


Coursework Coversheet School of Geography FACULTY OF ENVIRONMENT		 UNIVERSITY OF LEEDS	
Student ID	201577162	Word count	2,430
Module title / code	GEOG5917M Big data and Consumer Analytics	Mark Less deduction (state reason) Final Mark	
Assignment title	House price prediction using Machine Learning		
Marker			

To improve your work for next time:
1.
2.
3.

Justification of mark (using specific text from criteria)

Additional comments

Introduction

Every year house prices varies, demanding the implementation of a method to predict future house values. House price predictions can assist an owner in determining the selling price of a home and can also help client in determining the best time to buy a home in terms of the finances that will be needed to do so, hence it is important to do it in a correct manner using efficient and accurate methodologies. This prediction of the house prices can be done with the help of various machine learning techniques, in this report we will focus on tuning and constructing Gradient Boosting Machine model which is an efficient machine learning technique for the Boston House dataset which consists of the housing data for 506 census tracts of Boston from the 1970 census and validate this model using measures of fit chosen as RMSE here. Variation in the house prices can be due to many factors such as the properties of locality, crime rate, number of rooms, accessibility to radial highways etc. We will try and analyze how combinations of these factors affect the house prizes in our Boston House dataset using GBM model that combines the predictions from multiple decision trees to generate the final predictions.

All the weak learners in a gradient boosting machine are decision trees. The Gradient Boosting Machines seeks to build predictive models through back-fittings and non-parametric regressions. Instead of building a single model, the GBM starts by generating an initial model and constantly fits new models through loss function minimization to produce the most precise model. [1] The GBM model often provides a predictive accuracy that cannot be trumped. It has a faster training speed and high efficiency. It also can optimize on different loss functions and provide several hyper parameter tuning options that makes the function fit very flexible. Due to their high flexibility, GBMs can be customized to any data-driven activity.

Methods

1. The Boston Housing dataset consisting of the housing data for 506 census tracts on 14 variables, from the 1970 census, 'medv' being the target variable is described in the following table:

Predictor variables	Variables	Description	Data Type
	crim	per capita crime rate by town	Numerical
	zn	Proportion of residential land zoned for lots over 25,000 sq. ft	Numerical
	indus	Proportion of non-retail business acres per town	Numerical
	chas	Charles River dummy variable (=1 if tract bounds river; 0 otherwise)	Categorical (binary)
	nox	Nitric oxides concentration (parts per 10 million)	Numerical

Predictor variables	rm	average number of rooms per dwelling	Numerical
	age	Proportion of owner-occupied units built prior to 1940	Numerical
	dis	weighted distances to five Boston employment centres	Numerical
	rad	Index of accessibility to radial highways	Numerical
	tax	full-value property-tax rate per USD 10,000	Numerical
	ptratio	pupil-teacher ratio by town	Numerical
	b	$(1000(B - 0.63)^2)$ where B is the proportion of blacks by town	Numerical
	lstat	Percentage of lower status of the population	Numerical
Target variable	medv	Median value of owner-occupied homes in USD 1000's	Numerical

Table 1: Description of BostonHousing data for 506 tracts of Boston from the 1970 census

2. A certain level of preprocessing of the data needs to be done before running any machine learning model on it. First, a random split of the data is done into two subsets using an informed heuristic partition method:

- Train dataset: Used to fit the machine learning model (70% of data)
- Test dataset: Used to evaluate the fit machine learning model (30% of data)

This split is done in order to evaluate the accuracy of the GBM model in the end for completely unfamiliar data which was initially hold out (test dataset). It's important to have the similar distribution of target variable in both subsets for good accuracy.

3. Then the scaling of the data is done to create Z-scores of each column except the target variable, if it's not done there's a danger that the model can be dominated by a large dimension i.e., each of the columns are rescaled to have mean zero and standard deviation of one.
4. Initially all the predictor variables are considered in the GBM model for predicting our target variable 'medv'.
5. Cross validation is then chosen as a resampling method. The idea is to use initial training data to generate multiple mini train-test splits to tune our model. We partition the data into k subsets called folds in a standard k-fold cross validation in order to determine how well the model will generalise i.e. predict using an independent dataset.
6. After tuning is done, the GBM model is trained with the best set of parameters and validated using the test dataset after carrying out the data preprocessing steps.

