AD699

Data Mining for Business Analytics

Spring 2021

Professor Greg Page

Assignment 4

Kunfei Chen

U15575304

Classification Tree

Task 1

The data files “telecom_users.csv” are downloaded from the class blackboard site and imported to R-studio for analysis.

Task 2

The variable Churn is converted into a factor (See Figure 1).

Figure 1

```
$ Churn           : Factor w/ 2 levels "No","Yes": 1 1 2 1 1 1 1 1 1 1 .
```

Task 3

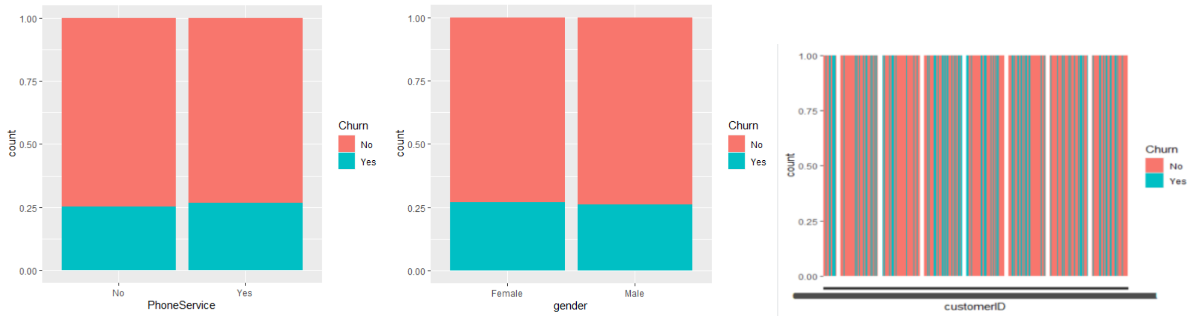
Generally, the variables in the dataset can be classified into two types: categorical variables and numeric variables.

As for the categorical variables, the “chi-square test” is applied to identify if a categorical variable is corelated to the variable Churn. Different variables’ p value of chi-square test is shown in Figure 2. It is seen that the p value of “Phone Service”, “Customer ID”, and “gender” are relatively high, which means there is no high co-relation between Churn and these variables. It is also clearly displayed from the three variables’ bar plots that there is almost no difference of Churn variables proportion when these variables change (See Figure 3). Since there are so many unique values for “Customer ID” variable, it does not have any predictive value and thus being removed.

Figure 2

```
> chisq_df
                 chisq_p_value
PhoneService      4.966298e-01
customerID        4.939231e-01
gender            4.780481e-01
MultipleLines     1.914407e-02
Partner           9.151167e-30
SeniorCitizen     5.640303e-31
Dependents        4.150645e-35
PaperlessBilling  3.124100e-48
StreamingTV       1.575655e-67
StreamingMovies   8.160493e-68
DeviceProtection 2.808213e-102
OnlineBackup     3.200344e-109
PaymentMethod    3.623649e-115
InternetService  3.451875e-130
TechSupport      5.608629e-148
OnlineSecurity   3.606867e-156
Contract         1.456383e-218
```

Figure 3

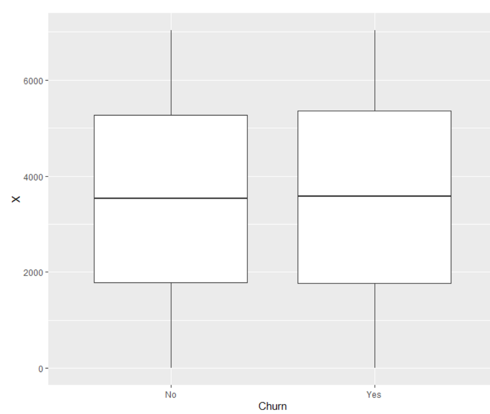


As for numeric variables, the "t-test" method is used to identify if they are corelated to the variable Churn. Each variable's t-test p value is shown in Figure 4. It is clear that variable X has relatively high p value, which means it is unrelated to the variable Churn. It can also be demonstrated from the boxplot in Figure 5 that the distribution of variable X is almost the same in different Churn conditions, so this variable is removed.

