Credit Risk Evaluation—Input Engineering

Liang Hu

1. Build decision tree model based on present attributes

Load libraries.

```
library(caret)

library(RWeka)
library(e1071)
library(DMwR)
```

Read data.

```
setwd(getwd())
credit <- read.csv("credit-g.csv")
```

Use cross-validation to evaluate error.

```
ctrl_cv10 <- trainControl(method='cv', number=10, savePredictions = T,
                          classProbs = T)
```

First, build a decision tree model without input engineering as the reference. The model accuracy is 0.708. Looking at the confusion matrix, 35% of bad loaners are predicted as good.

```
set.seed(1992)
model_DT <- train(class~., data=credit, method='J48', trControl=ctrl_cv10,
tuneGrid=expand.grid(C=0.16, M=5))
model_DT

## C4.5-like Trees
##
## 1000 samples
##   20 predictor
##    2 classes: 'bad', 'good'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 900, 900, 900, 900, 900, 900, ...
## Resampling results:
```

```
## 
##   Accuracy  Kappa    
##   0.708     0.2500149
## 
## Tuning parameter 'C' was held constant at a value of 0.16
## Tuning
##  parameter 'M' was held constant at a value of 5
```

```
prediction_DT = predict(model_DT, credit)
confusionMatrix(prediction_DT, credit$class)
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction bad good
##       bad  195   73
##       good 105  627
##                                           
##                Accuracy : 0.822           
##                  95% CI : (0.7969, 0.8452)
##     No Information Rate : 0.7             
##     P-Value [Acc > NIR] : < 2e-16         
##                                           
##                   Kappa : 0.5629          
##  Mcnemar's Test P-Value : 0.02015         
##                                           
##             Sensitivity : 0.6500          
##             Specificity : 0.8957          
##          Pos Pred Value : 0.7276          
##          Neg Pred Value : 0.8566          
##              Prevalence : 0.3000          
##          Detection Rate : 0.1950          
##    Detection Prevalence : 0.2680          
##       Balanced Accuracy : 0.7729          
##                                           
##        'Positive' Class : bad             
## 
```

2. Select attributes to improve model

The Univariate Filters select 4 variables—*duration*, *credit_amount, installment_commitment*, and *age*. By looking at the variable importance, *checking_status* is the most important variable, and it score is much higher that the rest. Therefore, we choose 5 variables—*duration*, *credit_amount, installment_commitment*, *age,* and *checking_status* to improve the model.

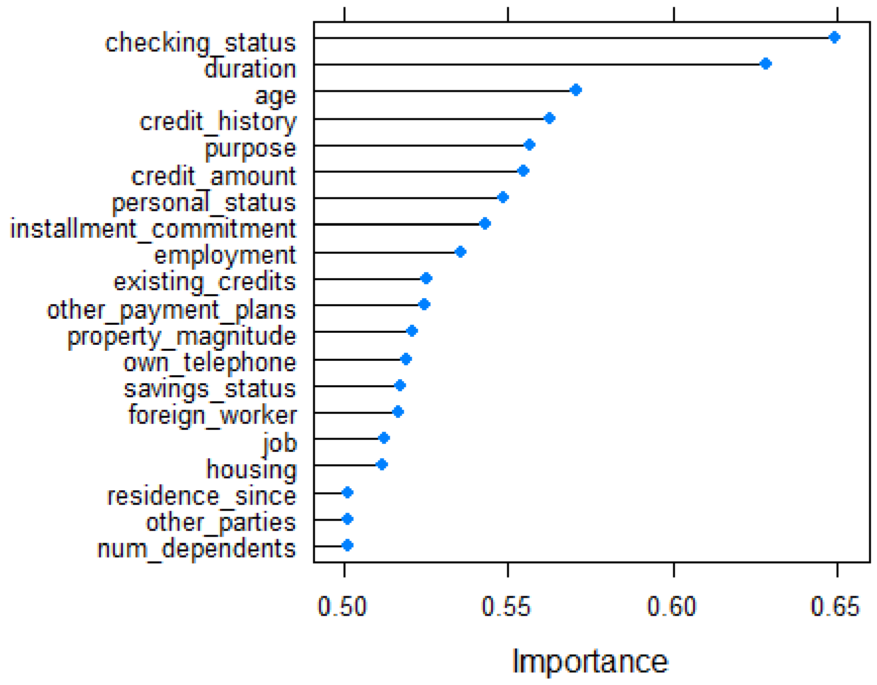
```
filterCtrl <- sbfControl(functions = treebagSBF, method = "repeatedcv",
                         repeats = 5)
sbf(x=credit, y=credit$class, sbfControl = filterCtrl)
## Selection By Filter
##
## Outer resampling method: Cross-Validated (10 fold, repeated 5 times)
##
## Resampling performance:
##
##  Accuracy  Kappa AccuracySD KappaSD
##    0.6664 0.1511     0.0407 0.09603
##
## Using the training set, 4 variables were selected:
##    duration, credit_amount, installment_commitment, age.
##
## During resampling, the top 5 selected variables (out of a possible 5):
##    age (100%), credit_amount (100%), duration (100%),
installment_commitment (82%), existing_credits (2%)
##
## On average, 3.8 variables were selected (min = 3, max = 5)
```

