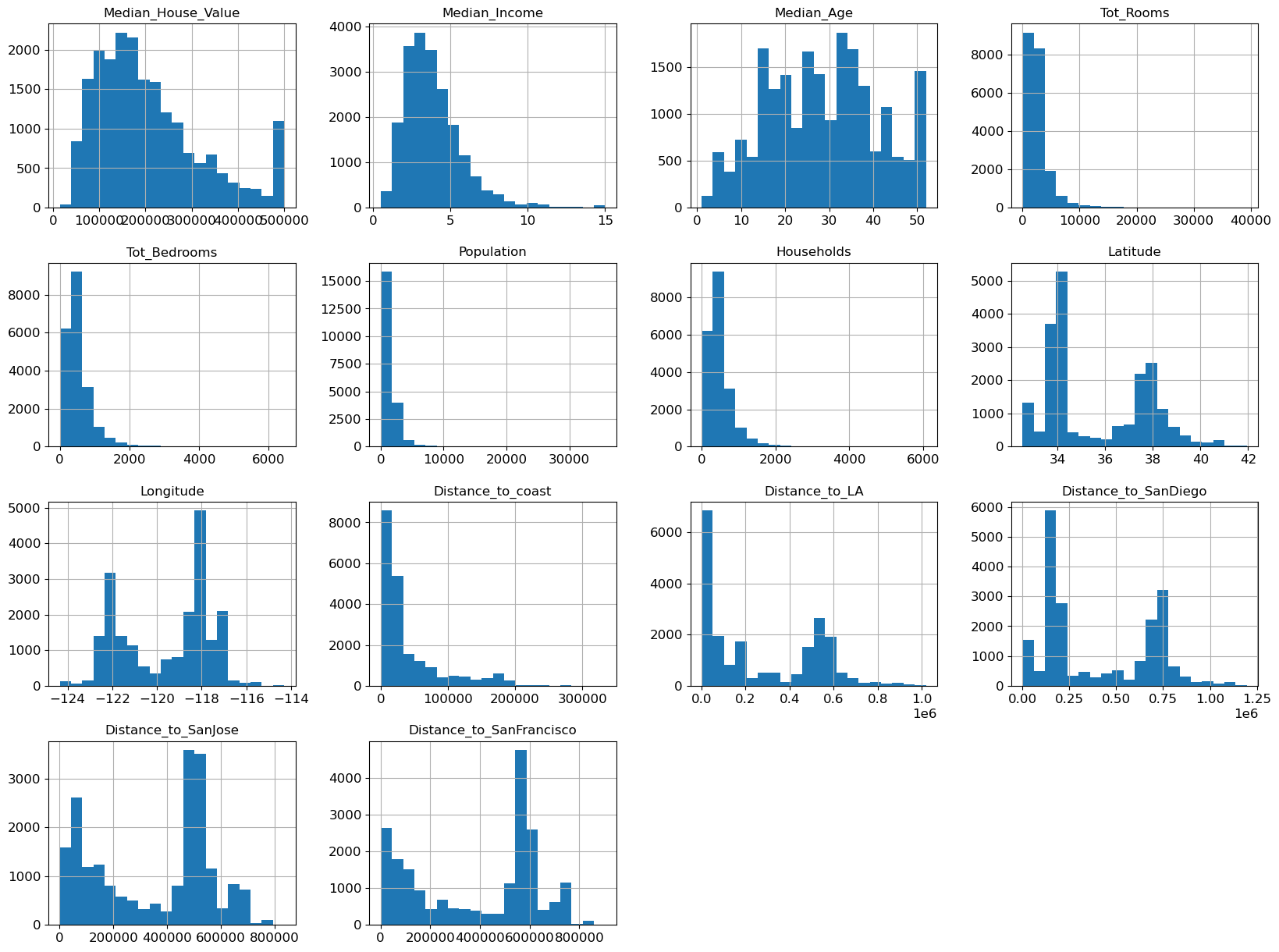
**Part 1 – Data Exploration**

*1.1. Univariate Analysis*

In this part, we explore the dataset by:

* Conducting univariate and bivariate analysis on quantitative and categorical variables
* Visualizing some the data
* Creating test sets through stratified random sampling

Firstly, let’s look at the summary of all quantitative variables with the following histogram.



Here we select some of the variables that we are interested in according to their importance.

