**House Price Prediction**

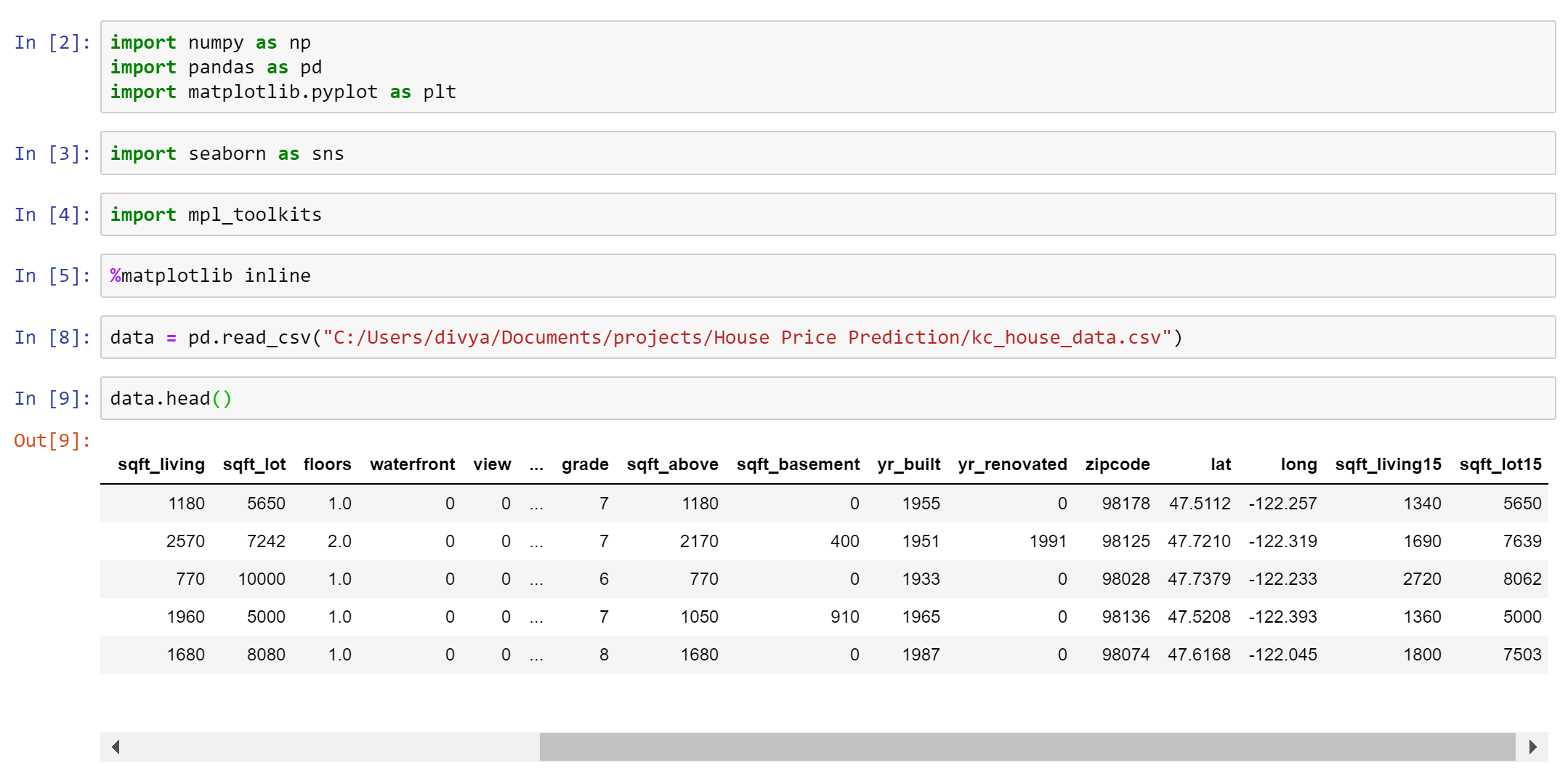
In a Nutshell:

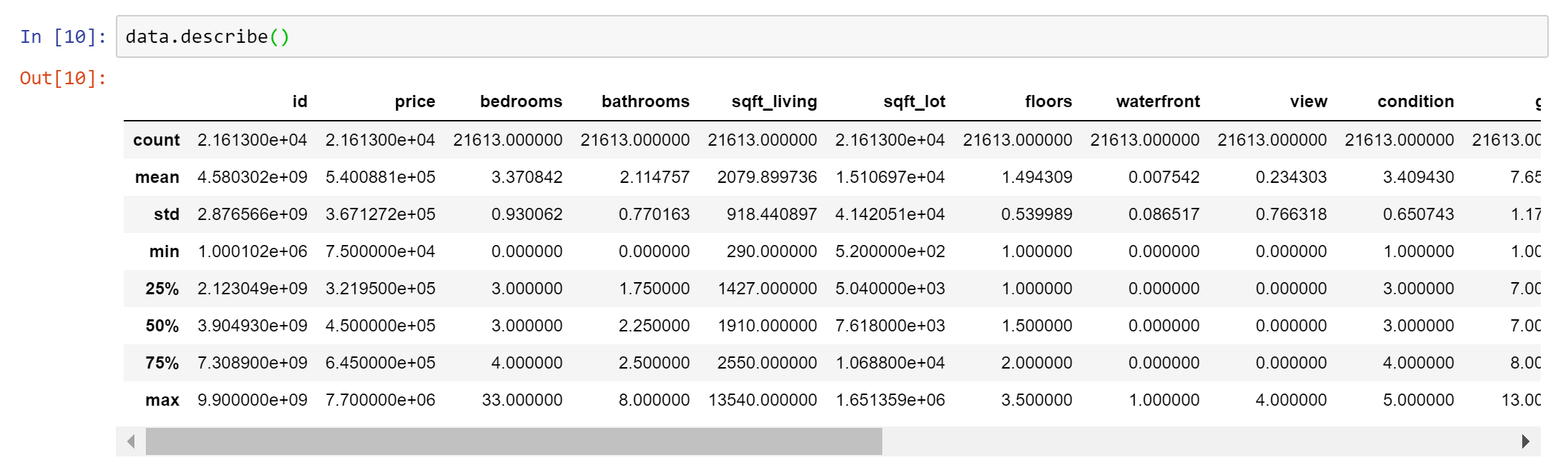
Analyzed the various features affecting the price of a house such as, No. of Bedrooms, Location, No. of floors, Square feet, etc. and plotted them using the matplotlib library. Used the Linear Regression technique, and predicted the price based on various features mentioned above, and obtained an accuracy of 73%. For improved accuracy, used the Gradient Boosting Regression algorithm and obtained an accuracy of 91%.

