Logit模型和Probit模型

宋歌 2015080086 数52

5/15/2018

1 实验目的

给定数据集pgaBinary.txt，利用logit和probit模型进行建模和分析，并对两个模型的回归结果进行比较。

2 实验原理

- 对于一般的线性回归模型，我们的充分预测变量通常具有$E(Y|X=x)=x^T\beta$的线性形式；
- 而当数据集中的响应变量为二值变量时，我们依然希望$E(Y|X=x)=P(Y=1|X=x)=\mu(x)$具有某种与$x$的线性关系或者至少是$x^T\beta$ 的函数。从而我们可以选取适当的连续可微函数g，对于取值在$[0,1]$上的$\mu(x)$进行某种变换，使得变换后的函数$g(\mu(x))=x^T\beta$；
- 由$\mu(x)\in[0,1], g(\mu(x))\in\mathbb{R}$，我们容易想到两个常见的将$\mathbb{R}$映到$[0,1]$的函数：Sigmoid函数$\frac{1}{1+\exp(-x)}$ 和标准正态概率分布函数Φ。将$g^{-1}(x^T\beta)=\mu(x)$分别取为这两个函数，即

$$\mu(x)=P(Y=1|X=x)=\frac{1}{1+\exp(-x^T\beta)}$$
$$\mu(x)=P(Y=1|X=x)=\Phi(x^T\beta)$$

就得到了我们常用的两个模型：Logit模型和Probit模型：

$$\text{Logistic regression model}:\quad \log\frac{P(Y=1|X=x)}{1-P(Y=1|X=x)}=x^T\beta$$
$$\text{Probit regression model}:\quad \Phi^{-1}(P(Y=1|X=x))=x^T\beta$$

3 实验过程及结果讨论

3.1 读取并描述数据

```
#读取数据
dat = read.table('pgaBinary.txt', header = TRUE)

#数据维数
dim(dat)
```

```
## [1] 400   4
```

```
#数据中的变量名
names(dat)
```

```
## [1] "admit" "gre"   "gpa"   "rank"
```

```
#查看数据的前6条记录
head(dat)
```

```
##   admit gre  gpa rank
## 1     0 380 3.61    3
## 2     1 660 3.67    3
## 3     1 800 4.00    1
## 4     1 640 3.19    4
## 5     0 520 2.93    4
## 6     1 760 3.00    2
```

```
#总结数据信息
summary(dat)
```

```
##      admit             gre             gpa             rank      
##  Min.   :0.0000   Min.   :220.0   Min.   :2.260   Min.   :1.000  
##  1st Qu.:0.0000   1st Qu.:520.0   1st Qu.:3.130   1st Qu.:2.000  
##  Median :0.0000   Median :580.0   Median :3.395   Median :2.000  
##  Mean   :0.3175   Mean   :587.7   Mean   :3.390   Mean   :2.485  
##  3rd Qu.:1.0000   3rd Qu.:660.0   3rd Qu.:3.670   3rd Qu.:3.000  
##  Max.   :1.0000   Max.   :800.0   Max.   :4.000   Max.   :4.000  
```