* **Median House Value (Target)**: Most of the median house values distribute in the range of 120,000 to 260,000 (standard deviation: 115,000, mean 207,000) with an obvious spike at 500,000.
* **Median Income**: Most median income distributes in the range of 2.6 to 4.7 with a mean of 3.9 and standard deviation of 1.9. Outliers are observed over 10 here.
* **Median Age**: The median age is rather distributed equally in the age groups of 20s, 30s, 40s and 50s. The mean is 28.6 years old, while the standard deviation is 12.6. The minimal age is 1 years old which is believed to be an outlier.
* **Total Numbers of Rooms and Population:** They have a similar pattern as the counts are mostly at less than 6,000 and 2,000 respectively. Small numbers of counts follow.
* **Total Numbers of Bedroom and Household**: They have a similar pattern as well, since most values are in the range of around 300 to 600 and there are small number of counts from then.
* **Distance to Coast**: The common distance to the coast is from 9,000 to 50,000, and there are fewer counts further from the coast.

From the above variables, we pick Median Income as the variable (predictor) to take a closer look, since we suspect that it may relate to the target (Median House Value). The plots below further show the distribution of median income in details by increasing the number of bins. The majority is from 2.6 to 4.7, while some counts scatter above the value of 8.

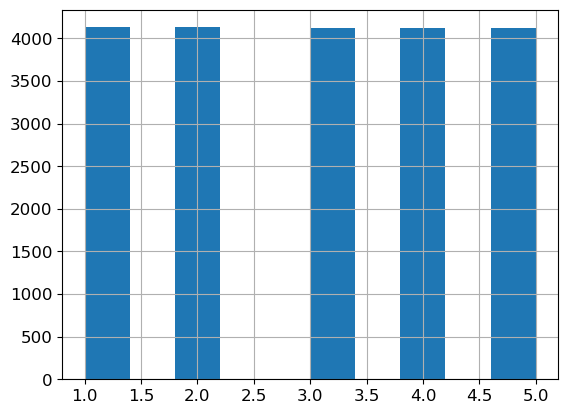
|  |  |
| --- | --- |
| **Median Income Histogram** | **Median Income Boxplot** |

For easier analysis, we turn the median income into a categorical variable by dividing the counts into 5 groups evenly with quantile function, according to the counts. We obtain the following value counts and chart.

Value counts

* Group 1: 4,130 counts
* Group 2: 4,131 counts
* Group 3: 4,123 counts
* Group 4: 4,128 counts
* Group 5: 4,128 counts

Income Category Histogram



*1.2. Bivariate Analysis and Data Visualization*

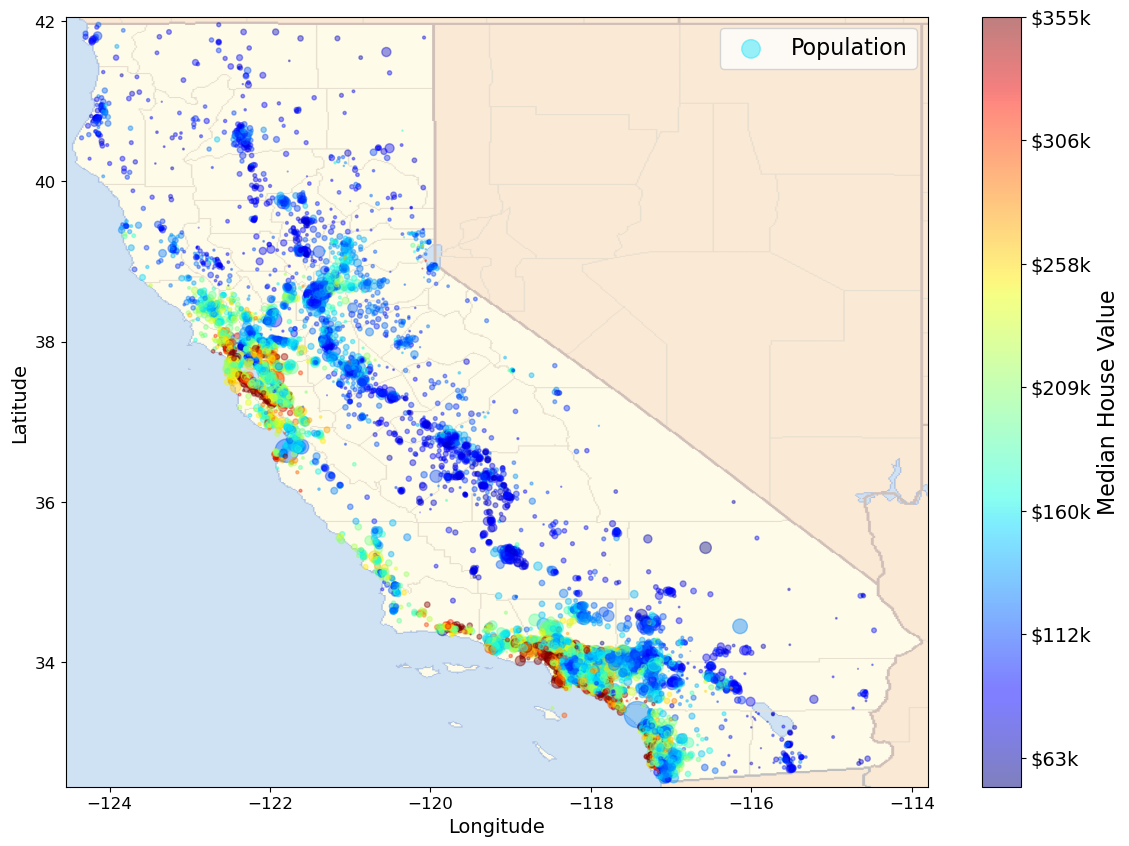
1.2.1. Geographical Data Visualization and Analysis

