Spam Classification

Abirami Sabbani

3/9/2022

```r
library(tidyverse)
```

```
## ── Attaching packages ─────────────────────────────────────── tidyverse 1.3.1 ──
```

```
## ✓ ggplot2 3.3.5     ✓ purrr   0.3.4
## ✓ tibble  3.1.6     ✓ dplyr   1.0.8
## ✓ tidyr   1.2.0     ✓ stringr 1.4.0
## ✓ readr   2.1.2     ✓ forcats 0.5.1
```

```
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```r
library(latex2exp)
library(ggforce)
#library(e1071)
library(scales)
```

```
##
## Attaching package: 'scales'
```

```
## The following object is masked from 'package:purrr':
##
##     discard
```

```
## The following object is masked from 'package:readr':
##
##     col_factor
```

```r
library(matrixStats)
```

```
##
## Attaching package: 'matrixStats'
```

```
## The following object is masked from 'package:dplyr':
##
##     count
```

```r
library(caret)
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

```r
library(MASS)
```

```
##
## Attaching package: 'MASS'
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

1. Standardized the columns so that they all have zero mean and unit variance.

```r
train = read.csv('spam-train.txt', header = FALSE)
```

```r
test = read.csv('spam-test.txt', header = FALSE)
```

```r
stan_train = scale(train)
```

```r
colMeans(stan_train)
```

```
##            V1            V2            V3            V4            V5 
## -1.352131e-17  7.475657e-18  2.565547e-17  3.893654e-18 -7.217174e-18 
##            V6            V7            V8            V9           V10 
## -5.180980e-18  1.708027e-17 -1.149544e-17  2.739019e-17 -9.692562e-18 
##           V11           V12           V13           V14           V15 
##  7.671923e-18 -6.222833e-17  1.570810e-17 -6.175321e-18 -2.093433e-17 
##           V16           V17           V18           V19           V20 
## -3.686641e-18  1.560742e-17  8.080293e-18 -1.258820e-17 -3.338226e-18 
##           V21           V22           V23           V24           V25 
##  2.263002e-18 -1.007972e-17 -3.206099e-17  1.963060e-17 -3.735425e-17 
##           V26           V27           V28           V29           V30 
## -1.363401e-17 -2.017175e-17 -4.727927e-18 -3.880079e-18 -3.369900e-18 
##           V31           V32           V33           V34           V35 
##  2.895919e-18  5.972833e-18 -7.852918e-18  1.638004e-18 -7.983291e-18 
##           V36           V37           V38           V39           V40 
##  1.494226e-17 -1.819339e-17 -3.637998e-18 -1.350675e-17  7.959819e-18 
##           V41           V42           V43           V44           V45 
## -1.225110e-17  6.182109e-19 -3.893654e-18  1.751253e-17 -2.673833e-17 
##           V46           V47           V48           V49           V50 
## -1.710572e-17 -2.162890e-18  3.617637e-18 -7.765815e-19  6.578092e-17 
##           V51           V52           V53           V54           V55 
## -1.876013e-17  4.286427e-17  4.484687e-17 -7.693417e-18  1.377541e-18 
##           V56           V57           V58 
## -1.930877e-17 -4.867407e-17  7.985497e-17
```

```
colVars(as.matrix(stan_train))
```

```
##  [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [39] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

```
stan_test = scale(test)
```

```
colMeans(stan_test)
```

```
##            V1            V2            V3            V4            V5 
##  1.320832e-17  7.007876e-18 -3.689793e-17 -4.508698e-18  3.006025e-17 
##            V6            V7            V8            V9           V10 
##  6.640350e-18 -1.669134e-18 -1.822025e-17  1.414467e-17 -3.528251e-19 
##           V11           V12           V13           V14           V15 
##  1.247553e-17 -5.867979e-18  8.872647e-18  1.270397e-17 -8.699061e-18 
##           V16           V17           V18           V19           V20 
## -1.531623e-17  1.318345e-17  3.666780e-18 -5.272021e-17  2.714039e-19 
##           V21           V22           V23           V24           V25 
## -1.222731e-17  1.313369e-17  1.985094e-17 -4.957645e-18  8.439531e-18 
##           V26           V27           V28           V29           V30 
## -3.063133e-17  1.344354e-17 -2.968707e-17  1.636340e-18 -3.967021e-18 
##           V31           V32           V33           V34           V35 
## -6.043261e-18  1.404063e-17 -2.684637e-18 -7.739536e-18 -1.169412e-17 
##           V36           V37           V38           V39           V40 
##  8.289128e-19  1.053273e-17 -1.031335e-18  9.218687e-18  1.291204e-17 
##           V41           V42           V43           V44           V45 
##  6.834856e-18 -1.360865e-17  1.110494e-17 -6.355375e-19  6.905534e-18 
##           V46           V47           V48           V49           V50 
##  5.249404e-18 -4.708858e-18  1.098281e-17  7.983799e-19 -4.784738e-17 
##           V51           V52           V53           V54           V55 
## -2.002961e-17 -1.433564e-17 -2.676439e-17 -1.191322e-17  2.231167e-18 
##           V56           V57           V58 
## -2.067193e-18  3.397949e-17 -5.905750e-17
```

```
colVars(stan_test)
```

```
##  [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [39] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

2. Transformed the features using log(xij + 1).

```
log_train = log(train + 1)
head(log_train)
```

	V1 <dbl>	V2 <dbl>	V3 <dbl>	V4 <dbl>	V5 <dbl>	V6 <dbl>	V7 <dbl>	V8 <dbl>	V9 <dbl>
1	0.00000000	0	0.0000000	0.00000	0.0000000	0.0000000	0	0.0000000	0.00000
2	0.00000000	0	0.4637340	0.10436	0.0000000	0.0000000	0	0.0000000	0.10436
3	0.05826891	0	0.3364722	0.00000	0.1222176	0.1222176	0	0.1222176	0.00000
4	0.00000000	0	0.0000000	0.00000	0.0000000	0.0000000	0	0.0000000	0.00000
5	0.00000000	0	0.0000000	0.00000	0.0000000	0.3646431	0	0.0000000	0.00000
6	0.00000000	0	0.4252677	0.00000	0.0000000	0.4252677	0	0.0000000	0.00000

6 rows | 1-10 of 59 columns

```
log_test = log(test + 1)
head(log_test)
```

	V1	V2	V3	...	V5	V6	V7	V8	V9
	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
1	0.1133287	0.1133287	0.2151114	0	0.8501509	0.1133287	0.0000000	0.1133287	0.0000000
2	0.0000000	0.0000000	0.2776317	0	0.4946962	0.4946962	0.4946962	0.2776317	0.2776317
3	0.0000000	0.0000000	0.0000000	0	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
4	0.0000000	0.0000000	0.3364722	0	0.3364722	0.1823216	0.0000000	0.0000000	0.0000000
5	0.4121097	0.3576744	0.2546422	0	0.1310283	0.0295588	0.0000000	0.1655144	0.4317824
6	0.0000000	0.0000000	0.0000000	0	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000

6 rows | 1-10 of 59 columns

3. Discretized each feature using I(xij > 0).

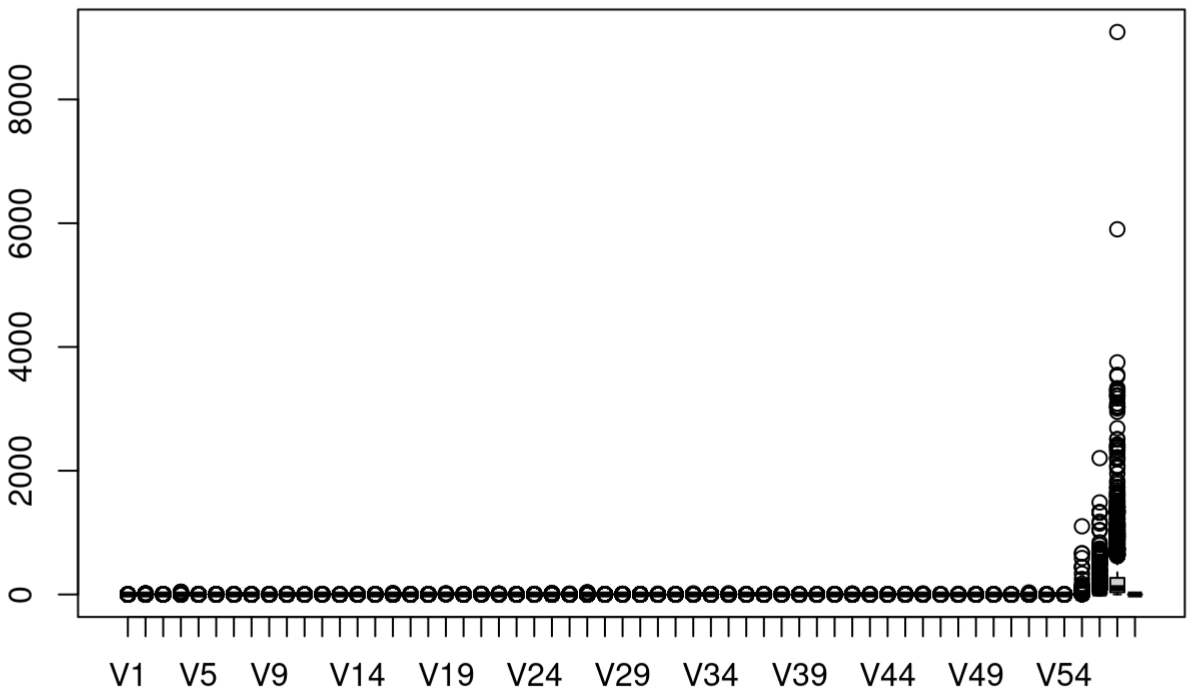
```
I_train = (train > 0 ) * 1
```

```
I_test = (test > 0)*1
head(I_test)
```

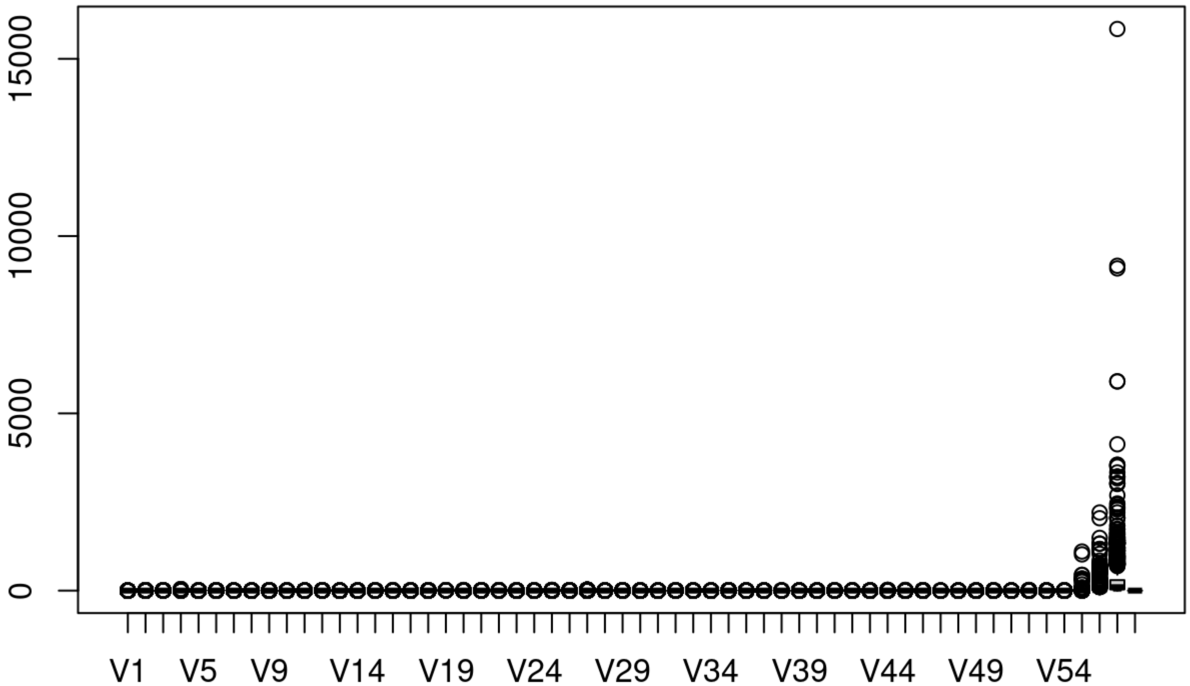
```
##      V1 V2 V3 V4 V5 V6 V7 V8 V9 V10 V11 V12 V13 V14 V15 V16 V17 V18 V19 V20 V21
## [1,]  1  1  1  0  1  1  0  1  0   0   1   1   0   0   0   1   1   0   1   0   1
## [2,]  0  0  1  0  1  1  1  1  1   0   0   1   1   0   0   1   1   1   1   0   1
## [3,]  0  0  0  0  0  0  0  0  0   0   0   0   0   0   0   0   0   0   0   0   0
## [4,]  0  0  1  0  1  1  0  0  0   1   1   1   0   0   0   1   1   1   1   0   1
## [5,]  1  1  1  0  1  1  0  1  1   1   1   1   1   1   1   1   1   1   1   0   1
## [6,]  0  0  0  0  0  0  0  0  0   0   0   1   0   0   0   0   0   0   1   0   1
##      V22 V23 V24 V25 V26 V27 V28 V29 V30 V31 V32 V33 V34 V35 V36 V37 V38 V39
## [1,]   0   0   0   0   0   0   0   0   0   0   0   0   0   0   1   0   0   0
## [2,]   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
## [3,]   0   0   0   1   1   0   0   0   0   0   0   0   0   0   0   1   0   0
## [4,]   0   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
## [5,]   0   1   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
## [6,]   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0   1   0   0
##      V40 V41 V42 V43 V44 V45 V46 V47 V48 V49 V50 V51 V52 V53 V54 V55 V56 V57
## [1,]   1   0   0   0   0   1   0   0   0   1   1   0   1   1   0   1   1   1
## [2,]   0   0   0   0   0   1   0   0   0   0   1   0   0   1   0   1   1   1
## [3,]   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   1   1   1
## [4,]   1   0   0   0   0   0   0   0   0   0   1   0   0   1   0   1   1   1
## [5,]   0   0   0   0   0   1   0   0   0   1   1   0   1   1   1   1   1   1
## [6,]   0   0   1   0   0   1   1   0   0   0   1   0   1   0   0   1   1   1
##      V58
## [1,]   1
## [2,]   1
## [3,]   0
## [4,]   1
## [5,]   1
## [6,]   0
```

