Project@DataMining

Kowshik Kumar B

May 10, 2019

Setting the seed to '12345' for rerunning and loading all required libraries.

```
set.seed(12345)

library(leaps)

## Warning: package 'leaps' was built under R version 3.6.1

library(MASS)
library(ROCR)

## Warning: package 'ROCR' was built under R version 3.6.1

## Loading required package: gplots

## Warning: package 'gplots' was built under R version 3.6.1

##
## Attaching package: 'gplots'

## The following object is masked from 'package:stats':
##
##     lowess

library(ggplot2)

## Warning: package 'ggplot2' was built under R version 3.6.1

library(e1071)

## Warning: package 'e1071' was built under R version 3.6.1

df <- read.csv("Fundraising.csv",header = T)
test.df<-read.csv("FutureFundraising.csv",header = T)
dim(test.df)

## [1] 2000   24

head(df)

##   Row.Id Row.Id. zipconvert_2 zipconvert_3 zipconvert_4 zipconvert_5
## 1      1      17            0            1            0            0
## 2      2      25            1            0            0            0
## 3      3      29            0            0            0            1
## 4      4      38            0            0            0            1
```

```
## 5      5      40            0            1            0            0
## 6      6      53            0            1            0            0
##   homeowner.dummy NUMCHLD INCOME gender.dummy WEALTH   HV Icmed Icavg IC15
## 1               1       1      5            1      9 1399   637   703    1
## 2               1       1      1            0      7  698   422   463    4
## 3               0       2      5            1      8  828   358   376   13
## 4               1       1      3            0      4 1471   484   546    4
## 5               1       1      4            0      8  547   386   432    7
## 6               1       1      4            1      8  482   242   275   28
##   NUMPROM RAMNTALL MAXRAMNT LASTGIFT totalmonths TIMELAG  AVGGIFT TARGET_B
## 1      74      102        6        5          29       3 4.857143        1
## 2      46       94       12       12          34       6 9.400000        1
## 3      32       30       10        5          29       7 4.285714        1
## 4      94      177       10        8          30       3 7.080000        0
## 5      20       23       11       11          30       6 7.666667        0
## 6      38       73       10       10          31       3 7.300000        1
##   TARGET_D
## 1        5
## 2       10
## 3        5
## 4        0
## 5        0
## 6        8

## STEP #0: Deleting unnecessary columns from our data.

df$Row.Id  <- NULL
df$Row.Id. <- NULL
df$TARGET_D <- NULL
```