Geo data is visualized by using different Python packages, such as matplotlib.pyplot and geopandas.

*(\*Regarding the coding, please refer to the notebook for detailed explanations. Results of the process are captured below.)*

|  |  |
| --- | --- |
| **Geographical Points of the Districts in California** | **Geographical Points of the Districts in California** |
| **matplotlib.pyplot version** | **geopandas version** |

**Geographic map of California houses values per district with population density**



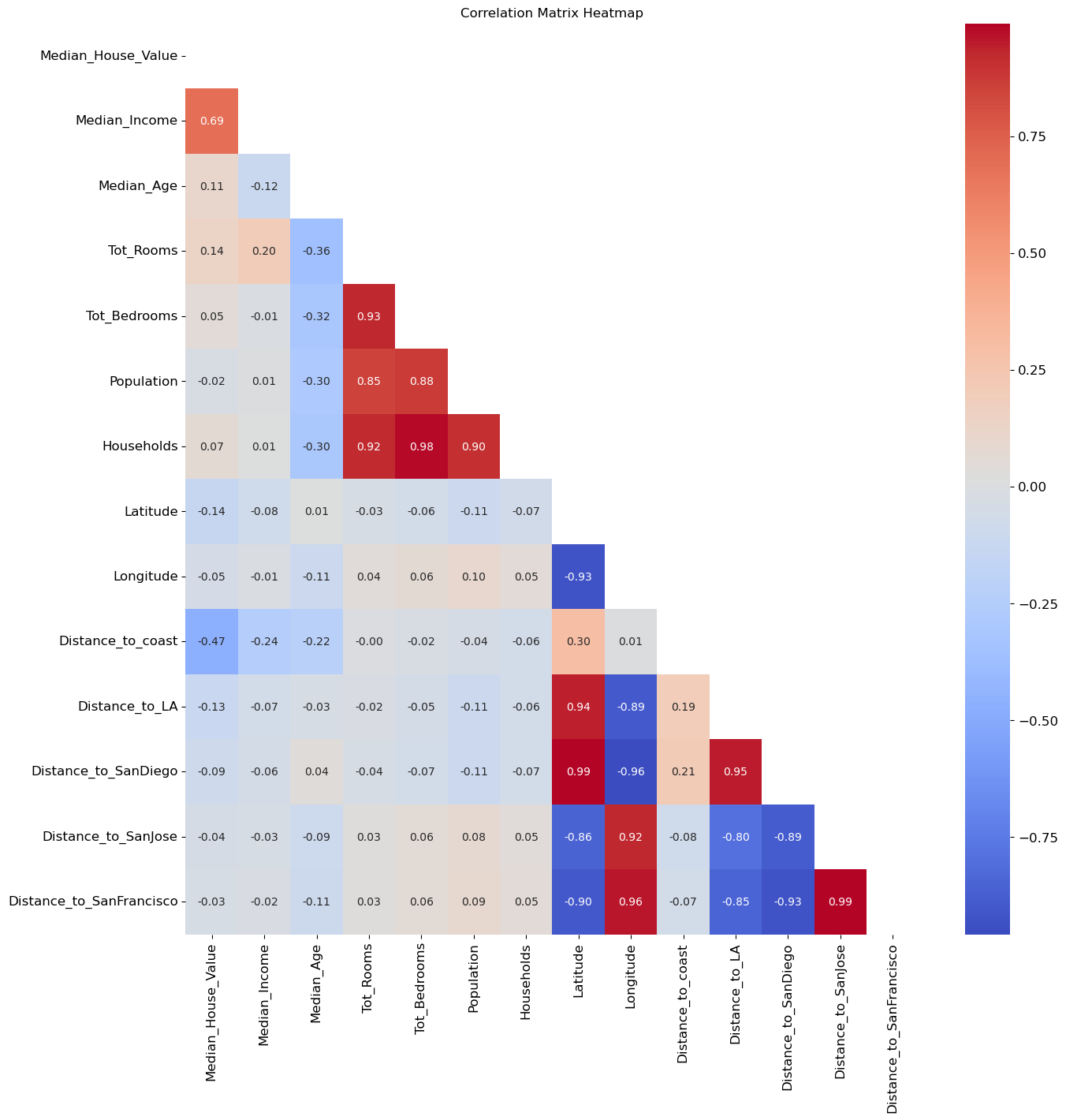
From the plots above, we observe that:

* There are two groups of houses close to the coast. Houses by the coast generally have higher median house values than those located further from the sea.
* There is a group of houses at the middle of California with median house value less than $160k.
* So as the median house values, the population density is higher next to the coast. Population scatters at the middle of California.

1.2.2. Bivariate Analysis

We perform bivariate analysis on all quantitative variables in our data set in this section. Median House Value is chosen as the target here, because the independent variables in the data set potentially affect the values of houses, such as income, age of district, number of rooms, population density, household size (single or family) and distance from the coast. In other words, Median House Value depends on these variables, and it has a higher business value to study.

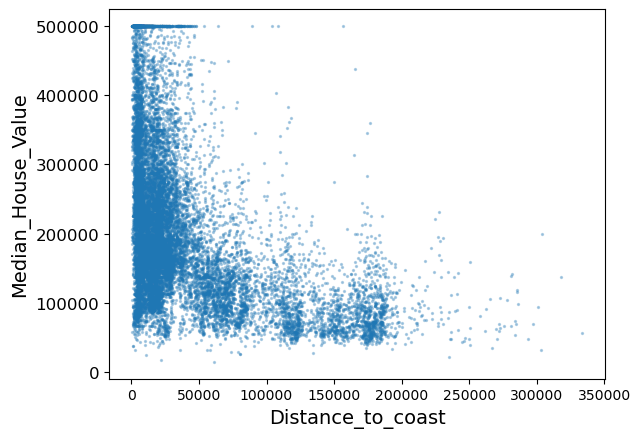
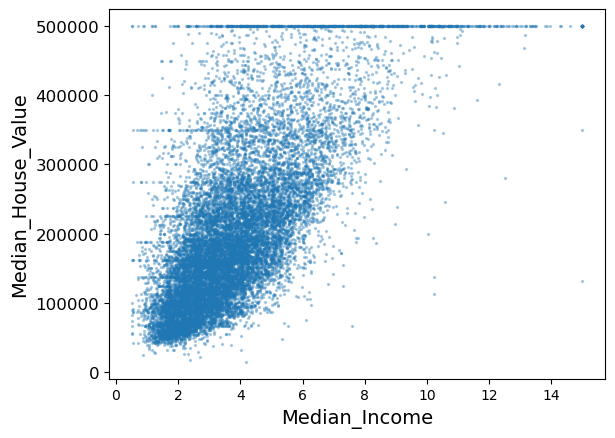
Firstly, let’s take a look at the correlation matrix.

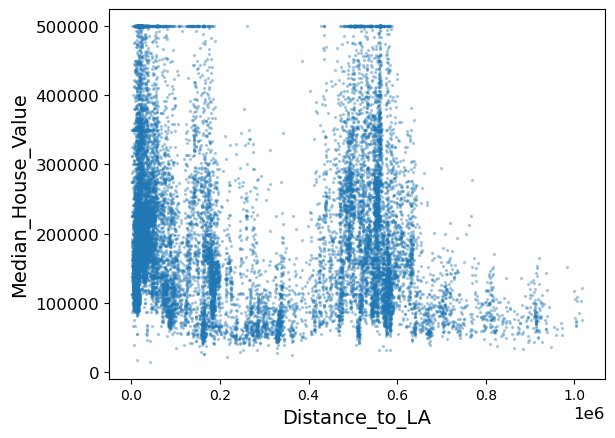
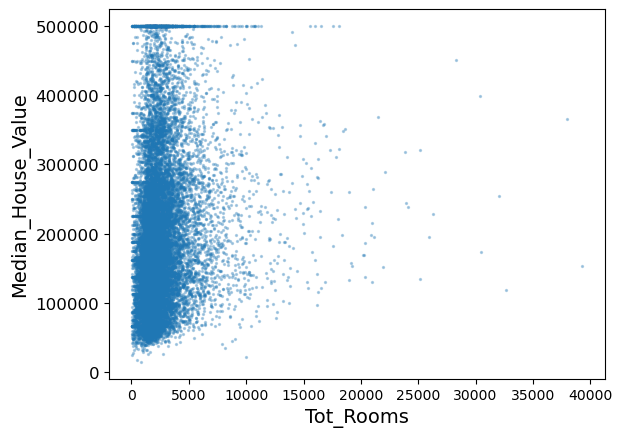


Median Income is the most important correlation with the target variable (Median House Value), as you can see from the table below as well.