variable importance.

```
DT_Imp <- varImp(model_DT, scale = FALSE)
DT_Imp

## ROC curve variable importance
##
##                        Importance
## checking_status            0.6496
## duration                   0.6286
## age                        0.5706
## credit_history             0.5630
## purpose                    0.5566
## credit_amount              0.5549
## personal_status            0.5489
## installment_commitment     0.5434
## employment                 0.5361
## existing_credits           0.5251
## other_payment_plans        0.5247
## property_magnitude         0.5208
## own_telephone              0.5195
## savings_status             0.5175
## foreign_worker             0.5169
## job                        0.5123
## housing                    0.5119
## residence_since            0.5015
## other_parties              0.5015
## num_dependents             0.5012
```

```r
plot(DT_Imp)
```



From the plot, "checking_status" is very important.

```r
credit_select <- credit[, c('checking_status', 'duration', 'credit_amount',
'installment_commitment', 'age', 'class')]
```

So we build the decision tree model based on the 5 selected attributes. The model accuracy increases by a little bit to 0.711. However, more (64%) bad loaners are predicted as good, which is not what we what.

```r
model_DT_select <- train(class~., data=credit_select, method='J48',
trControl=ctrl_cv10, tuneGrid=expand.grid(C=0.16, M=5))
model_DT_select
```

```
## C4.5-like Trees
##
## 1000 samples
##    5 predictor
##    2 classes: 'bad', 'good'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 900, 900, 900, 900, 900, 900, ...
## Resampling results:
##
```

```
##   Accuracy  Kappa
##   0.711     0.2269466
##
## Tuning parameter 'C' was held constant at a value of 0.16
## Tuning
##  parameter 'M' was held constant at a value of 5
```

```
prediction_DT_select = predict(model_DT_select, credit_select)
confusionMatrix(prediction_DT_select, credit_select$class)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction bad good
##       bad  107   29
##       good 193  671
##
##                Accuracy : 0.778
##                  95% CI : (0.7509, 0.8034)
##     No Information Rate : 0.7
##     P-Value [Acc > NIR] : 1.917e-08
##
##                   Kappa : 0.3736
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.3567
##             Specificity : 0.9586
##          Pos Pred Value : 0.7868
##          Neg Pred Value : 0.7766
##              Prevalence : 0.3000
##          Detection Rate : 0.1070
##    Detection Prevalence : 0.1360
##       Balanced Accuracy : 0.6576
##
##        'Positive' Class : bad
##
```

3. Construct attributes to improve model

This section constructs several new attributed to improve the model. Create 3 new attributes as listed in the table as follow.

