Chapter 5

Logistic Regression

Regression analysis is one of the most widely used statistical techniques. For example, Capital Asset Pricing Model (CAPM) model is a regression of an individual asset's return on the market rate of return. That is, $E(R_i) = r + \beta_i[E(R_m) - r]$, where
R_i = relative return of the *i-th* asset; R_m = relative return of the market
r = risk free interest rate
β_i = sensitivity of the expected excess returns to the expected excess market returns.

In fact, the above CAPM model is a simple linear regression model. We can extend it to multiple linear regression model. The general form of this model is:

$$y_i = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip} + \varepsilon_i \quad \text{where } i = 1, \ldots, n$$

This regression procedure is included in every statistical package and each will return with $\hat{\beta}$, the least square estimate of β, the fitted values

$\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_{i1} + \cdots + \hat{\beta}_p x_{ip}$, and residuals $e_i = y_i - \hat{y}_i$, i=1,…,n.

However, ordinary linear regression has its limitations and can be easily abused. Basically, regression has the following assumptions:

1. The regression function of y is a linear function of x, i.e.
   $E(y \mid x) = \beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p$.
2. ε_i independently identically distributed as $N(0, \sigma^2)$ (or just i.i.d as martingale differences).
3. The error variance σ^2 is constant.

Usually, residuals plots are used to check for these assumptions. QQ-normal plot of the residuals is used to check for the normality assumption, and the plot of residuals versus fitted values or the observation number is used to check the linearity and constant error variance assumptions. The plot of residuals versus lag(residuals) is to check for independency. Let us look at the following example and demonstrate how the linear regression can be abused.

5.1 Blue Chips in Hong Kong Stock Exchange

The data file "fin-ratio.csv" contains financial ratios of 680 securities listed in the main board of Hong Kong Stock Exchange in 2002. There are six financial variables, namely, **Earning Yield** (EY), **Cash Flow to Price** (CFTP), **logarithm of Market Value** (ln_MV), **Dividend Yield** (DY), **Book to Market Equity** (BTME), **Debt to Equity Ratio** (DTE). These financial variables are publicly available information. Among these companies, there are 32 Blue Chips which are the Hang Seng Index Constitute Stocks. The last column HSI is a binary variable indicating whether the stock is a Blue Chip or not. Now suppose we are interested in finding the relationship between HSI and their six financial variables. We run an ordinary least square regression of HSI on these six financial variables using R's built-in function *lm()* (stand for linear model).

```
> d<-read.csv("fin-ratio.csv")     # read in data
> names(d)                         # display the variable names
[1] "EY"    "CFTP"  "ln_MV" "DY"    "BTME"  "DTE"   "HSI"
> summary(lm(HSI~EY+CFTP+ln_MV+DY+BTME+DTE,data=d)) # linear model

Call:
lm(formula = d$HSI ~ d$EY + d$CFTP + d$ln_MV + d$DY + d$BTME + d$DTE)

Residuals:
     Min       1Q   Median       3Q      Max
-0.32104 -0.08546 -0.01672  0.05592  0.73866

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.4591209  0.0268310 -17.112  < 2e-16 ***
d$EY        -0.0017172  0.0016181  -1.061  0.28896
d$CFTP      -0.0103792  0.0037321  -2.781  0.00557 **
d$ln_MV      0.0810286  0.0040887  19.818  < 2e-16 ***
d$DY        -0.0027336  0.0017826  -1.534  0.12561
d$BTME       0.0004798  0.0007938   0.604  0.54575
d$DTE        0.0010610  0.0018035   0.588  0.55655
---
Signif. codes:  0 `***' 0.001 `**' 0.01 `*' 0.05 `.' 0.1 ` ' 1

Residual standard error: 0.1689 on 673 degrees of freedom
Multiple R-Squared: 0.3708,     Adjusted R-squared: 0.3652
F-statistic: 66.09 on 6 and 673 DF,  p-value: < 2.2e-16
```

From the output, the p-value of *d$DTE* is largest, this means the coefficient of *DTE* is not significantly different from zero. Therefore we should exclude it in our regression. We then continue to fit the regression using the other five financial variables only:

```
> summary(lm(HSI~EY+CFTP+ln_MV+DY+BTME,data=d))
```

We will find that the p-value of *d$BTME* is large and *d$BTME* should be excluded. We keep on excluding the variable with largest p-value one by one until all the p-values are small (say, less that *0.1*). This is known as the **backward elimination** of modeling selection in regression.

Finally, we arrived at the following model:

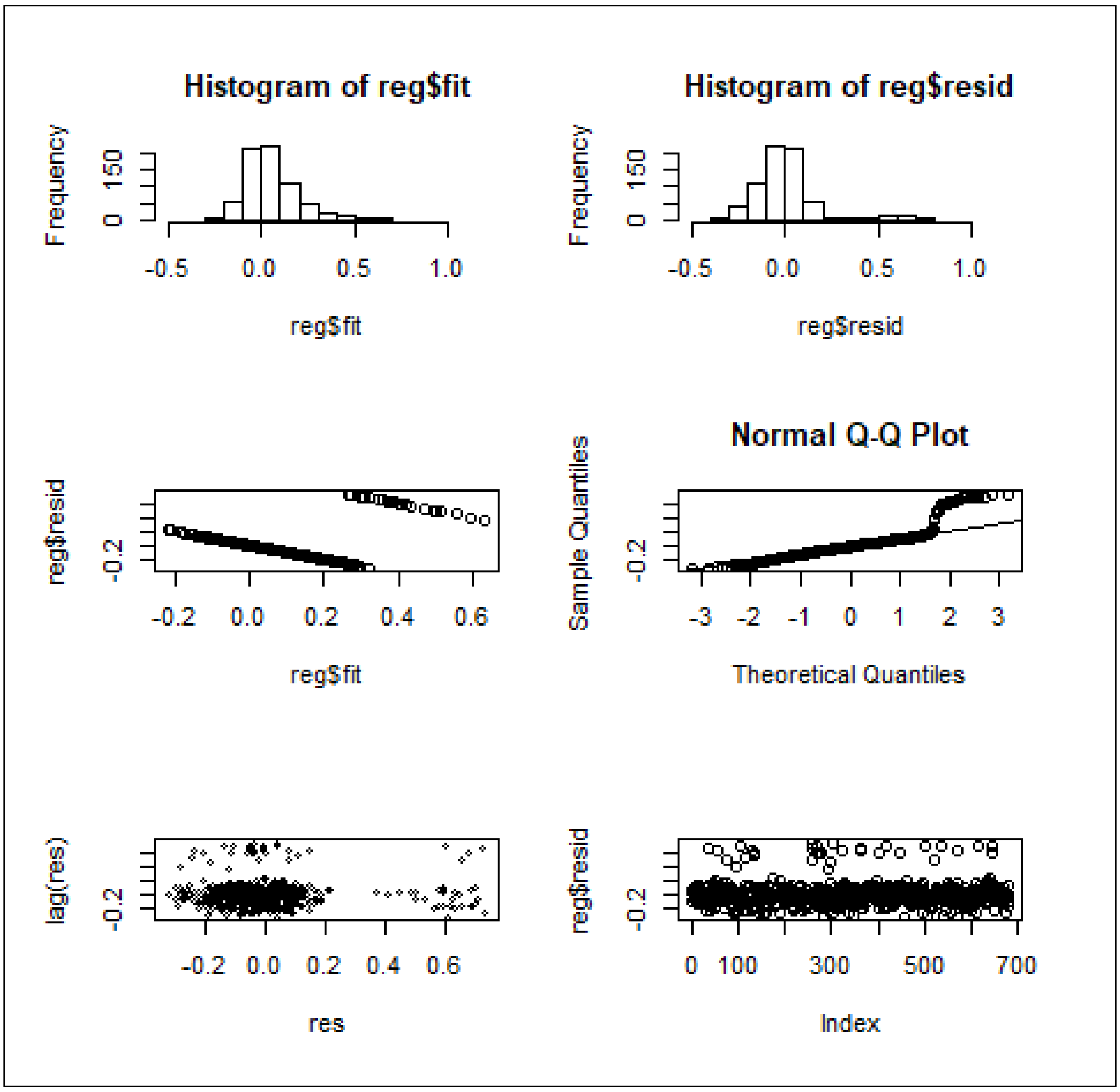
```
> summary(lm(HSI~CFTP+ln_MV,data=d))
Call:
lm(formula = d$HSI ~ d$CFTP + d$ln_MV)
Residuals:
     Min       1Q   Median       3Q      Max 
-0.32409 -0.08559 -0.01729  0.05688  0.73488 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept) -0.454781   0.026284 -17.303  < 2e-16 ***
d$CFTP      -0.012026   0.003475  -3.461 0.000573 ***
d$ln_MV      0.079630   0.004032  19.751  < 2e-16 ***
---
Signif. codes:  0 `***' 0.001 `**' 0.01 `*' 0.05 `.' 0.1 ` ' 1 

Residual standard error: 0.1689 on 677 degrees of freedom
Multiple R-Squared: 0.3666,     Adjusted R-squared: 0.3648 
F-statistic: 195.9 on 2 and 677 DF,  p-value: < 2.2e-16
```

Although we have the least square regression model:

$$HSI = -0.454781 - 0.01026(CFTP) + 0.07963(ln\_MV),$$

it does not mean that this model is correct. First, the fitted value of *HSI* can be any real number other than 0 or 1. Second, we need to look at the residuals of this regression model to check for the validity of the assumptions.

```
> reg<-lm(HSI~CFTP+ln_MV,data=d)   # save the regression results
> names(reg)                       # display the items contained in reg
 [1] "coefficients"  "residuals"     "effects"       "rank"         
 [5] "fitted.values" "assign"        "qr"            "df.residual"  
 [9] "xlevels"       "call"          "terms"         "model"        
> par(mfrow=c(3,2))                # set a 3x2 multiple frame for graphics
> plot(reg$fitted.values,reg$residuals)    # residuals vs fitted values
> qqnorm(reg$residuals)                    # qq-normal plot of residuals
> qqline(reg$residuals)                    # add reference line
> res<-as.ts(reg$residuals)                # change res to time series
> plot(res,lag(res))                       # residuals vs lag(residuals)
> plot(reg$residuals)                      # residuals vs index number
```



If the assumption is correct, the normal Q-Q plot should be close to a straight line and all other plots should be random. While HSI takes only binary values of 0 or 1, the fitted values of HSI distribute continuously over a large range of values. Also, the residues, while dominantly distribute around 0, has a secondary peak with large positive value. This secondary peak of large positive residue values is likely associated with those data with HSI=1, and the resulting QQ-plot shows heavy-tail features. From the output, the linear regression assumptions are seriously violated. It is not surprising and is mainly due to the fact that *y* (=HSI) is binary.

5.2 Binary Logistic regression

A commonly used statistical method to deal with binary response variable is the logistic regression model. First we define a parameter

$$\pi_i = \Pr(Y_i = 1 \mid x_i) \qquad (5.1)$$

This parameter can be interpreted as the probability of Y_i (probability of "success") given x_i. The crucial assumption in the logistic regression model is:

$$\ln[\pi_i/(1-\pi_i)] = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip} = x'\beta \qquad (5.2)$$

The left hand side of (5.2) is the **log-odd ratio** of probability of success. In the logistic regression we assume that the log-odd ratio of success probability is a linear function of x_i. Re-express (5.2) as

$$\pi_i = \exp(x_i'\beta)/[1+\exp(x_i'\beta)]. \qquad (5.3)$$

(5.3) known as the **logit** transformation of x_i. Note that π_i always lies between 0 and 1 and is consistent with the interpretation of success probability. Also, Eq. (5.3) satisfies: $\nabla_\beta \pi = x\pi(1-\pi)$, for $p\times 1$ column vectors β (excluding β_0) and x.