**Complete Code Explanation:**

**1. Analyze the dataset**

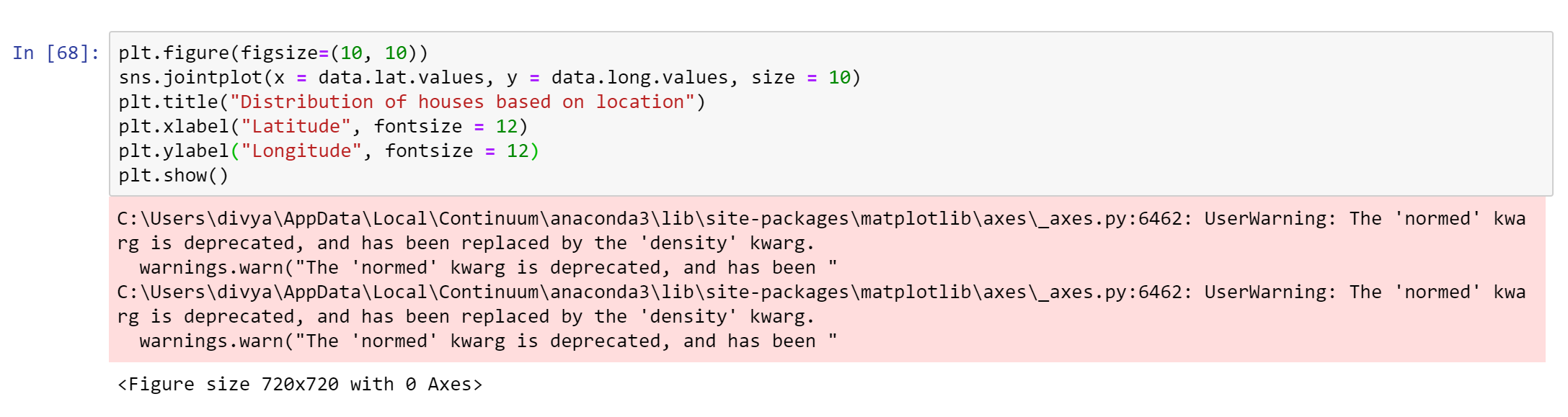
We import our libraries, and load our dataset using pandas.

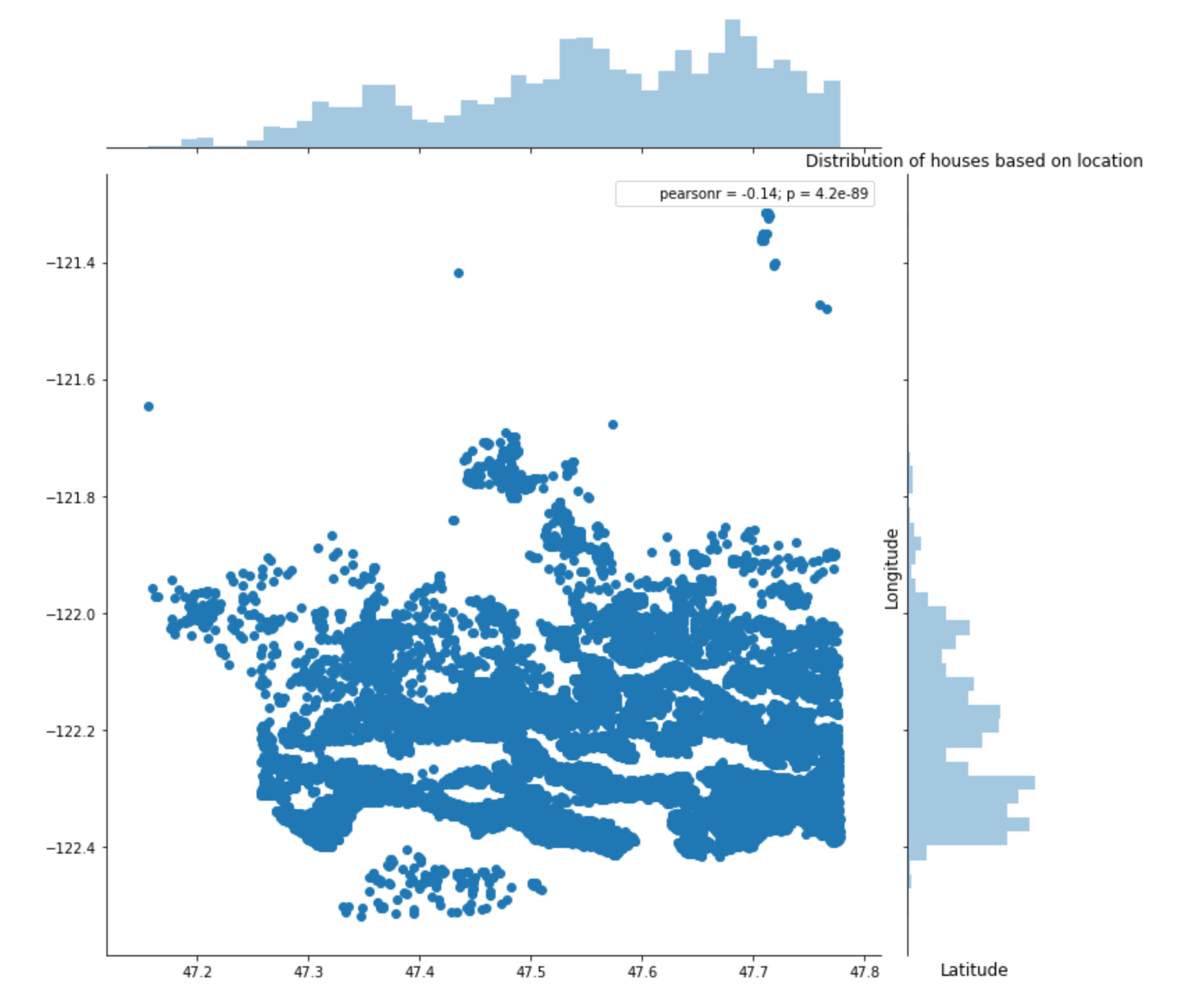


We then examine our dataset

From this, we can obtain information like the maximum no. of bedrooms, minimum no. of bedrooms, the maximum and minimum price of houses, etc.

