**Project 1 : California Housing Price Prediction**

* Code Snippet followed a Screenshot of the output

Q1. Load the data and print first few rows of this data

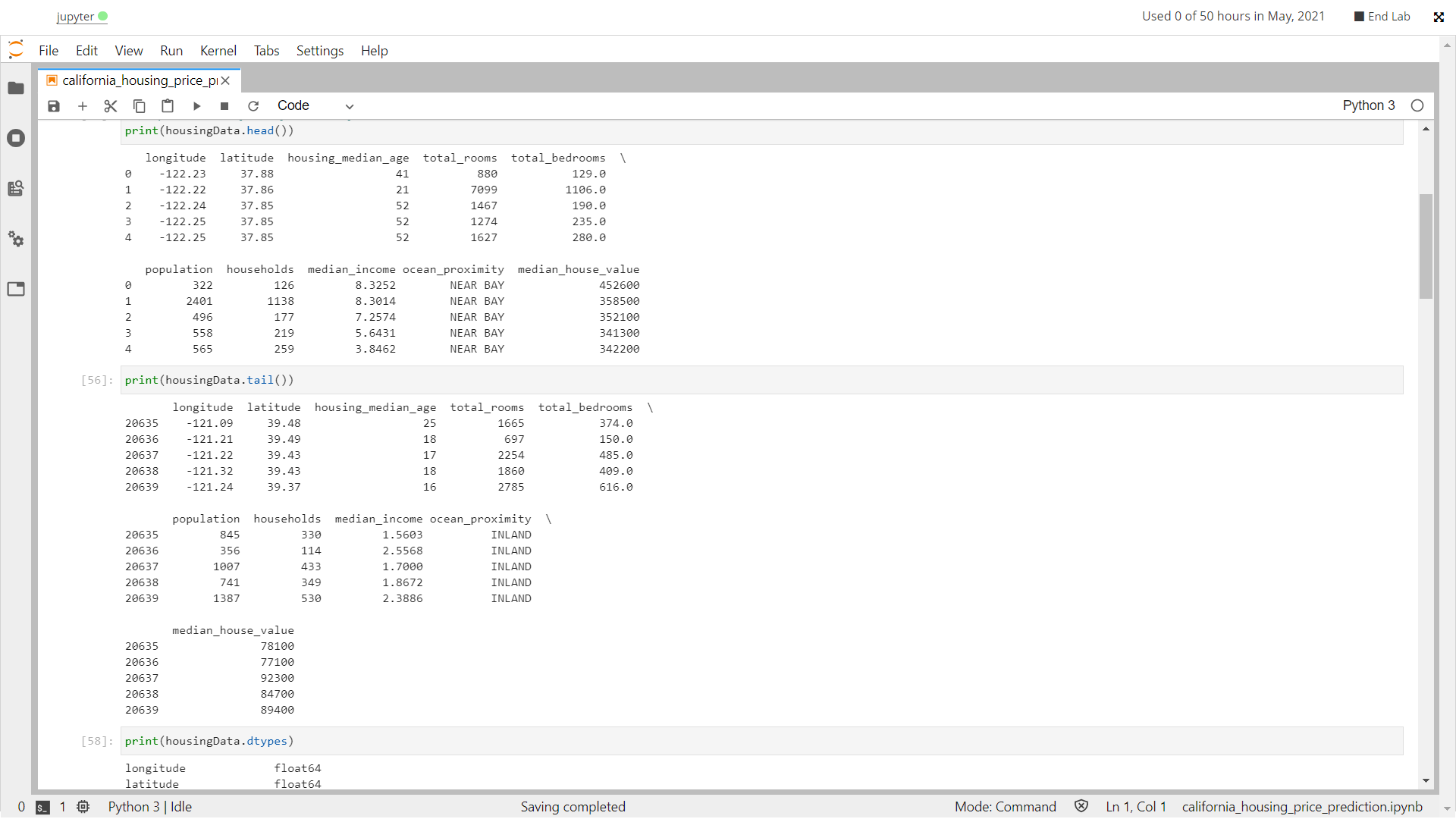
# Step1: Load the data

# Step1.1: Read the “housing. xlsx'” file from the folder into the program

housingData =pd.read\_excel (r'housing.xlsx')

