ELİF DOĞAN DAR
2206753
IAM 557- TERM PROJECT
13.01.2017

SOME REGRESSION AND CLASSIFICATION TECHNIQUES ON BOSTON HOUSE DATA

BOSTON Housing data is a data set taken from the StatLib library which is maintained at Carnegie Mellon University(*). It is a famous set which has been used in many papers. In this project, I will implement some regression techniques on the data and after changing dependent variable a bit also classification techniques. This way I hope to transform theoretical knowledge I attained in class to practical information. I will be using MatLab, which I started using only a few months ago, for this course. As a beginner, this project will be a great opportunity for me to learn both MatLab and how to learn from data. The reasons I chose Boston data are these: First it is a clean data set, since I am a beginner and instead of cleaning data I want to focus on learning MatLab and techniques I thought it would be more appropriate to start with clean data. Second it a very understandable data set from real life.

Boston Housing Data concerns housing value in suburbs of Boston. It has 506 instances and 13 attributes which are real valued where one of them is binary. 14^{Th} is the dependent variable. Attribute information is the following:

1. CRIM per capita crime rate by town
2. ZN proportion of residential land zoned for lots over 25,000 sq.ft.
3. INDUS proportion of non-retail business acres per town
4. CHAS Charles River dummy variable (= 1 if tract bounds river; 0 otherwise)
5. NOX nitric oxides concentration (parts per 10 million)
6. RM average number of rooms per dwelling
7. AGE proportion of owner-occupied units built prior to 1940
8. DIS weighted distances to five Boston employment centers
9. RAD index of accessibility to radial highways
10. TAX full-value property-tax rate per $10,000
11. PTRATIO pupil-teacher ratio by town
12. B 1000(Bk - 0.63)^2 where Bk is the proportion of blacks by town
13. LSTAT % lower status of the population
14. MEDV Median value of owner-occupied homes in $1000's

First, I will implement regression and then after changing dependent variable from continuous prices to 'high price', 'medium price', 'low price', I will study classification techniques.

PRELIMINARY INVESTIGATION ON DATA:

Here I used corrcoef() function to find correlation coefficients, plotmatrix() function to find scatter plots of all features and dependent variables at once, and imagesc() function for visualization of correlation coefficients.

As we can see from scatter plots and correlation coefficients(Figure 1), there is a positive linear relation between dependent variable and RM average number of rooms, which is expected. You expect houses to be more expensive as room number gets bigger. Also, there is a strong negative linear relation between dependent variable and LSTAT, percentage of the lower status of the population, again not surprising. People with lower status must settle on cheaper neighborhoods.

While trying to extract information from data, relation between features are important. When some features are highly dependent on each other we can simply delete one of them. Because high correlation means they tell the same story about the data. Adding both instead of one will only make the model more complex with nothing much in return. To find those I checked correlation coefficients(variable R in codes) and scatter plots (Figure 2). One relation was especially prominent. Relation between RAD index of accessibility to radial highways and TAX full-value property-tax rate. First when I looked at the scatter plot it did not seem right. Data didn't look like following a certain linear pattern, there were only two outliers and only two among 506 instances couldn't do any difference. Apparently, just trusting your eyes were wrong. When I tried to check who were these outliers it turned out to be it wasn't just two points. There were too many data points on the same location. So, there were a secret crowd gathered on that point which caused the strong linear relationship. I will remove one of these features from the data set, but also do regression without erasing too and check if there is an important difference among them.

Other strong relations are not strong enough to get rid of one of them. But still there is a relationship. For example, between INDUS and NOX, wherever big industry is, there is air pollution. Also, understandable relation between INDUS and TAX, high industry rate brings more tax. There is a negative relationship between INDUS and DIS weighted distances to five Boston employment centers, because employment centers would be closer to industrial areas.

