**Supermarket Organic product data Analysis**

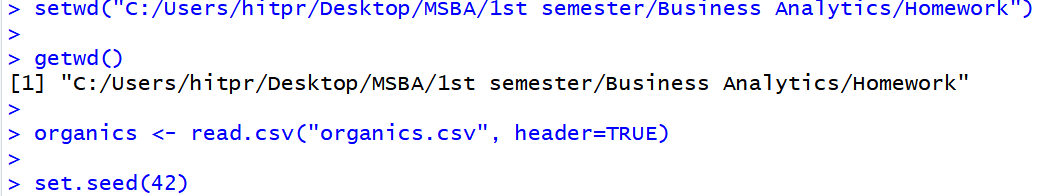
A supermarket is offering a new line of organic products. The supermarket's management wants to determine which customers are likely to purchase these products. The supermarket has a customer loyalty program. As an initial buyer incentive plan, the supermarket provided coupons for the organic products to all of the loyalty program participants and collected data that includes whether these customers purchased any of the organic products.

The ORGANICS data set contains 13 variables and over 22,000 observations. The variables in the data set are shown below with the appropriate roles and levels:

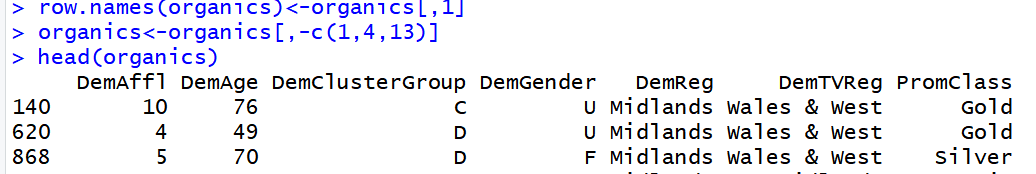
|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Model Role** | **Data Type** | **Description** |
| ID | ID | Categoric | Customer loyalty identification number |
| DemAffl | Input | Numeric | Affluence grade on a scale from 1 to 30 |
| DemAge | Input | Numeric | Age, in years |
| DemCluster | Rejected | Categoric | Type of residential neighborhood |
| DemClusterGroup | Input | Categoric | Neighborhood group |
| DemGender | Input | Categoric | M = male, F = female, U = unknown |
| DemRegion | Input | Categoric | Geographic region |
| DemTVReg | Input | Categoric | Television region |
| PromClass | Input | Categoric | Loyalty status: tin, silver, gold, or platinum |
| PromSpend | Input | Numeric | Total amount spent |
| PromTime | Input | Numeric | Time as loyalty card member |
| TargetBuy | Target | Numeric | Organics purchased? 1 = Yes, 0 = No |
| TargetAmt | Rejected | Numeric | Number of organic products purchased |
| Although two target variables are listed, these exercises concentrate on the binary target variable TargetBuy. | | | |

**Data cleaning and missing value imputation:**

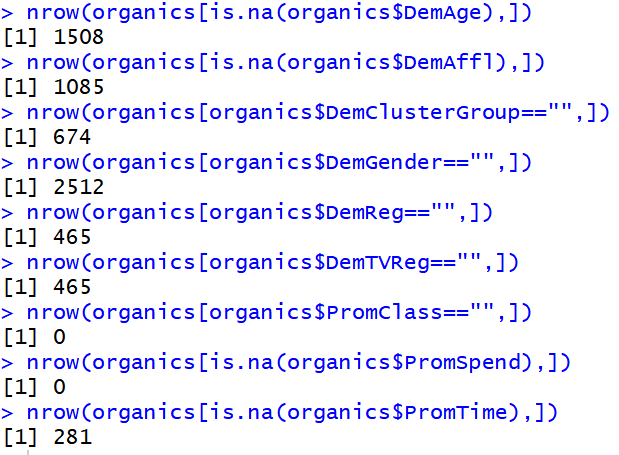
Install packages “rpart” and “rpart.plot”. Import required libraries. Import the data file organics.csv and set seed to 42.



Set row names as the ID for organics i.e. column 1. Then remove the variables- ID, DemCluster and Target Amount i.e columns 1, 4 and 13 respectively.



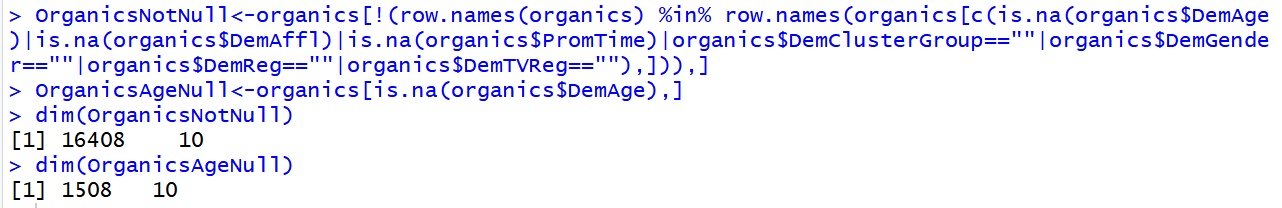
Now check the columns that have missing values or NA. Display the count of null values for each column



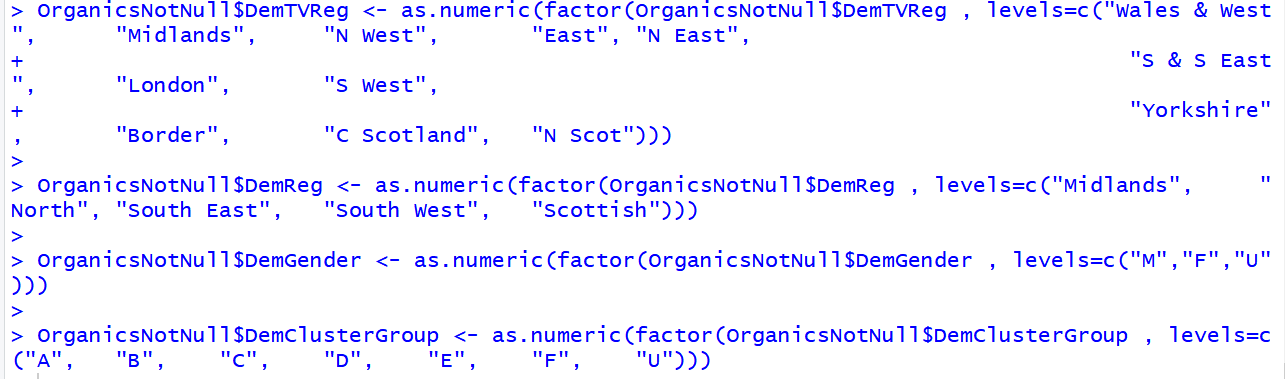
Other than PromClass and PromSpend all other variables contain null values. We need to impute values for all continuous variables using linear regression.

There are 3 variables- DemAge, DemAffl, PromTime that are continuous and whose values need to be imputed using linear regression. But before we build our regression model we need to build a dataframe for the model that does not contain rows that have missing value for any column.

So create two data frames- OrganicsNotNull that contains only the records that do not have any missing values or NA in any column and OrganicsAgeNull that contains only the records where DemAge is missing.