# Step2.2: Print first few rows of this data

housingData.head()



Q2. Separate features and labels, deal with missing value, Encode categorical data, Split the dataset into training and testing set, standardized the data

# Step2.3: Extract input (X) and output (y) data from the dataset

X = housingData.iloc[:, :-1].values

y = housingData.iloc[:, [-1]].values

# Step3: Handle missing values:

# Fill the missing values with the mean of the respective column

from sklearn.impute import SimpleImputer

missingValueImputer = SimpleImputer(missing\_values=np.nan, strategy='mean')

X[:, :-1] = missingValueImputer.fit\_transform(X[:, :-1])

y = missingValueImputer.fit\_transform(y)

# Step4: Encode categorical data:

# Convert categorical column in the dataset to numerical data

from sklearn.preprocessing import LabelEncoder

X\_labelencoder = LabelEncoder()

X[:, -1] = X\_labelencoder.fit\_transform(X[:, -1])

# Step5: Split the dataset: Split the data into

# 80% training dataset and 20% test dataset

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

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size = 0.2, random\_state = 0)

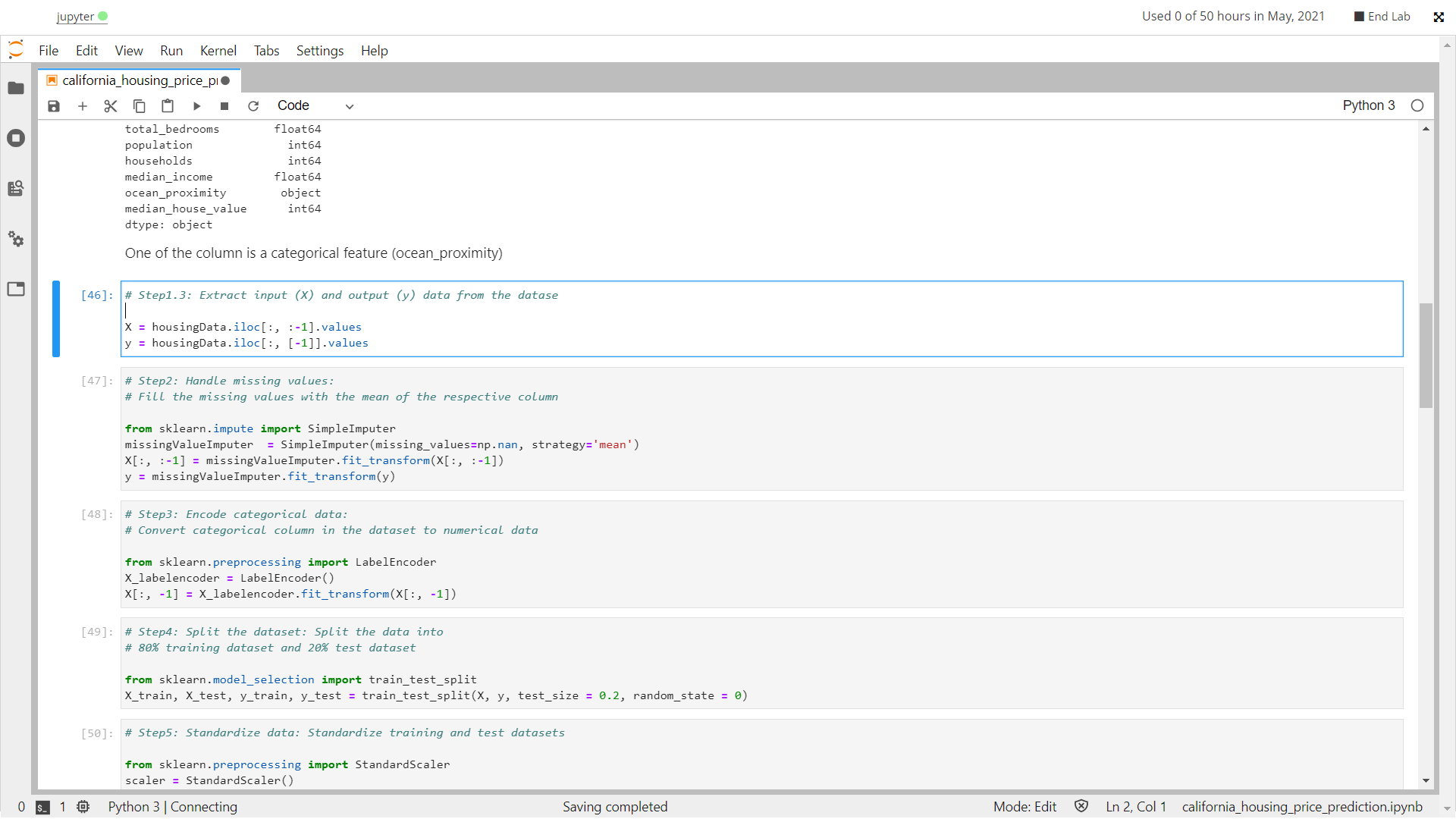
# Step6: Standardize data: Standardize training and test datasets from sklearn.preprocessing import StandardScaler

