***Course: IE 7275 16692 – Data Mining in Engineering***

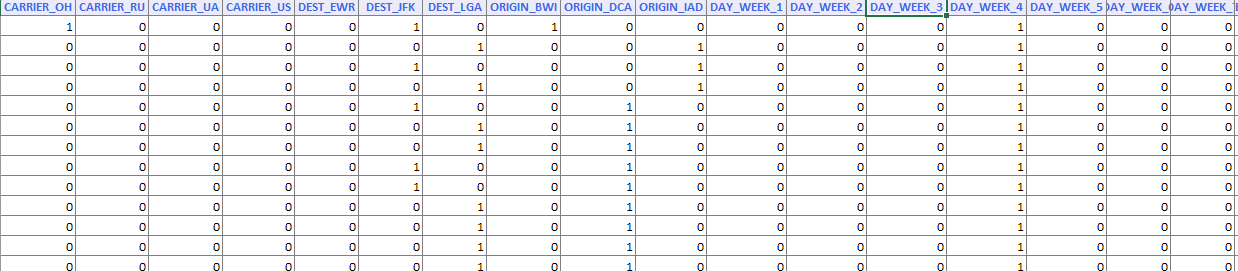
***Group 8***

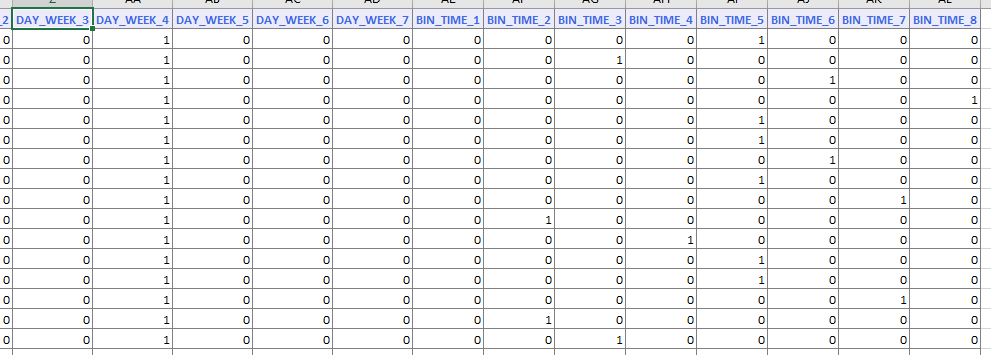
***Names of group members and NUID: 1) Gandhar Kothari (NUID: 001676935)***

***2) Siddharth Patil (NUID: 001680396)***

***Document name: Homework 6 – Submission***

1. **Data Pre-processing –**
2. As required, I have created the required dummy variables in XL Miner as can be seen in the following pictures:





1. Importing the dataset and deleting the DEP\_TIME variable which is in column 2

Code:

DS <- read.xlsx(file.choose(), sheetName = 'Flight\_delays', header = TRUE)

DataSetdf <- as.data.frame(DS)

DSdf <- DataSetdf[,-2]

DSdf <- DataSetdf[,-7]

1. Creating training and test datasets

Code:

#Creating validation and test data sets

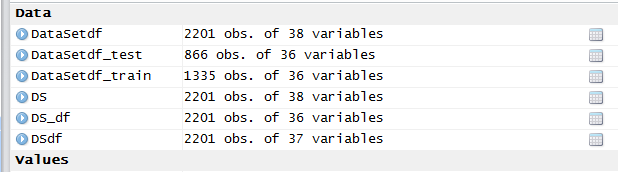
set.seed(123)

ind <- sample(2, nrow(DataSetdf), replace=TRUE, prob=c(0.60, 0.40))

DataSetdf\_train<- DataSetdf[ind==1, c(1:38)]

DataSetdf\_test<- DataSetdf[ind==2, c(1:38)]

Result for b) & c):