The likelihood function is $L(\beta)=\prod_{i=1}^{n}[\pi_i^{y_i}(1-\pi_i)^{(1-y_i)}]$, or the log-likelihood function is

$$\log L(\beta) = \sum_{i=1}^{n}[y_i \log \pi_i + (1-y_i)\log(1-\pi_i)], \qquad (5.4)$$

If we have the maximum likelihood estimate of β by maximizing (5.4), then we can compute the predicted probability of success given x_i using (5.3). The maximization is carried out through the gradient of $\log L(\beta)$ over β, which is,

$$\begin{aligned}
\nabla_\beta \log L(\beta) &= \sum_{i=1}^{n}\left(y_i \frac{\nabla_\beta \pi}{\pi_i} + (1-y_i)\frac{(-\nabla_\beta \pi)}{1-\pi_i})\right) \\
&= \sum_{i=1}^{n}\left(y_i \frac{x_i\pi_i(1-\pi_i)}{\pi_i} + (1-y_i)\left(\frac{-x_i\pi_i(1-\pi_i)}{1-\pi_i}\right)\right) \\
&= \sum_{i=1}^{n}\left(y_i x_i(1-\pi_i) - (1-y_i)x_i\pi_i\right) \\
&= \sum_{i=1}^{n} x_i(y_i-\pi_i).
\end{aligned}$$

Let us illustrate this by our HSI example. R has a built-in *glm()* function (stand for generalized linear model) to perform logistic regression. The output format is quite similar to *lm()*. The last option binomial is the **link function** of logistic regression in *glm()*.

```
> summary(glm(HSI~EY+CFTP+ln_MV+DY+BTME+DTE,data=d,binomial))
Call:
glm(formula = d$HSI ~ d$EY + d$CFTP + d$ln_MV + d$DY + d$BTME +
    d$DTE, family = binomial)
Deviance Residuals:
       Min          1Q        Median          3Q        Max
    -8.490e+00   -2.107e-08  -2.107e-08    -2.107e-08  8.490e+00
Coefficients:
              Estimate Std. Error    z value Pr(>|z|)
(Intercept) -4.121e+15  1.066e+07 -386410689   <2e-16 ***
d$EY         1.516e+13  6.431e+05   23570628   <2e-16 ***
d$CFTP      -6.364e+13  1.483e+06  -42902735   <2e-16 ***
d$ln_MV      4.945e+14  1.625e+06  304287297   <2e-16 ***
d$DY        -1.144e+14  7.085e+05 -161536188   <2e-16 ***
d$BTME      -7.907e+12  3.155e+05  -25063060   <2e-16 ***
d$DTE        8.744e+12  7.168e+05   12198713   <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:  258.08  on 679  degrees of freedom
Residual deviance: 1153.40  on 673  degrees of freedom
AIC: 1167.4
```

The MLE of the coefficients are extremely large, this may be due to many outliers exist in the dataset. However, let us save these logistic regression output and see how the model perform.

```
> lreg<-glm(HSI~EY+CFTP+ln_MV+DY+BTME+DTE,data=d,binomial)
> names(lreg)        # display the items in lreg
 [1] "coefficients"      "residuals"         "fitted.values"
 [4] "effects"           "R"                 "rank"
 [7] "qr"                "family"            "linear.predictors"
[10] "deviance"          "aic"               "null.deviance"
[13] "iter"              "weights"           "prior.weights"
[16] "df.residual"       "df.null"           "y"
[19] "converged"         "boundary"          "model"
[22] "call"              "formula"           "terms"
[25] "data"              "offset"            "control"
[28] "method"            "contrasts"         "xlevels"

> pr<-(lreg$fitted.values>0.5)     # set pr=True if fitted >0.5 or otherwise
> table(pr,d$HSI)                  # Cross tabulation of pr and HSI

pr      0   1
  FALSE 634   2
  TRUE   14  30
```

The *glm()* output is saved in *lreg*. There are many items contained in *lreg*. One important item is the fitted.vaules. This is the estimated success probability according to (5.3) for each 680 stocks. These numbers lies between 0 and 1 and represent the probability of HSI=1. Since our MLE of the coefficients are extremely large, the fitted values are either very close to 0 or 1. Furthermore, we can assign *pr=True* if *fitted value > 0.5* or otherwise. This can be interpreted as the stock is predicted as HIS constitute stocks or not. Finally, we can produce a cross tabulation table of pr versus HSI to see how well this logistic model predicts. From the output, there are 634+30=664 correct classifications, and 14+2=16 misclassifications. There are 2 stocks with HSI=1 are misclassified as *pr=False* while there are 14 stocks with *HSI=0* are misclassified as *pr=True*. The correct classification rate is 664/680=97.65%

5.3 Outlier detection

The Mahalanobis distance defined in (1.7) in Section 1.5 provides us a useful tool for detecting outliers. Recall that if $X_1,\ldots,X_n$ i.i.d. $N_p(\mu,\Sigma)$, then the Mahalanobis distance $D^2=(x-\bar{x})'S^{-1}(x-\bar{x})\sim\chi_p^2$ approximately.
Let us write a function mdist() to computer the Mahalanobis distance.