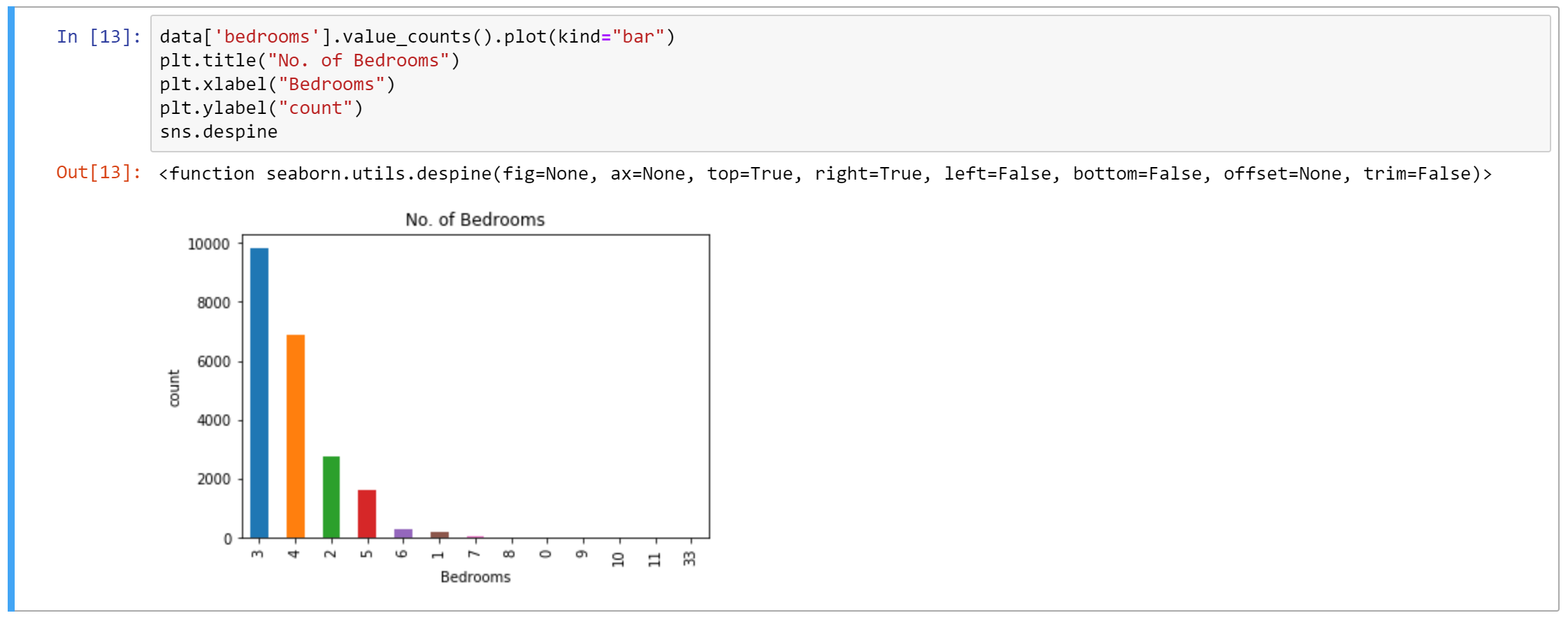
We now see the distribution of houses based on Latitude and Longitude to find out which area has the highest no. of houses, and which has the least.





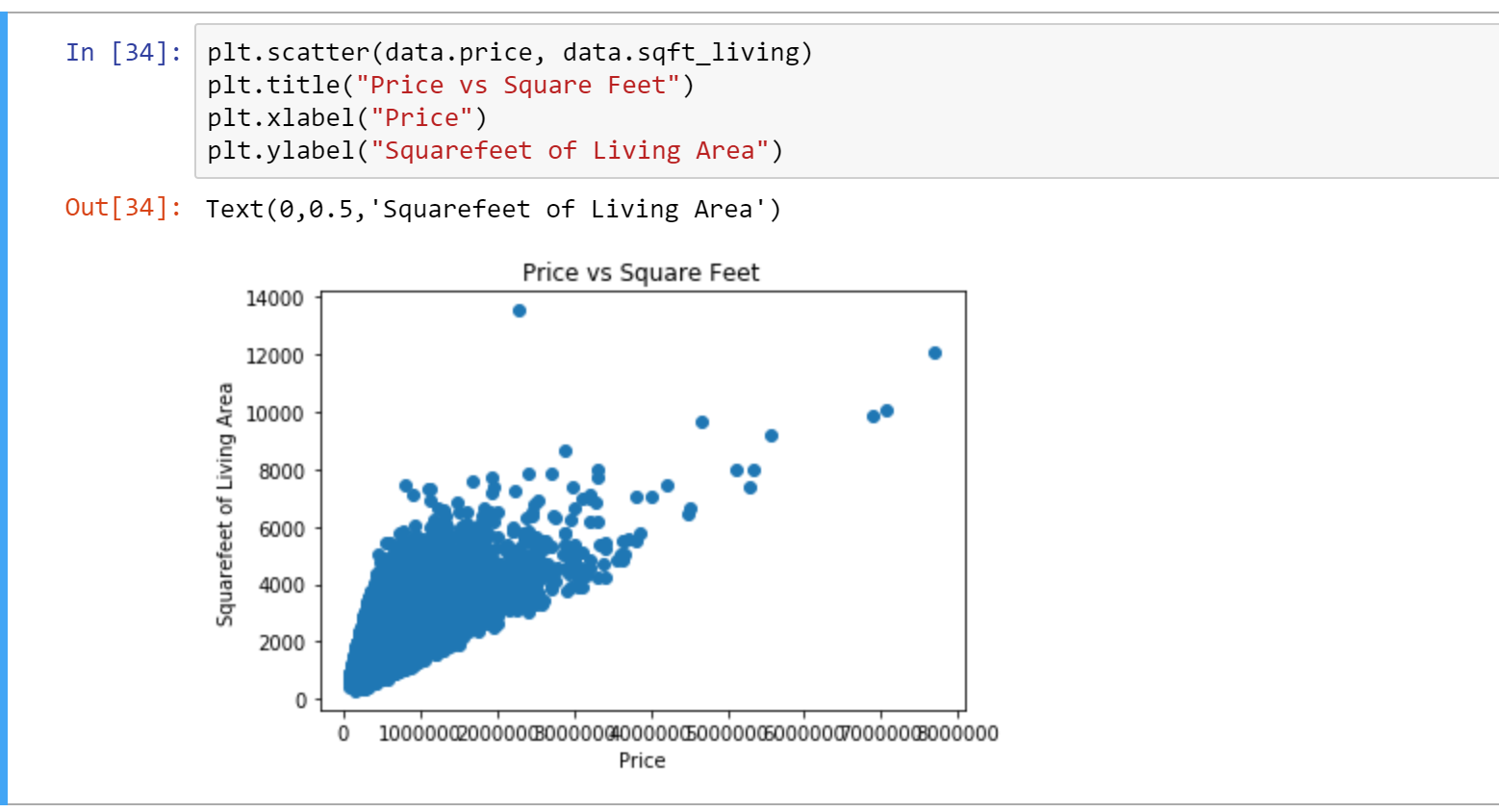
We can observe that there are maximum houses at around latitude 47.7 and between longitudes -122.2 and -122.4

We can also know the total count of 2-bedroom houses, total count of 3-bedroom houses, etc.