Visualization for original train and test data

```
boxplot(train)
```

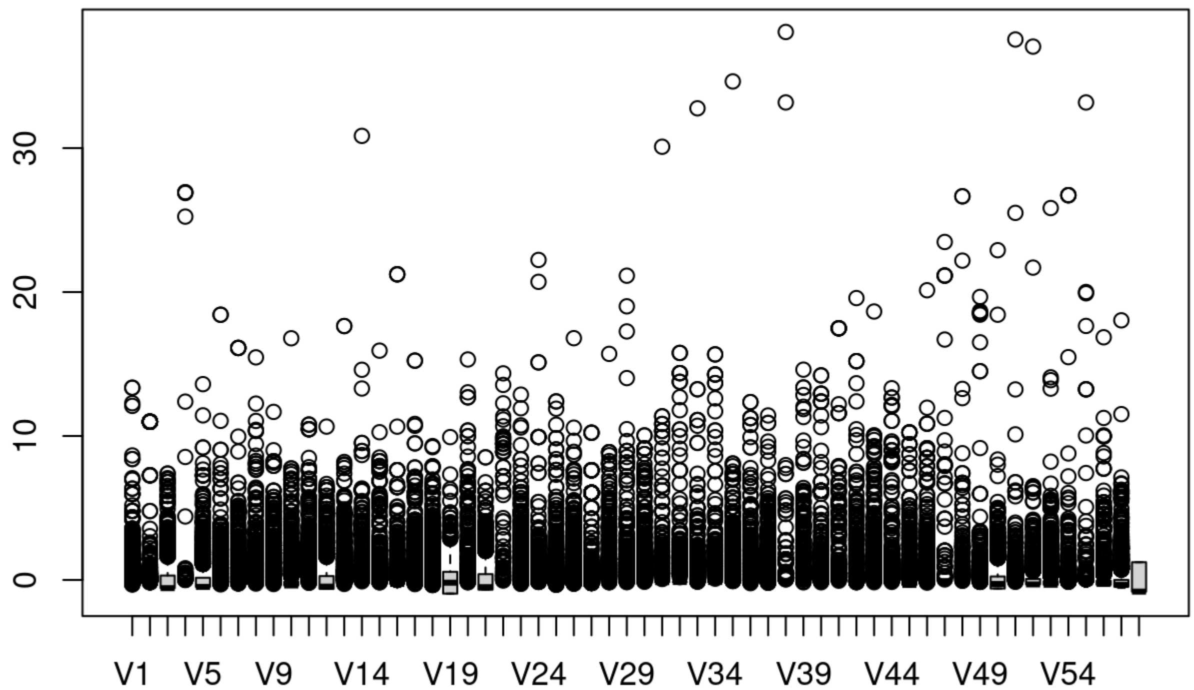



```
boxplot(test)
```

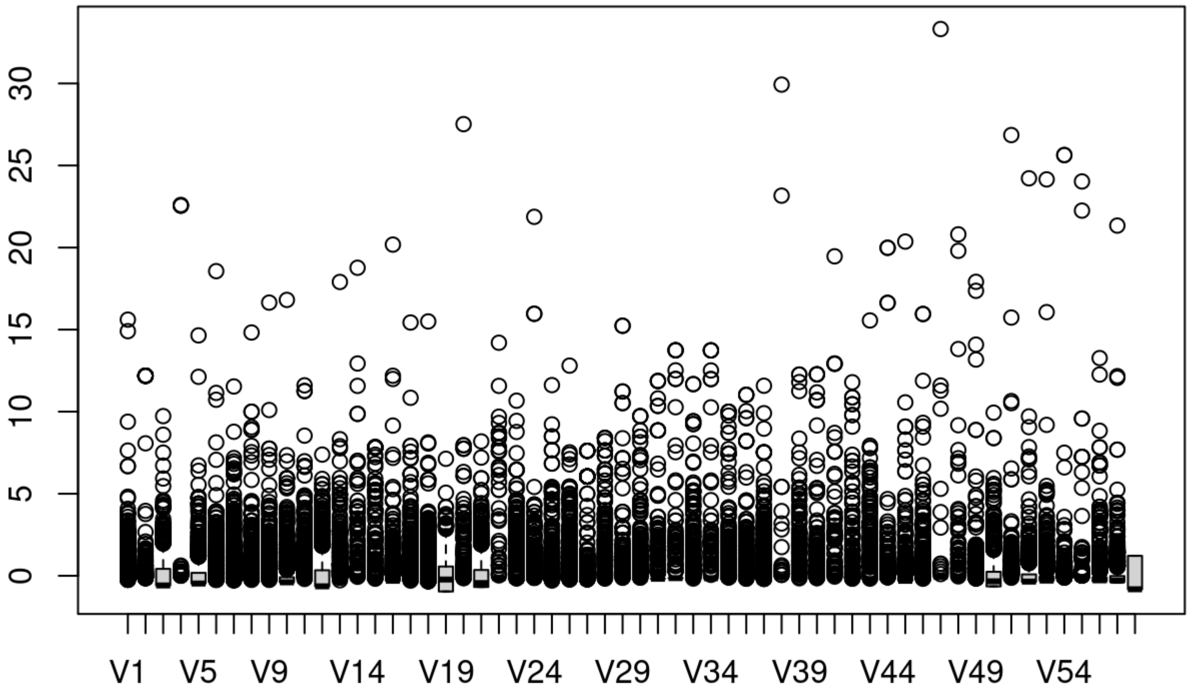


Visualization for standardized train and test data

```
boxplot(stan_train)
```

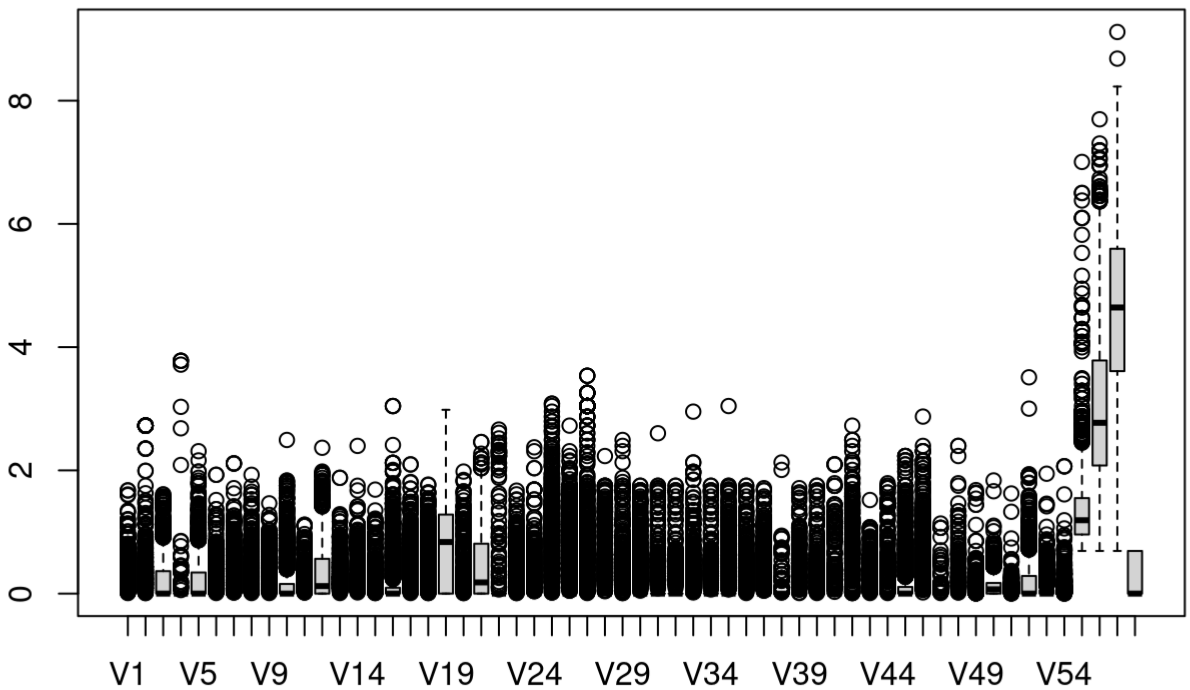



```
boxplot(stan_test)
```

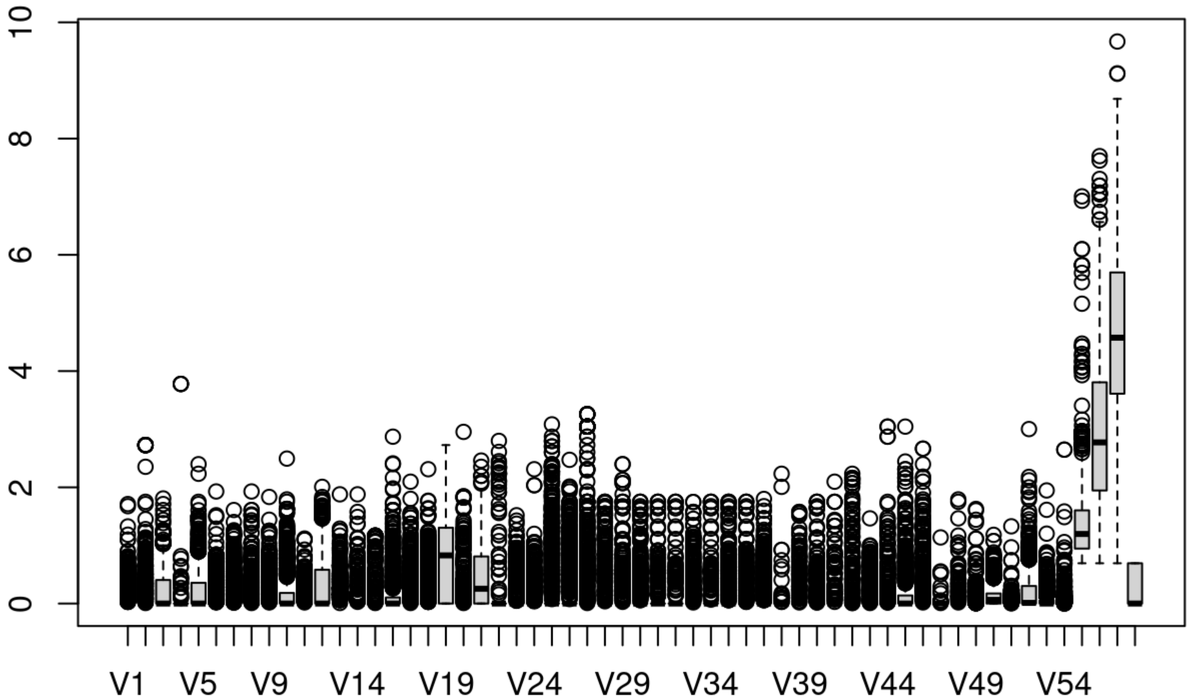


Visualization for log transformed train and test data

```
boxplot(log_train)
```

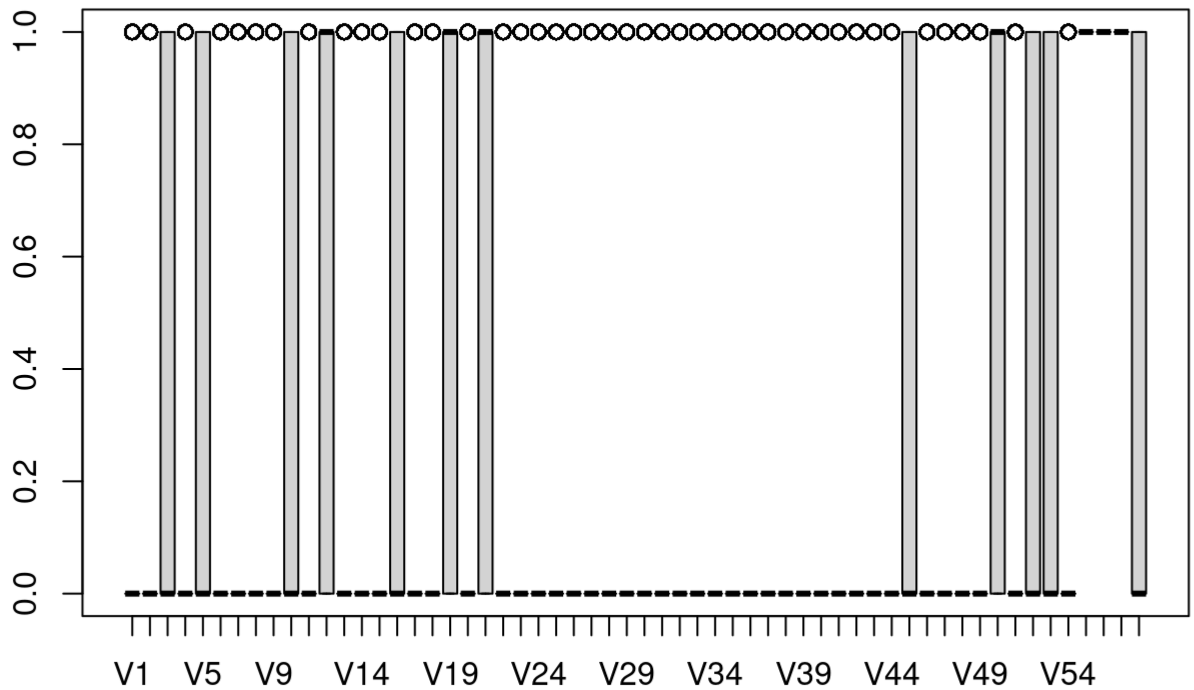



```
boxplot(log_test)
```

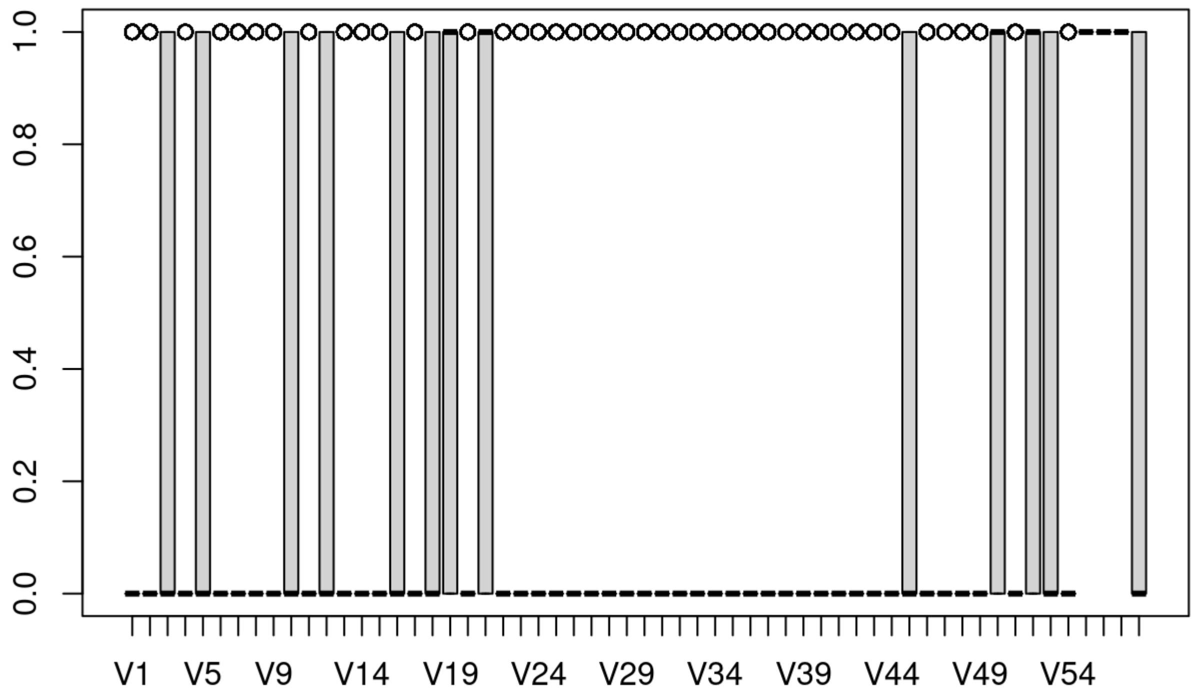


Visualization for discretized train and test data

```
boxplot(I_train)
```




```
boxplot(I_test)
```



Since the train and test datasets have a different amount of data, the scale is different but the ratios are about the same. Also the log transformation feature shows a high variance for features 56 and 57, but it is not as noticeable when the feature is standardized.

```
stan_train = as.data.frame(stan_train)
log_train = as.data.frame(log_train)
I_train = as.data.frame(I_train)
```

```
stan_train$V58 <- train$V58
log_train$V58 <- train$V58
I_train$V58 <- train$V58
```

```
test = as.data.frame(test)
stan_test = as.data.frame(stan_test)
log_test = as.data.frame(log_test)
I_test = as.data.frame(I_test)
```

```
stan_test$V58 <- test$V58
log_test$V58 <- test$V58
I_test$V58 <- test$V58
stan_train$V58 <- train$V58
log_train$V58 <- train$V58
I_train$V58 <- train$V58
```

4. **Linear Regression on original train and test data**

```
train = as.data.frame(train)
```

```
test = as.data.frame(test)
```

```
lr_train <- glm(V58 ~ ., data = train, family = "binomial")
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
summary(lr_train)
```

```
##
## Call:
## glm(formula = V58 ~ ., family = "binomial", data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -4.3245  -0.1988  -0.0001   0.0940   3.6053
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.696e+00  1.745e-01  -9.718  < 2e-16 ***
## V1          -2.225e-01  2.698e-01  -0.825 0.409508
## V2          -1.662e-01  1.067e-01  -1.557 0.119379
## V3           5.119e-02  1.487e-01   0.344 0.730612
## V4           3.418e+00  1.660e+00   2.059 0.039464 *
## V5           6.358e-01  1.379e-01   4.611 4.00e-06 ***
## V6           2.709e-01  1.845e-01   1.469 0.141965
## V7           2.950e+00  4.472e-01   6.595 4.24e-11 ***
## V8           5.384e-01  1.957e-01   2.752 0.005931 **
## V9           7.796e-01  3.616e-01   2.156 0.031095 *
## V10          8.869e-02  9.414e-02   0.942 0.346145
## V11         -1.053e+00  4.049e-01  -2.600 0.009319 **
## V12         -4.140e-02  8.510e-02  -0.486 0.626655
## V13         -4.274e-01  3.580e-01  -1.194 0.232431
## V14         -1.051e-02  1.952e-01  -0.054 0.957058
## V15          1.137e+00  8.499e-01   1.338 0.181023
## V16          1.184e+00  1.769e-01   6.690 2.24e-11 ***
## V17          1.299e+00  3.048e-01   4.260 2.04e-05 ***
## V18         -5.944e-02  1.666e-01  -0.357 0.721327
## V19          8.908e-02  4.513e-02   1.974 0.048423 *
## V20          4.469e+00  1.262e+00   3.540 0.000400 ***
## V21          4.288e-01  7.294e-02   5.879 4.13e-09 ***
## V22          1.136e-01  1.721e-01   0.660 0.509332
## V23          3.575e+00  7.346e-01   4.866 1.14e-06 ***
## V24          2.291e-01  1.421e-01   1.612 0.106930
## V25         -2.001e+00  3.557e-01  -5.626 1.84e-08 ***
## V26         -5.373e-01  4.684e-01  -1.147 0.251312
## V27         -5.814e+00  1.191e+00  -4.880 1.06e-06 ***
## V28          4.695e-01  3.270e-01   1.436 0.151031
## V29         -4.657e+00  3.187e+00  -1.461 0.143936
## V30          2.470e-02  2.085e-01   0.118 0.905705
## V31         -2.007e-01  6.221e-01  -0.323 0.746941
## V32         -1.252e+00  3.188e+00  -0.393 0.694553
## V33         -8.380e-01  4.466e-01  -1.876 0.060610 .
## V34          2.837e+00  3.361e+00   0.844 0.398662
## V35         -1.066e+00  6.155e-01  -1.732 0.083302 .
## V36          2.017e-01  4.490e-01   0.449 0.653264
## V37         -6.741e-01  3.853e-01  -1.749 0.080214 .
## V38         -7.805e-01  6.252e-01  -1.248 0.211871
## V39         -8.586e-02  4.989e-01  -0.172 0.863349
## V40         -4.763e-01  5.347e-01  -0.891 0.373028
## V41         -4.579e+01  3.016e+01  -1.518 0.128996
```

```
## V42         -2.345e+00  8.064e-01  -2.907 0.003644 **
## V43         -2.399e+00  1.264e+00  -1.899 0.057599 .
## V44         -1.979e+00  9.628e-01  -2.055 0.039857 *
## V45         -1.126e+00  2.253e-01  -4.997 5.83e-07 ***
## V46         -1.321e+00  3.107e-01  -4.252 2.11e-05 ***
## V47         -7.138e+00  3.614e+00  -1.975 0.048234 *
## V48         -1.345e+00  1.026e+00  -1.311 0.189745
## V49         -9.399e-01  4.588e-01  -2.049 0.040502 *
## V50          2.118e-01  2.676e-01   0.791 0.428765
## V51         -6.083e-01  1.191e+00  -0.511 0.609646
## V52          2.853e-01  6.688e-02   4.266 1.99e-05 ***
## V53          4.432e+00  7.071e-01   6.268 3.66e-10 ***
## V54          2.301e+00  1.308e+00   1.759 0.078551 .
## V55         -1.699e-02  6.947e-03  -2.446 0.014445 *
## V56          8.473e-03  2.299e-03   3.686 0.000228 ***
## V57          1.260e-03  2.886e-04   4.367 1.26e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 4121.0  on 3066  degrees of freedom
## Residual deviance: 1157.4  on 3009  degrees of freedom
## AIC: 1273.4
##
## Number of Fisher Scoring iterations: 13
```

From the summary, the result indicates that features: 4,5, 7, 8, 9, 11, 16, 17, 19, 20, 21, 23, 25, 27, 42, 44, 45, 46, 47, 49, 52, 52, 55, 56, and 57 are statistically significant because their p-values are less then 0.05

```
pred_train <- (predict(lr_train, train) > 0) * 1
pred_test <- (predict(lr_train, test) > 0) * 1
```

```
cm_lr_train = confusionMatrix(as.factor(pred_train), as.factor(train$V58), positive = "1")
cm_lr_train
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1762  133
##          1   87 1085
##
##                Accuracy : 0.9283
##                  95% CI : (0.9186, 0.9372)
##     No Information Rate : 0.6029
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8492
##
##  Mcnemar's Test P-Value : 0.002414
##
##             Sensitivity : 0.8908
##             Specificity : 0.9529
##          Pos Pred Value : 0.9258
##          Neg Pred Value : 0.9298
##              Prevalence : 0.3971
##          Detection Rate : 0.3538
##    Detection Prevalence : 0.3821
##       Balanced Accuracy : 0.9219
##
##        'Positive' Class : 1
##
```

The accuracy is 92.83% for the classification error of the Logistic Regression of train data.

```
cm_lr_test = confusionMatrix(as.factor(pred_test), as.factor(test$V58), positive = "1")
cm_lr_test
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 876  72
##          1  40 546
##
##                Accuracy : 0.927
##                  95% CI : (0.9128, 0.9395)
##     No Information Rate : 0.5971
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.847
##
##  Mcnemar's Test P-Value : 0.003398
##
##             Sensitivity : 0.8835
##             Specificity : 0.9563
##          Pos Pred Value : 0.9317
##          Neg Pred Value : 0.9241
##              Prevalence : 0.4029
##          Detection Rate : 0.3559
##    Detection Prevalence : 0.3820
##       Balanced Accuracy : 0.9199
##
##        'Positive' Class : 1
##
```

The accuracy is 92.7% for the classification error of the Logistic Regression of test data.

Linear Regression on Standardized Train and Test Data

```
lr_stan_train <- glm(V58 ~ ., data = stan_train, family = "binomial")
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
summary(lr_stan_train)
```

```
##
## Call:
## glm(formula = V58 ~ ., family = "binomial", data = stan_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -4.3245  -0.1988  -0.0001   0.0940   3.6053
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -7.36294    1.76165  -4.180 2.92e-05 ***
## V1           -0.07047    0.08544  -0.825 0.409508
## V2           -0.21268    0.13656  -1.557 0.119379
## V3            0.02573    0.07472   0.344 0.730612
## V4            5.42487    2.63430   2.059 0.039464 *
## V5            0.41029    0.08897   4.611 4.00e-06 ***
## V6            0.08488    0.05780   1.469 0.141965
## V7            1.30763    0.19827   6.595 4.24e-11 ***
## V8            0.20112    0.07309   2.752 0.005931 **
## V9            0.21642    0.10039   2.156 0.031095 *
## V10           0.05737    0.06090   0.942 0.346145
## V11          -0.19561    0.07523  -2.600 0.009319 **
## V12          -0.03552    0.07302  -0.486 0.626655
## V13          -0.13217    0.11069  -1.194 0.232431
## V14          -0.00339    0.06296  -0.054 0.957058
## V15           0.31084    0.23239   1.338 0.181023
## V16           1.10038    0.16449   6.690 2.24e-11 ***
## V17           0.59641    0.13999   4.260 2.04e-05 ***
## V18          -0.02993    0.08391  -0.357 0.721327
## V19           0.15357    0.07781   1.974 0.048423 *
## V20           1.80199    0.50899   3.540 0.000400 ***
## V21           0.49973    0.08500   5.879 4.13e-09 ***
## V22           0.10473    0.15871   0.660 0.509332
## V23           1.17267    0.24101   4.866 1.14e-06 ***
## V24           0.09945    0.06169   1.612 0.106930
## V25          -3.27164    0.58150  -5.626 1.84e-08 ***
## V26          -0.44855    0.39100  -1.147 0.251312
## V27         -18.55268    3.80185  -4.880 1.06e-06 ***
## V28           0.24526    0.17081   1.436 0.151031
## V29          -2.42887    1.66214  -1.461 0.143936
## V30           0.01145    0.09666   0.118 0.905705
## V31          -0.08296    0.25709  -0.323 0.746941
## V32          -0.37441    0.95348  -0.393 0.694553
## V33          -0.46280    0.24665  -1.876 0.060610 .
## V34           0.85386    1.01167   0.844 0.398662
## V35          -0.61202    0.35339  -1.732 0.083302 .
## V36           0.07618    0.16958   0.449 0.653264
## V37          -0.26049    0.14890  -1.749 0.080214 .
## V38          -0.15147    0.12133  -1.248 0.211871
## V39          -0.02633    0.15297  -0.172 0.863349
## V40          -0.15745    0.17675  -0.891 0.373028
## V41         -18.56408   12.22870  -1.518 0.128996
```

```
## V42          -1.69535    0.58310  -2.907 0.003644 **
## V43          -0.45417    0.23919  -1.899 0.057599 .
## V44          -0.73394    0.35711  -2.055 0.039857 *
## V45          -0.88579    0.17727  -4.997 5.83e-07 ***
## V46          -1.08493    0.25513  -4.252 2.11e-05 ***
## V47          -0.64235    0.32519  -1.975 0.048234 *
## V48          -0.50262    0.38329  -1.311 0.189745
## V49          -0.20714    0.10111  -2.049 0.040502 *
## V50           0.04754    0.06007   0.791 0.428765
## V51          -0.06586    0.12898  -0.511 0.609646
## V52           0.24800    0.05813   4.266 1.99e-05 ***
## V53           1.01664    0.16220   6.268 3.66e-10 ***
## V54           0.59058    0.33572   1.759 0.078551 .
## V55          -0.56200    0.22976  -2.446 0.014445 *
## V56           1.08271    0.29373   3.686 0.000228 ***
## V57           0.61655    0.14118   4.367 1.26e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 4121.0  on 3066  degrees of freedom
## Residual deviance: 1157.4  on 3009  degrees of freedom
## AIC: 1273.4
##
## Number of Fisher Scoring iterations: 13
```

The features: 4, 5, 7, 8, 9, 11, 16, 17, 19, 20, 21, 23, 25, 27, 42, 44, 45, 46, 47, 49, 52, 53, 55, 56, 57 are statistically significant

```
pred_stdtrain <- (predict(lr_stan_train, train) > 0) * 1
pred_stdtest <- (predict(lr_stan_train, test) > 0) * 1
```

```
cm_lr_stan_train = confusionMatrix(as.factor(pred_stdtrain), as.factor(train$V58), positive =
"1")
cm_lr_stan_train
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0  446    3
##          1 1403 1215
##
##                Accuracy : 0.5416
##                  95% CI : (0.5237, 0.5593)
##     No Information Rate : 0.6029
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.1996
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.9975
##             Specificity : 0.2412
##          Pos Pred Value : 0.4641
##          Neg Pred Value : 0.9933
##              Prevalence : 0.3971
##          Detection Rate : 0.3962
##    Detection Prevalence : 0.8536
##       Balanced Accuracy : 0.6194
##
##        'Positive' Class : 1
##
```

The accuracy is 54.16% for the classification error of the Logistic Regression of standardized train data.

```
cm_lr_stan_test = confusionMatrix(as.factor(pred_stdtest), as.factor(test$V58), positive = "1")
cm_lr_stan_test
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 231   2
##          1 685 616
##
##                Accuracy : 0.5522
##                  95% CI : (0.5269, 0.5772)
##     No Information Rate : 0.5971
##     P-Value [Acc > NIR] : 0.9998
##
##                   Kappa : 0.211
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.9968
##             Specificity : 0.2522
##          Pos Pred Value : 0.4735
##          Neg Pred Value : 0.9914
##              Prevalence : 0.4029
##          Detection Rate : 0.4016
##    Detection Prevalence : 0.8481
##       Balanced Accuracy : 0.6245
##
##        'Positive' Class : 1
##
```

The accuracy is 55.22% for the classification error of the Logistic Regression of standardized test data.

