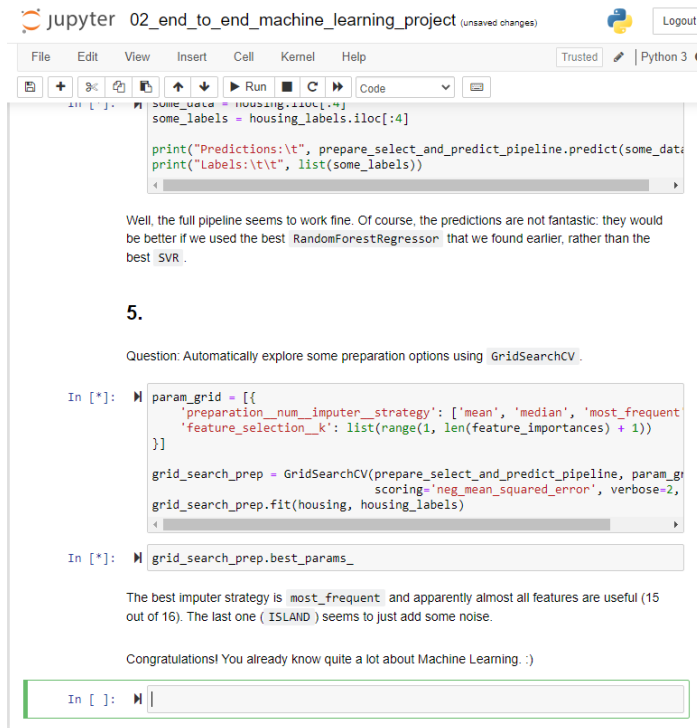


1. Include a snapshot of the Juniper notebook in your machine that shows the completion of the project.



The screenshot shows a Jupyter Notebook interface with the title '02\_end\_to\_end\_machine\_learning\_project (unsaved changes)'. The notebook contains the following content:

```
some_data = housing.iloc[:4]
some_labels = housing_labels.iloc[:4]

print("Predictions:\t", prepare_select_and_predict_pipeline.predict(some_data))
print("Labels:\t\t", list(some_labels))
```

Well, the full pipeline seems to work fine. Of course, the predictions are not fantastic; they would be better if we used the best `RandomForestRegressor` that we found earlier, rather than the best `SVR`.

5.

Question: Automatically explore some preparation options using `GridSearchCV`.

```
In [*]: param_grid = [{
    'preparation__num_imputer_strategy': ['mean', 'median', 'most_frequent'],
    'feature_selection__k': list(range(1, len(feature_importances) + 1))
}]

grid_search_prep = GridSearchCV(prepare_select_and_predict_pipeline, param_grid,
                                scoring='neg_mean_squared_error', verbose=2,
                                grid_search_prep.fit(housing, housing_labels))
```

```
In [*]: grid_search_prep.best_params_
```

The best imputer strategy is `most_frequent` and apparently almost all features are useful (15 out of 16). The last one (`ISLAND`) seems to just add some noise.

Congratulations! You already know quite a lot about Machine Learning. :)

```
In [ ]: |
```

2. Which python functions help explore data?

To visually explore the data, **.plot()** was used to create a scatterplot of geographical information, with various parameters to display patterns more distinctly. To see how much each attribute correlates with the median house value, **.corr()** generated the correlation coefficient running from -1 to 1, and **.sort\_values()** helped organize them. Also, **scatter\_matrix()** with 4 most promising attributes created plots revealing crucial information about the data.

In a broader sense, various functions from python packages were utilized for data exploration. To load the data to begin exploring, `.read_csv()` from pandas was used, and for quick overview, summary and information about the data, `.describe()`, `.value_count()`, `.hist()`, etc. Also, functions from sklearn package help exploring samples of data, cleaning out the data for better exploration, handling data quality issues, scaling the data, transforming the data, and more. To list few more used for the project, `.plot()`, `corr()`, `.split()`, `.drop()`, `.fit_transform()` (for `SimplerImputer()`, `OrdinalEncoder()`), `StandardScaler()`, `pipeline()`, `ColumnTransformer()` `.fit()`, `.predict()` (for regression models)

**3.** Indicate the modal distributions of data in Figure 2-8 (e.g., symmetric/skewed, unimodal/bimodal, truncated) ignoring outliers.

- a. Household: unimodal, right-skewed
- b. Housing\_median\_age: bimodal, symmetric (ignoring the outlier)
- c. Latitude: bimodal, right-skewed
- d. Longitude: bimodal, left-skewed
- e. Median\_house\_value: unimodal, right-skewed (ignoring the outlier)
- f. Median\_income: unimodal, right-skewed
- g. Population: unimodal, right-skewed
- h. Total\_bedrooms: unimodal, right-skewed
- i. Total\_rooms: unimodal, right-skewed

**4.** Which sampling approaches were used, and which better reflected the original data?

**Random Sampling method and stratified sampling methods were used. Stratified sampling reflected the original data better** as it generated the test set that is almost identical to the overall data set while random data set resulted in much higher percentage error, and it's because for stratified sampling, the right number of instances is pulled out from each stratum(homogeneous subgroups), therefore representing the overall population

**5.** Which pair of attributes had the most correlation?

**"median\_income" and "median\_house\_value"**

**6.** Which new attributes were generated from the original data?

For stratified sampling based on the income category, the new attribute **"income\_cat"** was generated with 5 categories (labeled from 1 to 5). For attribute combinations, **"rooms\_per\_household", "bedrooms\_per\_room" and "population\_per\_household"** were generated. `housing.reset_index()` # adds an `'index'` column

**7.** How were the following data quality issues handled (a. missing values, b. textual attributes, c. varying attribute value ranges)?

**a. Missing values:**

Out of 3 options suggested,

1. get rid of the corresponding districts
2. get rid of the whole attribute
3. set the values to some values,

**the project proceeded further with the third option of setting the values to some values, the median** for this case, by using a handy class, `SimplerImputer()`

**b. Textual attributes:**

**OneHotEncoder()** class imported from `sklearn.preprocessing` package was used to resolve issues generated from the use of `OrdinalEncoder()` that makes machine learning algorithms to assume that two nearby values are more similar than two distant values. So, the `OneHotEncoder()` class converted categorical values for the “ocean\_proximity” column into one-hot vectors. `.fit_transform()` generated a Scipy sparse matrix, then the `.toarray()` method converted it to a dense NumPy array.

**c. Varying attribute value ranges:**

**Standardization** was used to get all attributes to have the same scale since it's much less affected by outlier than min-max scaling as known as normalization. To be specific, `StandardScaler`, a transformer from `sklearn.preprocessing` was used to scale the ranges of values.

**8.** Which model performed the best?

**Random forest Regressor Model** performed the best though the model seems it needs more training data and be regularized.

**9.** What are two important take-home messages from this exercise?

The exercise covered a lot of helpful use of functions from python packages, and one of the two most important messages from this exercise is that we have to view a big picture/ frame of the problem with a business objective. The second one is that there is a lot of decision making in order to get to the final model, hence, at every step dealing with the data, we have to make a right decision with logical thinking and evidence. For example, we have to decide one method over another, one model over another, and one test set over the other, therefore, making an optimal decision considering how the model will be used in the future to achieve a business goal is very important.