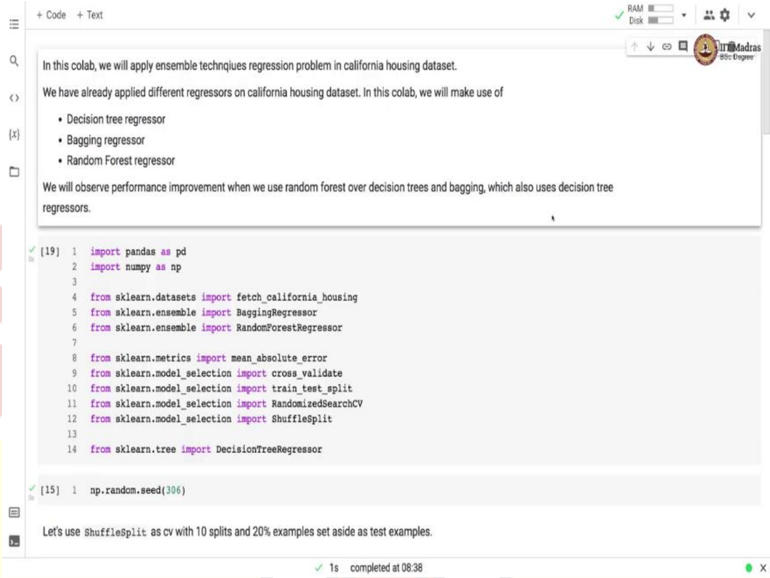
IIT Madras

ONLINE DEGREE

Machine Learning Practice
Professor. Ashish Tendulkar
Indian Institute of Technology, Madras
Bagging and Random Forest Regressor on California Housing Dataset

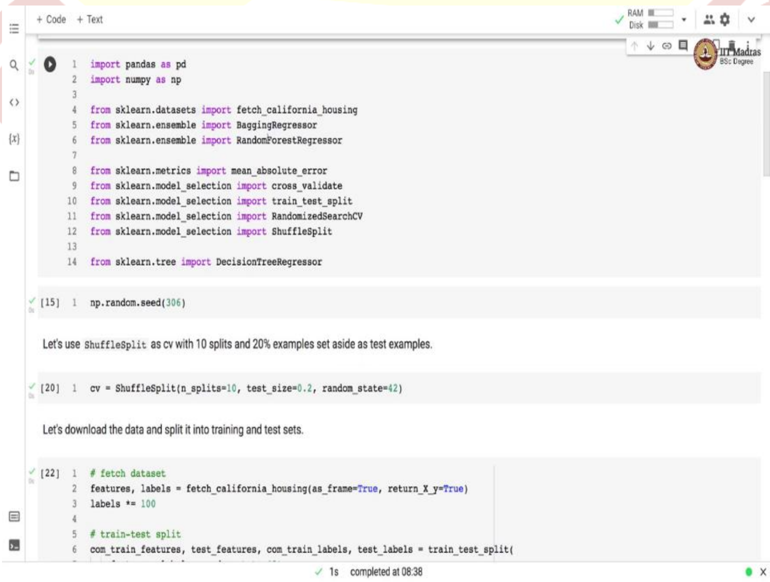
(Refer Slide Time: 0:10)



Namaste! Welcome to the next video of Machine Learning Practice Course. In this video, we will apply ensemble technique to a regression problem in California Housing Dataset. We are already applied different regressors on California housing dataset in this course. In this collab, we will make use of decision tree regressor, bagging regressor and random forest regressor.

We will observe performance improvement when we use random forest over decision tree and bagging and bagging also uses decision tree regressor internally.