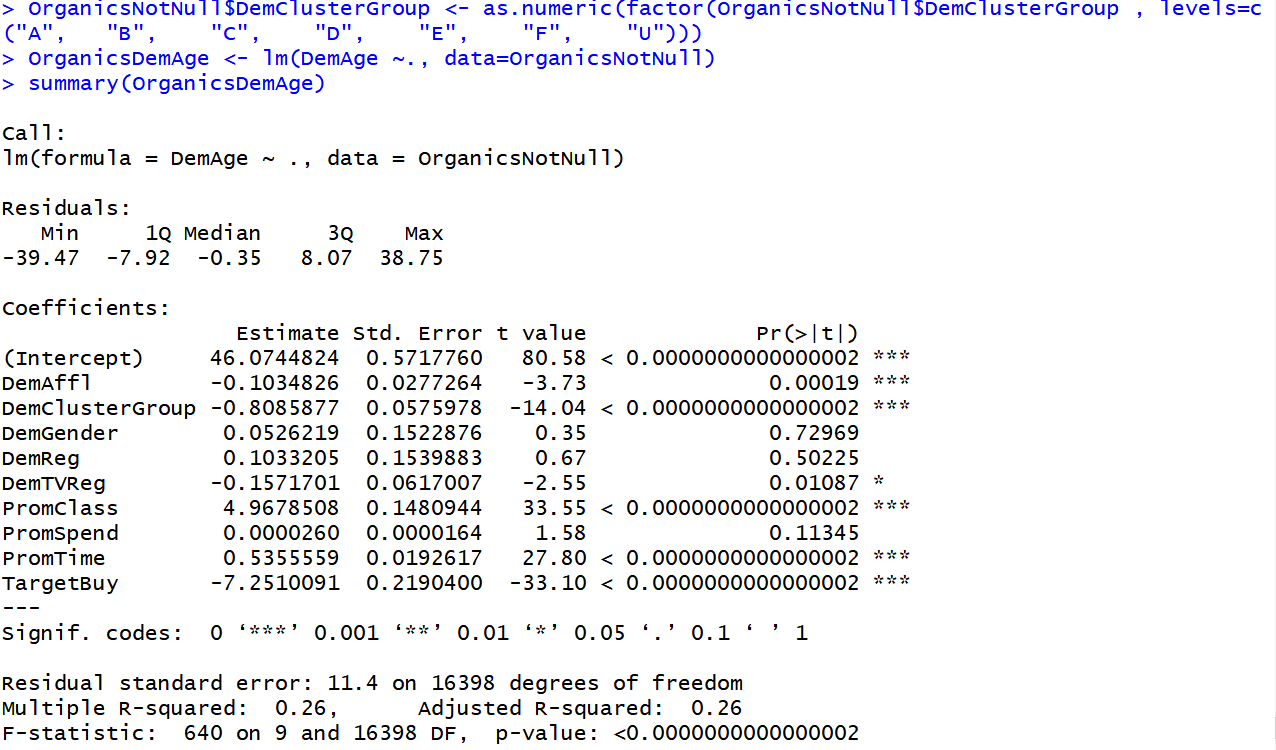


Now, to create a regression model using OrganicsNotNull we need to change all the categorical variables that have character values to numeric values by specifying levels for each.

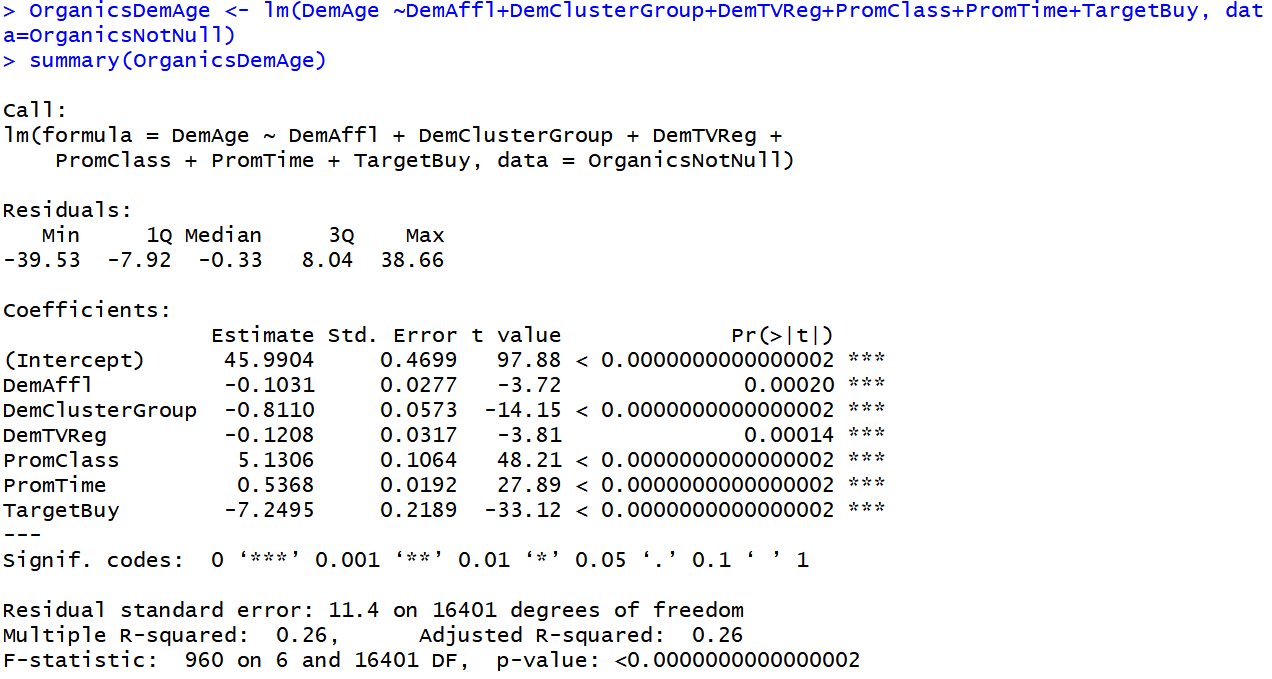


After converting to numeric values create a regression model for DemAge using all variables.

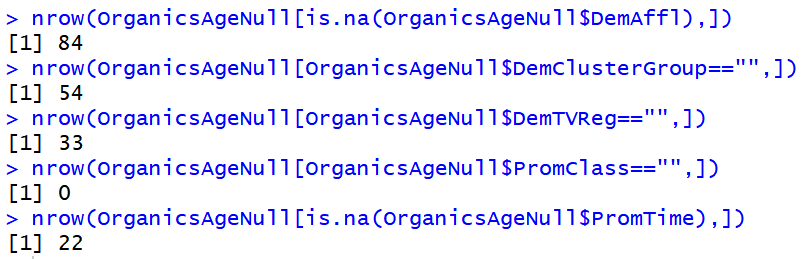
It can be seen in this model that DemAffl, DemClusterGroup, DemTVReg, PromClass, PromTime and TargetBuy are significant variables.

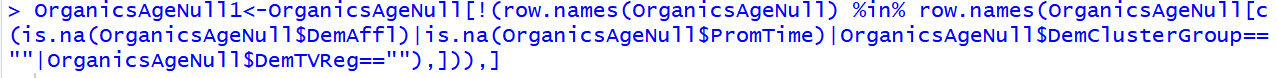


Run another regression using just the significant variables.

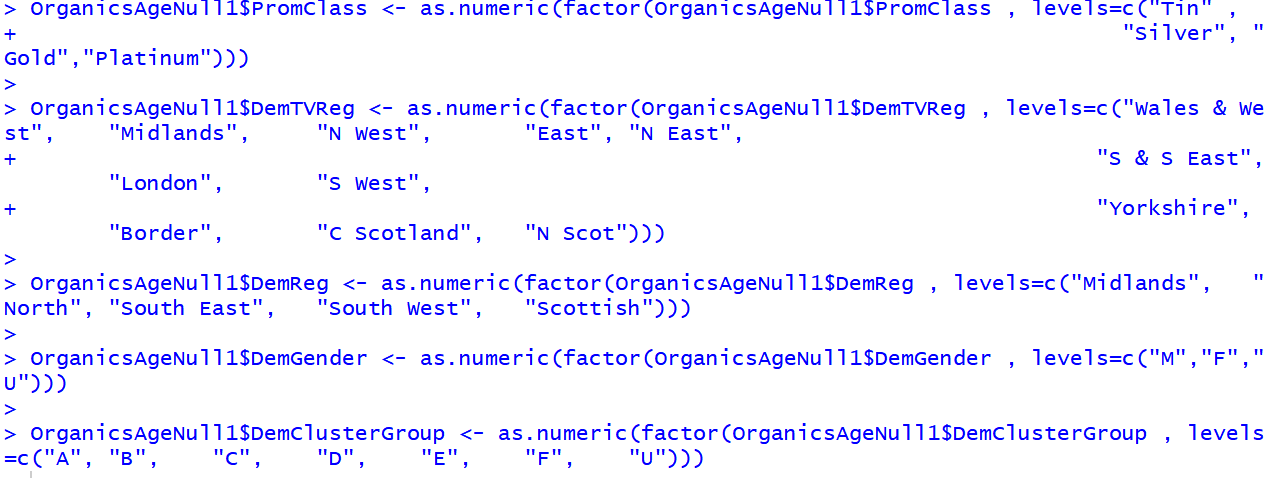


We need to predict DemAge in the data frame we defined earlier i.e. OrganicsAgeNull. However, if there are rows with missing values in columns that are used in the model we created, we cannot predict the DemAge value for these rows. So first check the columns that have missing values in OrganicsAgeNull and remove these rows. Create a new data frame OrganicsAgeNull1 where you want to predict DemAge.