The “Fundraising.csv” dataset was partitioned into 60% training and 40% validation sets. We deleted the Row Id, Row Id., and TARGET_D columns from the dataset. In this particular case, responders to the fundraising drive is the rarer class and more important than the non-responders. Such an asymmetric response and cost requires oversampling the donor class to obtain more useful data for improving the performance of the classifiers. Since the response rate is 5.1%, a simple random sampling from the original dataset would have yielded too few relevant classes to build a strong predictive model. Therefore, a stratified sampling with a disproportionate weighting of the donor class is used for the training data. We have used logistic regression to build our first model.

```
## STEP #1: Partitioning our dataset. (60% Training, 40% Validation).

train.rows <- sample(rownames(df), dim(df)[1]*0.6)
train.df <- df[train.rows, ]

valid.rows <- setdiff(rownames(df), train.rows)
valid.df <- df[valid.rows, ]
```

```r
# Full Model.
full.model <- glm(TARGET_B ~ ., data = train.df, family = "binomial")
summary(full.model)

##
## Call:
## glm(formula = TARGET_B ~ ., family = "binomial", data = train.df)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.7789  -1.1692   0.7801   1.1426   2.1951
##
## Coefficients:
##                   Estimate Std. Error z value Pr(>|z|)
## (Intercept)     -1.258e+01  3.088e+02  -0.041   0.9675
## zipconvert_2     1.356e+01  3.088e+02   0.044   0.9650
## zipconvert_3     1.357e+01  3.088e+02   0.044   0.9650
## zipconvert_4     1.361e+01  3.088e+02   0.044   0.9648
## zipconvert_5     1.363e+01  3.088e+02   0.044   0.9648
## homeowner.dummy  7.983e-03  1.192e-01   0.067   0.9466
## NUMCHLD         -2.038e-01  1.420e-01  -1.435   0.1513
## INCOME           7.250e-02  3.316e-02   2.187   0.0288 *
## gender.dummy     1.832e-01  9.700e-02   1.889   0.0589 .
## WEALTH           2.086e-02  2.279e-02   0.915   0.3601
## HV               1.116e-04  9.084e-05   1.228   0.2194
## Icmed            1.146e-03  1.219e-03   0.940   0.3474
## Icavg           -1.350e-03  1.322e-03  -1.021   0.3072
## IC15             4.683e-03  5.746e-03   0.815   0.4151
## NUMPROM          6.043e-03  2.863e-03   2.111   0.0348 *
## RAMNTALL        -2.390e-04  4.039e-04  -0.592   0.5540
## MAXRAMNT         1.587e-03  3.542e-03   0.448   0.6542
## LASTGIFT        -2.294e-02  1.071e-02  -2.142   0.0322 *
## totalmonths     -4.958e-02  1.263e-02  -3.924  8.7e-05 ***
## TIMELAG          1.413e-02  8.722e-03   1.620   0.1052
## AVGGIFT          1.099e-02  1.462e-02   0.752   0.4519
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2594.2  on 1871  degrees of freedom
## Residual deviance: 2528.2  on 1851  degrees of freedom
## AIC: 2570.2
##
## Number of Fisher Scoring iterations: 12

# Model after AIC method.
step.model <- stepAIC(full.model, direction = "both", trace = TRUE)
```

```
## Start:  AIC=2570.18
## TARGET_B ~ zipconvert_2 + zipconvert_3 + zipconvert_4 + zipconvert_5 +
##     homeowner.dummy + NUMCHLD + INCOME + gender.dummy + WEALTH +
##     HV + Icmed + Icavg + IC15 + NUMPROM + RAMNTALL + MAXRAMNT +
##     LASTGIFT + totalmonths + TIMELAG + AVGGIFT
##
##                   Df Deviance    AIC
## - homeowner.dummy  1   2528.2 2568.2
## - MAXRAMNT         1   2528.4 2568.4
## - RAMNTALL         1   2528.6 2568.6
## - AVGGIFT          1   2528.7 2568.7
## - IC15             1   2528.8 2568.8
## - WEALTH           1   2529.0 2569.0
## - Icmed            1   2529.1 2569.1
## - Icavg            1   2529.2 2569.2
## - HV               1   2529.7 2569.7
## <none>                 2528.2 2570.2
## - NUMCHLD          1   2530.3 2570.3
## - TIMELAG          1   2530.8 2570.8
## - gender.dummy     1   2531.8 2571.8
## - zipconvert_2     1   2532.3 2572.3
## - zipconvert_3     1   2532.3 2572.3
## - zipconvert_4     1   2532.4 2572.4
## - zipconvert_5     1   2532.5 2572.5
## - NUMPROM          1   2532.7 2572.7
## - INCOME           1   2533.0 2573.0
## - LASTGIFT         1   2533.1 2573.1
## - totalmonths      1   2543.9 2583.9
##
## Step:  AIC=2568.18
## TARGET_B ~ zipconvert_2 + zipconvert_3 + zipconvert_4 + zipconvert_5 +
##     NUMCHLD + INCOME + gender.dummy + WEALTH + HV + Icmed + Icavg +
##     IC15 + NUMPROM + RAMNTALL + MAXRAMNT + LASTGIFT + totalmonths +
##     TIMELAG + AVGGIFT
##
##                   Df Deviance    AIC
## - MAXRAMNT         1   2528.4 2566.4
## - RAMNTALL         1   2528.6 2566.6
## - AVGGIFT          1   2528.8 2566.8
## - IC15             1   2528.8 2566.8
## - WEALTH           1   2529.0 2567.0
## - Icmed            1   2529.1 2567.1
## - Icavg            1   2529.2 2567.2
## - HV               1   2529.7 2567.7
## <none>                 2528.2 2568.2
## - NUMCHLD          1   2530.3 2568.3
## - TIMELAG          1   2530.9 2568.9
## - gender.dummy     1   2531.8 2569.8
## + homeowner.dummy  1   2528.2 2570.2
## - zipconvert_2     1   2532.3 2570.3
```

```
## - zipconvert_3     1   2532.3 2570.3
## - zipconvert_4     1   2532.4 2570.4
## - zipconvert_5     1   2532.5 2570.5
## - NUMPROM          1   2532.7 2570.7
## - LASTGIFT         1   2533.1 2571.1
## - INCOME           1   2533.6 2571.6
## - totalmonths      1   2543.9 2581.9
##
## Step:  AIC=2566.45
## TARGET_B ~ zipconvert_2 + zipconvert_3 + zipconvert_4 + zipconvert_5 +
##     NUMCHLD + INCOME + gender.dummy + WEALTH + HV + Icmed + Icavg +
##     IC15 + NUMPROM + RAMNTALL + LASTGIFT + totalmonths + TIMELAG +
##     AVGGIFT
##
##                  Df Deviance    AIC
## - RAMNTALL        1   2528.6 2564.6
## - IC15            1   2529.1 2565.1
## - AVGGIFT         1   2529.2 2565.2
## - WEALTH          1   2529.3 2565.3
## - Icmed           1   2529.3 2565.3
## - Icavg           1   2529.5 2565.5
## - HV              1   2530.0 2566.0
## <none>                2528.4 2566.4
## - NUMCHLD         1   2530.6 2566.6
## - TIMELAG         1   2531.1 2567.1
## - gender.dummy    1   2532.1 2568.1
## + MAXRAMNT        1   2528.2 2568.2
## + homeowner.dummy 1   2528.4 2568.4
## - zipconvert_2    1   2532.6 2568.6
## - zipconvert_3    1   2532.6 2568.6
## - zipconvert_4    1   2532.7 2568.7
## - zipconvert_5    1   2532.8 2568.8
## - NUMPROM         1   2532.8 2568.8
## - LASTGIFT        1   2533.4 2569.4
## - INCOME          1   2533.8 2569.8
## - totalmonths     1   2544.3 2580.3
##
## Step:  AIC=2564.61
## TARGET_B ~ zipconvert_2 + zipconvert_3 + zipconvert_4 + zipconvert_5 +
##     NUMCHLD + INCOME + gender.dummy + WEALTH + HV + Icmed + Icavg +
##     IC15 + NUMPROM + LASTGIFT + totalmonths + TIMELAG + AVGGIFT
##
##                  Df Deviance    AIC
## - IC15            1   2529.2 2563.2
## - AVGGIFT         1   2529.3 2563.3
## - WEALTH          1   2529.4 2563.4
## - Icmed           1   2529.5 2563.5
## - Icavg           1   2529.7 2563.7
## - HV              1   2530.2 2564.2
## <none>                2528.6 2564.6
```

```
## - NUMCHLD          1   2530.7 2564.7
## - TIMELAG          1   2531.4 2565.4
## - gender.dummy     1   2532.2 2566.2
## + RAMNTALL         1   2528.4 2566.4
## + MAXRAMNT         1   2528.6 2566.6
## + homeowner.dummy  1   2528.6 2566.6
## - zipconvert_2     1   2532.7 2566.7
## - zipconvert_3     1   2532.7 2566.7
## - zipconvert_4     1   2532.8 2566.8
## - zipconvert_5     1   2532.9 2566.9
## - NUMPROM          1   2533.3 2567.3
## - LASTGIFT         1   2533.5 2567.5
## - INCOME           1   2533.9 2567.9
## - totalmonths      1   2544.3 2578.3
##
## Step:  AIC=2563.24
## TARGET_B ~ zipconvert_2 + zipconvert_3 + zipconvert_4 + zipconvert_5 +
##     NUMCHLD + INCOME + gender.dummy + WEALTH + HV + Icmed + Icavg +
##     NUMPROM + LASTGIFT + totalmonths + TIMELAG + AVGGIFT
##
##                   Df Deviance    AIC
## - WEALTH           1   2529.9 2561.9
## - AVGGIFT          1   2530.0 2562.0
## - Icmed            1   2530.1 2562.1
## - Icavg            1   2530.8 2562.8
## <none>                 2529.2 2563.2
## - NUMCHLD          1   2531.3 2563.3
## - HV               1   2531.4 2563.4
## - TIMELAG          1   2532.0 2564.0
## + IC15             1   2528.6 2564.6
## - gender.dummy     1   2533.0 2565.0
## + RAMNTALL         1   2529.1 2565.1
## + MAXRAMNT         1   2529.2 2565.2
## + homeowner.dummy  1   2529.2 2565.2
## - zipconvert_2     1   2533.5 2565.5
## - zipconvert_3     1   2533.5 2565.5
## - zipconvert_4     1   2533.6 2565.6
## - zipconvert_5     1   2533.7 2565.7
## - NUMPROM          1   2533.7 2565.7
## - LASTGIFT         1   2534.1 2566.1
## - INCOME           1   2534.5 2566.5
## - totalmonths      1   2545.1 2577.1
##
## Step:  AIC=2561.86
## TARGET_B ~ zipconvert_2 + zipconvert_3 + zipconvert_4 + zipconvert_5 +
##     NUMCHLD + INCOME + gender.dummy + HV + Icmed + Icavg + NUMPROM +
##     LASTGIFT + totalmonths + TIMELAG + AVGGIFT
##
##                   Df Deviance    AIC
## - AVGGIFT          1   2530.6 2560.6
```

```
## - Icmed            1   2530.8 2560.8
## - Icavg            1   2531.3 2561.3
## <none>                 2529.9 2561.9
## - NUMCHLD          1   2531.9 2561.9
## - HV               1   2531.9 2561.9
## - TIMELAG          1   2532.5 2562.5
## + WEALTH           1   2529.2 2563.2
## + IC15             1   2529.4 2563.4
## - gender.dummy     1   2533.5 2563.5
## - NUMPROM          1   2533.7 2563.7
## + RAMNTALL         1   2529.7 2563.7
## + MAXRAMNT         1   2529.8 2563.8
## + homeowner.dummy  1   2529.9 2563.9
## - zipconvert_3     1   2534.2 2564.2
## - zipconvert_2     1   2534.2 2564.2
## - zipconvert_4     1   2534.4 2564.4
## - zipconvert_5     1   2534.4 2564.4
## - LASTGIFT         1   2534.7 2564.7
## - INCOME           1   2535.3 2565.3
## - totalmonths      1   2546.8 2576.8
##
## Step:  AIC=2560.57
## TARGET_B ~ zipconvert_2 + zipconvert_3 + zipconvert_4 + zipconvert_5 +
##     NUMCHLD + INCOME + gender.dummy + HV + Icmed + Icavg + NUMPROM +
##     LASTGIFT + totalmonths + TIMELAG
##
##                   Df Deviance    AIC
## - Icmed            1   2531.5 2559.5
## - Icavg            1   2532.1 2560.1
## <none>                 2530.6 2560.6
## - NUMCHLD          1   2532.6 2560.6
## - HV               1   2532.8 2560.8
## - TIMELAG          1   2533.3 2561.3
## + AVGGIFT          1   2529.9 2561.9
## + WEALTH           1   2530.0 2562.0
## - NUMPROM          1   2534.0 2562.0
## - gender.dummy     1   2534.1 2562.1
## + IC15             1   2530.2 2562.2
## + MAXRAMNT         1   2530.3 2562.3
## + RAMNTALL         1   2530.5 2562.5
## + homeowner.dummy  1   2530.6 2562.6
## - zipconvert_3     1   2535.0 2563.0
## - zipconvert_2     1   2535.0 2563.0
## - zipconvert_4     1   2535.2 2563.2
## - zipconvert_5     1   2535.2 2563.2
## - INCOME           1   2536.1 2564.1
## - LASTGIFT         1   2540.0 2568.0
## - totalmonths      1   2547.7 2575.7
##
## Step:  AIC=2559.5
```

```
## TARGET_B ~ zipconvert_2 + zipconvert_3 + zipconvert_4 + zipconvert_5 +
##     NUMCHLD + INCOME + gender.dummy + HV + Icavg + NUMPROM +
##     LASTGIFT + totalmonths + TIMELAG
##
##                   Df Deviance    AIC
## - Icavg            1   2532.3 2558.3
## <none>                 2531.5 2559.5
## - NUMCHLD          1   2533.6 2559.6
## - HV               1   2534.0 2560.0
## - TIMELAG          1   2534.1 2560.1
## + Icmed            1   2530.6 2560.6
## + AVGGIFT          1   2530.8 2560.8
## + WEALTH           1   2530.9 2560.9
## - NUMPROM          1   2534.9 2560.9
## - gender.dummy     1   2535.1 2561.1
## + IC15             1   2531.1 2561.1
## + MAXRAMNT         1   2531.3 2561.3
## + RAMNTALL         1   2531.5 2561.5
## + homeowner.dummy  1   2531.5 2561.5
## - zipconvert_2     1   2535.9 2561.9
## - zipconvert_3     1   2535.9 2561.9
## - zipconvert_4     1   2536.1 2562.1
## - zipconvert_5     1   2536.1 2562.1
## - INCOME           1   2537.1 2563.1
## - LASTGIFT         1   2540.9 2566.9
## - totalmonths      1   2548.6 2574.6
##
## Step:  AIC=2558.34
## TARGET_B ~ zipconvert_2 + zipconvert_3 + zipconvert_4 + zipconvert_5 +
##     NUMCHLD + INCOME + gender.dummy + HV + NUMPROM + LASTGIFT +
##     totalmonths + TIMELAG
##
##                   Df Deviance    AIC
## - HV               1   2534.1 2558.1
## <none>                 2532.3 2558.3
## - NUMCHLD          1   2534.5 2558.5
## - TIMELAG          1   2534.8 2558.8
## + IC15             1   2531.2 2559.2
## + Icavg            1   2531.5 2559.5
## + AVGGIFT          1   2531.5 2559.5
## - NUMPROM          1   2535.9 2559.9
## - gender.dummy     1   2535.9 2559.9
## + Icmed            1   2532.1 2560.1
## + WEALTH           1   2532.1 2560.1
## + MAXRAMNT         1   2532.1 2560.1
## + RAMNTALL         1   2532.3 2560.3
## + homeowner.dummy  1   2532.3 2560.3
## - zipconvert_3     1   2536.7 2560.7
## - zipconvert_2     1   2536.7 2560.7
## - zipconvert_4     1   2536.8 2560.8
```

```
## - zipconvert_5     1   2536.9 2560.9
## - INCOME           1   2537.2 2561.2
## - LASTGIFT         1   2541.6 2565.6
## - totalmonths      1   2549.1 2573.1
##
## Step:  AIC=2558.12
## TARGET_B ~ zipconvert_2 + zipconvert_3 + zipconvert_4 + zipconvert_5 +
##     NUMCHLD + INCOME + gender.dummy + NUMPROM + LASTGIFT + totalmonths +
##     TIMELAG
##
##                   Df Deviance    AIC
## <none>                 2534.1 2558.1
## + HV               1   2532.3 2558.3
## - NUMCHLD          1   2536.3 2558.3
## - TIMELAG          1   2536.4 2558.4
## + AVGGIFT          1   2533.2 2559.2
## - gender.dummy     1   2537.4 2559.4
## + WEALTH           1   2533.5 2559.5
## - NUMPROM          1   2537.5 2559.5
## + Icmed            1   2533.8 2559.8
## + IC15             1   2533.8 2559.8
## + MAXRAMNT         1   2533.9 2559.9
## + Icavg            1   2534.0 2560.0
## + RAMNTALL         1   2534.1 2560.1
## + homeowner.dummy  1   2534.1 2560.1
## - zipconvert_3     1   2538.3 2560.3
## - zipconvert_2     1   2538.4 2560.4
## - zipconvert_4     1   2538.5 2560.5
## - zipconvert_5     1   2538.8 2560.8
## - INCOME           1   2541.1 2563.1
## - LASTGIFT         1   2542.7 2564.7
## - totalmonths      1   2551.2 2573.2
```

```
summary(step.model)
```

```
##
## Call:
## glm(formula = TARGET_B ~ zipconvert_2 + zipconvert_3 + zipconvert_4 +
##     zipconvert_5 + NUMCHLD + INCOME + gender.dummy + NUMPROM +
##     LASTGIFT + totalmonths + TIMELAG, family = "binomial", data =
train.df)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.7243  -1.1705   0.8174   1.1374   2.1058
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -12.317709 308.975776  -0.040  0.96820
## zipconvert_2  13.610016 308.975485   0.044  0.96487
```

```
## zipconvert_3  13.592923 308.975485   0.044  0.96491
## zipconvert_4  13.652887 308.975486   0.044  0.96475
## zipconvert_5  13.742982 308.975478   0.044  0.96452
## NUMCHLD       -0.209066   0.141897  -1.473  0.14066
## INCOME         0.077901   0.029492   2.641  0.00826 **
## gender.dummy   0.175827   0.096307   1.826  0.06790 .
## NUMPROM        0.004029   0.002183   1.846  0.06494 .
## LASTGIFT      -0.014207   0.005163  -2.752  0.00593 **
## totalmonths   -0.050861   0.012456  -4.083 4.44e-05 ***
## TIMELAG        0.013125   0.008665   1.515  0.12983
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2594.2  on 1871  degrees of freedom
## Residual deviance: 2534.1  on 1860  degrees of freedom
## AIC: 2558.1
##
## Number of Fisher Scoring iterations: 12
```

```r
# Reduced model (after dropping "zipconvert_2", "zipconvert_3",
"zipconvert_4", "zipconvert_5" due to extremely high p-values).
reduced.model <- glm(TARGET_B ~ NUMCHLD + INCOME + NUMPROM + LASTGIFT +
totalmonths, data = train.df, family = "binomial")
summary(reduced.model)
```

```
##
## Call:
## glm(formula = TARGET_B ~ NUMCHLD + INCOME + NUMPROM + LASTGIFT +
##     totalmonths, family = "binomial", data = train.df)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.7351  -1.1775   0.8246   1.1441   2.1282
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  1.482309   0.463944   3.195  0.00140 **
## NUMCHLD     -0.217947   0.140946  -1.546  0.12203
## INCOME       0.078958   0.029211   2.703  0.00687 **
## NUMPROM      0.004530   0.002153   2.104  0.03534 *
## LASTGIFT    -0.013731   0.005119  -2.682  0.00731 **
## totalmonths -0.049771   0.012339  -4.034 5.49e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2594.2  on 1871  degrees of freedom
```

```
## Residual deviance: 2546.0  on 1866  degrees of freedom
## AIC: 2558
##
## Number of Fisher Scoring iterations: 4
```

```
# Further reduced model/final model (after dropping "NUMPROM" due to p-value
> 10%).
final.model <- glm(TARGET_B ~ NUMCHLD + INCOME + LASTGIFT + totalmonths, data
= train.df, family = "binomial")
summary(final.model)
```

```
##
## Call:
## glm(formula = TARGET_B ~ NUMCHLD + INCOME + LASTGIFT + totalmonths,
##     family = "binomial", data = train.df)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.7149  -1.1810   0.8426   1.1420   2.0950
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  1.969789   0.401404   4.907 9.24e-07 ***
## NUMCHLD     -0.244870   0.140320  -1.745  0.08097 .
## INCOME       0.074661   0.029094   2.566  0.01028 *
## LASTGIFT    -0.013805   0.005107  -2.703  0.00687 **
## totalmonths -0.056674   0.011841  -4.786 1.70e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2594.2  on 1871  degrees of freedom
## Residual deviance: 2550.5  on 1867  degrees of freedom
## AIC: 2560.5
##
## Number of Fisher Scoring iterations: 4
```

```
reduced.df<-df[,c(6,7,17)]
head(reduced.df)
```

```
##   NUMCHLD INCOME LASTGIFT
## 1       1      5        5
## 2       1      1       12
## 3       2      5        5
## 4       1      3        8
## 5       1      4       11
## 6       1      4       10
```