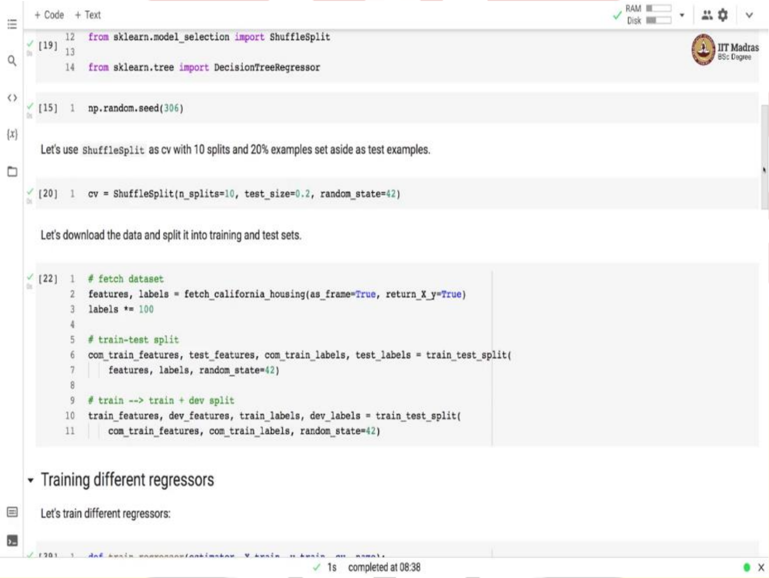
(Refer Slide Time: 0:51)



We begin by importing necessary libraries and packages we import Pandas and NumPy for fetching California housing dataset we import fetch _California _housing from sklearn.datasets module, we import BaggingRegressor, RandomForestRegressor and DecisionTreeRegressor.

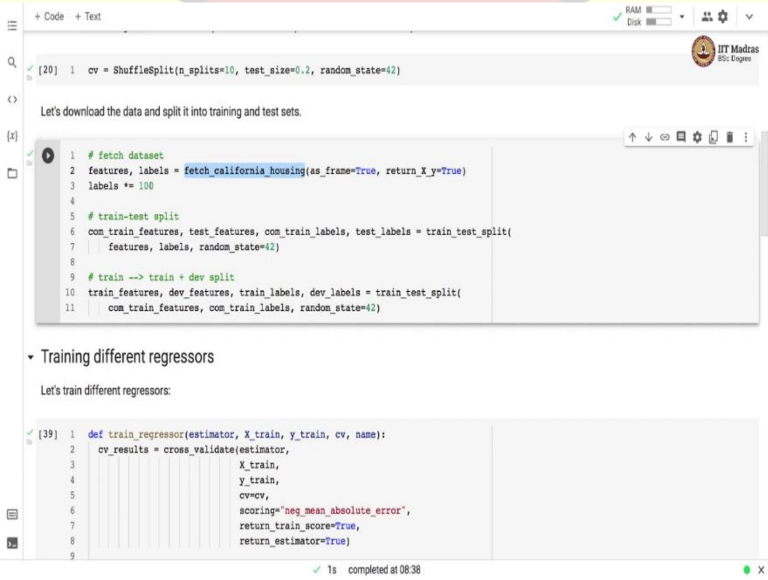
Then we use mean _absolute _error as a metric. And for model selection we are going to use cross _validate, train _test _split, RandomizedSearchCV and ShuffleSplit.

(Refer Slide Time: 01:28)



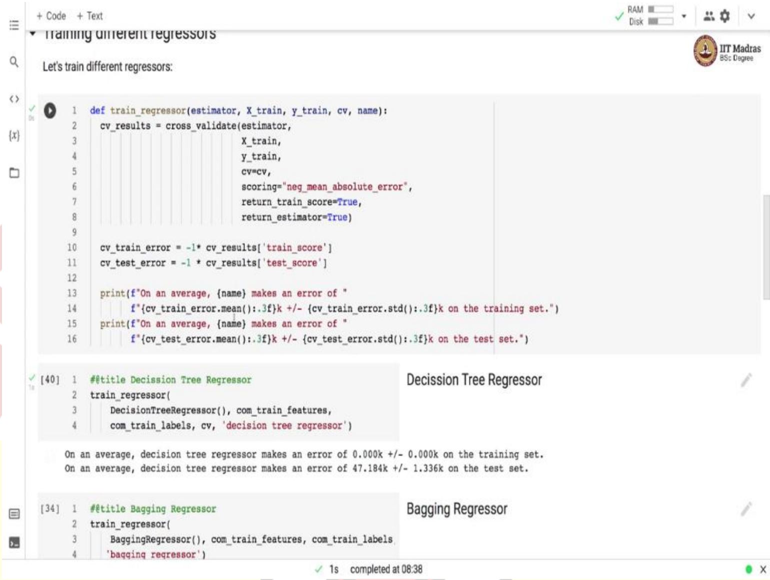
We initialize the random seed to 306. And we are going to use ShuffleSplit as cross-validation with 10 splits and 20% example set aside as test examples.

