Gradient Boosting Machine

Author: Rajat Jain

Last Updated: 2018-05-29

Contents

Setting Training Control Params

Using 10-fold Cross Validation with 10 repeatitions.

```
ctrl <- trainControl(method="repeatedcv",
                     number=10,
                     repeats=10,
                     classProbs=TRUE,
                     savePredictions=TRUE,
                     allowParallel=TRUE)
set.seed(123)
```

Training - GBM Model

Caret is Awesome! So using caret to train the model.

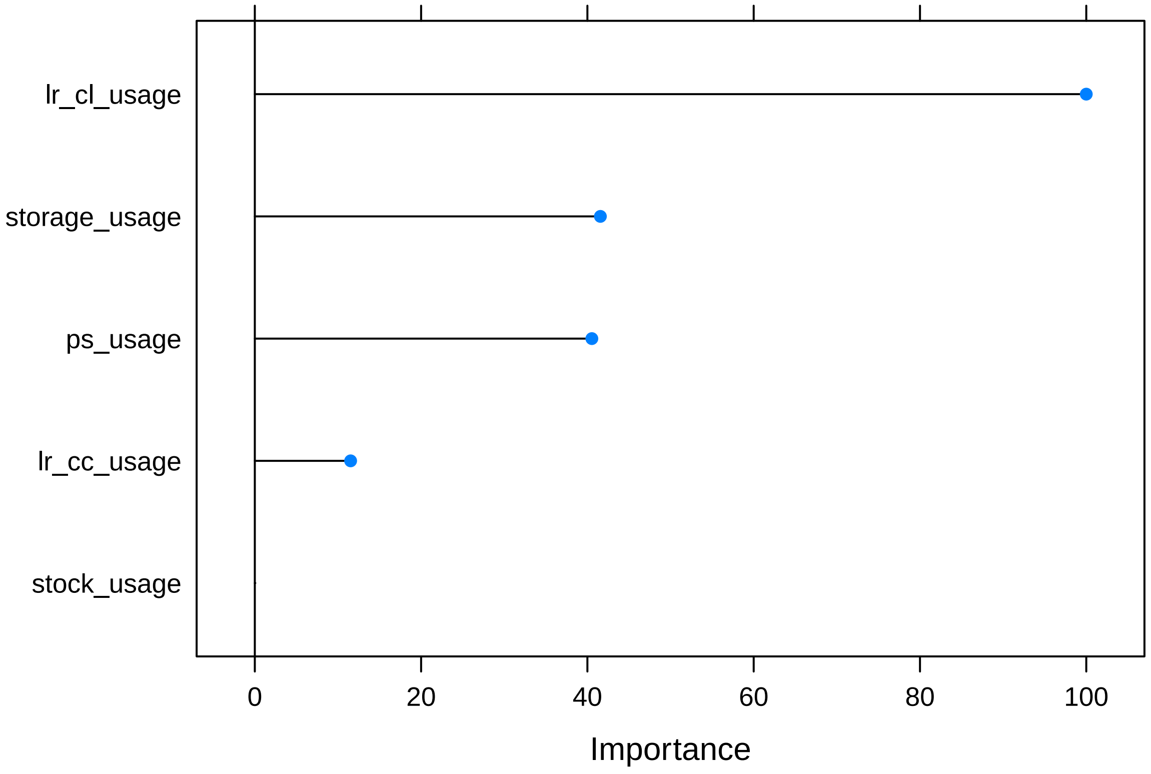
```
fit <- train(class ~ lr_cc_usage + lr_cl_usage + storage_usage + ps_usage + stock_usage,
             data=usage.data, method = "gbm", trControl = ctrl)
```

```
## Iter   TrainDeviance   ValidDeviance   StepSize   Improve
##      1        1.3614             nan     0.1000    0.0028
##      2        1.3557             nan     0.1000    0.0024
##      3        1.3515             nan     0.1000    0.0010
##      4        1.3481             nan     0.1000    0.0016
##      5        1.3457             nan     0.1000    0.0009
##      6        1.3434             nan     0.1000    0.0007
##      7        1.3411             nan     0.1000    0.0006
##      8        1.3390             nan     0.1000    0.0010
##      9        1.3372             nan     0.1000    0.0006
##     10        1.3357             nan     0.1000    0.0002
##     20        1.3268             nan     0.1000   -0.0001
##     40        1.3151             nan     0.1000   -0.0006
##     50        1.3109             nan     0.1000   -0.0003
```