A logistic regression analysis was conducted using all the predictor variables initially on the training set. The model was further refined by using stepwise regression to eliminate predictors that were not significant. The predictors for our final model include NUMCHLD, INCOME, LASTGIFT, and totalmonths. We used Excel to calculate the optimum cutoff value for the model. A value of 0.498 resulted in the maximum profit and minimum cost. Therefore 0.5 was set as the cutoff for classifying the entries in the model. The accuracy of the model obtained from the validation data is 50.32%

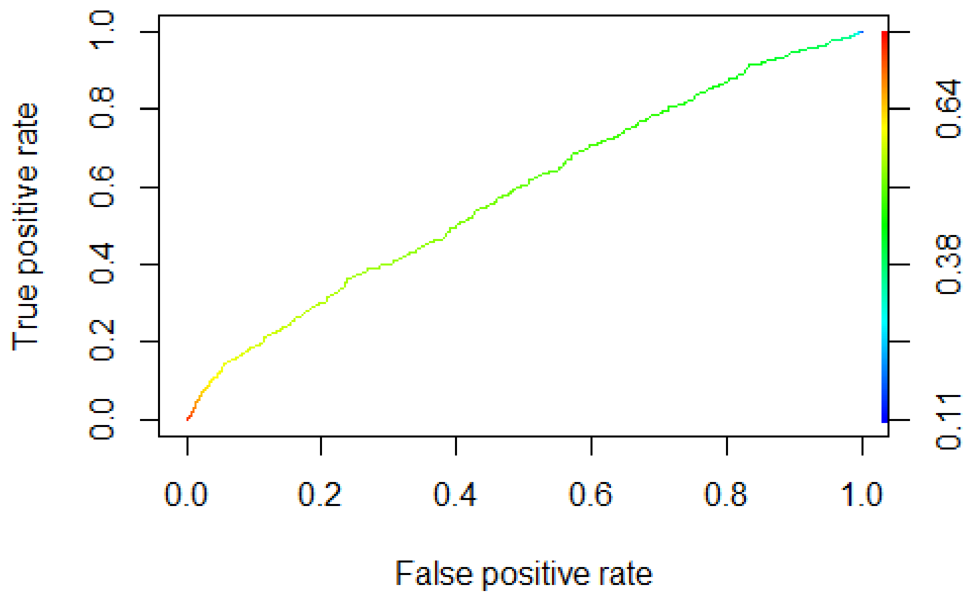
```
predict_valid <- predict(final.model, valid.df, type = "response")
table(valid.df$TARGET_B,predict_valid>0.5)

##
##     FALSE TRUE
##   0   311  334
##   1   194  409

#Profit from the validation data
Profit_valid<-(358*((13-0.68)/9.8))
Profit_valid

## [1] 450.0571

#cost from the validation data
cost_valid<-(272*((0-0.68)/0.53))
cost_valid

## [1] -348.9811
```

Net profit for donors = $13 - $0.68 = $12.32 Net profit for non-donors = -$0.68 Adjusted net profit for donors = $12.32 / 9.38 = $1.3134 Adjusted net profit for non-donors = -$0.68 / 0.53 = -$1.283 For the first model, net profit for validation data = $450.0571 - $348.9811 = $101.076

```
predict_train <- predict(final.model, train.df, type = "response")
table(train.df$TARGET_B,predict_train>0.5)

##
##     FALSE TRUE
##   0   436  479
##   1   356  601

#Profit from the training data
Profit_train<-(533*((13-0.68)/9.8))
Profit_train

## [1] 670.0571

#cost from the training data
cost_train<-(397*((0-0.68)/0.53))
cost_train

## [1] -509.3585
```