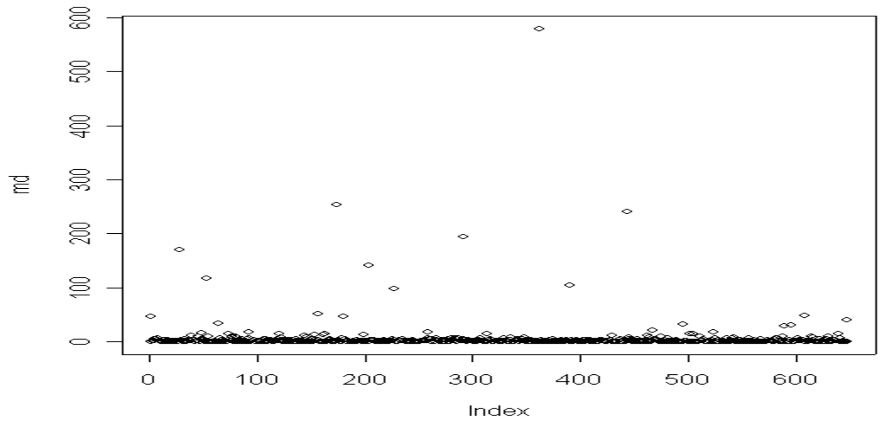
```
mdist<-function(x) {
   t<-as.matrix(x)        # transform x to a matrix
   m<-apply(t,2,mean)     # compute column mean
   s<-var(t)              # compute sample covariance matrix
   mahalanobis(t,m,s)     # using built-in mahalanobis function
}
```

The dataset “fin-ratio.csv” probably contains many outliers. Let us illustrate outlier detection using this as an example. First we read in the data and separate the data into two parts, *d0* for *HSI=0* and *d1* for *HSI=1*.

```
> d<-read.csv("fin-ratio.csv")     # read in dataset
> d0<-d[d$HSI==0,]                 # select HSI=0
> d1<-d[d$HSI==1,]                 # select HSI=1
> dim(d0)
[1] 648   7
> dim(d1)
[1] 32  7
```

We only detect and throw away the outliers in *d0*. Since *d1* contains only 32 cases, we cannot afford to lose any of them.

```
> source("mdist.r")           # load the mdist function
> x<-d0[,1:6]                 # save d0 to x
> md<-mdist(x)                # compute mdist
> plot(md)                    # plot md
```



As shown in the plot, points with large distance are potential outliers. We want to throw away points with large distance. But what is the cut-off value? A more scientific way is to look at the percentile point from a Chi-square distribution with *p* (=6 in our example, the six financial variables) degrees of freedom.

```
> (c<-qchisq(0.99,df=6))  # p=6, and type-I error = 0.01
 [1] 16.81189

> d2<-d0[md<c,]           # select cases from d0 with md<c
> dim(d2)                 # we have throw away 648-626=22 cases
[1] 626   7
> d3<-rbind(d1,d2)        # combine d1 with d2 to form a cleaned dataset
> dim(d3)
[1] 658   7
#save the cleaned dataset to ” fin-ratio1.csv”
> write.csv(d3,file="fin-ratio1.csv",row.names=F)
```

We try to fit a logistic regression to this cleaned dataset and eliminate the financial variables with large p-value one by one and finally arrived at the following model:

```
> summary(glm(HSI~CFTP+ln_MV+BTME,data=d3,binomial))

Call:
glm(formula = HSI ~ CFTP + ln_MV + BTME, family = binomial, data = d3)

Deviance Residuals:
       Min          1Q      Median          3Q         Max
-2.377e+00  -1.943e-04  -8.005e-06  -3.054e-07   1.738e+00

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -69.9309    21.3821  -3.271  0.00107 **
CFTP         -3.0376     1.2178  -2.494  0.01262 *
ln_MV         7.2561     2.2284   3.256  0.00113 **
BTME          1.3222     0.6418   2.060  0.03940 *
---
Signif. codes:  0 `***' 0.001 `**' 0.01 `*' 0.05 `.' 0.1 ` ' 1
```

Finally, we save the output and produce the classification table.

```
> lreg<-glm(HSI~CFTP+ln_MV+BTME,data=d3,binomial)   # save the output
> pr<-(lreg$fit>0.5)                                 # prediction
> table(pr,d3$HSI)                                 # classification table

pr       0   1
FALSE 624   3
TRUE    2  29
```

The correct classification rate is now 653/658=99.24%. Throwing away the outliers actually gives a simpler model and some better classification results.

5.4 Dummy variable in logistic regression

As in regression, there may be categorical variables as the predictors in the logistic regression model. However, the interpretation is similar. In chapter 6 (classification tree), we shall notice that *ln_MV>9.4766* is an important rule to classify *HSI=0* or *1*. Now we create a categorical variable g and try to fit a logistic regression using g and all the interaction terms with other variables, eliminating variables with large p-values one by one.

```
> g<-(d3$ln_MV>9.4766)+1  # create dummy var g=2 if ln_MV>9.4766; g=1
otherwise
>summary(glm(HSI~EY+CFTP+g+DY+BTME+DTE+EY*g+CFTP*g+DY*g+BTME*g+DTE*g,
data=d3, binomial))

Call:
glm(formula = HSI ~ EY + CFTP + g + DY + BTME + DTE + EY * g +
    CFTP * g + DY * g + BTME * g + DTE * g, family = binomial,
    data = d3)
```

```
Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -21.0469     5.6643  -3.716 0.000203 ***
EY           29.0955    38.2277   0.761 0.446592
CFTP         -0.1916     3.8931  -0.049 0.960745
g            15.2686     5.3888   2.833 0.004606 **
DY            1.0979     0.8182   1.342 0.179637
BTME          1.4157     1.5673   0.903 0.366398
DTE          -0.3659     0.8731  -0.419 0.675166
EY:g        -28.8939    38.2187  -0.756 0.449641
CFTP:g        0.1269     3.6225   0.035 0.972054
g:DY         -0.9840     0.7821  -1.258 0.208316
g:BTME       -1.3220     1.4948  -0.884 0.376499
g:DTE         0.2058     0.5665   0.363 0.716429
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Note that *CFTP*g* has the largest p-value(=0.972). We exclude *CFTP* and *CFTP*g* in the next trial.