7. In the end, the model evaluation is done with the help of some accuracy measures to check how accurately the prediction of target variable 'medv' is done. In order to see which of the predictor variables has the most impact on target variable, variable importance is analyzed to see whether we could skip some of the predictor variables while training our GBM model.

Results

The primary idea behind boosting is to sequentially add new models to the ensemble. A new weak, base-learner model is trained with respect to the error of the entire ensemble learned so far at each iteration. A gradient-descent based formulation of boosting methods was derived in order to establish a connection with the statistical framework. [2] This formulation of boosting methods and the corresponding models were called the gradient boosting machines.

The learning mechanism in gradient boosting machines, or simply GBMs, fits new models consecutively in order to offer a more accurate estimate of the response variable. The basic idea behind this algorithm is to build new base-learners that are maximally correlated with the loss function's negative gradient, which is associated with the entire ensemble. In general, the researcher has control over choosing the loss function, with a wide range of loss functions derived so far as well as to design one's own task-specific loss. [3]

GBM also provides the methodological foundation for further gradient boosting model development by providing the necessary justifications for the model hyperparameters. It provides several hyper parameter tuning options in order to configure and control the model training process. In order to precisely predict the housing price 'medv' which is our target variable in 'BostonHousing' data we will now need to vary our tuning parameters, and identify which combination of parameters gives best **measures of fit**. We're measuring the fit of our model using root mean squared error (RMSE) which is the square root of mean of difference between predicted and observed values. So lower the RMSE, the better our model will be. There are four main tuning hyperparameters for GBM :

Tuning hyperparameters	Description
Number of trees	Integer specifying the total number of trees to fit. This is equivalent to the number of iterations and the number of basic functions in the additive expansion. Default is 100.
Maximum depth of each tree	Integer specifying the maximum depth of each tree (i.e., the highest level of variable interactions allowed). A value of 1 implies an additive model, a value of 2 implies a model with up to 2-way interactions, etc. Default is 1.
Minimum observations in node	Integer specifying the minimum number of observations in the terminal nodes of the trees.

Learning Rate	A shrinkage parameter applied to each tree in the expansion. Also known as the learning rate or step-size reduction; 0.001 to 0.1 usually work, but a smaller learning rate typically requires more trees. Default is 0.1.
---------------	--

Table 2 [1]: Description of tuning hyperparameters for GBM

Once the tuning parameters have been defined, the type of resampling needs to be specified to prevent overfitting which here is cross-validation. The following tuning strategy is being followed:

- 1) A standard k-fold cross validation is followed for resampling with k=10 folds.
- 2) GBM model tuning is done by setting the values for hyperparameters as follows in the experiment 1 :
 - a. Learning rate (shrinkage) = 0.01, 0.02.
 - b. Maximum depth of each tree (interaction.depth) = 1,3,5,6,7.
 - c. Minimum observation in node (n.minobsinnode) = 2.
 - d. Number of trees (n.trees) = 0 to 50 by 50.
- 3) It took around 40 seconds for the GBM model tuning to complete having 510 combinations of the parameters defined above.
- 4) The results of GBM model tuning for the best tune parameter combination in terms of measures of fit is as follows :

Learning rate	Interaction Depth	Minimum observation in node	Number of trees	RMSE	R squared	MAE	RMSE-SD	R squared-SD	MAESD
0.01	5	2	2450	3.179	0.884	2.218	0.832	0.067	0.372

Table 3: GBM model tuning experiment 1

- 5) Parameter tuning in the experiment 2 is done in the following manner :
 - a. Learning rate (shrinkage) = 0.01, 0.02.
 - b. Maximum depth of each tree (interaction.depth) = 7, 8, 9.
 - c. Minimum observation in node (n.minobsinnode) = 2.
 - d. Number of trees (n.trees) = 0 to 100 by 50.
- 6) In this case the execution time is over a minute which is more than double as compared to experiment 1 as now there are 606 combinations of the parameters .
- 7) The best tune parameter combination in terms of measures of fit for experiment 2 of GBM model tuning are as follows:

Learning rate	Interaction Depth	Minimum observation in node	Number of trees	RMSE	R squared	MAE	RMSE-SD	R squared-SD	MAESD
0.02	8	2	800	3.172	0.885	2.187	0.803	0.0654	0.368

Table 4: GBM model tuning experiment 2

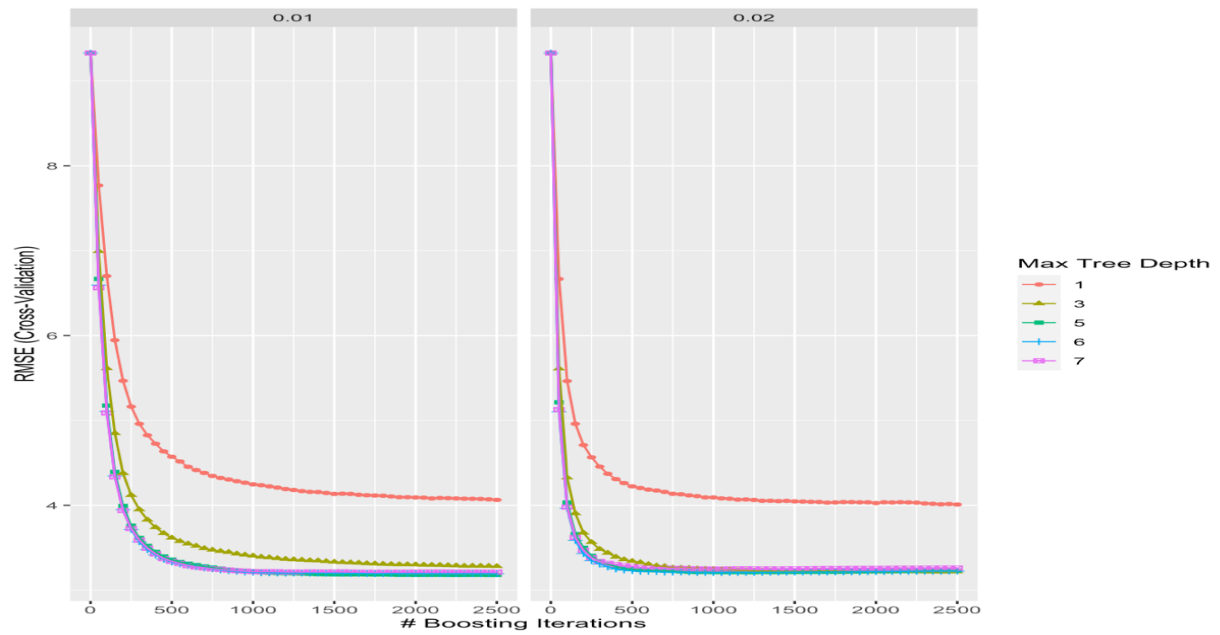


figure 1 : plot of GBM model tuning in experiment 1.