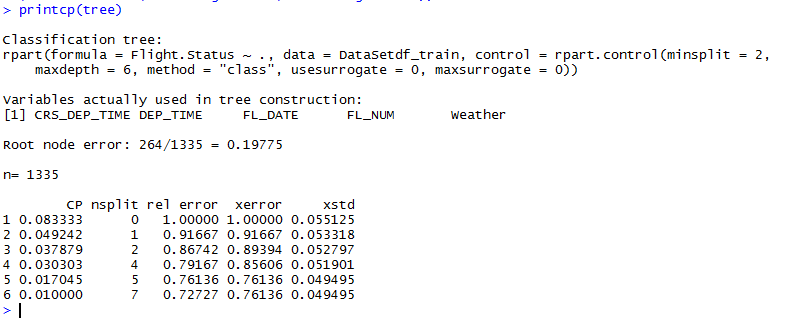
1. **Fitting a classification tree:**

Code:

tree<- rpart(Flight.Status~.,data = DataSetdf\_train,control = rpart.control(minsplit = 2,maxdepth = 6,method='class',usesurrogate = 0, maxsurrogate = 0))

printcp(tree)

Result:



Summary:

Code: summary(tree)

Result:

> summary(tree)

Call:

rpart(formula = Flight.Status ~ ., data = DataSetdf\_train, control = rpart.control(minsplit = 2,

maxdepth = 6, method = "class", usesurrogate = 0, maxsurrogate = 0))

n= 1335

CP nsplit rel error xerror xstd

1 0.08333333 0 1.0000000 1.0000000 0.05512545

2 0.04924242 1 0.9166667 0.9166667 0.05331792

3 0.03787879 2 0.8674242 0.8939394 0.05279713

4 0.03030303 4 0.7916667 0.8560606 0.05190097

5 0.01704545 5 0.7613636 0.7613636 0.04949485

6 0.01000000 7 0.7272727 0.7613636 0.04949485

Variable importance

DEP\_TIME Weather CRS\_DEP\_TIME FL\_NUM FL\_DATE

41 28 21 6 4

Node number 1: 1335 observations, complexity param=0.08333333

predicted class=ontime expected loss=0.1977528 P(node) =1

class counts: 264 1071

probabilities: 0.198 0.802

left son=2 (22 obs) right son=3 (1313 obs)

Primary splits:

Weather < 0.5 to the right, improve=28.792910, (0 missing)

DEP\_TIME < 1456.5 to the right, improve=18.334750, (0 missing)

CRS\_DEP\_TIME < 1415 to the right, improve= 9.901410, (0 missing)

FL\_NUM < 2808 to the right, improve= 8.510356, (0 missing)

CARRIER\_US < 0.5 to the left, improve= 8.265873, (0 missing)

Node number 2: 22 observations

predicted class=delayed expected loss=0 P(node) =0.0164794

class counts: 22 0

probabilities: 1.000 0.000

Node number 3: 1313 observations, complexity param=0.04924242

predicted class=ontime expected loss=0.1843107 P(node) =0.9835206

class counts: 242 1071

probabilities: 0.184 0.816

left son=6 (13 obs) right son=7 (1300 obs)

Primary splits:

DEP\_TIME < 2147.5 to the right, improve=17.472060, (0 missing)

CRS\_DEP\_TIME < 1415 to the right, improve= 9.912664, (0 missing)

FL\_NUM < 2808 to the right, improve= 6.677721, (0 missing)

CARRIER\_US < 0.5 to the left, improve= 6.444505, (0 missing)

DISTANCE < 213.5 to the left, improve= 4.531339, (0 missing)

Node number 6: 13 observations

predicted class=delayed expected loss=0 P(node) =0.009737828

class counts: 13 0

probabilities: 1.000 0.000

Node number 7: 1300 observations, complexity param=0.03787879

predicted class=ontime expected loss=0.1761538 P(node) =0.9737828

class counts: 229 1071

probabilities: 0.176 0.824

left son=14 (600 obs) right son=15 (700 obs)

Primary splits:

DEP\_TIME < 1456.5 to the right, improve=13.854400, (0 missing)

CRS\_DEP\_TIME < 1415 to the right, improve= 7.647819, (0 missing)

CARRIER\_US < 0.5 to the left, improve= 5.446538, (0 missing)

FL\_NUM < 2185 to the right, improve= 4.847652, (0 missing)

