Team Caserta - California Census Data

1. MAVILLAPALLI VENKATA TARUN KUMAR (P37000126)
2. GATTEM PRIYA MADHURI (P37000162)
3. KARRI RAHUL REDDY (P37000157)

Presenting to :
ELIO MASCIARI

Agenda

Exploration of California census data

Data Pre-Processing

Data Preparation

Model Development and Training

Model Evaluation and Validation

Fine Tuning




Introduction

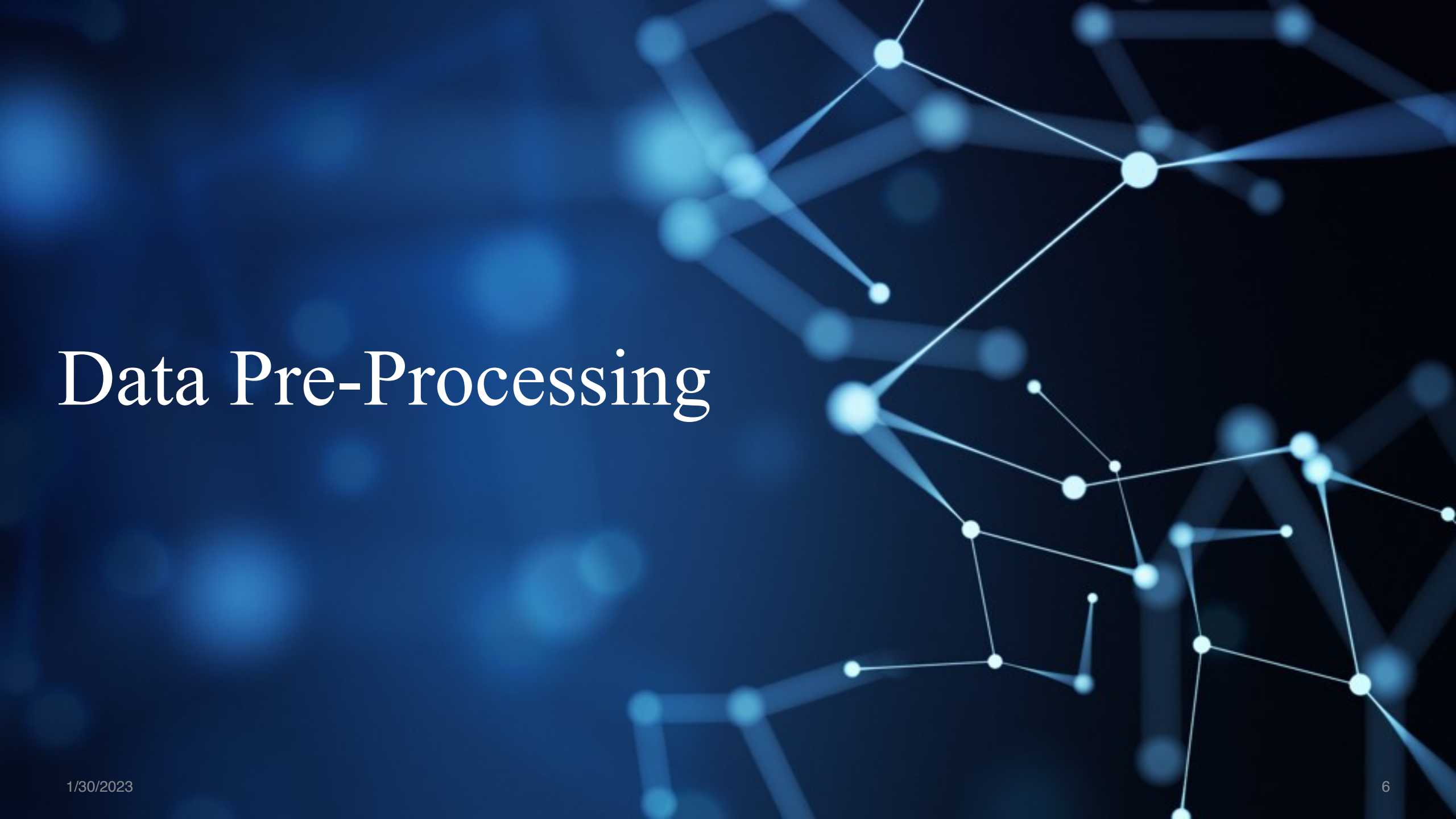
- Our task is to use the California census data to build a model to find the housing prices in the state. This data includes features such as Population, Median Income and Median housing price for each block group in California.
- A block group has a population between 600 to 3,000. We will call them "districts" for short. Our model should be able to predict the median housing price for any district.
- Also to measure the performance of the model we have used the Root Mean Squared Error (RMSE).