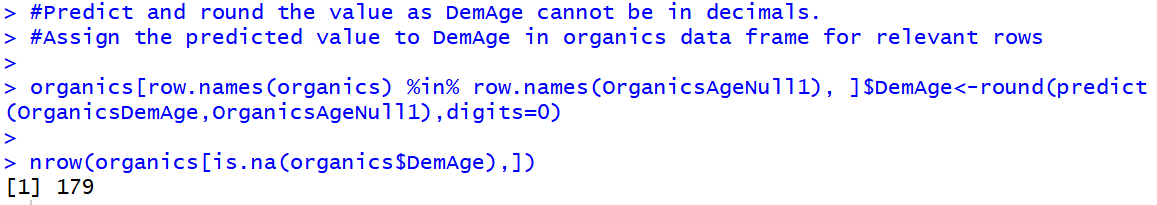




Before running prediction we first need to convert the categorical variables from characters to numeric as we did while developing the model. This is because the model was developed on predictor numeric variables and will therefore rum on numeric variables used as predictors.



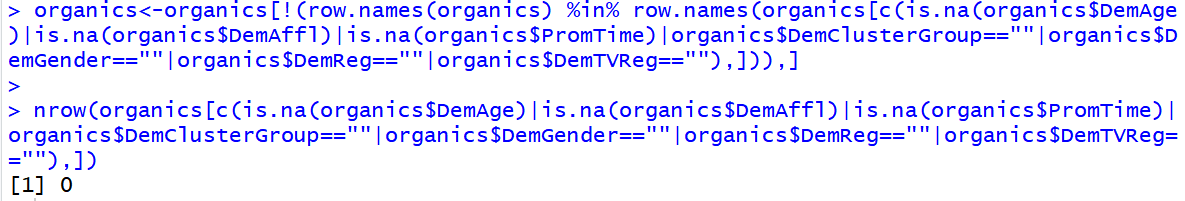
#Predict on OrganicsAgeNull1 and round the values, as DemAge cannot be in decimals. Assign the predicted value to DemAge in organics data frame to only relevant rows



As can be seen, out of 1508 missing values of DemAge 1329 missing values have been predicted and 179 values are remaining. These 179 rows are the ones that have some missing predictor variable and hence DemAge cannot be predicted.

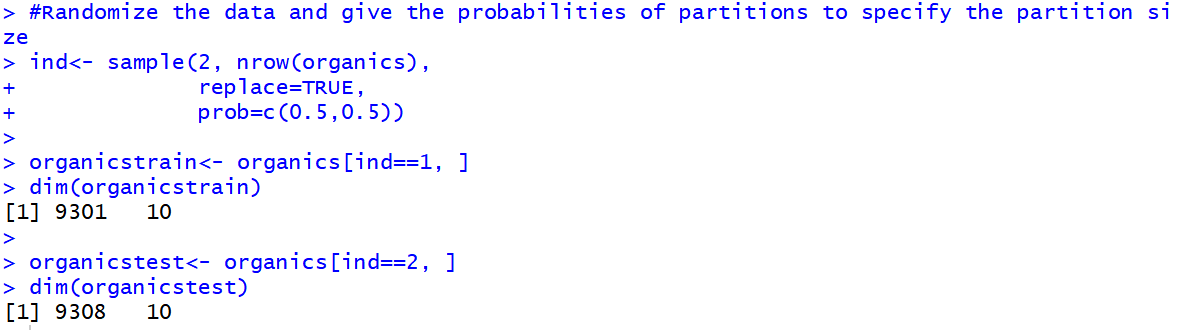
Carry out the same process for DemAffl and PromTime exactly as we did for DemAge to impute their respective values.

Since the loss of rows is minimized to an extent we can remove the remaining rows with missing values for other columns (including categorical variables) and check if there are still any missing values after doing this process.

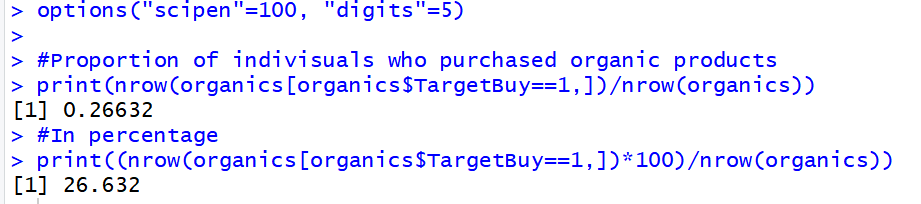


After the data cleaning and imputation is done we can now proceed to our main tasks i.e. applying classification.

1. Randomize the data and give the probabilities of partitions to specify the partition size i.e.0.5 for each.



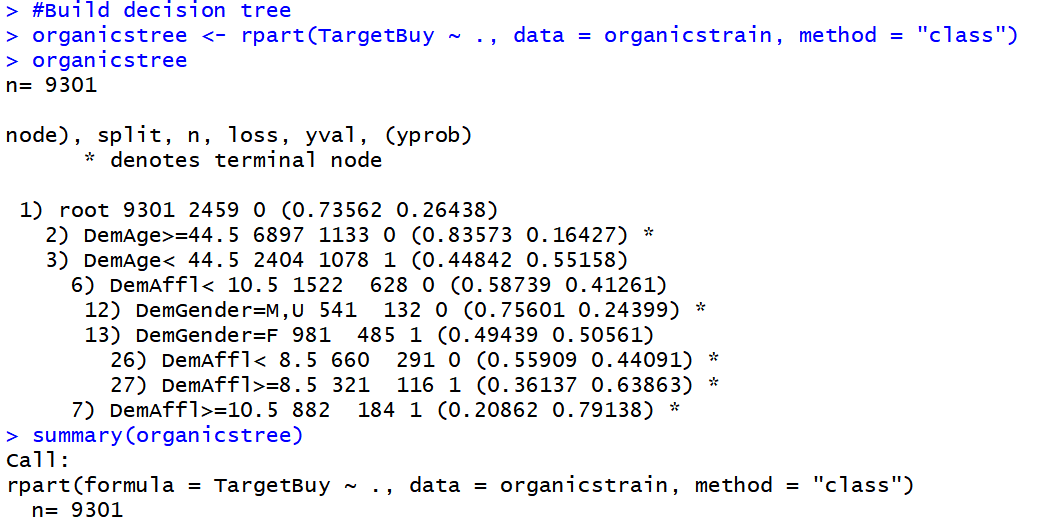
1. Use scipen to remove scientific notation from any values so it does not appear in terms of “e” i.e. exponent of 10. Then examine the distribution of the target variables. The proportion of individuals who purchased organic products is 0.26632 i.e. 26.632%



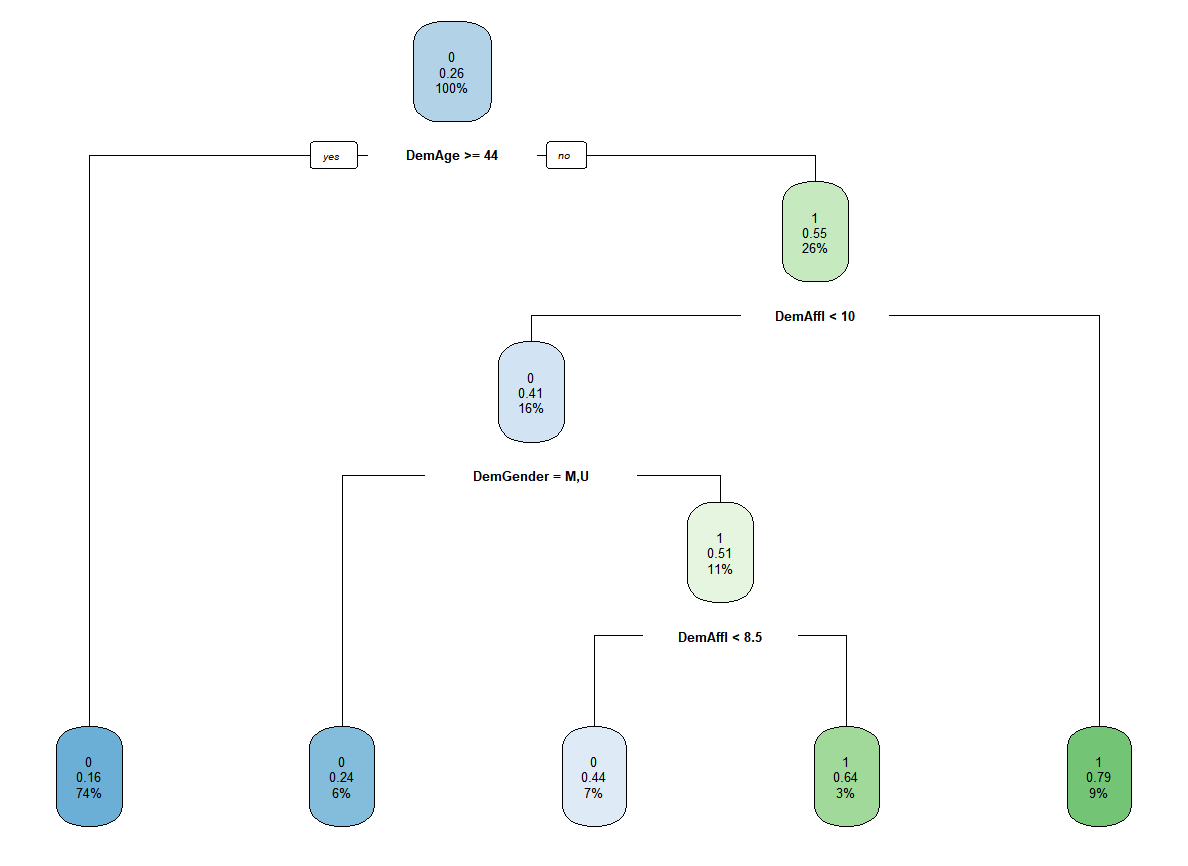
1. Only TargetBuy will be used for this analysis and should have a role of Target. This is because in the Assignment question it is specifically stated that “The supermarket's management wants to determine which customers are likely to purchase these products”. This means that they just want to know if the product will be purchased(1) or not(0). So the result should be a binary variable i.e TargetBuy and therefore can be predicted using binary decision tree classification or logistic regression.