DISTANCE < 213.5 to the left, improve= 4.037538, (0 missing)

Node number 14: 600 observations, complexity param=0.03787879

predicted class=ontime expected loss=0.255 P(node) =0.4494382

class counts: 153 447

probabilities: 0.255 0.745

left son=28 (44 obs) right son=29 (556 obs)

Primary splits:

CRS\_DEP\_TIME < 1477.5 to the left, improve=21.180920, (0 missing)

BIN\_TIME\_5 < 0.5 to the right, improve= 9.468480, (0 missing)

FL\_NUM < 1287 to the left, improve= 9.040175, (0 missing)

DISTANCE < 213.5 to the left, improve= 6.234159, (0 missing)

BIN\_TIME\_8 < 0.5 to the left, improve= 5.322469, (0 missing)

Node number 15: 700 observations

predicted class=ontime expected loss=0.1085714 P(node) =0.5243446

class counts: 76 624

probabilities: 0.109 0.891

Node number 28: 44 observations, complexity param=0.03030303

predicted class=delayed expected loss=0.2727273 P(node) =0.0329588

class counts: 32 12

probabilities: 0.727 0.273

left son=56 (28 obs) right son=57 (16 obs)

Primary splits:

DEP\_TIME < 1504.5 to the right, improve=11.454550, (0 missing)

DAY\_WEEK\_3 < 0.5 to the left, improve= 2.004545, (0 missing)

FL\_NUM < 7557.5 to the left, improve= 1.810101, (0 missing)

DEST\_JFK < 0.5 to the left, improve= 1.636364, (0 missing)

CRS\_DEP\_TIME < 1442.5 to the left, improve= 1.454545, (0 missing)

Node number 29: 556 observations, complexity param=0.01704545

predicted class=ontime expected loss=0.2176259 P(node) =0.4164794

class counts: 121 435

probabilities: 0.218 0.782

left son=58 (22 obs) right son=59 (534 obs)

Primary splits:

FL\_NUM < 1287 to the left, improve=6.383562, (0 missing)

CARRIER\_CO < 0.5 to the right, improve=6.383562, (0 missing)

CARRIER\_MQ < 0.5 to the right, improve=4.003611, (0 missing)

DISTANCE < 213.5 to the left, improve=3.545499, (0 missing)

FL\_DATE < 37989.5 to the right, improve=3.521395, (0 missing)

Node number 56: 28 observations

predicted class=delayed expected loss=0 P(node) =0.02097378

class counts: 28 0

probabilities: 1.000 0.000

Node number 57: 16 observations

predicted class=ontime expected loss=0.25 P(node) =0.01198502

class counts: 4 12

probabilities: 0.250 0.750

Node number 58: 22 observations, complexity param=0.01704545

predicted class=delayed expected loss=0.4090909 P(node) =0.0164794

class counts: 13 9

probabilities: 0.591 0.409

left son=116 (15 obs) right son=117 (7 obs)

Primary splits:

FL\_DATE < 38007.5 to the left, improve=4.1220780, (0 missing)

DEP\_TIME < 1731.5 to the right, improve=2.9696970, (0 missing)

DAY\_WEEK\_5 < 0.5 to the left, improve=1.9775400, (0 missing)

DAY\_WEEK\_1 < 0.5 to the right, improve=1.6363640, (0 missing)

CRS\_DEP\_TIME < 1680 to the left, improve=0.7363636, (0 missing)

