R Notebook: Advanced methods

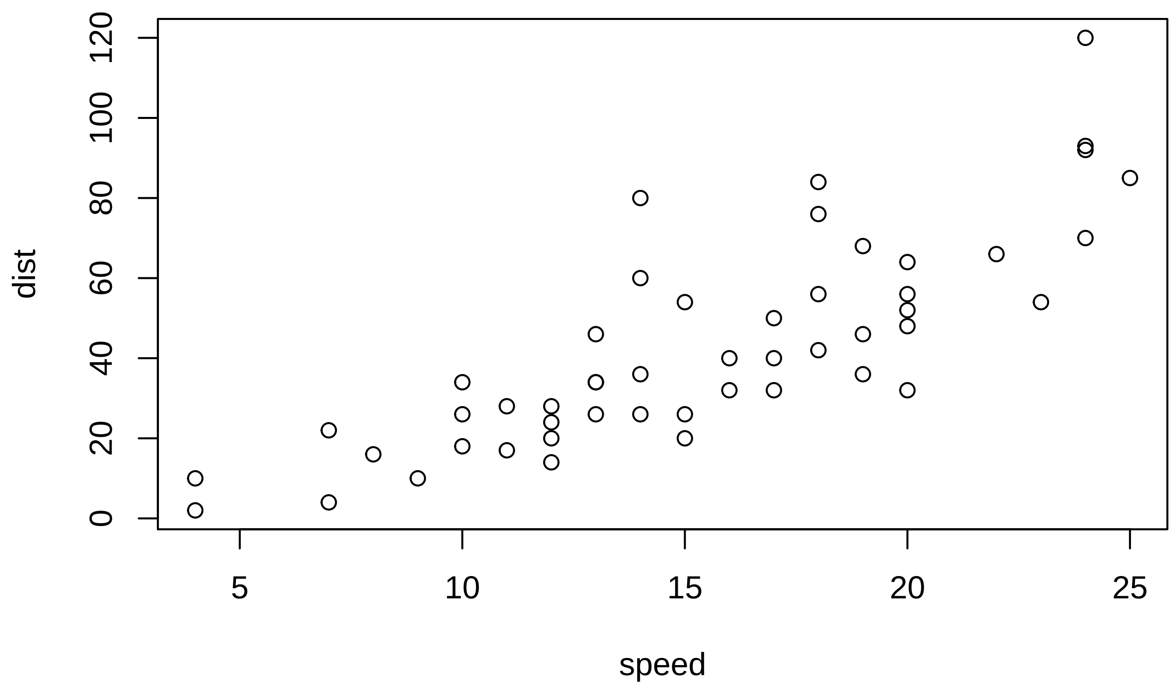
Montserrat Guillen

2017

This is an R Markdown Notebook. When you execute code within the notebook, the results appear beneath the code.

Try executing this chunk by clicking the *Run* button within the chunk or by placing your cursor inside it and pressing *Ctrl+Shift+Enter*.

```
plot(cars)
```



Add a new chunk by clicking the *Insert Chunk* button on the toolbar or by pressing *Ctrl+Alt+I*.

When you save the notebook, an HTML file containing the code and output will be saved alongside it (click the *Preview* button or press *Ctrl+Shift+K* to preview the HTML file).

The preview shows you a rendered HTML copy of the contents of the editor. Consequently, unlike *Knit*, *Preview* does not run any R code chunks. Instead, the output of the chunk when it was last run in the editor is displayed.

Introduction

In this document we do a more advanced Data Analysis linked to the article by S. Moro, P. Cortez and P. Rita. A Data-Driven Approach to Predict the Success of Bank Telemarketing. Decision Support Systems, Elsevier, 62:22-31, June 2014. The two datasets ontain similar information, but not exactly the same.. Here we will analyse the smaller data set (called **bank.csv**). The file can be downloaded from: https://archive.ics.uci.edu/ml/datasets/bank+marketing

or (for this course)

http://www.ub.edu/rfa/docs/DATA/bank.csv

We will see logistic regression, decision tree, random forest and svm. We could also try Bayesian networks and neural networks.

Reading the data

```
getwd()
```

```
## [1] "C:/Users/UBrisk/Desktop/POSTGRAU-DATA-SCIENCE/CARPETA_APUNTS/TO-GIVE/FINAL-DAY-2"
```

```
bank<-read.csv("bank.csv",header=T,sep=";", dec=".")
```

Training and test data sets

We try to build a model with dividing training data and test data set. We divide 75% of whole bank data set as training data, and rest 25% of bank data set.

```
set.seed(123456789)
random<-sample(1:nrow(bank))
num.bank.training<-as.integer(0.75*length(random))
bank.indices<-random[1:num.bank.training]
train<-bank[bank.indices,]
testing.indices<-random[(num.bank.training+1):length(random)]
testing.set<-bank[testing.indices,]
```

Logistic regression

Full model

We estimate the full model

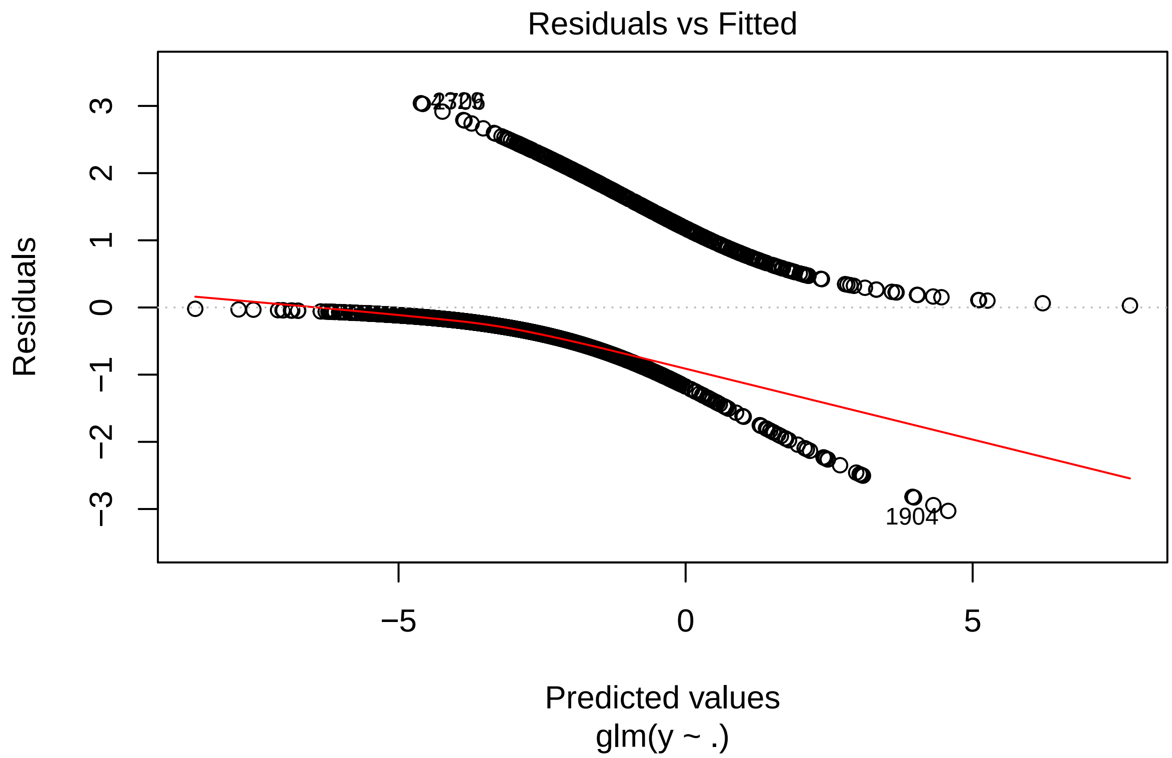
```
logis<-glm(y ~ ., data=train,family=binomial)
summary(logis)
```

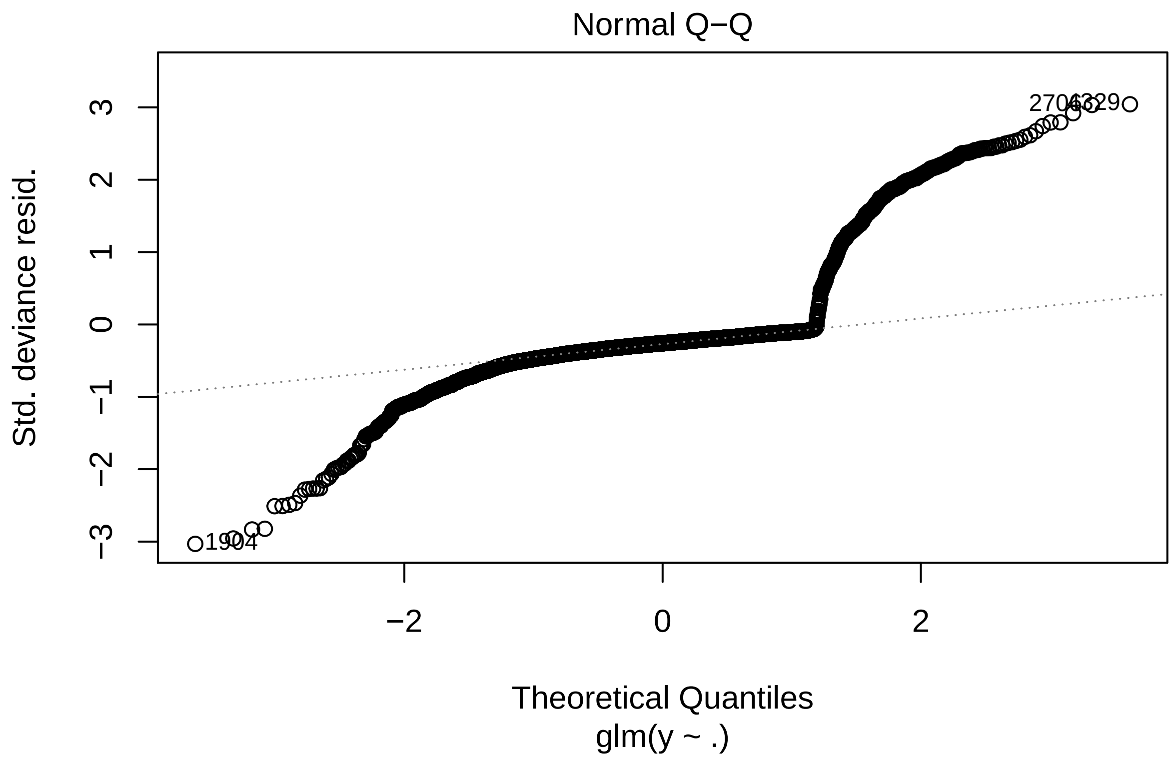
```
##
## Call:
## glm(formula = y ~ ., family = binomial, data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.0280  -0.3874  -0.2585  -0.1516   3.0401
```

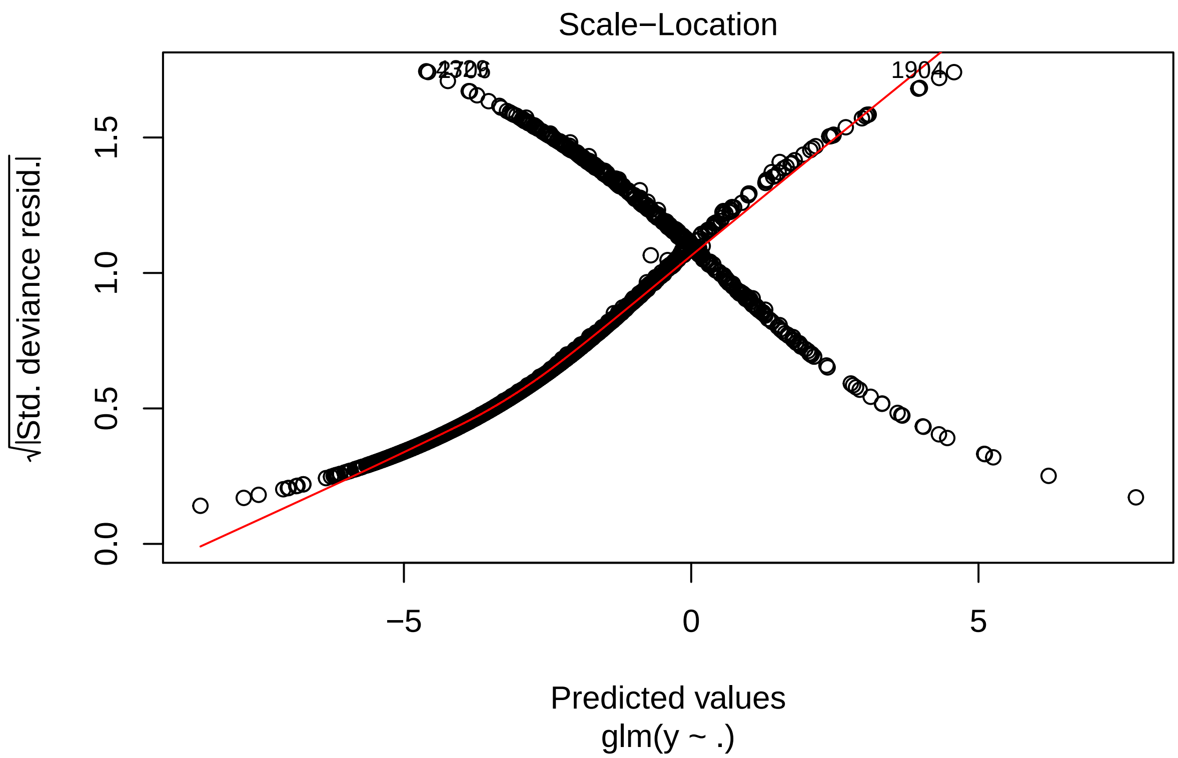
```
##
## Coefficients:
##                      Estimate Std. Error z value Pr(>|z|)
## (Intercept)        -2.720e+00  6.987e-01  -3.894 9.88e-05 ***
## age                 1.410e-03  8.070e-03   0.175   0.8613
## jobblue-collar     -4.045e-01  2.767e-01  -1.462   0.1438
## jobentrepreneur    -3.145e-01  4.519e-01  -0.696   0.4864
## jobhousemaid       -5.562e-01  4.858e-01  -1.145   0.2523
## jobmanagement      -2.128e-01  2.835e-01  -0.751   0.4528
## jobretired          3.727e-01  3.629e-01   1.027   0.3045
## jobself-employed   -1.625e-01  3.944e-01  -0.412   0.6804
## jobservices        -1.188e-01  3.086e-01  -0.385   0.7001
## jobstudent         -1.765e-02  4.659e-01  -0.038   0.9698
## jobtechnician      -2.234e-01  2.682e-01  -0.833   0.4050
## jobunemployed      -8.416e-01  5.020e-01  -1.677   0.0936 .
## jobunknown          1.838e-01  6.533e-01   0.281   0.7785
## maritalmarried     -4.818e-01  1.986e-01  -2.425   0.0153 *
## maritalsingle      -2.682e-01  2.337e-01  -1.148   0.2510
## educationsecondary  1.036e-02  2.299e-01   0.045   0.9640
## educationtertiary   3.023e-01  2.673e-01   1.131   0.2581
## educationunknown   -4.279e-01  3.990e-01  -1.072   0.2835
## defaultyes          7.211e-01  4.802e-01   1.502   0.1331
## balance            -2.523e-06  1.863e-05  -0.135   0.8923
## housingyes         -2.668e-01  1.587e-01  -1.681   0.0928 .
## loanyes            -4.262e-01  2.158e-01  -1.975   0.0483 *
## contacttelephone    1.618e-02  2.653e-01   0.061   0.9514
## contactunknown     -1.574e+00  2.702e-01  -5.827 5.66e-09 ***
## day                 1.494e-02  9.519e-03   1.570   0.1165
## monthaug           -6.274e-02  2.975e-01  -0.211   0.8330
## monthdec            3.881e-02  8.088e-01   0.048   0.9617
## monthfeb            5.898e-01  3.412e-01   1.728   0.0839 .
## monthjan           -5.631e-01  4.249e-01  -1.325   0.1851
## monthjul           -5.619e-01  2.930e-01  -1.918   0.0551 .
## monthjun            8.392e-01  3.536e-01   2.374   0.0176 *
## monthmar            2.083e+00  4.640e-01   4.489 7.16e-06 ***
## monthmay           -3.150e-01  2.744e-01  -1.148   0.2509
## monthnov           -5.958e-01  3.165e-01  -1.883   0.0597 .
## monthoct            1.479e+00  3.742e-01   3.952 7.74e-05 ***
## monthsep            8.415e-01  4.722e-01   1.782   0.0747 .
## duration            4.296e-03  2.360e-04  18.202  < 2e-16 ***
## campaign           -8.679e-02  3.409e-02  -2.546   0.0109 *
## pdays              -1.185e-04  1.186e-03  -0.100   0.9204
## previous           -2.134e-02  4.223e-02  -0.505   0.6133
## poutcomeother       5.990e-01  3.037e-01   1.972   0.0486 *
## poutcomesuccess     2.434e+00  3.224e-01   7.549 4.37e-14 ***
## poutcomeunknown    -1.422e-01  3.674e-01  -0.387   0.6987
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2440.3  on 3389  degrees of freedom
## Residual deviance: 1633.2  on 3347  degrees of freedom
## AIC: 1719.2
```