Linear Regression on Log Transformation train and test data

```
lr_log_train <- glm(V58 ~ ., data = log_train, family = "binomial")
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
summary(lr_log_train)
```

```
##
## Call:
## glm(formula = V58 ~ ., family = "binomial", data = log_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -4.0831  -0.1646  -0.0010   0.0738   3.7853
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -5.55361    0.47536 -11.683  < 2e-16 ***
## V1           -0.50525    0.52078  -0.970 0.331955
## V2           -0.48375    0.41287  -1.172 0.241325
## V3           -0.34268    0.32461  -1.056 0.291122
## V4            2.49036    2.49963   0.996 0.319109
## V5            1.68052    0.26735   6.286 3.26e-10 ***
## V6            0.49007    0.49976   0.981 0.326779
## V7            3.81919    0.63656   6.000 1.98e-09 ***
## V8            1.11891    0.39254   2.850 0.004366 **
## V9            0.22162    0.61448   0.361 0.718349
## V10           0.20794    0.26664   0.780 0.435466
## V11          -1.73051    0.64790  -2.671 0.007563 **
## V12          -0.13019    0.21705  -0.600 0.548628
## V13          -1.47819    0.59699  -2.476 0.013284 *
## V14           0.49815    0.49244   1.012 0.311724
## V15           2.35454    1.31509   1.790 0.073389 .
## V16           2.00188    0.30550   6.553 5.64e-11 ***
## V17           2.00033    0.49917   4.007 6.14e-05 ***
## V18          -0.62599    0.34041  -1.839 0.065927 .
## V19           0.04966    0.17069   0.291 0.771075
## V20           4.74708    1.75988   2.697 0.006989 **
## V21           0.92793    0.20837   4.453 8.46e-06 ***
## V22           0.19783    0.59582   0.332 0.739860
## V23           3.39784    0.89163   3.811 0.000139 ***
## V24           1.27695    0.41124   3.105 0.001902 **
## V25          -3.97126    0.60152  -6.602 4.06e-11 ***
## V26          -0.43395    0.74531  -0.582 0.560401
## V27          -5.92242    1.42772  -4.148 3.35e-05 ***
## V28           1.27690    0.58913   2.167 0.030202 *
## V29          -5.52545    3.47037  -1.592 0.111344
## V30          -0.08833    0.47636  -0.185 0.852892
## V31          -1.17924    2.44793  -0.482 0.629997
## V32          -4.26131    4.43665  -0.960 0.336814
## V33          -1.44590    0.73243  -1.974 0.048368 *
## V34           0.86735    4.05419   0.214 0.830595
## V35          -2.60252    1.20495  -2.160 0.030784 *
## V36           0.44061    0.70994   0.621 0.534840
## V37          -1.55260    0.59961  -2.589 0.009615 **
## V38          -1.10219    1.36375  -0.808 0.418971
## V39           0.09940    0.80741   0.123 0.902025
## V40          -1.66152    1.14748  -1.448 0.147622
## V41         -45.30209   35.39198  -1.280 0.200542
```

```
## V42          -4.12654    1.24565  -3.313 0.000924 ***
## V43          -5.08561    1.94170  -2.619 0.008815 **
## V44          -2.90440    1.49695  -1.940 0.052354 .
## V45          -2.02986    0.41499  -4.891 1.00e-06 ***
## V46          -2.21581    0.52201  -4.245 2.19e-05 ***
## V47          -7.41904    4.88356  -1.519 0.128715
## V48          -2.02099    1.39842  -1.445 0.148405
## V49          -1.58851    0.79263  -2.004 0.045059 *
## V50          -0.01172    0.62116  -0.019 0.984945
## V51          -3.40426    2.64864  -1.285 0.198693
## V52           2.24783    0.29972   7.500 6.39e-14 ***
## V53           4.93003    0.88667   5.560 2.70e-08 ***
## V54          -0.01276    2.13277  -0.006 0.995225
## V55           0.57047    0.33492   1.703 0.088513 .
## V56           0.09317    0.19497   0.478 0.632744
## V57           0.75138    0.13167   5.707 1.15e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 4121.01  on 3066  degrees of freedom
## Residual deviance:  930.67  on 3009  degrees of freedom
## AIC: 1046.7
##
## Number of Fisher Scoring iterations: 12
```

The features: 5, 7, 8, 11, 13, 16, 17, 20, 21, 23, 24, 25, 27, 28, 33, 35, 37, 42, 43, 45, 46, 49, 52, 53, 57 are statistically significant

```
pred_logtrain <- (predict(lr_log_train, train) > 0) * 1
pred_logtest <- (predict(lr_log_train, test) > 0) * 1
```

```
cm_lr_log_train = confusionMatrix(as.factor(pred_logtrain), as.factor(train$V58), positive = "1"
)
cm_lr_log_train
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0  424    2
##          1 1425 1216
##
##                Accuracy : 0.5347
##                  95% CI : (0.5169, 0.5525)
##     No Information Rate : 0.6029
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.1898
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.9984
##             Specificity : 0.2293
##          Pos Pred Value : 0.4604
##          Neg Pred Value : 0.9953
##              Prevalence : 0.3971
##          Detection Rate : 0.3965
##    Detection Prevalence : 0.8611
##       Balanced Accuracy : 0.6138
##
##        'Positive' Class : 1
##
```

The accuracy is 53.47% for the classification error of the Logistic Regression of log transformation of train data.

```
cm_lr_log_test = confusionMatrix(as.factor(pred_logtest), as.factor(test$V58), positive = "1")
cm_lr_log_test
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 212   0
##          1 704 618
##
##                Accuracy : 0.5411
##                  95% CI : (0.5157, 0.5662)
##     No Information Rate : 0.5971
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.1953
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 1.0000
##             Specificity : 0.2314
##          Pos Pred Value : 0.4675
##          Neg Pred Value : 1.0000
##              Prevalence : 0.4029
##          Detection Rate : 0.4029
##    Detection Prevalence : 0.8618
##       Balanced Accuracy : 0.6157
##
##        'Positive' Class : 1
##
```

The accuracy is 54.11% for the classification error of the Logistic Regression of log transformation of test data.

Logistic Regression on Discretized train and test data

```
lr_I_train <- glm(V58 ~ ., data = I_train, family = "binomial")
summary(lr_I_train)
```

```
##
## Call:
## glm(formula = V58 ~ ., family = "binomial", data = I_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.6393  -0.1904  -0.0130   0.0600   3.9295
##
## Coefficients: (3 not defined because of singularities)
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -2.102414   0.189853 -11.074  < 2e-16 ***
## V1          -0.303292   0.289818  -1.046 0.295335
## V2          -0.378470   0.275804  -1.372 0.169989
## V3          -0.199095   0.212662  -0.936 0.349167
## V4           1.096282   0.824259   1.330 0.183511
## V5           1.268090   0.216147   5.867 4.44e-09 ***
## V6           0.251840   0.273000   0.922 0.356271
## V7           2.986605   0.386285   7.732 1.06e-14 ***
## V8           0.875957   0.316310   2.769 0.005618 **
## V9           0.228813   0.325213   0.704 0.481695
## V10          0.742343   0.238269   3.116 0.001836 **
## V11         -1.162239   0.334525  -3.474 0.000512 ***
## V12         -0.078381   0.194282  -0.403 0.686624
## V13         -1.161887   0.311432  -3.731 0.000191 ***
## V14          0.941421   0.452030   2.083 0.037283 *
## V15          2.006003   0.693342   2.893 0.003813 **
## V16          1.984579   0.226463   8.763  < 2e-16 ***
## V17          1.096497   0.319793   3.429 0.000606 ***
## V18         -0.857063   0.264975  -3.235 0.001219 **
## V19          0.006163   0.224878   0.027 0.978137
## V20          1.670892   0.554536   3.013 0.002586 **
## V21          0.834548   0.210275   3.969 7.22e-05 ***
## V22          0.811703   0.555363   1.462 0.143859
## V23          1.787937   0.392435   4.556 5.21e-06 ***
## V24          1.385796   0.343260   4.037 5.41e-05 ***
## V25         -3.611845   0.473164  -7.633 2.29e-14 ***
## V26         -0.640878   0.497465  -1.288 0.197646
## V27         -4.432733   0.740612  -5.985 2.16e-09 ***
## V28          1.981086   0.457457   4.331 1.49e-05 ***
## V29         -1.174992   0.668922  -1.757 0.078996 .
## V30         -0.183166   0.519469  -0.353 0.724387
## V31         -1.558298   1.033703  -1.507 0.131685
## V32         -2.211046   1.150863  -1.921 0.054706 .
## V33         -0.926369   0.562091  -1.648 0.099337 .
## V34          0.536636   1.068210   0.502 0.615408
## V35         -0.973451   0.565672  -1.721 0.085273 .
## V36          0.636619   0.417226   1.526 0.127050
## V37         -1.440826   0.348518  -4.134 3.56e-05 ***
## V38          1.173486   0.741369   1.583 0.113453
## V39          0.037749   0.413235   0.091 0.927214
## V40         -0.611572   0.557756  -1.096 0.272866
## V41         -5.823151   3.179731  -1.831 0.067051 .
```

```
## V42         -2.410825   0.508741  -4.739 2.15e-06 ***
## V43         -1.500599   0.638114  -2.352 0.018692 *
## V44         -1.301660   0.521227  -2.497 0.012514 *
## V45         -1.391117   0.235936  -5.896 3.72e-09 ***
## V46         -1.789877   0.363562  -4.923 8.52e-07 ***
## V47         -0.695873   1.130612  -0.615 0.538235
## V48         -1.512213   0.617515  -2.449 0.014331 *
## V49         -0.070815   0.275485  -0.257 0.797135
## V50          0.185424   0.196176   0.945 0.344561
## V51         -0.056805   0.409948  -0.139 0.889793
## V52          1.476322   0.186253   7.926 2.26e-15 ***
## V53          1.858618   0.250030   7.434 1.06e-13 ***
## V54         -0.794196   0.338409  -2.347 0.018933 *
## V55                NA         NA      NA       NA
## V56                NA         NA      NA       NA
## V57                NA         NA      NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 4121.0  on 3066  degrees of freedom
## Residual deviance: 1014.6  on 3012  degrees of freedom
## AIC: 1124.6
##
## Number of Fisher Scoring iterations: 9
```

The features: 5, 7, 8, 10, 11, 13, 14, 15, 16, 17, 18, 20, 21, 23, 24, 25, 27, 28, 37, 42, 43, 44, 45, 46, 48, 52, 53, 54 are statistically significant. Also have features 55, 56, 57 as NA in the summary function because these features are singularities, meaning that their respective columns are either all 0s or all 1s so cannot get a p-value from it.

```
pred_I_train <- (predict(lr_I_train, train) > 0) * 1
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from a rank-deficient fit may be misleading
```

```
pred_I_test <- (predict(lr_I_train, test) > 0) * 1
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from a rank-deficient fit may be misleading
```

```
cm_lr_I_train = confusionMatrix(as.factor(pred_I_train), as.factor(train$V58), positive = "1")
cm_lr_I_train
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction    0    1
##          0 1741  193
##          1  108 1025
##                                           
##                Accuracy : 0.9019          
##                  95% CI : (0.8908, 0.9122)
##     No Information Rate : 0.6029          
##     P-Value [Acc > NIR] : < 2.2e-16       
##                                           
##                   Kappa : 0.7926          
##                                           
##  Mcnemar's Test P-Value : 1.287e-06       
##                                           
##             Sensitivity : 0.8415          
##             Specificity : 0.9416          
##          Pos Pred Value : 0.9047          
##          Neg Pred Value : 0.9002          
##              Prevalence : 0.3971          
##          Detection Rate : 0.3342          
##    Detection Prevalence : 0.3694          
##       Balanced Accuracy : 0.8916          
##                                           
##        'Positive' Class : 1               
## 
```

The accuracy is 90.19% for the classification error of the Logistic Regression of discretize transformation of train data.

```
cm_lr_I_test = confusionMatrix(as.factor(pred_I_test), as.factor(test$V58), positive = "1")
cm_lr_I_test
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 861 103
##          1  55 515
##
##                Accuracy : 0.897
##                  95% CI : (0.8807, 0.9118)
##     No Information Rate : 0.5971
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7832
##
##  Mcnemar's Test P-Value : 0.0001847
##
##             Sensitivity : 0.8333
##             Specificity : 0.9400
##          Pos Pred Value : 0.9035
##          Neg Pred Value : 0.8932
##              Prevalence : 0.4029
##          Detection Rate : 0.3357
##    Detection Prevalence : 0.3716
##       Balanced Accuracy : 0.8866
##
##        'Positive' Class : 1
##
```

The accuracy is 89.7% for the classification error of the Logistic Regression of discretize transformation of test data.

```
cm_lr_train$overall['Accuracy']
```

```
##  Accuracy
## 0.9282687
```

```
cm_lr_test$overall['Accuracy']
```

```
##  Accuracy
## 0.9269883
```

```
cm_lr_stan_train$overall['Accuracy']
```

```
##  Accuracy
## 0.5415716
```

```
cm_lr_stan_test$overall['Accuracy']
```

```
##  Accuracy
## 0.5521512
```

```
cm_lr_log_train$overall['Accuracy']
```

```
##  Accuracy
## 0.5347245
```

```
cm_lr_log_test$overall['Accuracy']
```

```
##  Accuracy
## 0.5410691
```

```
cm_lr_I_train$overall['Accuracy']
```

```
##  Accuracy
## 0.9018585
```

```
cm_lr_I_test$overall['Accuracy']
```

```
##  Accuracy
## 0.8970013
```

```
accuracy.lr.table <- matrix( c(cm_lr_train$overall['Accuracy'], cm_lr_test$overall['Accuracy'],
 cm_lr_stan_train$overall['Accuracy'], cm_lr_stan_test$overall['Accuracy'], cm_lr_log_train$over
all['Accuracy'], cm_lr_log_test$overall['Accuracy'], cm_lr_I_train$overall['Accuracy'], cm_lr_I_
test$overall['Accuracy']), ncol=4)

accuracy.lr.table
```

```
##           [,1]      [,2]      [,3]      [,4]
## [1,] 0.9282687 0.5415716 0.5347245 0.9018585
## [2,] 0.9269883 0.5521512 0.5410691 0.8970013
```

```
colnames(accuracy.lr.table) <- c("lr original", "lr standardized", "lr log", "lr I")
rownames(accuracy.lr.table) <- c("train", "test")

accuracy.lr.table
```

```
##       lr original lr standardized    lr log      lr I
## train   0.9282687       0.5415716 0.5347245 0.9018585
## test    0.9269883       0.5521512 0.5410691 0.8970013
```

The classification accuracies are in the table above for the training and testing datasets.

5. Applying both linear and quadratic discriminant analysis methods to the standardized data, and the log transformed data.

LDA for standardized train and test Data

```
lda_train_stan <- lda(V58 ~ ., data = stan_train)
qda_train_stan <- qda(V58 ~ ., data = stan_train)
```

```
lda_predict_train_stan <- predict(lda_train_stan, stan_train)$class
lda_predict_test_stan <- predict(lda_train_stan, stan_test)$class
```

```
cm_lda_stan_test = confusionMatrix(as.factor(lda_predict_test_stan), as.factor(stan_test$V58), p
ositive = "1")
cm_lda_stan_test
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 873 115
##          1  43 503
##
##                Accuracy : 0.897
##                  95% CI : (0.8807, 0.9118)
##     No Information Rate : 0.5971
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7818
##
##  Mcnemar's Test P-Value : 1.619e-08
##
##             Sensitivity : 0.8139
##             Specificity : 0.9531
##          Pos Pred Value : 0.9212
##          Neg Pred Value : 0.8836
##              Prevalence : 0.4029
##          Detection Rate : 0.3279
##    Detection Prevalence : 0.3559
##       Balanced Accuracy : 0.8835
##
##        'Positive' Class : 1
##
```

The accuracy is 89.7% for the classification error of the lda standardized test data.

```
cm_lda_stan_train = confusionMatrix(as.factor(lda_predict_train_stan), as.factor(stan_train$V5
8), positive = "1")
cm_lda_stan_train
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1770  233
##          1   79  985
##
##                Accuracy : 0.8983
##                  95% CI : (0.887, 0.9087)
##     No Information Rate : 0.6029
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7829
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.8087
##             Specificity : 0.9573
##          Pos Pred Value : 0.9258
##          Neg Pred Value : 0.8837
##              Prevalence : 0.3971
##          Detection Rate : 0.3212
##    Detection Prevalence : 0.3469
##       Balanced Accuracy : 0.8830
##
##        'Positive' Class : 1
##
```

The accuracy is 89.83% for the classification error of the lda standardized train data.

QDA for standardized train and test data

```
qda_predict_train_stan <- predict(qda_train_stan, stan_train)$class
qda_predict_test_stan <- predict(qda_train_stan, stan_test)$class
```

```
cm_qda_stan_test = confusionMatrix(as.factor(qda_predict_test_stan), as.factor(stan_test$V58), p
ositive = "1")
cm_qda_stan_test
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 673  25
##          1 243 593
##
##                Accuracy : 0.8253
##                  95% CI : (0.8053, 0.844)
##     No Information Rate : 0.5971
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6566
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9595
##             Specificity : 0.7347
##          Pos Pred Value : 0.7093
##          Neg Pred Value : 0.9642
##              Prevalence : 0.4029
##          Detection Rate : 0.3866
##    Detection Prevalence : 0.5450
##       Balanced Accuracy : 0.8471
##
##        'Positive' Class : 1
##
```

The accuracy is 82.53% for the classification error of the qda standardized test data.

```
cm_qda_stan_train = confusionMatrix(as.factor(qda_predict_train_stan), as.factor(stan_train$V5
8), positive = "1")
cm_qda_stan_train
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1369   68
##          1  480 1150
##
##                Accuracy : 0.8213
##                  95% CI : (0.8073, 0.8347)
##     No Information Rate : 0.6029
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6472
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9442
##             Specificity : 0.7404
##          Pos Pred Value : 0.7055
##          Neg Pred Value : 0.9527
##              Prevalence : 0.3971
##          Detection Rate : 0.3750
##    Detection Prevalence : 0.5315
##       Balanced Accuracy : 0.8423
##
##        'Positive' Class : 1
##
```

The accuracy is 82.13% for the classification error of the qda standardized train data.