Node number 59: 534 observations

predicted class=ontime expected loss=0.2022472 P(node) =0.4

class counts: 108 426

probabilities: 0.202 0.798

Node number 116: 15 observations

predicted class=delayed expected loss=0.2 P(node) =0.01123596

class counts: 12 3

probabilities: 0.800 0.200

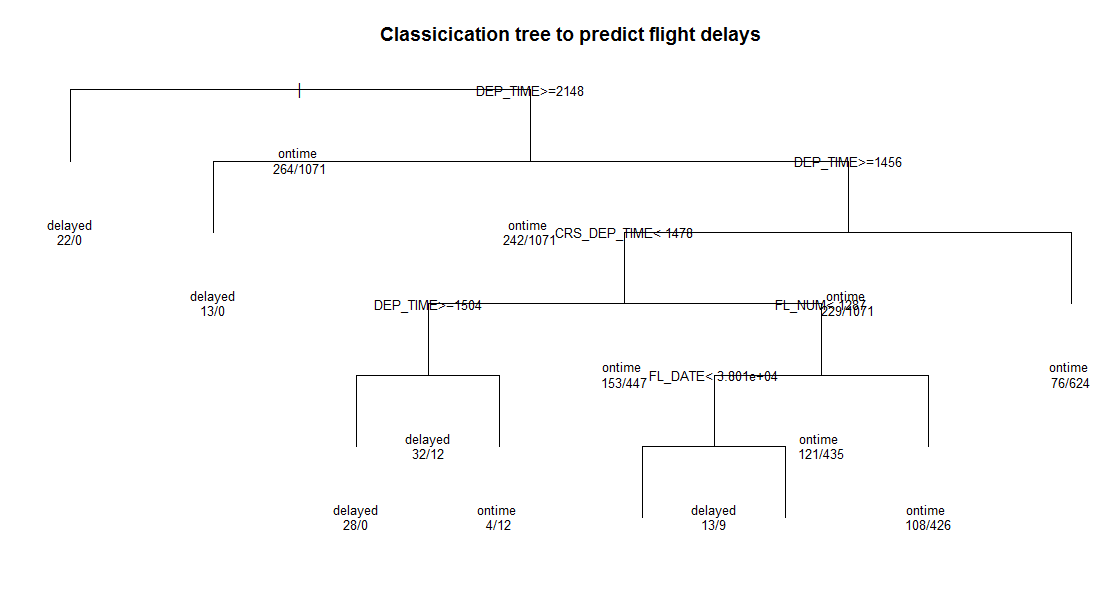
Node number 117: 7 observations

predicted class=ontime expected loss=0.1428571 P(node) =0.005243446

class counts: 1 6

probabilities: 0.143 0.857

Plot for the tree:



**Defining the rules set by the tree:**

Code:

asRules(tree)

Result:

> asRules(tree)

Rule number: 15 [Flight.Status=ontime cover=700 (52%) prob=0.89]

Weather< 0.5

DEP\_TIME< 2148

DEP\_TIME< 1456

Rule number: 117 [Flight.Status=ontime cover=7 (1%) prob=0.86]

Weather< 0.5

DEP\_TIME< 2148

DEP\_TIME>=1456

CRS\_DEP\_TIME>=1478

FL\_NUM< 1287

FL\_DATE>=3.801e+04

Rule number: 59 [Flight.Status=ontime cover=534 (40%) prob=0.80]

Weather< 0.5

DEP\_TIME< 2148

DEP\_TIME>=1456

CRS\_DEP\_TIME>=1478

FL\_NUM>=1287

Rule number: 57 [Flight.Status=ontime cover=16 (1%) prob=0.75]

Weather< 0.5

DEP\_TIME< 2148

DEP\_TIME>=1456

CRS\_DEP\_TIME< 1478

DEP\_TIME< 1504

Rule number: 116 [Flight.Status=delayed cover=15 (1%) prob=0.20]

Weather< 0.5

DEP\_TIME< 2148

DEP\_TIME>=1456

CRS\_DEP\_TIME>=1478

FL\_NUM< 1287

FL\_DATE< 3.801e+04

Rule number: 56 [Flight.Status=delayed cover=28 (2%) prob=0.00]

Weather< 0.5

DEP\_TIME< 2148

DEP\_TIME>=1456

CRS\_DEP\_TIME< 1478

DEP\_TIME>=1504

Rule number: 6 [Flight.Status=delayed cover=13 (1%) prob=0.00]

Weather< 0.5

DEP\_TIME>=2148

Rule number: 2 [Flight.Status=delayed cover=22 (2%) prob=0.00]

Weather>=0.5

**Part b)**

It is not possible to determine the required information using the tree that we have because we’ll have to take in consideration the destination airport and airport of origin of the flight.