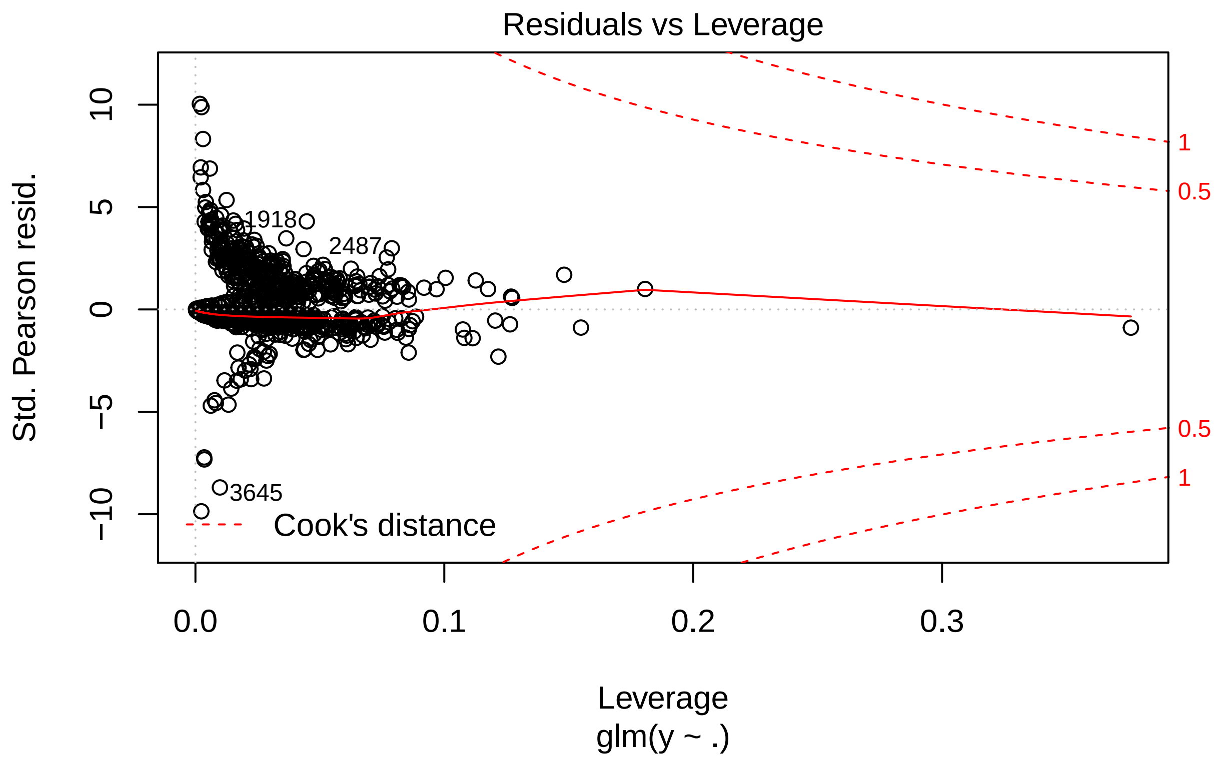
```
##
## Number of Fisher Scoring iterations: 6
```

```r
plot(logis)
```

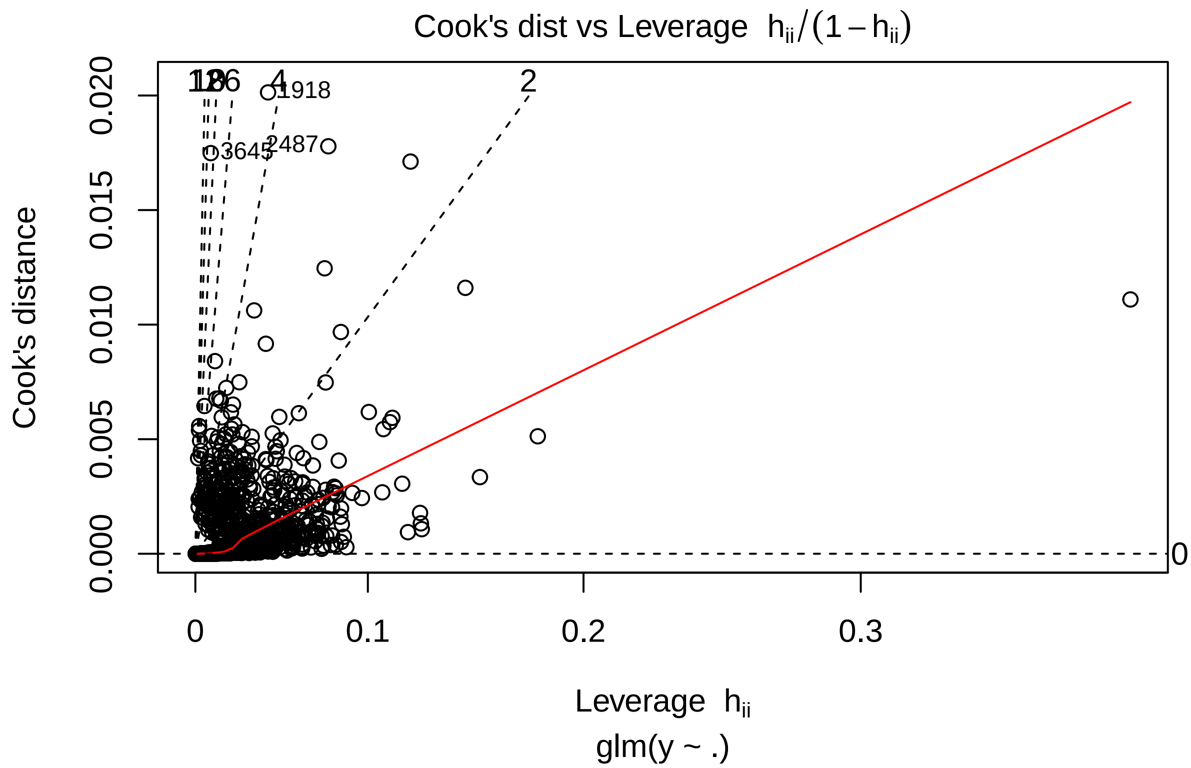



Normal Q−Q
Std. deviance resid.
Theoretical Quantiles
glm(y ~ .)
−3
−2
−1
0
1
2
3
−2
0
2
1904
2704
329


Scale−Location
√|Std. deviance resid.|
0.0
0.5
1.0
1.5
−5
0
5
Predicted values
glm(y ~ .)
1904



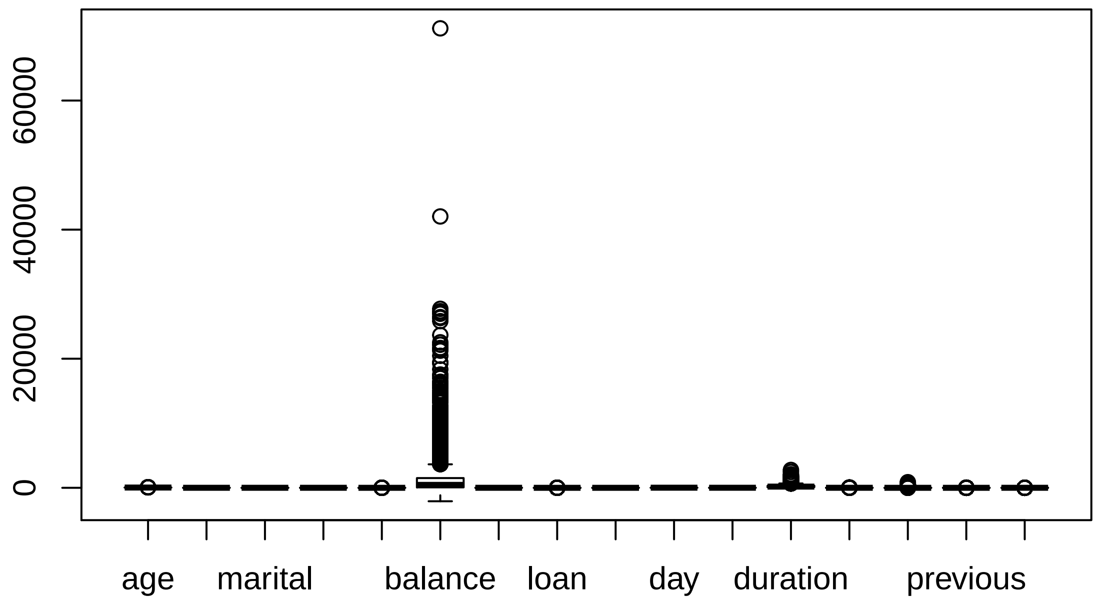

```
plot(logis,which=6)
```



Boxplots

Boxplot for dependent variable -> however, since y is categorical variable, no need to transformed

```
boxplot(train[,1:(ncol(bank)-1)])
```




```
bank1<-train
bank1$dffits<-0
bank1$dffits<-dffits(logis)
bank2<-bank1[!bank1$dffits>2*sqrt(ncol(bank)/nrow(bank)),]

bank1$dfitts<-NULL
bank2$dfitts<-NULL
bank1$out<-NULL
bank2$out<-NULL
```

Outliers

We remove the observations 1676, 52, 1904 to remove influential outliers

```
logit<-glm(y~.,data=bank2,family=binomial)
```

```
## Warning: glm.fit: algorithm did not converge
```