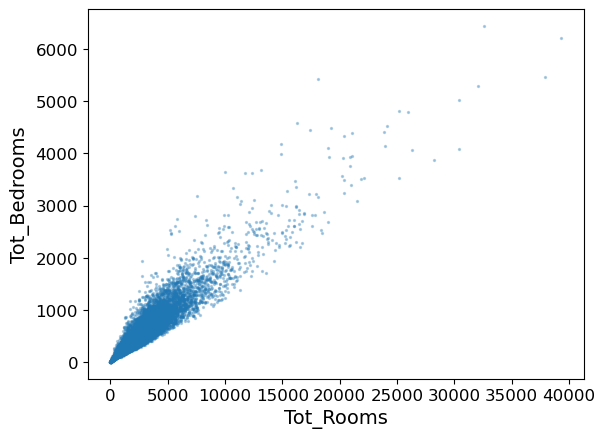
|  |  |
| --- | --- |
|  | **Observations**   * The correlation of Median Income and Median House Value is 0.69 that indicated a moderately strong and positive relation. * The effect size is large which shows a high significance. |

The below scatterplots provide clearer explanations on the correlations between the target and predictors. We show Median Income, Distance to Coast, Total Number of Rooms and Distance to LA here, since their correlation is stronger compared to the other variables.



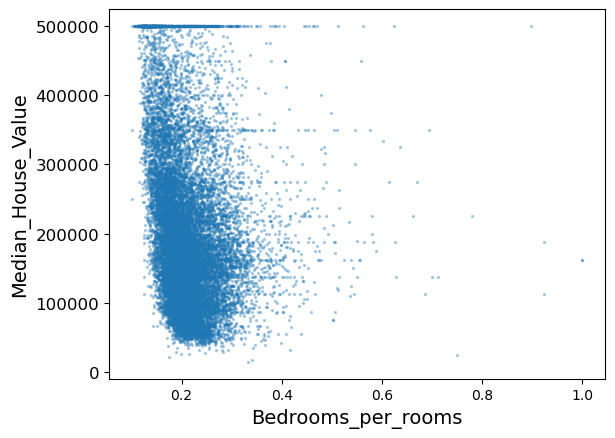
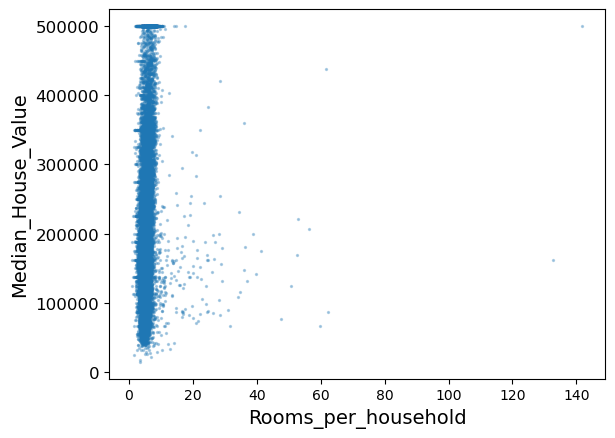


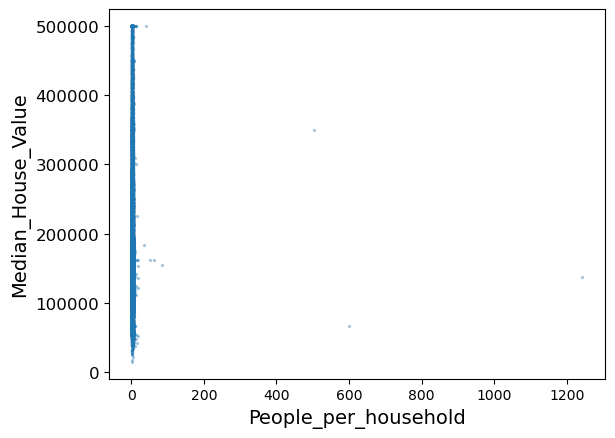
We can see that Median Income has a positive correlation with the target, while Distance to Coast have a negative correlation with the target. The other two do not show strong correlations comparatively.