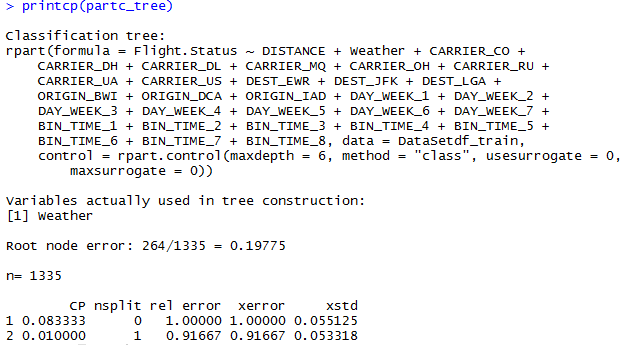
**Part c)**

1. **Building the tree**

Code:

partc\_tree <- rpart(Flight.Status ~ DISTANCE+ Weather+CARRIER\_CO+CARRIER\_DH+CARRIER\_DL+CARRIER\_MQ+CARRIER\_OH+CARRIER\_RU+CARRIER\_UA+CARRIER\_US+DEST\_EWR+DEST\_JFK+DEST\_LGA+ORIGIN\_BWI+ORIGIN\_DCA+ORIGIN\_IAD+DAY\_WEEK\_1+DAY\_WEEK\_2+DAY\_WEEK\_3+DAY\_WEEK\_4+DAY\_WEEK\_5+DAY\_WEEK\_6+DAY\_WEEK\_7+BIN\_TIME\_1+BIN\_TIME\_2+BIN\_TIME\_3+BIN\_TIME\_4+BIN\_TIME\_5+BIN\_TIME\_6+BIN\_TIME\_7+BIN\_TIME\_8,data = DataSetdf\_train, control = rpart.control(maxdepth = 6,method='class',usesurrogate = 0, maxsurrogate = 0))

Result:



> summary(partc\_tree)

Call:

rpart(formula = Flight.Status ~ DISTANCE + Weather + CARRIER\_CO +

CARRIER\_DH + CARRIER\_DL + CARRIER\_MQ + CARRIER\_OH + CARRIER\_RU +

CARRIER\_UA + CARRIER\_US + DEST\_EWR + DEST\_JFK + DEST\_LGA +

ORIGIN\_BWI + ORIGIN\_DCA + ORIGIN\_IAD + DAY\_WEEK\_1 + DAY\_WEEK\_2 +

DAY\_WEEK\_3 + DAY\_WEEK\_4 + DAY\_WEEK\_5 + DAY\_WEEK\_6 + DAY\_WEEK\_7 +

BIN\_TIME\_1 + BIN\_TIME\_2 + BIN\_TIME\_3 + BIN\_TIME\_4 + BIN\_TIME\_5 +

BIN\_TIME\_6 + BIN\_TIME\_7 + BIN\_TIME\_8, data = DataSetdf\_train,

control = rpart.control(maxdepth = 6, method = "class", usesurrogate = 0,

maxsurrogate = 0))

n= 1335

CP nsplit rel error xerror xstd

1 0.08333333 0 1.0000000 1.0000000 0.05512545

2 0.01000000 1 0.9166667 0.9166667 0.05331792

Variable importance

Weather

100

Node number 1: 1335 observations, complexity param=0.08333333

predicted class=ontime expected loss=0.1977528 P(node) =1

class counts: 264 1071

probabilities: 0.198 0.802

left son=2 (22 obs) right son=3 (1313 obs)

Primary splits:

Weather < 0.5 to the right, improve=28.792910, (0 missing)

CARRIER\_US < 0.5 to the left, improve= 8.265873, (0 missing)

ORIGIN\_DCA < 0.5 to the left, improve= 4.647760, (0 missing)

DISTANCE < 213.5 to the left, improve= 4.592011, (0 missing)

DEST\_LGA < 0.5 to the left, improve= 4.519161, (0 missing)