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

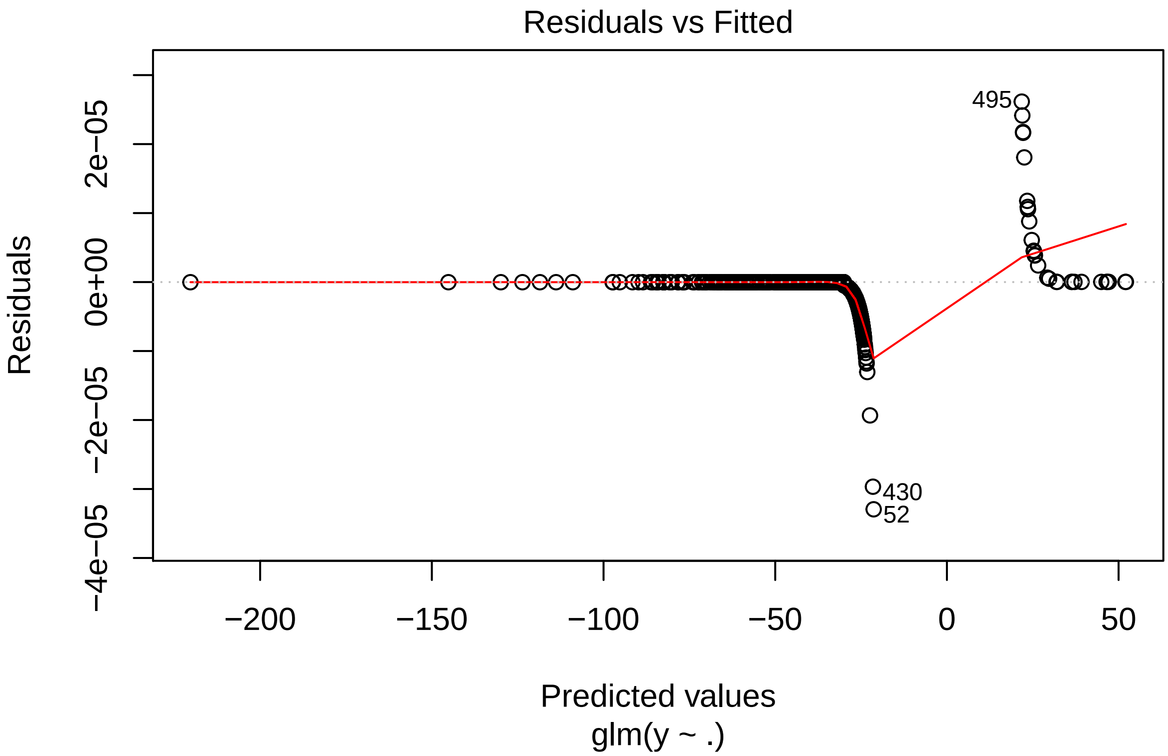
```
summary(logit)
```

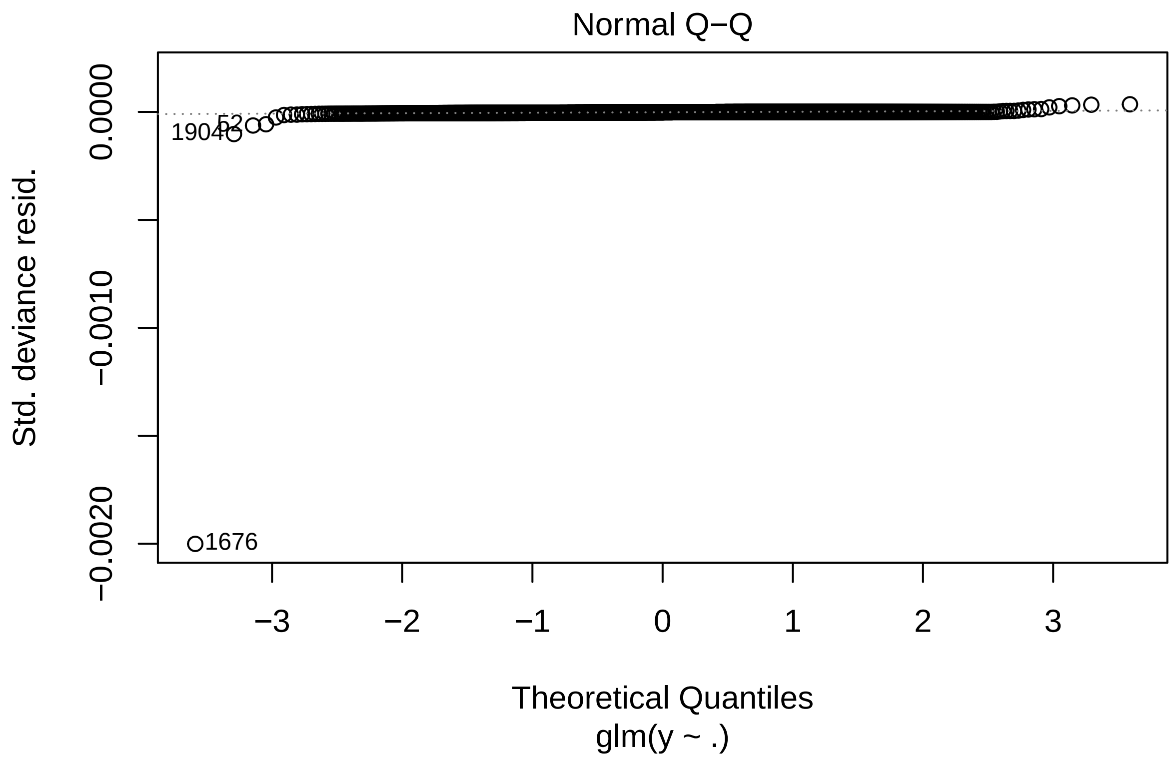
```
##
## Call:
## glm(formula = y ~ ., family = binomial, data = bank2)
##
## Deviance Residuals:
```

```
##        Min          1Q      Median          3Q         Max
## -3.294e-05  -2.912e-06  -1.587e-06  -2.100e-08   2.616e-05
##
## Coefficients:
##                      Estimate Std. Error z value Pr(>|z|)
## (Intercept)        -1.642e+01  7.434e+04   0.000    1.000
## age                 1.331e-02  7.485e+02   0.000    1.000
## jobblue-collar     -2.353e+00  2.405e+04   0.000    1.000
## jobentrepreneur    -1.794e-01  4.049e+04   0.000    1.000
## jobhousemaid       -1.621e+00  4.829e+04   0.000    1.000
## jobmanagement      -1.859e+00  3.062e+04   0.000    1.000
## jobretired          1.867e+00  4.472e+04   0.000    1.000
## jobself-employed    1.569e-01  3.801e+04   0.000    1.000
## jobservices        -5.034e-01  2.839e+04   0.000    1.000
## jobstudent          2.221e+00  5.358e+04   0.000    1.000
## jobtechnician      -1.735e+00  2.555e+04   0.000    1.000
## jobunemployed      -2.623e+00  4.376e+04   0.000    1.000
## jobunknown          4.002e+00  7.805e+04   0.000    1.000
## maritalmarried     -3.582e+00  1.738e+04   0.000    1.000
## maritalsingle      -1.923e+00  1.879e+04   0.000    1.000
## educationsecondary -5.034e-01  2.107e+04   0.000    1.000
## educationtertiary   1.178e+00  2.844e+04   0.000    1.000
## educationunknown   -1.185e+00  3.995e+04   0.000    1.000
## defaultyes          6.876e+00  5.329e+04   0.000    1.000
## balance             9.283e-05  2.193e+00   0.000    1.000
## housingyes         -1.988e+00  1.529e+04   0.000    1.000
## loanyes            -1.573e+00  1.798e+04   0.000    1.000
## contacttelephone    8.474e-01  2.774e+04   0.000    1.000
## contactunknown     -7.964e+00  2.131e+04   0.000    1.000
## day                 8.232e-02  8.884e+02   0.000    1.000
## monthaug           -1.692e+00  3.074e+04   0.000    1.000
## monthdec            8.399e+00  1.126e+05   0.000    1.000
## monthfeb            3.842e+00  4.012e+04   0.000    1.000
## monthjan           -2.119e+00  4.156e+04   0.000    1.000
## monthjul           -3.703e+00  3.038e+04   0.000    1.000
## monthjun            4.099e+00  3.462e+04   0.000    1.000
## monthmar            1.721e+01  8.687e+04   0.000    1.000
## monthmay           -1.961e+00  2.827e+04   0.000    1.000
## monthnov           -4.525e+00  3.478e+04   0.000    1.000
## monthoct           -2.363e+00  6.684e+04   0.000    1.000
## monthsep            7.205e+00  8.318e+04   0.000    1.000
## duration            2.554e-02  1.345e+01   0.002    0.998
## campaign           -2.476e-01  2.417e+03   0.000    1.000
## pdays              -5.704e-03  1.766e+02   0.000    1.000
## previous            5.463e-02  6.510e+03   0.000    1.000
## poutcomeother       3.973e+00  3.669e+04   0.000    1.000
## poutcomesuccess     2.698e+01  8.144e+04   0.000    1.000
## poutcomeunknown    -2.558e+00  5.119e+04   0.000    1.000
## dffits              1.667e+02  1.011e+05   0.002    0.999
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 3.2725e+02  on 3023  degrees of freedom
## Residual deviance: 2.4917e-08  on 2980  degrees of freedom
```

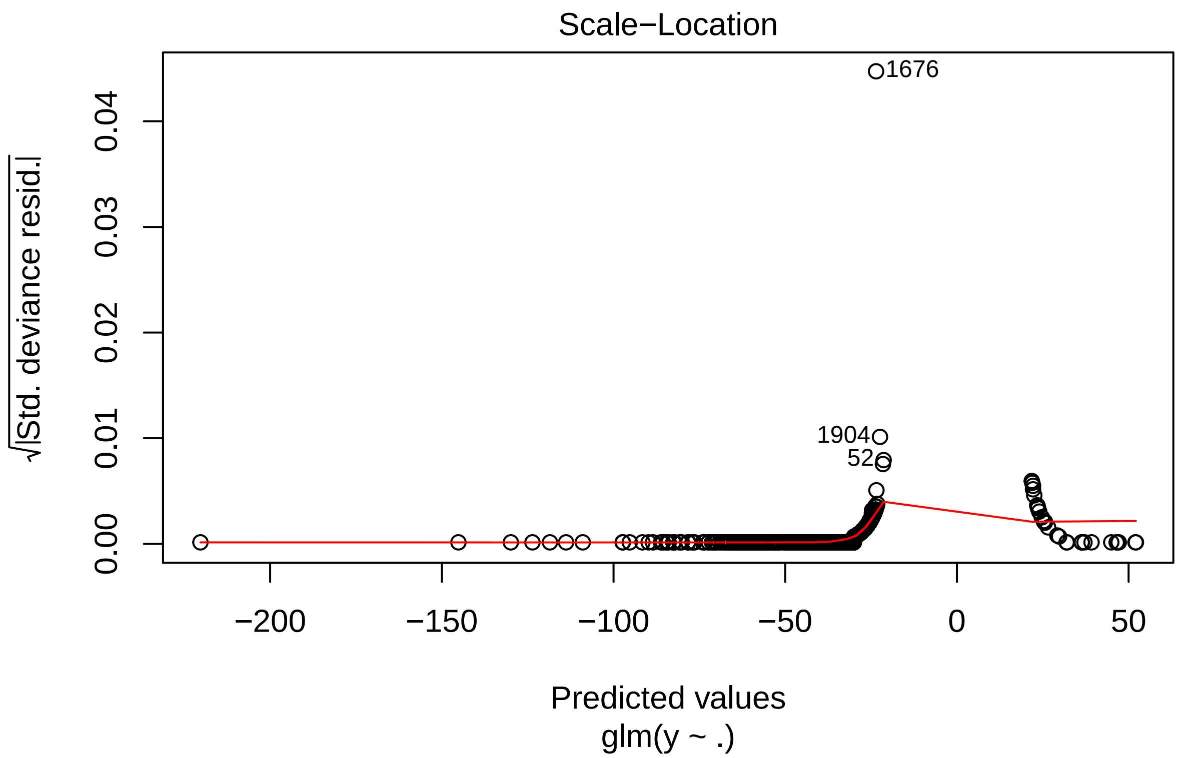
```
## AIC: 88
##
## Number of Fisher Scoring iterations: 25
```

```r
plot(logit)
```

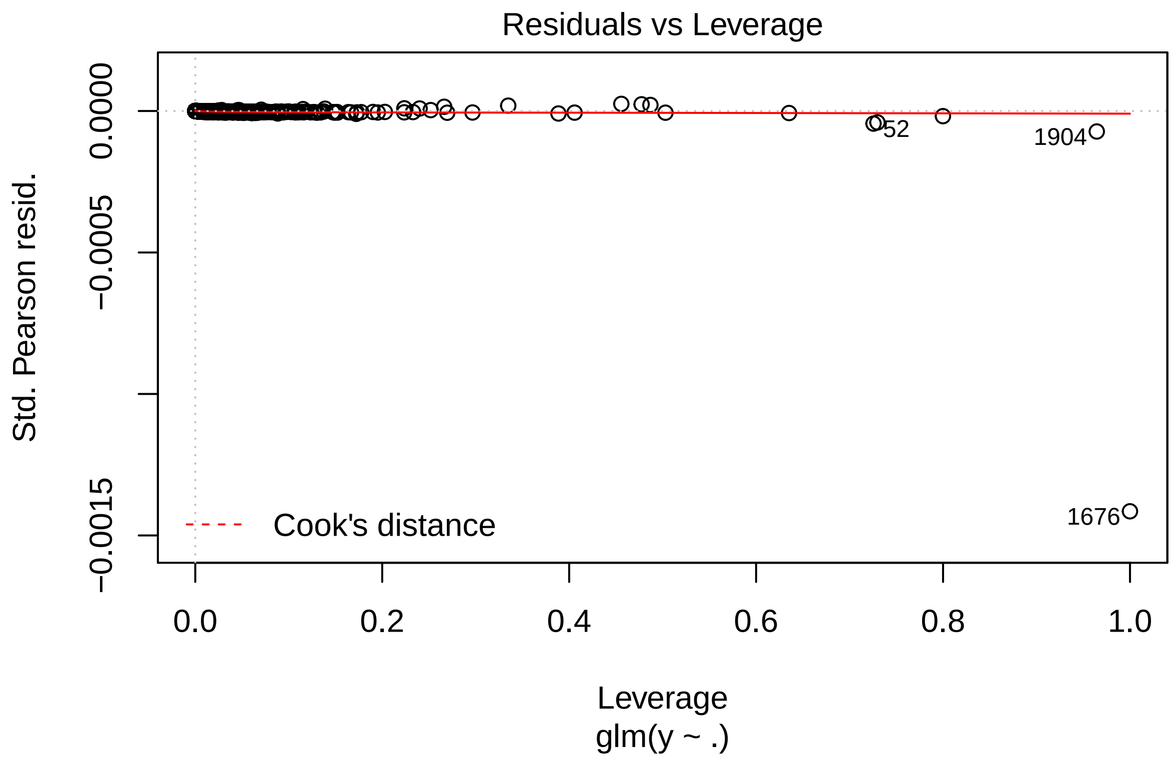




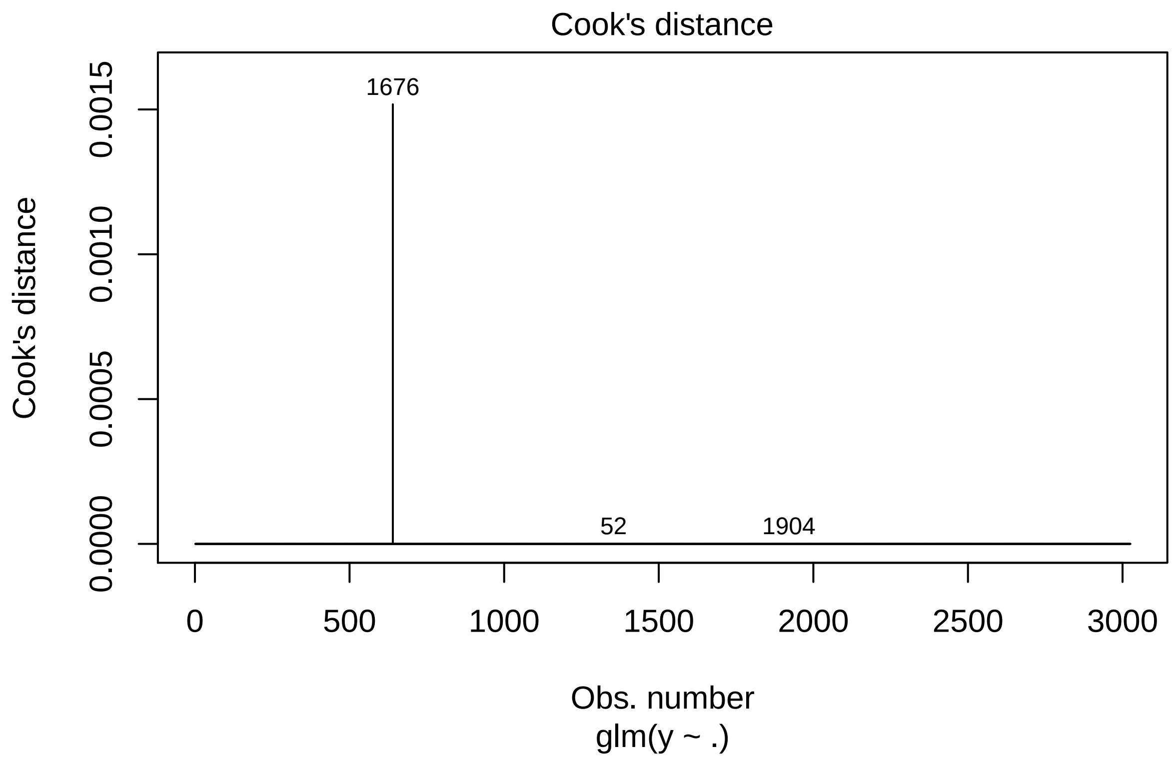
Normal Q−Q
Std. deviance resid.
0.0000
−0.0010
−0.0020
−3
−2
−1
0
1
2
3
Theoretical Quantiles
glm(y ~ .)
1904
52
1676




```
## Warning in sqrt(crit * p * (1 - hh)/hh): Se han producido NaNs
## Warning in sqrt(crit * p * (1 - hh)/hh): Se han producido NaNs
```




```
plot(logit,which=4)
library("car")
```




```r
outlierTest(logit)
```

```
##
## No Studentized residuals with Bonferonni p < 0.05
## Largest |rstudent|:
##          rstudent unadjusted p-value Bonferonni p
## 1676 -0.001414704            0.99887           NA
```