Another positive linear relation is among NOX nitric oxides concentration, and AGE proportion of owner-occupied units built prior to 1940. Apparently aged areas tend to be more contaminated, which is not very expected by me. It is another beauty of data science; It shows hidden relationships that your instincts fail to catch.
Relation between NOX and DIS is negative. When the distance from employment centers gets smaller, pollution becomes higher. Finally, we have negative relationship between DIS and AGE. Aged buildings are closer to employment areas.

REGRESSION TECHNIQUES

1) REGRESSION TREES: Trees from graph theory are being used as a model in regression trees. Graphs without cycles are called trees. Root means the top node in a tree, child means a node directly connected to another node called parent, the one near to the root called parent, the node far from the root called child. Leaves are the nodes with no children. Depth of a tree is the number of edges of the longest path from root to leaves. Each leaf corresponds to a decision and each split from a parent corresponds to a binary split of a feature. If the value of that feature bigger than a certain value tree produces a child on the right and if the opposite a child on the left. It continues that way until stopping by a stopping criteria.

Regression trees uses mean squared error(MSE) to decide where to split. Algorithm checks MSE for every possible split on each feature and picks the one with the least MSE. But also, there are other

concerns of the algorithm, for example we do not want the observations on a leaf to be too few, to avoid this, algorithm makes the choosing criterion subject to MinLeafSize constraint, etc. Algorithm stops when a a node is pure, which means the error fell under a threshold. Also, we wouldn't want leaves to have too few observations and we stop producing new children from that node if those children have fewer observations than MinLeafSize. Same applies for MinParentSize. Also if we set MaxNumSplits constraint in the algorithm, it stops if it exceeds that number.
While implementing regression tree technique I chose randomly 450 instances among 506 and set them as training data. I kept 56 instances left as test data to be used at the end of data training process. First, I wanted to see some effect of the change of parameters, I chose maximum number of splits parameter to check. To produce a tree I used fitrtree() function and set MaxNumSplits as n, while n changes from 1 to 50, then checked the loss function of each tree with kfoldLoss().
Here instead of finding loss for the training set once with resubLoss() function, I am making cross validation. When we try to compare trees, and try to choose the best one among them we actually try to find the one with minimum test error. To estimate that we are using training error. However, this is not a very good estimator. Our algorithm might memorize the data and give bad predictions although it gives perfect results on the training set; it might overfit. Therefore, instead of using training set once we can divide training set into k parts and produce k folds, where each fold consists of 9 of these training parts as training set and 1 for validation. Then we look loss for each fold and take average of them. This way loss is more trustable since we are not using validation data while training. Here first I use function crossval() to produce k-fold and then kfoldLoss() to find average loss among folds. As you can see in the figure k fold loss is almost always greater than simple loss, therefore simple loss function on the training set is too optimistic estimator for the test loss(Figure 3).
As you can see in the figure, k fold loss is decreasing at first then starts to fluctuate; there is not much difference to choose between 6 and 45. Of course if there is not much gain, instead of using a bigger number we would choose smaller number to make complexity less. Because we would like to gain from computational costs. Therefore I choose 6, and call the tree as tree1. I visualize tree1 with view() function(Figure 6), predicted test y and training y values with predict() function, plot test y and predicted test y on the same figure(Fıgure 4), and in the end calculate test(13.09) and training error(16.96).

Here as you have noticed I optimized only one parameter. If I had wanted to optimize two parameters instead, for example both 'maximum number of splits' and 'minimum leaf size' ; then I would have to write two loops inside each other, because choice of one affects the other. Thankfully we do not have to do this process manually for each parameter, there is an option in fitrtree() function which is 'OptimizeHyperparameters'. It directly gives us an automatically optimized tree. However, since there is no upper bound for the maximum number of splits, it has found too complicated trees like with 250 splits. Since we don’t need perfect optimization, to get rid of this problem I put a constraint maximum number of splits as 100. Then I checked test error of this optimized tree too, which is 12.32. Test error of optimized tree is less than the first tree as expected.