Exploration Of California Census Data

- The data is California census data to build a model of the housing prices in the state.
- The CSV file provided contains 10 columns and 20,640 rows.
- The below table shows the column names their description and datatype

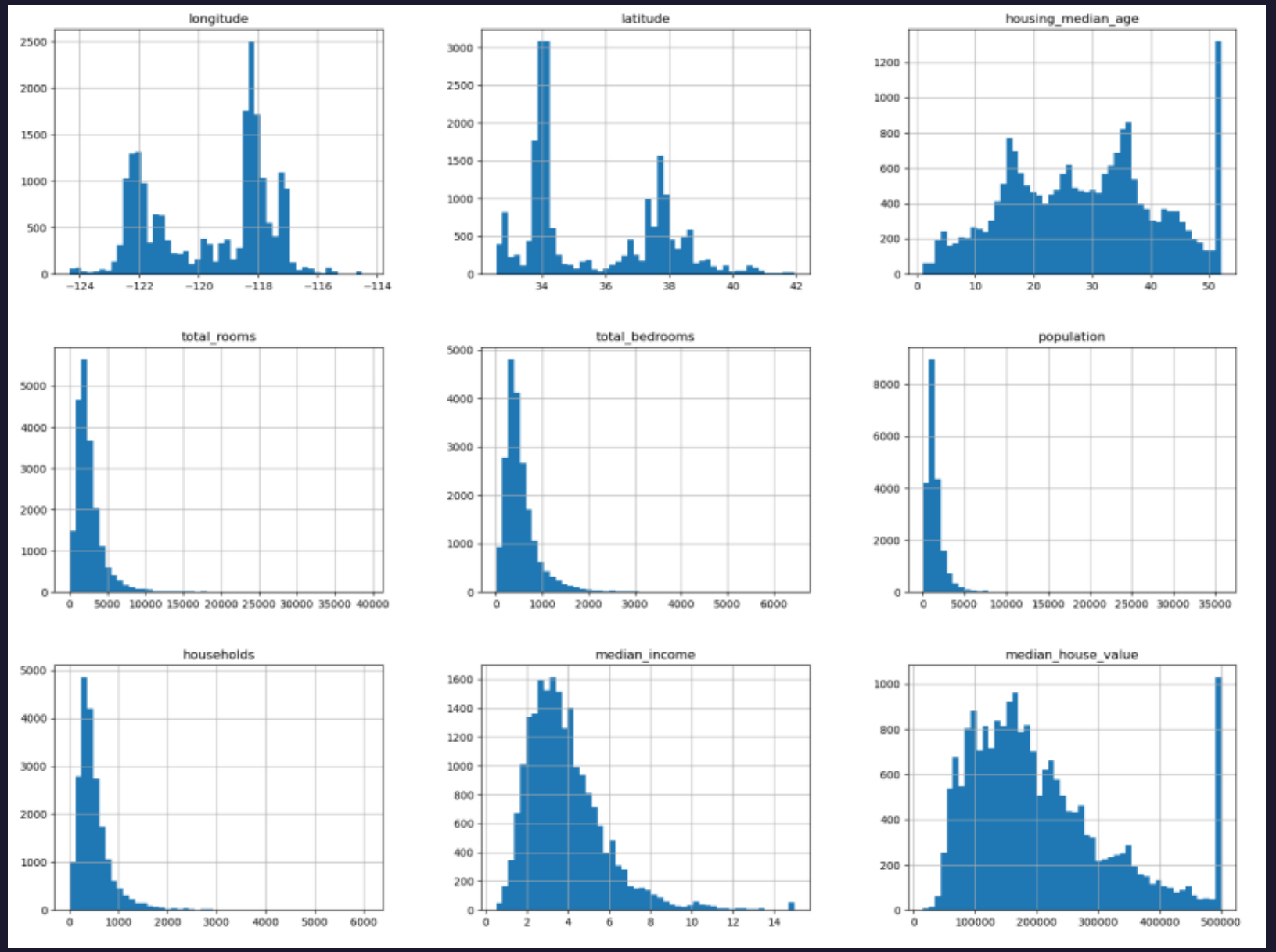
Column Name	Description	Data Type
longitude	A measure of how far west a house is; a more negative value is farther west	float64
latitude	A measure of how far north a house is; a higher value is farther north	float64
housing_median_age	Median age of a house within a block; a lower number is a newer building	float64
total_rooms	Total number of rooms within a block	float64
total_bedrooms	Total number of bedrooms within a block	float64
population	Total number of people residing within a block	float64
households	Total number of households, a group of people residing within a home unit, for a block	float64
median_income	Median income for households within a block of houses (measured in tens of thousands of US Dollars)	float64
median_house_value	Median house value for households within a block (measured in US Dollars)	float64
ocean_proximity	Distance of house from ocean	Object