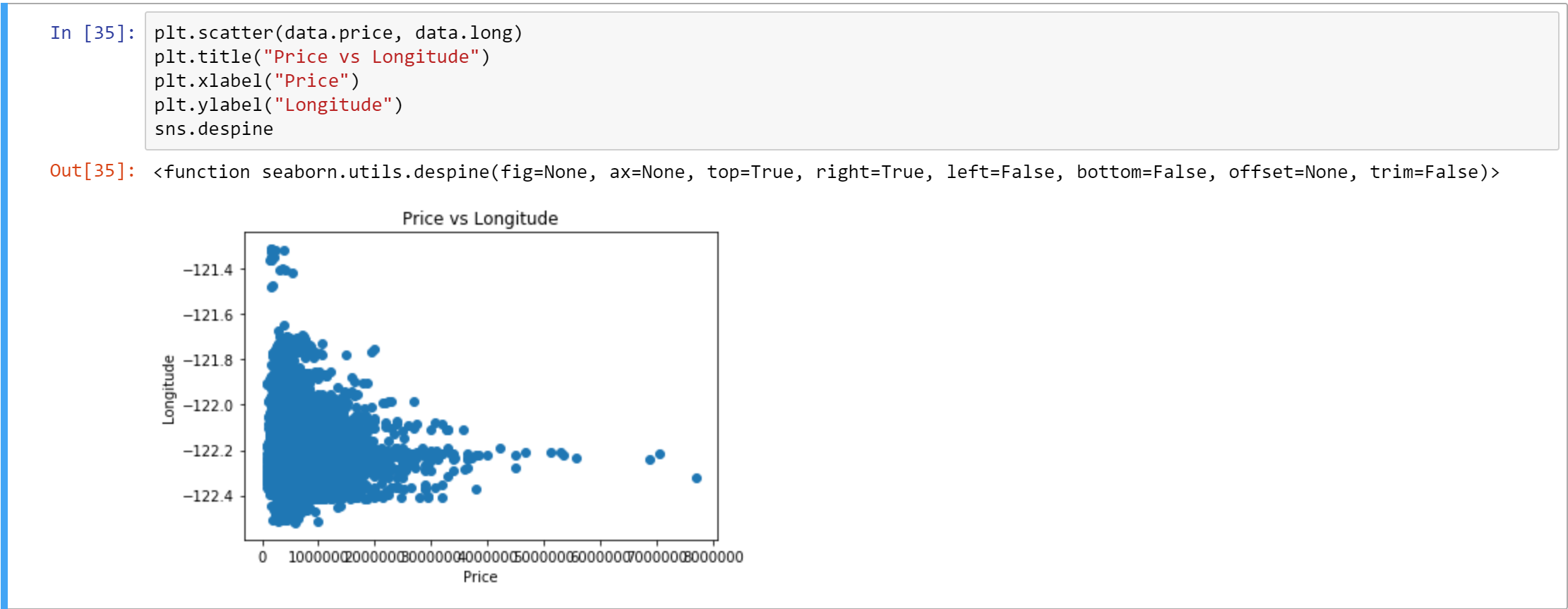


We then analyze how various features affect the Price using scatter plots.