```
> summary(glm(HSI~EY+g+DY+BTME+DTE+EY*g+DY*g+BTME*g+DTE*g,data=d3,
binomial))

Call:
glm(formula = HSI ~ EY + g + DY + BTME + DTE + EY * g + DY *
    g + BTME * g + DTE * g, family = binomial, data = d3)

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -21.0169     5.5409  -3.793 0.000149 ***
EY           28.6141    27.2787   1.049 0.294200
g            15.2425     5.2592   2.898 0.003752 **
DY            1.0968     0.7752   1.415 0.157089
BTME          1.4089     1.4991   0.940 0.347279
DTE          -0.3558     0.8474  -0.420 0.674536
EY:g        -28.4191    27.2658  -1.042 0.297274
g:DY         -0.9858     0.7396  -1.333 0.182540
g:BTME       -1.3146     1.4225  -0.924 0.355404
g:DTE         0.2015     0.5513   0.366 0.714692
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

*DTE*g* has the largest p-value, therefore we exclude *DTE* and *DTE*g* in the next trial and we keep on excluding variables with large p-values. After several stages, we finally arrive at the following model:

```
> summary(glm(HSI~g+DY+DY*g,data=d3,binomial))

Call:
glm(formula = HSI ~ g + DY + DY * g, family = binomial, data = d3)
```

```
Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -18.6304     3.4117  -5.461 4.74e-08 ***
g            12.9236     3.1251   4.135 3.54e-05 ***
DY            1.4036     0.6485   2.165   0.0304 *
g:DY         -1.2860     0.6082  -2.115   0.0345 *
---
Signif. codes:  0 `***' 0.001 `**' 0.01 `*' 0.05 `.' 0.1 ` ' 1
```

Note that all the variables in the model are significant. This model is simpler than the model in the previous section and yet the classification result is good.

```
> lreg1<-glm(HSI~g+DY+DY*g,data=d3,binomial)
> pr<-(lreg1$fit>0.5)
> table(pr,d3$HSI)
  pr      0   1
  FALSE 624   3
  TRUE    2  29
```

The interpretation of this threshold-type logistic model is as follow:

$$\ln[\pi/(1-\pi)] = -18.6304 + 12.9236g + 1.4036DY - 1.286(g*DY)$$

$$\Leftrightarrow \begin{cases} \ln[\pi/(1-\pi)] = (-18.6304+12.936)+(1.4036-1.286)DY, & \text{for } g=1 \\ \qquad = -5.6944+0.1176DY & \\ & \\ \ln[\pi/(1-\pi)] = (-18.6304+12.9236\times 2)+(1.4036-1.286\times 2)DY, & \text{for } g=2 \\ \qquad = 7.2416-1.1684DY & \end{cases}$$

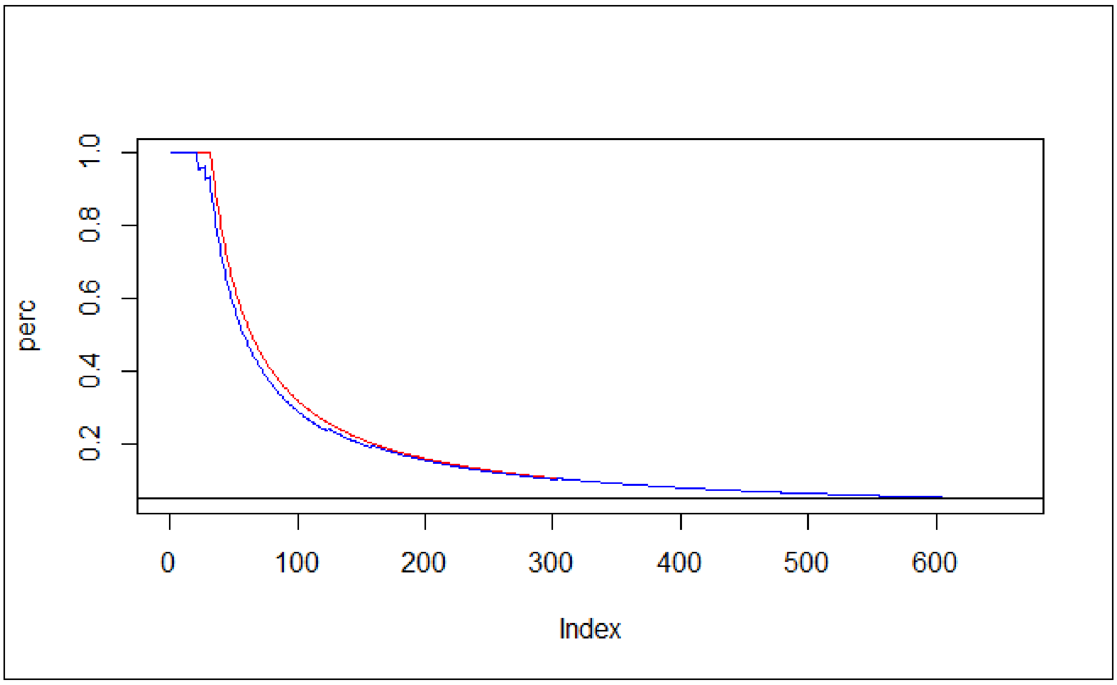
where π = probability of HSI=1. Note that for g=2, the intercept is much bigger than that of g=1, hence the probability of HSI=1 is higher.

5.5 Lift chart

The fitted value in the logistic regression output (*lreg$fit*) gives the probability of "success", i.e., *Prob(Y=1|x)*. Now if you sort the variable *Y* (*d3$HSI* in our example) according to the decreasing order of *lreg$fit*, we would expect that most of the records with *Y=1* will appear on the top of the list, while due to inaccuracy of the model, some of the records with $Y=1$ will appear low on the list. We can actually compute the cumulative percentage of Y=1 and plot them against the index. This plot is called the **lift chart**. It measures how well this logistic regression model predicts.