Data Pre-Processing



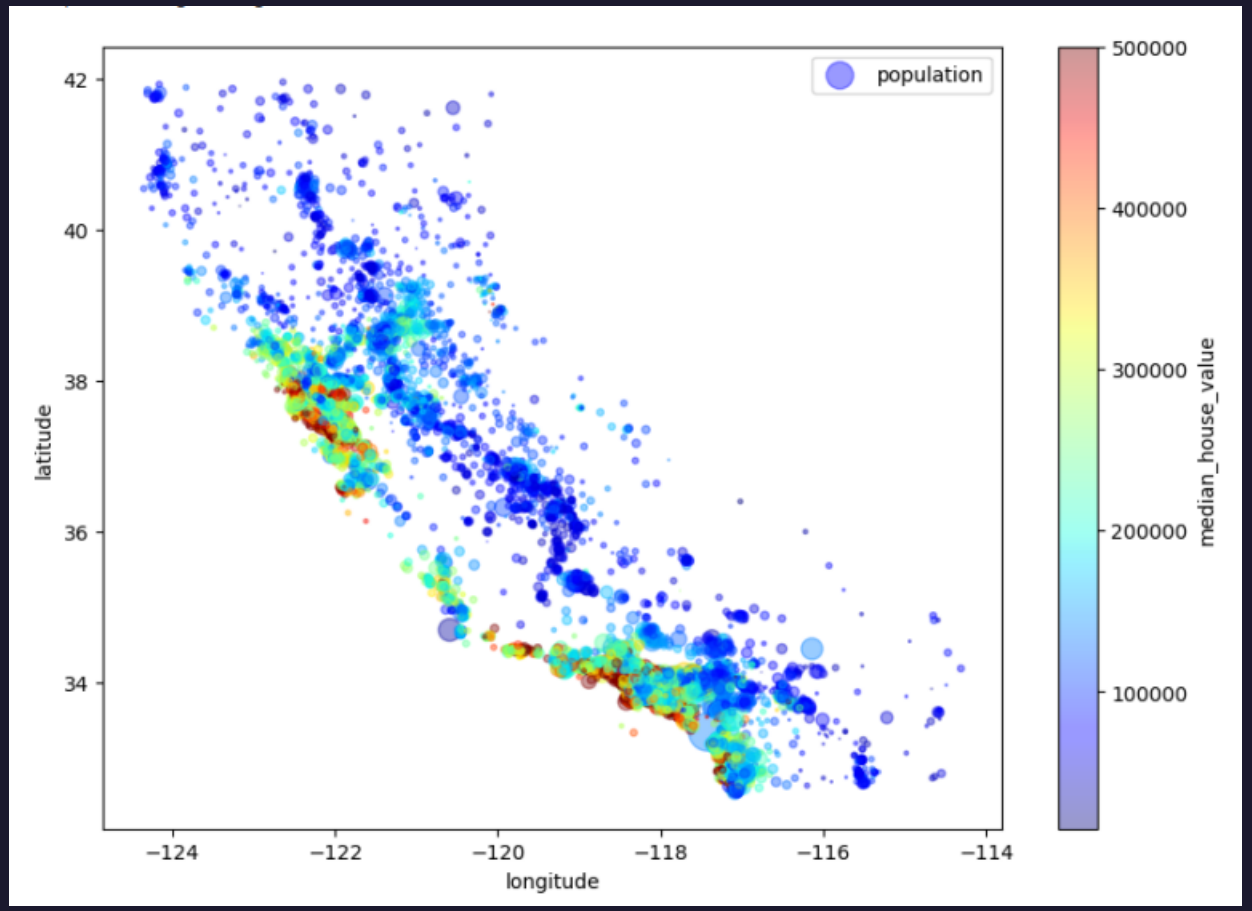
Histograms for the housing data

- The right side plots shows spread of data of all the columns in the data set.
- The attributes have very different scales.
- The attributes are tail-heavy & they often extend to the right than to the left.
- As a result, It will be difficult for many machine learning algorithms to find patterns within the data.