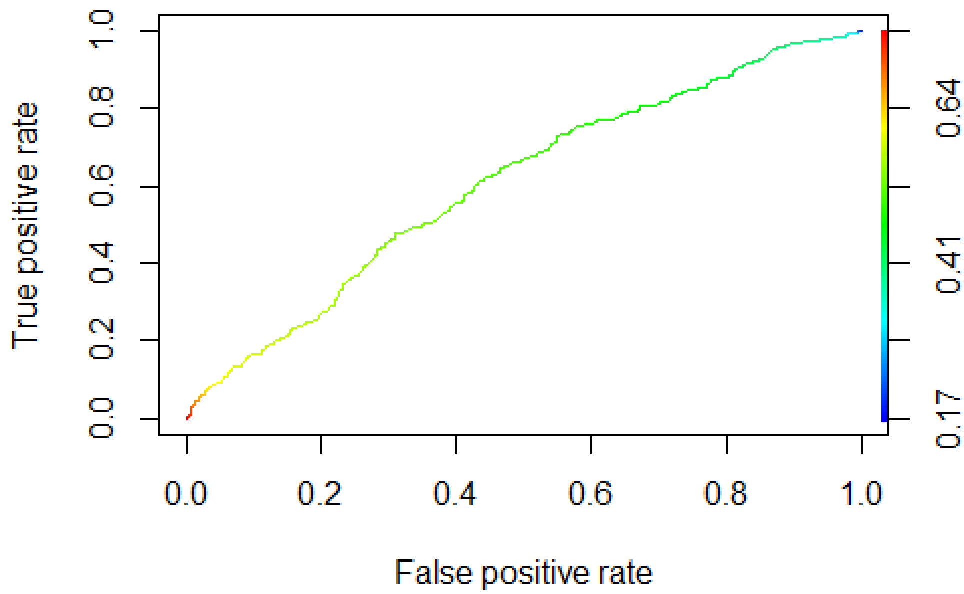
Net profit for the training data = $670.0571 - $509.3585 = $160.6986

```r
#ROCR Curve

ROCRpred <- prediction(predict_train, train.df$TARGET_B)
ROCRperf <- performance(ROCRpred, 'tpr','fpr')
plot(ROCRperf, colorize = TRUE, text.adj = c(-0.2,1.7))
```



The above Roc plot is for the training dataset

```r
ROCRpred <- prediction(predict_valid, valid.df$TARGET_B)
ROCRperf <- performance(ROCRpred, 'tpr','fpr')
plot(ROCRperf, colorize = TRUE, text.adj = c(-0.2,1.7))
```



The above ROC plot is for the validation dataset. The ROC curve gives a better indication of the model performance in cases where asymmetric costs are involved by plotting the sensitivity over specificity. The closer the curve gets to the top-left corner, the better is the performance. In this case, the ROC curve signifies a slightly better performance than the naive rule which would have yielded a diagonal. The model was run on the FutureFundraising dataset to classify the entries.

```
predict_test <- predict(final.model, test.df, type = "response")
test_TARGET_B <- ifelse(predict_test>0.5,1,0)
head(test_TARGET_B)

## 1 2 3 4 5 6
## 1 0 0 1 1 1
```

The probabilities of the entries in FutureFundraising dataset has been sorted in descending order.

```
pred_test <- predict(final.model, newdata = test.df, type = "response")
fund.pred <- pred_test[order(-pred_test)]
head(fund.pred)

##       854      1460      1755         9       120      1610
## 0.7733655 0.7711937 0.7567172 0.7482622 0.7448589 0.7374373
```