scaler = StandardScaler() X\_train = scaler.fit\_transform(X\_train)

X\_test = scaler.transform(X\_test)

y\_train = scaler.fit\_transform(y\_train)

y\_test = scaler.transform(y\_test)



Q3. Task1: Perform Linear Regression

# Task1.1: Perform Linear Regression on training data from sklearn.linear\_model import LinearRegression

linearRegression = LinearRegression()

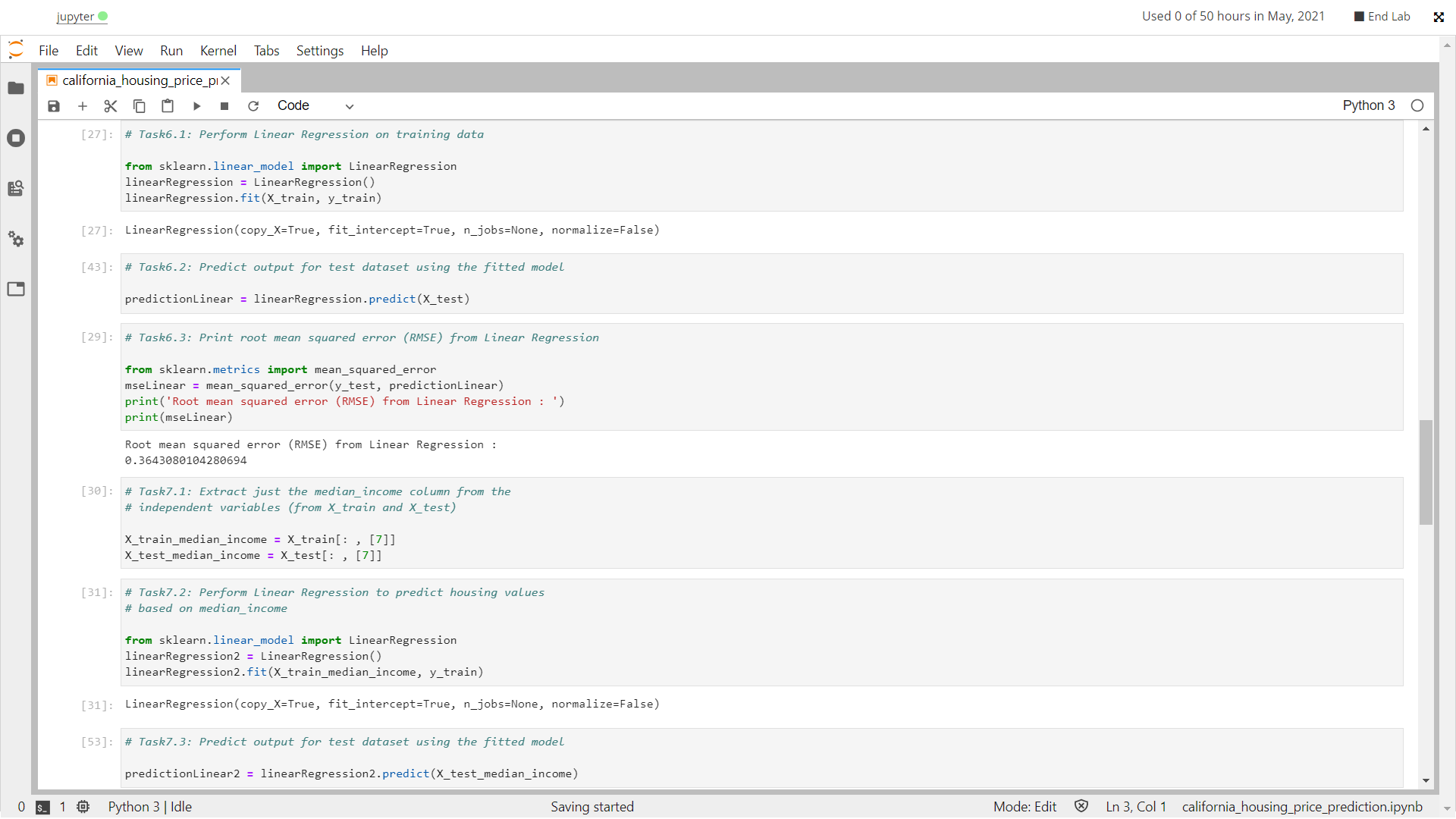
linearRegression.fit(X\_train, y\_train)