Figure 4

```
> arrange(t_test_df, desc(t_test_p_value))
                t_test_p_value
X                 6.094460e-01
MonthlyCharges    1.754652e-57
TotalCharges      3.945119e-63
tenure           2.805528e-192
>
```

Figure 5



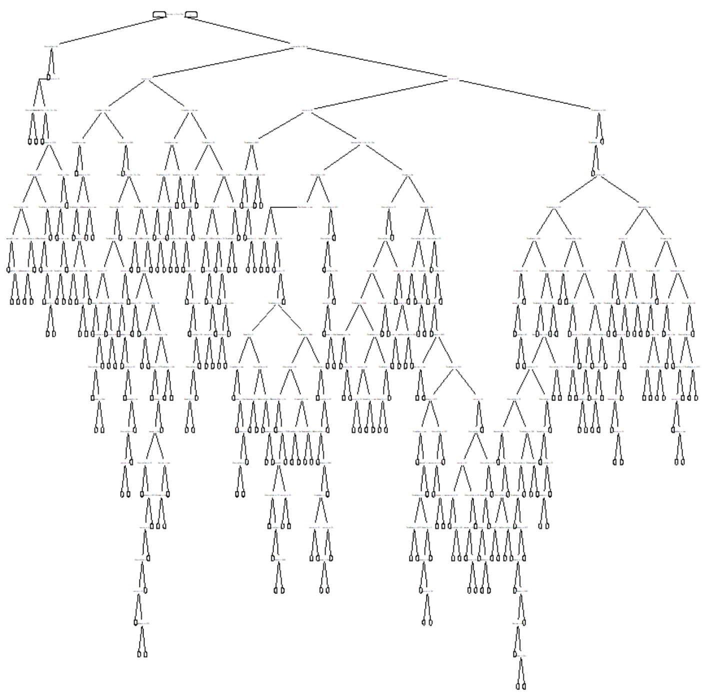
Task 4

After setting the seed value to be 30, the entire dataset are split into training set with 60 percent and validation set with 40 percent via the "sample_n ()" function.

Task 5

The largest tree is built by setting minsplit equals 2, minbucket equals 1, maxdepth equals 30 and cp equals 0.001. The plot of this tree model is shown in Figure 6.

Figure 6



According to the results of confusion matrix, the accuracy of this model on training set is 92.12% and the accuracy on validation set is 77.44% (See Figure 7).

Figure 7

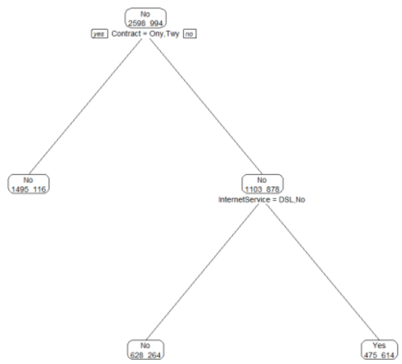
```
Confusion Matrix and Statistics

          Reference
Prediction   No  Yes
       No  2522  207
       Yes   76  787

               Accuracy : 0.9212
                 95% CI : (0.9119, 0.9298)
    No Information Rate : 0.7233
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.7948

 Mcnemar's Test P-Value : 1.095e-14

            Sensitivity : 0.9707
            Specificity : 0.7918
         Pos Pred Value : 0.9241
         Neg Pred Value : 0.9119
             Prevalence : 0.7233
         Detection Rate : 0.7021
   Detection Prevalence : 0.7597
      Balanced Accuracy : 0.8812

       'Positive' Class : No
```

```
Confusion Matrix and Statistics

          Reference
Prediction   No  Yes
       No  1549  288
       Yes  252  305

               Accuracy : 0.7744
                 95% CI : (0.7572, 0.791)
    No Information Rate : 0.7523
    P-Value [Acc > NIR] : 0.006083

                  Kappa : 0.3822

 Mcnemar's Test P-Value : 0.132026

            Sensitivity : 0.8601
            Specificity : 0.5143
         Pos Pred Value : 0.8432
         Neg Pred Value : 0.5476
             Prevalence : 0.7523
         Detection Rate : 0.6470
   Detection Prevalence : 0.7673
      Balanced Accuracy : 0.6872

       'Positive' Class : No
```