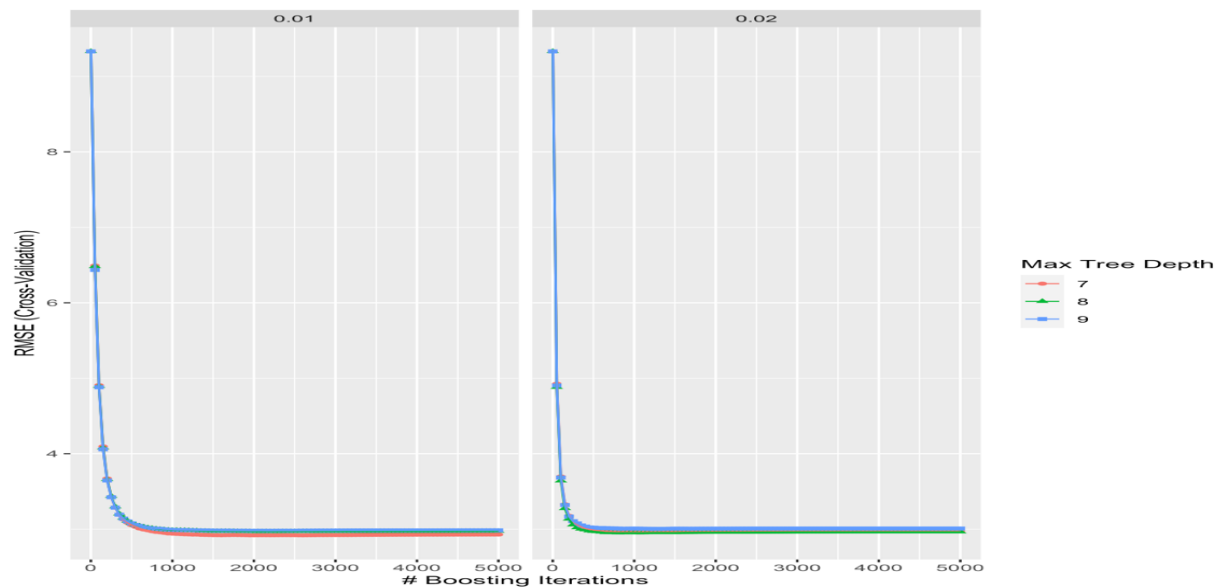


figure 2 : plot of GBM model tuning in experiment 2.

As the number of iterations increases, we can observe that RMSE is dropping. From figure 1 which is a ggplot of GBM model tuning in experiment 1, the lowest RMSE value is observed when learning rate is 0.01, interaction depth is 5 (green), number of trees is 2450 which justifies the results obtained in Table 1. As the number of parameter combinations are now increased in experiment 2, the lowest RMSE is observed when learning rate is 0.02, interaction depth is 8 (green) but the number of trees in this case is observed to be 800 which is less than in experiment 1 as can be seen from figure 2.

After tuning the parameters in this experiment the RMSE hardly changes, so the model can be said to be somewhat consistent irrespective of the parameters in terms of measures of fit. By tuning the parameters carefully, the execution time can be reduced (as in experiment 1) with the measures of fit being unaffected as it can be observed that the RMSE values which

only differs in decimal points (3.179 & 3.172) in the two parameter tuning experiments carried out for GBM model. After having trained a well-tuned model, next step is to evaluate the model by applying it to the hold out data in the initial phase of data pre-processing for testing purpose.

Model evaluation :

Following the training of a well-tuned model, the GBM model is now evaluated by applying it to validation dataset which is hold out in the initial phase of data pre-processing for testing purpose. The generalizability ability of the model is being checked indicating how the model would perform for an unknown test dataset. The plot of predicted against observed values from GBM model is shown in the figure below for test data, where model fit is indicated by blue line and variation in prediction by red line. The model fit for both the models (experiment 1 and 2 model tuning) looks approximately similar and is descent enough as well as the RMSE for both the models are in very close range as can be seen from the Figure 3 and table 3, however experiment 2 takes more execution time as the parameter combinations are more. [4]