SVM

Here,We used SVM to build our second model. The svm() function was used to fit the model to the training data.

```
library(e1071)

df <- read.csv("Fundraising.csv",header = T)
head(df)

##   Row.Id Row.Id. zipconvert_2 zipconvert_3 zipconvert_4 zipconvert_5
## 1      1      17            0            1            0            0
## 2      2      25            1            0            0            0
## 3      3      29            0            0            0            1
## 4      4      38            0            0            0            1
## 5      5      40            0            1            0            0
## 6      6      53            0            1            0            0
##   homeowner.dummy NUMCHLD INCOME gender.dummy WEALTH   HV Icmed Icavg IC15
## 1               1       1      5            1      9 1399   637   703    1
## 2               1       1      1            0      7  698   422   463    4
## 3               0       2      5            1      8  828   358   376   13
## 4               1       1      3            0      4 1471   484   546    4
## 5               1       1      4            0      8  547   386   432    7
## 6               1       1      4            1      8  482   242   275   28
##   NUMPROM RAMNTALL MAXRAMNT LASTGIFT totalmonths TIMELAG  AVGGIFT TARGET_B
## 1      74      102        6        5          29       3 4.857143        1
## 2      46       94       12       12          34       6 9.400000        1
## 3      32       30       10        5          29       7 4.285714        1
## 4      94      177       10        8          30       3 7.080000        0
## 5      20       23       11       11          30       6 7.666667        0
## 6      38       73       10       10          31       3 7.300000        1
##   TARGET_D
## 1        5
## 2       10
## 3        5
## 4        0
## 5        0
## 6        8

test.df<-read.csv("FutureFundraising.csv",header = T)
## STEP #0:Removing unnecessary columns from our dataset.

df$Row.Id  <- NULL
df$Row.Id. <- NULL
df$TARGET_D <- NULL

# Encoding the target feature as factor
df$TARGET_B = factor(df$TARGET_B, levels = c(0, 1))

## STEP #1: Partitioning our data. (60% Training, 40% Validation).

train.rows <- sample(rownames(df), dim(df)[1]*0.6)
train.df <- df[train.rows, ]
```

```r
valid.rows <- setdiff(rownames(df), train.rows)
valid.df <- df[valid.rows, ]

## STEP #2: Model Building.

attach(train.df)
attach(test.df)

## The following objects are masked from train.df:
##
##     AVGGIFT, gender.dummy, homeowner.dummy, HV, IC15, Icavg,
##     Icmed, INCOME, LASTGIFT, MAXRAMNT, NUMCHLD, NUMPROM, RAMNTALL,
##     TARGET_B, TIMELAG, totalmonths, WEALTH, zipconvert_2,
##     zipconvert_3, zipconvert_4, zipconvert_5

# Fitting SVM to the Training set


classifier = svm(formula = TARGET_B ~ .,  data = train.df, type = 'C-
classification', kernel = 'linear')
classifier

##
## Call:
## svm(formula = TARGET_B ~ ., data = train.df, type = "C-classification",
##     kernel = "linear")
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  linear
##        cost:  1
##
## Number of Support Vectors:  1713

# Predicting the Test set results for validation data
y_pred_valid = predict(classifier, newdata = valid.df)
y_pred_valid

##    4   6   7  11  12  13  17  20  27  29  31  32  33  36  40
##    1   0   0   1   0   0   1   1   0   0   0   1   1   0   1
##   41  44  49  51  52  55  57  64  66  69  70  72  73  77  82
##    1   0   1   1   0   0   1   1   1   1   1   0   0   0   0
##   87  89  91  94  95  96 100 103 105 107 108 109 119 120 122
##    0   1   0   1   1   0   0   0   0   0   0   0   1   0   0
##  124 129 131 134 136 137 138 139 140 141 144 145 149 155 157
##    0   1   1   1   0   0   0   1   1   0   0   1   0   1   1
##  158 161 165 168 169 171 175 177 178 179 180 184 187 190 191
##    1   1   0   1   1   1   1   0   0   1   0   0   1   0   1
##  193 198 199 204 206 212 215 217 220 221 224 230 231 234 235
```

```
##    0    0    0    0    1    0    0    0    1    1    1    0    1    0    0
##  236  246  247  253  256  257  259  261  263  265  267  268  270  274  277
##    0    0    1    1    0    1    1    0    1    1    1    0    0    0    0
##  282  285  286  293  295  296  301  303  308  309  314  317  318  319  323
##    0    1    0    1    0    0    1    1    0    1    1    1    0    1    0
##  329  330  334  339  341  343  344  346  350  353  355  359  362  363  368
##    1    1    1    1    1    0    0    1    0    0    1    0    1    0    0
##  373  374  375  381  385  386  388  391  393  395  396  399  410  411  414
##    1    0    1    0    0    0    0    0    1    1    1    1    1    0    0
##  418  420  421  422  424  425  426  433  437  441  446  454  458  460  463
##    1    0    1    0    1    1    0    0    1    1    0    1    0    1    1
##  469  473  474  476  478  479  481  482  485  486  496  497  499  502  509
##    1    1    1    0    1    1    1    0    1    1    0    1    1    0    0
##  513  514  516  517  519  520  524  525  526  528  529  530  531  540  545
##    1    1    1    1    0    1    1    1    0    1    0    1    1    0    0
##  547  549  550  552  553  555  557  558  560  562  576  577  582  588  591
##    0    0    1    1    1    1    0    0    0    0    0    1    0    1    1
##  592  593  594  600  602  604  605  608  611  617  619  622  625  629  632
##    0    0    0    1    0    1    0    0    0    1    0    1    1    0    0
##  633  634  635  637  641  642  643  644  645  648  650  651  654  656  658
##    1    1    1    0    0    1    0    1    0    0    1    0    1    0    0
##  659  660  661  662  667  670  672  673  679  681  682  684  687  688  690
##    0    1    1    1    1    0    1    1    1    0    1    0    1    1    1
##  691  694  697  699  702  707  709  710  714  715  716  721  725  732  733
##    0    1    0    1    1    0    1    1    0    0    0    1    1    0    0
##  734  735  738  739  742  746  748  751  755  758  759  760  763  765  769
##    1    0    0    0    1    1    0    1    0    0    0    0    1    1    0
##  770  774  777  778  780  785  786  794  795  796  798  803  805  807  808
##    1    1    1    0    1    0    0    1    1    0    0    1    0    1    1
##  809  812  814  815  817  821  822  824  825  827  831  840  841  844  846
##    0    0    0    1    0    0    1    0    0    1    1    0    0    0    1
##  848  849  850  859  861  862  863  866  871  873  875  876  877  885  887
##    0    1    0    0    1    0    1    1    1    0    1    0    0    1    1
##  890  891  892  894  895  898  899  900  901  902  904  905  906  907  908
##    1    1    0    0    0    1    1    1    0    0    0    0    0    1    1
##  910  911  916  917  918  920  922  923  934  937  938  942  943  944  945
##    0    1    1    0    0    1    0    1    1    1    0    1    0    0    1
##  947  954  955  956  957  958  966  969  972  975  978  981  982  987  991
##    1    1    1    1    0    1    1    1    0    1    1    1    1    1    0
##  993  994  995  997 1001 1004 1011 1012 1013 1015 1016 1022 1025 1026 1027
##    0    0    0    0    1    1    1    0    0    0    0    1    0    1    0
## 1028 1032 1034 1037 1040 1041 1045 1049 1053 1054 1055 1056 1060 1063 1067
##    1    1    1    1    0    1    1    0    0    1    0    0    0    0    0
## 1069 1070 1073 1074 1075 1080 1085 1086 1087 1088 1089 1090 1094 1097 1101
##    0    1    1    0    0    1    1    0    1    1    1    0    1    1    0
## 1109 1111 1113 1114 1117 1122 1123 1124 1128 1131 1133 1137 1143 1145 1146
##    0    0    1    1    1    0    1    1    1    0    0    1    0    0    0
## 1147 1150 1151 1152 1154 1155 1157 1158 1163 1165 1168 1169 1173 1176 1179
##    0    0    1    0    0    1    0    1    0    0    1    0    1    1    1
## 1181 1182 1183 1185 1186 1188 1189 1190 1191 1192 1193 1194 1199 1201 1202
```

```
##    1    1    0    1    0    1    1    1    0    1    1    0    0    0    0
## 1205 1209 1215 1218 1219 1220 1223 1226 1229 1232 1237 1243 1244 1246 1248
##    0    1    1    1    0    1    1    0    1    0    1    1    0    0    0
## 1251 1252 1254 1256 1258 1260 1261 1262 1263 1264 1265 1267 1272 1278 1280
##    0    1    1    0    1    1    0    1    1    0    1    1    0    1    0
## 1285 1286 1288 1291 1292 1293 1295 1298 1303 1304 1306 1307 1310 1311 1313
##    0    1    1    0    1    0    1    1    1    1    0    1    1    0    1
## 1314 1315 1316 1317 1320 1322 1325 1329 1330 1331 1333 1334 1336 1342 1343
##    1    0    1    1    1    1    1    1    1    1    1    0    0    0    0
## 1344 1348 1352 1354 1355 1358 1359 1360 1362 1364 1366 1367 1368 1369 1376
##    0    0    0    1    1    1    1    1    0    0    0    0    0    0    0
## 1377 1380 1381 1383 1391 1392 1397 1401 1402 1403 1404 1406 1408 1414 1418
##    0    0    1    1    0    0    1    0    0    1    1    0    0    0    0
## 1420 1422 1424 1425 1431 1435 1438 1441 1444 1451 1455 1459 1460 1463 1466
##    0    1    0    1    0    1    1    0    1    1    1    0    0    0    1
## 1468 1469 1472 1473 1474 1475 1477 1487 1488 1489 1490 1491 1495 1496 1500
##    0    1    0    0    1    1    1    0    0    1    0    0    1    1    0
## 1501 1502 1506 1513 1516 1518 1519 1523 1526 1528 1529 1531 1532 1533 1536
##    0    0    0    1    0    0    0    0    1    0    0    1    0    0    1
## 1538 1539 1540 1542 