Task 6

A very small tree is built by setting cp equals 0.01 and max depth equals 2. The plot of this tree model is shown in Figure 8.

Figure 8



According to the results of confusion matrix, the accuracy of this model on training set is 76.2% and the accuracy on validation set is 75.69% (See Figure 9).

Figure 9

```
Confusion Matrix and Statistics

          Reference
Prediction   No  Yes
       No  2123  380
       Yes  475  614

               Accuracy : 0.762
                 95% CI : (0.7477, 0.7758)
    No Information Rate : 0.7233
    P-Value [Acc > NIR] : 7.906e-08

                  Kappa : 0.4224

 Mcnemar's Test P-Value : 0.001306

            Sensitivity : 0.8172
            Specificity : 0.6177
         Pos Pred Value : 0.8482
         Neg Pred Value : 0.5638
             Prevalence : 0.7233
         Detection Rate : 0.5910
   Detection Prevalence : 0.6968
      Balanced Accuracy : 0.7174

       'Positive' Class : No
```

```
Confusion Matrix and Statistics

          Reference
Prediction   No  Yes
       No  1456  237
       Yes  345  356

               Accuracy : 0.7569
                 95% CI : (0.7392, 0.774)
    No Information Rate : 0.7523
    P-Value [Acc > NIR] : 0.3106

                  Kappa : 0.3852

 Mcnemar's Test P-Value : 9.195e-06

            Sensitivity : 0.8084
            Specificity : 0.6003
         Pos Pred Value : 0.8600
         Neg Pred Value : 0.5078
             Prevalence : 0.7523
         Detection Rate : 0.6082
   Detection Prevalence : 0.7072
      Balanced Accuracy : 0.7044

       'Positive' Class : No
```

Task 7

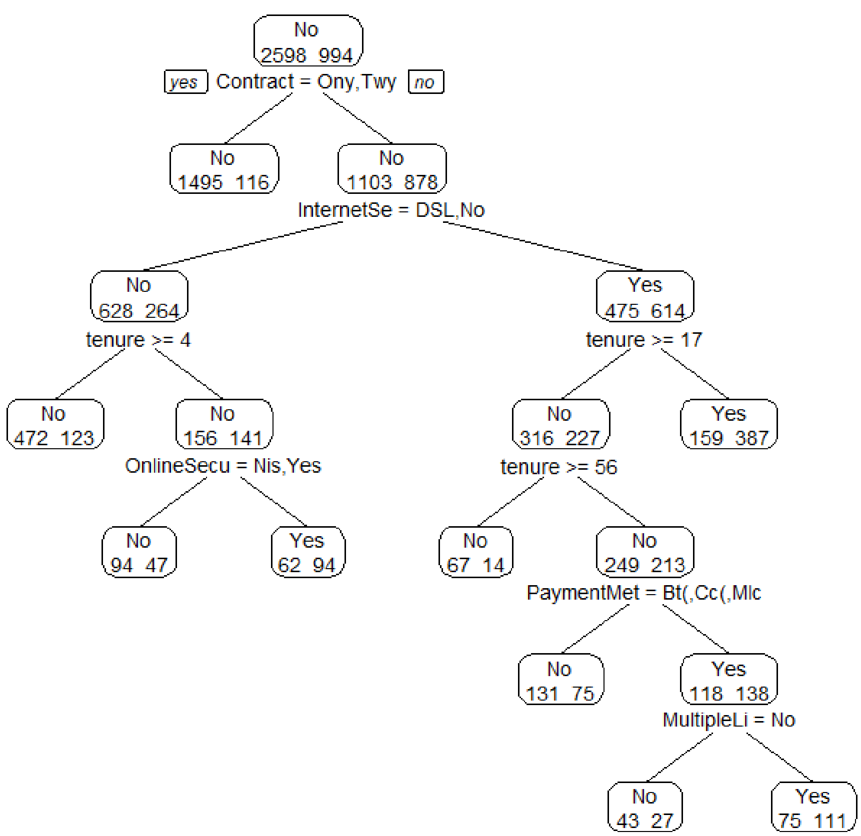
In order to build an optimally-sized tree, a temp model is built by "rpart ()" function. According to this model's complexity parameter information, the cross validation error achieves the minimum at 0.7495 when the complexity parameter equals 0.01. Therefore, the optimally-sized tree is built by setting cp equals 0.01 (See Figure 10).