TargetAmt should not be used as an input for a model used to predict TargetBuy because these two are directly related. TargetAmt=0 means TargetBuy=0 and TargetAmt=1,2,3 means TargetBuy=1.

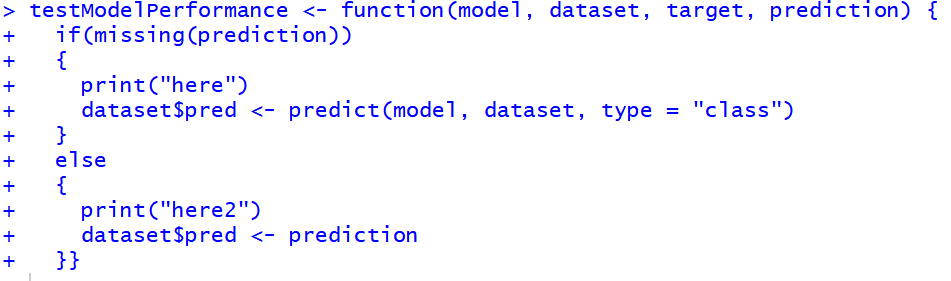
1. Seed is already set to 42. Implementing decision tree on training data.



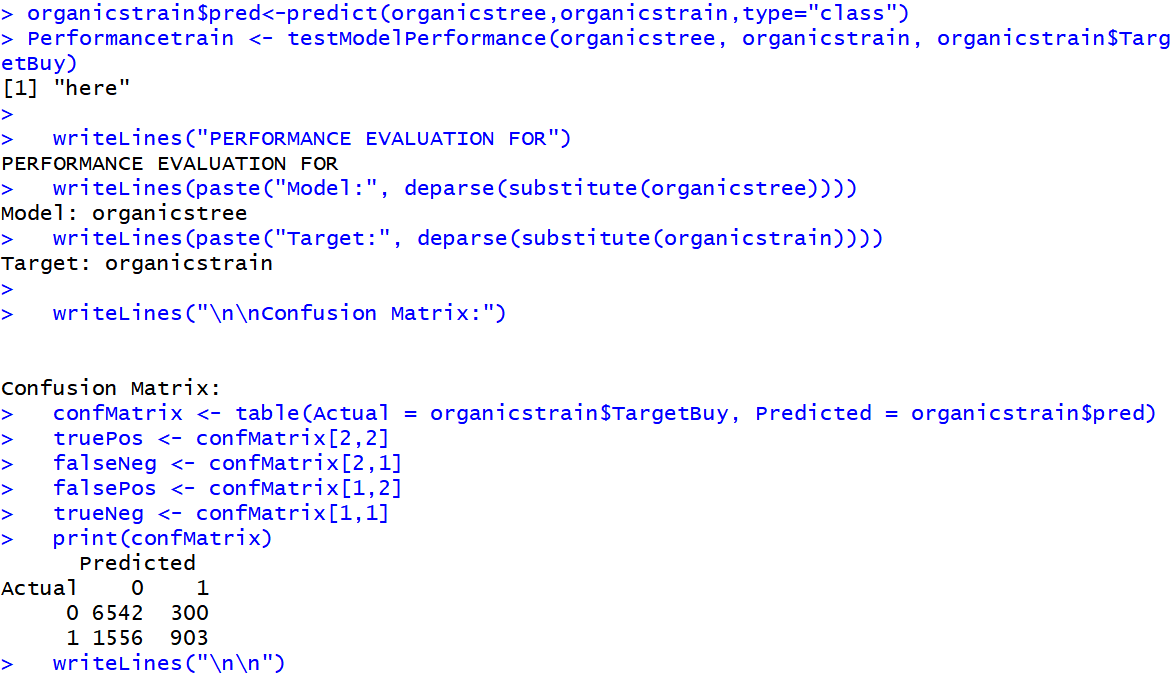
Next plot tree



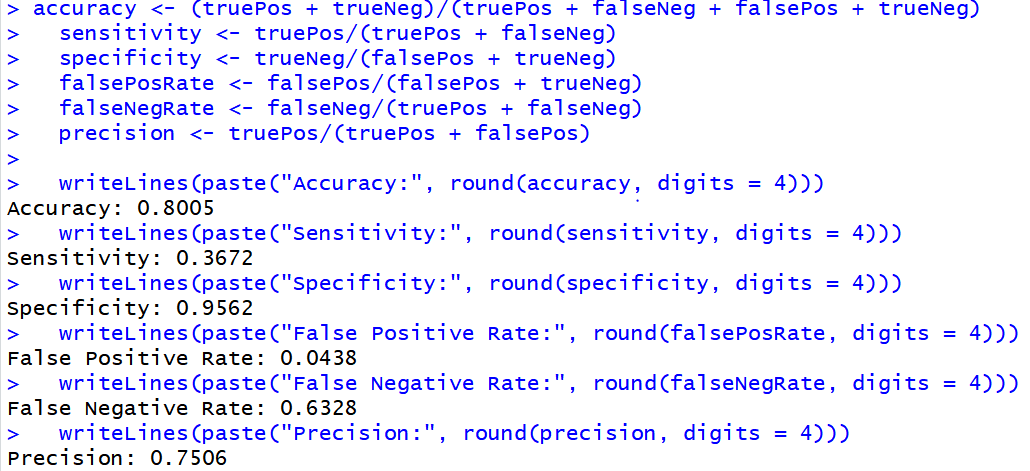
1. As seen above, there are 5 leaves
2. The first split as in the root node is done on DemAge
3. To create a 2x2 confusion matrix first create the testModelPerformance function.