Suppose that the original data sample are randomly arranged according to index the $i=1,2,...,n$. We sort the data in descending order of $\pi_{i'} = lreg\$fit$, which we label this new index as $i'=1,2,..,n$. The following R commands will produce this lift chart of i' vs $f_2(i') = \frac{1}{i'}\sum_{j=1}^{i'} HSI_j$:

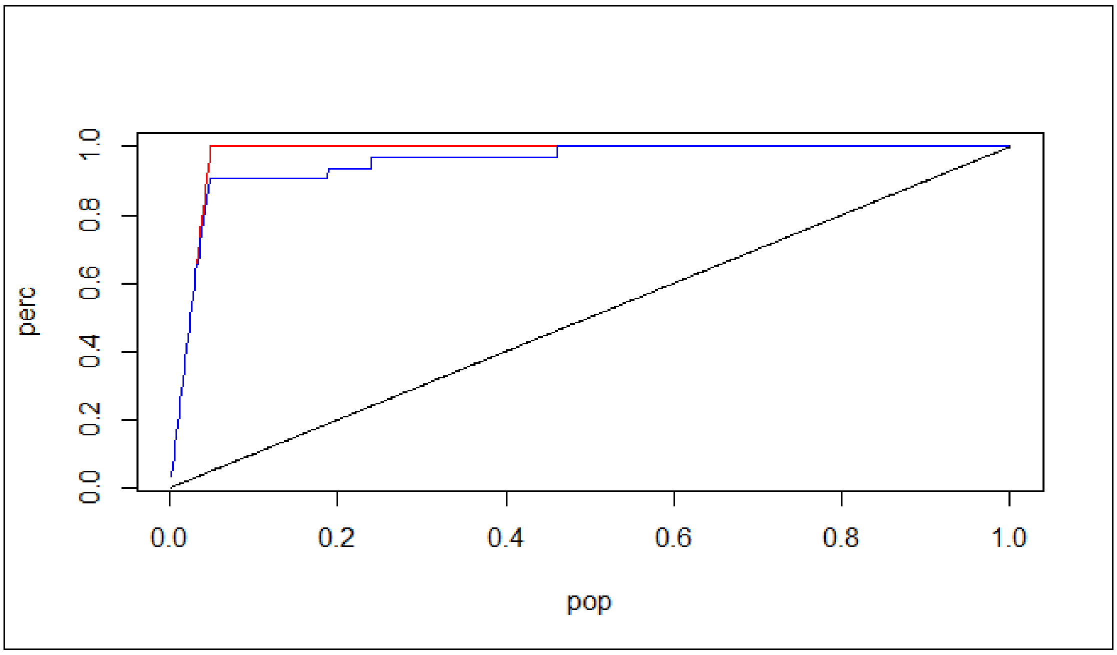
```
ysort<-d3$HSI[order(lreg$fit,decreasing=T)]     # sort y according to lreg$fit
n<-length(ysort)                                # get length of ysort
perc1<-cumsum(ysort)/(1:n)                      # compute cumulative percentage
plot(perc1,type="l", col=' blue' )                   # plot perc with line type
abline(h=sum(d3$HSI)/n)                    # add the baseline
yideal <- c(rep(1,sum(d3$HSI)),rep(0,length(d3$HSI)-sum(d3$HSI)))  # the ideal case
perc_ideal <- cumsum(yideal)/(1:n)                   # compute cumulative percentage
of ideal case
lines(perc_ideal, type="l", col="red")          # plot the ideal case in red line
```



Or we can plot this cumulative percentage of success with the cumulative proportion,

i' vs $f_1(i') = \frac{\sum_{j=1}^{i'} HSI_j}{\sum_{j=1}^{n} HSI_j}$:

```
perc2<-cumsum(ysort)/sum(ysort)            # cumulative perc. of success
pop<-(1:n)/n                          # x-coordinate
plot(pop,perc2,type="l")                   # plot
lines(pop,pop)                        # add the reference line
perc_ideal2 <- cumsum(yideal)/sum(yideal)  # cumulative perc. of success for ideal
case
lines(pop,perc_ideal2, type="l",col="red") # plot the ideal case in red line
```



In both plots, we also plot the ideal case (in red lines), where all *Y=1* records are ranked higher than *Y=0* records in the sorted index i'. Deviations of the actual cases (blue lines) from the ideal cases (red lines) reflect inaccuracy of the model.

5.6 Model Selection

In ordinary linear regression, binary logistic regression or multinomial logit model, selecting useful and important variables included in the model is an important issue. As in section 5.3, start with all the variables included in the model and eliminate variable with largest p-value one-by-one is called the **backward elimination**. One commonly used method is the **stepwise** regression procedure. In each step, variable with largest p-value will be removed from the model and variable with smallest p-value will be entered in the model. There are other alternative for model selection. An **Information Criterion (*IC*)** is also computed in each step to help us to choose the suitable model, where $IC = -2\log L + k * edf$. *Log L* is the log-likelihood function value; *edf* is the effective degrees of freedom (i.e., the number of free parameters in the model); *k=2* for **Akaike Information Criterion (*AIC*)** and *k=log(n)* for **Bayesian Information Criterion (*BIC*)**. In general, as more parameters are included in the model, the log likelihood function $\log L$ value should keep on increasing, and the model become too complex, which over-fit the data. Information criterion penalizes such excessive inclusion of parameters, so that the optimal model with reasonable accuracy and model complexity is indicated at the IC minimum.