Figure 10

```
> printcp(model_temp)

Classification tree:
rpart(formula = Churn ~ ., data = train, method = "class", minsplit = 1)

Variables actually used in tree construction:
[1] Contract        InternetService MultipleLines   OnlineSecurity  PaymentMethod
[6] tenure

Root node error: 994/3592 = 0.27673

n= 3592

        CP nsplit rel error  xerror     xstd
1 0.069920      0   1.00000 1.00000 0.026975
2 0.016097      3   0.77062 0.77565 0.024756
3 0.010060      5   0.73843 0.76861 0.024674
4 0.010000      8   0.70221 0.74950 0.024447
```

The plot of this tree model is shown in Figure 11.

Figure 11



According to the results of confusion matrix, the accuracy of this model on training set is 80.57% and the accuracy on validation set is 79.28% (See Figure 12).

Figure 12

```
Confusion Matrix and Statistics

          Reference
Prediction   No  Yes
       No  2302  402
       Yes  296  592

               Accuracy : 0.8057
                 95% CI : (0.7924, 0.8185)
    No Information Rate : 0.7233
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.498

 Mcnemar's Test P-Value : 7.058e-05

            Sensitivity : 0.8861
            Specificity : 0.5956
         Pos Pred Value : 0.8513
         Neg Pred Value : 0.6667
             Prevalence : 0.7233
         Detection Rate : 0.6409
   Detection Prevalence : 0.7528
      Balanced Accuracy : 0.7408

       'Positive' Class : No
```

```
Confusion Matrix and Statistics

          Reference
Prediction   No  Yes
       No  1562  257
       Yes  239  336

               Accuracy : 0.7928
                 95% CI : (0.776, 0.8089)
    No Information Rate : 0.7523
    P-Value [Acc > NIR] : 1.616e-06

                  Kappa : 0.4384

 Mcnemar's Test P-Value : 0.4453

            Sensitivity : 0.8673
            Specificity : 0.5666
         Pos Pred Value : 0.8587
         Neg Pred Value : 0.5843
             Prevalence : 0.7523
         Detection Rate : 0.6525
   Detection Prevalence : 0.7598
      Balanced Accuracy : 0.7170

       'Positive' Class : No
```