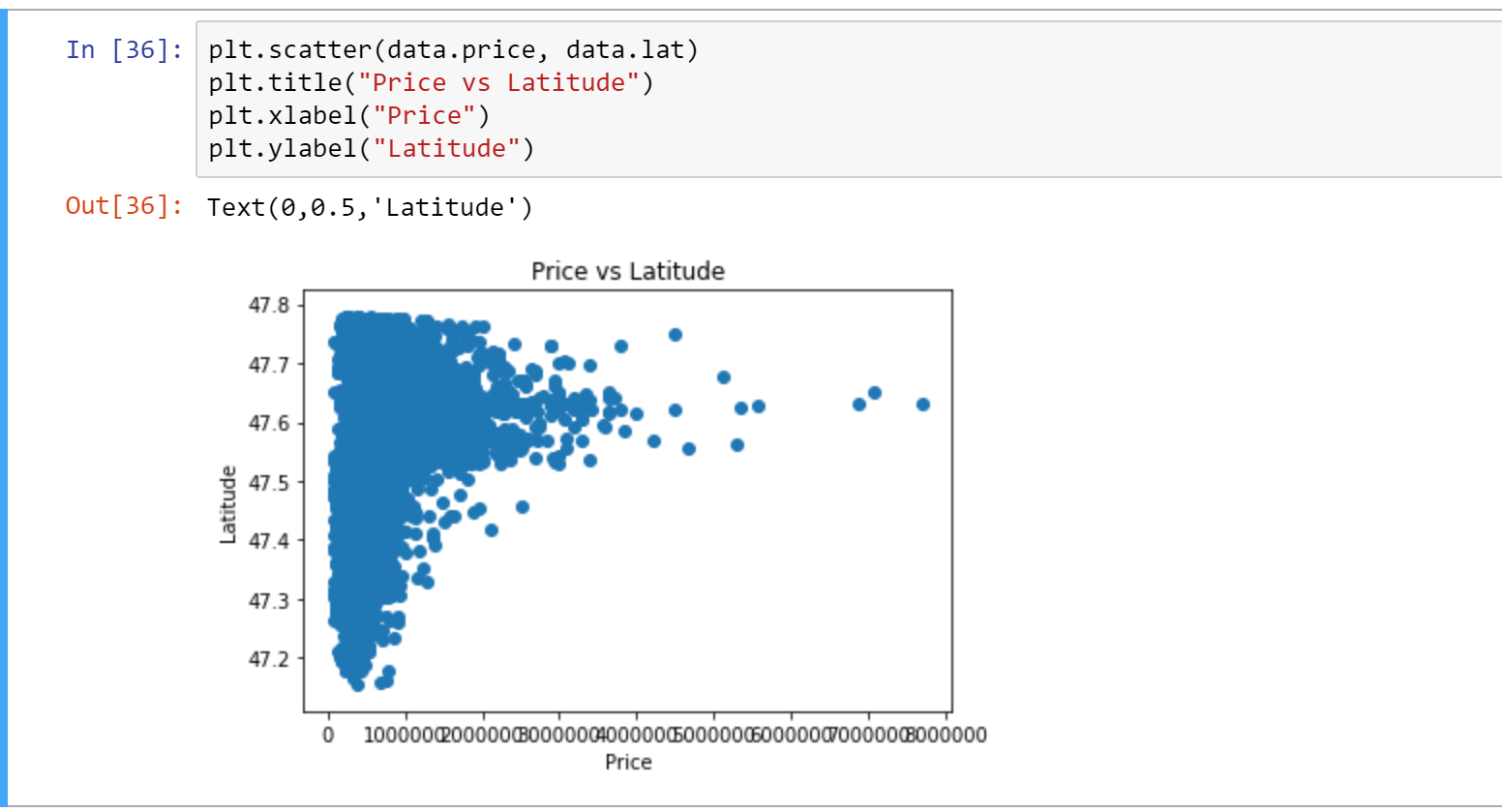
1. Square feet of Living Area



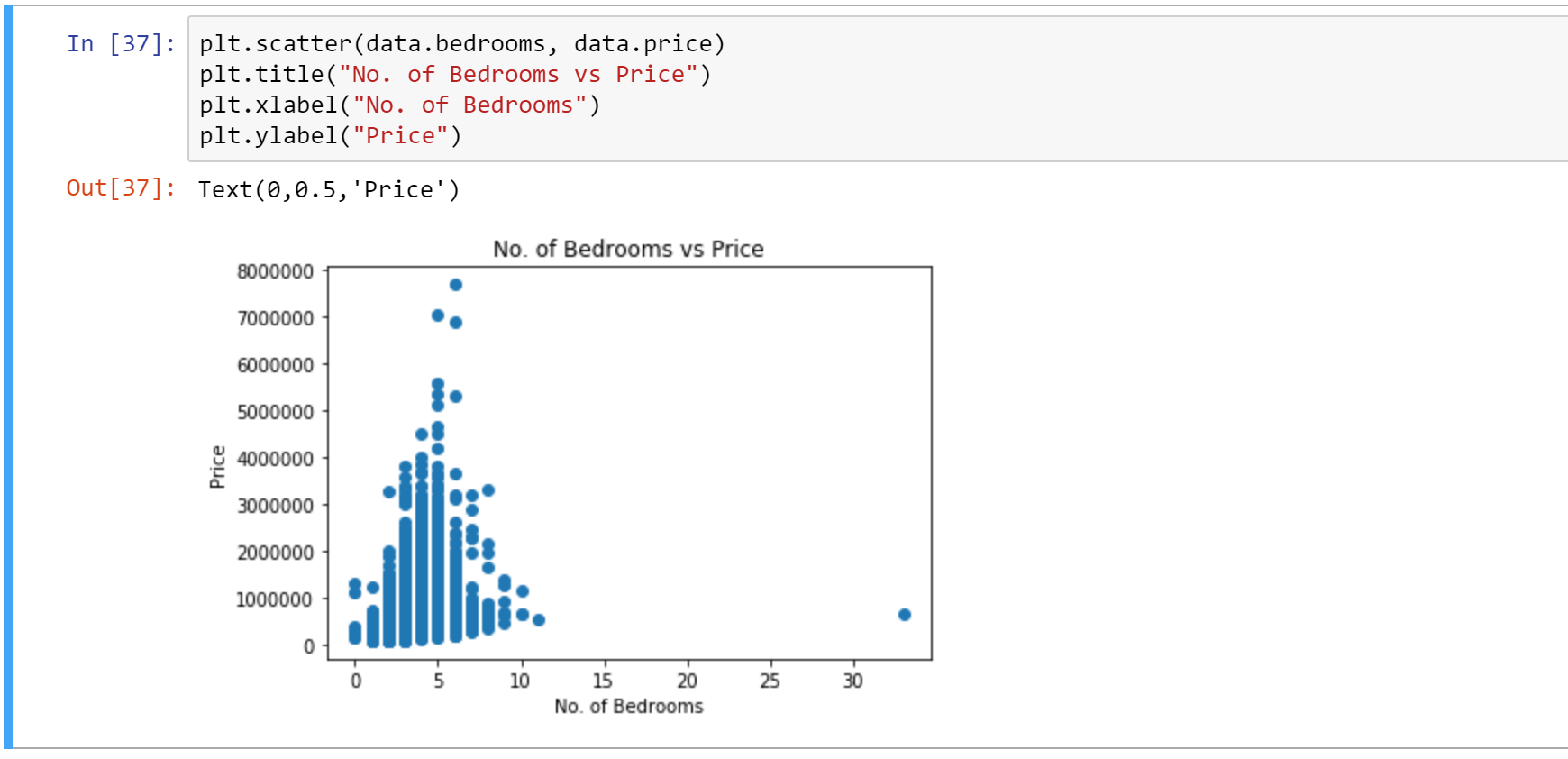
1. Longitude



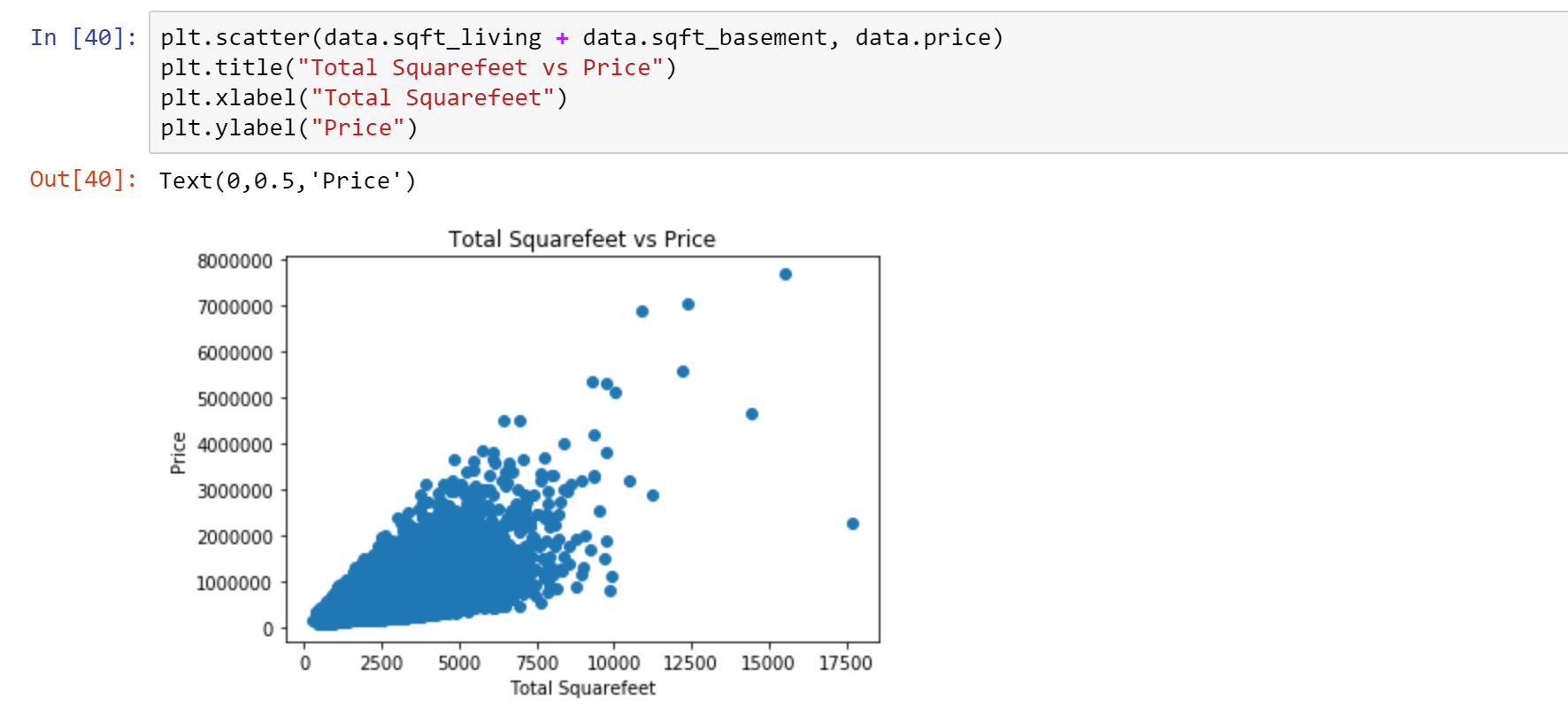
1. Latitude