1543 1544 1545 1547 1551 1552 1554 1560 1561 1562 1563
##    0    0    0    1    1    0    0    1    0    1    1    1    1    0    0
## 1564 1566 1567 1576 1577 1578 1580 1583 1585 1586 1587 1588 1592 1595 1596
##    0    1    0    1    1    0    1    1    1    1    0    1    1    0    1
## 1597 1600 1601 1602 1605 1610 1612 1613 1619 1621 1627 1630 1631 1635 1640
##    0    1    1    1    0    0    1    0    0    0    0    1    1    0    0
## 1643 1644 1649 1651 1654 1655 1658 1660 1661 1663 1664 1665 1668 1670 1674
##    0    0    0    1    0    0    0    1    1    1    1    0    1    0    0
## 1675 1677 1678 1679 1680 1681 1683 1685 1688 1689 1690 1691 1692 1693 1697
##    0    0    0    1    1    0    0    1    1    1    1    1    0    1    1
## 1700 1701 1702 1705 1706 1712 1717 1718 1720 1728 1734 1736 1741 1743 1746
##    1    1    0    0    0    0    1    1    1    0    0    1    1    0    0
## 1749 1750 1751 1757 1758 1764 1765 1766 1767 1770 1771 1776 1779 1781 1782
##    0    1    0    1    1    1    1    1    1    0    1    1    0    1    1
## 1784 1791 1794 1796 1798 1801 1806 1808 1811 1815 1823 1824 1826 1830 1846
##    0    0    0    0    1    1    0    1    0    0    0    0    1    0    1
## 1848 1853 1855 1856 1859 1862 1863 1865 1866 1876 1877 1879 1881 1883 1886
##    0    1    1    1    0    0    0    0    0    1    0    0    1    0    1
## 1888 1890 1891 1893 1895 1898 1900 1901 1905 1915 1916 1917 1925 1926 1929
##    1    0    0    0    1    0    1    1    0    0    1    0    1    0    0
## 1931 1935 1942 1946 1950 1952 1955 1956 1958 1961 1962 1964 1967 1972 1975
##    1    1    0    1    1    1    1    1    1    1    1    1    1    0    0
## 1980 1981 1982 1983 1986 1989 1991 1993 1996 1998 2002 2007 2008 2011 2012
##    1    1    1    0    0    1    0    0    1    0    0    0    1    1    0
## 2016 2020 2023 2024 2025 2034 2035 2036 2039 2040 2041 2045 2048 2051 2054
##    1    1    0    0    0    0    0    0    1    0    1    1    0    1    1
## 2057 2058 2059 2063 2066 2067 2068 2069 2070 2074 2075 2076 2079 2080 2085
##    1    1    1    1    0    0    1    0    1    1    0    0    0    1    1
## 2086 2088 2090 2091 2095 2096 2098 2099 2101 2102 2111 2112 2114 2117 2118
##    0    0    1    0    0    1    0    0    1    1    1    0    0    0    1
## 2120 2121 2124 2125 2126 2127 2134 2135 2136 2139 2140 2143 2144 2146 2148
```

```
##    0    0    1    0    0    0    1    1    1    0    0    0    0    1    0
## 2149 2151 2153 2154 2159 2161 2164 2165 2166 2167 2171 2172 2173 2178 2179
##    0    0    0    0    0    1    0    0    0    1    1    0    1    1    1
## 2180 2182 2183 2185 2186 2188 2194 2195 2197 2202 2204 2205 2206 2209 2210
##    0    1    0    0    0    1    1    1    0    1    0    0    1    1    0
## 2212 2216 2218 2219 2221 2224 2226 2229 2231 2234 2235 2236 2237 2242 2243
##    0    0    1    0    1    0    0    0    1    1    1    0    0    0    1
## 2244 2245 2247 2248 2250 2257 2259 2262 2266 2267 2271 2272 2273 2274 2275
##    1    1    0    0    1    0    0    1    1    1    0    1    1    1    0
## 2276 2280 2283 2292 2294 2296 2297 2299 2303 2304 2305 2310 2311 2313 2314
##    0    0    1    1    1    1    0    0    0    1    0    0    0    0    1
## 2315 2316 2317 2321 2322 2325 2330 2331 2335 2336 2337 2343 2352 2354 2355
##    0    0    1    0    0    1    0    0    0    0    0    1    0    1    0
## 2356 2357 2358 2359 2360 2361 2365 2366 2368 2370 2371 2373 2375 2376 2379
##    0    0    1    1    1    1    0    0    1    0    1    0    0    1    1
## 2382 2388 2390 2392 2393 2394 2395 2397 2401 2404 2407 2408 2410 2413 2417
##    0    1    0    1    1    1    0    0    1    1    1    1    1    0    0
## 2422 2427 2428 2431 2432 2433 2434 2435 2437 2442 2444 2445 2447 2449 2452
##    0    0    1    1    0    0    0    0    0    1    0    0    1    1    1
## 2454 2457 2459 2460 2461 2465 2473 2475 2486 2490 2494 2496 2497 2500 2501
##    1    0    1    1    1    0    1    1    1    1    1    1    0    1    1
## 2507 2508 2509 2511 2514 2515 2518 2521 2522 2523 2528 2530 2531 2534 2535
##    0    0    0    0    1    0    1    0    1    1    1    1    0    0    0
## 2537 2538 2541 2543 2548 2553 2554 2555 2556 2557 2563 2567 2569 2570 2571
##    0    0    1    1    1    0    1    0    1    1    1    1    1    0    0
## 2575 2577 2578 2579 2581 2583 2584 2586 2589 2593 2596 2597 2599 2601 2602
##    0    0    0    0    1    1    1    1    0    1    0    1    0    0    0
## 2605 2608 2610 2612 2614 2619 2620 2621 2622 2627 2632 2635 2637 2641 2644
##    0    1    1    0    1    1    0    1    0    1    1    0    1    0    0
## 2651 2652 2653 2655 2657 2658 2659 2660 2666 2668 2670 2671 2672 2676 2678
##    0    0    0    1    0    0    1    1    1    0    0    0    0    0    0
## 2679 2680 2688 2692 2693 2695 2696 2697 2699 2702 2710 2712 2714 2718 2719
##    1    1    1    0    0    0    1    0    0    0    1    0    0    0    1
## 2720 2721 2722 2723 2732 2734 2735 2740 2742 2746 2749 2751 2755 2759 2762
##    1    1    1    0    1    1    1    1    1    1    1    0    1    1    0
## 2765 2766 2770 2773 2778 2780 2781 2783 2784 2787 2788 2791 2794 2798 2800
##    1    1    1    0    1    1    0    1    1    1    0    0    1    0    0
## 2805 2806 2808 2809 2810 2813 2815 2818 2819 2821 2824 2825 2827 2830 2831
##    0    0    1    0    0    0    1    1    1    0    1    1    1    1    0
## 2833 2840 2841 2842 2850 2851 2852 2853 2854 2855 2858 2860 2864 2866 2870
##    1    0    1    0    0    1    0    1    1    1    1    0    1    1    1
## 2871 2877 2878 2880 2881 2883 2884 2885 2886 2890 2893 2899 2900 2903 2905
##    1    0    1    0    1    1    0    1    1    1    1    0    0    1    0
## 2906 2907 2909 2910 2913 2917 2918 2923 2926 2931 2937 2940 2941 2945 2949
##    0    0    1    0    0    0    1    0    1    0    0    1    1    0    1
## 2951 2952 2954 2956 2960 2961 2963 2967 2968 2970 2972 2973 2974 2975 2977
##    1    0    1    0    0    0    1    1    1    0    0    0    0    1    0
## 2980 2984 2986 2988 2989 2990 2994 2997 2999 3000 3001 3004 3007 3009 3011
##    1    1    1    1    0    1    1    0    0    1    0    0    0    0    0
## 3012 3014 3017 3018 3020 3022 3025 3028 3030 3033 3038 3039 3041 3043 3044
```

```
##    0    0    0    0    1    1    1    1    0    0    0    0    1    0    0
## 3047 3048 3051 3052 3054 3056 3060 3062 3064 3066 3068 3071 3072 3079 3081
##    1    1    1    1    0    0    0    0    1    1    1    1    1    0    1
## 3085 3088 3091 3092 3093 3096 3099 3101 3102 3103 3106 3108 3112 3113 3115
##    1    1    1    0    1    0    1    1    0    1    1    1    0    1    0
## 3116 3117 3119
##    1    0    1
## Levels: 0 1
```

```
# Making the Confusion Matrix for validation data
cm_valid = table(valid.df$TARGET_B, y_pred_valid)
cm_valid
```

```
##    y_pred_valid
##       0   1
##   0 344 291
##   1 278 335
```

```
#Profit from the validation data
Profit_valid<-(362*((13-0.68)/9.8))
Profit_valid
```

```
## [1] 455.0857
```

```
#cost from the validation data
cost_valid<-(253*((0-0.68)/0.53))
cost_valid
```

```
## [1] -324.6038
```

```
# Predicting the Test set results for training data
y_pred_train = predict(classifier, newdata = train.df)
y_pred_train
```

```
## 2814 2862 1079   10 2260 2387  747  855 3075 1000 1373  195  678  299 1604
##    0    1    0    0    1    0    1    0    0    0    1    0    1    0    0
## 2028 2426 1227 2189   28  609 2683 1269 1747 1127 1797 1875  813 1082 1115
##    1    1    0    0    0    0    0    1    0    1    1    1    1    0    0
## 2927 2350 2779 1618  712 1483 2309  708 2108  159  539  864  356 2492 1212
##    0    1    1    1    0    0    0    0    1    1    1    0    1    1    0
## 1149  214 2348 1930 2026 1284  250  988 2643  242  394 2391 1593  237 2093
##    0    1    1    0    1    1    1    0    1    0    1    0    1    1    1
## 1039 1503 3105 2222 2628 1224 2896 2959   35  535 1413 2230 2346 1819 1051
##    1    1    0    1    0    0    1    0    1    0    0    0    1    1    0
## 2731  192 2690  472 2562 2630 2256 2279 1966 1558  313 2835 2647  686  930
##    0    0    1    0    0    0    0    0    1    1    0    1    1    0    1
##  705  757 2326 1709 1048 2845  337  371  685 2525  367 2820 1175 2745  597
##    1    0    1    0    1    0    1    0    0    1    0    0    1    0    1
##  150 3032  596 2929  542  929 2744 3061 2978 1372 1579 1370 2912 2290 1257
##    1    0    0    0    1    0    1    0    0    1    1    1    0    1    0
##   99 1030 2295 2736 1099 2287  442  182 2774 3046 1356 1974 2491  183 1233
##    1    0    0    0    0    1    1    1    1    0    1    0    0    1    0
```

```