Node number 2: 22 observations

predicted class=delayed expected loss=0 P(node) =0.0164794

class counts: 22 0

probabilities: 1.000 0.000

Node number 3: 1313 observations

predicted class=ontime expected loss=0.1843107 P(node) =0.9835206

class counts: 242 1071

probabilities: 0.184 0.816

|  |
| --- |
| > plot(partc\_tree, uniform = T, main="Classicication tree for predicting flight delays")  > text(partc\_tree, use.n= T, all=T, cex=.8) |
|  |
| |  | | --- | |  | |

**Defining the rules for this tree**

Result:

> asRules(partc\_tree)

Rule number: 3 [Flight.Status=ontime cover=1313 (98%) prob=0.82]

Weather< 0.5

Rule number: 2 [Flight.Status=delayed cover=22 (2%) prob=0.00]

Weather>=0.5

**Part c questions:**

1. **The rules for this tree are defined as above. Since, this tree has only one node, any new record will be assigned the majority class which is ‘ontime’**
2. **This rule is equivalent to the Naïve Bayes classifier since any new record will be assigned the majority class.**
3. **As can be seen from the summary of this tree, ‘Weather’, ‘CARRIER\_US’, and ‘ORIGIN\_DCA’ are the top three predictors.**
4. **The best pruned tree is defined as the tree that has an error within one standard deviation of the minimum error. The function stops executing when it arrives at a number of nodes which gives this error. Hence, the best pruned tree has only one node.**
5. **The dis-advantage of using the top levels of the classification tree as opposed to the best pruned is that this method may give less error for training data but is bound to give more error for a new dataset.**
6. **a) Logistic regression creates a decision boundary that is not linear as opposed to one created by the classification tree which is linear boundary. This enables logistic regression to partition the data in a such a way that it reduces the impurity in a better way than classification trees. This is, however, best applicable only when there is one decision boundary.**