LDA for log transformation train and test data

```
lda_train_log <- lda(V58 ~ ., data = log_train)
qda_train_log <- qda(V58 ~ ., data = log_train)
```

```
lda_predict_train_log <- predict(lda_train_log, log_train)$class
lda_predict_test_log <- predict(lda_train_log, log_test)$class
```

```
cm_lda_log_test = confusionMatrix(as.factor(lda_predict_test_log), as.factor(log_test$V58), posi
tive = "1")
cm_lda_log_test
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 885  69
##          1  31 549
##
##                Accuracy : 0.9348
##                  95% CI : (0.9213, 0.9466)
##     No Information Rate : 0.5971
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8631
##
##  Mcnemar's Test P-Value : 0.0002156
##
##             Sensitivity : 0.8883
##             Specificity : 0.9662
##          Pos Pred Value : 0.9466
##          Neg Pred Value : 0.9277
##              Prevalence : 0.4029
##          Detection Rate : 0.3579
##    Detection Prevalence : 0.3781
##       Balanced Accuracy : 0.9273
##
##        'Positive' Class : 1
##
```

The accuracy is 93.48% for the classification error of the lda log test data.

```
cm_lda_log_train = confusionMatrix(as.factor(lda_predict_train_log), as.factor(log_train$V58), p
ositive = "1")
cm_lda_log_train
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1795  131
##          1   54 1087
##
##                Accuracy : 0.9397
##                  95% CI : (0.9307, 0.9478)
##     No Information Rate : 0.6029
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8727
##
##  Mcnemar's Test P-Value : 2.302e-08
##
##             Sensitivity : 0.8924
##             Specificity : 0.9708
##          Pos Pred Value : 0.9527
##          Neg Pred Value : 0.9320
##              Prevalence : 0.3971
##          Detection Rate : 0.3544
##    Detection Prevalence : 0.3720
##       Balanced Accuracy : 0.9316
##
##        'Positive' Class : 1
##
```

The accuracy is 93.48% for the classification error of the lda log train data.

QDA for log transformation train and test data

```
qda_predict_train_log <- predict(qda_train_log, log_train)$class
qda_predict_test_log <- predict(qda_train_log, log_test)$class
```

```
cm_qda_log_test = confusionMatrix(as.factor(qda_predict_test_log), as.factor(log_test$V58), posi
tive = "1")
cm_qda_log_test
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 702  27
##          1 214 591
##
##                Accuracy : 0.8429
##                  95% CI : (0.8237, 0.8608)
##     No Information Rate : 0.5971
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6888
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9563
##             Specificity : 0.7664
##          Pos Pred Value : 0.7342
##          Neg Pred Value : 0.9630
##              Prevalence : 0.4029
##          Detection Rate : 0.3853
##    Detection Prevalence : 0.5248
##       Balanced Accuracy : 0.8613
##
##        'Positive' Class : 1
##
```

The accuracy is 84.29% for the classification error of the qda log test data.

```
cm_qda_log_train = confusionMatrix(as.factor(qda_predict_train_log), as.factor(log_train$V58), p
ositive = "1")
cm_qda_log_train
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1433   71
##          1  416 1147
##
##                Accuracy : 0.8412
##                  95% CI : (0.8278, 0.854)
##     No Information Rate : 0.6029
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6837
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9417
##             Specificity : 0.7750
##          Pos Pred Value : 0.7338
##          Neg Pred Value : 0.9528
##              Prevalence : 0.3971
##          Detection Rate : 0.3740
##    Detection Prevalence : 0.5096
##       Balanced Accuracy : 0.8584
##
##        'Positive' Class : 1
##
```

The accuracy is 84.12% for the classification error of the qda log train data.

```
cm_lda_stan_train$overall['Accuracy']
```

```
##  Accuracy
## 0.8982719
```

```
cm_lda_stan_test$overall['Accuracy']
```

```
##  Accuracy
## 0.8970013
```

```
cm_lda_log_train$overall['Accuracy']
```

```
##  Accuracy
## 0.9396805
```

```
cm_lda_log_test$overall['Accuracy']
```

```
## Accuracy
## 0.934811
```

```
cm_qda_stan_train$overall['Accuracy']
```

```
##  Accuracy
## 0.8213238
```

```
cm_qda_stan_test$overall['Accuracy']
```

```
##  Accuracy
## 0.8252934
```

```
cm_qda_log_train$overall['Accuracy']
```

```
##  Accuracy
## 0.8412129
```

```
cm_qda_log_test$overall['Accuracy']
```

```
##  Accuracy
## 0.8428944
```

```
accuracy.da.table <- matrix( c(cm_lda_stan_train$overall['Accuracy'], cm_lda_stan_test$overall[
'Accuracy'], cm_lda_log_train$overall['Accuracy'], cm_lda_log_test$overall['Accuracy'], cm_qda_s
tan_train$overall['Accuracy'], cm_qda_stan_test$overall['Accuracy'], cm_qda_log_train$overall['A
ccuracy'], cm_qda_log_test$overall['Accuracy']), ncol=4)

accuracy.da.table
```

```
##           [,1]      [,2]      [,3]      [,4]
## [1,] 0.8982719 0.9396805 0.8213238 0.8412129
## [2,] 0.8970013 0.9348110 0.8252934 0.8428944
```

```
colnames(accuracy.da.table) <- c("lda stan", "lda log", "qda stan", "qda log")
rownames(accuracy.da.table) <- c("train", "test")

accuracy.da.table
```

```
##        lda stan   lda log  qda stan   qda log
## train 0.8982719 0.9396805 0.8213238 0.8412129
## test  0.8970013 0.9348110 0.8252934 0.8428944
```

For all of the above, LDA and QDA for standardized and log transformed data on both test and train data sets, the LDA performed better than the QDA.

6. Applying linear and nonlinear support vector machine classifiers to each version of the data.

```
library(e1071)
```

Linear SVM for original data

```
new_train = train
new_train$V58 = as.factor(new_train$V58)
new_test = test
new_test$V58 = as.factor(new_test$V58)
```

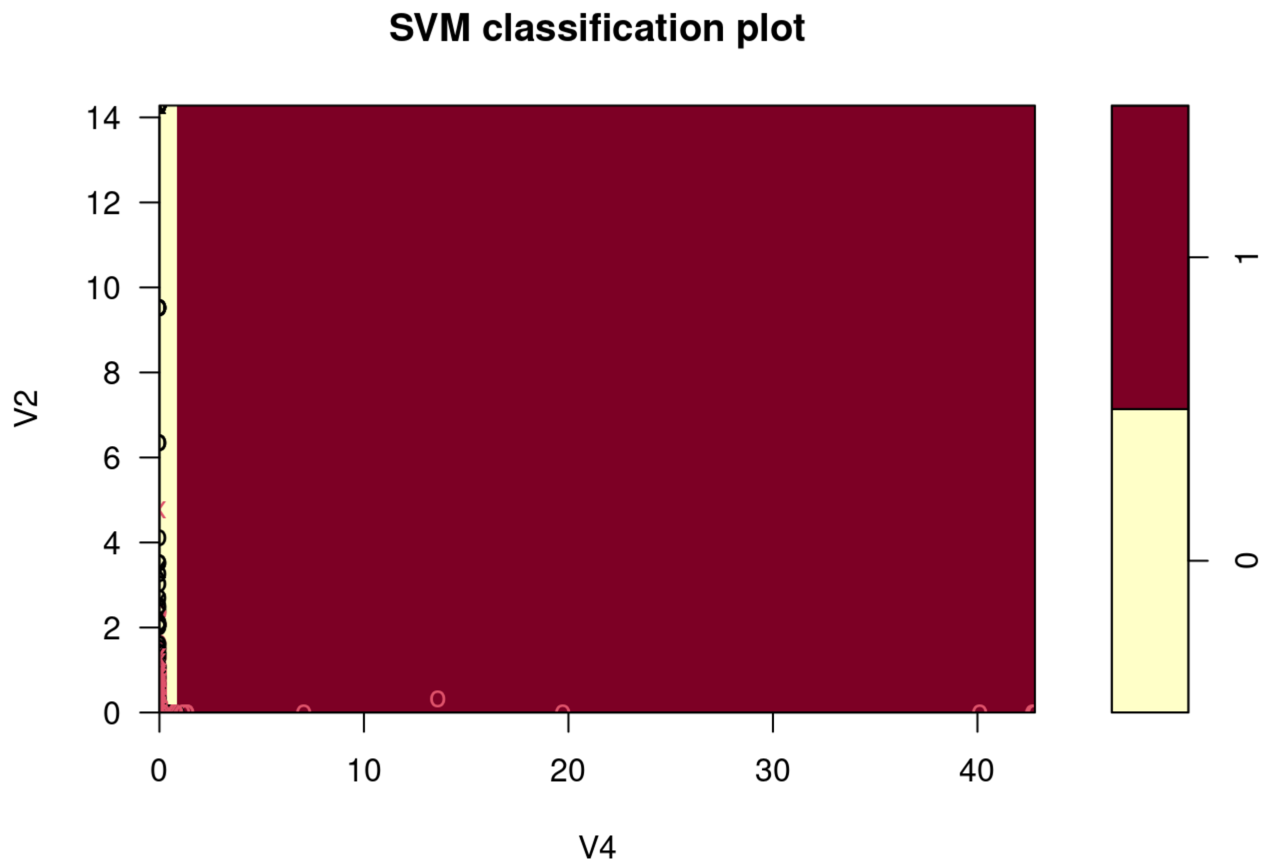
```
tune.linear = tune(svm, V58 ~ ., data = new_train, kernel = "linear", ranges = list(cost=c(0.001
, 0.01, 0.1, 1, 5, 10)), validation.x = new_test)
```

```
summary(tune.linear)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost
##    10
##
## - best performance: 0.07238402
##
## - Detailed performance results:
##    cost      error dispersion
## 1 1e-03 0.10954844 0.02793485
## 2 1e-02 0.08151200 0.01646617
## 3 1e-01 0.07760001 0.01898033
## 4 1e+00 0.07466841 0.02357943
## 5 5e+00 0.07271189 0.02376119
## 6 1e+01 0.07238402 0.02318753
```

```
train_svm <- svm(V58 ~ ., kernel = "linear", data=new_train, cost=10)
```

```
plot(train_svm, new_train, formula = V2 ~ V4)
```




```
svm_predict_train <- predict(train_svm,new_train)
```

```
cm_linear_train <- confusionMatrix(as.factor(svm_predict_train), as.factor(new_train$V58), posit
ive = "1")
cm_linear_train
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1775  118
##          1   74 1100
##
##                Accuracy : 0.9374
##                  95% CI : (0.9282, 0.9457)
##     No Information Rate : 0.6029
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8685
##
##  Mcnemar's Test P-Value : 0.001914
##
##             Sensitivity : 0.9031
##             Specificity : 0.9600
##          Pos Pred Value : 0.9370
##          Neg Pred Value : 0.9377
##              Prevalence : 0.3971
##          Detection Rate : 0.3587
##    Detection Prevalence : 0.3828
##       Balanced Accuracy : 0.9315
##
##        'Positive' Class : 1
##
```

The accuracy is 93.74% for the linear svm classifier of the train data.

```
svm_predict_test <- predict(train_svm,new_test)
```

```
cm_linear_test <- confusionMatrix(as.factor(svm_predict_test), as.factor(new_test$V58), positive
= "1")
cm_linear_test
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction   0   1
##          0 876  61
##          1  40 557
##                                           
##                Accuracy : 0.9342          
##                  95% CI : (0.9206, 0.9461)
##     No Information Rate : 0.5971          
##     P-Value [Acc > NIR] : < 2e-16         
##                                           
##                   Kappa : 0.8624          
##                                           
##  Mcnemar's Test P-Value : 0.04658         
##                                           
##             Sensitivity : 0.9013          
##             Specificity : 0.9563          
##          Pos Pred Value : 0.9330          
##          Neg Pred Value : 0.9349          
##              Prevalence : 0.4029          
##          Detection Rate : 0.3631          
##    Detection Prevalence : 0.3892          
##       Balanced Accuracy : 0.9288          
##                                           
##        'Positive' Class : 1               
## 
```

The accuracy is 93.42% for the linear svm classifier of the test data.

Linear SVM for standardized data

```
new_stan_train = stan_train
new_stan_train$V58 = as.factor(new_stan_train$V58)
```

```
new_stan_test = stan_test
new_stan_test$V58 = as.factor(new_stan_test$V58)
```

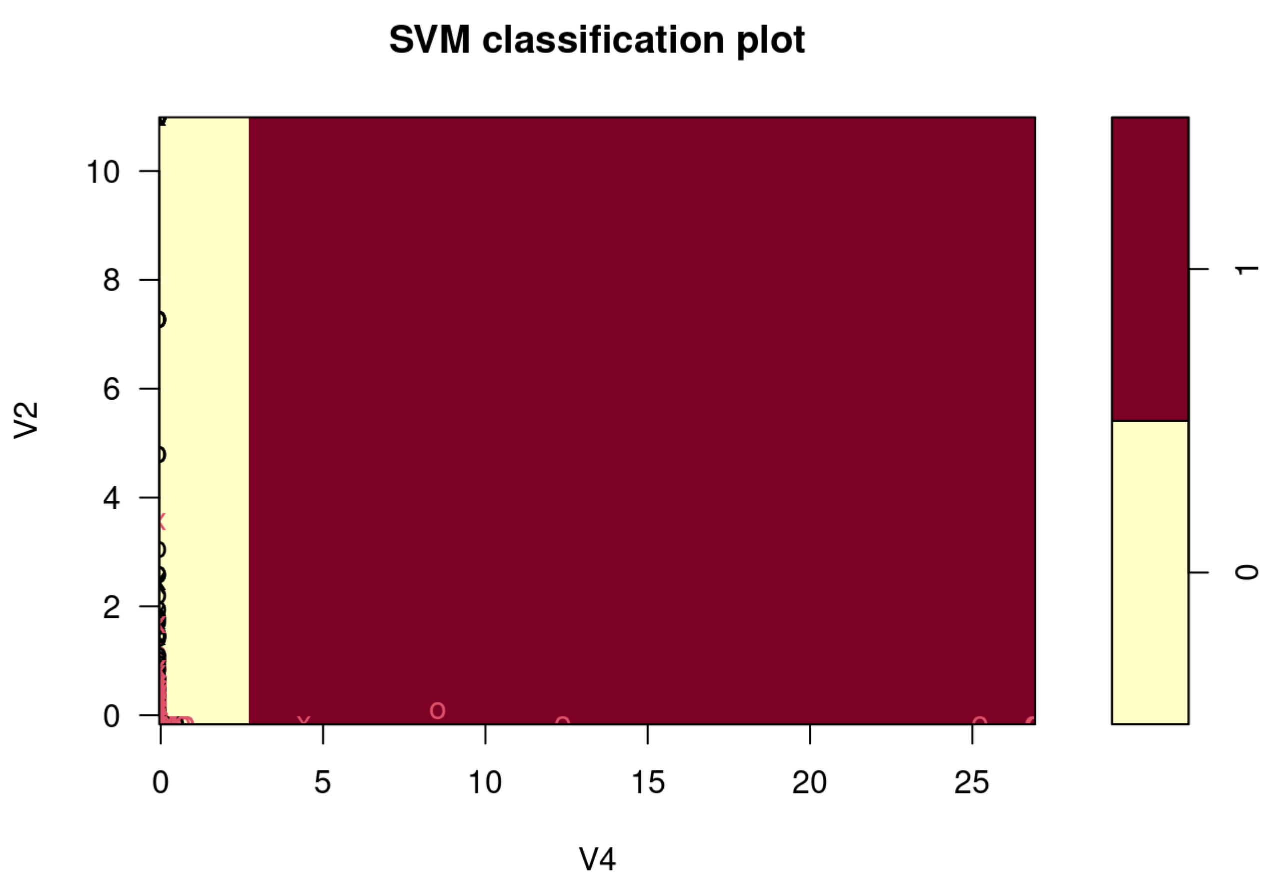
```
tune.linear = tune(svm, V58 ~ ., data = new_stan_train, kernel = "linear", ranges = list(cost=c(
0.001, 0.01, 0.1, 1, 5, 10)), validation.x = new_stan_test)
```

```
summary(tune.linear)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost
##     1
##
## - best performance: 0.07368802
##
## - Detailed performance results:
##    cost      error dispersion
## 1 1e-03 0.11020204 0.01214033
## 2 1e-02 0.08444892 0.01600749
## 3 1e-01 0.07825360 0.01521989
## 4 1e+00 0.07368802 0.01279367
## 5 5e+00 0.07401482 0.01435890
## 6 1e+01 0.07368802 0.01163016
```

```
stan_train_svm <- svm(V58 ~ ., kernel = "linear", data=new_stan_train, cost=0.1)
```

```
plot(stan_train_svm, new_stan_train, formula = V2 ~ V4)
```



```
svm_predict_train_stan <- predict(stan_train_svm,new_stan_train)
```

```
cm_linear_stan_train <- confusionMatrix(as.factor(svm_predict_train_stan), as.factor(new_stan_tr
ain$V58), positive = "1")
cm_linear_stan_train
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1768  142
##          1   81 1076
##
##                Accuracy : 0.9273
##                  95% CI : (0.9175, 0.9362)
##     No Information Rate : 0.6029
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8468
##
##  Mcnemar's Test P-Value : 5.872e-05
##
##             Sensitivity : 0.8834
##             Specificity : 0.9562
##          Pos Pred Value : 0.9300
##          Neg Pred Value : 0.9257
##              Prevalence : 0.3971
##          Detection Rate : 0.3508
##    Detection Prevalence : 0.3772
##       Balanced Accuracy : 0.9198
##
##        'Positive' Class : 1
##
```

The accuracy is 92.73% for the linear svm classifier of the standardized train data.

```
svm_predict_test_stan <- predict(stan_train_svm,new_stan_test)
```

```
cm_linear_stan_test <- confusionMatrix(as.factor(svm_predict_test_stan), as.factor(new_stan_test
$V58), positive = "1")
cm_linear_stan_test
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 875  63
##          1  41 555
##
##                Accuracy : 0.9322
##                  95% CI : (0.9184, 0.9443)
##     No Information Rate : 0.5971
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.8583
##
##  Mcnemar's Test P-Value : 0.03947
##
##             Sensitivity : 0.8981
##             Specificity : 0.9552
##          Pos Pred Value : 0.9312
##          Neg Pred Value : 0.9328
##              Prevalence : 0.4029
##          Detection Rate : 0.3618
##    Detection Prevalence : 0.3885
##       Balanced Accuracy : 0.9266
##
##        'Positive' Class : 1
##
```

The accuracy is 93.22% for the linear svm classifier of the standardized test data.