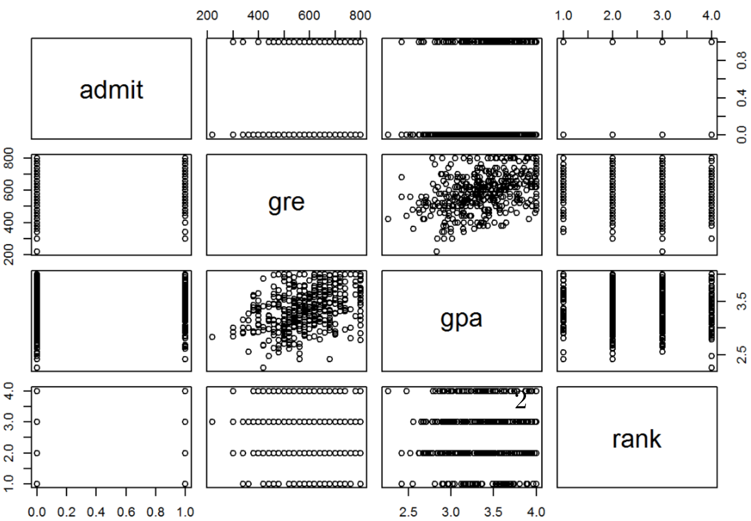
```
#整理输出数据的方差
sapply(dat,sd)
```

```
##       admit         gre         gpa        rank 
##   0.4660867 115.5165364   0.3805668   0.9444602 
```

```
#列联表输出数据中admit和rank的频数
xtabs(~admit + rank, data = dat);
```

```
##      rank
## admit  1  2  3  4
##     0 28 97 93 55
##     1 33 54 28 12
```

```
#画数据散点图
plot(dat)
```



- 我们看到，该数据集中有四个变量admit，gre，gpa，rank，且每个变量长度均为400；
- summary中列出了四个变量各自的最值、均值、中位数、1/4和3/4分位数；sapply中列出了四个变量各自的方差；
- 最后一张图画出了数据集中各个变量之间的散点图。易看出admit和rank是高度离散的分类数据。

3.2 用Logit模型分析数据

```r
#将数据集中的rank数据类型转换为factor因子类型
dat$rank = factor(dat$rank)

#用glm中的logistic回归模型对数据集进行回归，响应变量为admit，预测变量为gre,gpa,rank
mylogit = glm(admit ~ gre + gpa + rank, data = dat, family = "binomial")

#查看回归结果
summary(mylogit)
```

```
##
## Call:
## glm(formula = admit ~ gre + gpa + rank, family = "binomial",
##     data = dat)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6268  -0.8662  -0.6388   1.1490   2.0790
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -3.989979   1.139951  -3.500 0.000465 ***
## gre          0.002264   0.001094   2.070 0.038465 *
## gpa          0.804038   0.331819   2.423 0.015388 *
## rank2       -0.675443   0.316490  -2.134 0.032829 *
## rank3       -1.340204   0.345306  -3.881 0.000104 ***
## rank4       -1.551464   0.417832  -3.713 0.000205 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 499.98  on 399  degrees of freedom
## Residual deviance: 458.52  on 394  degrees of freedom
## AIC: 470.52
##
## Number of Fisher Scoring iterations: 4
```

3.2.1 模型建立

- 在该模型中，响应变量$Y = \text{admit}$，预测变量$X = (\text{gre}, \text{gpa}, \text{rank})$.
- 其中变量rank被化为了因子类型的变量，在回归的时候被视为

$$\text{rank} = \text{rank1} + \text{rank2} + \text{rank3} + \text{rank4}$$

$$\dim(\text{rank}j) = 400, \quad j = 1, 2, 3, 4$$

$$\text{rank}j[i] = \begin{cases} 1 & rank[i] = j \\ 0 & else \end{cases}$$

- 现象：观察到以上回归结果中，没有rank1变量。解释：R检测到X矩阵非满秩，故删去了变量rank1，实质上进行的回归是

$$\text{admit} \sim \text{gre} + \text{gpa} + \text{rank2} + \text{rank3} + \text{rank4}$$