Let us try *fin-ratio.csv* and choose the model with smallest *AIC*. The *step()* function implements the stepwise regression. Let use the “fin-ratio1.csv” again to illustrate this *step()* function.

```
> d<-read.csv("fin-ratio1.csv")          # read in data
> lreg<-glm(HSI~.,data=d,binomial)       # save the logistic reg
> step(lreg)                             # perform stepwise selection
Start:  AIC=36.47
HSI ~ EY + CFTP + ln_MV + DY + BTME + DTE

        Df Deviance     AIC
- DTE    1   22.495  34.495
- DY     1   22.769  34.769
- EY     1   22.822  34.822
<none>       22.468  36.468
- BTME   1   27.628  39.628
- CFTP   1   30.586  42.586
- ln_MV  1  245.018 257.018

Step:  AIC=31.09
HSI ~ CFTP + ln_MV + BTME

        Df Deviance     AIC
<none>       23.087  31.087
- BTME   1   28.051  34.051
- CFTP   1   33.623  39.623
- ln_MV  1  246.700 252.700

Call:  glm(formula = HSI ~ CFTP + ln_MV + BTME, family = binomial, data = d)

Coefficients:
(Intercept)         CFTP        ln_MV         BTME
    -69.931       -3.038        7.256        1.322
```

Note that the final model (with smallest AIC) is *HSI~CFTP+ln_MV+BTME* is same as the model in Section 5.3 selected by the backward elimination

5.7 Training and Testing dataset

In data mining, the number of observation is huge so that we can divide or sample the whole dataset into two exclusive parts: training dataset and testing datasets. The training dataset is used to build the statistical model (such as logistic regression model or other model for classification) and the testing dataset is used to assess the accuracy of the model. Let us illustrate this by randomly select *560* observations ($\approx 85\%$) from *fin-ratio1.csv* as training dataset to build a logistic regression model and use the rest as the testing dataset to test the validity of the model.

```
n<-nrow(d)                    # no. of obs
set.seed(12345)               # set random seed
id<-sample(1:n,size=560)      # generate id
d1<-d[id,]                    # training dataset
d2<-d[-id,]                   # testing dataset
dim(d1)                       # display dim of d1 and d2
[1] 560   7
dim(d2)
[1] 98  7

lreg<-glm(HSI~.,data=d1,binomial)      # save logistic reg
step(lreg)                             # perform stepwise selection

lreg<-glm(HSI~CFTP+ln_MV+BTME,data=d1,binomial) # save the selected model
pr<-(lreg$fit>0.5)+1                   # pred
table(pr,d1$HSI)                       # classification table of d1
pr    0   1
  1 530   3
  2   2  25

c<-predict(lreg,d2)                    # return x'b
pi<-exp(c)/(1+exp(c))                  # estimate prob. of success
pr<-(pi>0.5)+1                         # prediction
table(pr,d2$HSI)                       # classification table of d2
pr   0  1
  1 94  1
  2  0  3
```

The logistic regression model can be used to make prediction using the training dataset d1 or the testing dataset d2. When using d1, it is called **in-sample** assessment while using d2, it is called the **out-sample** assessment. From the output, the training error rate is 5/560=0.009 while the testing error rate is 1/98=0.01. Usually, the testing error rate is higher than training error rate. However, training error rate is over optimistic and the testing error rate is more realistic to reflect the true performance of the model.

5.8 Measure of Accuracy

When measuring the accuracy of a classification model, the overall accuracy computed from the classification table is not the only measure and can be misleading in some situations. We can have the **precision, recall** and the **F1 score** to measure the accuracy of the model. In medical diagnostic test, we call the result of the test is positive if the test predicts that the subject have certain disease. For example:

Test \ Truth	**Disease**	**No Disease**	**row sum**
Positive	*True Positive (TP)*	*False Positive (FP)*	*TP+FP*
Negative	*False Negative (FN)*	*True Negative (TN)*	*FN+TN*
column sum	*TP+FN*	*FP+TN*	Total

The precision, recall and F1 score of the test is defined as follows:

Precision =TP/(TP+FP)   percentage correct among positive results, *Recall = TP/(TP+FN)* percentage correct among true disease cases.

$$F1 = \frac{2}{1/Precision + 1/Recall} = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

This F1 score is the harmonic mean of precision and recall. The advantage of F1 is illustrated by the following example. Consider a diagnostic test of a rare disease:

Test \ Truth	**Disease**	**No Disease**	**row sum**
Positive	2	2	4
Negative	1	195	196
column sum	3	197	200

The overall accuracy of the test is (2+195)/200=98.5%. The overall accuracy is high only due to the disease is rare; most of the correct predictions are come from true negative. However, the precision of the test is only 2/4=50% and the recall of the test is 2/3=66.7% and the F1 score is 4/7=57.14%.

5.9 Decision Threshold

The binary logistic regression and the multinomial logit model can be used to estimate the probability of default as well as classification in credit rating system. Making wrong classification is inevitable. Similar to hypothesis testing, there are two types of error. For example in loan application:

Decision \ Borrower	Non-default (actual Y=0)	Default (actual Y=1)
Accept (predict Y=0)	correct	error (cost=*$c2*)
Reject (predict Y=1)	error (cost=*$c1*)	correct

In practice, these two costs of misclassification errors are not equal. We can easily account for these asymmetric costs by considering the expected loss of each decision.

Let $p(x) = \Pr\{Default \mid x\}$, $1 - p(x) = \Pr\{Non-default \mid x\}$.

Expected loss of rejecting a non-default borrower = $c1[1 - p(x)]$,

Expected loss of accepting a default borrower = $c2\ p(x)$.

Hence, we reject an application if $c1[1 - p(x)] < c2\ p(x) \Rightarrow p(x) > c1/(c1 + c2) = c$.

This c is called the **decision threshold**. When the cost $$c$1 equals to $$c$2, then c=1/2.