Linear SVM for log transformed data

```
new_log_train = log_train
new_log_train$V58 = as.factor(new_log_train$V58)
```

```
new_log_test = log_test
new_log_test$V58 = as.factor(new_log_test$V58)
```

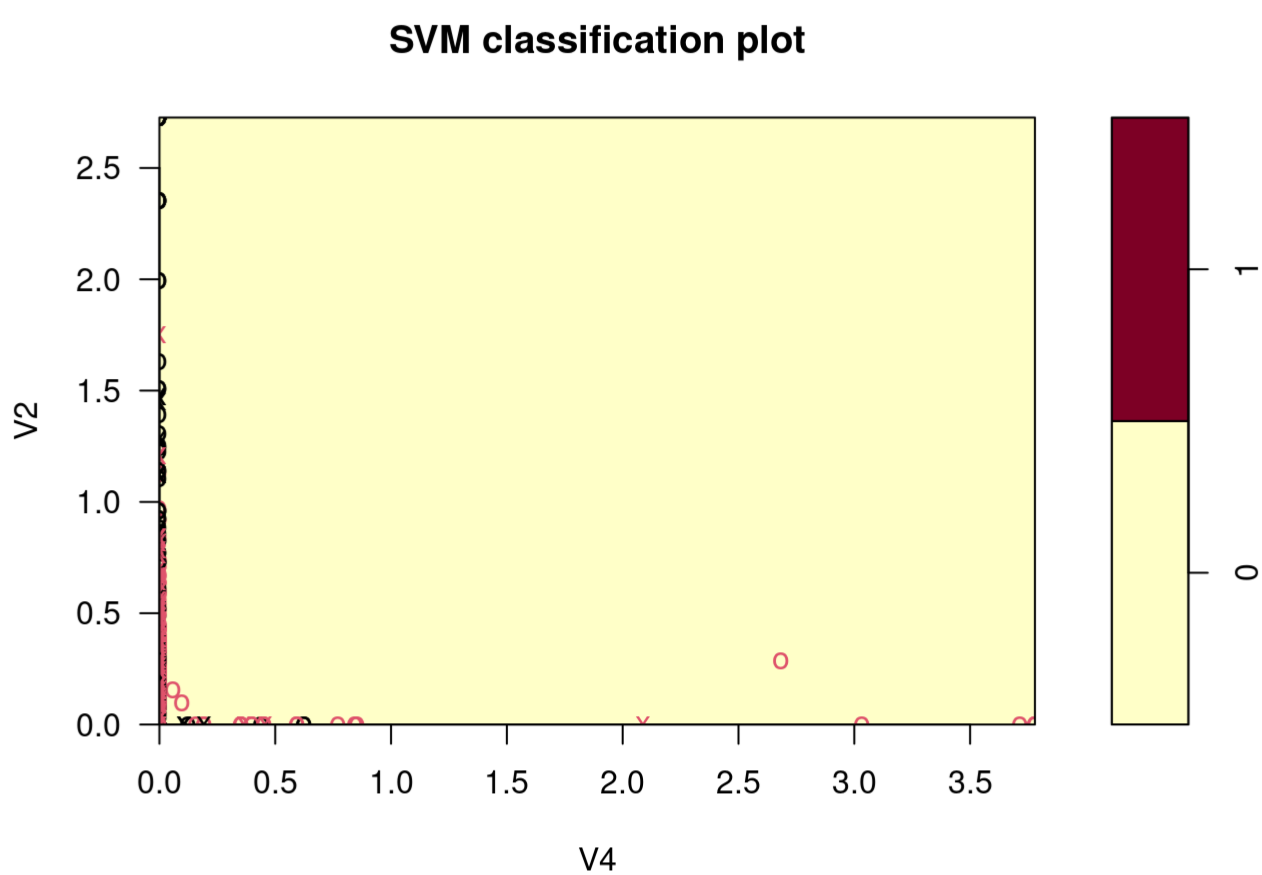
```
tune.linear = tune(svm, V58 ~ ., data = new_log_train, kernel = "linear", ranges = list(cost=c(
0.001, 0.01, 0.1, 1, 5, 10)), validation.x = new_log_test)
```

```
summary(tune.linear)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost
##  0.01
##
## - best performance: 0.05771327
##
## - Detailed performance results:
##    cost      error  dispersion
## 1 1e-03 0.06912563 0.011083682
## 2 1e-02 0.05771327 0.008587184
## 3 1e-01 0.06325499 0.011116892
## 4 1e+00 0.06455898 0.010968706
## 5 5e+00 0.06325605 0.010471460
## 6 1e+01 0.06325605 0.010471460
```

```
log_train_svm <- svm(V58 ~ ., kernel = "linear", data=new_log_train, cost=0.01)
```

```
plot(log_train_svm, new_log_train, formula = V2 ~ V4)
```



```r
svm_predict_train_log <- predict(log_train_svm,new_log_train)
```

```r
cm_linear_log_train <- confusionMatrix(as.factor(svm_predict_train_log), as.factor(new_log_train
$V58), positive = "1")
cm_linear_log_train
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction    0    1
##          0 1777  109
##          1   72 1109
## 
##                Accuracy : 0.941
##                  95% CI : (0.9321, 0.9491)
##     No Information Rate : 0.6029
##     P-Value [Acc > NIR] : < 2.2e-16
## 
##                   Kappa : 0.8761
## 
##  Mcnemar's Test P-Value : 0.007454
## 
##             Sensitivity : 0.9105
##             Specificity : 0.9611
##          Pos Pred Value : 0.9390
##          Neg Pred Value : 0.9422
##              Prevalence : 0.3971
##          Detection Rate : 0.3616
##    Detection Prevalence : 0.3851
##       Balanced Accuracy : 0.9358
## 
##        'Positive' Class : 1
## 
```

The linear svm accuracy for the log transformed train data is 94.10%.

```r
svm_predict_test_log <- predict(log_train_svm,new_log_test)
```

```r
cm_linear_log_test <- confusionMatrix(as.factor(svm_predict_test_log), as.factor(new_log_test$V5
8), positive = "1")
cm_linear_log_test
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 880  53
##          1  36 565
##
##                Accuracy : 0.942
##                  95% CI : (0.9291, 0.9532)
##     No Information Rate : 0.5971
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.8789
##
##  Mcnemar's Test P-Value : 0.08989
##
##             Sensitivity : 0.9142
##             Specificity : 0.9607
##          Pos Pred Value : 0.9401
##          Neg Pred Value : 0.9432
##              Prevalence : 0.4029
##          Detection Rate : 0.3683
##    Detection Prevalence : 0.3918
##       Balanced Accuracy : 0.9375
##
##        'Positive' Class : 1
##
```

The linear svm accuracy for the log transformed test data is 94.20%.

Linear SVM for Discretized I data

```
new_I_train = I_train
new_I_train$V58 = as.factor(new_I_train$V58)
```

```
new_I_test = I_test
new_I_test$V58 = as.factor(new_I_test$V58)
```

```
tune.linear = tune(svm, V58 ~ ., data = new_I_train, kernel = "linear", ranges = list(cost=c(0.0
01, 0.01, 0.1, 1, 5, 10)), validation.x = new_I_test)
```

```
## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
```

```
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.
```

```
## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.
```

```
## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.
```

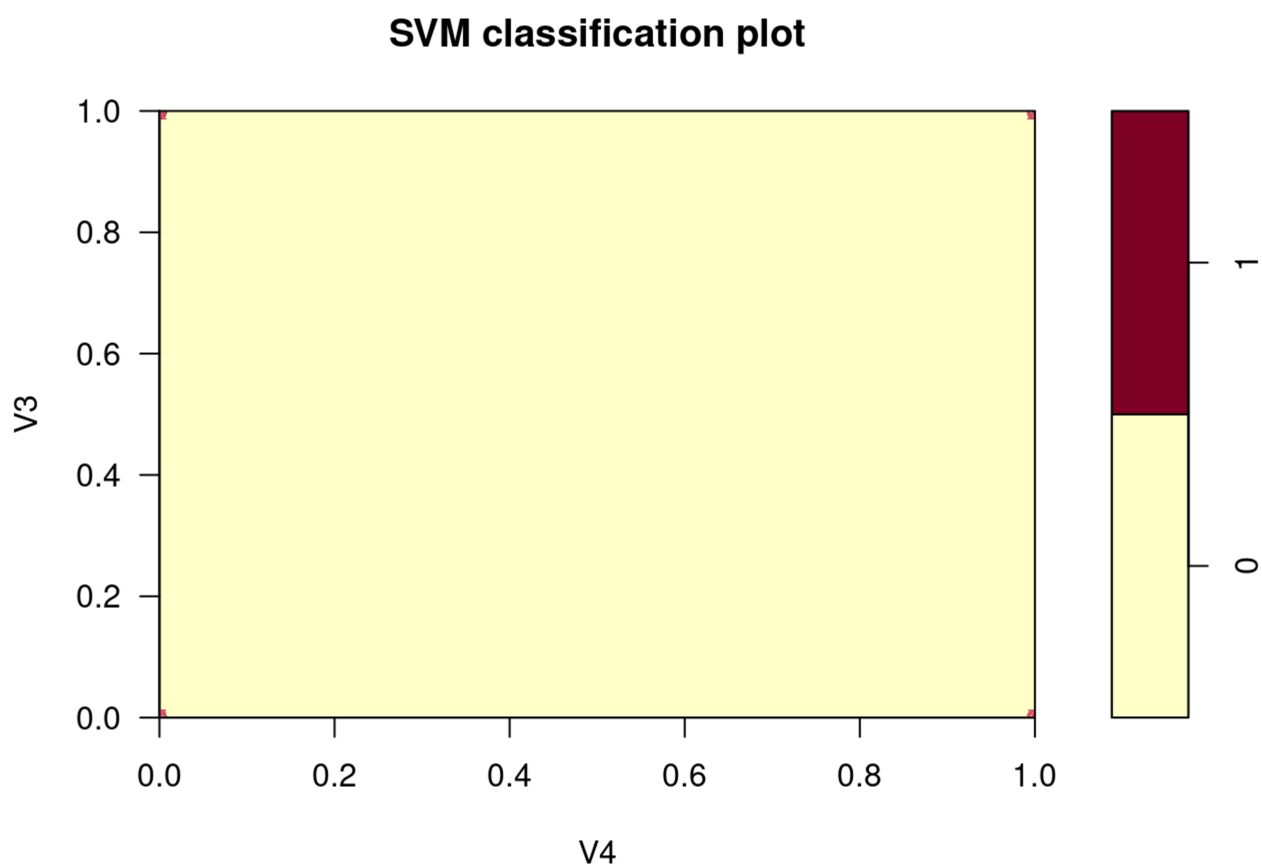
```
summary(tune.linear)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost
##     1
##
## - best performance: 0.06813779
##
## - Detailed performance results:
##    cost      error dispersion
## 1 1e-03 0.13075089 0.01864518
## 2 1e-02 0.07629708 0.01152786
## 3 1e-01 0.07042005 0.01380397
## 4 1e+00 0.06813779 0.01330741
## 5 5e+00 0.07075004 0.01535053
## 6 1e+01 0.07140257 0.01426862
```

```
I_train_svm <- svm(V58 ~ ., kernel = "linear", data=new_I_train, cost=5)
```

```
## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.
```

```
plot(I_train_svm, new_I_train, formula = V3 ~ V4)
```




```
svm_predict_train_I <- predict(I_train_svm,new_I_train)
```

```
cm_linear_I_train <- confusionMatrix(as.factor(svm_predict_train_I), as.factor(new_I_train$V58),
positive = "1")
cm_linear_I_train
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1776  107
##          1   73 1111
##
##                Accuracy : 0.9413
##                  95% CI : (0.9324, 0.9494)
##     No Information Rate : 0.6029
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.8768
##
##  Mcnemar's Test P-Value : 0.01391
##
##             Sensitivity : 0.9122
##             Specificity : 0.9605
##          Pos Pred Value : 0.9383
##          Neg Pred Value : 0.9432
##              Prevalence : 0.3971
##          Detection Rate : 0.3622
##    Detection Prevalence : 0.3860
##       Balanced Accuracy : 0.9363
##
##        'Positive' Class : 1
##
```

The linear SVM accuracy for the discretized train data is 94.13%.

```
svm_predict_test_I <- predict(I_train_svm,new_I_test)
```

```
cm_linear_I_test <- confusionMatrix(as.factor(svm_predict_test_I), as.factor(new_I_test$V58), po
sitive = "1")
cm_linear_I_test
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 864  64
##          1  52 554
##
##                Accuracy : 0.9244
##                  95% CI : (0.91, 0.9371)
##     No Information Rate : 0.5971
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.8423
##
##  Mcnemar's Test P-Value : 0.3071
##
##             Sensitivity : 0.8964
##             Specificity : 0.9432
##          Pos Pred Value : 0.9142
##          Neg Pred Value : 0.9310
##              Prevalence : 0.4029
##          Detection Rate : 0.3611
##    Detection Prevalence : 0.3950
##       Balanced Accuracy : 0.9198
##
##        'Positive' Class : 1
##
```

The linear SVM accuracy for the discretized test data is 92.44%.

Gaussian SVM for original data

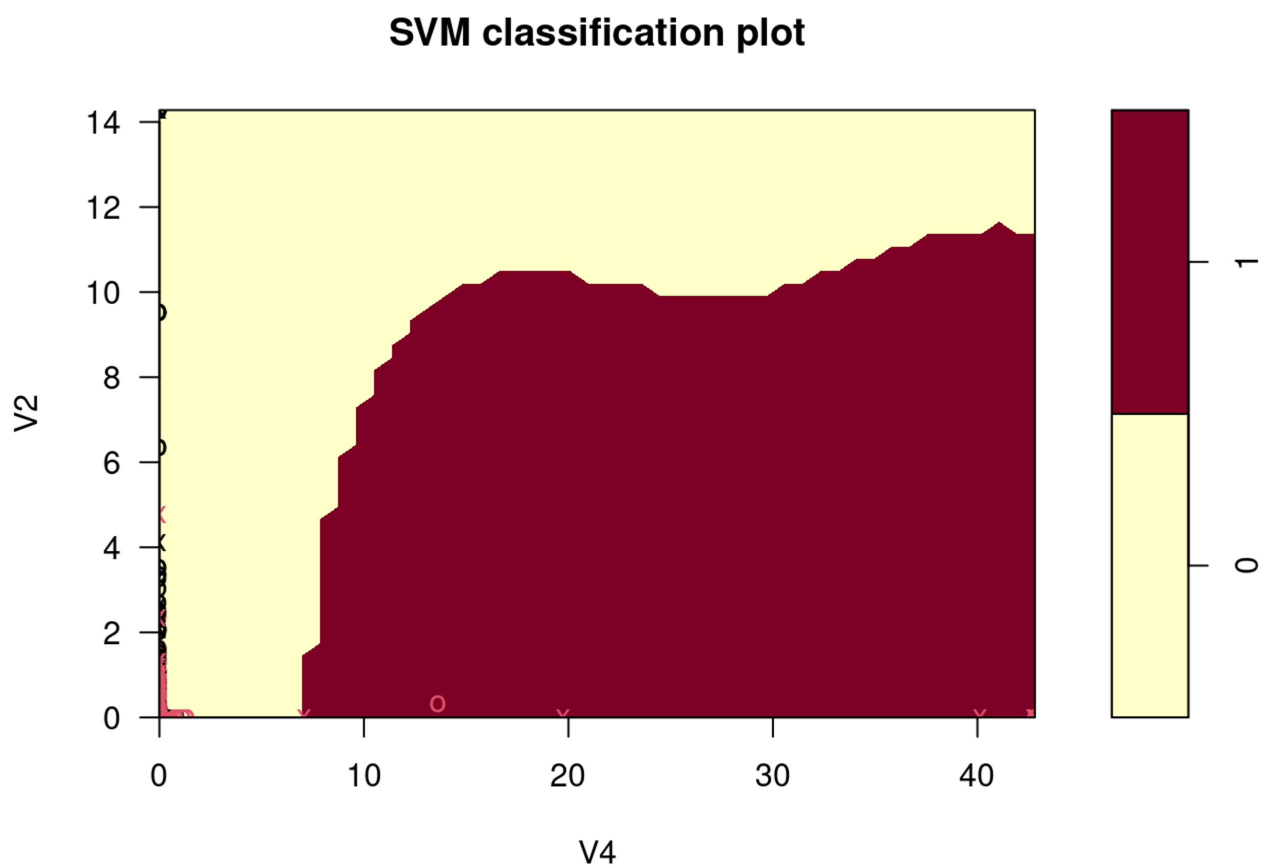
```
tune.gaussian = tune(svm, V58 ~ ., data = new_train, kernel = "radial", ranges = list(cost=c(0.0
01, 0.01, 0.1, 1, 5, 10),
                                                                                       gamma = c(
0.001, 0.01, 0.1, 1)), validation.x = new_test)
```

```
summary(tune.gaussian)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost gamma
##    10  0.01
##
## - best performance: 0.05803687
##
## - Detailed performance results:
##     cost gamma      error  dispersion
## 1  1e-03 0.001 0.39712056 0.028732231
## 2  1e-02 0.001 0.39712056 0.028732231
## 3  1e-01 0.001 0.20738541 0.027120756
## 4  1e+00 0.001 0.08771476 0.018238630
## 5  5e+00 0.001 0.07532946 0.016637956
## 6  1e+01 0.001 0.07108535 0.015684356
## 7  1e-03 0.010 0.39712056 0.028732231
## 8  1e-02 0.010 0.37332077 0.036749828
## 9  1e-01 0.010 0.08934449 0.020821958
## 10 1e+00 0.010 0.06651445 0.013668092
## 11 5e+00 0.010 0.05901194 0.011312565
## 12 1e+01 0.010 0.05803687 0.009807594
## 13 1e-03 0.100 0.39712056 0.028732231
## 14 1e-02 0.100 0.39712056 0.028732231
## 15 1e-01 0.100 0.18943391 0.030241220
## 16 1e+00 0.100 0.06911711 0.009154609
## 17 5e+00 0.100 0.06357114 0.009447035
## 18 1e+01 0.100 0.06259288 0.010111047
## 19 1e-03 1.000 0.39712056 0.028732231
## 20 1e-02 1.000 0.39712056 0.028732231
## 21 1e-01 1.000 0.37788848 0.031612823
## 22 1e+00 1.000 0.13008665 0.013522564
## 23 5e+00 1.000 0.12747972 0.013184899
## 24 1e+01 1.000 0.12682826 0.012828926
```

```
train_svm <- svm(V58 ~ ., kernel = "radial", data=new_train, cost=10, gamma =0.01)
```

```
plot(train_svm, new_train, formula = V2 ~ V4)
```




```
svm_predict_train <- predict(train_svm,new_train)
```

```
cm_gaussian_train <- confusionMatrix(as.factor(svm_predict_train), as.factor(new_train$V58), pos
itive = "1")
cm_gaussian_train
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1811   76
##          1   38 1142
##
##                Accuracy : 0.9628
##                  95% CI : (0.9555, 0.9692)
##     No Information Rate : 0.6029
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.922
##
##  Mcnemar's Test P-Value : 0.0005295
##
##             Sensitivity : 0.9376
##             Specificity : 0.9794
##          Pos Pred Value : 0.9678
##          Neg Pred Value : 0.9597
##              Prevalence : 0.3971
##          Detection Rate : 0.3724
##    Detection Prevalence : 0.3847
##       Balanced Accuracy : 0.9585
##
##        'Positive' Class : 1
##
```

The accuracy is 96.28% for the guassian svm classifier of the train data.

```
svm_predict_test <- predict(train_svm,new_test)
```

```
cm_gaussian_test <- confusionMatrix(as.factor(svm_predict_test), as.factor(new_test$V58), positi
ve = "1")
cm_gaussian_test
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 886  54
##          1  30 564
##
##                Accuracy : 0.9452
##                  95% CI : (0.9327, 0.9561)
##     No Information Rate : 0.5971
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.8855
##
##  Mcnemar's Test P-Value : 0.01209
##
##             Sensitivity : 0.9126
##             Specificity : 0.9672
##          Pos Pred Value : 0.9495
##          Neg Pred Value : 0.9426
##              Prevalence : 0.4029
##          Detection Rate : 0.3677
##    Detection Prevalence : 0.3872
##       Balanced Accuracy : 0.9399
##
##        'Positive' Class : 1
##
```

The accuracy is 94.52% for the Guassian svm classifier of the test data.