检验：运行如下代码

```r
c <- rep(1,400)

#把拆分出来的rank1234与常数项合并成一个矩阵
X <- data.frame(c,rank1,rank2,rank3,rank4)
qr(X)$rank
```

```
## [1] 4
```

```r
#把rank1去掉
X1 <- data.frame(c,rank2,rank3,rank4)
qr(X1)$rank
```

```
## [1] 4
```

```r
#把rank拆成rank234
r1 = glm(admit ~ gre + gpa + rank2 + rank3 + rank4, data = dat, family = "binomial")
summary(r1)
```

```
##
## Call:
## glm(formula = admit ~ gre + gpa + rank2 + rank3 + rank4, family = "binomial",
##     data = dat)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6268  -0.8662  -0.6388   1.1490   2.0790
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -3.989979   1.139951  -3.500 0.000465 ***
## gre          0.002264   0.001094   2.070 0.038465 *
## gpa          0.804038   0.331819   2.423 0.015388 *
## rank2       -0.675443   0.316490  -2.134 0.032829 *
## rank3       -1.340204   0.345306  -3.881 0.000104 ***
## rank4       -1.551464   0.417832  -3.713 0.000205 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 499.98  on 399  degrees of freedom
## Residual deviance: 458.52  on 394  degrees of freedom
## AIC: 470.52
##
## Number of Fisher Scoring iterations: 4
```

可见R在回归时会默认预测变量中有常数项1，若将rank拆为rank1 + rank2 + rank3 + rank4，则由这四个变量和常数项组成的预测变量的子阵X不是满秩的，即存在共线性性，即预测变量之间有高度的相关性。又去掉rank1后所得的X1矩阵满秩，故在这种情况下R会去掉变量rank1，对剩余的rank234进行回归。从以上运行结果可以看到，summary(r1)与summary(mylogit)的结果完全一样，从而实质上进行的回归是admit ∼ gre + gpa + rank2 + rank3 + rank4.

- 进一步探讨：如果不自动增加常数项的话，rank矩阵是满秩的。那么如果强制进行无常数项的回归呢？

```r
#强制进行无常数项的回归
r2 = glm(admit ~ 0 + gre + gpa + rank, data = dat, family = "binomial")
summary(r2)
```

```
##
## Call:
## glm(formula = admit ~ 0 + gre + gpa + rank, family = "binomial",
##     data = dat)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6268  -0.8662  -0.6388   1.1490   2.0790
##
## Coefficients:
##        Estimate Std. Error z value Pr(>|z|)
## gre    0.002264   0.001094   2.070 0.038465 *
## gpa    0.804038   0.331819   2.423 0.015388 *
## rank1 -3.989979   1.139951  -3.500 0.000465 ***
## rank2 -4.665422   1.109370  -4.205 2.61e-05 ***
## rank3 -5.330183   1.149538  -4.637 3.54e-06 ***
## rank4 -5.541443   1.138072  -4.869 1.12e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 554.52  on 400  degrees of freedom
## Residual deviance: 458.52  on 394  degrees of freedom
## AIC: 470.52
##
## Number of Fisher Scoring iterations: 4
```

现象：r2模型中rank1对应mylogit模型的截距项，rank234的系数比起mylogit模型都增加了rank1。
解释：假设

$$\text{在mylogit模型中对rank部分的回归结果为：} y = \tilde{\beta}_1 + \tilde{\beta}_2 rank2 + \tilde{\beta}_3 rank3 + \tilde{\beta}_4 rank4$$
$$\text{在r2模型中对rank部分的回归结果为：} y = \hat{\beta}_1 rank1 + \hat{\beta}_2 rank2 + \hat{\beta}_3 rank3 + \hat{\beta}_4 rank4$$

则当rank$j = 1$时，应有$\hat{\beta}_j$rank$j = \tilde{\beta}_1 + \tilde{\beta}_j$rank$j$

3.2.2 结果分析

- Deviance Residuals
  输出了残差统计量的最值、中位数、第一第三四分位数；
- Coefficients
  - Estimate：由普通最小二乘法计算出来的估计回归系数；
  - Std. Error：估计的回归系数的标准误差；
  - z value：估计的系数Estimate除以标准差Std. Error得到了z统计量，衡量系数的显著性。
  - $Pr(> |z|)$：利用z统计量构造P值，衡量系数的显著性，p值越大，越倾向于拒绝0假设。
- NULL deviance：0假设下的残差平方和，自由度为$n - 1 = 400 - 1 = 399$。
- Residual deviance：模型的残差平方和，自由度为$n - p = 400 - 6 = 394$。其中p为预测变量矩阵的秩。