Visualization Of The Geographical Data

- We can see in the scatter plot the data represents map of California.
- This image tells us that the median housing price is related to location.
- The houses closer to the sea tend to be more expensive.
- The highest population can be spotted where the house price is low (around 100000$)



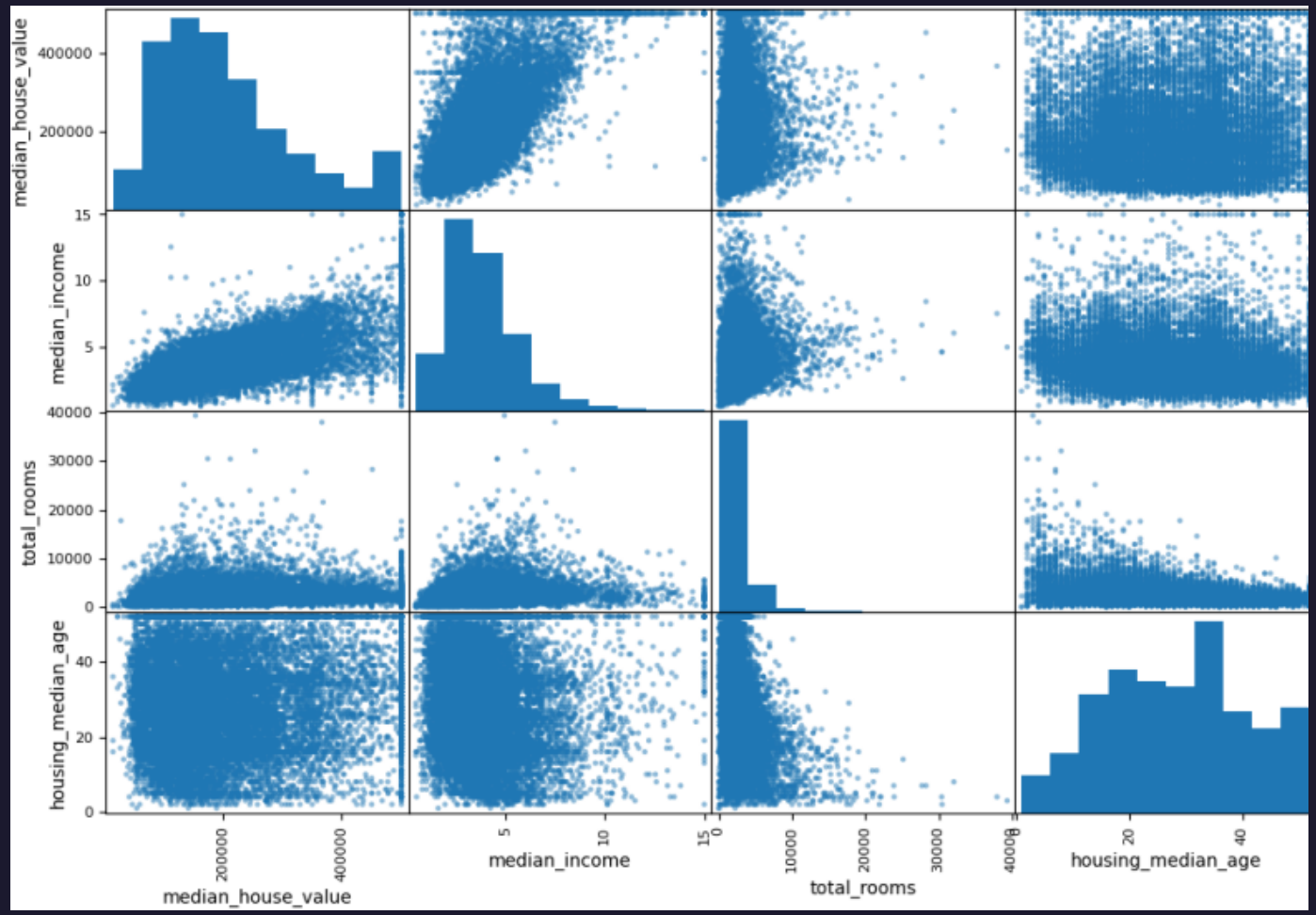
- We have framed a problem, plot histograms & draw conclusions from it, split data into training & testing subsets and We worked on stratified sampling.
- We also plotted scatterplots, computed a correlation coefficient table.