# Task1.2: Predict output for test dataset using the fitted model predictionLinear = linearRegression.predict(X\_test)

# Task1.3: Print root mean squared error (RMSE) from Linear Regression from sklearn.metrics

import mean\_squared\_error mseLinear = mean\_squared\_error(y\_test, predictionLinear)

print('Root mean squared error (RMSE) from Linear Regression = ') print(mseLinear)



Bonus exercise: Perform Linear Regression with one independent variable

# Task: Extract just the median\_income column from the independent

# variables (from X\_train and X\_test)

X\_train\_median\_income = X\_train[: , [7]]

X\_test\_median\_income = X\_test[: , [7]]

# Task: Perform Linear Regression to predict housing values

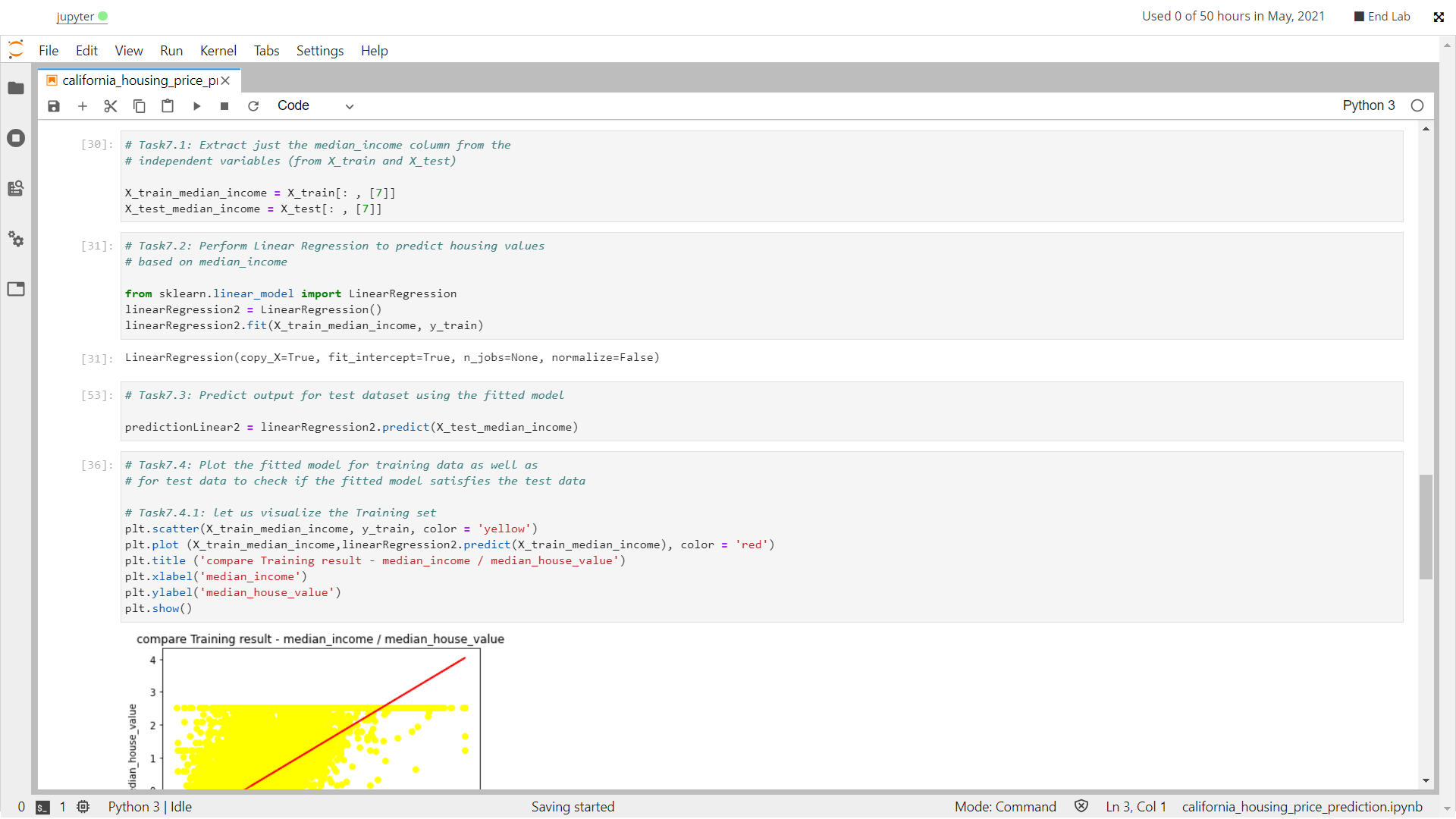
# based on median\_income

from sklearn.linear\_model import LinearRegression

linearRegression2 = LinearRegression() linearRegression2.fit(X\_train\_median\_income, y\_train)

# Task: Predict output for test dataset using the fitted model

predictionLinear2 = linearRegression2.predict(X\_test\_median\_income)