3.3 评价Logit模型的参数估计

3.3.1 置信区间

```r
#输出参数估计的置信区间
confint(mylogit)
```

```
## Waiting for profiling to be done...
```

```
##                   2.5 %       97.5 %
## (Intercept) -6.2716202334 -1.792547080
## gre          0.0001375921  0.004435874
## gpa          0.1602959439  1.464142727
## rank2       -1.3008888002 -0.056745722
## rank3       -2.0276713127 -0.670372346
## rank4       -2.4000265384 -0.753542605
```

```r
#输出默认基于渐近正态性的置信区间
confint.default(mylogit)
```

```
##                   2.5 %       97.5 %
## (Intercept) -6.2242418514 -1.755716295
## gre          0.0001202298  0.004408622
## gpa          0.1536836760  1.454391423
## rank2       -1.2957512650 -0.055134591
## rank3       -2.0169920597 -0.663415773
## rank4       -2.3703986294 -0.732528724
```

3.3.2 变量的显著性

```r
#对mylogit模型里的参数进行wald显著性检验，其中rank234被联合起来检验
wald.test(b = coef(mylogit), Sigma = vcov(mylogit), Terms = 4:6)
```

```
## Wald test:
## ----------
##
## Chi-squared test:
## X2 = 20.9, df = 3, P(> X2) = 0.00011
```

```r
#对六个变量进行线性组合后检验显著性
l = cbind(0,0,0,1,-1,0)
wald.test(b = coef(mylogit), Sigma = vcov(mylogit), L = l)
```

```
## Wald test:
## ----------
##
## Chi-squared test:
## X2 = 5.5, df = 1, P(> X2) = 0.019
```

```r
exp(coef(mylogit))
```

```
## (Intercept)         gre         gpa       rank2       rank3       rank4
##   0.0185001   1.0022670   2.2345448   0.5089310   0.2617923   0.2119375
```

- 第一组结果，p值远小于0.05，拒绝零假设，变量均很显著；
- 第二组结果，p值小于0.05，倾向于拒绝零假设，变量线性组合rank2 − rank3显著。

3.4 利用Logit模型进行预测

3.4.1 小规模预测

```
#利用dat中gre和gpa的均值，以及rank的四种类别，生成新的X
newdat1 = with(dat, data.frame(gre = mean(gre), gpa = mean(gpa), rank = factor(1:4)))

#在新的X上利用logit模型对响应变量的值进行预测
newdat1$rankP = predict(mylogit, newdat = newdat1, type = "response")

#输出预测结果
newdat1
```

```
##     gre    gpa rank     rankP
## 1 587.7 3.3899    1 0.5166016
## 2 587.7 3.3899    2 0.3522846
## 3 587.7 3.3899    3 0.2186120
## 4 587.7 3.3899    4 0.1846684
```

3.4.2 大规模预测

- 定义被预测变量集

```
#gpa都取均值，gre为200到800之间等间隔的100个数，每种情况都分别取rank1234
newdat2 = with(dat, data.frame(gre = rep(seq(from = 200, to = 800, length.out = 100), 4), gpa = mean(gpa), rank = factor(rep(1:4, eac
h = 100))))

newdat2[1:3,]
```

```
##        gre    gpa rank
## 1 200.0000 3.3899    1
## 2 206.0606 3.3899    1
## 3 212.1212 3.3899    1
```

```
newdat2[101:103,]
```

```
##          gre    gpa rank
## 101 200.0000 3.3899    2
## 102 206.0606 3.3899    2
## 103 212.1212 3.3899    2
```

```
newdat2[201:203,]
```

```
##          gre    gpa rank
## 201 200.0000 3.3899    3
## 202 206.0606 3.3899    3
## 203 212.1212 3.3899    3
```

```
newdat2[301:303,]
```

```
##          gre    gpa rank
## 301 200.0000 3.3899    4
## 302 206.0606 3.3899    4
## 303 212.1212 3.3899    4
```