Summary of the trained Model

```
## Stochastic Gradient Boosting
##
## 2650 samples
##    5 predictor
##    2 classes: 'OTHER', 'PHOTOGRAPHER'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 10 times)
## Summary of sample sizes: 2385, 2385, 2385, 2385, 2385, 2385, ...
## Resampling results across tuning parameters:
##
##   interaction.depth  n.trees  Accuracy   Kappa
##   1                   50      0.5980377  0.1458549
##   1                  100      0.5965283  0.1460418
##   1                  150      0.5975094  0.1486170
##   2                   50      0.6008679  0.1548544
##   2                  100      0.5996981  0.1549888
##   2                  150      0.5976226  0.1502246
##   3                   50      0.5981132  0.1492871
##   3                  100      0.5971698  0.1505453
##   3                  150      0.5950566  0.1467029
##
## Tuning parameter 'shrinkage' was held constant at a value of 0.1
##
## Tuning parameter 'n.minobsinnode' was held constant at a value of 10
## Accuracy was used to select the optimal model using the largest value.
## The final values used for the model were n.trees = 50, interaction.depth
##  = 2, shrinkage = 0.1 and n.minobsinnode = 10.
```

Variable Importance



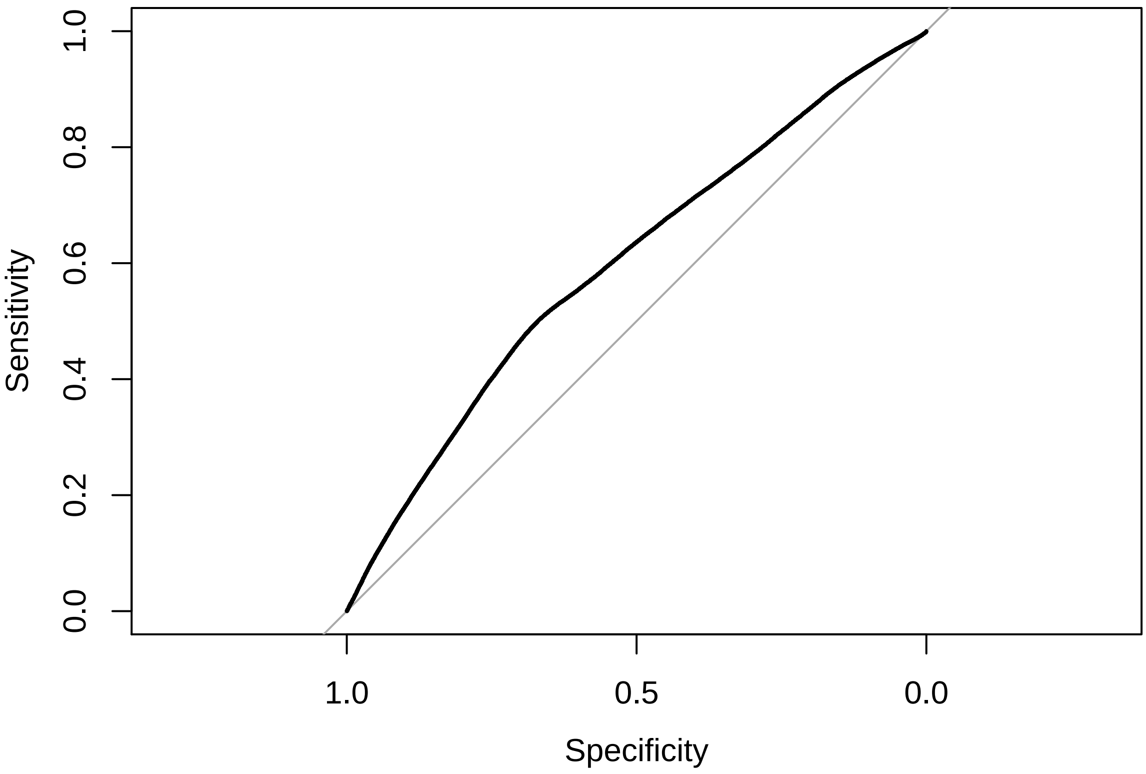
Performance

Based on the measure defined in the FPS, we will use classification accuracy as our performance measure.

Confusion Matrix

```
## Cross-Validated (10 fold, repeated 10 times) Confusion Matrix
##
## (entries are percentual average cell counts across resamples)
##
##               Reference
## Prediction     OTHER PHOTOGRAPHER
##   OTHER         43.8         27.1
##   PHOTOGRAPHER  12.8         16.3
##
##  Accuracy (average) : 0.6009
```

ROC Curve



```
##
## Call:
## plot.roc.default(x = fit$pred$obs, predictor = fit$pred$PHOTOGRAPHER)
##
## Data: fit$pred$PHOTOGRAPHER in 135000 controls (fit$pred$obs OTHER) < 103500 cases (fit$pred$obs PHO
## Area under the curve: 0.5986
```

Accuracy

- Kohen’s Kappa: 0.15
- Observed Accuracy : 60.09%
- Desired accuracy : 70%
- Performance is Not Satisfactory.