# Task: Plot the fitted model for training data as well as

# for test data to check if the fitted model satisfies the test data

# Task: let us visualize the Training set

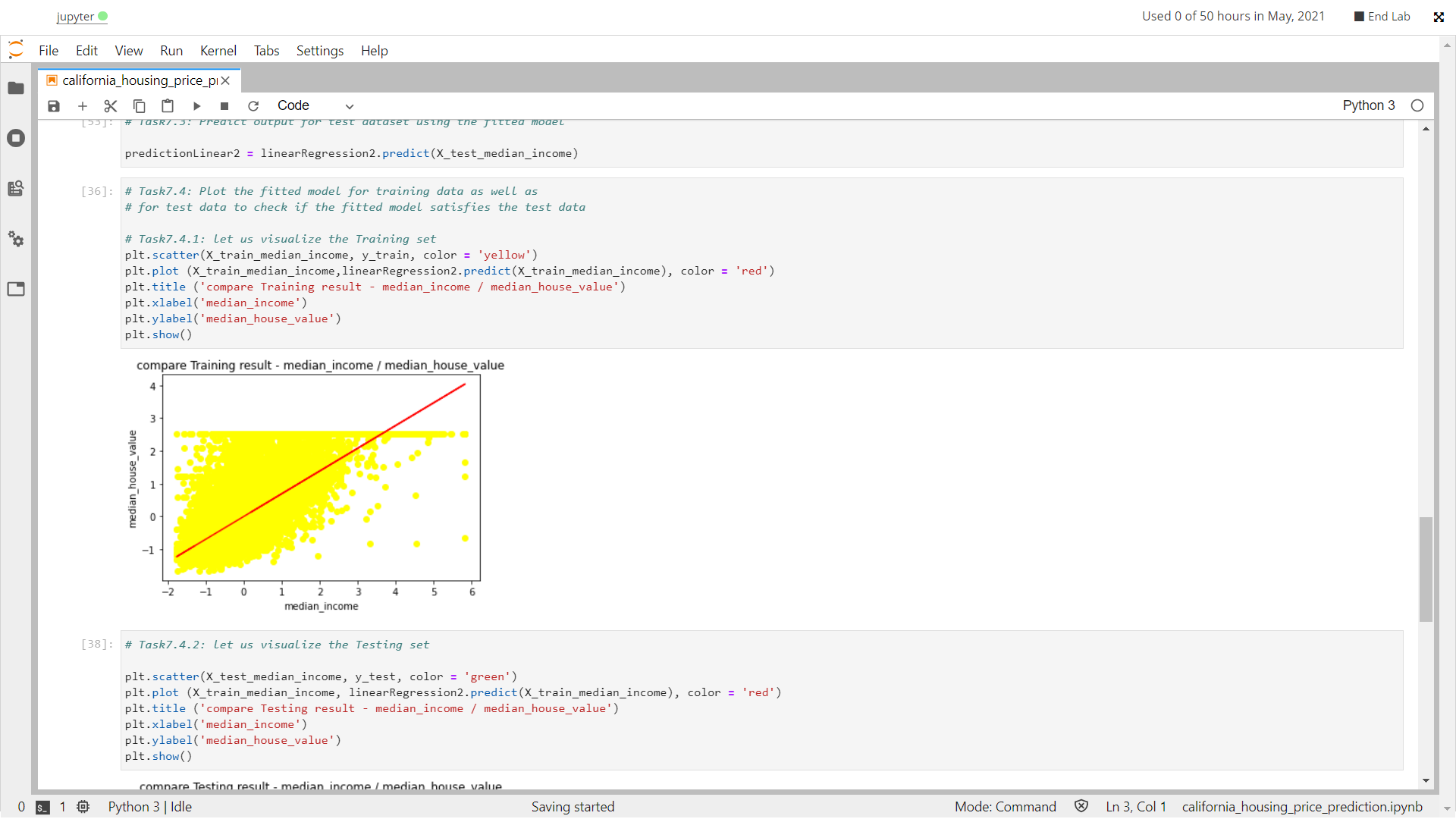
plt.scatter(X\_train\_median\_income, y\_train, color = yellow)

plt.plot (X\_train\_median\_income, linearRegression2.predict(X\_train\_median\_income), color = 'red')

plt.title ('compare Training result - median\_income / median\_house\_value') plt.xlabel('median\_income')

plt.ylabel('median\_house\_value')

plt.show()