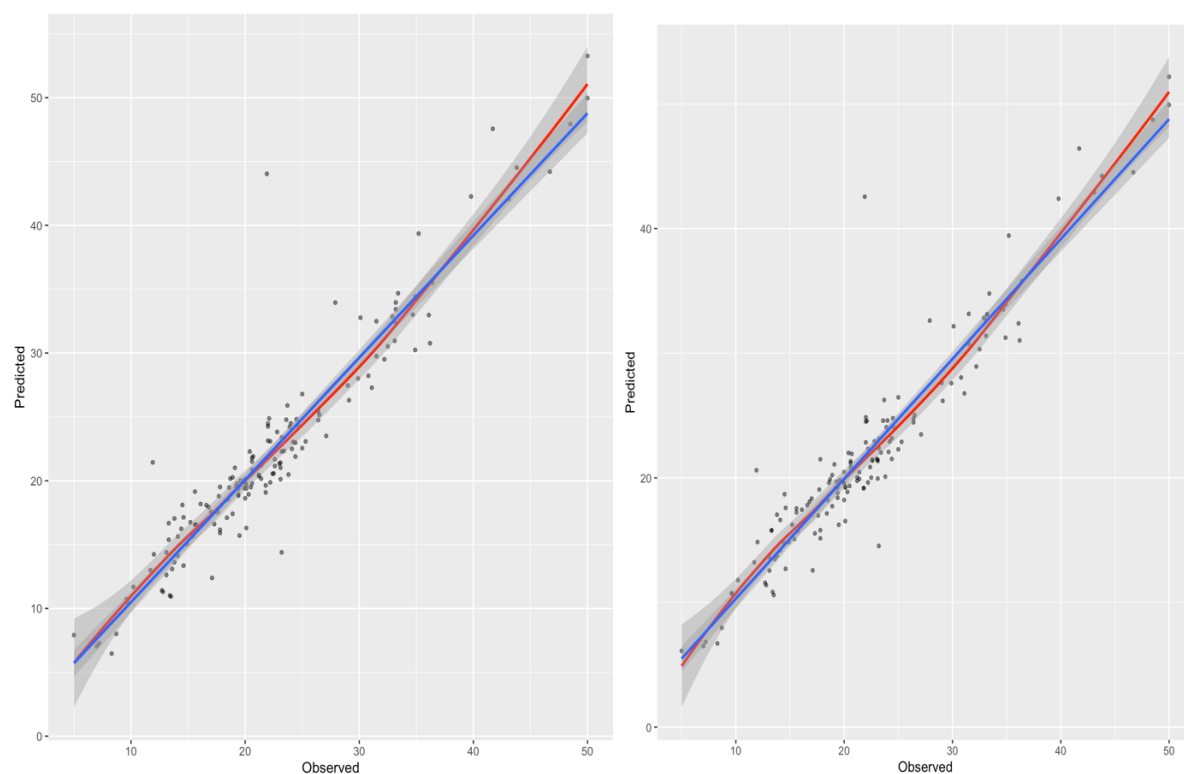


Figure 3 : predicted against observed house price ('medv') in BostonHousing dataset for model tuning experiment 1 and 2

	RMSE	R-squared	MAE
Experiment 1	2.880	0.893	1.828
Experiment 2	2.853	0.896	1.816

Table 3: Prediction accuracy measures of model fit for validation dataset

Evaluation of **variable importance** is done that is which of the predictor variables contributes the most and the least for the prediction of our target variable housing prices 'medv' for the validation dataset. As can be seen from the figure 4 the most important predictor variable is 'lstat': percentage of lower status of the population followed by 'rm': average number of room per dwelling whereas 'indus': proportion of non-retail business acres per town contributes the least for both of the experiment 1 and experiment 2. [4]

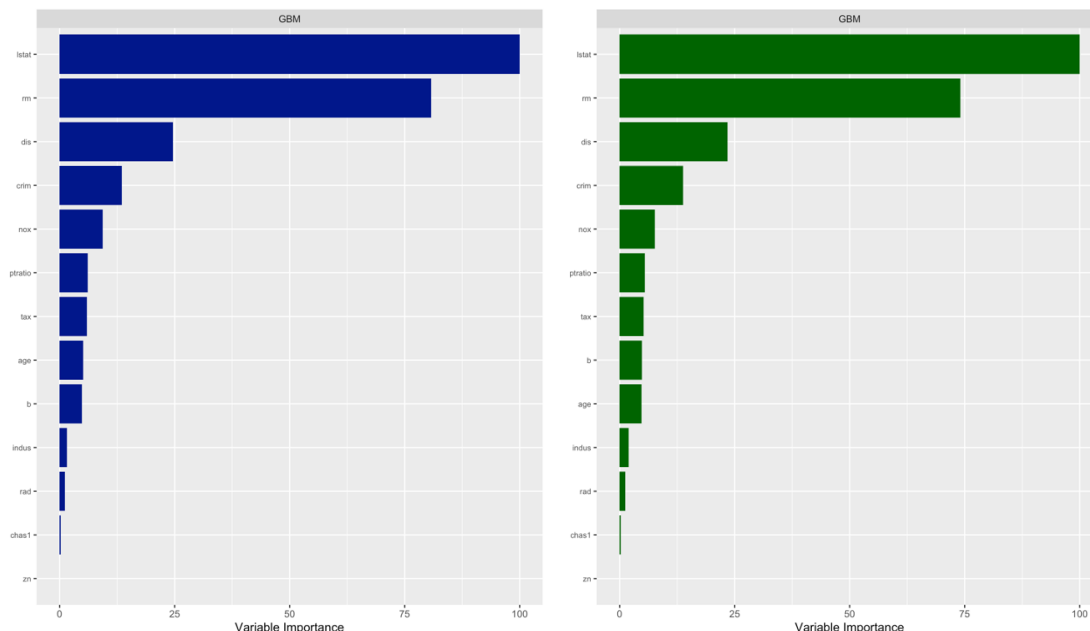


Figure 4: Bar chart of the predictor variables in terms of impacting the variation of the final prediction of the housing prices for experiment 1 and 2.