Now since at the beginning we noticed that feature TAX can be avoided, I will check the difference of the test errors between tree at the beginning and tree that is being constructed from the predictors except TAX, which will be called as tree2 in codes. Test error for this tree is 13.09 which is same as the first tree.

2) LINEAR REGRESSION: Linear regression is a model which assumes that effects of features on the dependent variable are linear. It is fitted by least squares approximation. To implement it in MatLab i used fitlm() function and feval() function for prediction. Test error for linear regression became 24.13 which is higher compared to the regression trees. R-squared became 0.736, which means nearly 73 percent of the variability of the response data around its mean is

being explained by the model, which is acceptable but not as good as trees for this data set. I also checked p-values of the features INDUS and AGE have very high people values 0.76 and 0.68, which suggests that they can be avoidable, they say very less about the data. Therefore, I made another fitted linear model without them, lmnew in codes. It gave test error 28.57, which is not very different from the earlier test error.

I also implemented quadratic linear regression, which takes pairwise multiplication of variables and squares of the variables too as predictors. It makes linear regression on a bigger dimensional place and projects back, which means boundaries look like quadratic instead of linear. Here test error became 13.47 which is way better than simple linear regression and as good as trees. It R-squared value is also very big large 0.929.

CLASSIFICATION TECHNIQUES

Now I will change dependent variable in data to implement some classification techniques. Y variable is the mean value of the homes, I will classify houses with value less than 15 as 1(cheap), the ones between 15 and 30 as 2(medium), and more than 30 as 3(expensive). Then again by using training data, I will try to build a model to guess y values in the future.

1) K-NEAREST NEIGHBORHOOD : In that technique for each point we look at k nearest neighbors of that point and take mean value of their classes, round to the nearest class, that rounded mean class is the assigned class of the point. I built the model by using fitcknn() function with k=5 and predict the values with predict() function. Here we use confusion matrix; its dimensions are number of classes. If there is integer l on the (i,j)th entry it means the number of data points which belongs to I and classified as j. Therefore, sum of numbers on the diagonals gives the number of correctly classified points and sum of number on non-diagonal entries give the number of misclassified points. For this model and data set we got 7 misclassifications.

But why did we choose number of neighbors 5, it could be any positive integer. To find the optimal parameter I have found resubstitution error and k fold error by using resubLoss() and kfoldLoss() functions for each number of neighborhood from 1 to 10. As expected resubstitution error is overly optimistic. In our data as you can see in the Figure 5, difference between choosing different number of k's is not visible. Therefore, we can continue with the choice of 5.

2)LINEAR and QUADRATIC DISCRIMINANT ANALYSIS: In linear and quadratic discriminant analysis, our model assume that probability of the classes follow Gaussian distribution with different parameters. To make the model I used fitcdiscr() function and predict() function for predictions. Here I had 6 misclassifications.

I also implemented quadratic discriminant analysis with same functions with 'DiscrimType' 'quadratic'. However here I got an error. Some part of the algorithm standardize data and since standard deviation of one of the features for one of the classes was zero. Therefore, I changed the type from 'quadratic' to 'pseudoquadratic'. Here I had 13 misclassifications.