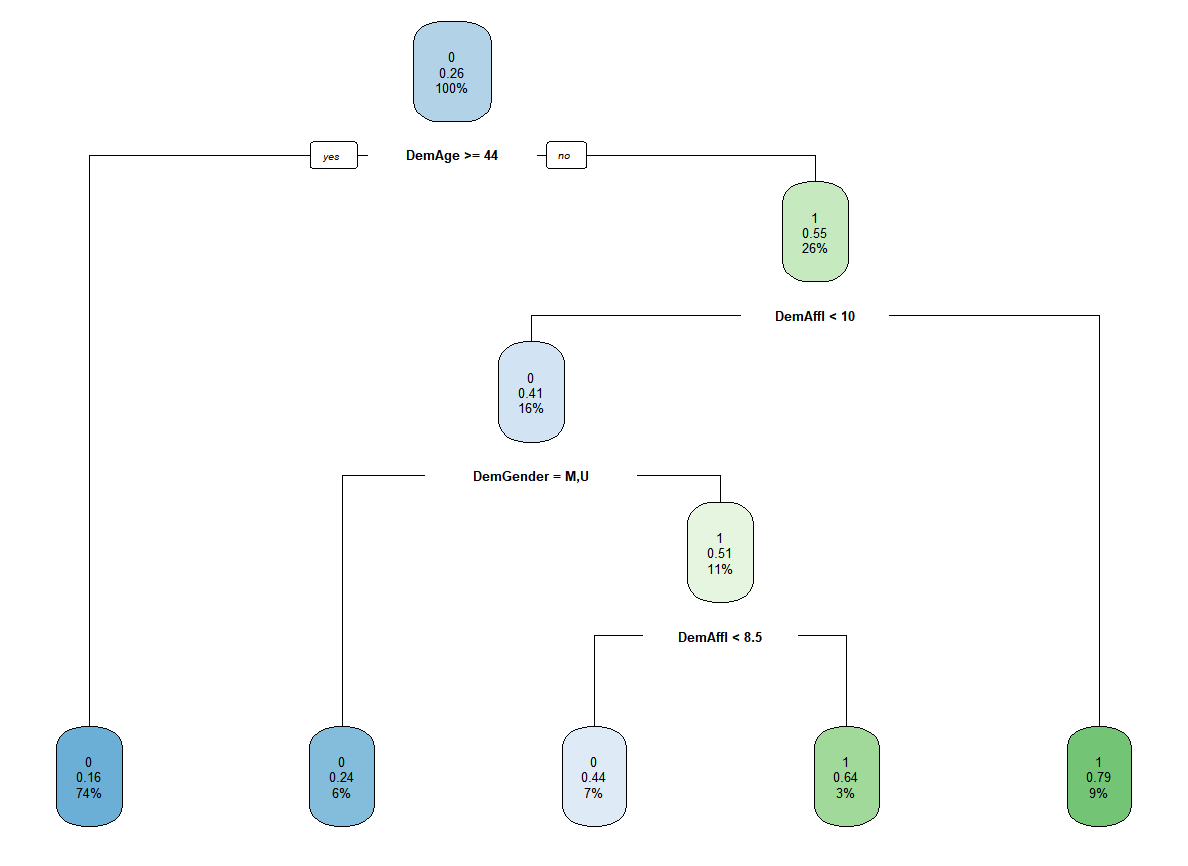
Now, use this function and build the confusion matrix



Now find the metrics like accuracy, sensitivity, specificity, precision etc.



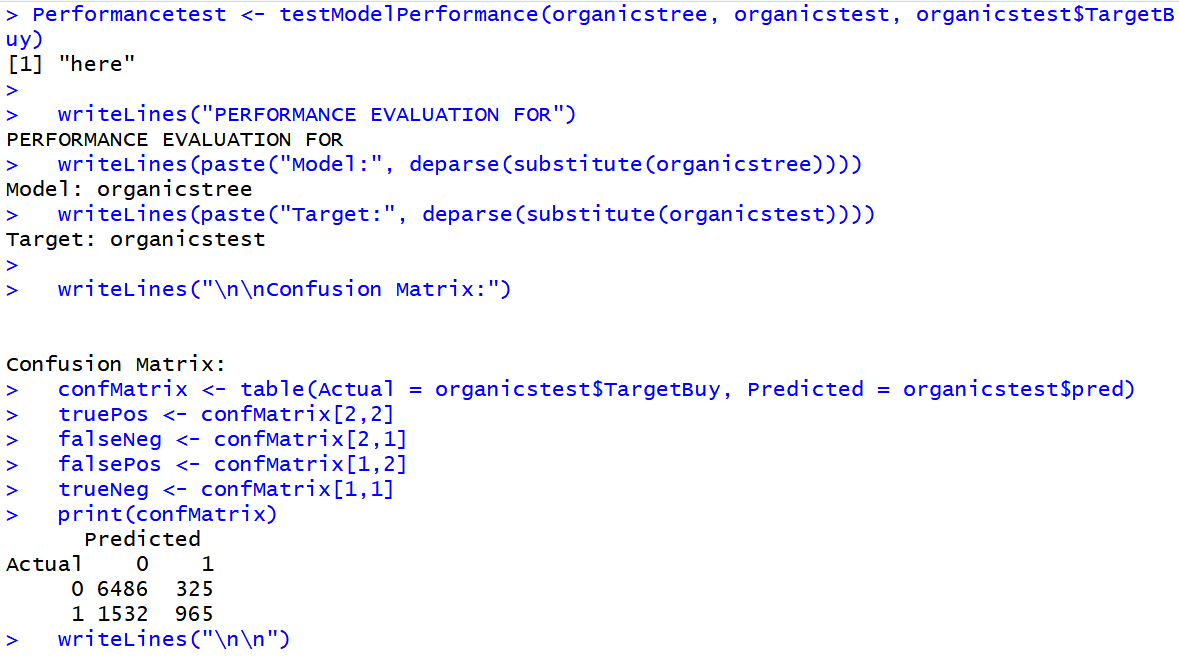
1. The final decision tree screenshot is as below



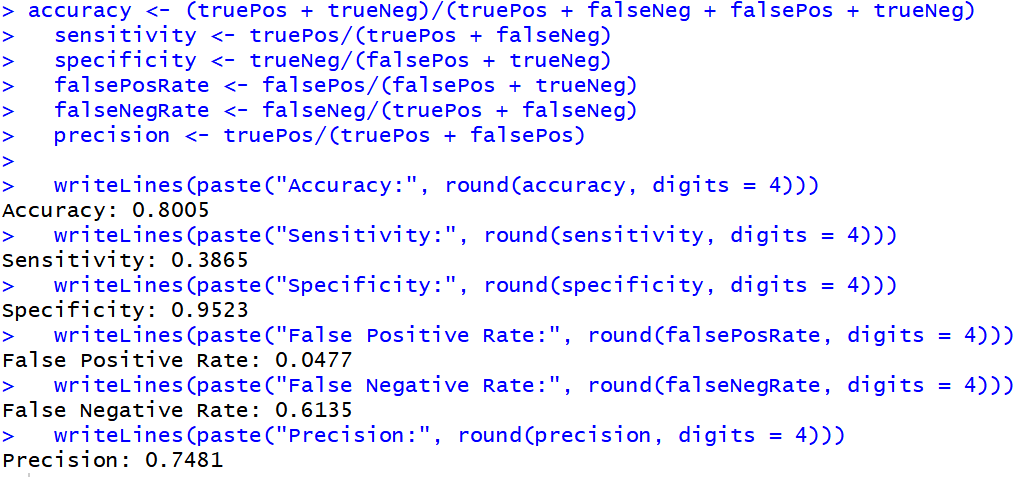
1. Applying classification model from the training data set to the test data.



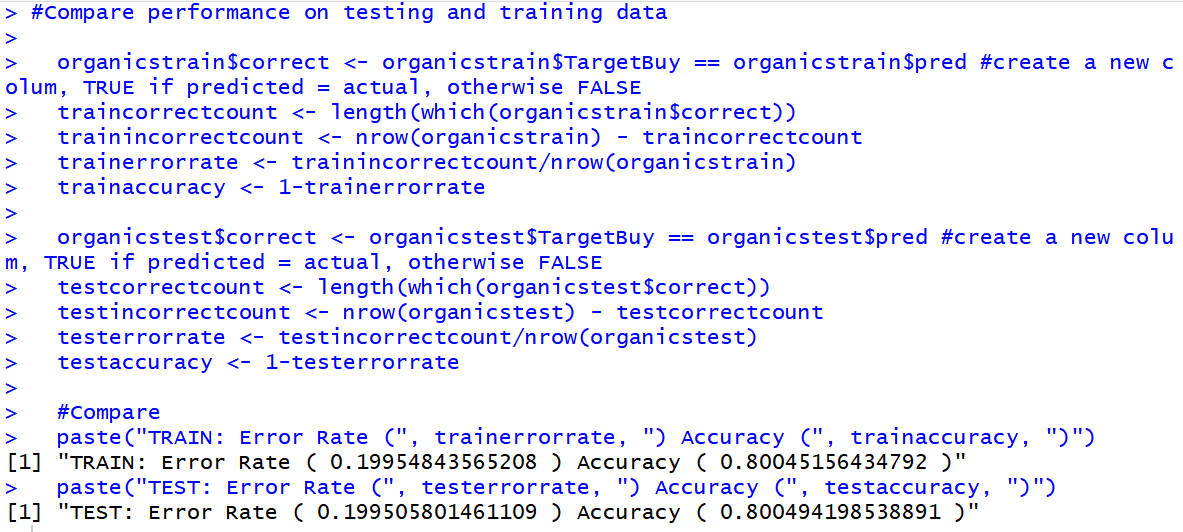
Then build the confusion matrix.