(Refer Slide Time: 01:42)



Let us download the data and split it into training and test set. So, we downloaded the data with fetch _california _housing, we × label by 100. So, that we get values in 1000 of dollars, then we perform training and test split and we also perform training and their split.

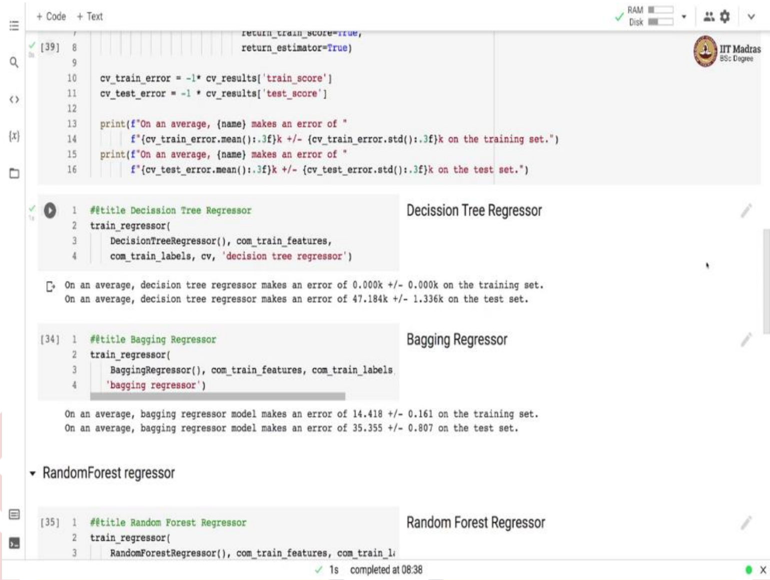
(Refer Slide Time: 02:06)



In order to train different regressors, we define a train _regressor function that performs training of the regressor with cross-validation, it takes estimator as an argument along with training feature matrix, training label vector, the cross-validation, the scoring which is negative mean absolute error, and we said return train _score and return estimator both these flags to true.

Then we get the training error by × the train _score by -1 and test _error by × test _score by -1. Remember the score here is negative mean absolute error and hence we × it by -1. Then we print the error on training and test sets.

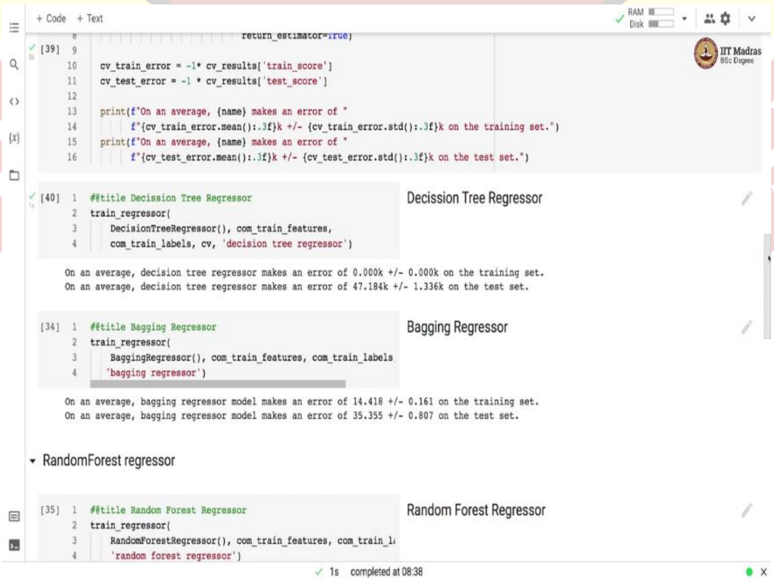
(Refer Slide Time: 02:58)



We are going to train 3 regressors, Decision Tree Regressor, Bagging Regressor and Random Forest Regressor. After defining this function, this is quite a straightforward task we call the function train _regressor by passing the appropriate estimator which is DecisionTreeRegressor here the training feature matrix and training label vector along with the cross-validation scheme and the name of the regressor so that we can print it appropriately in the output.