We run a new model

```r
bank3<-bank2[-c(1676, 52, 1904),]

logit.2<-glm(y~factor(contact) + factor(month) + duration +  balance + previous + factor(loan)+ factor(d
```

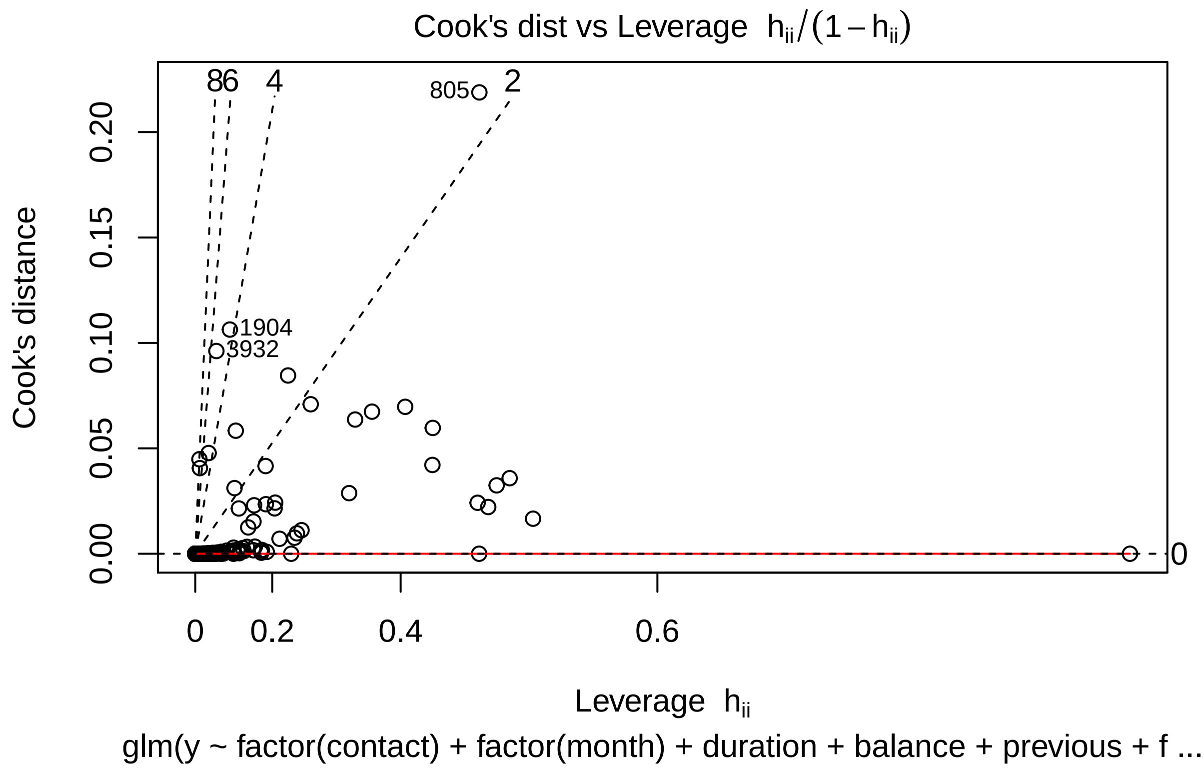
```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```r
summary(logit.2)
```

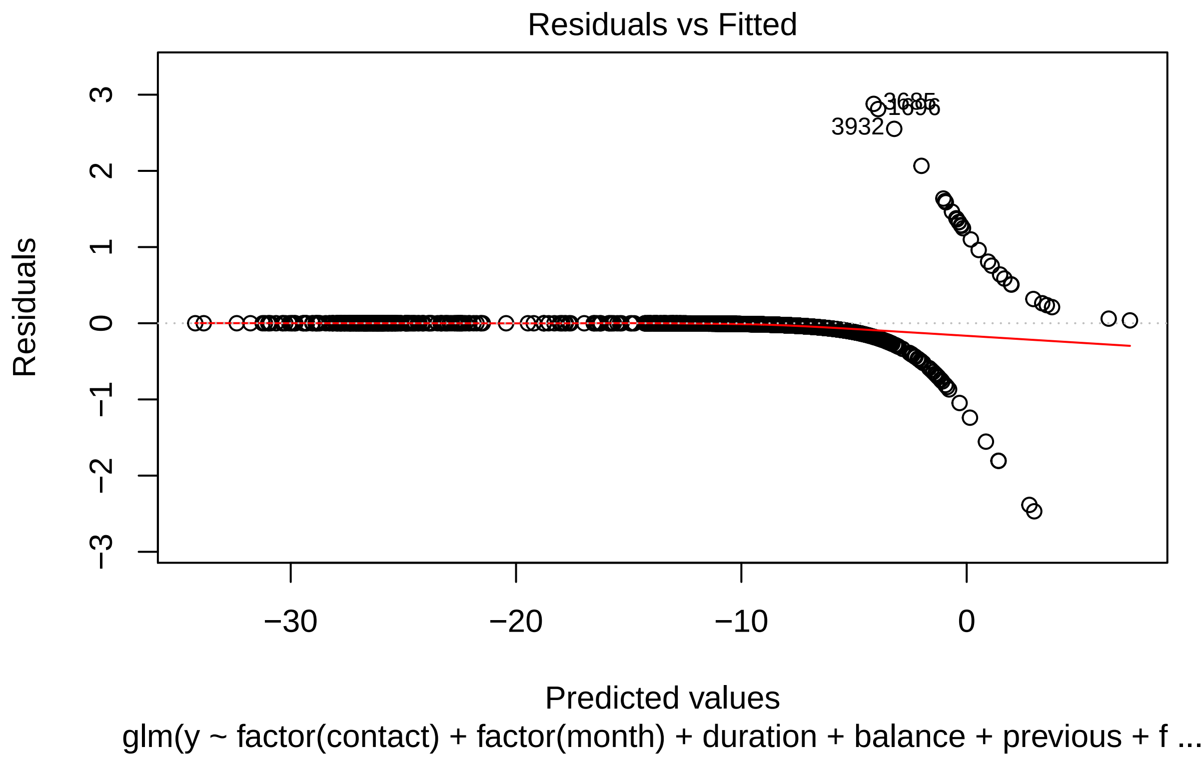
```
##
## Call:
## glm(formula = y ~ factor(contact) + factor(month) + duration +
##     balance + previous + factor(loan) + factor(default), family = binomial,
##     data = bank3)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
```

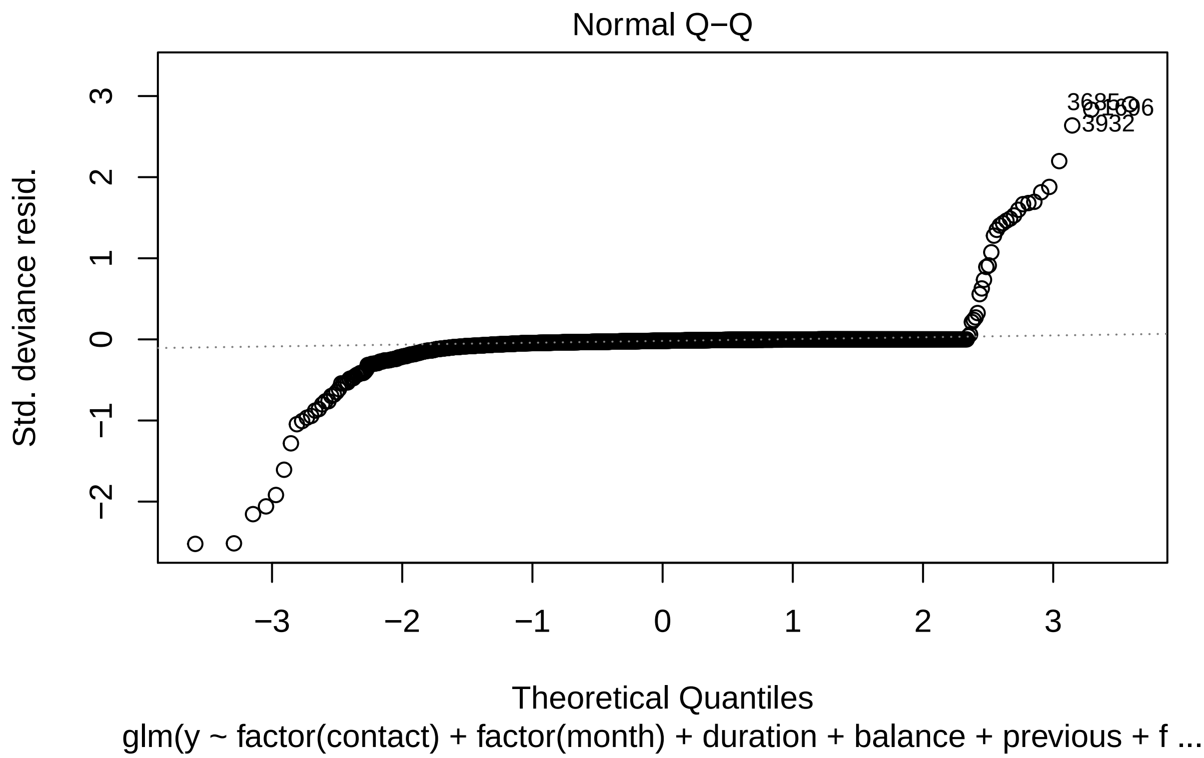
```
## -2.46820  -0.03438  -0.01654  -0.00399   2.87913
##
## Coefficients:
##                             Estimate Std. Error z value Pr(>|z|)
## (Intercept)               -8.392e+00  1.223e+00  -6.864 6.67e-12 ***
## factor(contact)telephone  9.074e-01  8.864e-01   1.024   0.3060
## factor(contact)unknown   -2.441e+00  1.377e+00  -1.773   0.0763 .
## factor(month)aug          3.483e-01  1.160e+00   0.300   0.7640
## factor(month)dec          6.041e-01  1.871e+00   0.323   0.7468
## factor(month)feb         -6.285e-01  1.942e+00  -0.324   0.7462
## factor(month)jan         -1.707e-03  1.571e+00  -0.001   0.9991
## factor(month)jul         -1.195e-01  1.295e+00  -0.092   0.9265
## factor(month)jun          1.480e+00  1.389e+00   1.066   0.2866
## factor(month)mar          5.576e+00  1.276e+00   4.369 1.25e-05 ***
## factor(month)may         -2.183e+00  1.444e+00  -1.511   0.1307
## factor(month)nov         -1.868e+01  1.321e+03  -0.014   0.9887
## factor(month)oct          2.386e+00  1.451e+00   1.644   0.1002
## factor(month)sep         -1.482e+01  5.001e+03  -0.003   0.9976
## duration                  6.393e-03  7.930e-04   8.062 7.50e-16 ***
## balance                  -3.603e-04  2.631e-04  -1.370   0.1708
## previous                  1.635e-01  9.338e-02   1.751   0.0800 .
## factor(loan)yes          -1.220e+00  1.377e+00  -0.886   0.3754
## factor(default)yes        1.825e+00  2.191e+00   0.833   0.4049
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 327.191  on 3020  degrees of freedom
## Residual deviance:  93.539  on 3002  degrees of freedom
## AIC: 131.54
##
## Number of Fisher Scoring iterations: 20
```