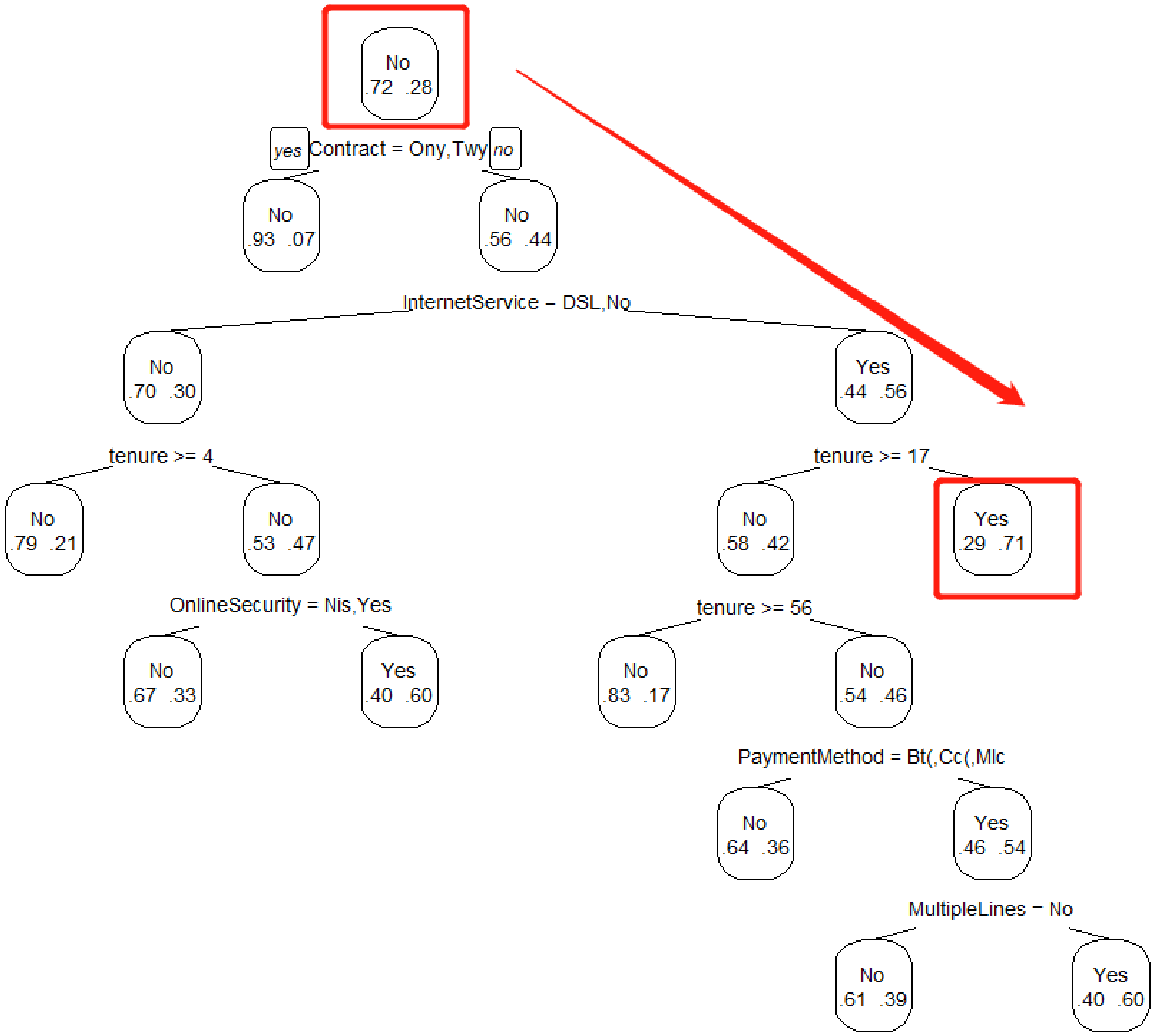
Task 8

To analyze the accuracy of the three build models, it can be found the tree model will have better performance on training set with more complexity in larger size. However, this pattern dose not apply to validation set. Although the largest tree model achieves the highest accuracy for training set at 92.12%, its performance on validation set is only 77.44%, which is just a little bit higher than that of small tree model. Instead, the optimally-sized tree model achieves the best performance on validation set with 79% accuracy. The reasons of such a result might be that the largest model is overfitted, which results in high bias, whereas for the smallest tree, it is underfitted, which results in high variance. Only the optimally-size tree achieves a balance between bias and variance, so as to get the highest accuracy in validation set.