## 1522  440  415  312 1571 1812  232 1628 1949 1361 1608  439 2420  310 1726
##    0    0    0    0    1    0    1    1    1    1    0    0    0    1    0
##  186 1200 2233 1903  722 2374 2253 1922 1948  445 1029   86  538 1511 2873
##    1    0    0    0    1    0    1    1    1    1    0    1    1    0    0
##  522 2532  470 1399 2440 1756 1603 1429 1046 3078 2377  962 1517 2207 1803
##    0    0    0    0    0    1    0    0    1    1    1    1    1    0    0
## 2656  273 1860 2758 1748  737   98  884 1222 2935 2836 2415  429 2061  114
##    0    1    0    1    1    1    1    0    1    0    0    1    1    1    1
## 3087 2933  543 1907 2014 1446  998  160  345 2738 1584  559  578 2876 1482
##    1    0    1    1    0    1    0    1    0    1    1    1    0    0    1
##  251 1119  971  432  704 1851 2030  143  238 2421 2466  306 2286   15 2673
##    0    1    1    1    0    0    1    0    0    1    1    0    1    0    1
##   81  882  431 1238 2502 2849 1738 2958 2293  788  556 1386 2203 2225 1378
##    1    0    0    0    1    0    1    1    1    0    0    1    0    1    1
## 1458 2409 2227  147 1739  222  563 1081  583  383 2598 1775  142  512   88
##    0    0    1    0    0    1    0    1    1    1    1    0    0    1    1
##  804 2142 2184 1174  280   48  360 1433  216 1954 1614 2369  271 2323 2661
##    1    0    1    0    0    1    0    1    0    0    1    0    1    0    0
## 3098   79 1838  249 1909 2319 2861 1461 1795  321 1270 2646 1737 2031 1415
##    1    1    0    1    1    1    0    0    1    0    0    1    0    1    0
##   19 1699 1019 3069    9 1464 3111 2513 2957 1787 2405 1437  797 2084  896
##    0    1    1    0    0    0    0    0    0    1    1    1    1    0    0
##  639 2122 1407 2451 3042  148 2468 1379 2812 1196 1221 2700  197  447 3031
##    0    1    1    0    1    0    1    1    0    1    0    0    1    1    1
##  784 1198 1575  766  508 2341  121  674 1745 1947 2689   50 1868   63   61
##    0    1    1    1    0    1    1    0    1    1    0    0    1    0    1
##  328 2757 1187  671 1874 2258 3118 2769  897 2092 2160  598 3095 2089  276
##    0    1    1    1    0    0    1    0    0    0    1    1    1    1    1
## 2705 2483   38 1098 2334 2493 2789  484 2241 2767 1510  118 1208  761 1419
##    0    0    1    0    0    0    1    1    1    1    0    0    1    0    1
##  912  417 1759 1493 2981 1928 1857 1648  229 1897 2962 1805 1389 2029 1973
##    0    1    1    0    0    0    1    0    1    1    1    1    0    1    1
## 2308 1250 1282 2707  985 2381  492 2869 2925 2254 2302 2985 1102   80 1231
##    1    0    0    0    1    0    1    1    1    1    1    1    1    0    0
## 2901 1410 1632 2636 2887 2499 2512 2269 2104  847 1836  762 2874  787 2625
##    1    0    1    0    1    1    1    1    0    0    0    0    0    1    1
## 1778 1641 1480 1018  352  799 1120 3067  782 1300 2201 1616 1638 1710 1275
##    0    0    0    1    1    0    0    1    1    1    0    1    1    0    1
##   24 2549 1711  868 2065 1997 1761 2482 1214 1134  915 2462 2320  615 2726
##    1    1    1    1    1    0    0    0    0    1    0    1    0    0    1
##  837 1309  409  696 1581 2991 2469 2803 2333 3045 2129 1318 2645 2298 2669
##    1    1    1    1    1    0    1    1    1    1    1    1    0    0    1
## 2050 1820 1565 2639  657 1880 2856  201  451 1017 2064  406 1339 1240 1714
##    0    1    1    0    1    0    1    0    1    0    0    1    1    1    0
## 2600 2384  258  503 2510 1326 1035  376  872  416  970 1457 1210  163  505
##    0    1    0    0    1    0    1    0    1    1    0    1    1    1    0
##  468   34 2544  724 1234 2332 3036 2270 2729 1943  626 2526  680  401 2955
##    1    1    1    1    0    0    1    1    0    0    1    0    0    0    0
##   65 2081 2754 1999 1281 2934 2792  854 1167  567 1774 2327 1732  209 1091
##    1    0    1    0    0    0    0    1    1    1    1    1    0    0    0
```

```
## 2300 2479 2545   92 1792 2463 1385 1132  856 2772 2649 2615 1598  730  117
##    1    0    1    1    0    1    1    0    1    1    0    0    1    1    0
##  335  461 2606 2291 2455  434 1274 3029 1642  284 1064 1960 2363 2703 1590
##    1    0    0    0    1    0    1    0    1    0    0    0    0    0    0
## 2027 1633 1854  624  801  649 1772  603  194 1969 1829  435  548 1646 1305
##    0    0    0    1    0    1    1    0    0    1    1    0    1    1    0
## 2003  527 1727 2706 1448   83 1347 2966 3059  800  211  948 2547 2071 1724
##    1    0    1    1    1    0    1    0    1    0    1    1    0    1    0
## 1162 2367  297  397 2013 1375  294  860  407  400 2228  783 3008 2993  357
##    1    1    1    1    0    1    1    1    1    0    0    1    0    1    1
## 2953 1951 2470  980 1136 2979 2163  428  311 2551 2288  829 2282  126  960
##    1    1    1    0    1    0    1    1    0    1    1    0    1    1    1
##   59 2730  487 1405 1976  806  449 2591 2498  326 2489 2588 2015 1704 1484
##    1    1    1    1    1    1    1    0    1    1    1    0    1    0    0
## 1968 1047  378 3063 2053 1268 2423 1374  727 2638 1814  750 2829  351  153
##    0    0    0    1    1    1    0    1    0    0    1    1    1    1    1
## 3114  419 2558 3050  338  252 2564 1786 1599 1793  883 1606  404 1769 1142
##    1    1    0    1    0    0    0    0    0    0    1    0    0    1    1
## 1266 1479 2364 1494   53 1349  181  968 1977  845  551 2847 1659 2603  455
##    0    1    1    1    1    1    1    1    0    0    0    1    1    0    1
## 2187 1106 1440 2741   75 2147  290  523 2456 2345 2411  490 1918 3055 1110
##    0    0    0    1    0    0    0    1    0    1    1    1    0    1    1
##  913 2252 2859 2000 2009 2018 1573 2675 1932 1520 2170 1919 1159 1937 1698
##    0    1    1    0    0    0    1    1    0    0    1    1    1    1    0
##  302  664 1350 2214   14  507 1813 1818  729 3034   71  879  283  703 2801
##    1    0    0    1    1    1    0    0    1    0    0    1    1    0    1
## 1639 1253 1411 2529  713 3016   47  188  992  260 1549  475 1061 1822  384
##    0    1    0    0    1    1    0    0    0    0    0    1    1    1    0
## 1884 1499 1259 1594  818  663 3083  745 1327 2892   60 2349 1434 1486 1239
##    1    1    1    0    1    0    1    0    0    1    1    0    1    1    1
##  711 2816  534 3010  101 3077 2611 1324 2817  361 1467 3015  544 2103  272
##    1    1    1    1    0    1    0    0    0    0    1    0    1    1    0
## 2944 1843 1296 2889 1290  241 1033  723  833 1371  102 1864  483 2640  554
##    0    1    1    1    1    0    1    0    0    1    1    0    0    0    0
## 1719 2339 1809 2211   67 1341 1666 1283  935 2943 1541 1837 1957 2383 1302
##    0    1    1    0    1    0    0    1    1    0    0    0    1    0    0
## 2582  569 1896 2559  772  185 1546 2607  495 1555 1534 1730 1832  689 1118
##    1    1    0    0    0    1    0    1    1    1    1    0    1    0    0
## 2196 2329  152  773 1559  692  358 2265  946 1321 2119 1733  819  961  698
##    0    1    0    0    1    0    1    1    0    0    0    1    1    0    1
## 2796 2022 1626 2618 1696  791  254  927  408 2458 1141 2654 2446  638  448
##    1    0    0    1    0    0    1    1    1    1    1    0    1    0    0
## 2677 2938 1508  924  402  802  112 2342  811  647 1384  537 2799 2950 2761
##    1    1    0    0    1    0    0    0    1    1    0    1    1    1    0
## 2418 3082 2895  631  438  959 1197 1340 2790  790  226  500 3019 1504 1382
##    1    0    0    0    1    1    0    0    0    0    1    0    1    0    0
##  146 1589 1807 2590 2594  244  164 1637  132  510 1870 2471 1889  369 1492
##    0    1    0    0    1    1    1    0    0    0    0    1    0    1    0
## 2476  123 1024 2249 3049 1279 2965 1617 2137 2307  332  325 1452  219 2157
##    1    1    1    0    0    0    1    1    0    1    0    1    0    0    0
```

```
##  315 1582 1050 1840 2739 1249  676 1428 1921 2839  324  976 1421 1515  387
##    1    0    0    1    1    1    1    1    1    0    1    1    0    0    1
##  744 1723 2284  342 2992 2220 1744 1107 2107 1620  933 1104  584 2920 1328
##    0    0    0    0    1    0    0    0    0    0    1    0    0    0    0
##  115 2504 1140  878 1715  300  581 1216  683  665  630 1574 2485 2911 1858
##    1    1    1    1    0    1    0    1    0    1    1    0    0    0    1
## 1036  354 1878 1363 1236 1105 1076 1827  823 2520 1777 2261  489  607 1021
##    1    0    1    0    1    0    1    0    0    1    1    0    0    1    1
## 2402 1939 3057  281  480 1703 1908  390  613 1953 2715 2403  601  964  322
##    0    1    1    0    1    1    0    1    1    1    0    1    1    0    0
##  606 2764 1867 1255   62 1338 2156  853 2616 3086 1570 1800 2001  810  172
##    1    1    1    0    0    1    1    0    0    0    1    1    1    1    1
## 1400  444  977 1667  504 2786 2378 2145 2436 2939 2399 2629 2857 2667  623
##    0    1    1    0    0    0    0    0    1    1    1    0    0    0    0
## 1911   54  677  370  989 2438 1971 2832 2837 3058 1462 2285 1553 2037 3013
##    1    0    1    0    1    0    0    0    1    0    1    1    1    1    1
## 1481 1892 2277  348 1844  720  979 1432 2915  653 1010 1713 1525 2495 2753
##    1    0    1    0    1    0    1    0    0    1    0    0    1    0    0
##  701  379  333 1443 2750 2344 1887 2665  151 2443 1108 2109 1008 2983  243
##    1    1    1    1    0    1    1    1    1    0    1    0    1    0    1
##  675 2353 2351 2005  952 1153 1357 1427  700  110  753  793  304 1885 1204
##    1    1    1    1    1    1    0    0    0    1    1    0    1    1    0
##  166  196 1319 1821  826  227 2441 2848 1171 2777 2687 2324 2663 1505 3073
##    0    0    1    0    1    1    1    1    0    1    1    0    1    1    1
## 2743 2474 2123 2867 2448  532  627  668 1624 2694  377 2624 1042  570 2875
##    0    0    0    1    1    0    1    0    0    1    1    0    1    0    1
## 2155 1789 2113 2662 2177  965 2175  843 1742 1708 2987 1059 1695 1914 1007
##    1    0    0    1    1    1    1    0    1    0    1    0    1    1    0
## 1062  839 2682 1160 2823 1671 1773 2756  706 2318 