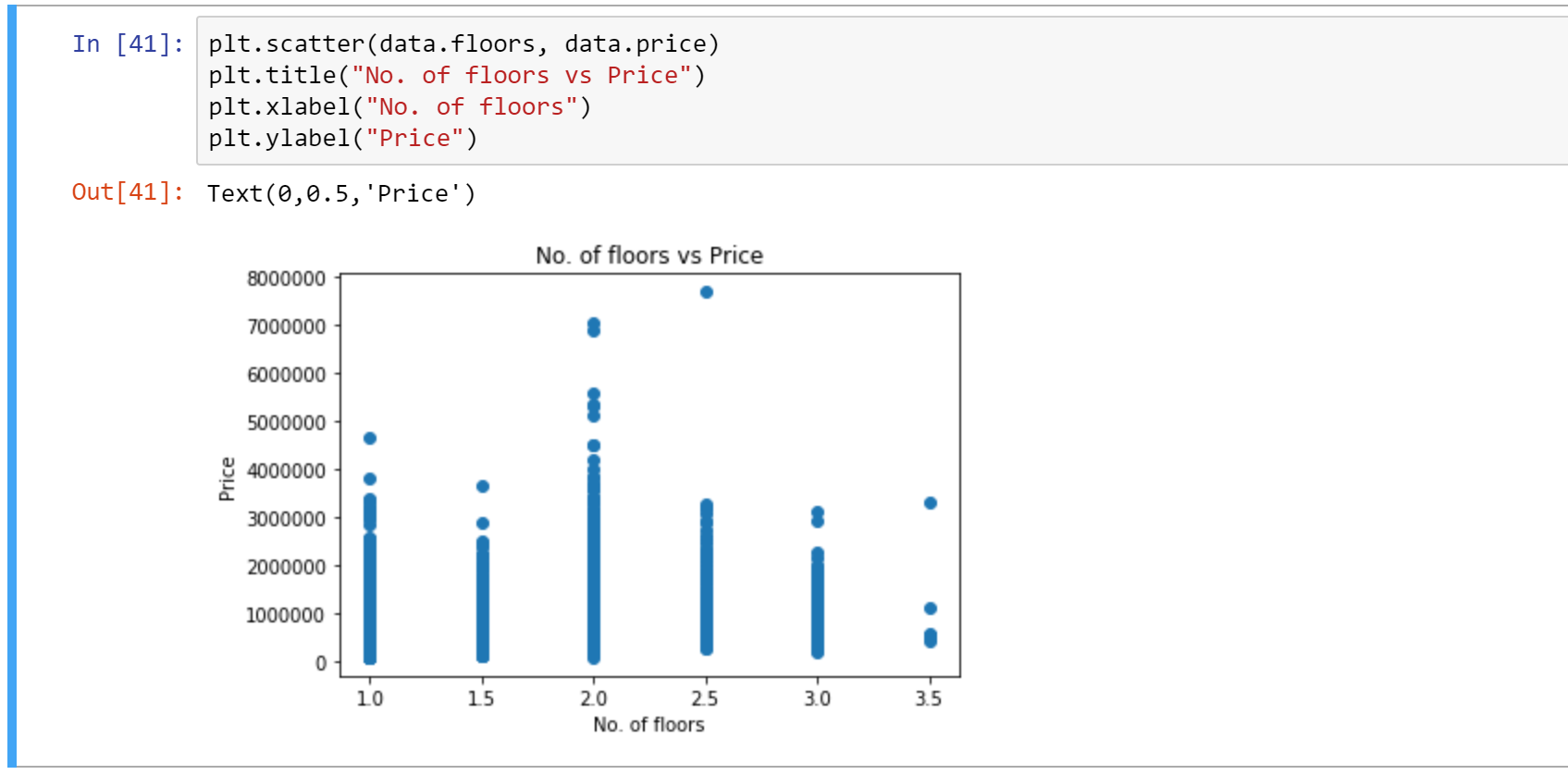
1. No. of Bedrooms



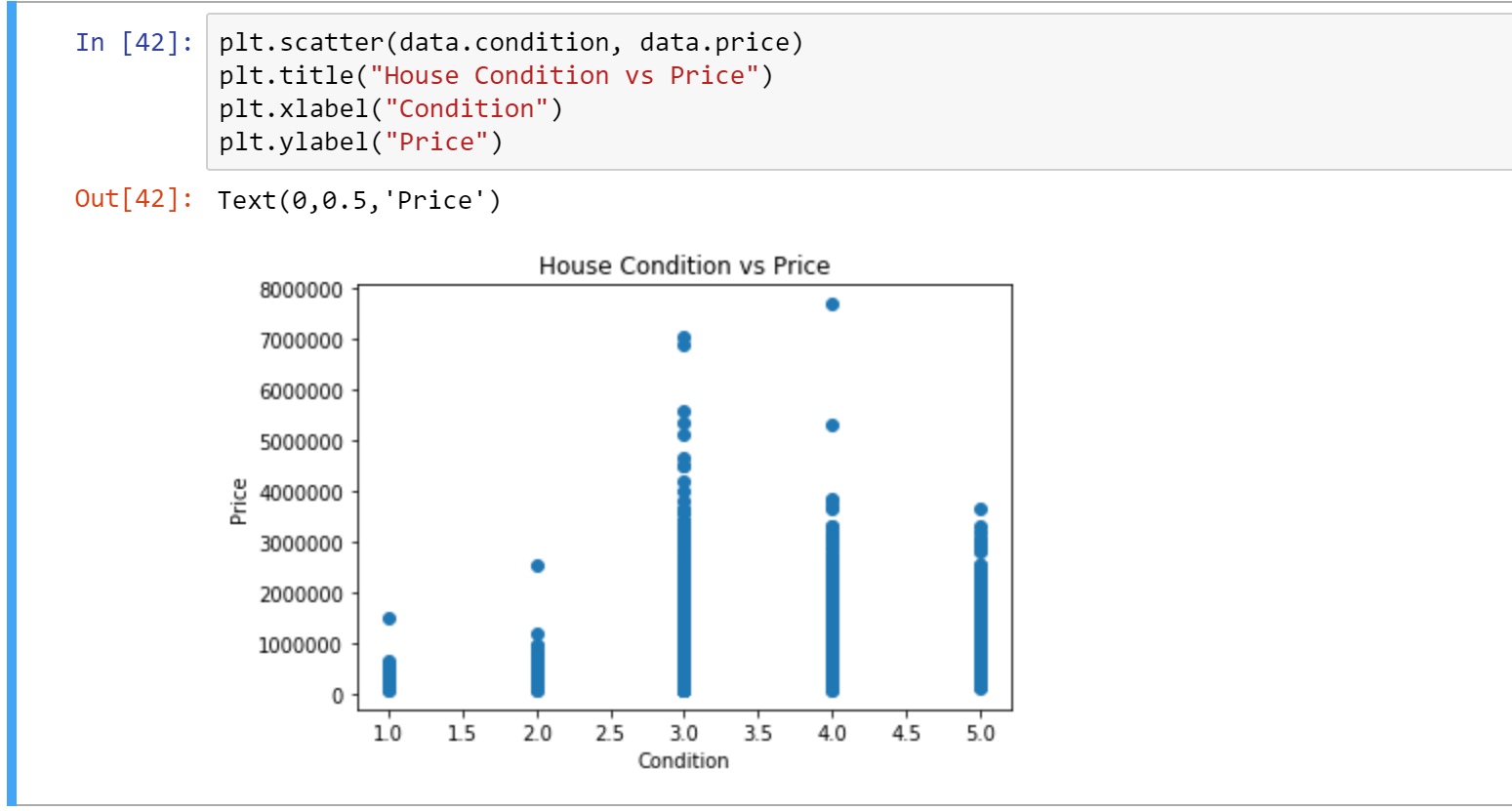
1. Total Square feet of the house



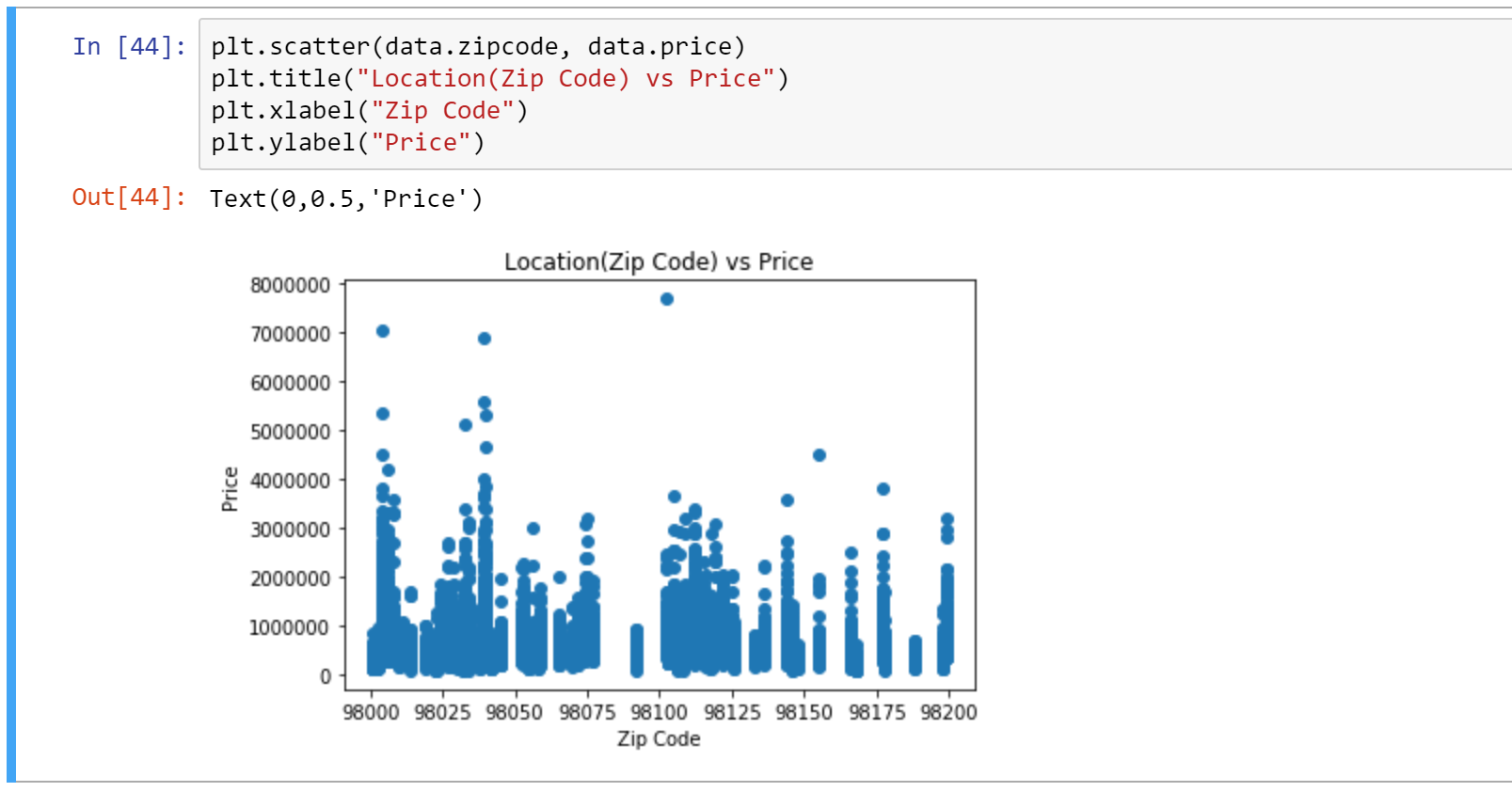