- 进行预测

```r
#将newdat2和在newdat2上预测得到的(fit, se.fit, residual.scale)横向合成newdat3
newdat3 = cbind(newdat2, predict(mylogit, newdat = newdat2, type = "link", se = TRUE))

#对newdat3中的变量进行如下操作，并合并到newdat3里面
newdat3 = within(newdat3, {PredictedProb = plogis(fit) #Logistic分布函数在预测值上的值
LL = plogis(fit - (1.96 * se.fit)) #95%置信区间的下界
UL = plogis(fit + (1.96 * se.fit)) #95%置信区间的上界
})
head(newdat3)
```

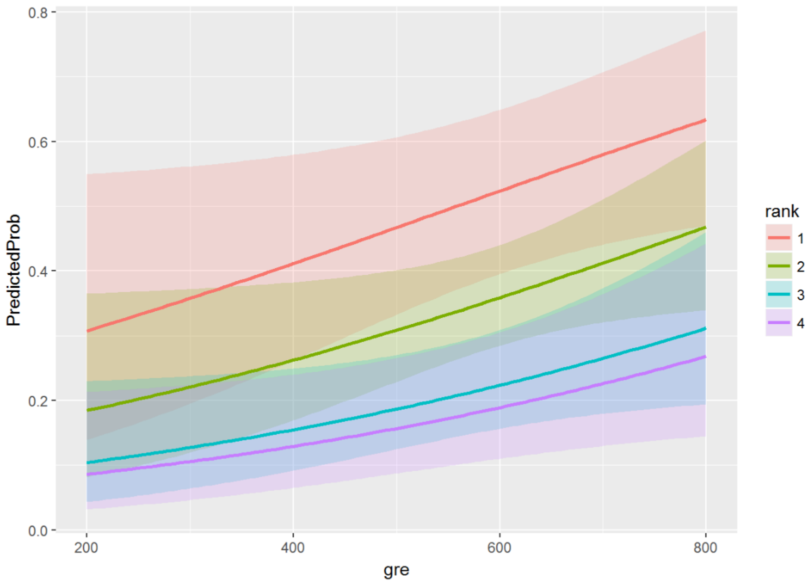
```
##        gre    gpa rank        fit    se.fit residual.scale        UL
## 1 200.0000 3.3899    1 -0.8114870 0.5147714              1 0.5492064
## 2 206.0606 3.3899    1 -0.7977632 0.5090986              1 0.5498513
## 3 212.1212 3.3899    1 -0.7840394 0.5034491              1 0.5505074
## 4 218.1818 3.3899    1 -0.7703156 0.4978239              1 0.5511750
## 5 224.2424 3.3899    1 -0.7565919 0.4922237              1 0.5518545
## 6 230.3030 3.3899    1 -0.7428681 0.4866494              1 0.5525464
##          LL PredictedProb
## 1 0.1393812     0.3075737
## 2 0.1423880     0.3105042
## 3 0.1454429     0.3134499
## 4 0.1485460     0.3164108
## 5 0.1516973     0.3193867
## 6 0.1548966     0.3223773
```

- 预测分析

```r
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 3.4.4
```

```r
#作变量gre和预测响应变量的关系图，在其上添加带表示预测值的上下界，添加线表示对应的rank
ggplot(newdat3, aes(x = gre, y = PredictedProb)) + geom_ribbon(aes(ymin = LL, ymax = UL, fill = rank), alpha = 0.2) + geom_line(aes(c
olour = rank), size = 1)
```



不同颜色的线条代表相应rank下预测值与预测变量gre之间的关系；
不同颜色的色带代表相应rank下预测值的置信区间，可见各个颜色的线条基本处于相应色带区域内；
该图的总体走势意味着，gre分数越高，越倾向于录取(admit = 1)