**b) Logistics regression is less likely to produce an overfitted model as opposed to the classification tree.**

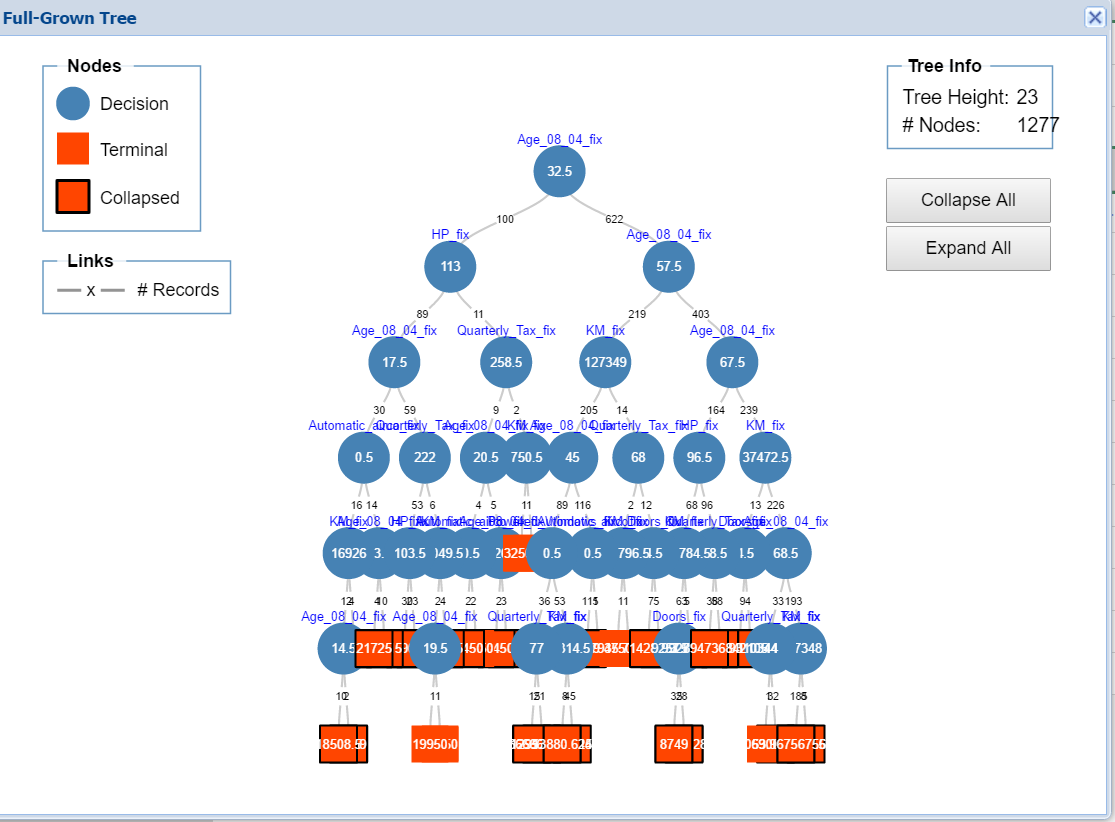
**Problem2**

**(Predicting Price of Used Car, CART) [50 points]**

The file ToyotaCorolla.xlsx contains the data on used cars (Toyota Corolla) on sale during late summer of 2004 in The Netherlands. It has 1436 records containing details on 38 attributes, including Price, Age, Kilometres, HP, and other specifications. The goal is to predict the price of a used Toyota Corolla based on its specifications.

Data Pre processing: Create dummy variables for the categorical predictors (Fuel Type and Color). Split the data into training (50%), validation (30%), and test (20%) datasets.

1. **Run a regression tree (RT) using the Prediction menu in XLMiner with the Output variable Price and input variables Age\_08\_04, KM, Fuel\_Type, HP, Automatic,Doors, Quarterly\_Tax, Mfg\_Guarantee, Guarantee\_Period, Airco, Automatic\_Airco, CD Player, Powered\_Windows, Sport\_Model, and Tow\_Bar. Keep the minimum number of records in a terminal node to 1, maximum number of tree levels to 100, and the scoring option to Full Tree, to make the run least restrictive**.



1. From the regression tree above, the three important car specifications for predicting car price are as follows:

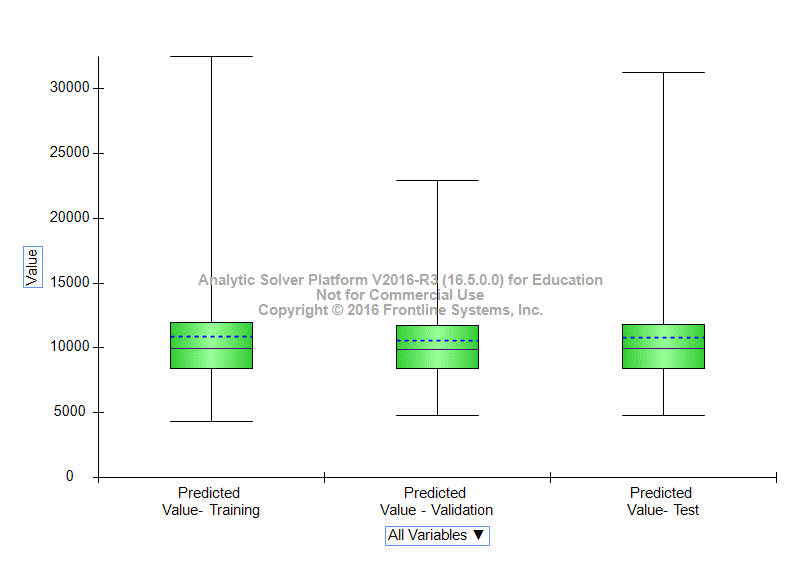
From the above Regression Tree obtained from