1. No. of Floors



1. Condition of the House



1. Location based on Zip Code

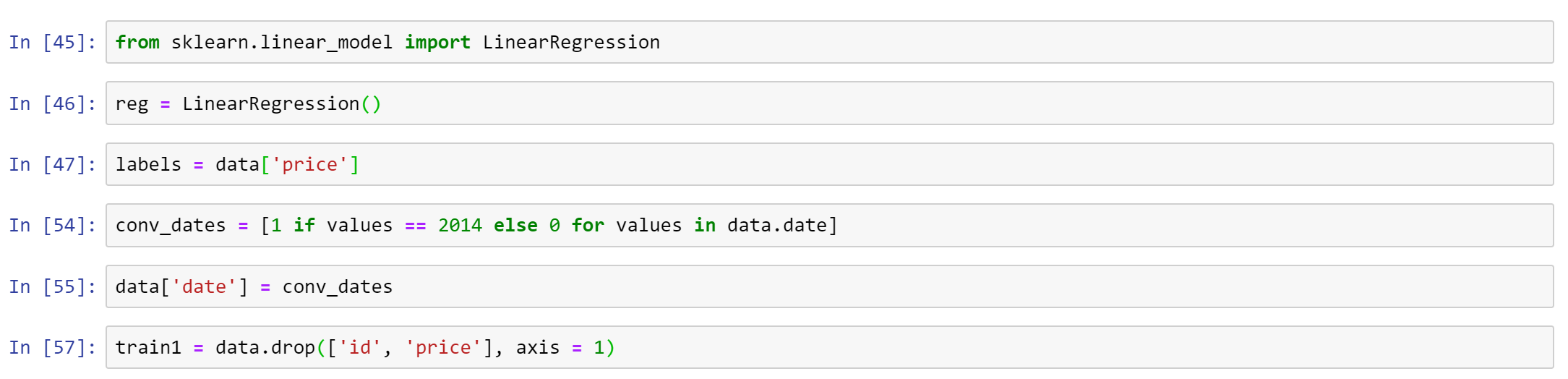


We can go on playing around with our data and analyze various trends.