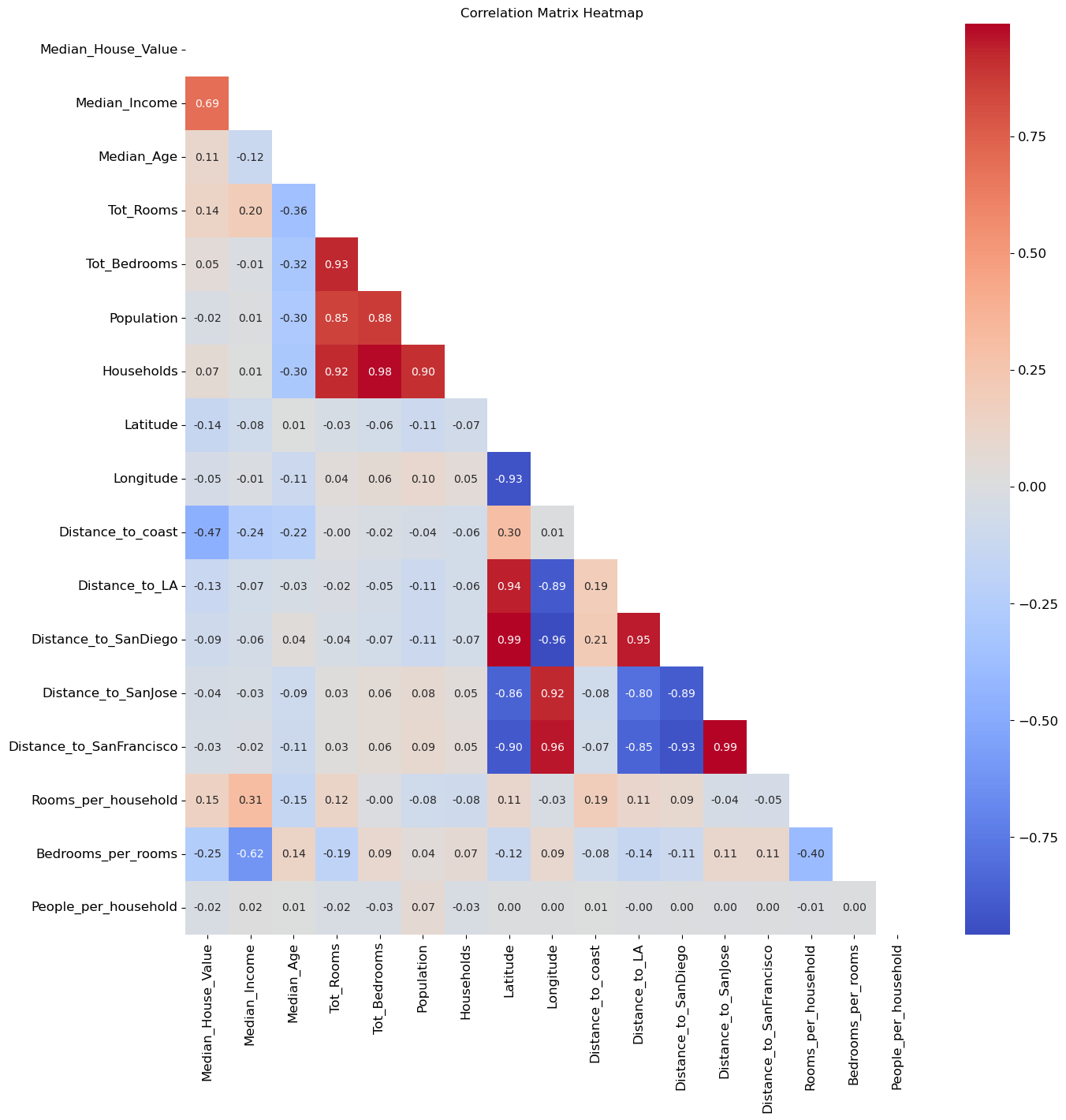


Besides, we found that the total number of rooms or bedrooms per district is not quite meaningful, as they have a very high correlation value of 0.93. We suggest that one of the columns can be dropped.

To facilitate the analysis, three new variables are created, namely Rooms per Household, Bedrooms per Rooms and People per Household. The following scatter plots and updated correlation matrix indicates correlation with the target.







Their correlations with the Target – Median House Values are as followed.

* Rooms per household: A very weak negative correlation coefficient of 0.15
* Bedrooms per rooms: A weak negative correlation of -0.25
* People per household: Considered no relation with a correlation coefficient of 0.02

*1.3. Create a Test Set for Machine Learning*

In this section, we create a test set through stratified random sampling on the income variable. The reason is that we would like to avoid overtraining our dataset. Stratified random sampling ensures the same proportion of the income variable in both of our test and trained dataset.