New attributes	Description
worker	Whether the loaner is a worker, based on the employment attribute. Workers are more likely to pay the debt.

gender	Male or female, based on the personal_status attribute.
married	Have the loaner ever got married?

```
credit_constr <- credit[, -ncol(credit)]
#whether a worker?
credit_constr$worker <- 'yes'
credit_constr$worker[credit_constr$employment=='unemployed'] <- 'no'
credit_constr$worker <- as.factor(credit_constr$worker)
#gender
credit_constr$gender <- ifelse(credit_constr$personal_status=="'female
div/dep/mar'", 'female', 'male')
credit_constr$gender <- as.factor(credit_constr$gender)
#married?
credit_constr$married <- ifelse(credit_constr$personal_status=="'male
single'", 'no', 'yes')
credit_constr$married <- as.factor(credit_constr$married)
credit_constr$class <- credit$class
```

So we build decision tree model based on the old and new attributes. The model accuracy also increases, to 0.72. 35% of bad loaners are predicted as good, which is the same as the reference model.

```
model_DT_constr <- train(class~., data=credit_constr, method='J48',
trControl=ctrl_cv10, tuneGrid=expand.grid(C=0.16, M=5))
model_DT_constr

## C4.5-like Trees
##
## 1000 samples
##   23 predictor
##    2 classes: 'bad', 'good'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 900, 900, 900, 900, 900, 900, ...
## Resampling results:
##
##   Accuracy  Kappa
##   0.72      0.2861011
##
## Tuning parameter 'C' was held constant at a value of 0.16
## Tuning
##  parameter 'M' was held constant at a value of 5
```

```r
prediction_DT_constr = predict(model_DT_constr, credit_constr)
confusionMatrix(prediction_DT_constr, credit_constr$class)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction bad good
##       bad  195   73
##       good 105  627
##
##                Accuracy : 0.822
##                  95% CI : (0.7969, 0.8452)
##     No Information Rate : 0.7
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.5629
##  Mcnemar's Test P-Value : 0.02015
##
##             Sensitivity : 0.6500
##             Specificity : 0.8957
##          Pos Pred Value : 0.7276
##          Neg Pred Value : 0.8566
##              Prevalence : 0.3000
##          Detection Rate : 0.1950
##    Detection Prevalence : 0.2680
##       Balanced Accuracy : 0.7729
##
##        'Positive' Class : bad
##
```

4. Under-sampling to improve minority class prediction

A big concern with the model is the prediction results for bad loaners. It is more risky to predict bad loaners as good than predict the good as bad. In the raw dataset, there is 30% loaners classified as bad and 70% as good. To improve the minority class (the bad) prediction, we use under-sampling to achieve this goal.

After under-sampling, bad and good loaners are both 300. Build decision tree model based on the under-sampling data. The model accuracy reduces to 0.657. However, the minority class prediction improves significantly—now 21.7% of bad loaners are predicted as good, decreasing by 13.3%.

```r
set.seed(1991)
credit_down <- downSample(x = credit[, -ncol(credit)],
                         y = credit$class)
table(credit_down$Class)
```

```
##
##  bad good
##  300  300
```

build decision tree model based on under-sampling data

```
model_DT_down <- train(Class~., data=credit_down, method='J48',
trControl=ctrl_cv10, tuneGrid=expand.grid(C=0.16, M=5))
model_DT_down
```

```
## C4.5-like Trees
##
## 600 samples
##  20 predictor
##   2 classes: 'bad', 'good'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 540, 540, 540, 540, 540, 540, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.6566667  0.3133333
##
## Tuning parameter 'C' was held constant at a value of 0.16
## Tuning
##  parameter 'M' was held constant at a value of 5
```

```
prediction_DT_down = predict(model_DT_down, credit_down)
confusionMatrix(prediction_DT_down, credit_down$Class)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction bad good
##       bad  235   58
##       good  65  242
##
##                Accuracy : 0.795
##                  95% CI : (0.7604, 0.8266)
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.59
##  Mcnemar's Test P-Value : 0.5885
##
##             Sensitivity : 0.7833
##             Specificity : 0.8067
##          Pos Pred Value : 0.8020
##          Neg Pred Value : 0.7883
##              Prevalence : 0.5000
##          Detection Rate : 0.3917
##    Detection Prevalence : 0.4883
##       Balanced Accuracy : 0.7950
##
##        'Positive' Class : bad
##
```