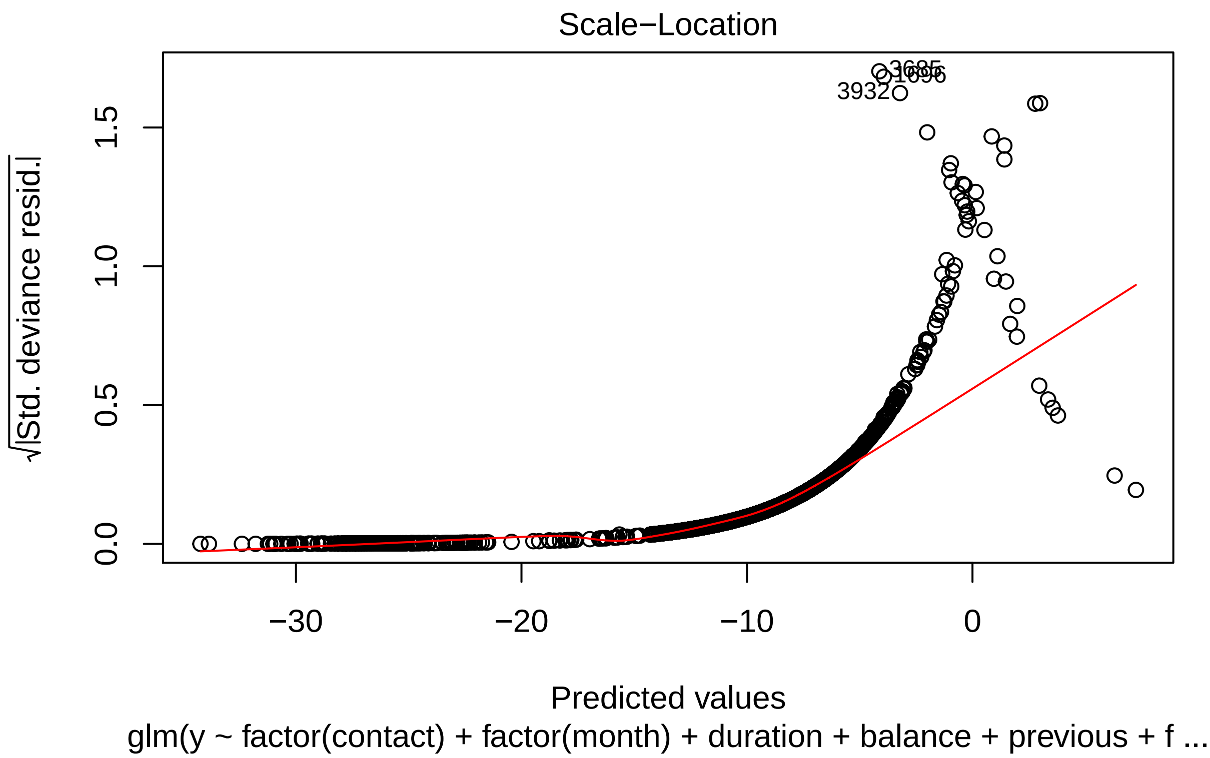
```
plot(logit.2,which=6)
```

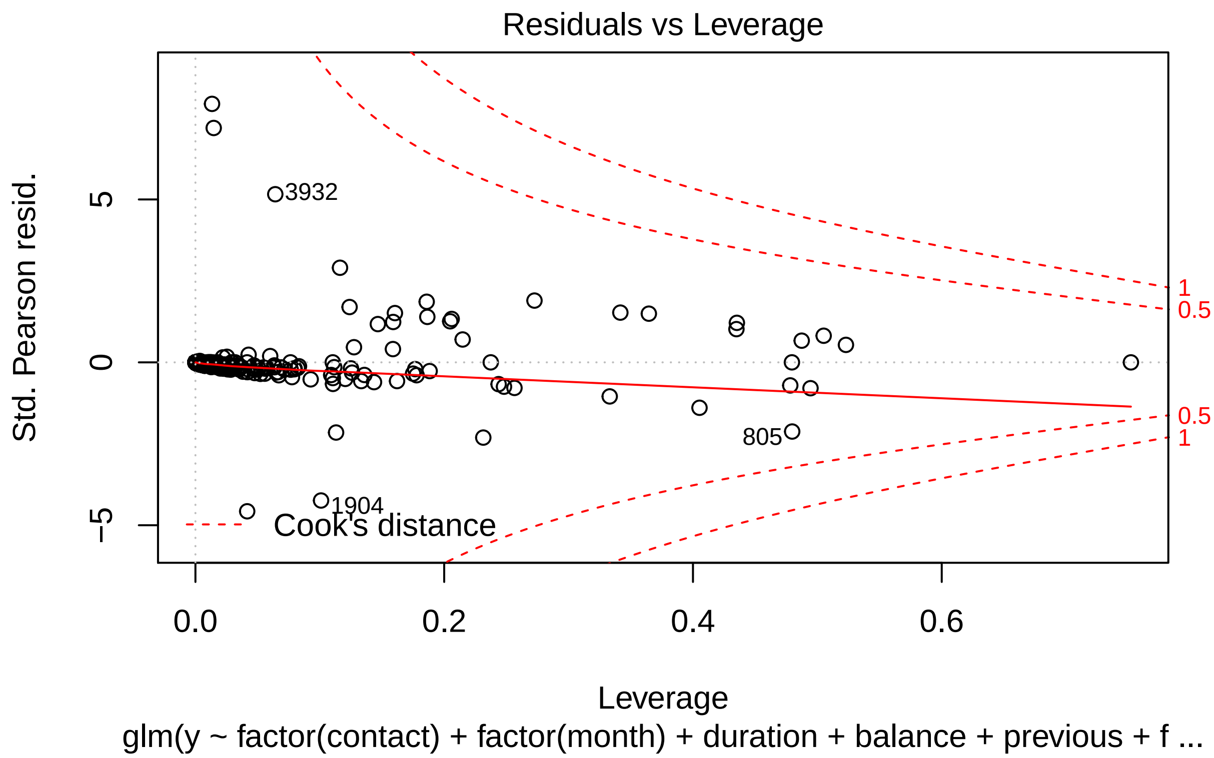



```r
plot(logit.2)
```


Residuals vs Fitted
Residuals
3
2
1
0
−1
−2
−3
3685
1696
3932
−30
−20
−10
0
Predicted values
glm(y ~ factor(contact) + factor(month) + duration + balance + previous + f ...


Normal Q−Q
Std. deviance resid.
Theoretical Quantiles
glm(y ~ factor(contact) + factor(month) + duration + balance + previous + f ...
3685
3696
3932
−3
−2
−1
0
1
2
3


Scale−Location
3685
1696
3932
√|Std. deviance resid.|
0.0
0.5
1.0
1.5
−30
−20
−10
0
Predicted values
glm(y ~ factor(contact) + factor(month) + duration + balance + previous + f ...



Check multicollinearity and variable selection

```r
table(bank3$y)
```

```
##
##   no  yes
## 2992   29
```

```r
library("stats")
library("MASS")
stepAIC(logit.2,k=2)
```

```
## Start:  AIC=131.54
## y ~ factor(contact) + factor(month) + duration + balance + previous +
##     factor(loan) + factor(default)

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

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

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

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

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

##                     Df Deviance    AIC
## - factor(default)  1   94.212 130.21
## - factor(loan)     1   94.431 130.43
```

```
## <none>                 93.539 131.54
## - previous         1   96.186 132.19
## - balance          1   96.252 132.25
## - factor(contact)  2   98.449 132.45
## - factor(month)   11  144.933 160.93
## - duration         1  271.016 307.02

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

##
## Step:  AIC=130.21
## y ~ factor(contact) + factor(month) + duration + balance + previous +
##     factor(loan)

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

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

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

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

##                   Df Deviance    AIC
## - factor(loan)     1   95.153 129.15
## <none>                 94.212 130.21
## - factor(contact)  2   98.748 130.75
## - previous         1   96.929 130.93
## - balance          1   97.052 131.05
## - factor(month)   11  146.093 160.09
## - duration         1  271.975 305.98

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

##
## Step:  AIC=129.15
## y ~ factor(contact) + factor(month) + duration + balance + previous

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

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

##                   Df Deviance    AIC
## <none>                 95.153 129.15
## - previous         1   97.834 129.83
## - balance          1   97.892 129.89
## - factor(contact)  2   99.943 129.94
## - factor(month)   11  148.431 160.43
## - duration         1  273.157 305.16

##
## Call:  glm(formula = y ~ factor(contact) + factor(month) + duration +
##     balance + previous, family = binomial, data = bank3)
##
## Coefficients:
##             (Intercept)  factor(contact)telephone
##              -8.377e+00                 8.993e-01
##   factor(contact)unknown          factor(month)aug
##              -2.277e+00                 3.163e-01
```

```
##         factor(month)dec         factor(month)feb
##                6.259e-01               -6.240e-01
##         factor(month)jan         factor(month)jul
##                8.148e-04               -6.775e-01
##         factor(month)jun         factor(month)mar
##                1.552e+00                5.569e+00
##         factor(month)may         factor(month)nov
##               -2.215e+00               -1.865e+01
##         factor(month)oct         factor(month)sep
##                2.338e+00               -1.485e+01
##                 duration                  balance
##                6.363e-03               -3.562e-04
##                 previous
##                1.599e-01
##
## Degrees of Freedom: 3020 Total (i.e. Null);  3004 Residual
## Null Deviance:       327.2
## Residual Deviance: 95.15     AIC: 129.2
```

From AIC test, the best model would be glm(formula = y ~ factor(contact) + factor(month) + duration + balance + previous, family = binomial, data = bank3)

```
logit.aic<- glm(formula = y ~ factor(contact) + factor(month) + duration +
    balance + previous, family = binomial, data = bank3)
```

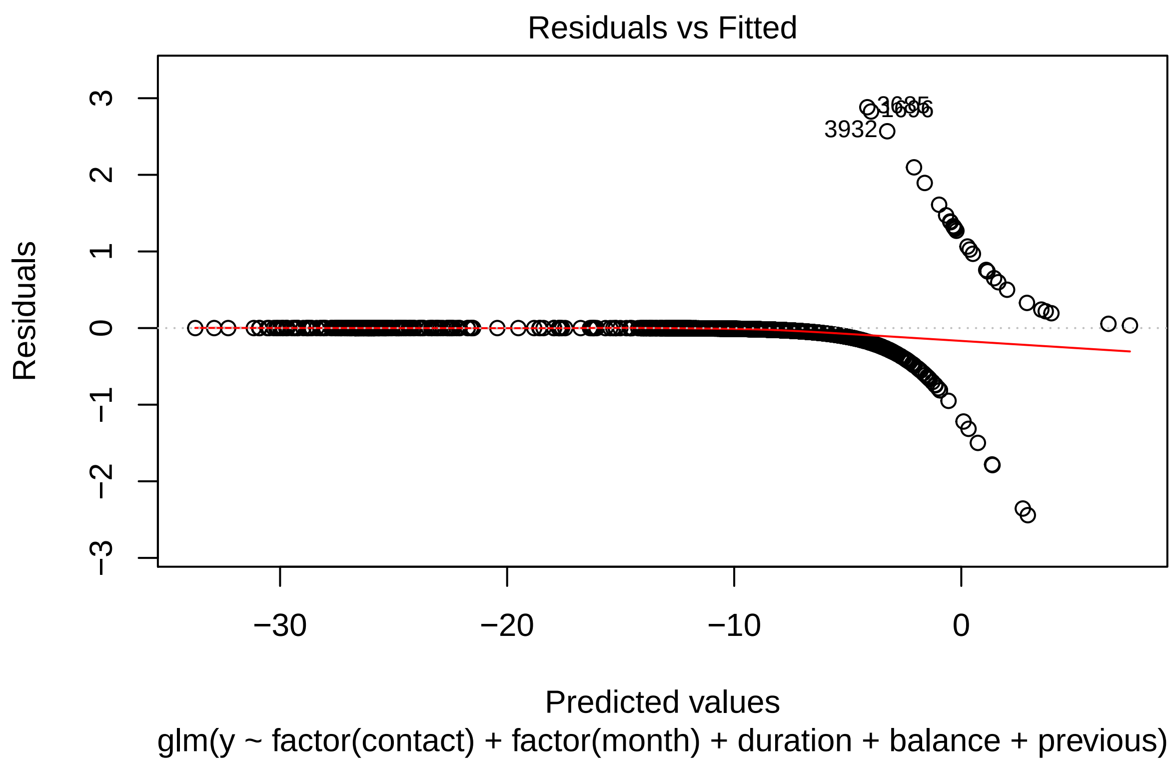
```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

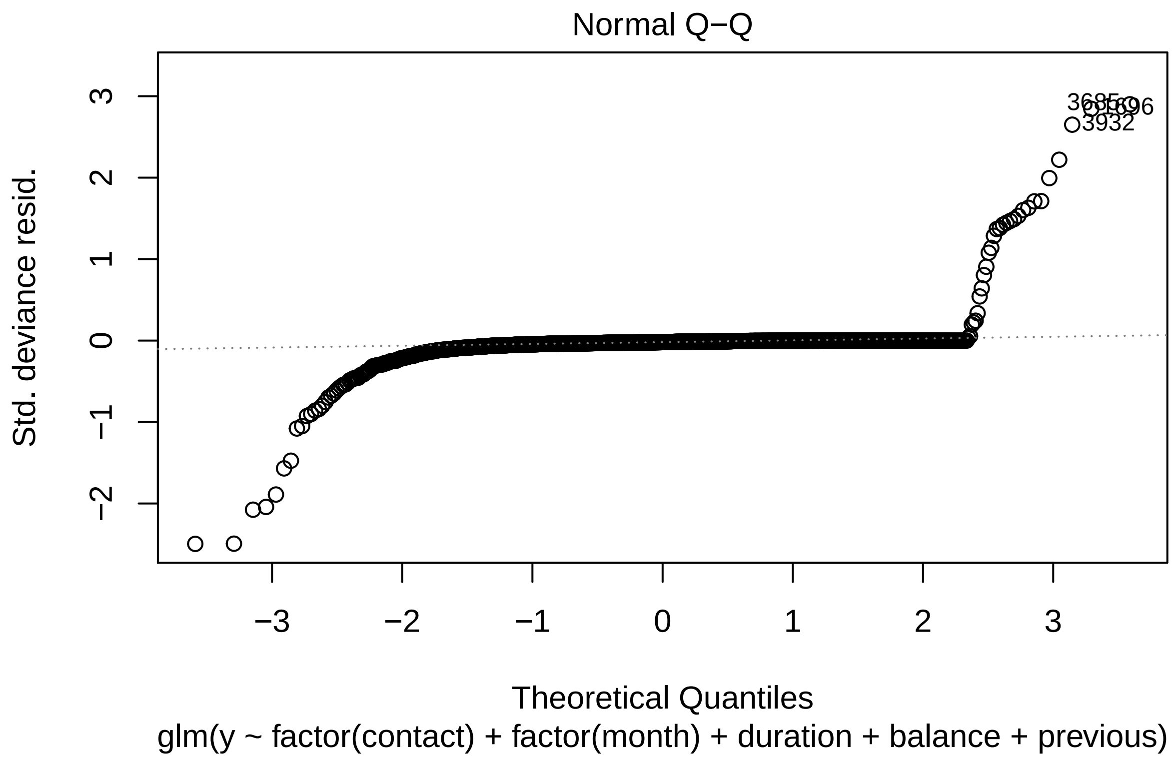
```
summary(logit.aic)
```