Discussion

In both of the GBM model tuning experiments carried out, cross validation with k=10 folds is done in order to avoid overfitting. The number of folds here can be altered in order to optimise the model performance by conducting more experiments with variation in the these folds. The value of k needs a careful thought if the sample size is small. As the sample size increases the possible combinations of the parameters becomes large so it doesn't carry a great risk of having duplicates.

The two sets of GBM model tuning experiments are being carried out here. In experiment 2 the possible parameter combinations are increased by varying the number of trees and the interaction depth in order to see whether the performance of the model in terms of measures of fit will improve but as observed from table 2 and table 3 there is not much difference in the results which means model is consistent, though the execution time of second experiment was almost double the first one. However if the model is tuned in a particular way there might be a possibility out there of better model performance, so the experiment of model tuning can be carried out multiple times further in order to find the best possible parameter combinations resulting into better measure of fit which is chosen as RMSE here. However if there are high end configuration machines on which our model can be run, we can explore a

very large number of tuning parameter combinations that might give the better results and the execution time won't be much as well.

After the model is well tuned, the model evaluation is done by applying it to the validation dataset. From the figure 3 we can observe that there is no significant difference in observed and the predicted values of our target variable 'medv' (house price). We can infer that the model performance is somewhat good and consistent for both the experiments as the blue line indicating model fit and the red line giving indication of the variation is almost aligned to each other however as the house prices increases it can be observed to slightly differ from each other. The model can be said to be doing reasonably well at predicting the house prices as the data points of predicted against observed lie somewhat on or very close to the model fit (blue line). As it can also be inferred from the table 3 consisting of accuracy measures of the model where the RMSE values are ranging from 2.880 to 2.853 for the two experiments which is considerably small and can said to be a good measure of fit for the model. So overall aim of predicting the final house prices using a GBM model can said to be satisfactory with a decent model accuracy but there might be further room for improvement by using some other machine learning technique. As this is a problem of supervised learning, there are a lot more machine learning techniques out there such as k-nearest neighbour, bagged regression trees, support vector machine etc. which could have been used here to check whether they give better accuracy measures than GBM.

Initially all the variables were chosen but some of the predictor variables which had negligible influence on the variation of our target variable housing prices 'medv' could have been left out while training the GBM model such as: 'zn', 'chas', 'rad', 'indus' as can be seen in figure 4. It is clear that the predictor variable 'lstat' and 'rm' has the most impact on prediction of target variable housing prices in 'BostonHousing' dataset. In the very beginning while understanding the dataset, the intuition was that the number of rooms per dwelling would have the most impact on housing prices but turns out that's not the case here, 'lstat': percentage of lower status of the population has more impact on it so this analysis helped to get the facts correct. It is largely dependent on how the socioeconomic status of the society is as well. The 'dis' is the third most important predictor variable which makes sense as people will prefer the employment centres to be near to make the daily travel convenient. However this might not be the case in the real world scenario. As the data is small, it might not be the correct representation of the real world. It might be the case that some other predictor variable can have more impact on the prediction of housing prices in real. So it might not be a good idea to jump on the conclusions based on the results we obtained in this report.

References

- [1] He, Zhiyuan, et al. "Gradient Boosting Machine: A Survey." *Arxiv.org*, 19 Aug. 2019
- [2] Friedman, J. "Greedy Function Approximation: A Gradient Boosting Machine." 2001
- [3] Natekin A and Knoll "A Gradient boosting machines, a tutorial." *Front. Neurorobot*, 04 December 2013

[4] Lex Comber “Practical 4: Machine Learning” University of Leeds , January 2022