**2. Predicting the Price using Linear Regression:**

Linear Regression:

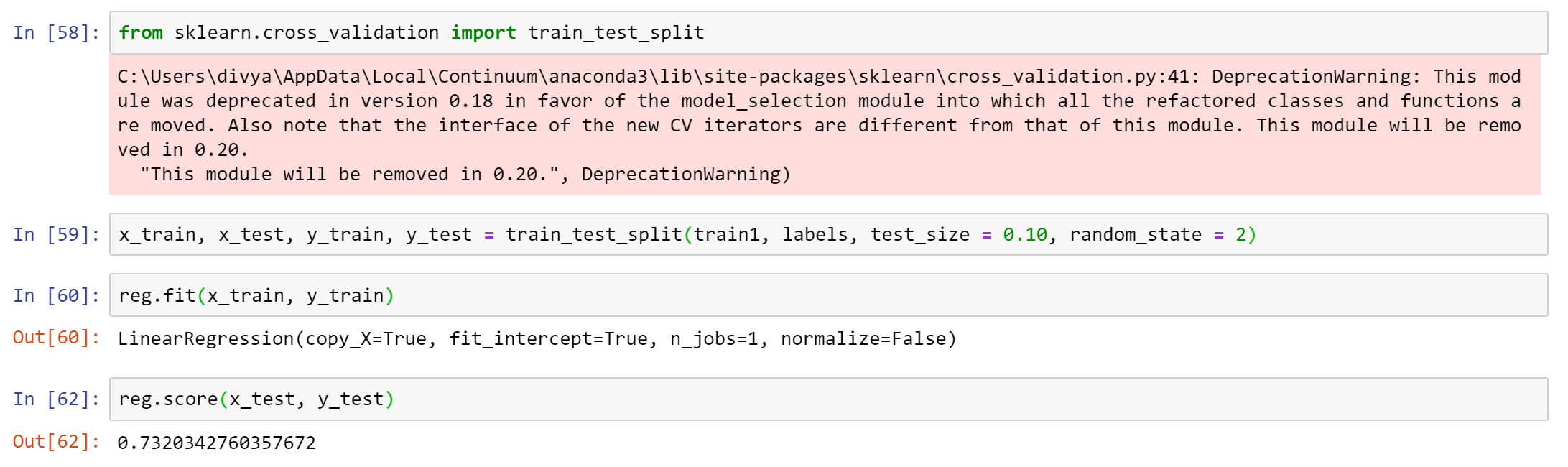
Linear Regression works on the line equation: y = mx + c. The variable we are predicting is called the criterion variable and is referred to as Y. The variable we are basing our predictions on is called the predictor variable and is referred to as X. When there is only one predictor variable, the prediction method is called **Simple Regression. If multiple predictor variables are present, then multiple regression.**



We import the Linear Regression from sklearn (built-in library) and initialize it to a variable reg.

Since prices are to be predicted, we set labels(output) as price columns.

We convert dates to 0’s and 1’s. 0 for new houses (that are built after 2014).



We now use cross validation and import another dependency train\_test\_split to split our data into training and test data. We have used 90% for training and 10% of data for testing.

We have obtained an accuracy of 73% which is less for our problem.

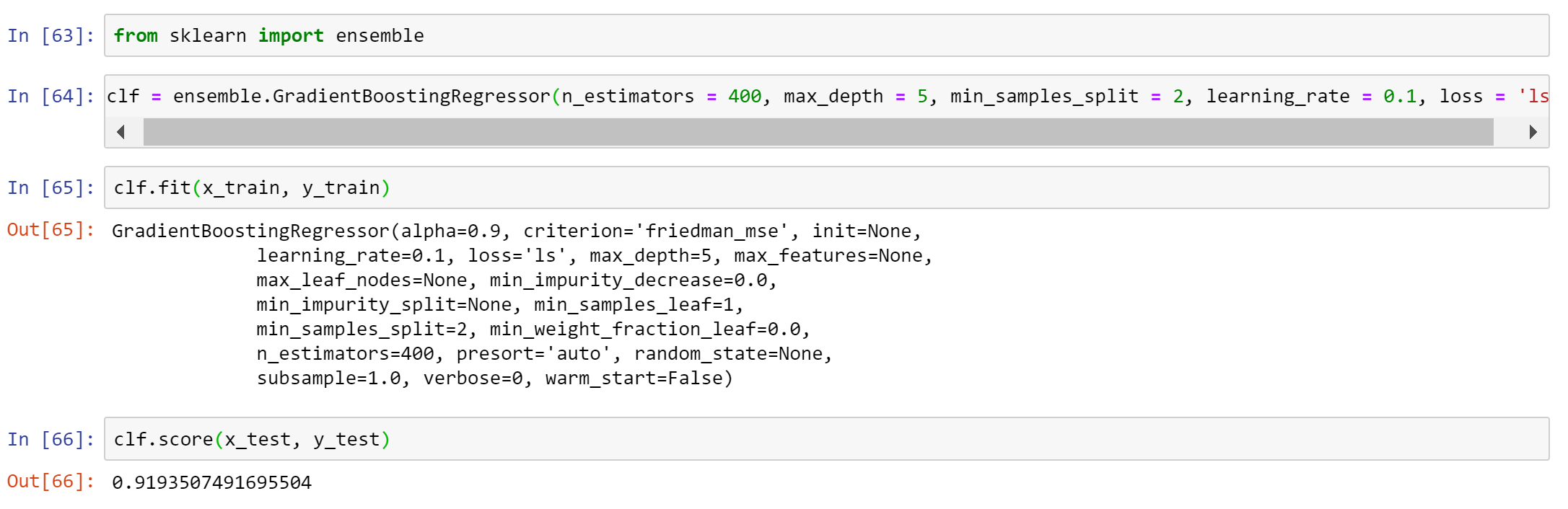
The reasons for lesser accuracy might be some lost data or the slow learning rate of the machine.

**3. Gradient Boosting Regression:**

Gradient Boosting:

Gradient boosting is a [machine learning](https://en.wikipedia.org/wiki/Machine_learning) technique for [regression](https://en.wikipedia.org/wiki/Regression_(machine_learning)) and [classification](https://en.wikipedia.org/wiki/Classification_(machine_learning)) problems, which produces a prediction model in the form of an [ensemble](https://en.wikipedia.org/wiki/Ensemble_learning) of weak prediction models, typically [decision trees](https://en.wikipedia.org/wiki/Decision_tree_learning). It builds the model in a stage-wise fashion like other [boosting](https://en.wikipedia.org/wiki/Boosting_(machine_learning)) methods do, and it generalizes them by allowing optimization of an arbitrary [differentiable](https://en.wikipedia.org/wiki/Differentiable_function) [loss function](https://en.wikipedia.org/wiki/Loss_function).

For improved accuracy, we use Gradient Boosting Regressor.



The various parameters are:

n\_estimator — The number of boosting stages to perform. We should not set it too high which would overfit our model.

max\_depth — The depth of the tree node.

learning\_rate — Rate of learning the data.

loss — loss function to be optimized. ‘ls’ refers to least squares regression

minimum sample split — Number of sample to be split for learning the data

On fitting the model with the same training and test data, we obtain an accuracy of 91%, which is good!