```
##
## Call:
## glm(formula = y ~ factor(contact) + factor(month) + duration +
##     balance + previous, family = binomial, data = bank3)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.44204  -0.03419  -0.01839  -0.00445   2.88222
##
## Coefficients:
##                           Estimate Std. Error z value Pr(>|z|)
## (Intercept)              -8.377e+00  1.200e+00  -6.982 2.92e-12 ***
## factor(contact)telephone  8.993e-01  8.847e-01   1.016   0.3094
## factor(contact)unknown   -2.277e+00  1.303e+00  -1.748   0.0805 .
## factor(month)aug          3.163e-01  1.142e+00   0.277   0.7819
## factor(month)dec          6.259e-01  1.862e+00   0.336   0.7368
## factor(month)feb         -6.240e-01  1.914e+00  -0.326   0.7444
## factor(month)jan          8.148e-04  1.560e+00   0.001   0.9996
## factor(month)jul         -6.775e-01  1.200e+00  -0.565   0.5723
## factor(month)jun          1.552e+00  1.365e+00   1.137   0.2554
## factor(month)mar          5.569e+00  1.262e+00   4.414 1.01e-05 ***
## factor(month)may         -2.215e+00  1.426e+00  -1.553   0.1204
## factor(month)nov         -1.865e+01  1.326e+03  -0.014   0.9888
## factor(month)oct          2.338e+00  1.444e+00   1.619   0.1054
## factor(month)sep         -1.485e+01  5.023e+03  -0.003   0.9976
## duration                  6.363e-03  7.787e-04   8.172 3.04e-16 ***
## balance                  -3.562e-04  2.584e-04  -1.379   0.1680
```

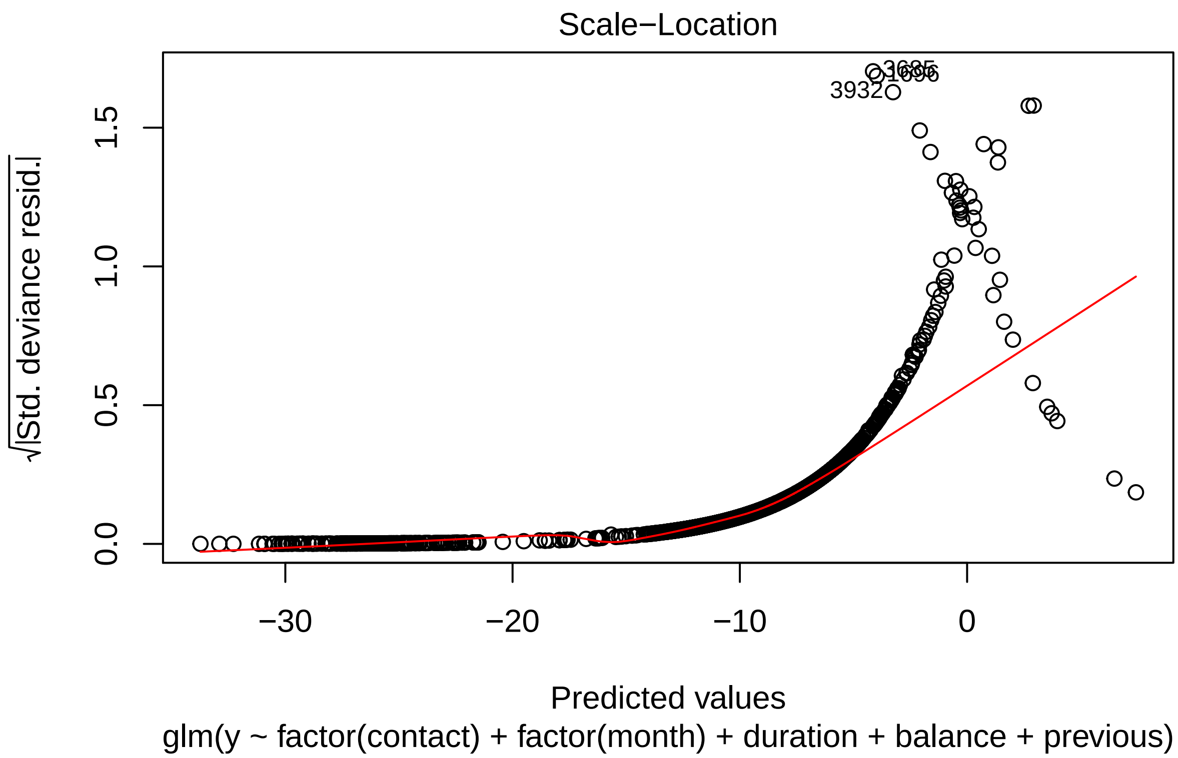
```
## previous                    1.599e-01  9.045e-02   1.768   0.0770 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 327.191  on 3020  degrees of freedom
## Residual deviance:  95.153  on 3004  degrees of freedom
## AIC: 129.15
##
## Number of Fisher Scoring iterations: 20
```

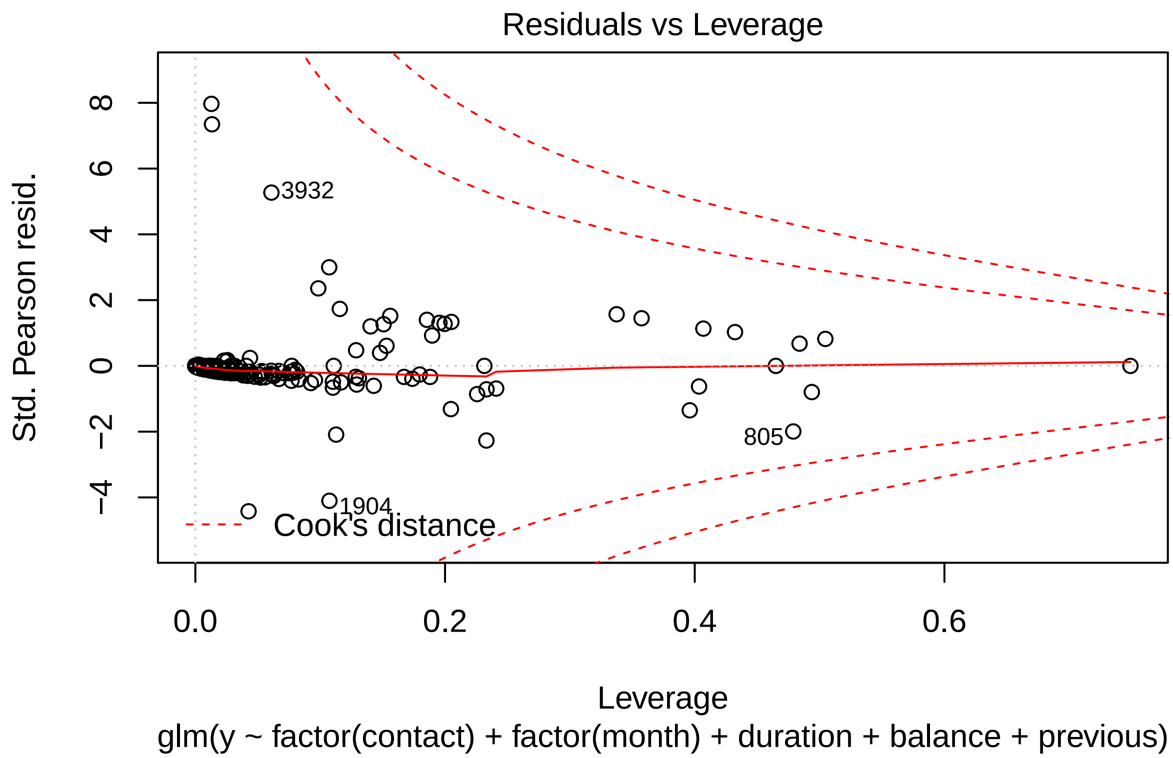
```r
plot(logit.aic)
```





Normal Q−Q
Std. deviance resid.
Theoretical Quantiles
glm(y ~ factor(contact) + factor(month) + duration + balance + previous)
3932


Scale−Location
√|Std. deviance resid.|
0.0
0.5
1.0
1.5
−30
−20
−10
0
3685
3932
Predicted values
glm(y ~ factor(contact) + factor(month) + duration + balance + previous)



If we use BIC, the function is stepAIC(logit.2,k=log(length(bank3[,1])))

```
stepAIC(logit.2,k=log(length(bank3[,1])))
```

```
## Start:  AIC=245.79
## y ~ factor(contact) + factor(month) + duration + balance + previous +
##     factor(loan) + factor(default)

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

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

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

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

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

##                   Df Deviance    AIC
## - factor(month)   11  144.933 209.04
## - factor(contact)  2   98.449 234.68
## - factor(default)  1   94.212 238.45
## - factor(loan)     1   94.431 238.67
## - previous         1   96.186 240.43
## - balance          1   96.252 240.49
## <none>                 93.539 245.79
## - duration         1  271.016 415.26
##
```

```
## Step:  AIC=209.04
## y ~ factor(contact) + duration + balance + previous + factor(loan) +
##     factor(default)
##
##                   Df Deviance    AIC
## - factor(contact)  2   153.58 201.66
## - factor(default)  1   146.09 202.19
## - previous         1   146.40 202.50
## - balance          1   146.72 202.82
## - factor(loan)     1   147.25 203.34
## <none>                 144.93 209.04
## - duration         1   313.17 369.26
##
## Step:  AIC=201.66
## y ~ duration + balance + previous + factor(loan) + factor(default)
##
##                   Df Deviance    AIC
## - factor(default)  1   153.93 194.00
## - factor(loan)     1   155.50 195.56
## - balance          1   155.64 195.71
## - previous         1   157.04 197.11
## <none>                 153.58 201.66
## - duration         1   315.97 356.04
##
## Step:  AIC=194
## y ~ duration + balance + previous + factor(loan)
##
##                Df Deviance    AIC
## - factor(loan)  1   155.89 187.95
## - balance       1   156.05 188.10
## - previous      1   157.32 189.38
## <none>              153.93 194.00
## - duration      1   316.47 348.53
##
## Step:  AIC=187.95
## y ~ duration + balance + previous
##
##            Df Deviance    AIC
## - balance   1   157.73 181.77
## - previous  1   159.68 183.72
## <none>          155.89 187.95
## - duration  1   319.05 343.09
##
## Step:  AIC=181.77
## y ~ duration + previous
##
##            Df Deviance    AIC
## - previous  1   162.30 178.33
## <none>          157.73 181.77
## - duration  1   322.27 338.29
##
## Step:  AIC=178.33
## y ~ duration
##
```

```
##             Df Deviance    AIC
## <none>          162.30 178.33
## - duration  1   327.19 335.20
```

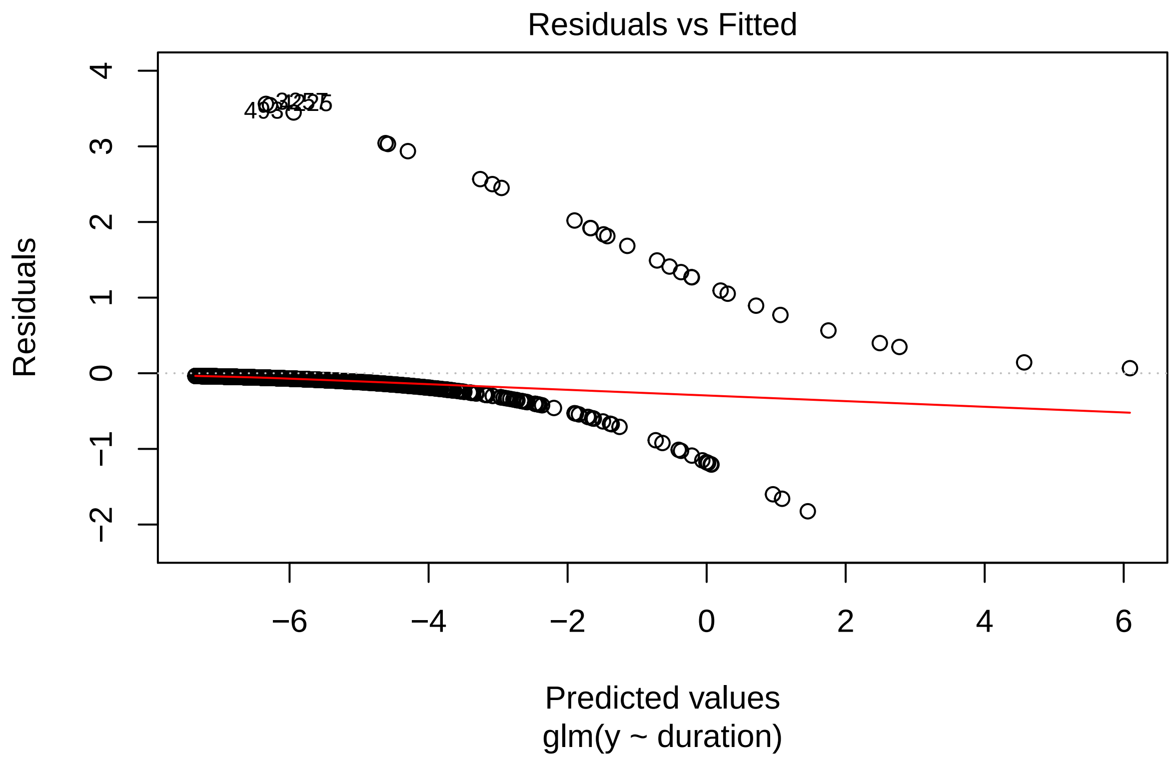
```
##
## Call:  glm(formula = y ~ duration, family = binomial, data = bank3)
##
## Coefficients:
## (Intercept)     duration
##   -7.376387     0.004863
##
## Degrees of Freedom: 3020 Total (i.e. Null);  3019 Residual
## Null Deviance:       327.2
## Residual Deviance: 162.3     AIC: 166.3
```

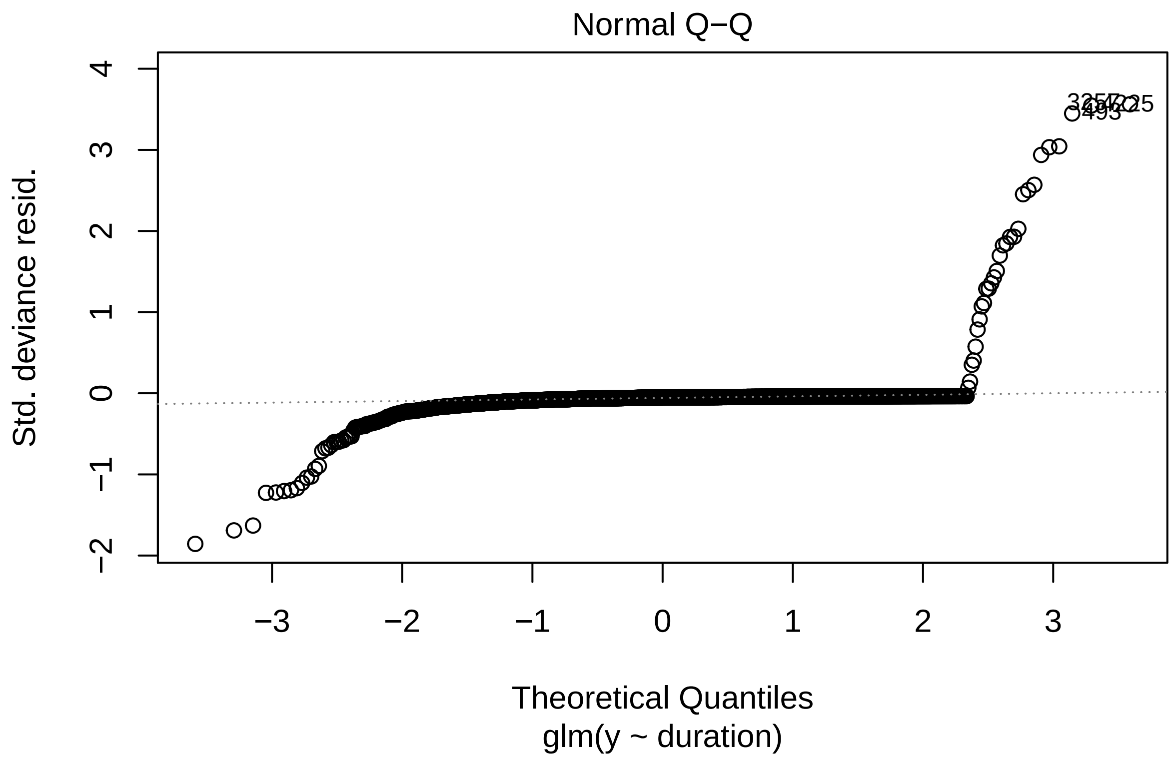
From BIC test the best model would be glm(formula = y ~duration, family = binomial, data = bank3)

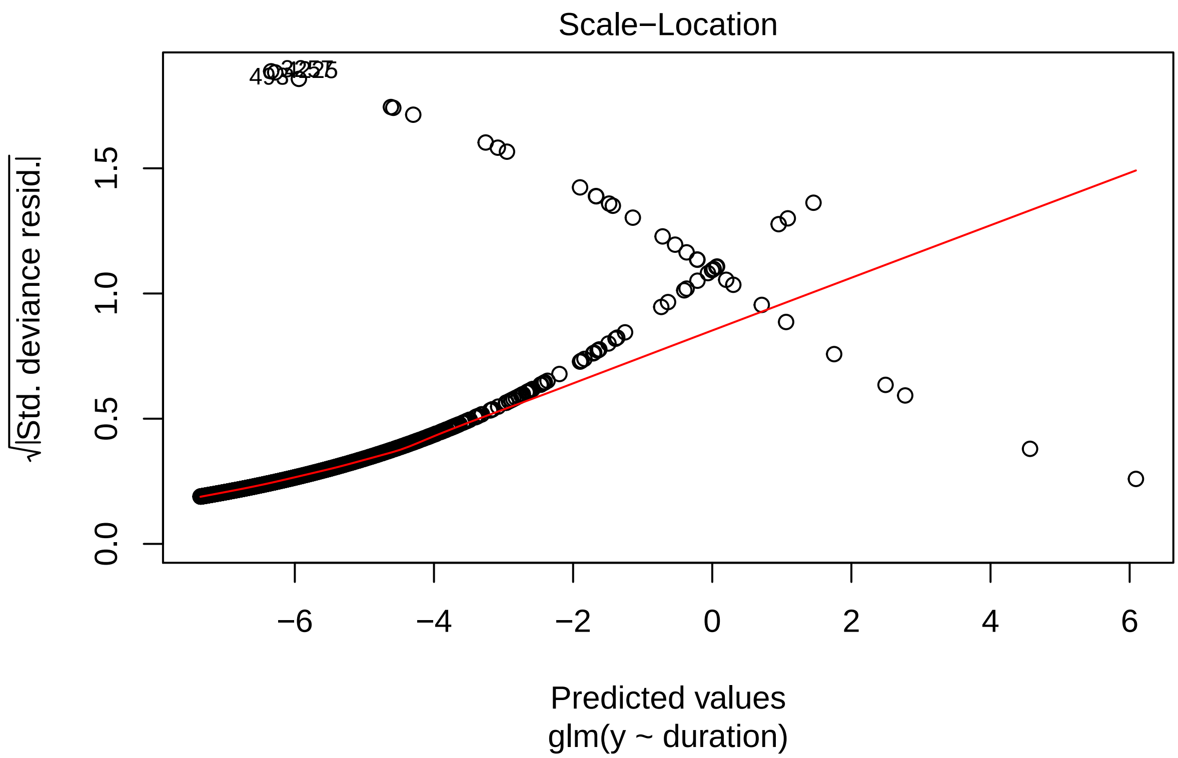
```
logit.bic<-glm(formula = y ~duration, family = binomial, data = bank3)