Gaussian SVM for standardized data

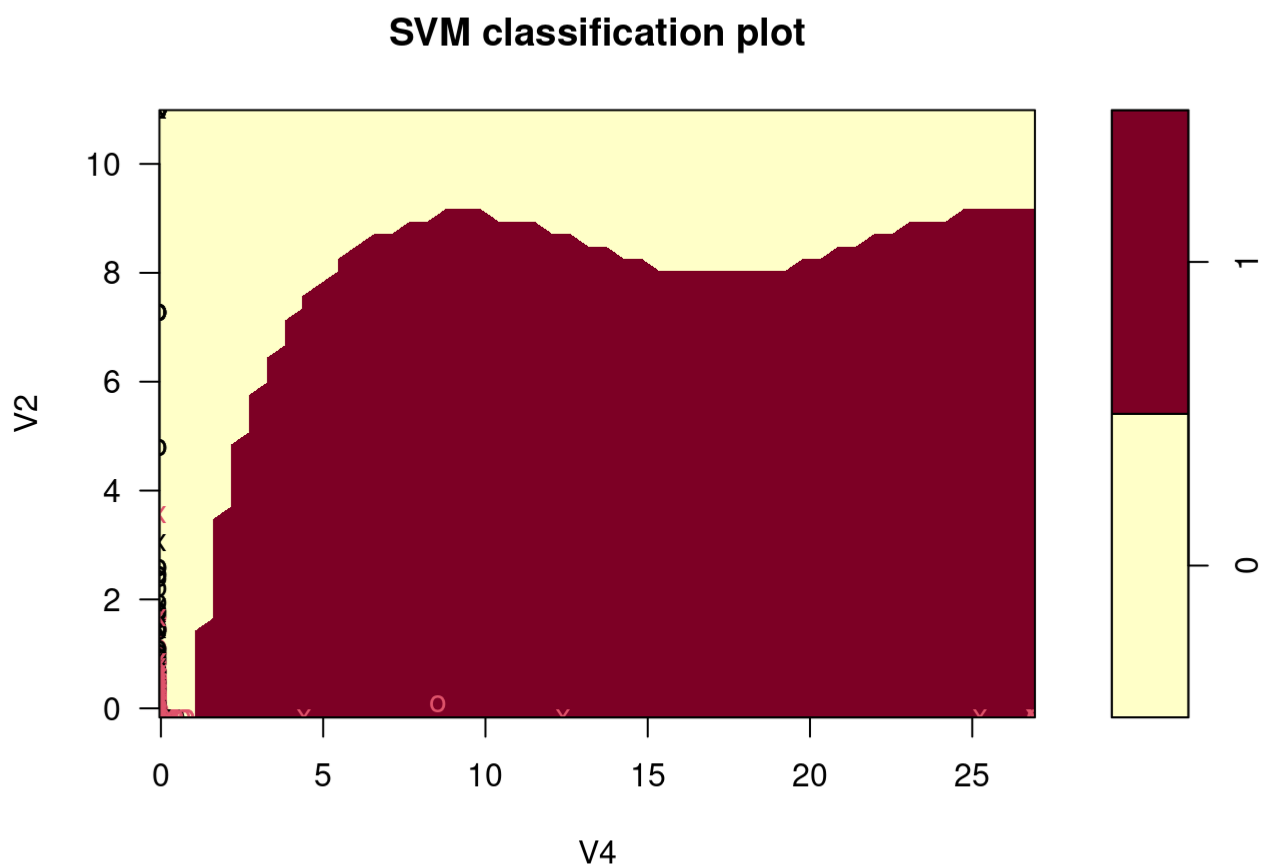
```
tune.gaussian = tune(svm, V58 ~ ., data = new_stan_train, kernel = "radial", ranges = list(cost=
c(0.001, 0.01, 0.1, 1, 5, 10),
                                                                                     gamma = c(
0.001, 0.01, 0.1, 1)), validation.x = new_stan_test)
```

```
summary(tune.gaussian)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost gamma
##    10  0.01
##
## - best performance: 0.05641353
##
## - Detailed performance results:
##     cost gamma      error dispersion
## 1  1e-03 0.001 0.39714079 0.01933927
## 2  1e-02 0.001 0.39714079 0.01933927
## 3  1e-01 0.001 0.20606225 0.02469546
## 4  1e+00 0.001 0.08640544 0.01402377
## 5  5e+00 0.001 0.07304188 0.01872841
## 6  1e+01 0.001 0.07076281 0.01695256
## 7  1e-03 0.010 0.39714079 0.01933927
## 8  1e-02 0.010 0.37105767 0.02260411
## 9  1e-01 0.010 0.08706755 0.01724684
## 10 1e+00 0.010 0.06619936 0.01892769
## 11 5e+00 0.010 0.05836793 0.01828958
## 12 1e+01 0.010 0.05641353 0.01713466
## 13 1e-03 0.100 0.39714079 0.01933927
## 14 1e-02 0.100 0.39714079 0.01933927
## 15 1e-01 0.100 0.18912308 0.02581312
## 16 1e+00 0.100 0.07369121 0.01576950
## 17 5e+00 0.100 0.07075749 0.01739920
## 18 1e+01 0.100 0.07010709 0.01934572
## 19 1e-03 1.000 0.39714079 0.01933927
## 20 1e-02 1.000 0.39714079 0.01933927
## 21 1e-01 1.000 0.37627046 0.01934698
## 22 1e+00 1.000 0.13401993 0.02095515
## 23 5e+00 1.000 0.13206234 0.01916641
## 24 1e+01 1.000 0.13206234 0.01916641
```

```
stan_train_svm <- svm(V58 ~ ., kernel = "radial", data=new_stan_train, cost=10, gamma = 0.01)
```

```
plot(stan_train_svm, new_stan_train, formula = V2 ~ V4)
```




```
svm_predict_train_stan <- predict(stan_train_svm,new_stan_train)
```

```
cm_gaussian_stan_train <- confusionMatrix(as.factor(svm_predict_train_stan), as.factor(new_stan_
train$V58), positive = "1")
cm_gaussian_stan_train
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1811   76
##          1   38 1142
##
##                Accuracy : 0.9628
##                  95% CI : (0.9555, 0.9692)
##     No Information Rate : 0.6029
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.922
##
##  Mcnemar's Test P-Value : 0.0005295
##
##             Sensitivity : 0.9376
##             Specificity : 0.9794
##          Pos Pred Value : 0.9678
##          Neg Pred Value : 0.9597
##              Prevalence : 0.3971
##          Detection Rate : 0.3724
##    Detection Prevalence : 0.3847
##       Balanced Accuracy : 0.9585
##
##        'Positive' Class : 1
##
```

The accuracy is 96.28% for the guassian svm classifier of the standardized train data.

```
svm_predict_test_stan <- predict(stan_train_svm,new_stan_test)
```

```
cm_gaussian_stan_test <- confusionMatrix(as.factor(svm_predict_test_stan), as.factor(new_stan_te
st$V58), positive = "1")
cm_gaussian_stan_test
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 884  58
##          1  32 560
##
##                Accuracy : 0.9413
##                  95% CI : (0.9284, 0.9526)
##     No Information Rate : 0.5971
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8772
##
##  Mcnemar's Test P-Value : 0.008408
##
##             Sensitivity : 0.9061
##             Specificity : 0.9651
##          Pos Pred Value : 0.9459
##          Neg Pred Value : 0.9384
##              Prevalence : 0.4029
##          Detection Rate : 0.3651
##    Detection Prevalence : 0.3859
##       Balanced Accuracy : 0.9356
##
##        'Positive' Class : 1
##
```

The accuracy is 94.13% for the gaussian svm classifier of the standardized test data.

Gaussian SVM for log transformed data

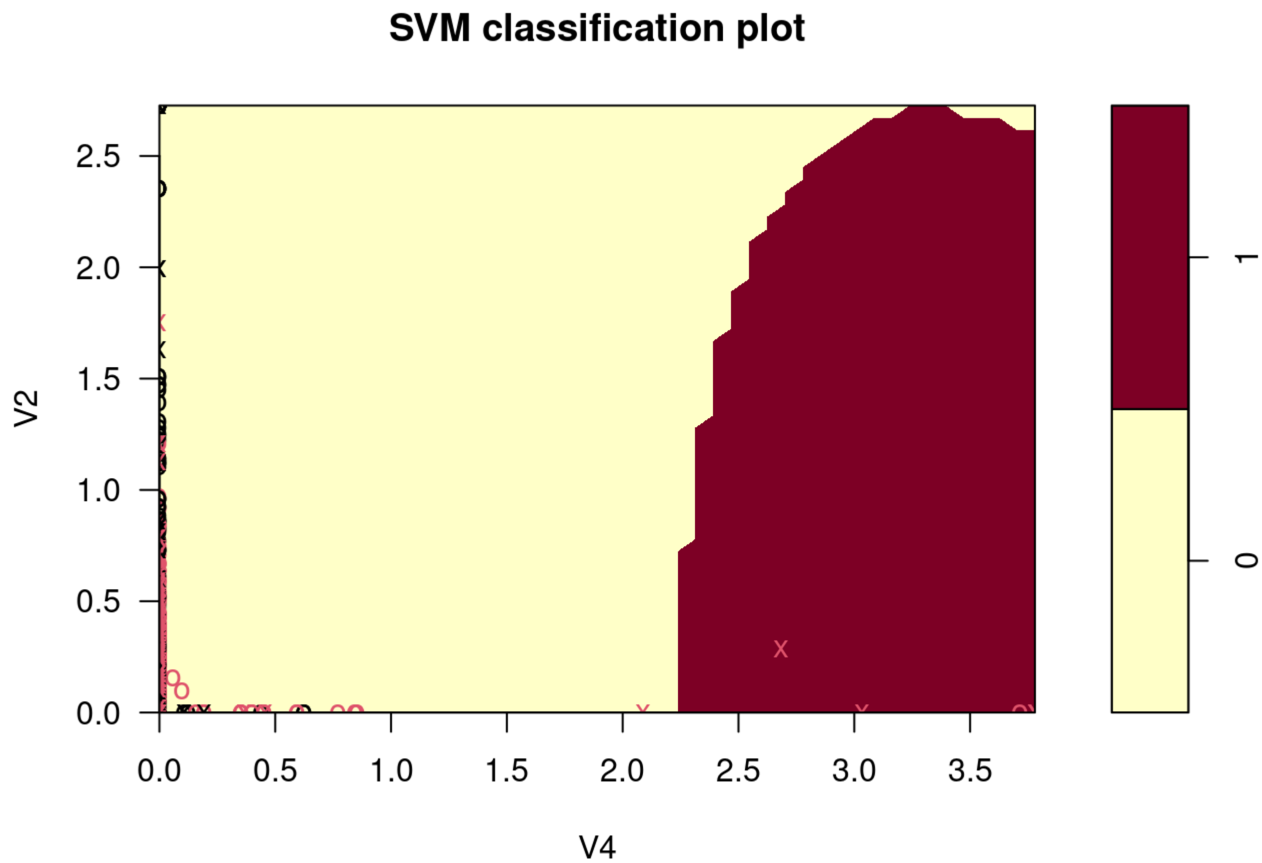
```
tune.gaussian = tune(svm, V58 ~ ., data = new_log_train, kernel = "radial", ranges = list(cost=c
(0.001, 0.01, 0.1, 1, 5, 10),
                                                                                           gamma = c(
0.001, 0.01, 0.1, 1)), validation.x = new_log_test)
```

```
summary(tune.gaussian)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost gamma
##    10  0.01
##
## - best performance: 0.04631262
##
## - Detailed performance results:
##      cost gamma      error dispersion
## 1  1e-03 0.001 0.39716208 0.03006158
## 2  1e-02 0.001 0.39716208 0.03006158
## 3  1e-01 0.001 0.10599412 0.02086794
## 4  1e+00 0.001 0.06163910 0.01477330
## 5  5e+00 0.001 0.06130804 0.01484794
## 6  1e+01 0.001 0.05902791 0.01530110
## 7  1e-03 0.010 0.39716208 0.03006158
## 8  1e-02 0.010 0.17871027 0.02581543
## 9  1e-01 0.010 0.06359882 0.01707959
## 10 1e+00 0.010 0.05772391 0.01431630
## 11 5e+00 0.010 0.04892274 0.01389900
## 12 1e+01 0.010 0.04631262 0.01271322
## 13 1e-03 0.100 0.39716208 0.03006158
## 14 1e-02 0.100 0.39716208 0.03006158
## 15 1e-01 0.100 0.25468268 0.03046838
## 16 1e+00 0.100 0.06685934 0.01822492
## 17 5e+00 0.100 0.06065551 0.01311680
## 18 1e+01 0.100 0.06228630 0.01434946
## 19 1e-03 1.000 0.39716208 0.03006158
## 20 1e-02 1.000 0.39716208 0.03006158
## 21 1e-01 1.000 0.38314279 0.03008008
## 22 1e+00 1.000 0.14184816 0.01954362
## 23 5e+00 1.000 0.13956697 0.02029855
## 24 1e+01 1.000 0.13956697 0.02029855
```

```
log_train_svm <- svm(V58 ~ ., kernel = "radial", data=new_log_train, cost=10, gamma = 0.01)
```

```
plot(log_train_svm, new_log_train, formula = V2 ~ V4)
```




```
svm_predict_train_log <- predict(log_train_svm,new_log_train)
```

```
cm_gaussian_log_train <- confusionMatrix(as.factor(svm_predict_train_log), as.factor(new_log_tra
in$V58), positive = "1")
cm_gaussian_log_train
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1826   45
##          1   23 1173
##
##                Accuracy : 0.9778
##                  95% CI : (0.972, 0.9827)
##     No Information Rate : 0.6029
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.9536
##
##  Mcnemar's Test P-Value : 0.01088
##
##             Sensitivity : 0.9631
##             Specificity : 0.9876
##          Pos Pred Value : 0.9808
##          Neg Pred Value : 0.9759
##              Prevalence : 0.3971
##          Detection Rate : 0.3825
##    Detection Prevalence : 0.3900
##       Balanced Accuracy : 0.9753
##
##        'Positive' Class : 1
##
```

The accuracy is 97.78% for the gaussian svm classifier of the log transformed train data.

```
svm_predict_test_log <- predict(log_train_svm,new_log_test)
```

```
cm_gaussian_log_test <- confusionMatrix(as.factor(svm_predict_test_log), as.factor(new_log_test
$V58), positive = "1")
cm_gaussian_log_test
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 892  34
##          1  24 584
##
##                Accuracy : 0.9622
##                  95% CI : (0.9514, 0.9712)
##     No Information Rate : 0.5971
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.9212
##
##  Mcnemar's Test P-Value : 0.2373
##
##             Sensitivity : 0.9450
##             Specificity : 0.9738
##          Pos Pred Value : 0.9605
##          Neg Pred Value : 0.9633
##              Prevalence : 0.4029
##          Detection Rate : 0.3807
##    Detection Prevalence : 0.3963
##       Balanced Accuracy : 0.9594
##
##        'Positive' Class : 1
##
```

The accuracy is 96.22% for the gaussian svm classifier of the log transform test data.

Gaussian kernel for discretized train and test data

```
tune.gaussian = tune(svm, V58 ~ ., data = new_I_train, kernel = "radial", ranges = list(cost=c(
0.001, 0.01, 0.1, 1, 5, 10),
                                                                                     gamma = c(
0.001, 0.01, 0.1, 1)), validation.x = new_I_test)
```

```
## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
```

```
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.
```

```
## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.
```

```
## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
```

```
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.
```

```
## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.
```

```
## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
```

```
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.
```

```
## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.
```

```
## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
```

```
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.
```

```
## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.
```

```
## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
```

```
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.
```

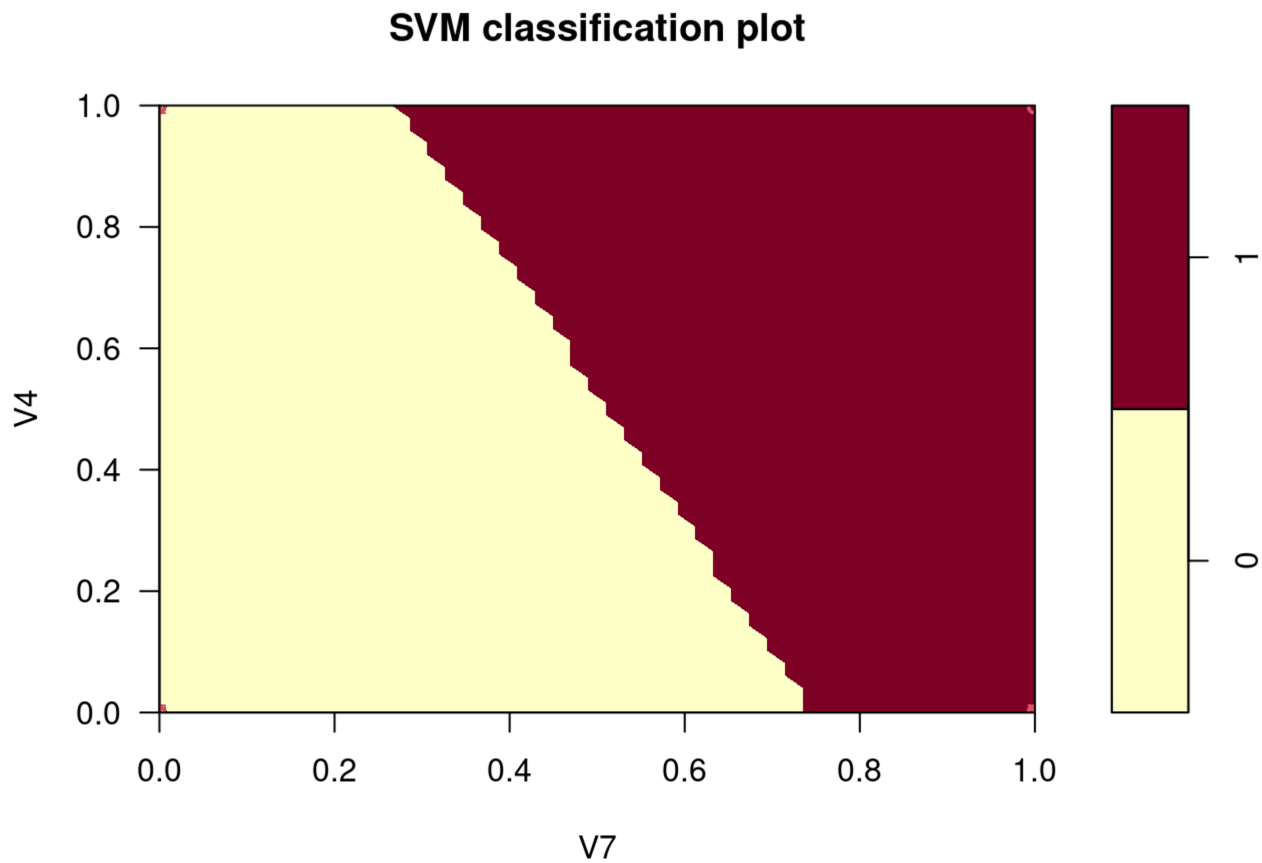
```
summary(tune.gaussian)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost gamma
##    10   0.1
##
## - best performance: 0.04760597
##
## - Detailed performance results:
##     cost gamma      error  dispersion
## 1  1e-03 0.001 0.39715463 0.032287980
## 2  1e-02 0.001 0.39715463 0.032287980
## 3  1e-01 0.001 0.34173426 0.037491000
## 4  1e+00 0.001 0.10824977 0.021114051
## 5  5e+00 0.001 0.07532094 0.014461172
## 6  1e+01 0.001 0.07075749 0.016062712
## 7  1e-03 0.010 0.39715463 0.032287980
## 8  1e-02 0.010 0.37955015 0.036116047
## 9  1e-01 0.010 0.10759831 0.021679645
## 10 1e+00 0.010 0.07010496 0.016364826
## 11 5e+00 0.010 0.06521151 0.012581776
## 12 1e+01 0.010 0.06260778 0.013047555
## 13 1e-03 0.100 0.39715463 0.032287980
## 14 1e-02 0.100 0.13107343 0.017096396
## 15 1e-01 0.100 0.07042537 0.017715638
## 16 1e+00 0.100 0.05966554 0.012569230
## 17 5e+00 0.100 0.04923570 0.010363912
## 18 1e+01 0.100 0.04760597 0.008994929
## 19 1e-03 1.000 0.39715463 0.032287980
## 20 1e-02 1.000 0.39715463 0.032287980
## 21 1e-01 1.000 0.36585659 0.030618842
## 22 1e+00 1.000 0.10727789 0.012018606
## 23 5e+00 1.000 0.10173511 0.012251083
## 24 1e+01 1.000 0.10173511 0.012251083
```

```
I_train_svm <- svm(V58 ~ ., kernel = "radial", data=new_I_train, cost=10, gamma = 0.1)
```

```
## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.
```

```
plot(I_train_svm, new_I_train, formula = V4 ~ V7)
```




```
svm_predict_train_I <- predict(I_train_svm,new_I_train)
```

```
cm_gaussian_I_train <- confusionMatrix(as.factor(svm_predict_train_I), as.factor(new_I_train$V5
8), positive = "1")
cm_gaussian_I_train
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1835   47
##          1   14 1171
##
##                Accuracy : 0.9801
##                  95% CI : (0.9745, 0.9848)
##     No Information Rate : 0.6029
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9583
##
##  Mcnemar's Test P-Value : 4.182e-05
##
##             Sensitivity : 0.9614
##             Specificity : 0.9924
##          Pos Pred Value : 0.9882
##          Neg Pred Value : 0.9750
##              Prevalence : 0.3971
##          Detection Rate : 0.3818
##    Detection Prevalence : 0.3864
##       Balanced Accuracy : 0.9769
##
##        'Positive' Class : 1
##
```

The accuracy is 98.01% for the gaussian svm classifier of the discretized train data.

```
svm_predict_test_I <- predict(I_train_svm,new_I_test)
```

```
cm_gaussian_I_test <- confusionMatrix(as.factor(svm_predict_test_I), as.factor(new_I_test$V58),
positive = "1")
cm_gaussian_I_test
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 884  43
##          1  32 575
##
##                Accuracy : 0.9511
##                  95% CI : (0.9391, 0.9614)
##     No Information Rate : 0.5971
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.8981
##
##  Mcnemar's Test P-Value : 0.2482
##
##             Sensitivity : 0.9304
##             Specificity : 0.9651
##          Pos Pred Value : 0.9473
##          Neg Pred Value : 0.9536
##              Prevalence : 0.4029
##          Detection Rate : 0.3748
##    Detection Prevalence : 0.3957
##       Balanced Accuracy : 0.9477
##
##        'Positive' Class : 1
##
```

The accuracy is 95.11% for the gaussian svm classifier of the discretized test data.