*(\*Regarding the coding, please refer to the notebook for detailed explanations. Results of the process are captured below.)*

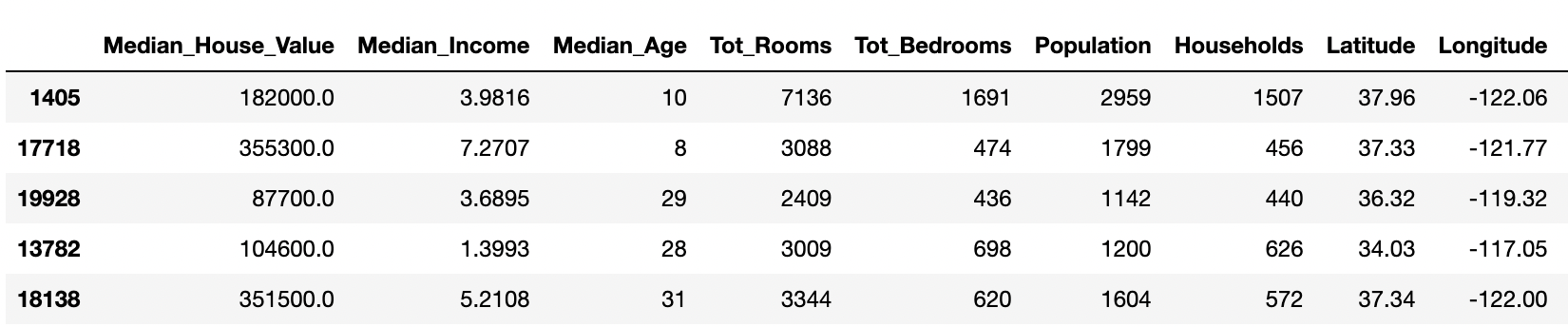
A data frame of income categories is created with 5 columns and 5 indexes (income categories).

* **Overall**: Show the values of the whole "housing" data frame.
* **Stratified**: Give the values of the stratified sample data frame "test\_strat."
* **Random**: Represent the values of the unstratified sample data frame "test\_random" which we created earlier.
* **Rand. %error**: Is the percentage of difference between the unstratified sample and the original data frame.
* **Strat. %error**: Refers to the percentage of difference between the stratified sample and the original data frame "housing."

|  |  |
| --- | --- |
|  | **Observations**   * Proportions of the overall, stratified and random are similar. * The Random. %error is larger than the Strat. %error. * Strat. %error is small. |

We conclude on the stratification that it is much more representative of the overall population to use a stratified split.

Below is a part of the test set for the upcoming part.



**Part 2 – Preparing data for Machine Learning**

In this section, we are going to explain the methodology of preparing data for machine learning.

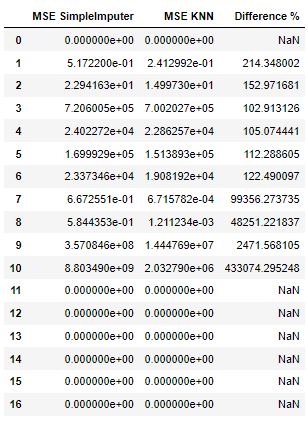
*(\*Regarding the coding, please refer to the notebook for detailed explanations. Results of the process are captured below.)*

*Step 1 - Missing Values*

As part of the simulation, we created missing values randomly, roughly 15% of the first 10 columns of our dataset, aka column index 1 to 10.

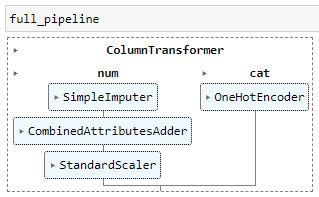
*Step 2 – Handling Missing Values*

* We filled the null values with 2 methods, namely Simple Imputer with median and KNN Imputer.
* From the comparison table below, we can see that the KNN imputation is better than Simple Imputation because its MSEs are lower in every column.
* Only the 10 first feature columns have been implemented with Nan That’s why MSE is >0 only there.

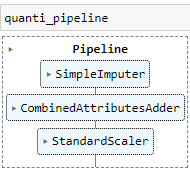


*Step 3 – Building a Pipeline*

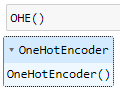
* We used the data frame housing\_df by copying the data frame – train\_strat which is a sample stratified on the column income category of our cleaned data frame from the previous part.
* With the data frame, we started constructing a pipeline that can also be called a transformer.
* Our final pipeline composes 2 transformers, a quantitative transformer and a categorical transformer. The structure is shown below.



* The quantitative pipeline (quanti\_pipeline) includes all of our quantitative features.



* 3 transformations are performed in this pipeline which is applied on the selected quantitative features (quanti\_features).
  + **Simple imputation**: Null values are filled with the medians of the quantitative variables.
  + **Adding combined attributes**: This transformer adds 3 attributes (columns) to our data frame, namely rooms per household, population per household and bedrooms per room. For more details, please refer to the notebook.
  + **Standard Scaler**: It standardizes the quantitative variables to simplify the workflow of machine learning models. It is important for certain machine learning models that assume that the features are normally distributed with mean 0 and variance 1.