summary(logit.bic)
```

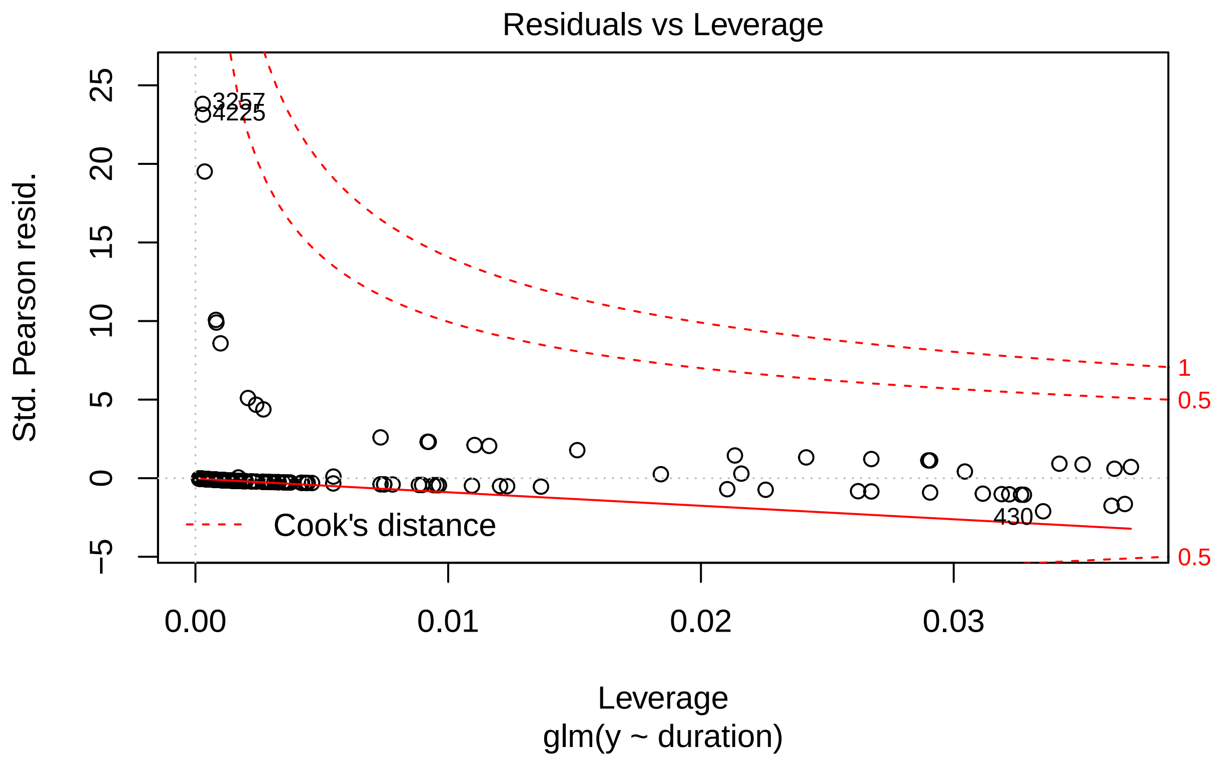
```
##
## Call:
## glm(formula = y ~ duration, family = binomial, data = bank3)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.8249  -0.0704  -0.0530  -0.0445   3.5615
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -7.3763867  0.4850989  -15.21   <2e-16 ***
## duration     0.0048633  0.0004541   10.71   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 327.19  on 3020  degrees of freedom
## Residual deviance: 162.30  on 3019  degrees of freedom
## AIC: 166.3
##
## Number of Fisher Scoring iterations: 9
```

```
plot(logit.bic)
```


Residuals vs Fitted
Residuals
4
3
2
1
0
−1
−2
−6
−4
−2
0
2
4
6
Predicted values
glm(y ~ duration)


Normal Q−Q
Std. deviance resid.
Theoretical Quantiles
glm(y ~ duration)
−3
−2
−1
0
1
2
3
4
3493


Scale−Location
√|Std. deviance resid.|
0.0
0.5
1.0
1.5
−6
−4
−2
0
2
4
6
Predicted values
glm(y ~ duration)



Predictive accuracy calculation

Let's do the predictions now:

```
prediction <- data.frame(predict(logit.bic,bank3,type="response"))
prediction[prediction<0.5]=0
prediction[prediction>=0.5]=1
predictions <- data.frame(Prediction = as.numeric(prediction[,1]),Actual = as.numeric(bank3$y)-1)
predictions$Correct <- (predictions$Actual == predictions$Prediction)
logistic_accuracy<-table(predictions$Correct)/length(predictions$Correct)*100
logistic_accuracy
```

```
##
##      FALSE       TRUE
##  0.8606422 99.1393578
```

```
#table(predictions$Actual, predictions$Prediction)
```

The accuracy is 99.14% which is really high.

```
prediction.test<-data.frame(predict(logit.bic,testing.set,type="response"))
prediction.test[prediction.test<0.5]=0
prediction.test[prediction.test>=0.5]=1
predictions.test <- data.frame(Prediction = as.numeric(prediction.test[,1]),Actual = as.numeric(testing
predictions.test$Correct <- (predictions.test$Actual == predictions.test$Prediction)
logistic_accuracy.test<-table(predictions.test$Correct)/length(predictions.test$Correct)*100
logistic_accuracy.test
```

```
##
##    FALSE     TRUE
## 11.05217 88.94783
```

```r
#table(predictions.test$Actual, predictions.test$Prediction)
```

The accuracy is 88.95% which is less than prediction of training.set.

Decision tree model

Next one would be decision tree model

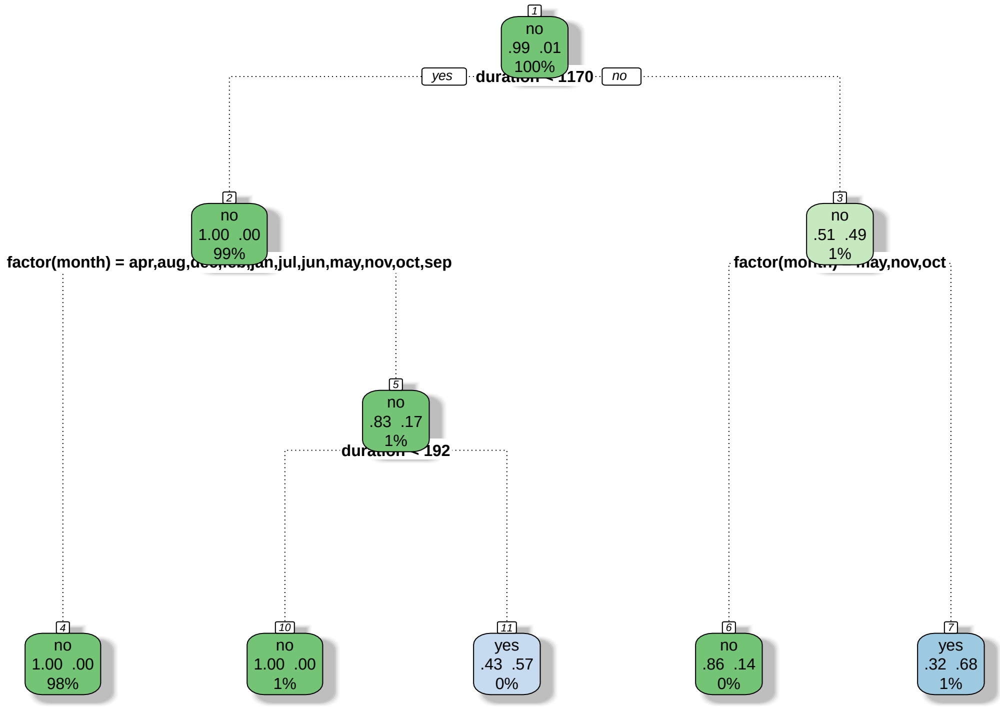
```r
#install.packages("ElemStatLearn")
#install.packages("tree")
#install.packages("rpart")
#install.packages("rattle")
#install.packages("rpart.plot")
#install.packages("RcolorBrewer")

library(ElemStatLearn)
library(tree)
require(rpart)
```

```
## Loading required package: rpart
```

```r
library(rpart)
tree <- rpart(y~factor(contact) + factor(month) + duration +
    balance + previous, data=bank3, method="class")
library(rattle)
```

```
## Rattle: A free graphical interface for data science with R.
## Versión 5.1.0 Copyright (c) 2006-2017 Togaware Pty Ltd.
## Escriba 'rattle()' para agitar, sacudir y  rotar sus datos.
```

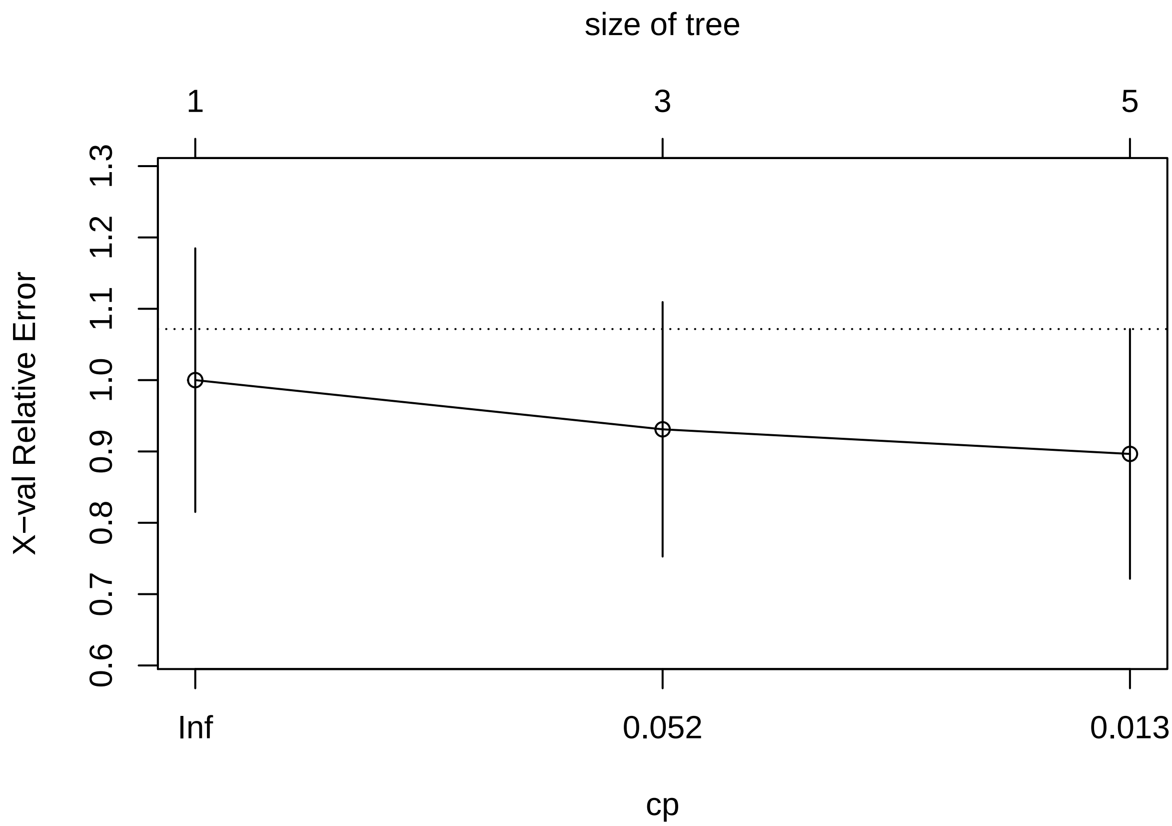
```r
library(rpart.plot)
library(RColorBrewer)
fancyRpartPlot(tree,main = "", sub = "",cex=0.5)
```



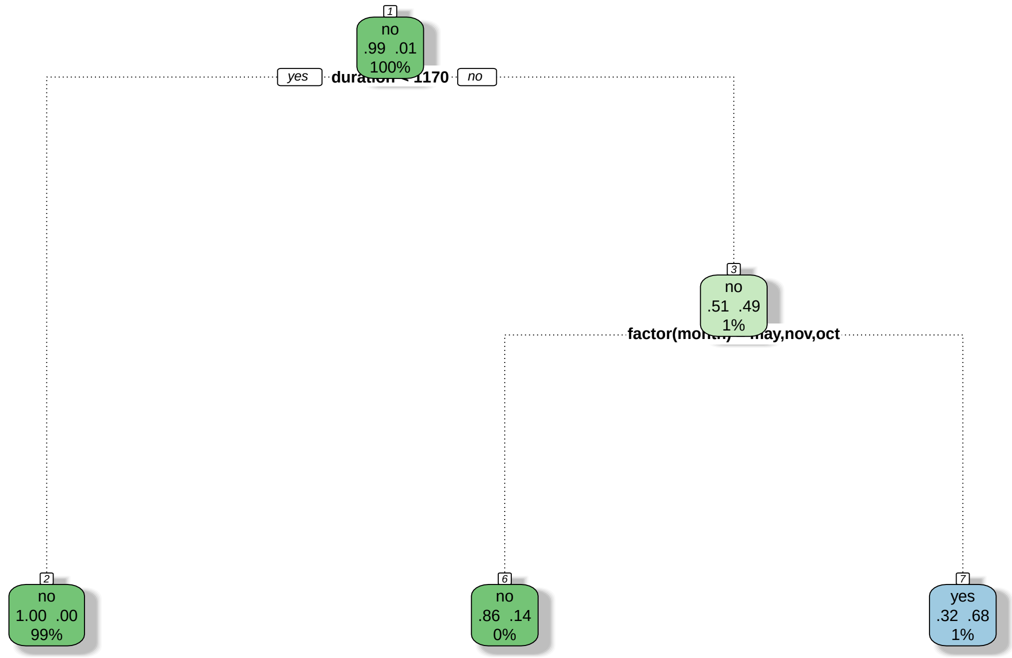