- As we can assume from below Correlation Matrix median_house_value and median_income are highly correlatable.

```
corr_matrix = housing.corr()
corr_matrix['median_house_value'].sort_values(ascending=False)
```

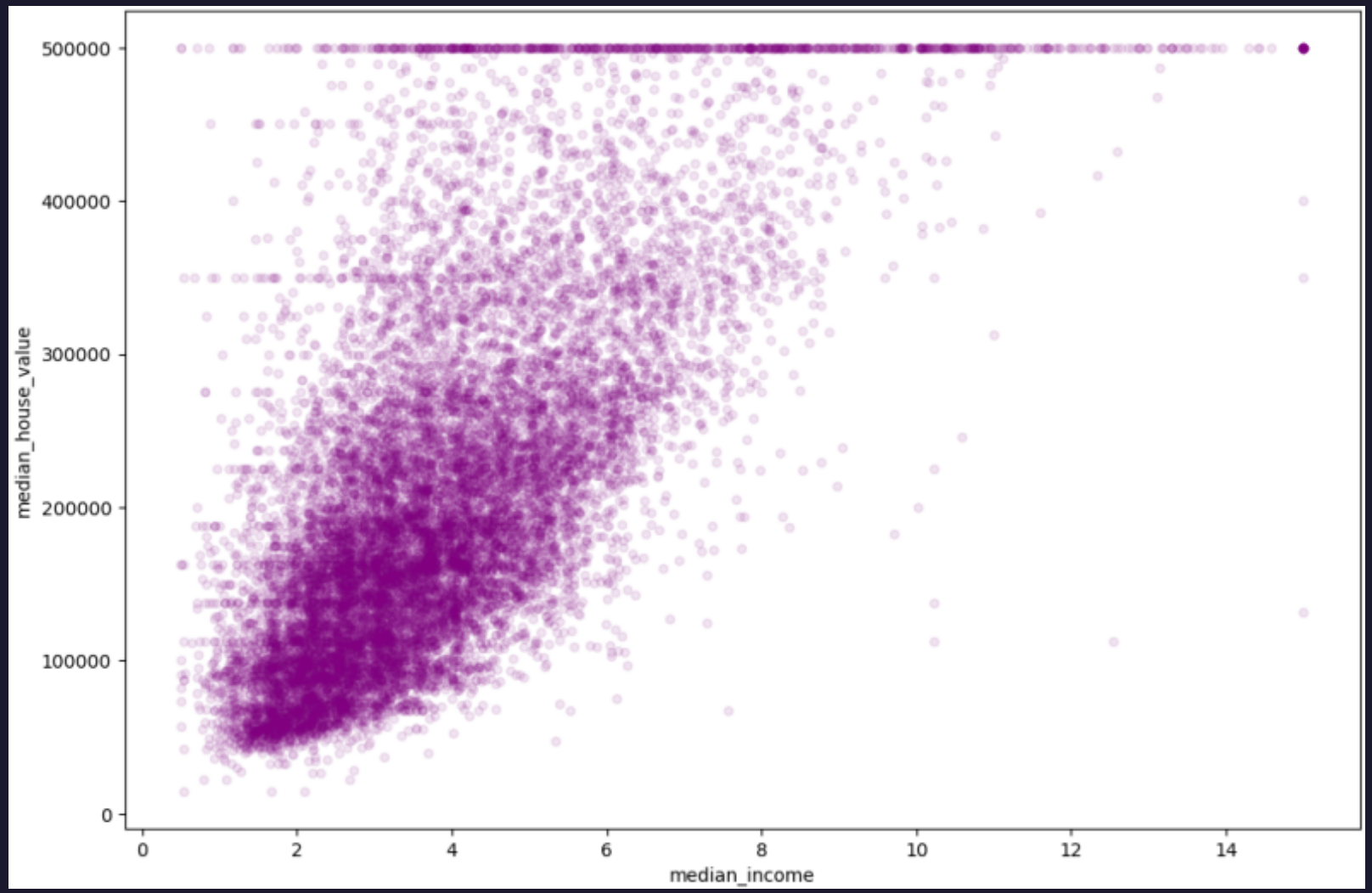
Out[19]:

```
median_house_value    1.000000
median_income         0.687151
total_rooms           0.135140
housing_median_age    0.114146
households            0.064590
total_bedrooms        0.047781
population           -0.026882
longitude            -0.047466
latitude             -0.142673
Name: median_house_value, dtype: float64
```