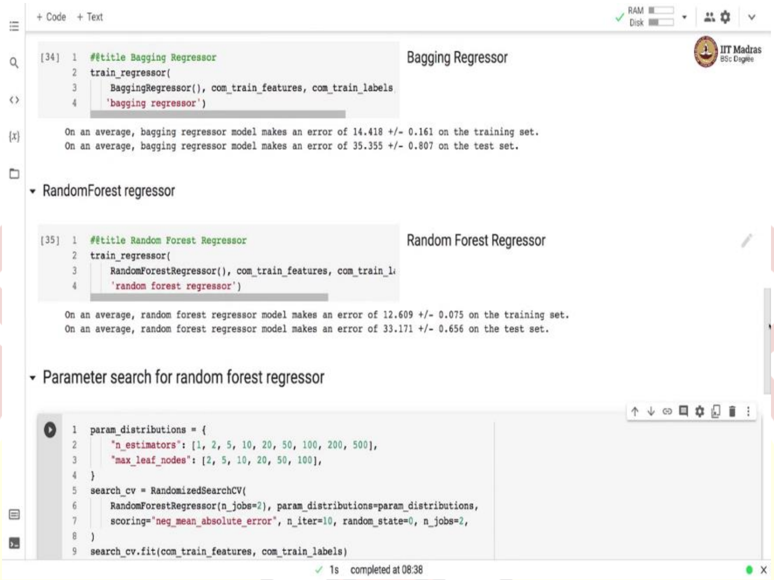
So, you can see here that the DecisionTreeRegressor makes 0 error on the training set, whereas it makes error of 47.18k on the test set with standard deviation of 1.33k.

(Refer Slide Time: 03:45)



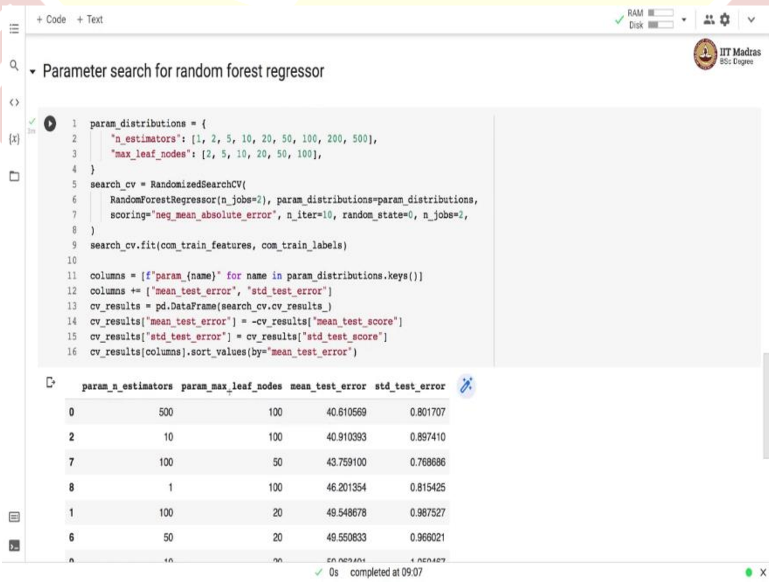
So, you can see that this is an example of over fitted model because it obtains a 0 error on the training, but quite high error on the test set. And this precise problem of over fitting is solved with bagging and random forest. Then we trained BaggingRegressor and when we train the BaggingRegressor, model makes error 14.41 on the training set, but the error on the test set is reduced from 47.18 to 35.35.

(Refer Slide Time: 04:20)



And when we apply a RandomForestRegressor the errors get reduced both on the training set as well as on the test set. Now on the training set we have error of 12.609 whereas on the test set we have error of 33.17. So, you can see that as we use BaggingRegressor and RandomForestRegressor the error has gone down on the test set with marginal increase in the training error. And this is the exact point that we were talking in the theory class that bagging helps us to reduce the variance or over fitting in the base classifiers.