```r
# proportion of "no", "yes" and sample proportion
printcp(tree)
```

```
##
## Classification tree:
## rpart(formula = y ~ factor(contact) + factor(month) + duration +
##     balance + previous, data = bank3, method = "class")
##
## Variables actually used in tree construction:
## [1] duration      factor(month)
##
## Root node error: 29/3021 = 0.0095995
##
## n= 3021
##
##         CP nsplit rel error  xerror    xstd
## 1 0.155172      0   1.00000 1.00000 0.18480
## 2 0.017241      2   0.68966 0.93103 0.17838
## 3 0.010000      4   0.65517 0.89655 0.17507
```

```r
plotcp(tree)
```




```r
tree.prune = prune(tree, cp = 0.05)
fancyRpartPlot(tree.prune,main = "", sub = "",cex=0.5)
```



Predictive accuracy calculation

Make prediction

```
prediction.tree <- data.frame(predict(tree.prune, bank3, type = "class"))
predictions.tree <- data.frame(Prediction = as.numeric(prediction.tree[,1])-1,Actual = as.numeric(bank3
predictions.tree$Correct <- (predictions.tree$Actual == predictions.tree$Prediction)
Tree_Accuracy <- table(predictions.tree$Correct)/length(predictions.tree$Correct)*100
Tree_Accuracy
```

```
##
##      FALSE       TRUE
##  0.6620324 99.3379676
```

predict with test

```
prediction.tree.test<-data.frame(predict(tree.prune,testing.set,
     type="class"))
predictions.tree.t <- data.frame(Prediction = as.numeric(prediction.tree.test[,1])-1,Actual = as.numeri
predictions.tree.t$Correct <- (predictions.tree.t$Actual == predictions.tree.t$Prediction)
Tree_Accuracy.t <- table(predictions.tree.t$Correct)/length(predictions.tree.t$Correct)*100
Tree_Accuracy.t
```

```
##
##    FALSE     TRUE
## 11.05217 88.94783
```

Random forest

Random Forrest

```
#install.packages("randomForest")
library(randomForest)
```

```
## randomForest 4.6-12
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

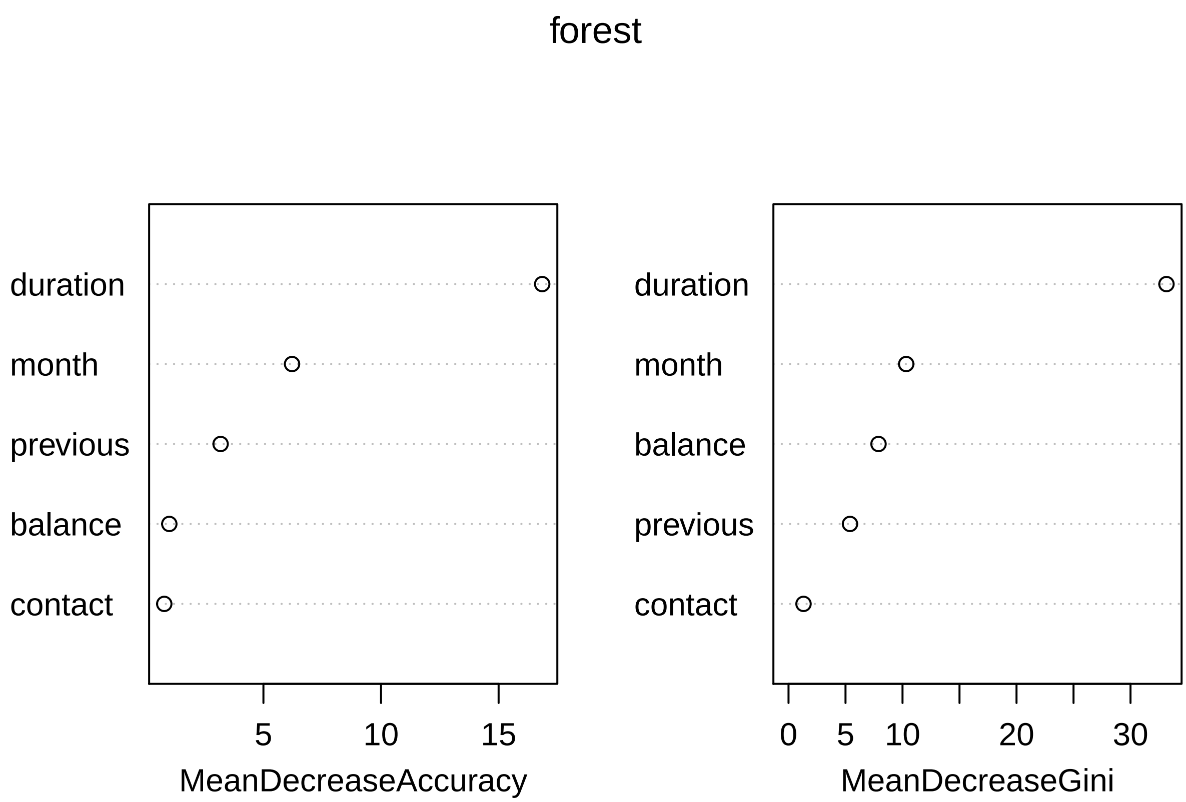
```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:rattle':
##
##     importance
```

```
forest<-randomForest(as.factor(y)~contact + month+ duration + balance+
                       previous,data=bank2, importance=TRUE, ntree=100)
#nstead of specifying method="class" as with rpart, we force the model
#to predict our classification by temporarily changing our target
#variable to a factor with only two levels using as.factor(). The
#importance=TRUE argument allows us to inspect variable importance
#as we'll see, and the ntree argument specifies how many trees we want to grow.
#If you were working with a larger dataset you may want to reduce
#the number of trees, at least for initial exploration, or restrict the complexity
#of each tree using nodesize as well as reduce the number of rows sampled with
#sampsize. You can also override the default number of variables to choose
#from with mtry, but the default is the square root of the total number
#available and that should work just fine. Since we only have a small
#dataset to play with, we can grow a large number of trees and not worry
#too much about their complexity, it will still run pretty fast.
summary(forest)
```

```
##                 Length Class  Mode
## call               5   -none- call
## type               1   -none- character
## predicted       3024   factor numeric
## err.rate         300   -none- numeric
## confusion          6   -none- numeric
## votes           6048   matrix numeric
## oob.times       3024   -none- numeric
## classes            2   -none- character
## importance        20   -none- numeric
## importanceSD      15   -none- numeric
## localImportance    0   -none- NULL
## proximity          0   -none- NULL
## ntree              1   -none- numeric
## mtry               1   -none- numeric
## forest            14   -none- list
## y               3024   factor numeric
## test               0   -none- NULL
## inbag              0   -none- NULL
## terms              3   terms  call
```

```r
varImpPlot(forest)
```




```r
#There's two types of importance measures shown above.
#The accuracy one tests to see how worse the model
#performs without each variable, so a high decrease
#in accuracy would be expected for very predictive variables.
#The Gini one digs into the mathematics behind decision trees,
#but essentially measures how pure the nodes are at the end of the tree.
#Again it tests to see the result if each variable is taken out and
#a high score means the variable was important.
```

Predictive accuracy calculation

Make prediction

```r
prediction.forest <- data.frame(predict(forest, bank2, type = "class"))
predictions.forest <- data.frame(Prediction = as.numeric(prediction.forest[,1])-1,Actual = as.numeric(ba
predictions.forest$Correct <- (predictions.forest$Actual == predictions.forest$Prediction)
forest_Accuracy <- table(predictions.forest$Correct)/length(predictions.forest$Correct)*100
forest_Accuracy
```

```
##
## TRUE
##  100
```

predict with test

```
prediction.forest.test<-data.frame(predict(forest,testing.set,type="class"))
predictions.forest.t <- data.frame(Prediction = as.numeric(prediction.forest.test[,1])-1,Actual = as.nu
predictions.forest.t$Correct <- (predictions.forest.t$Actual == predictions.forest.t$Prediction)
Forest_Accuracy.t <- table(predictions.forest.t$Correct)/length(predictions.forest.t$Correct)*100
Forest_Accuracy.t
```

```
##
##    FALSE     TRUE
## 11.40584 88.59416
```

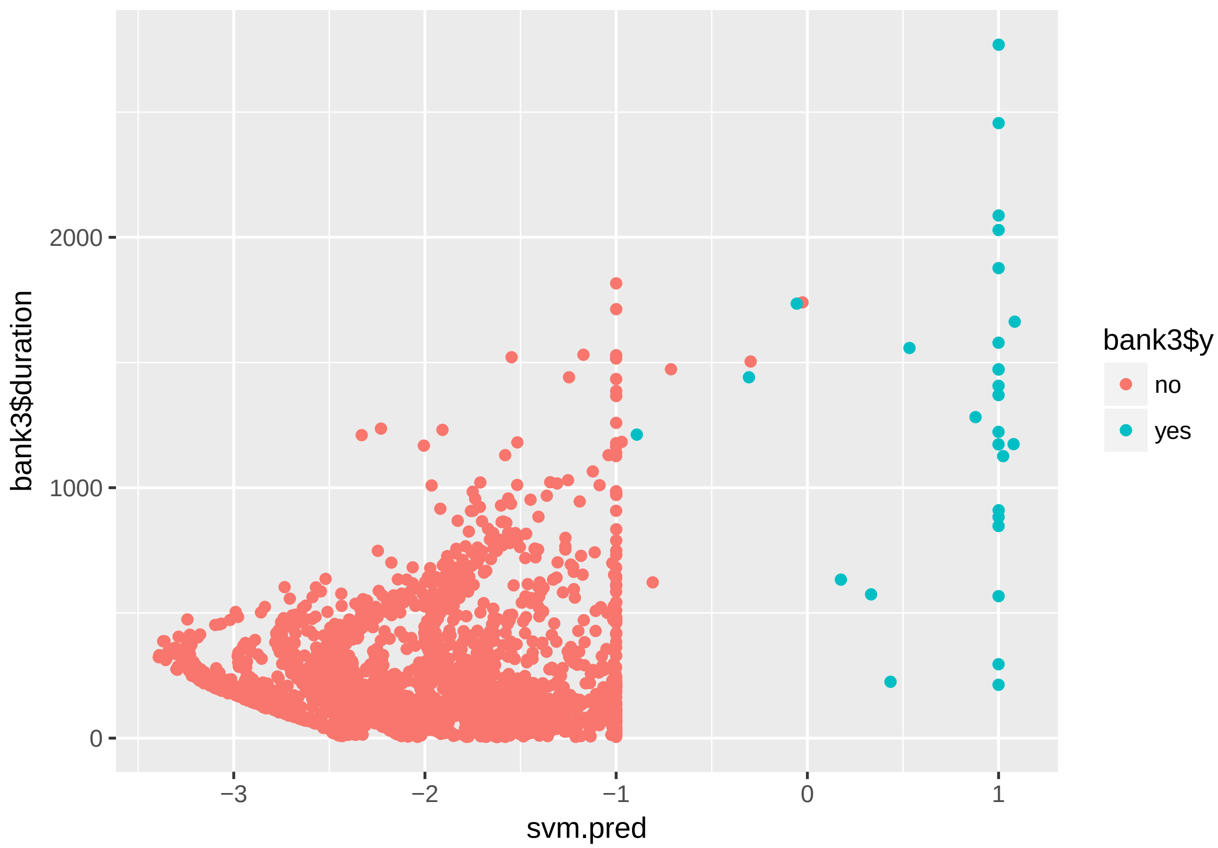
Support vector machine

Support vector machine

```
#install.packages("e1071")
#install.packages("kernlab")

library(e1071)
library(kernlab)

svm.fit = ksvm(y~ contact + month+ duration + balance+
                       previous, data = bank3, type="C-svc", kernel="rbfdot", C=10)
svm.pred <- predict(svm.fit, bank3, type = "decision")
library(ggplot2)
```

```
##
## Attaching package: 'ggplot2'

## The following object is masked from 'package:kernlab':
##
##     alpha

## The following object is masked from 'package:randomForest':
##
##     margin
```

```
qplot(svm.pred, bank3$duration, color=bank3$y)
```




```
summary(svm.fit)
```

```
## Length  Class   Mode
##      1   ksvm     S4
```

Predictive accuracy calculation

Make prediction

```
prediction.svm <- data.frame(predict(svm.fit, bank3))
predictions.svm <- data.frame(Prediction = as.numeric(prediction.svm[,1])-1,Actual = as.numeric(bank3$y)
predictions.svm$Correct <- (predictions.svm$Actual == predictions.svm$Prediction)
svm_Accuracy <- table(predictions.svm$Correct)/length(predictions.svm$Correct)*100
svm_Accuracy
```

```
##
##       FALSE        TRUE
##  0.09930487 99.90069513
```

predict with test

```
prediction.svm.test<-data.frame(predict(svm.fit,testing.set))
predictions.svm.t <- data.frame(Prediction = as.numeric(prediction.svm.test[,1])-1,Actual = as.numeric(t
predictions.svm.t$Correct <- (predictions.svm.t$Actual == predictions.svm.t$Prediction)
svm_Accuracy.t <- table(predictions.svm.t$Correct)/length(predictions.svm.t$Correct)*100
svm_Accuracy.t
```

```
##
```

```
##    FALSE     TRUE
## 11.49425 88.50575
```

save.image("Prog-09-allobjects.Rdata")