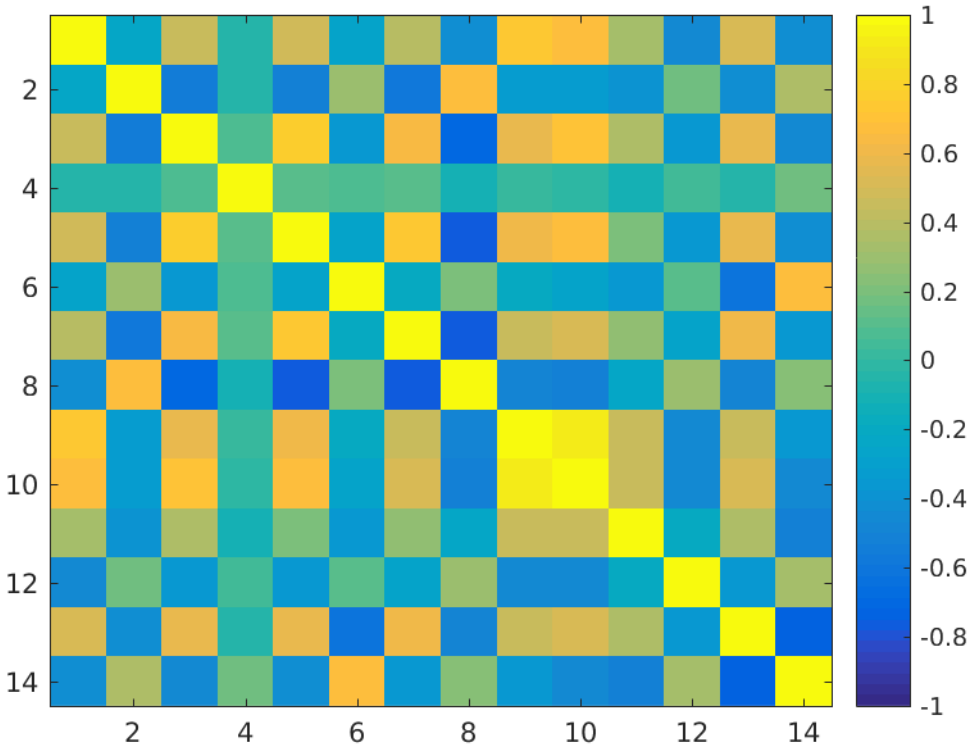


FIGURE 1

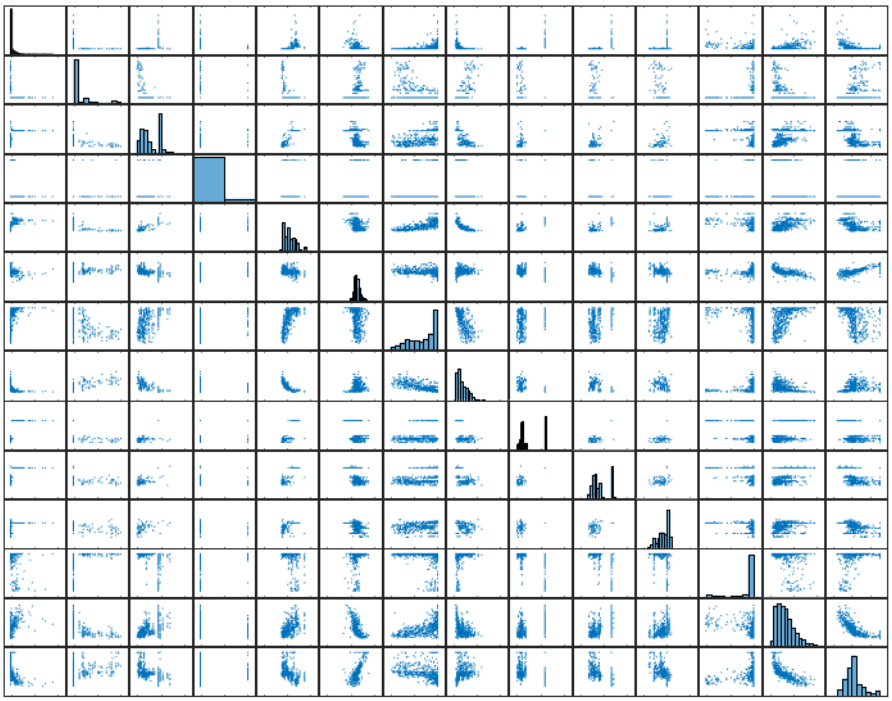


FIGURE 2

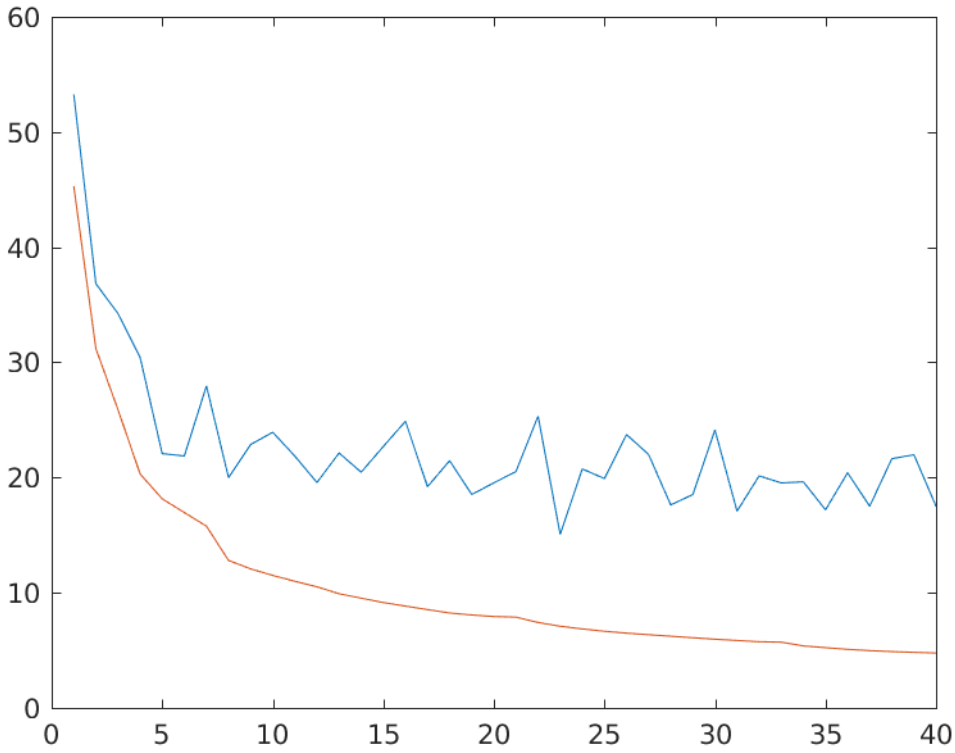


FIGURE 3: Resubstitution(red) and k fold loss(blue) of trees with different number of max number of splits