2592 1645 1301  919 1894
##    0    0    1    0    1    0    0    0    0    1    1    1    0    1    0
## 2797 1156  200 1177  752  572 1753  228  718 2916 1398 2406  208 1535   56
##    0    0    1    0    1    1    1    0    1    0    0    1    1    1    1
## 2328 1920 2181  832 1521  973 1206  781  349    5  130 1447 2503 2152   58
##    0    1    1    1    0    1    0    0    0    0    0    0    1    1    1
## 2725 1987 1673 1449 1006 2128 1390 1754  546   78 1735 2044 2709 1835 1388
##    1    0    0    0    1    1    1    1    1    0    0    0    1    0    1
## 3109 1609 1945  213 2191 2568  430 1548  210  889 2533 2674  771  233  869
##    0    1    1    1    0    0    1    1    0    0    1    1    1    0    1
## 1161 1979 2976  886  255  202  726  471 1959 1387 1662  511 1497  728 3100
##    0    1    0    1    0    1    0    0    0    0    0    0    1    1    0
## 1083 1902 2338 3023 3074 2574 2094 3104 2795 2838 1485   30  291  867 1072
##    0    1    1    1    0    0    0    0    1    1    1    0    0    0    1
## 2385   21 2631 2698  289  932 2826 2704 1129  262 2942 2398 2948 1672  203
##    0    1    0    1    1    0    0    1    1    0    0    1    0    0    0
##  636  340  571 2560 2416  655 1365  288 3107 2176 2264 1445 1990 2565  974
##    1    0    0    1    0    1    0    0    0    0    0    0    1    1    0
## 2116 1439  666   39    1   45  477  568 1426 1557 1273 1217 1669   46  465
##    0    1    0    1    0    1    0    1    1    1    0    0    1    0    0
## 1084 2648 2372 2450  574 3024 3084 3053 2633 2052   25 1912  320 1166 1456
##    1    0    0    1    1    0    0    0    1    0    1    0    1    0    0
```

```
## 2017 1780  789 2506 1393  298 2524  903 2278 1849   68 1804  857 2223 2038 
##    1    0    0    1    0    1    1    1    1    0    1    1    0    1    0 
##  380 1716 2439 1395  881 2517 2573 1941  413 2453  928  536 2477 1872 1963 
##    1    1    1    1    0    0    0    0    1    1    0    1    0    0    0 
##  269 1933 1323  488  950  494 1839 2133 1607  275 1078  467 2033  245 1740 
##    0    0    1    0    0    0    1    1    1    0    1    1    1    0    0 
##  618 1799 1762 1294 1470  104 3027 3002 2138 2130  940  316 2289 1507  174 
##    1    0    1    1    0    0    0    0    0    0    0    1    1    1    1 
## 2701 2169 1103 1873  996  398 1687 2879  926 1802 3076 2771 2894 2077 2082 
##    1    1    0    1    1    1    0    0    0    1    1    1    1    0    1 
## 1170 1995 1904 2617 1729 1471 1768 2232 2488  888 1816  779 2217 1346 1636 
##    0    0    0    1    0    1    0    0    0    1    1    1    0    0    0 
## 1944 1625 2419 2932  620 2043 2429  239  616 1825 2572 3097 3003 2626  731 
##    1    1    0    0    0    1    1    1    0    1    0    1    1    1    0 
##  450 2685  931 2193 1537 2481 1057  835  768 2047 1337 2846 2389  264 3035 
##    1    1    1    1    1    1    1    1    0    0    1    0    1    0    1 
## 2995 1899  984 1923 1203  612  292 2587 3090  336 2768  365 1130 2484  135 
##    1    0    0    0    0    1    1    1    1    1    1    0    1    0    1 
##  128 2400 1112 2891 2238 2131 2536 1940 2347 2106  963  287 2198 1242  792 
##    0    1    1    0    0    1    1    0    0    1    1    0    0    0    0 
##  403 2251 2733  693 1092  405  240   97 2708 2306 1498 1417  587  501  106 
##    1    1    0    0    1    1    0    0    1    1    1    0    0    0    0 
## 1005 1763 1396 2168 1276 1752  695 1096  816 1984  506  533 2414 1721 1694 
##    0    0    0    0    0    0    0    0    0    0    1    1    1    0    0 
## 2822 1241 2263 2576 2998 1144 2464 2865 1195  457 1978 2642 1245 2213  466 
##    1    1    0    1    0    0    0    1    1    1    1    0    1    1    0 
## 2922  392 1476  986 2199  573  248 2872 3065 2785 2539  278  939 1207 2834 
##    0    0    0    0    1    0    0    0    1    0    1    1    1    0    1 
## 2192 2174 1121 2623  566 2904 2650 1138 3080 2946 2843 1514  167 2921 2552 
##    0    0    0    0    0    1    1    0    1    0    0    0    0    0    0 
## 1009 1707  493 1213 2424 1550 1442  870 1071 1906 2487  717 1068  327 2542 
##    1    0    0    1    1    0    1    0    1    0    0    0    0    1    1 
## 2042 2158 2550 1093  218 2190 1882 1936 1869  880 1834 1126   90 2908 2380 
##    0    1    0    0    1    1    0    0    1    0    1    0    0    1    1 
## 1038 1647 3006 1423  331 1970  347 2340 2239 1412 1308 2609  610 2969  389 
##    0    0    1    0    1    0    1    0    1    1    0    0    0    1    0 
##  936 1164 2595 1852 2087  462 1611 2480 1924 1148  170  830 1228 1409 1722 
##    1    0    1    0    0    0    0    1    1    0    1    0    1    0    0 
## 1043  865  953 2141 1002 2268  452  776 1077 3089 1871  921 2032  564 2782 
##    1    1    0    1    1    1    1    0    1    0    1    1    1    1    1 
## 2727 1850 2828   43 2132 1116 2208  491    8 1180 1044 1861  364 1465 2021 
##    0    1    1    0    0    1    0    1    0    1    0    0    0    0    1 
## 1020  133  983  154 2083 2763  719   84 1686 2752 2078  820 2811 2396  518 
##    0    1    0    0    1    0    0    1    0    1    1    1    0    1    0 
## 1247  990 2897  111   85  541 1453    2 2748 1065 1785  652 2110  941 1512 
##    0    1    1    1    1    0    0    0    0    1    0    0    1    1    0 
## 2914   93 1847 1353 2467 2073 1988 2519 2902  967 1031 1591  949   22 1682 
##    1    1    0    0    0    0    0    1    1    0    1    1    0    1    1 
## 2604  176 2724 1934 2888 2072 2681 3026 1783   16 1684  223 1289 1992 2793 
##    0    0    1    1    1    1    1    1    0    0    1    1    0    0    1 
```

```
## 2971  498 2947 2930 2255 1509  515 2246 1066 2516 1842 2215  595 2737 2964
##    0    0    1    1    1    1    1    1    0    0    1    1    1    1    1
##   76  775  279 1095 1297 2301 2634 1003 1351 2097 2585 2580   26 1100 2430
##    0    0    0    1    1    0    0    1    1    1    0    0    1    0    0
## 1615  925 2546  767 1478 2281 1125 3110  741 2713 2868 2412  113 1910  749
##    1    0    0    1    1    1    0    1    0    0    0    1    0    0    1
## 1450 2362 1058 2802 1235 2540  858 2613 1828  225 2049 1810  207 2882 1572
##    1    1    1    1    1    1    1    0    0    1    0    0    0    0    1
## 1657 1653 2996 2425 1178  586 2691 2566    3 2561 3021 1277 1312 2105 1913
##    0    1    1    0    1    0    0    0    1    1    0    1    0    1    1
##  189  590 1623 1938  127  443  561  456  412 1833 1184 1841  307 1622 1927
##    0    0    1    1    1    1    0    0    0    0    0    0    1    0    0
##  436  836 1676  628  423  838   23  521  599 1650  893 2200 2062 1527 2924
##    1    0    1    1    1    1    0    0    0    1    1    1    0    0    0
## 1454 2312 2844  914 2478  842 1790 1965 2664 1530  372 2010  266 2060 2711
##    0    1    0    1    1    1    0    1    1    1    1    0    0    1    0
## 2505  459  740 3094 1652  743  579  640  614  565 1845 2472 2100 2936  669
##    1    0    1    0    0    0    0    1    1    0    1    0    1    0    1
## 1556   18 2728  834 1023 2004  852  874 2717 1629 2240  366  156 3070   74
##    0    1    1    0    0    1    0    0    0    0    1    1    0    1    1
## 1332 2527 1817 1225 1230 1656 1788 2686 1335 1731 2716 2928  205 2863  589
##    0    0    1    1    0    1    1    1    1    1    0    0    0    1    1
## 1985  580 1634 3120 1760  464 1524 1994 3037  951 1287 2919 1416  736 1831
##    0    1    0    0    0    0    0    0    1    1    0    1    0    0    0
## 1052  756 2162 1568  754  382  116 1271 1394 1569 2150 2760  621  125 2019
##    1    1    1    1    1    1    0    0    0    1    0    1    1    0    1
##  851 2006 2898  305 2776  828 2115 1755 1135  764   42 3005 3040   37  575
##    0    1    0    1    1    1    0    1    0    1    0    1    1    0    1
## 2807  162 2747 2804 1139 1211 2982 2046 1436 2775 1172 2386 1345  909 1014
##    0    0    0    1    0    1    0    0    0    0    1    0    1    1    1
##  646  999  453  427 2056 1725 1430  173  585 2684 1299 2055
##    0    1    1    0    0    0    0    1    1    0    0    0
## Levels: 0 1
```

```r
# Making the Confusion Matrix for training data
cm_train = table(train.df$TARGET_B, y_pred_train)
cm_train
```

```
##    y_pred_train
##       0   1
##   0 512 413
##   1 407 540
```

The accuracy of the model obtained from the validation data is 54.73%.

```r
#Profit from the training data
Profit_valid<-(572*((13-0.68)/9.8))
Profit_valid
```

```
## [1] 719.0857
```

```
#cost from the training data
cost_valid<-(373*((0-0.68)/0.53))
cost_valid

## [1] -478.566
```

For the second model, net profit for validation data = $455.0857 - $324.6038 = $130.4819

```
pred_test_svm <- predict(classifier, newdata = test.df, type = "response")
fund.pred_svm <- pred_test[order(-pred_test)]
head(fund.pred_svm)

##       854      1460      1755         9       120      1610
## 0.7733655 0.7711937 0.7567172 0.7482622 0.7448589 0.7374373
```

From the results obtained, we can conclude that the SVM model is superior than the logistic regression model. Since SVM provides class predictions, we can use the probabilities obtained from the logistic regression model to predict the donors and non-donors. We will use a cut-off value of 0.6 for predicting the donors. Therefore, all the entries having a probability greater than 0.6 will be considered for the mailing campaign.