Task 9

According to Figure 13, the root node of the optimally-size tree decision tree is a judgement about whether the input data's contract variable is one year or two year. Then the following layers of this decision tree will make judgement and classification according to other features. The reason why contract variable is significant as the root node is due to the fact that the Gini impurity of the dataset decrease most after split by contract variable. In other words, the contract variable has the best classification effect compared with other variables.

Figure 13



Task 10

As shown in Figure 13, to give an example of rule in this tree, one path is marked in red color. To follow this path, firstly if the input data's contract variable is not "one year" or "two year", the result will have 56% probability to be "NO", than if the Internet Service variable is "DSL" or "No", the result will have 56% probability to be "Yes". Finally, if the tenue variable is smaller than 17, the result will have 71% probability to be "Yes".

Task 11

To give an example of the terminal node in the left side of the second layer of this tree, the probability of “No” is 0.93, and that of “Yes” is 0.07. According to the formular (See Figure 14), the Gini impurity = 0.93^2 + 0.07^2 = 0.1302.

Figure 14

$$\text{Gini}(p) = \sum_{k=1}^{K} p_k(1-p_k) = 1 - \sum_{k=1}^{K} p_k^2$$

Task 12

- **a**

As shown in Figure 15, the Monthly Charges variable is distributed into 5 bins, each bin has similar number of records which is about 1500. The whole data set is than split into training set with 60% and validation set with 40%.

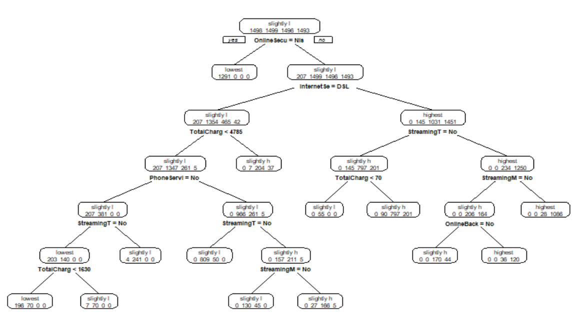
Figure 15

```
> bk <- c(-Inf,five_num[2:5])
> data_2$MonthlyCharges <- cut(data_2$MonthlyCharges, breaks=bk,
+                              labels = c("lowest","slightly low","slightly high","highest"))
> table(data_2$MonthlyCharges)

      lowest  slightly low slightly high       highest 
        1498          1499          1496          1493 
```

- **b**

The “rpart()” function is applied with default settings to build a new tree model for the new data set which consists of the newly-binned Monthly Charges variable as the outcome. The plot of tree is shown in Figure 16.

Figure 16



- **c**

According to the results of confusion matrix, the accuracy of this model on training set is 89.03% and the accuracy on validation set is 89.27%.

- **d**

A new data set is generated with the newly-binned Monthly Charges variable in which all the bins have unbalanced records (See Figure 17).

Figure 17

```
> table(data_3$MonthlyCharges)

       lowest   slightly low slightly high        highest
           29            968          4932             57
```

- **e**

A new tree model is created based on the new dataset.

- **f**

According to the results of confusion matrix, the accuracy of this model on training set is 98.86% and the accuracy on validation set is 99.04%.

- **g**

To compare the accuracy for the two new model, it is clear that the model with unbalanced bins label achieves very high accuracy, one of the reasons is that for the bin that has a lot of records, it has wide range, too. Therefore, such kind of bin is easy to be predicted and thus leading to high accuracy. However, since the range of bin is wide, the meaning and quality of the prediction is less. From the practical perspective, the first model with balanced bins is a better choice.

Association rules

Task 1

According to the "str ()" function, "Groceries" belongs to an instantiated "transactions" class which is available in R through the "arules" package (See Figure 18).