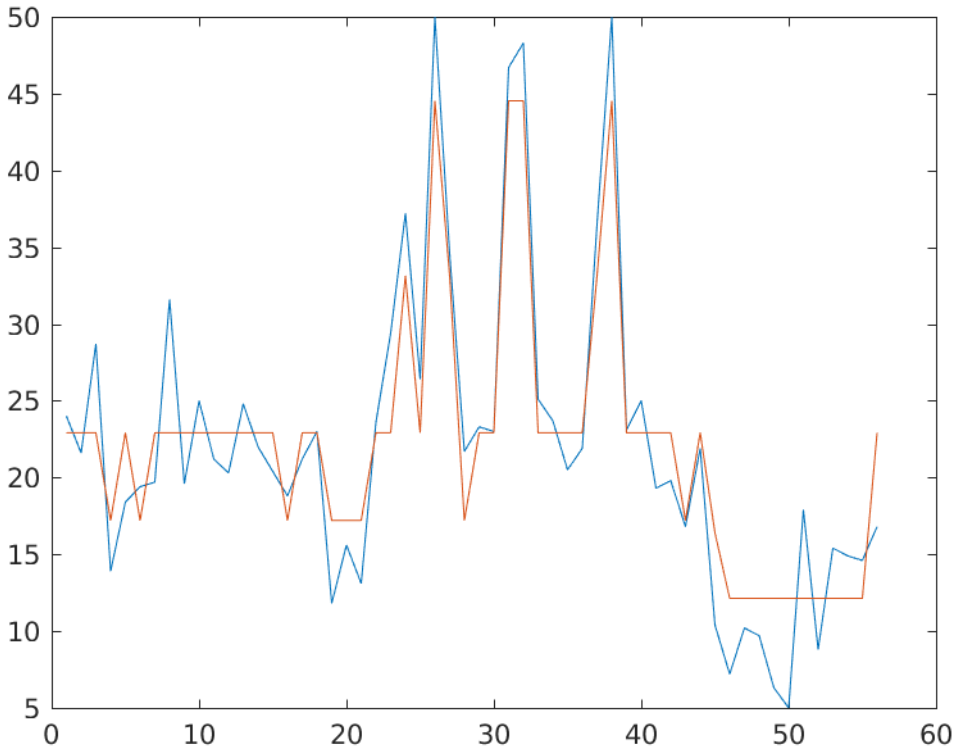


FIGURE 4: y and predicted y values of the test data set

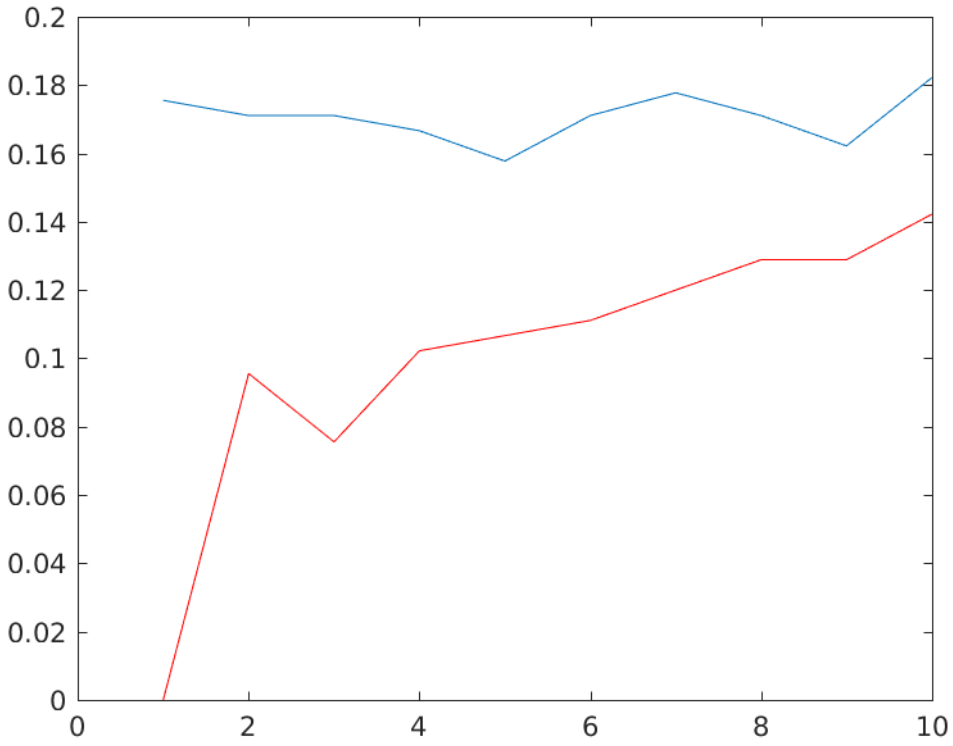


FIGURE 5: Simple loss(red) and kloss(blue) values for different number of neighbors