We observed from the plot:

- The correlation among the median_house_value and median_income is indeed very strong, we can clearly see the upward trend.
- The price cap that we noticed earlier is clearly visible at 500,000 USD, but the plot reveals other less obvious lines at: USD450K, USD350K, USD280K and so on.
- We may want to remove the corresponding districts so that the model to not learn these quirks.



Data Preparation

Data Preparation

- In terms of data preparation and cleaning, we talked about missing values and filled them
- We found that there were 207 missing values in total_bedrooms column and using scikit-learn's, SimpleImputer to calculate the median of all attributes and store them in .statistics_.
- Used the "trained or fitted" imputer to transform the numerical attributes by replacing missing values with their corresponding medians, hence transforming the dataset.
- We processed categorical attributes and text, using One hot encoding and reshaped arrays along the way.
- Furthermore we used the sklearn pipeline class, we used ColumnTransformers and used RMSE to evaluate our models and learned about under fitting.

Model Development and Training

Model Development and Training

- In Model development and training we used a Linear Regression model, a Decision Tree Regressor and a Random Forest Regressor.
- Linear Regression is our first model and RMSE value is 68627.87390018745 which is an example of a model overfitting the data.
- Then we performed with Decision Tree Regressor as this is a powerful model which is capable of finding non-linear relationships within the data RMSE value is 0.0
- Random Forest Regressor the RMSE value is 18714.068083362738

Model Evaluation and Validation

Model Evaluation and Validation

- Fortunately our data set is small enough to do the cross-validation
- We have used K-fold cross-validation. We randomly split the training data into 10 folds, we iteratively train the model on 9 folds and evaluate on 1, doing this 10 times. We will end up with 10 metric scores.
- We should notice that cross validation allows us to not only get an estimate of the performance of our model (mean), but how precise it is (std)
- The decision tree seems to perform worse than the linear regression model
- Random forests seem promising. We should notice, that the RMSE on the training set is still much lower than the validation RMSE, meaning the model overfitted, but not as badly as the decision tree model.
- We would not have this estimation if we used only one validation set. However, cross-validation comes at the cost of training the model several times, which is not always possible.

Here are the models and there overall performances.

Model	Mean(Performance of our model)	STD(how precise it is)	Root Mean Square Error
Decision tree	71267.74015794141	2775.775402478205	0.0(It is either the model is absolutely perfect, or it badly overfit the data.)
Linear Regression	69104.07998247063	2880.3282098180675	68627.87390018745
Random Forest	50258.86992738761	2301.608143750906	18714.068083362738

- From the above results of RMSE and doing cross validation for every model we found out Random Forest Regressor best suites for our data set.
- So we proceed forward with Random Forest Regressor for fine tuning.

Fine Tuning

Fine Tuning

- Random forest performs best among all the regressors we tried, so we choose random forest to perform fine-tuning using grid search. Fine-Tuning is to adjust precisely so as to bring to the highest level of performance or effectiveness.
- The model performs slightly better than a random forest with default hyper-parameters.
- Finally evaluated our system on the test set. Where the final_rmse is 47634.249089946396. In some cases, such a point estimate of the generalization error won't be enough
- We want to create a confidence interval of 95% around the metric. For this, we use the individual predictions for each test set element.
- We got an array([45714.24158649, 49569.76325317]) which is median_housing_price range for the dataset
- If we do a lot of hyper-parameter fine-tuning, we will end up with a slightly worse performance on the test set because we will sometimes overfit to the changing validation set.

Conclusion

We have measured the performance of different models using the Root Mean Squared Error (RMSE). Also predicted the median_housing_price.

Random forest model was best suited for our data set. Hence used it for the fine tuning using grid search.

Thank You

1. MAVILLAPALLI VENKATA TARUN KUMAR (P37000126)
2. GATTEM PRIYA MADHURI (P37000162)
3. KARRI RAHUL REDDY (P37000157)