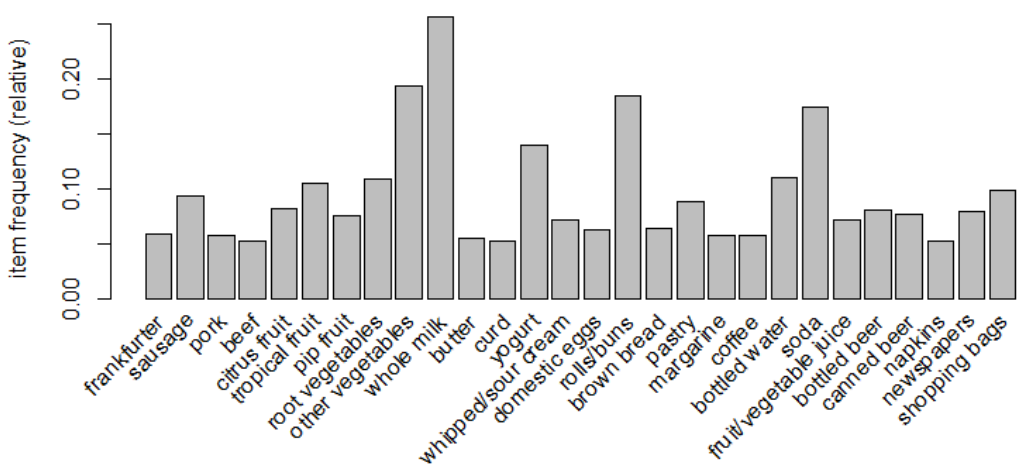
Figure 18

```
> str(Groceries)
Formal class 'transactions' [package "arules"] with 3 slots
  ..@ data       :Formal class 'ngCMatrix' [package "Matrix"] with 5 slots
  .. .. ..@ i       : int [1:43367] 13 60 69 78 14 29 98 24 15 29 ...
  .. .. ..@ p       : int [1:9836] 0 4 7 8 12 16 21 22 27 28 ...
  .. .. ..@ Dim     : int [1:2] 169 9835
  .. .. ..@ Dimnames:List of 2
  .. .. .. ..$ : NULL
  .. .. .. ..$ : NULL
  .. .. ..@ factors : list()
  ..@ itemInfo   :'data.frame': 169 obs. of  3 variables:
  .. ..$ labels: chr [1:169] "frankfurter" "sausage" "liver loaf" "ham" ...
  .. ..$ level2: Factor w/ 55 levels "baby food","bags",..: 44 44 44 44 44 44 44 42 42 41 ...
  .. ..$ level1: Factor w/ 10 levels "canned food",..: 6 6 6 6 6 6 6 6 6 6 ...
  ..@ itemsetInfo:'data.frame': 0 obs. of  0 variables
```

Task 2

As shown in Figure 19, the item frequency barplot for the grocery items with support rate greater than 5% is generated.

Figure 19



Task 3

As shown in Figure 20, one rule with the "rolls/buns" item on the left-hand side and another rule with this item on the right-hand side are created. Their parameters are set to be support equals to 0.006, confidence equals 0.1, minlen equals 2. For the left-hand rule, it means the qualified items set must have frequency higher than 0.006 which is the minimum support value. If the “rools/buns” as the left-hand side item is identified, the predicted right-hand side item’s frequency mush have frequency higher than 0.1. The coverage value means the frequency of “rools/buns” item and it equals to support divided by confidence. The right-hand rule if similar to the left-hand rule, but its left-hand item is default, and its right-hand item is “rolls/buns”.