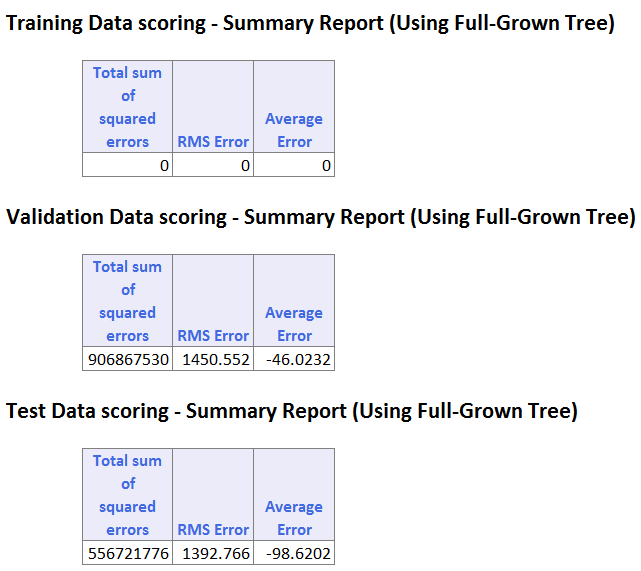
Age\_08\_04, HP and KM

As these factors are on the top level of the tree and are the deciding factors according to the Regression Tree Algorithm.

1. RMS Error is obtained as shown below

Box plot has been plotted for predicted values of the training, validation, and test sets by examining their RMS error.





1. If the full tree is used instead of the best pruned tree to score the validation set, the predictive performance for the validation set would decrease as the error would be greater compared to prediction with pruned tree. The decrease in the predictive performance of the full grown tree is because it contains noise or the error elements in it. Whereas, the pruned tree removes the noise that is the error elements and thus the predictive performance increases.

**b. Let us see the effect of turning the price variable into a categorical variable. First, create a new variable that categorizes price into 20 bins. Use Transform → Bin continuous data to categorize Price into 20 bins of equal counts (leave all other options at their default). Now repartition the data keeping Binned Price instead of Price. Run a classification tree (CT) using the Classification menu of XLMiner with the same set of input variables as in the RT, and with Binned Price as the output variable. Keep the minimum number of records in a terminal node to 1 and uncheck the P rune Tree option, to make the run least restrictive.**





**CLASSIFICATION TREE (FULL-GROWN TREE)**

It is seen that the tree generated by the CT is different than one generated by the RT, though the overall structure and size appears to be the same, the tree actually varies in the number of nodes but the top predictors are same as RT which are Age\_08\_04, HP and KM as they are factors on the top level of the tree and are the deciding factors according to the Classification Tree Algorithm and this is because of the data set provided to us and is not a general trend but in this case it is the same.

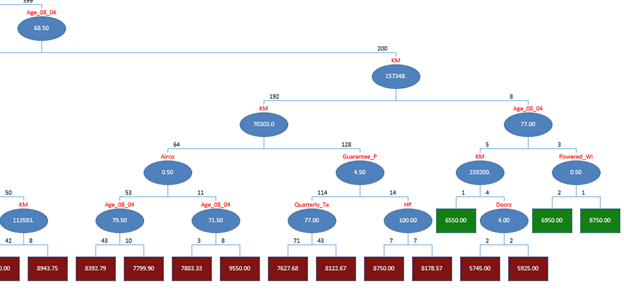
1. Compare the tree generated by the CT with the one generated by the RT. Are they different? (Look at structure, the top predictors, size of tree, etc.) Why?

The Regression tree and the classification tree obtained from has the same size, structure and the predictors variables that are important for the car specifications that are Age\_08\_04, HP and KM. The difference lies in the number of nodes obtained for both trees that is the number of nodes of regression tree is not same as the number of nodes of the classification tree.

1. Predict the price, using the RT and the CT, of a used Toyota Corolla with the

Specifications listed

The given specifications and the regression tree rules when compared, the node reached for the Price of used Toyota corolla is $8122.67.



Similarly, as the comparison used above we reach at the node 2.

We compute the price thereby inferring that the Node 2 belongs to Class 8 and the price comes down to $11260(rounded up).

1. Compare the predictions in terms of the predictors that were used, the

magnitude of the difference between the two predictions, and the advantages

and disadvantages of the two methods.

The main difference in Regression tree and the classification tree is their dependent variable. For the classification tree the predictor variables undertaken are categorical whereas the Regression tree has the continuous prediction variables. Classifications trees also have set amount of unsolder values where the Regression tree have ordered or the discrete values.

The advantage of decision tree they force you to consider as many possible outcomes of a decision as one and think off. The disadvantage is that the more decisions there are in the tree less in accurate any expected outcomes are likely to be.