Polynomial SVM for original data

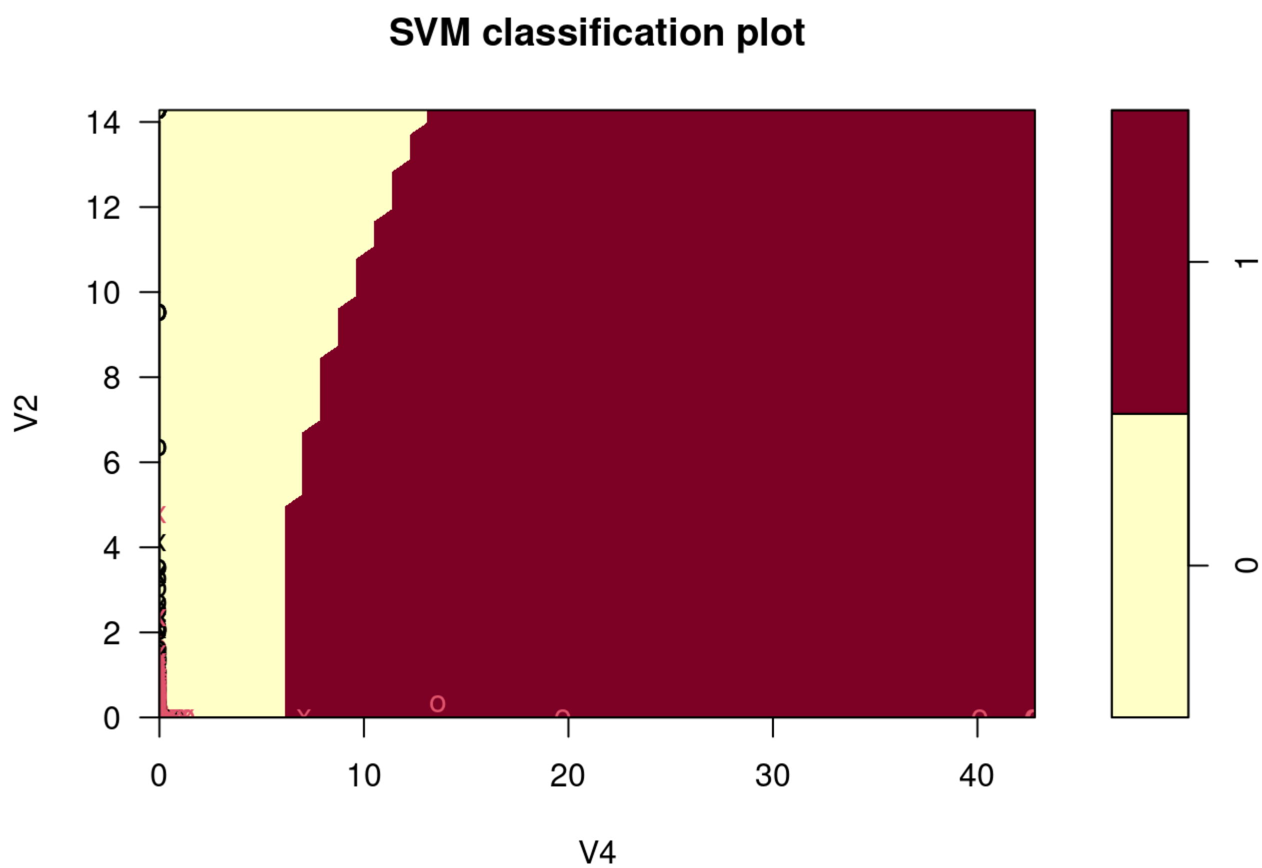
```
tune.polynomial = tune(svm, V58 ~ ., data = new_train, kernel = "polynomial", ranges = list(cost
=c(0.001, 0.01, 0.1, 1, 5, 10),
                                                                                     degree = c
(2, 3, 4, 5)), validation.x = new_test)
```

```
summary(tune.polynomial)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost degree
##    10      2
##
## - best performance: 0.07727747
##
## - Detailed performance results:
##     cost degree      error dispersion
## 1  1e-03      2 0.39648187 0.03019684
## 2  1e-02      2 0.37268527 0.02878043
## 3  1e-01      2 0.28268293 0.01654857
## 4  1e+00      2 0.15944200 0.02207160
## 5  5e+00      2 0.08477997 0.02268680
## 6  1e+01      2 0.07727747 0.02105691
## 7  1e-03      3 0.39224628 0.03142381
## 8  1e-02      3 0.36714462 0.02979141
## 9  1e-01      3 0.30289966 0.02241404
## 10 1e+00      3 0.23280322 0.01616452
## 11 5e+00      3 0.15064082 0.01345442
## 12 1e+01      3 0.11868493 0.01701813
## 13 1e-03      4 0.38963722 0.03035297
## 14 1e-02      4 0.36747035 0.02873118
## 15 1e-01      4 0.31952694 0.01948082
## 16 1e+00      4 0.27062336 0.01811365
## 17 5e+00      4 0.23018778 0.01733523
## 18 1e+01      4 0.19856401 0.01491854
## 19 1e-03      5 0.38605097 0.03119691
## 20 1e-02      5 0.36584275 0.02888525
## 21 1e-01      5 0.32703157 0.01951942
## 22 1e+00      5 0.29116263 0.02337492
## 23 5e+00      5 0.25855954 0.01537263
## 24 1e+01      5 0.24029933 0.01633268
```

```
train_svm <- svm(V58 ~ ., kernel = "polynomial", data=new_train, cost=10, degree = 2)
```

```
plot(train_svm, new_train, formula = V2 ~ V4)
```




```
svm_predict_train <- predict(train_svm,new_train)
```

```
cm_poly_train <- confusionMatrix(as.factor(svm_predict_train), as.factor(new_train$V58), positiv
e = "1")
cm_poly_train
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1815  105
##          1   34 1113
##
##                Accuracy : 0.9547
##                  95% CI : (0.9467, 0.9618)
##     No Information Rate : 0.6029
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9044
##
##  Mcnemar's Test P-Value : 2.897e-09
##
##             Sensitivity : 0.9138
##             Specificity : 0.9816
##          Pos Pred Value : 0.9704
##          Neg Pred Value : 0.9453
##              Prevalence : 0.3971
##          Detection Rate : 0.3629
##    Detection Prevalence : 0.3740
##       Balanced Accuracy : 0.9477
##
##        'Positive' Class : 1
##
```

The accuracy is 95.47% for the polynomial svm classifier of the train data.

```
svm_predict_test <- predict(train_svm,new_test)
```

```
cm_poly_test <- confusionMatrix(as.factor(svm_predict_test), as.factor(new_test$V58), positive =
"1")
cm_poly_test
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 886  89
##          1  30 529
##
##                Accuracy : 0.9224
##                  95% CI : (0.9079, 0.9353)
##     No Information Rate : 0.5971
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8362
##
##  Mcnemar's Test P-Value : 1.056e-07
##
##             Sensitivity : 0.8560
##             Specificity : 0.9672
##          Pos Pred Value : 0.9463
##          Neg Pred Value : 0.9087
##              Prevalence : 0.4029
##          Detection Rate : 0.3449
##    Detection Prevalence : 0.3644
##       Balanced Accuracy : 0.9116
##
##        'Positive' Class : 1
##
```

The accuracy is 92.24% for the polynomial svm classifier of the test data.

Polynomial SVM for standardized data

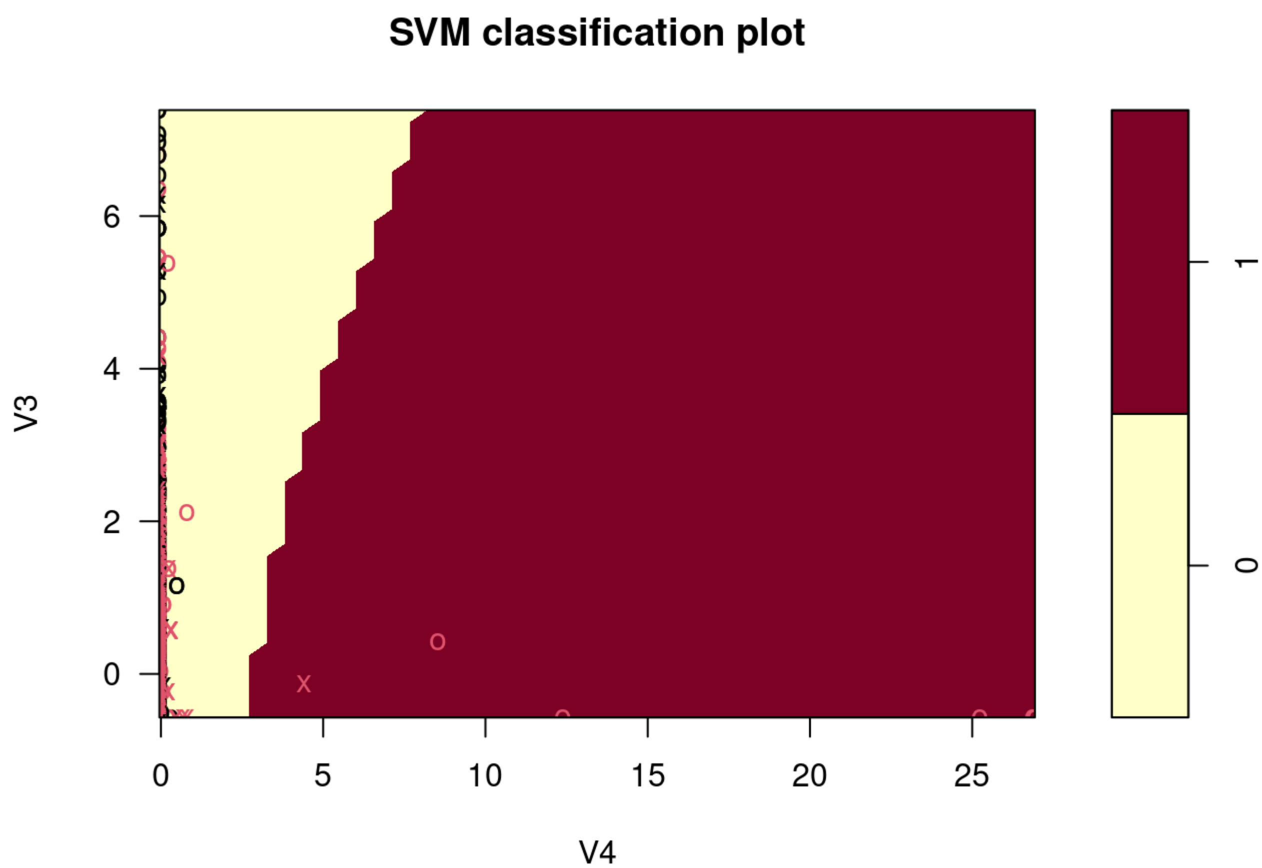
```
tune.polynomial = tune(svm, V58 ~ ., data = new_stan_train, kernel = "polynomial", ranges = list
(cost=c(0.001, 0.01, 0.1, 1, 5, 10),
                                                                                     degree = c
(2, 3, 4, 5)), validation.x = new_stan_test)
```

```
summary(tune.polynomial)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost degree
##    10      2
##
## - best performance: 0.07597347
##
## - Detailed performance results:
##     cost degree      error  dispersion
## 1  1e-03      2 0.39680654 0.019944036
## 2  1e-02      2 0.37430862 0.022744849
## 3  1e-01      2 0.28432011 0.017143964
## 4  1e+00      2 0.15715548 0.028329825
## 5  5e+00      2 0.08379958 0.010700926
## 6  1e+01      2 0.07597347 0.009386065
## 7  1e-03      3 0.39321922 0.021701704
## 8  1e-02      3 0.36745864 0.022015277
## 9  1e-01      3 0.30192140 0.018837322
## 10 1e+00      3 0.23215069 0.024551946
## 11 5e+00      3 0.14803283 0.030048001
## 12 1e+01      3 0.12063720 0.021227965
## 13 1e-03      4 0.38897937 0.020831184
## 14 1e-02      4 0.36713078 0.021171393
## 15 1e-01      4 0.32082881 0.018720903
## 16 1e+00      4 0.26932150 0.023557275
## 17 5e+00      4 0.23020268 0.026245218
## 18 1e+01      4 0.19824147 0.027195990
## 19 1e-03      5 0.38539099 0.020434740
## 20 1e-02      5 0.36582572 0.021573214
## 21 1e-01      5 0.32995785 0.020423686
## 22 1e+00      5 0.28888357 0.021755804
## 23 5e+00      5 0.25725980 0.024680382
## 24 1e+01      5 0.23900492 0.029844496
```

```
stan_train_svm <- svm(V58 ~ ., kernel = "polynomial", data=new_stan_train, cost=10, degree = 2)
```

```
plot(stan_train_svm, new_stan_train, formula = V3 ~ V4)
```




```
svm_predict_train_stan <- predict(stan_train_svm,new_stan_train)
```

```
cm_poly_stan_train <- confusionMatrix(as.factor(svm_predict_train_stan), as.factor(new_stan_trai
n$V58), positive = "1")
cm_poly_stan_train
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1815  105
##          1   34 1113
##
##                Accuracy : 0.9547
##                  95% CI : (0.9467, 0.9618)
##     No Information Rate : 0.6029
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9044
##
##  Mcnemar's Test P-Value : 2.897e-09
##
##             Sensitivity : 0.9138
##             Specificity : 0.9816
##          Pos Pred Value : 0.9704
##          Neg Pred Value : 0.9453
##              Prevalence : 0.3971
##          Detection Rate : 0.3629
##    Detection Prevalence : 0.3740
##       Balanced Accuracy : 0.9477
##
##        'Positive' Class : 1
##
```

The accuracy is 95.47% for the polynomial svm classifier of the standardized train data.

```
svm_predict_test_stan <- predict(stan_train_svm,new_stan_test)
```

```
cm_poly_stan_test <- confusionMatrix(as.factor(svm_predict_test_stan), as.factor(new_stan_test$V
58), positive = "1")
cm_poly_stan_test
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 884  90
##          1  32 528
##
##                Accuracy : 0.9205
##                  95% CI : (0.9058, 0.9335)
##     No Information Rate : 0.5971
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8321
##
##  Mcnemar's Test P-Value : 2.462e-07
##
##             Sensitivity : 0.8544
##             Specificity : 0.9651
##          Pos Pred Value : 0.9429
##          Neg Pred Value : 0.9076
##              Prevalence : 0.4029
##          Detection Rate : 0.3442
##    Detection Prevalence : 0.3651
##       Balanced Accuracy : 0.9097
##
##        'Positive' Class : 1
##
```

The accuracy is 92.05% for the polynomial svm classifier of the standardized test data.

Polynomial SVM for the log transformed data

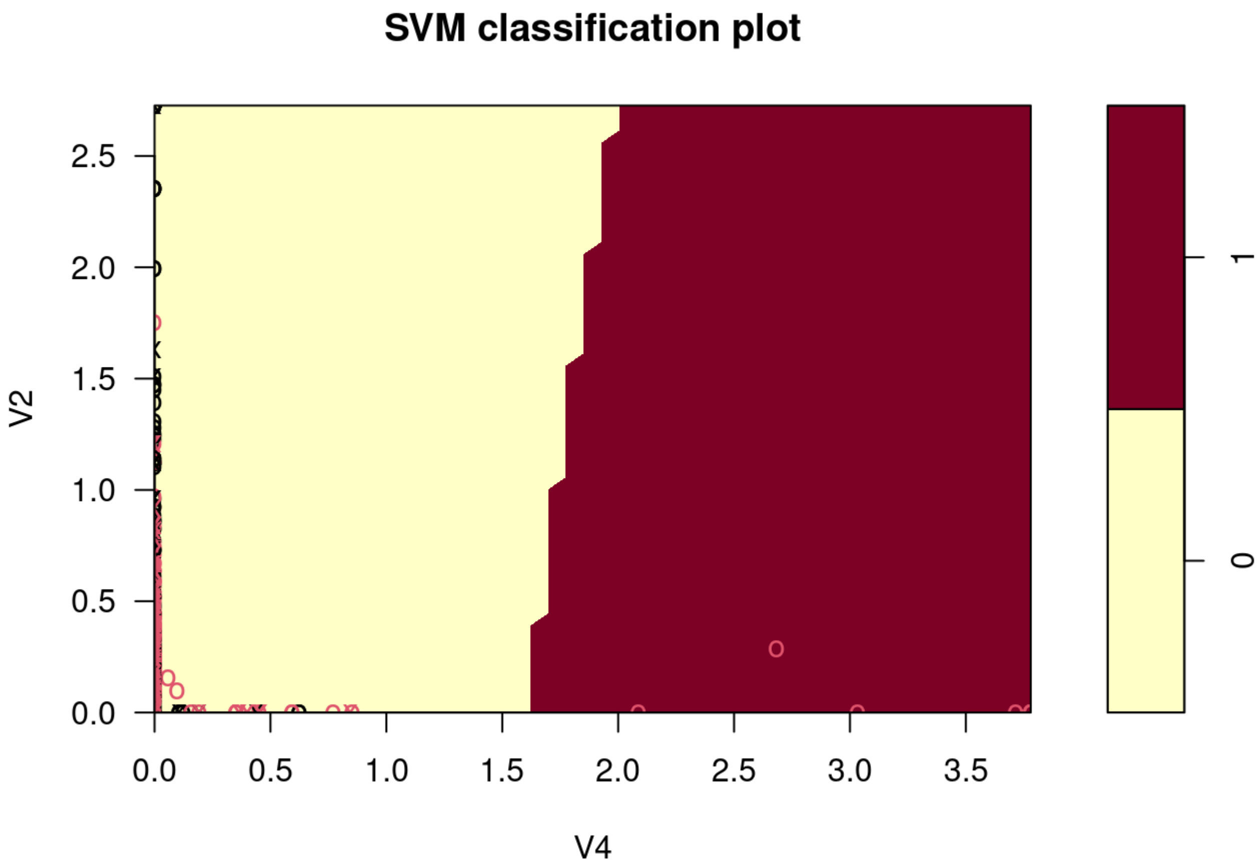
```
tune.polynomial = tune(svm, V58 ~ ., data = new_log_train, kernel = "polynomial", ranges = list
(cost=c(0.001, 0.01, 0.1, 1, 5, 10),
                                                                                    degree = c
(2, 3, 4, 5)), validation.x = new_log_test)
```

```
summary(tune.polynomial)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost degree
##     5      2
##
## - best performance: 0.05673501
##
## - Detailed performance results:
##     cost degree      error dispersion
## 1  1e-03      2 0.39714185 0.03020173
## 2  1e-02      2 0.36127717 0.03201246
## 3  1e-01      2 0.20282515 0.03251248
## 4  1e+00      2 0.07435120 0.01789309
## 5  5e+00      2 0.05673501 0.01203088
## 6  1e+01      2 0.05868728 0.01159128
## 7  1e-03      3 0.39551212 0.03025857
## 8  1e-02      3 0.35214707 0.03316304
## 9  1e-01      3 0.24521726 0.03498523
## 10 1e+00      3 0.10336271 0.02053258
## 11 5e+00      3 0.06880416 0.01758022
## 12 1e+01      3 0.05804858 0.01840725
## 13 1e-03      4 0.39453280 0.03019871
## 14 1e-02      4 0.35964318 0.03232134
## 15 1e-01      4 0.29217496 0.03812399
## 16 1e+00      4 0.20445594 0.02966783
## 17 5e+00      4 0.12260118 0.02809257
## 18 1e+01      4 0.09781993 0.02462372
## 19 1e-03      5 0.39127440 0.02902950
## 20 1e-02      5 0.35573120 0.03104101
## 21 1e-01      5 0.30260693 0.04036276
## 22 1e+00      5 0.23608397 0.03427826
## 23 5e+00      5 0.16695408 0.02852916
## 24 1e+01      5 0.14673628 0.02907865
```

```
log_train_svm <- svm(V58 ~ ., kernel = "polynomial", data=new_log_train, cost=10, degree = 2)
```

```
plot(log_train_svm, new_log_train, formula = V2 ~ V4)
```




```
svm_predict_train_log <- predict(log_train_svm,new_log_train)
```

```
cm_poly_log_train <- confusionMatrix(as.factor(svm_predict_train_log), as.factor(new_log_train$V
58), positive = "1")
cm_poly_log_train
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1827   48
##          1   22 1170
##
##                Accuracy : 0.9772
##                  95% CI : (0.9713, 0.9822)
##     No Information Rate : 0.6029
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9522
##
##  Mcnemar's Test P-Value : 0.002807
##
##             Sensitivity : 0.9606
##             Specificity : 0.9881
##          Pos Pred Value : 0.9815
##          Neg Pred Value : 0.9744
##              Prevalence : 0.3971
##          Detection Rate : 0.3815
##    Detection Prevalence : 0.3887
##       Balanced Accuracy : 0.9743
##
##        'Positive' Class : 1
##
```

The accuracy is 97.72% for the polynomial svm classifier of the log transformed train data.

```
svm_predict_test_log <- predict(log_train_svm,new_log_test)
```

```
cm_poly_log_test <- confusionMatrix(as.factor(svm_predict_test_log), as.factor(new_log_test$V5
8), positive = "1")
cm_poly_log_test
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction   0   1
##          0 892  52
##          1  24 566
##                                           
##                Accuracy : 0.9505          
##                  95% CI : (0.9384, 0.9608)
##     No Information Rate : 0.5971          
##     P-Value [Acc > NIR] : < 2.2e-16       
##                                           
##                   Kappa : 0.8963          
##                                           
##  Mcnemar's Test P-Value : 0.001954        
##                                           
##             Sensitivity : 0.9159          
##             Specificity : 0.9738          
##          Pos Pred Value : 0.9593          
##          Neg Pred Value : 0.9449          
##              Prevalence : 0.4029          
##          Detection Rate : 0.3690          
##    Detection Prevalence : 0.3846          
##       Balanced Accuracy : 0.9448          
##                                           
##        'Positive' Class : 1               
## 
```

The accuracy is 95.05% for the polynomial svm classifier of the log transformed test data.