# Task: let us visualize the Testing set

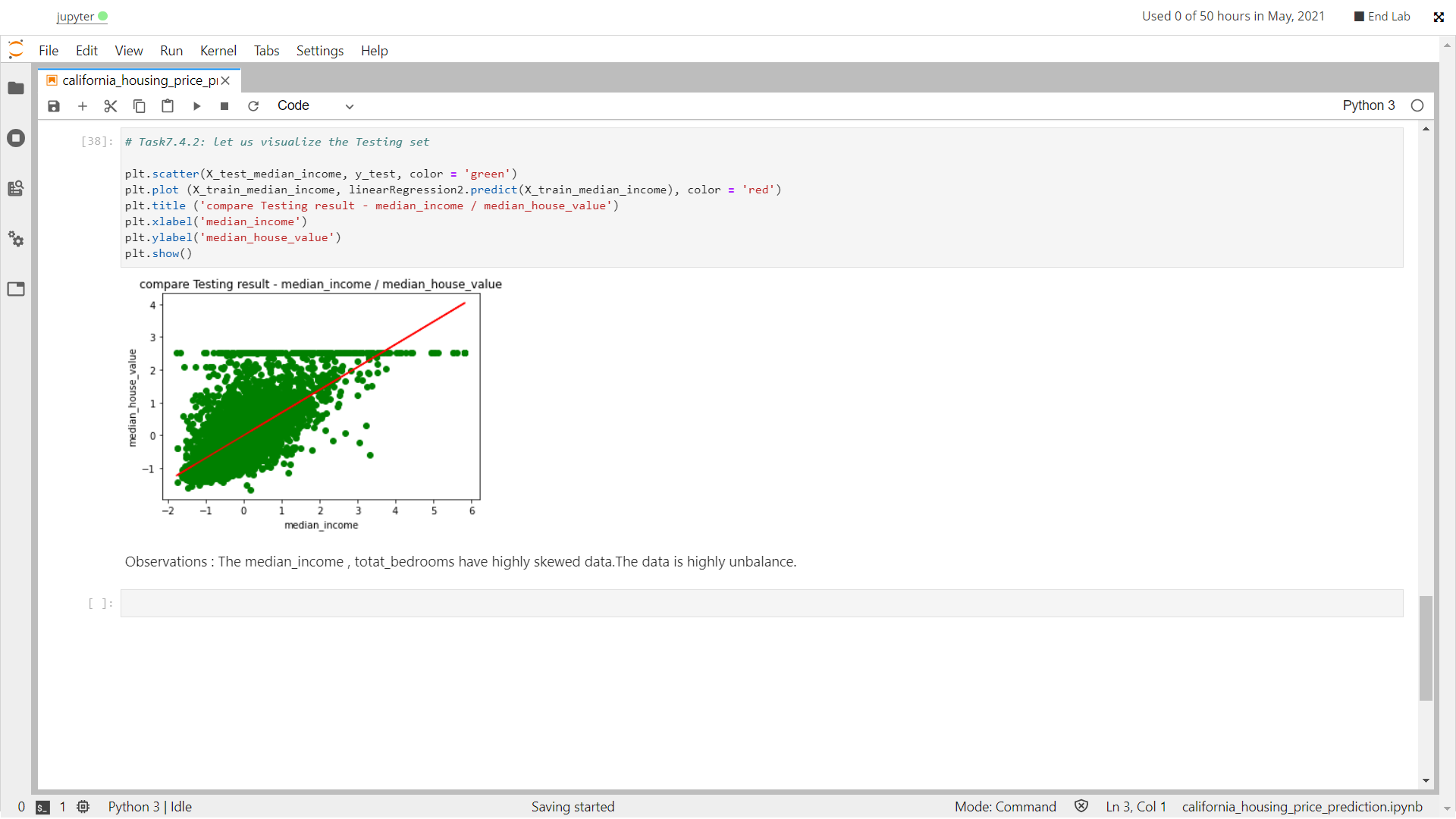
plt.scatter(X\_test\_median\_income, y\_test, color = green)

plt.plot (X\_train\_median\_income, linearRegression2.predict(X\_train\_median\_income), color = 'red')

plt.title ('compare Testing result - median\_income / median\_house\_value') plt.xlabel('median\_income')

plt.ylabel('median\_house\_value')

plt.show()



**Observations** : The median\_income , totat\_bedrooms have highly skewed data.The data is highly unbalance.

------------------------------------END----------------------------