* 1 transformation is included in this pipeline regarding our categorical variables (cat\_features).
  + **One Hot Encoding (OHE)** is conducted. We apply this on both nominal and ordinal features to prevent our models to be biased with hierarchical order or the ordinals.
* The output of the overall pipeline is an array of array in which each array is a row of our database ready for training our models.
* We need to convert this array into a data frame and reset the column names with the corresponding names.
* The data types of all variables are float, which are ready for building models.

**Part 3 – Machine Learning**

*3.1. Learning and evaluating with the training set*

Defining Target and Features

* The target variable for prediction is "Median\_House\_Value".
* The predictors (features) include: Median Age, Distance to coast, Rooms per household, Bedrooms per room, Population per household, 'Closest city' One Hot Encoded categorical variables, One Hot Encoded 'Income category' variables.

Linear Regression with Stochastic Gradient Descent (SGD)

* MSE: 6416067380.90
* MAE: 60107.33

The linear regression model using SGD has a noticeable error. It would be valuable to compare this performance with other models to determine if this is optimal.

Decision Tree Regression

* MSE: 0.0
* MAE: 0.0

The Decision Tree Regression shows a perfect score (MSE and MAE both are 0). This might suggest the model has overfitted the training data.

*3.2. Estimating the models with cross validation*

Linear Regression:

* MSE: 6408588781.48
* Mean of Cross Validation R^2: 0.518

Elastic Net:

* MSE: 6604816389.60
* Mean of Cross Validation R^2: 0.503

Decision Trees:

* MSE: 9534949874.78
* Mean of Cross Validation R^2: 0.281

Random Forests:

* MSE: 4984241350.50
* Mean of Cross Validation R^2: 0.624

Support Vector Machines:

* MSE: 9534949874.78
* Mean of Cross Validation R^2: 0.049

Among the five models, the Random Forest model has the highest mean score (0.5162), suggesting it's performing better in terms of R-squared. It also has the lowest MSE score which indicates that it is the most suitable algorithm for our model under these conditions.

Note: MSE for Decision Tree shows that cross validation removes overfitting by simply splitting the data set as train and test (Our cv is 5 so it trains the model 5 times with 5 different test train combinations.

3.3. *Performing hyperparameter tuning using GridSearch and RandomizedSearch*

Random Forest: {'n\_estimators': 100, 'max\_features': 8}

Elastic Net:

* Grid : {'alpha': 0.001, 'l1\_ratio': 0.8}
* Randomized: {'l1\_ratio': 1.0, 'alpha': 0.001}

Decision Tree:

* Grid: {'min\_samples\_leaf': 39, 'min\_samples\_split': 1}
* Randomized: {'min\_samples\_split': 8, 'min\_samples\_leaf': 45}

The tuning results for the Random Forest model suggest that both Grid Search and Randomized Search identified the same optimal hyperparameters and performance. The Other models have different optimal hyperparameters.

*3.4. Prediction and Accuracy*

We used hyperparameters that we found from previous part. Our RMSE results are:

Random Forest: 68963.03

ElasticNet: 77789.13(Grid), 77786.81(Randomized)

Decision Tree: 72713.63 (Grid), 72434.99 (Randomized)

Linear Regression: 77773.53

The Random Forest model has the lowest RMSE, indicating it's the most accurate among the four models for this dataset. The Linear Regression and ElasticNet models ha the highest RMSE, suggesting it may not be as suitable for this task compared to the other models.

*3.5. Conclusion*

**Overfitting**: The Decision Tree Regression, with its impeccable MSE and MAE of 0, suggests potential overfitting. This model might have adapted too closely to the training data, affecting its generalization to new data. The discrepancy between its initial scores and the cross-validation MSE reinforces this.

**Tuning a Learner**: Hyperparameter tuning plays a vital role in optimizing a model's performance. The application of GridSearch and RandomizedSearch on models like Random Forest and Elastic Net highlights the significance of refining model parameters. For the Random Forest model, consistent optimal parameters across both methods emphasize their robustness.

**Model Performance**: The Random Forest model, with the highest R^2 and the lowest MSE, stands out as the top performer for this dataset. In contrast, the Elastic Net and Linear Regression models, with their substantial errors, seem less suitable.

Additional Learning Experience:

**Feature Engineering**: Defining targets and predictors is crucial. One Hot Encoded variables can impact model performance by accommodating categorical data.

**Model Comparison**: Different models present varied strengths. While the Decision Tree suggested overfitting, the ensemble-based Random Forest showcased superior performance, emphasizing the advantages of ensemble methods.