Polynomial SVM of the discretized data

```
tune.polynomial = tune(svm, V58 ~ ., data = new_I_train, kernel = "polynomial", ranges = list(co
st=c(0.001, 0.01, 0.1, 1, 5, 10),
                                                                                      degree = c
(2, 3, 4, 5)), validation.x = new_I_test)
```

```
## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
```

```
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.
```

```
## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.
```

```
## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
```

```
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.
```

```
## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.
```

```
## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
```

```
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.
```

```
## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.
```

```
## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
```

```
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.
```

```
## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.
```

```
## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
```

```
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.

## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.
```

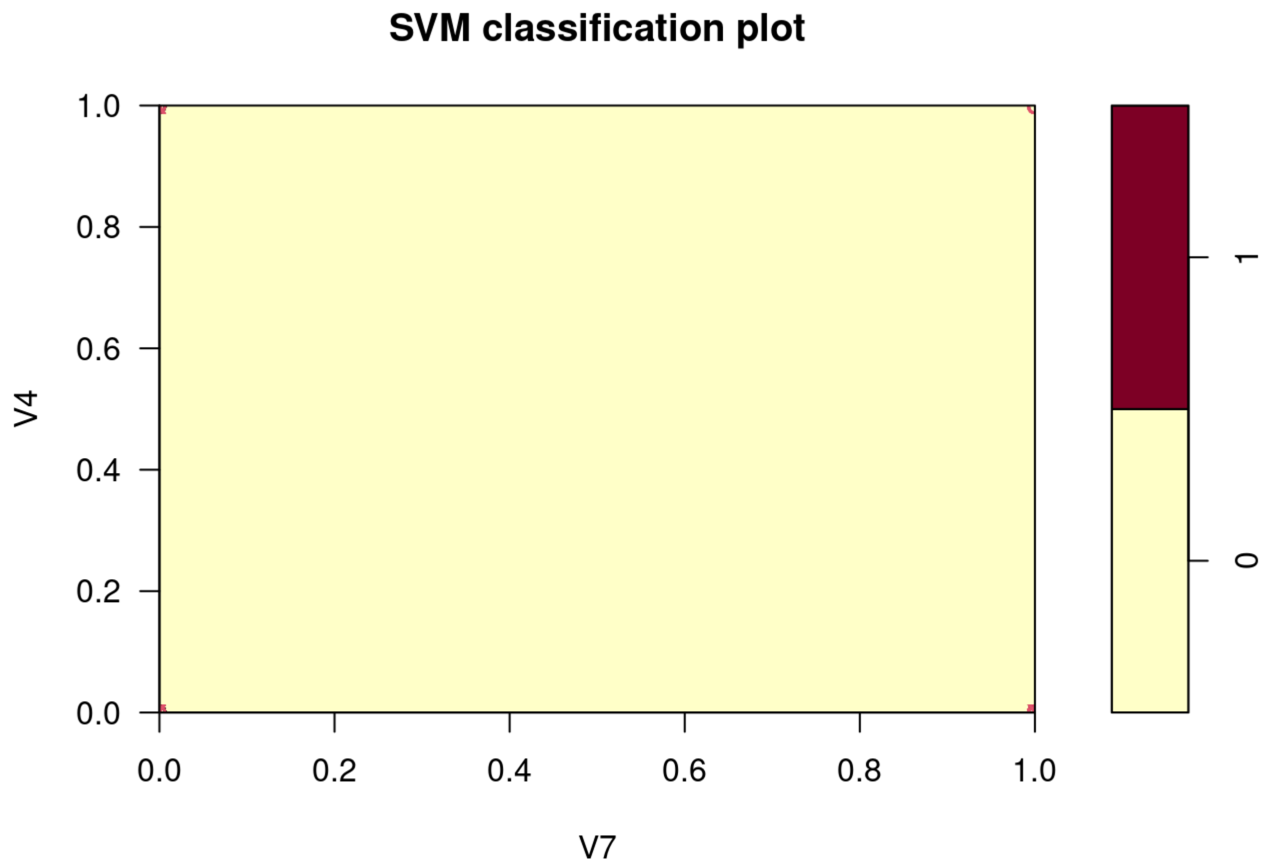
```
summary(tune.polynomial)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost degree
##    10      2
##
## - best performance: 0.06618871
##
## - Detailed performance results:
##     cost degree      error  dispersion
## 1  1e-03      2 0.39712908 0.022693219
## 2  1e-02      2 0.39712908 0.022693219
## 3  1e-01      2 0.19824360 0.024454789
## 4  1e+00      2 0.10988269 0.016296517
## 5  5e+00      2 0.07825147 0.011902827
## 6  1e+01      2 0.06618871 0.009971212
## 7  1e-03      3 0.39712908 0.022693219
## 8  1e-02      3 0.39712908 0.022693219
## 9  1e-01      3 0.32735943 0.016134043
## 10 1e+00      3 0.17313768 0.021098926
## 11 5e+00      3 0.12715825 0.021232422
## 12 1e+01      3 0.10694045 0.023634010
## 13 1e-03      4 0.39712908 0.022693219
## 14 1e-02      4 0.39712908 0.022693219
## 15 1e-01      4 0.39256456 0.023585826
## 16 1e+00      4 0.26541483 0.020010875
## 17 5e+00      4 0.18682911 0.019285814
## 18 1e+01      4 0.16172638 0.025512518
## 19 1e-03      5 0.39712908 0.022693219
## 20 1e-02      5 0.39712908 0.022693219
## 21 1e-01      5 0.39712908 0.022693219
## 22 1e+00      5 0.34431458 0.017747662
## 23 5e+00      5 0.26345830 0.021132680
## 24 1e+01      5 0.23084989 0.027118512
```

```
I_train_svm <- svm(V58 ~ ., kernel = "polynomial", data=new_I_train, cost=10, degree = 2)
```

```
## Warning in svm.default(x, y, scale = scale, ..., na.action = na.action):
## Variable(s) 'V55' and 'V56' and 'V57' constant. Cannot scale data.
```

```
plot(I_train_svm, new_I_train, formula = V4 ~ V7)
```




```
svm_predict_train_I <- predict(I_train_svm,new_I_train)
```

```
cm_poly_I_train <- confusionMatrix(as.factor(svm_predict_train_I), as.factor(new_I_train$V58), p
ositive = "1")
cm_poly_I_train
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1794  133
##          1   55 1085
##
##                Accuracy : 0.9387
##                  95% CI : (0.9296, 0.9469)
##     No Information Rate : 0.6029
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8706
##
##  Mcnemar's Test P-Value : 1.957e-08
##
##             Sensitivity : 0.8908
##             Specificity : 0.9703
##          Pos Pred Value : 0.9518
##          Neg Pred Value : 0.9310
##              Prevalence : 0.3971
##          Detection Rate : 0.3538
##    Detection Prevalence : 0.3717
##       Balanced Accuracy : 0.9305
##
##        'Positive' Class : 1
##
```

The accuracy is 93.87% for the polynomial svm classifier of the discretized train data.

```
svm_predict_test_I <- predict(I_train_svm,new_I_test)
```

```
cm_poly_I_test <- confusionMatrix(as.factor(svm_predict_test_I), as.factor(new_I_test$V58), posi
tive = "1")
cm_poly_I_test
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 878  81
##          1  38 537
##
##                Accuracy : 0.9224
##                  95% CI : (0.9079, 0.9353)
##     No Information Rate : 0.5971
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8369
##
##  Mcnemar's Test P-Value : 0.0001181
##
##             Sensitivity : 0.8689
##             Specificity : 0.9585
##          Pos Pred Value : 0.9339
##          Neg Pred Value : 0.9155
##              Prevalence : 0.4029
##          Detection Rate : 0.3501
##    Detection Prevalence : 0.3748
##       Balanced Accuracy : 0.9137
##
##        'Positive' Class : 1
##
```

The accuracy is 92.24% for the polynomial svm classifier of the discretized test data.

7. A report of classification errors using different methods and different preprocessed data.

```
cm_linear_train$overall['Accuracy']
```

```
##  Accuracy
## 0.9373981
```

```
cm_linear_test$overall['Accuracy']
```

```
##  Accuracy
## 0.9341591
```

```
cm_linear_stan_train$overall['Accuracy']
```

```
##  Accuracy
## 0.9272905
```

```
cm_linear_stan_test$overall['Accuracy']
```

```
##  Accuracy
## 0.9322034
```

```
cm_linear_log_train$overall['Accuracy']
```

```
##  Accuracy
## 0.9409847
```

```
cm_linear_log_test$overall['Accuracy']
```

```
##  Accuracy
## 0.9419817
```

```
cm_linear_I_train$overall['Accuracy']
```

```
##  Accuracy
## 0.9413107
```

```
cm_linear_I_test$overall['Accuracy']
```

```
##  Accuracy
## 0.9243807
```

```
cm_gaussian_train$overall['Accuracy']
```

```
##  Accuracy
## 0.9628301
```

```
cm_gaussian_test$overall['Accuracy']
```

```
##  Accuracy
## 0.9452412
```

```
cm_gaussian_stan_train$overall['Accuracy']
```

```
##  Accuracy
## 0.9628301
```

```
cm_gaussian_stan_test$overall['Accuracy']
```

```
##  Accuracy
## 0.9413299
```

```
cm_gaussian_log_train$overall['Accuracy']
```

```
##  Accuracy
## 0.9778285
```

```
cm_gaussian_log_test$overall['Accuracy']
```

```
##  Accuracy
## 0.9621904
```

```
cm_gaussian_I_train$overall['Accuracy']
```

```
##  Accuracy
## 0.9801109
```

```
cm_gaussian_I_test$overall['Accuracy']
```

```
##  Accuracy
## 0.9511082
```

```
cm_poly_train$overall['Accuracy']
```

```
##  Accuracy
## 0.9546788
```

```
cm_poly_test$overall['Accuracy']
```

```
## Accuracy
## 0.922425
```

```
cm_poly_stan_train$overall['Accuracy']
```

```
##  Accuracy
## 0.9546788
```

```
cm_poly_stan_test$overall['Accuracy']
```

```
##  Accuracy
## 0.9204694
```

```
cm_poly_log_train$overall['Accuracy']
```

```
##  Accuracy
## 0.9771764
```

```
cm_poly_log_test$overall['Accuracy']
```

```
##  Accuracy
## 0.9504563
```

```
cm_poly_I_train$overall['Accuracy']
```

```
##  Accuracy
## 0.9387023
```

```
cm_poly_I_test$overall['Accuracy']
```

```
## Accuracy
## 0.922425
```

```
accuracy.svm.table <- matrix( c(cm_linear_train$overall['Accuracy'], cm_linear_stan_train$overal
l['Accuracy'], cm_linear_log_train$overall['Accuracy'], cm_linear_I_train$overall['Accuracy'], c
m_linear_test$overall['Accuracy'], cm_linear_stan_test$overall['Accuracy'], cm_linear_log_test$o
verall['Accuracy'], cm_linear_I_test$overall['Accuracy'],  cm_gaussian_train$overall['Accuracy'
], cm_gaussian_stan_train$overall['Accuracy'], cm_gaussian_log_train$overall['Accuracy'], cm_gau
ssian_I_train$overall['Accuracy'], cm_gaussian_test$overall['Accuracy'], cm_gaussian_stan_test$o
verall['Accuracy'], cm_gaussian_log_test$overall['Accuracy'], cm_gaussian_I_test$overall['Accura
cy'], cm_poly_train$overall['Accuracy'], cm_poly_stan_train$overall['Accuracy'], cm_poly_log_tra
in$overall['Accuracy'], cm_poly_I_train$overall['Accuracy'], cm_poly_test$overall['Accuracy'], c
m_poly_stan_test$overall['Accuracy'], cm_poly_log_test$overall['Accuracy'], cm_poly_I_test$overa
ll['Accuracy']), ncol=3)

accuracy.svm.table
```

```
##           [,1]      [,2]      [,3]
## [1,] 0.9373981 0.9628301 0.9546788
## [2,] 0.9272905 0.9628301 0.9546788
## [3,] 0.9409847 0.9778285 0.9771764
## [4,] 0.9413107 0.9801109 0.9387023
## [5,] 0.9341591 0.9452412 0.9224250
## [6,] 0.9322034 0.9413299 0.9204694
## [7,] 0.9419817 0.9621904 0.9504563
## [8,] 0.9243807 0.9511082 0.9224250
```

For the SVM table,

```
colnames(accuracy.svm.table) <- c("linear", "gaussian", "polynomial")
rownames(accuracy.svm.table) <- c("train", "standardized train", "log train", "I train", "test",
"standardized test", "log test", "I test")

accuracy.svm.table
```

```
##                       linear  gaussian polynomial
## train              0.9373981 0.9628301  0.9546788
## standardized train 0.9272905 0.9628301  0.9546788
## log train          0.9409847 0.9778285  0.9771764
## I train            0.9413107 0.9801109  0.9387023
## test               0.9341591 0.9452412  0.9224250
## standardized test  0.9322034 0.9413299  0.9204694
## log test           0.9419817 0.9621904  0.9504563
## I test             0.9243807 0.9511082  0.9224250
```

For LDA and QDA table,

```
colnames(accuracy.da.table) <- c("lda stan", "lda log", "qda stan", "qda log")
rownames(accuracy.da.table) <- c("train", "test")

accuracy.da.table
```

```
##        lda stan   lda log  qda stan   qda log
## train 0.8982719 0.9396805 0.8213238 0.8412129
## test  0.8970013 0.9348110 0.8252934 0.8428944
```

For Logistic Regression table,

```
colnames(accuracy.lr.table) <- c("lr original", "lr standardized", "lr log", "lr I")
rownames(accuracy.lr.table) <- c("train", "test")

accuracy.lr.table
```

```
##       lr original lr standardized    lr log      lr I
## train   0.9282687       0.5415716 0.5347245 0.9018585
## test    0.9269883       0.5521512 0.5410691 0.8970013
```

Note that in all the tables above (SVM, LDA and QDA, and Logistic Regression), the I stands for discretized.

8. Used single method with properly chosen tuning parameter and a combination of several methods to design a classifier with test error rate as small as possible.

The log transformation Gaussian SVM classifier has the best test accuracy rate of 96.22% based on the table.

Combine PCA with the Gaussian SVM classisfier for the log transformed to get the smallest test error rate.

Continue single method tuning with more precise parameters.

Fine tune it even more to achieve a smaller test error.

Original Tune: Cost = 10, gamma = 0.01, accuracy = 96.22%

1st Adjustment: Cost = 9.5, gamma = 0.015, accuracy = 96.41%

2nd Adjustment: Cost = 9.6, gamma = 0.0175, accuracy = 96.61%

3rd Adjustment: Cost = 9.575, gamma = 0.017, accuracy = 96.61%

4th Adjustment: Cost = 9.565, gamma = 0.017, accuracy = 96.61%

5th Adjustment: Cost = 9.56, gamma = 0.0166, accuracy = 96.48%

6th Adjustment: Cost = 9.565, gamma = 0.018, accuracy = 96.22%

7th Adjustment: Cost = 9.565, gamma = 0.0175, accuracy = 96.22%

8th Adjustment: Cost = 9.56, gamma = 0.0172, accuracy = 96.21%

9th Adjustment: Cost = 9.57, gamma = 0.0174, accuracy = 96.22%

10th Adjustment: Cost = 9.58, gamma = 0.0177, accuracy = 96.24%

11th Adjustment: Cost = 9.585, gamma = 0.0179, accuracy = 96.21%

12th Adjustment: Cost = 9.59, gamma = 0.0175, accuracy = 96.21%

13th Adjustment: Cost = 9.59, gamma = 0.0171, accuracy = 96.21%

14th Adjustment: Cost = 9.51, gamma = 0.0171, accuracy = 96.22%

15th Adjustment: Cost = 9.52, gamma = 0.0172, accuracy = 96.22%

16th Adjustment: Cost = 9.4 , gamma = 0.0173, accuracy = 96.20%

17th Adjustment: Cost = 9.4, gamma = 0.0162, accuracy = 96.20%

18th Adjustment: Cost = 9.3, gamma = 0.0164, accuracy = 96.22%

19th Adjustment: Cost = 9.3, gamma = 0.0152, accuracy = 96.22%

20th Adjustment: Cost = 9.3, gamma = 0.0162, accuracy = 96.22%

Tuned 20 times to make the test error rate smaller. In conclusion, the optimal parameters for the gaussian classifier on the log transformation data is roughly: cost = 9.565, or cost = 9.575, or cost = 9.56, gamma = 0.017, or gamma = 0.0175 for an accuracy of 96.61%, an improvement of 0.4% compare to the original tuning parameter.

```
tune.gaussian = tune(svm, V58 ~ ., data = new_log_train, kernel = "radial", ranges = list(cost=c
(9.3, 9.31, 9.32, 9.33, 9.34), gamma = c(0.0162, 0.0163, 0.0164, 0.0165, 0.0166)), validation.x
 = new_log_test )
```

```
summary(tune.gaussian)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost  gamma
##  9.34 0.0162
##
## - best performance: 0.04205787
##
## - Detailed performance results:
##    cost  gamma      error dispersion
## 1  9.30 0.0162 0.04238466 0.01320277
## 2  9.31 0.0162 0.04238466 0.01320277
## 3  9.32 0.0162 0.04238466 0.01320277
## 4  9.33 0.0162 0.04238466 0.01320277
## 5  9.34 0.0162 0.04205787 0.01324044
## 6  9.30 0.0163 0.04205787 0.01324044
## 7  9.31 0.0163 0.04205787 0.01324044
## 8  9.32 0.0163 0.04205787 0.01324044
## 9  9.33 0.0163 0.04205787 0.01324044
## 10 9.34 0.0163 0.04205787 0.01324044
## 11 9.30 0.0164 0.04205787 0.01324044
## 12 9.31 0.0164 0.04205787 0.01324044
## 13 9.32 0.0164 0.04205787 0.01324044
## 14 9.33 0.0164 0.04205787 0.01324044
## 15 9.34 0.0164 0.04205787 0.01324044
## 16 9.30 0.0165 0.04238360 0.01372470
## 17 9.31 0.0165 0.04271040 0.01367972
## 18 9.32 0.0165 0.04271040 0.01367972
## 19 9.33 0.0165 0.04271040 0.01367972
## 20 9.34 0.0165 0.04271040 0.01367972
## 21 9.30 0.0166 0.04271040 0.01367972
## 22 9.31 0.0166 0.04271040 0.01367972
## 23 9.32 0.0166 0.04271040 0.01367972
## 24 9.33 0.0166 0.04271040 0.01367972
## 25 9.34 0.0166 0.04271040 0.01367972
```

```
svm_predict_test_log <- predict(log_train_svm,new_log_test)
```

```r
cm_gaussian_log_test <- confusionMatrix(as.factor(svm_predict_test_log), as.factor(new_log_test
$V58), positive = "1")
cm_gaussian_log_test
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 892  52
##          1  24 566
##
##                Accuracy : 0.9505
##                  95% CI : (0.9384, 0.9608)
##     No Information Rate : 0.5971
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8963
##
##  Mcnemar's Test P-Value : 0.001954
##
##             Sensitivity : 0.9159
##             Specificity : 0.9738
##          Pos Pred Value : 0.9593
##          Neg Pred Value : 0.9449
##              Prevalence : 0.4029
##          Detection Rate : 0.3690
##    Detection Prevalence : 0.3846
##       Balanced Accuracy : 0.9448
##
##        'Positive' Class : 1
##
```