Now find the metrics like accuracy, sensitivity, specificity, precision etc.



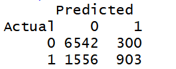
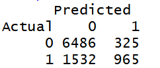
1. Compare the accuracy of classification of your test and training data sets using the decision tree classification approach.



The error rate and the accuracy are approximately same for training and test data.

The confusion matrix for both test and train data predictions.

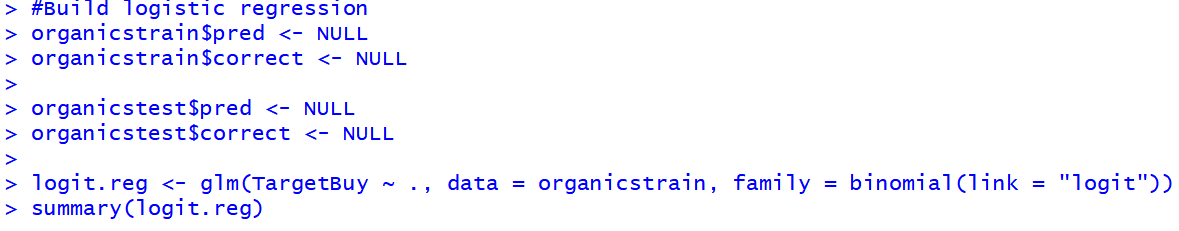
Training data Test data

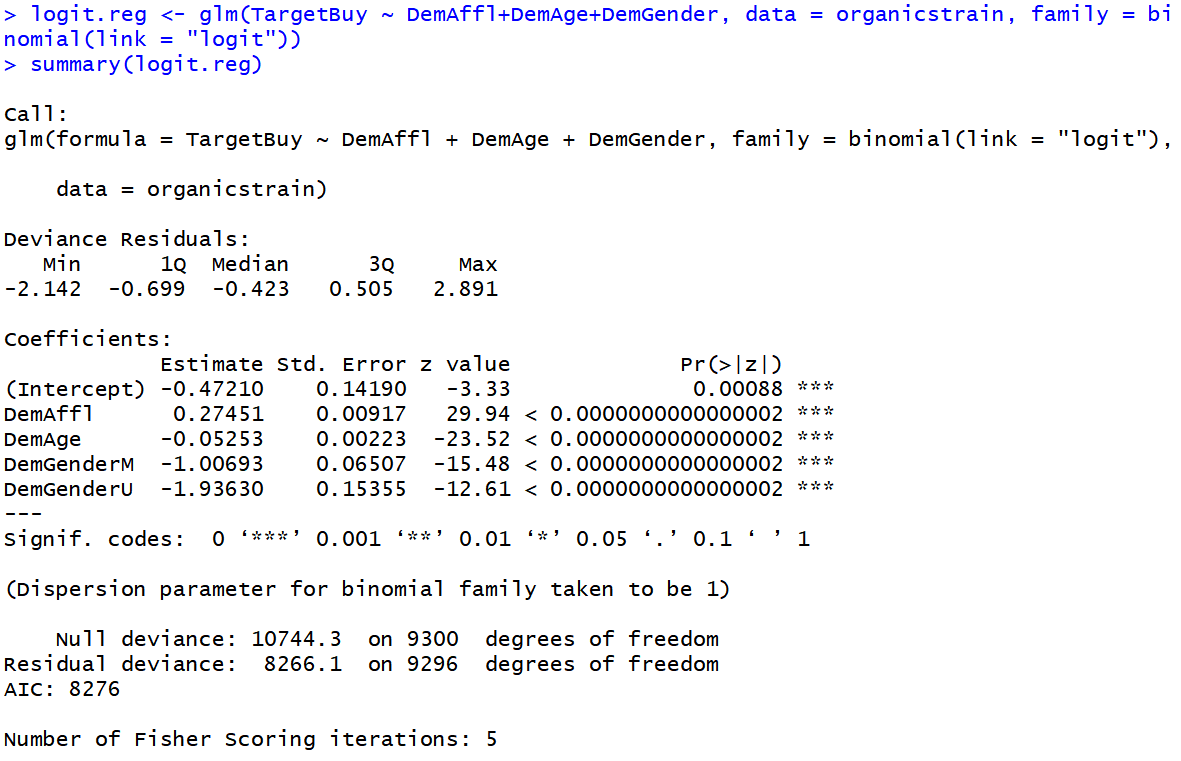
Below is the metrics comparison table:

|  |  |  |
| --- | --- | --- |
| **Metrics** | **Training data** | **Test data** |
| *Accuracy* | *0.8005* | *0.8005* |
| *Error Rate* | *0.1995* | *0.1995* |
| *Sensitivity* | *0.3672* | *0.3865* |
| *Specificity* | *0.9562* | *0.9523* |
| *False Positive Rate* | *0.0438* | *0.0477* |
| *False Negative Rate* | *0.6328* | *0.6135* |
| *Precision* | *0.7506* | *0.7481* |

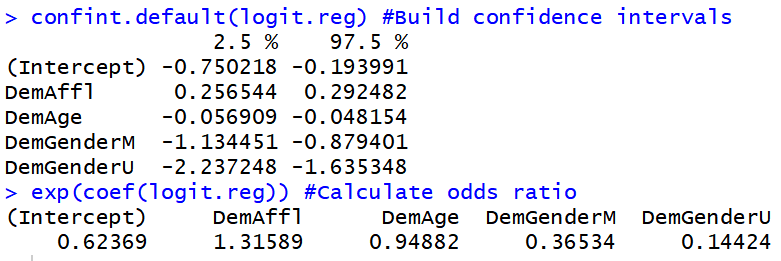
1. Build a logistic regression model for classification of the dataset.



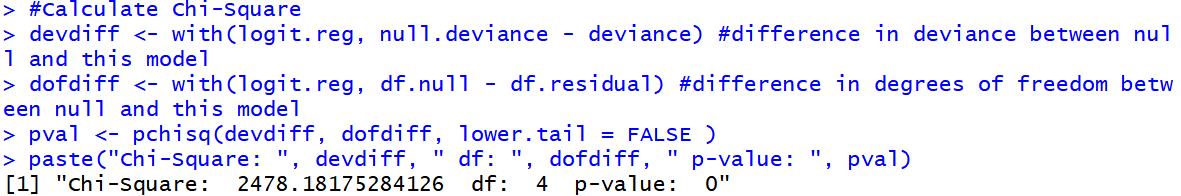
Use only the significant variables to build a new model



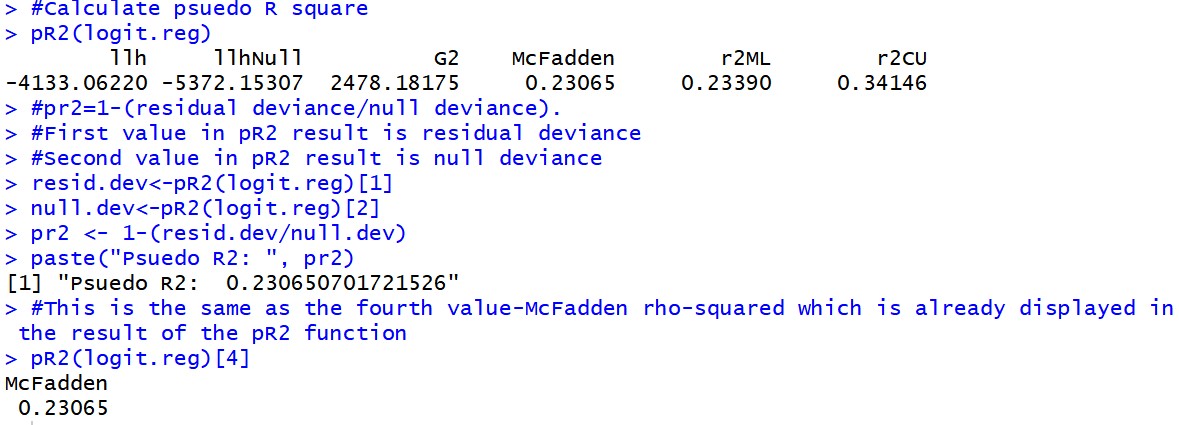
Build confidence intervals and calculate the odds ratio for this model: the odds ratio represents how the odds of the event occurring change with a 1 unit increase in that variable, all other things being equal. Here, the event is TargetBuy.