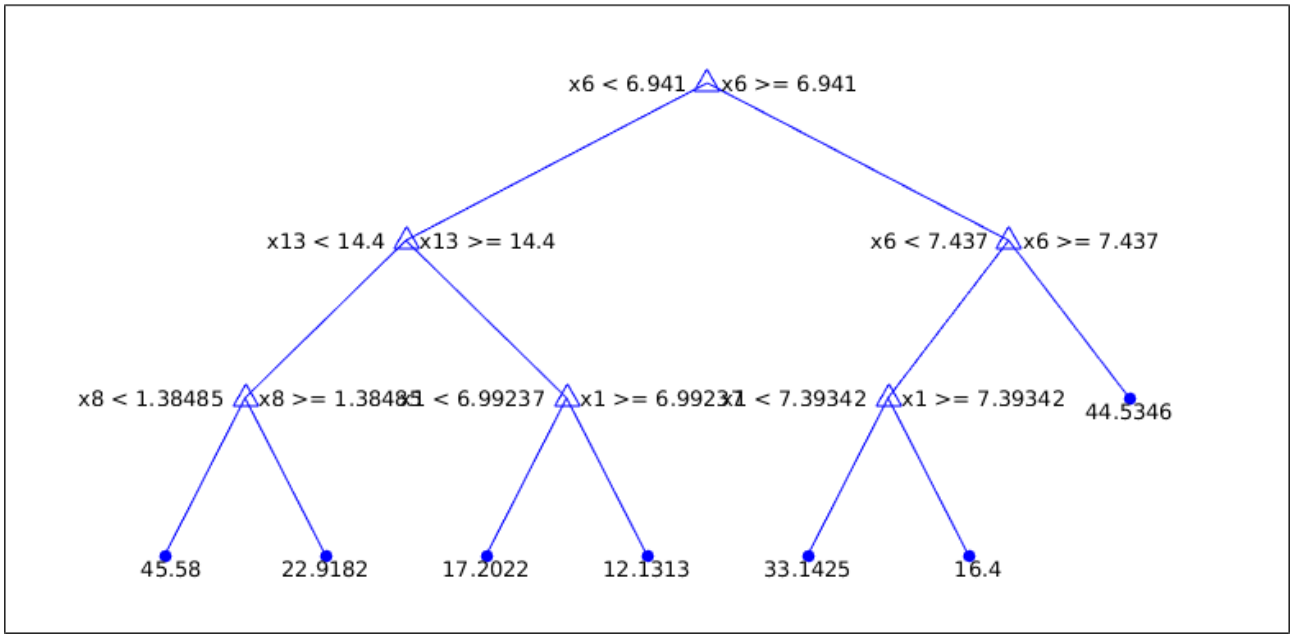
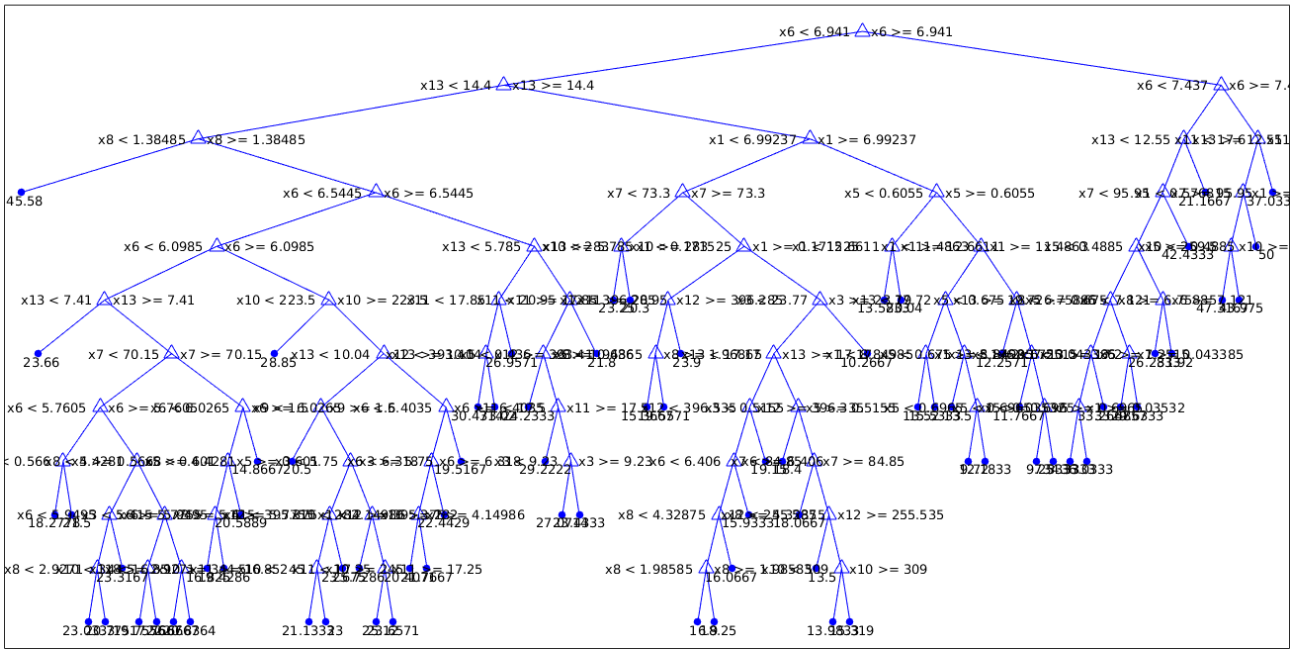
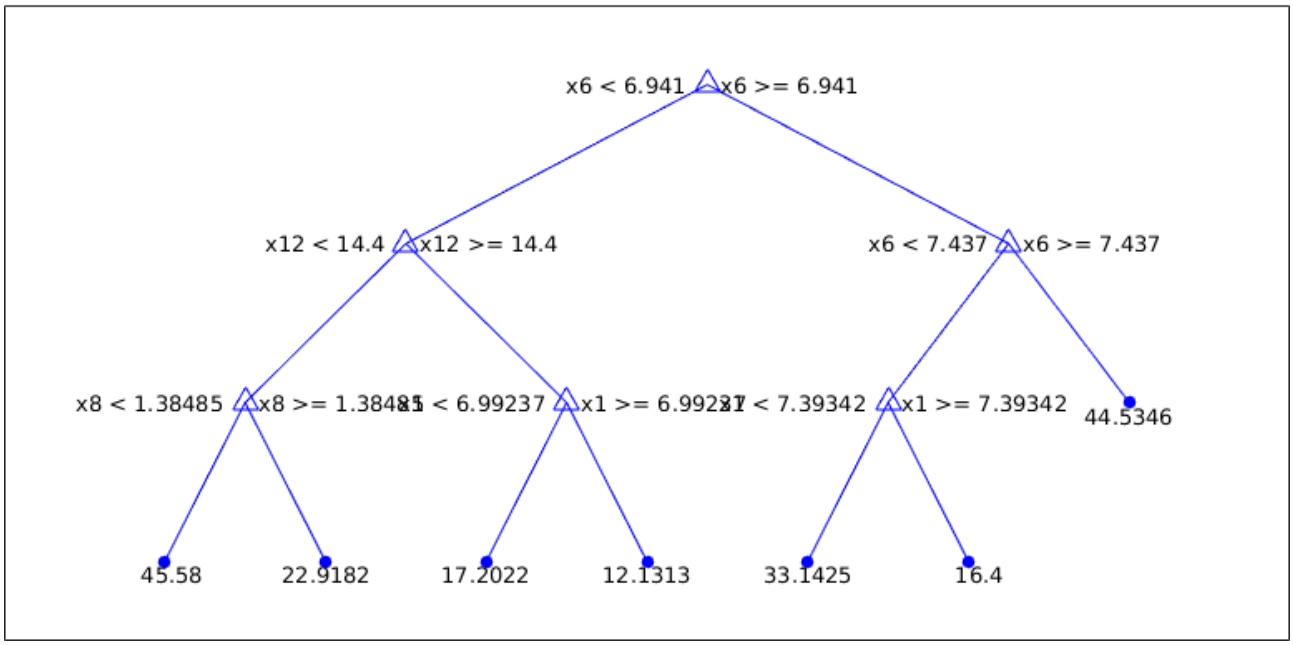


FIGURE 6: Tree1



Optimized tree



Tree2

(*) Harrison, D. and Rubinfeld, D.L. 'Hedonic prices and the demand for clean air', J. Environ. Economics & Management, vol.5, 81-102, 1978.