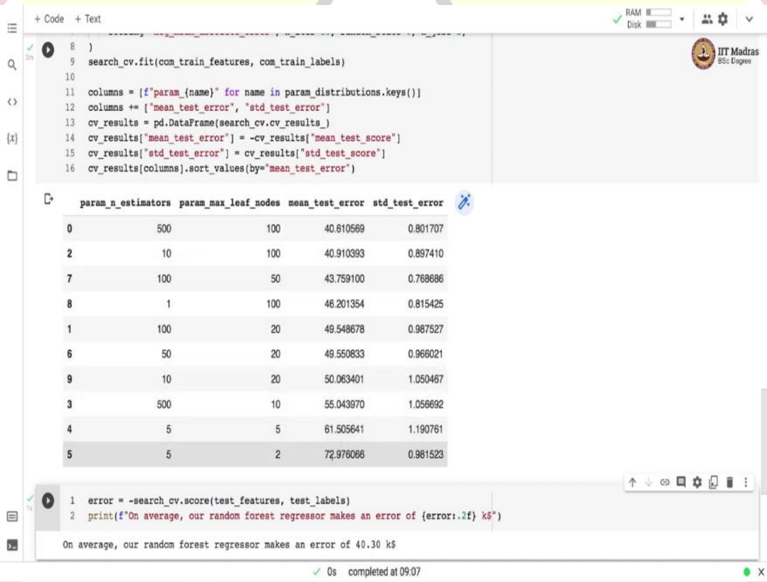
(Refer Slide Time: 05:05)



We demonstrate how to perform parameter search for random forest regressor. There are 2 configurable parameters, number of estimators and maximum number of leaf nodes. We define a parameter distribution for number of estimators with number of estimators to be 1, 2, 5, 10, 20, 50, 100, 200 and 500. We said the maximum number of leaf nodes to 2, 5, 10, 20, 50 and 100. We perform RandomizedSearchCV with this parameter distribution, and RandomForestRegressor estimator.

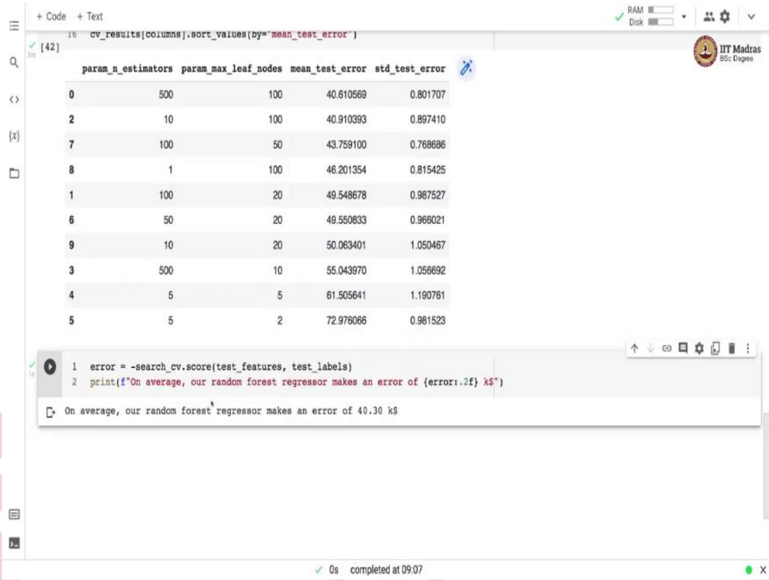
We use negative mean absolute error as a scoring function. And we perform 10 iterations of RandomizedSearchCV . After performing the RandomizedSearchCV , we have displayed the mean _test _error and the standard deviation in the test _error for different parameter configurations.

(Refer Slide Time: 06:09)



And we are assured this parameter configuration in the ascending order of the test _error. So, the best performance was obtained with number of estimators =500 and maximum number of leaf nodes =100. The maximum test _error was 72.97 and was obtained for number of estimators =5 and maximum number of leaf nodes =2.

(Refer Slide Time: 06:39)



Next, we calculate the error on the test set. So, on an average, a RandomForestRegressor makes an error of 40.30k on the test set, and this RandomForestRegressor is obtained to RandomizedSearchCV. So, in this video, we studied how to apply bagging techniques for regression problem. Specifically, we applied Decision Tree, Bagging Regressor and Random Forest Regressor on California Housing Dataset.

We use DecisionTreeRegressor to understand what is the baseline regressor on the dataset and then use bagging and RandomForestRegressor to show that we are able to reduce the over fitting effect in the DecisionTreeRegressor using the bagging technique. We also demonstrated how to perform randomized search for the best set of parameters for RandomForestRegressor.