Calculate Chi-Square value:

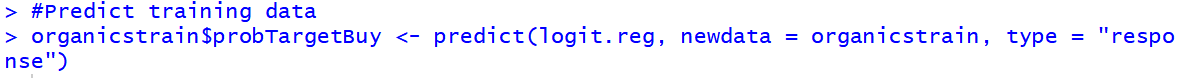


Calculate pseudo R square for the model:



Pseudo R squared is 0.23065. A value of pseudo R squared between 0.2 and 0.4 is considered good.

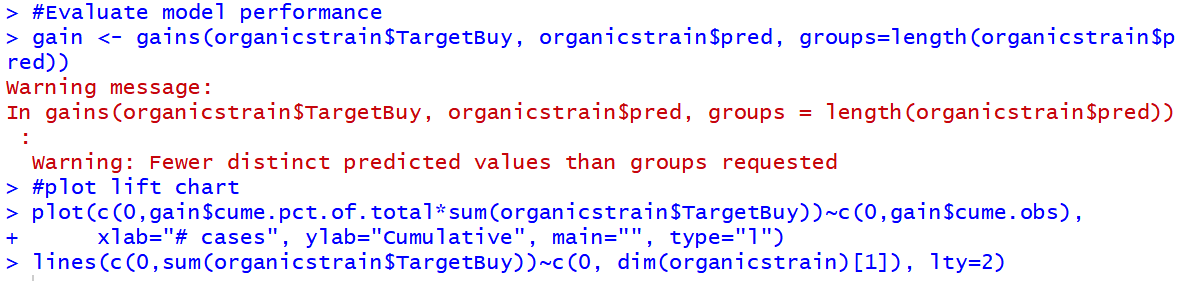
Apply prediction on the training data

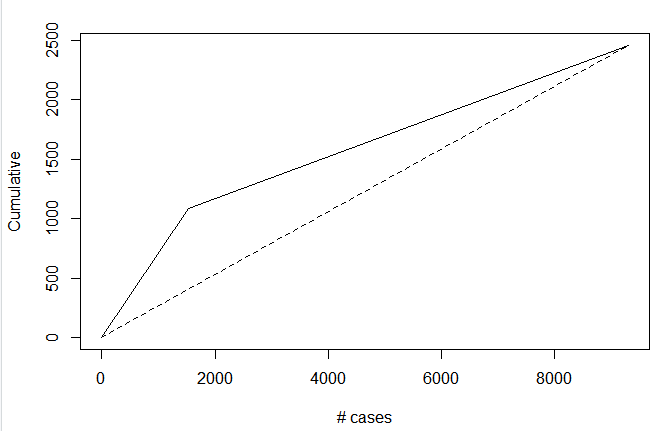


Convert probability in to a 0 or 1 prediction by rounding (cutoff = 0.5)

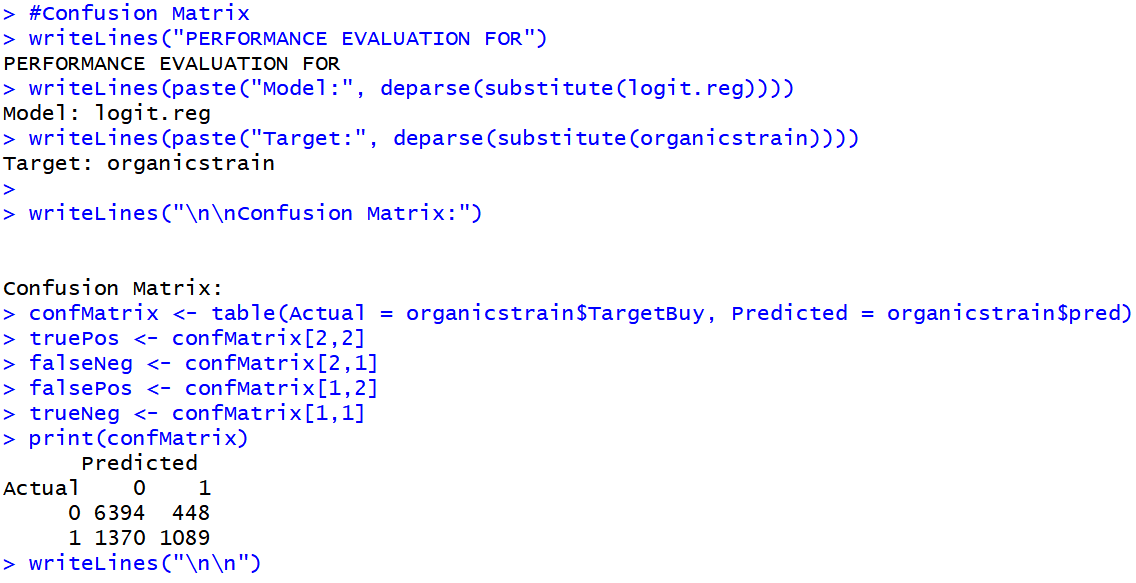


Evaluate performance of the model. Plot the lift chart on training data.

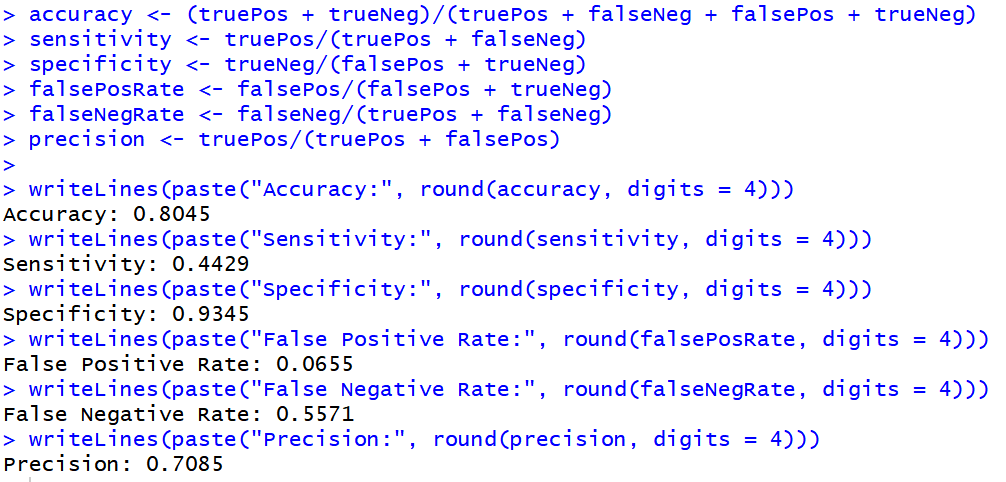




Next build the confusion matrix with prediction on training data



Calculate the metrics using the confusion matrix on the predicted training data



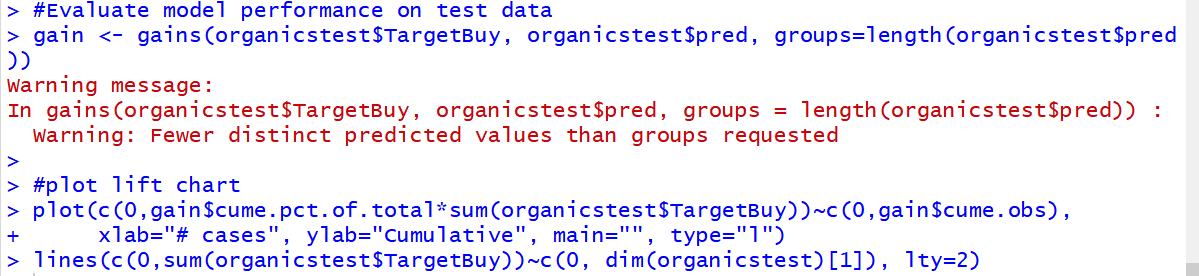
Apply prediction on the test data

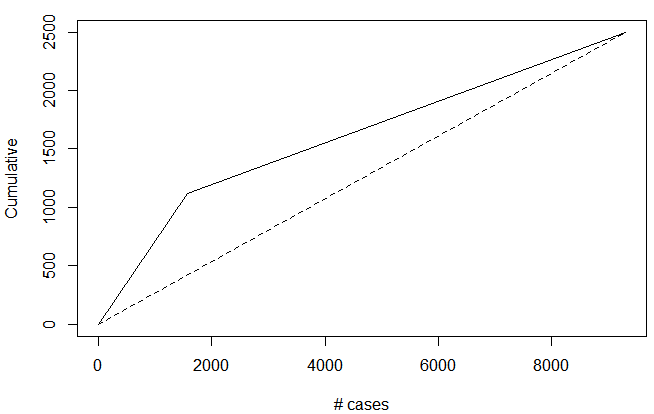


Convert probability in to a 0 or 1 prediction by rounding (cutoff = 0.5)