5. Over-sampling to improve minority class prediction

Moreover, we try over-sampling in order to improve the minority class prediction. The dataset after over-sampling has 700 bad and 700 good loaners. The decision tree model accuracy improves a lot, to 0.759. Another improvement is the minority class prediction—only 9.9% bad loaners are predicted as good.

```
set.seed(1991)
credit_up <- upSample(x = credit[, -ncol(credit)],
                         y = credit$class)
table(credit_up$Class)

##
##  bad good
##  700  700
```

build decision tree model based on over-sampling data

```
model_DT_up <- train(Class~., data=credit_up, method='J48',
trControl=ctrl_cv10, tuneGrid=expand.grid(C=0.16, M=5))
model_DT_up

## C4.5-like Trees
##
## 1400 samples
##   20 predictor
##    2 classes: 'bad', 'good'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 1260, 1260, 1260, 1260, 1260, 1260, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.7592857  0.5185714
##
## Tuning parameter 'C' was held constant at a value of 0.16
## Tuning
##  parameter 'M' was held constant at a value of 5
```

```
prediction_DT_up = predict(model_DT_up, credit_up)
confusionMatrix(prediction_DT_up, credit_up$Class)

## Confusion Matrix and Statistics
##
##           Reference
## Prediction bad good
##       bad  631   97
##       good  69  603
##
##                Accuracy : 0.8814
##                  95% CI : (0.8633, 0.8979)
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.7629
##  Mcnemar's Test P-Value : 0.03612
##
##             Sensitivity : 0.9014
##             Specificity : 0.8614
##          Pos Pred Value : 0.8668
##          Neg Pred Value : 0.8973
##              Prevalence : 0.5000
##          Detection Rate : 0.4507
##    Detection Prevalence : 0.5200
##       Balanced Accuracy : 0.8814
##
##        'Positive' Class : bad
##
```

6. SMOTE to improve minority class prediction

Use SMOTE to improve minority class prediction and now there are 900 bad and 1200 good loaners in the dataset. Compared to under-sampling and over-sampling, SMOTE improves the model accuracy most significantly, to 0.798. 13.6% of bad loaners are predicted as good. It is also a big improvement compared to the reference model, but not as significant as over-sampling.

```
set.seed(1991)
credit_smote <- SMOTE(class~., data=credit)
table(credit_smote$class)

##
##  bad good
##  900 1200
```

build decision tree model based on SMOTEed data

```
model_DT_smote <- train(class~., data=credit_smote, method='J48',
trControl=ctrl_cv10, tuneGrid=expand.grid(C=0.16, M=5))
model_DT_smote
```

```
## C4.5-like Trees
##
## 2100 samples
##   20 predictor
##    2 classes: 'bad', 'good'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 1890, 1890, 1890, 1890, 1890, 1890, ...
## Resampling results:
##
##   Accuracy  Kappa
##   0.797619  0.5792979
##
## Tuning parameter 'C' was held constant at a value of 0.16
## Tuning
##  parameter 'M' was held constant at a value of 5
```

```
prediction_DT_smote = predict(model_DT_smote, credit_smote)
confusionMatrix(prediction_DT_smote, credit_smote$class)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  bad good
##       bad   778   93
##       good  122 1107
##
##                Accuracy : 0.8976
##                  95% CI : (0.8839, 0.9103)
##     No Information Rate : 0.5714
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.7901
##  Mcnemar's Test P-Value : 0.05619
##
##             Sensitivity : 0.8644
##             Specificity : 0.9225
##          Pos Pred Value : 0.8932
##          Neg Pred Value : 0.9007
##              Prevalence : 0.4286
##          Detection Rate : 0.3705
##    Detection Prevalence : 0.4148
##       Balanced Accuracy : 0.8935
##
```

```
##        'Positive' Class : bad
##
```

7. Summary

The table below compares model accuracy and minority class prediction using different input engineering techniques. SMOTE not only improves model accuracy but also significantly improves minority class prediction.

Input engineering techniques	Model accuracy	Percent of bad predicted as good
None	0.708	35%
Attribute selection	0.711	64%
Attribute construction	0.72	35%
Under-sampling	0.657	21.7%
Over-sampling	0.759	9.9%
SMOTE	0.798	13.6%