Figure 20

```
> # 3
> lhs_rules <- apriori(Groceries,
+                      parameter = list(support = 0.006, confidence=0.1,minlen=2),
+                      appearance = list (default="rhs",lhs= "rolls/buns"),
+                      control = list (verbose=F))
> rhs_rules <- apriori(Groceries,
+                      parameter = list(support = 0.006, confidence=0.1,minlen=2),
+                      appearance = list (default="lhs",rhs= "rolls/buns"),
+                      control = list (verbose=F))
>
```

Task 4

As shown in Figure 21, take an example with the first row of results of left-hand rule which has the highest lift value. The support value is 0.0192, which means the frequency of items set {rolls/buns, frankfurter} is 0.0129 in the training set. The confidence is 0.104, which means among all the items set that contain s rolls/buns item, the frequency of item set that also contains frankfurter is 0.104. The coverage is 0.184, which means the frequency of “rolls/buns” item set is 0.184. The lift is 1.77, which means the value that confidence divided by support of frankfurter is 1.77. In other words, if a customer already bought rolls/buns, he is more likely to by frankfurter than others.

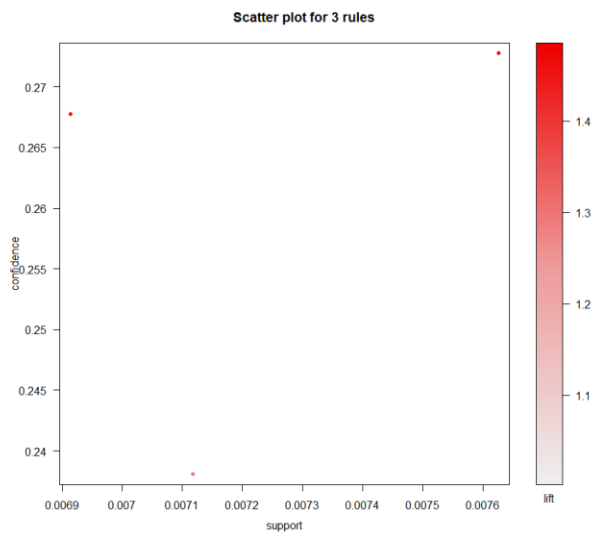
Figure 21

```
> inspect(lhs_rules)
     lhs             rhs                 support    confidence coverage  lift     count
[1]  {rolls/buns} => {frankfurter}       0.01921708 0.1044776  0.1839349 1.771616 189
[2]  {rolls/buns} => {newspapers}        0.01972547 0.1072416  0.1839349 1.343593 194
[3]  {rolls/buns} => {pastry}            0.02094560 0.1138751  0.1839349 1.279956 206
[4]  {rolls/buns} => {shopping bags}     0.01952211 0.1061360  0.1839349 1.077242 192
[5]  {rolls/buns} => {sausage}           0.03060498 0.1663903  0.1839349 1.771048 301
[6]  {rolls/buns} => {bottled water}     0.02419929 0.1315644  0.1839349 1.190373 238
[7]  {rolls/buns} => {tropical fruit}    0.02460600 0.1337756  0.1839349 1.274886 242
[8]  {rolls/buns} => {root vegetables}   0.02430097 0.1321172  0.1839349 1.212101 239
[9]  {rolls/buns} => {soda}              0.03833249 0.2084024  0.1839349 1.195124 377
[10] {rolls/buns} => {yogurt}            0.03436706 0.1868436  0.1839349 1.339363 338
[11] {rolls/buns} => {other vegetables}  0.04260295 0.2316197  0.1839349 1.197047 419
[12] {rolls/buns} => {whole milk}        0.05663447 0.3079049  0.1839349 1.205032 557
>
```

Task 5

As shown in Figure 22, a scatter plot of three rules involving “rools/buns” as the right-hand side item set is generated via the “arulesViz” package. The x axis represents support value, the y axis represents confidence value, the darkness of point represents the lift value.

Figure 22



Task 6

Figure 23 shows the plot for the same three rules in another way. the items grouped around a circle represent an itemset and the arrows indicate the relationship in rules. The size of the circle represents the level of confidence associated with the rule and the color represents the level of lift.

Figure 23