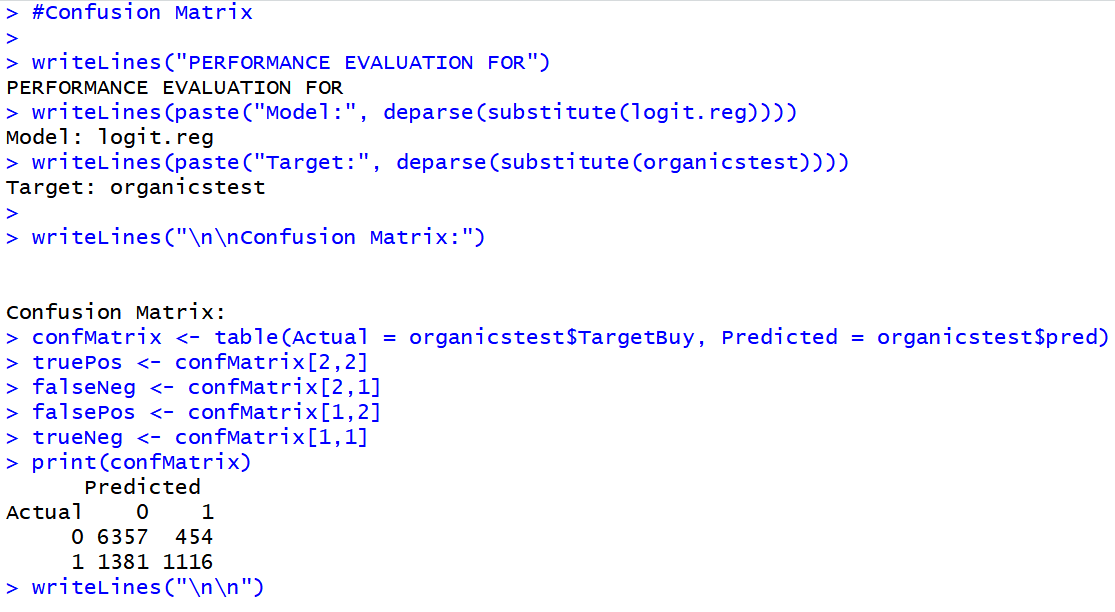


Evaluate performance of the model. Plot the lift chart on test data. For a given number of records (x-axis), the lift curve value on the y-axis tells us how much better we are doing compared to random assignment.

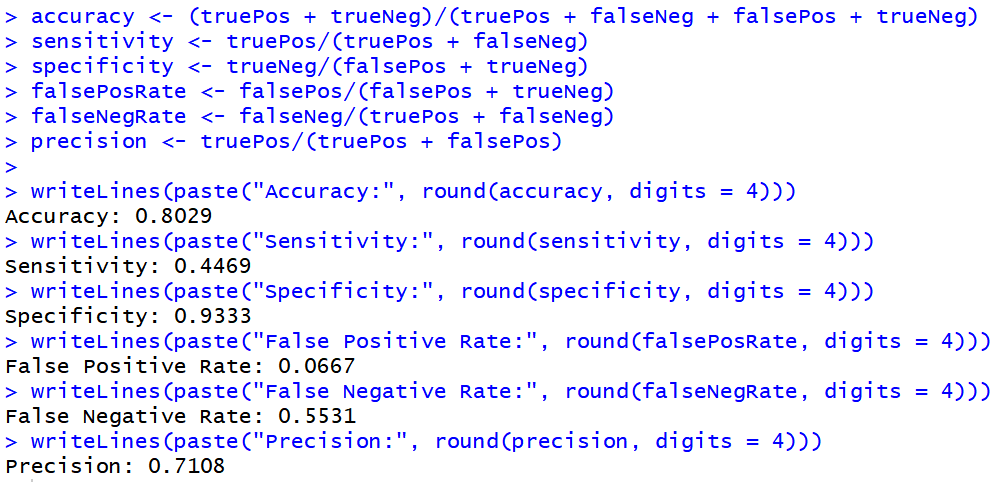




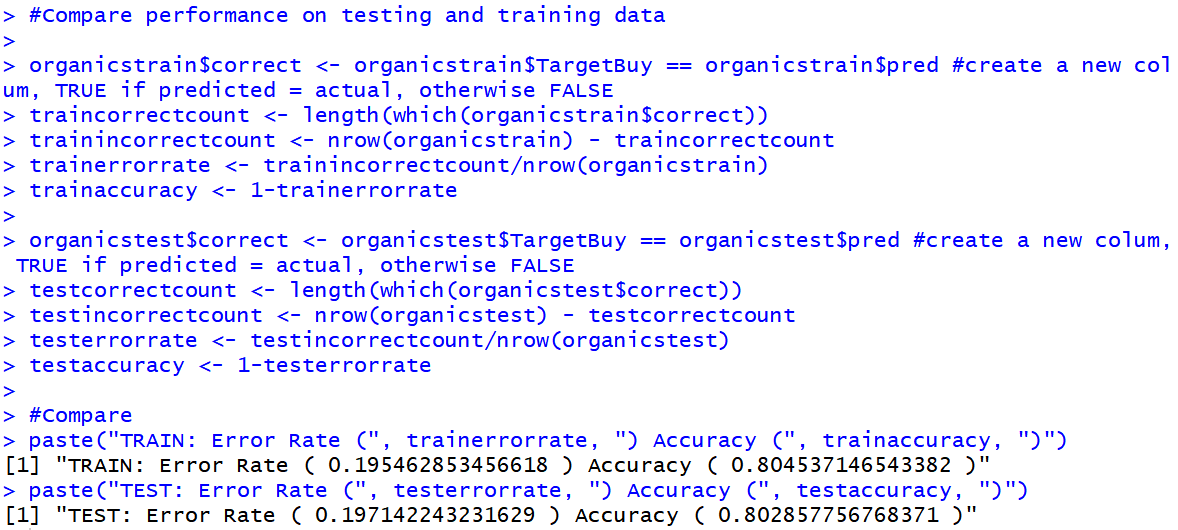
Next build the confusion matrix with prediction on test data



Calculate the metrics using the confusion matrix on the predicted test data



1. Compare performance of the logit prediction models on your test and training data sets.



Below is the metrics comparison table for prediction on training and test data using logit:

|  |  |  |
| --- | --- | --- |
| **Metrics** | **Training data** | **Test data** |
| *Accuracy* | *0.8045* | *0.8029* |
| *Error Rate* | *0.1955* | *0.1971* |
| *Sensitivity* | *0.4429* | *0.4469* |
| *Specificity* | *0.9345* | *0.9333* |
| *False Positive Rate* | *0.0655* | *0.0667* |
| *False Negative Rate* | *0.5571* | *0.5531* |
| *Precision* | *0.7085* | *0.7108* |

1. Comparison of metrics from Decision tree prediction and logit prediction on both training and test dataset.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Metrics** | **Decision** | **tree** | **Logistic** | **regression** |
| **Metrics** | **Training data** | **Test data** | **Training data** | **Test data** |
| *Accuracy* | *0.8005* | *0.8005* | *0.8045* | *0.8029* |
| *Error Rate* | *0.1995* | *0.1995* | *0.1955* | *0.1971* |
| *Sensitivity* | *0.3672* | *0.3865* | *0.4429* | *0.4469* |
| *Specificity* | *0.9562* | *0.9523* | *0.9345* | *0.9333* |
| *False Positive Rate* | *0.0438* | *0.0477* | *0.0655* | *0.0667* |
| *False Negative Rate* | *0.6328* | *0.6135* | *0.5571* | *0.5531* |
| *Precision* | *0.7506* | *0.7481* | *0.7085* | *0.7108* |

From the above comparison of the two classification techniques deployed, we can see that the accuracy of the logistic regression and decision tree is approximately the same. However, the Specificity of logistic regression is higher than that of the decision tree.

The observed differences are the false positive rate of the logistic regression is more than that of the decision tree. However, the false negative rate of logit is less than that of the decision tree. The sensitivity of the logit is higher but the specificity is lower than that of the decision tree. However, both are cases of low sensitivity and very high specificity. This means that they may overlook a positive, but they will rarely classify an actual negative as a positive.