5.10 Multinomial Logit (MNL)

The binary logistic regression model can be extended to the case where the response *Y* is a categorical variable with more than 2 groups. Suppose *Y* has *k* categories and we modeled the probabilities as follows:

$$\Pr(Y_i = 1 \mid x_i\} = \exp(x_i'\beta_1)/\xi \quad ,\ldots, \quad \Pr(Y_i = k \mid x_i\} = \exp(x_i'\beta_k)/\xi$$

where $\xi = \exp(x_i'\beta_1) + \ldots + \exp(x_i'\beta_k)$.

However, the $\beta_1,...,\beta_k$ in the above model cannot be uniquely estimated unless one of these betas are set to zero. It does not matter which one is set to zero. Suppose we set β_1 to zero, then the model becomes:

$\Pr(Y_i = 1 | x_i) = 1/\xi, \Pr(Y_i = 2 | x_i) = \exp(x_i'\beta_2)/\xi, ..., \Pr(Y_i = k | x_i) = \exp(x_i'\beta_k)/\xi,$

where $\xi = 1 + \exp(x_i'\beta_2) + \ldots + \exp(x_i'\beta_k)$. (5.5)

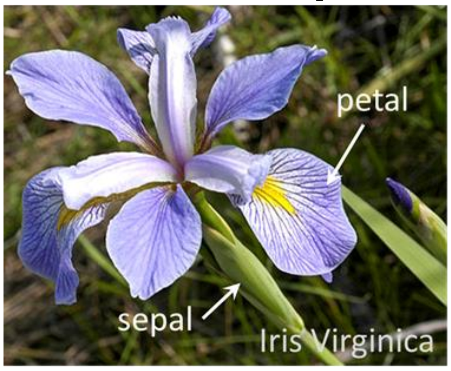
The group *Y=1* serve as the baseline in this MNL model. Using the likelihood function

$$L(\beta_2, \beta_3, ..., \beta_k | x_i) = \prod_{i=1}^{n} \{\xi^{-1} \cdot \prod_{j=2}^{k} [\exp(x_i'\beta_j)^{I(Y_i - j)}]\}, \quad (5.6)$$

where $I(Y_i = j) = \begin{cases} 0 & \text{if } Y_i \neq j \\ 1 & \text{if } Y_i = j \end{cases}$, $j = 2,...,k$

the model parameters $\beta_2, \beta_3, ..., \beta_k$ can be estimated from maximum likelihood estimation.

R has a built-in function *multinom()* in the *nnet* (stands for neural network) library to estimate the parameters in this MNL model. Let us illustrate this by the famous Fisher's Iris flower data. The data set iris.dat contains the information about Iris flower in 5 columns. The first four columns are the measurement of Sepal length, Sepal width, Petal length and Petal width respectively. The last column indicates three different species of the Iris flower (1= *setosa*, *2=versicolor* 3= *virginica*). There are 150 observations and 50 observations from each species.



```
> d<-read.csv("iris.csv")              # read in data
> names(d)                             # display names
[1] "Sepal_len" "Sepal_wid" "Petal_len" "Petal_wid" "Species"
> library(nnet)                        # load nnet library
> mnl<-multinom(Species~.,data=d)      # perform MNL
> summary(mnl)                         # display summary of MNL
Call:
multinom(formula = Species ~ ., data = d)

Coefficients:                          # b2 and b3
  (Intercept) Sepal_len Sepal_wid Petal_len Petal_wid
2    18.69037 -5.458424  -8.70740  14.24477 -3.097684
3   -23.83628 -7.923634 -15.37077  23.65978 15.135301

Std. Errors:
  (Intercept) Sepal_len Sepal_wid Petal_len Petal_wid
2    34.97116  89.89215  157.0415  60.19170  45.48852
3    35.76649  89.91153  157.1196  60.46753  45.93406
```

We can estimate $\Pr\{Y_i = k \mid x_i\}$ according to (5.5) and predict Y belongs to group j if $\Pr\{Y_i = j \mid x_i\}$ is maximum. R has a built-in function *predict()* to obtain these predictions.

```
> pred<-predict(mnl)          # prediction
> table(pred,d$Species)       # tabulate pred vs true species

pred 1  2  3
   1 50  0  0
   2  0 49  1
   3  0  1 49
```

From the above output, there are only two misclassification cases and the misclassification rate is 2/150=1.33%.

5.11 Logistic regression using EXCEL

In statistical package, the MLE of logistic regression is obtained by maximizing the log-likelihood function in page 5 by numerical methods. However, it is possible to obtain MLE by the *solver* function in EXCEL.

1. Setup the six financial ratios from “fin-ratio1.csv” in columns A to G.
2. Set the beta in M2:M5 to zero as initial values.
3. Input the formula `=$M$2+$M$3*B2+$M$4*C2+$M$5*E2` in I2 and copy it down to I659. Note that we only use CFTP, ln_MV and BTME as input.
4. Input the formula `=EXP(I2)/(1+EXP(I2))` in J2 and copy it down to J659.
5. Input the formula `=G2*LN(J2)+(1-G2)*LN(1-J2)` in K2 and copy it down to K659.
6. Enter `=SUM(K2:K659)` in M9. Using solver to maximize this cell with M2:M5 as variable cells.
7. Enter `=0+(J2>0.5)` in H2 and copy it down to H659. This is the prediction based

on the logistic regression.

8. Finally we can create the classification table by entering the following in

```
N12: =COUNTIFS(G2:G659,"=0",H2:H659,"=0")
N13: =COUNTIFS(G2:G659,"=0",H2:H659,"=1")
O12: =COUNTIFS(G2:G659,"=1",H2:H659,"=0")
O13: =COUNTIFS(G2:G659,"=1",H2:H659,"=1")
```

Reference:

Chapter 11 of Applied Multivariate Statistical Analysis, 5th ed., Richard Johnson and Dean Wichern, Prentice Hall.