3.5 其他性质

```r
#0假设下的完全平方和 - 残差平方和 = 回归平方和
with(mylogit, null.deviance - deviance)
```

```
## [1] 41.45903
```

```r
#对数似然
logLik(mylogit)
```

```
## 'log Lik.' -229.2587 (df=6)
```

```r
#0假设下的回归模型
mylogit0 = glm(admit~1, data = dat, family = "binomial")

#模型的方差分析
anova(mylogit0, mylogit)
```

```
## Analysis of Deviance Table
##
## Model 1: admit ~ 1
## Model 2: admit ~ gre + gpa + rank
##   Resid. Df Resid. Dev Df Deviance
## 1       399     499.98
## 2       394     458.52  5   41.459
```

以上结果与summary中的结果一致。

3.6 与Probit模型的比较

- 将回归代码中的link函数由logit改为probit：

$$\text{myprobit} = \text{glm}(\text{admit} \sim \text{gre} + \text{gpa} + \text{rank}, \text{data} = \text{dat}, \text{binomial}(\text{link} = \text{'probit'}))$$

- 将预测值表达式中的plogis改为pnorm：

$$\text{PredictedProb} = \text{pnorm}(\text{fit})$$

3.6.1 预测值与预测变量之间的关系

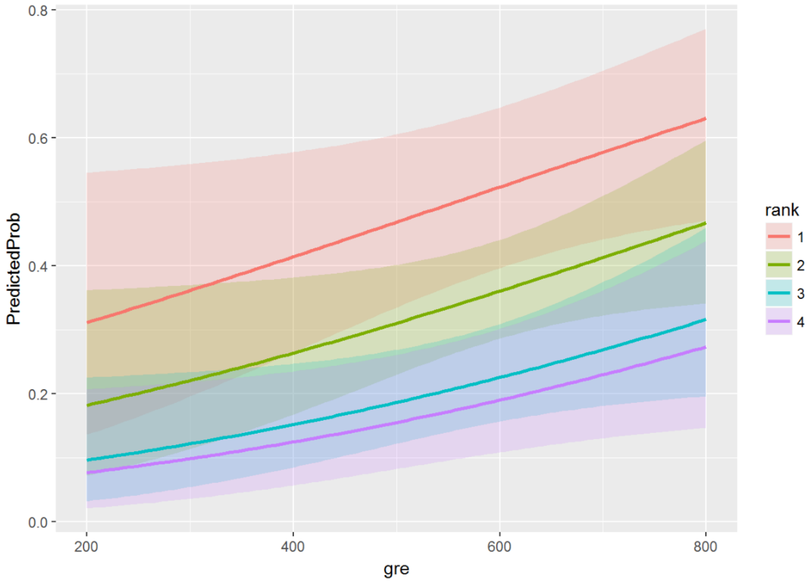
```r
dat$rank = factor(dat$rank)
myprobit = glm(admit ~ gre + gpa + rank, data = dat, binomial(link='probit'))
summary(myprobit)
```

```
##
## Call:
## glm(formula = admit ~ gre + gpa + rank, family = binomial(link = "probit"),
##     data = dat)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6163  -0.8710  -0.6389   1.1560   2.1035
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -2.386836   0.673946  -3.542 0.000398 ***
## gre          0.001376   0.000650   2.116 0.034329 *
## gpa          0.477730   0.197197   2.423 0.015410 *
## rank2       -0.415399   0.194977  -2.131 0.033130 *
## rank3       -0.812138   0.208358  -3.898 9.71e-05 ***
## rank4       -0.935899   0.245272  -3.816 0.000136 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 499.98  on 399  degrees of freedom
## Residual deviance: 458.41  on 394  degrees of freedom
## AIC: 470.41
##
## Number of Fisher Scoring iterations: 4
```

```r
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 3.4.4
```

```r
ggplot(newdat3, aes(x = gre, y = PredictedProb)) + geom_ribbon(aes(ymin = LL, ymax = UL, fill = rank), alpha = 0.2) + geom_line(aes(colour = rank), size = 1)
```



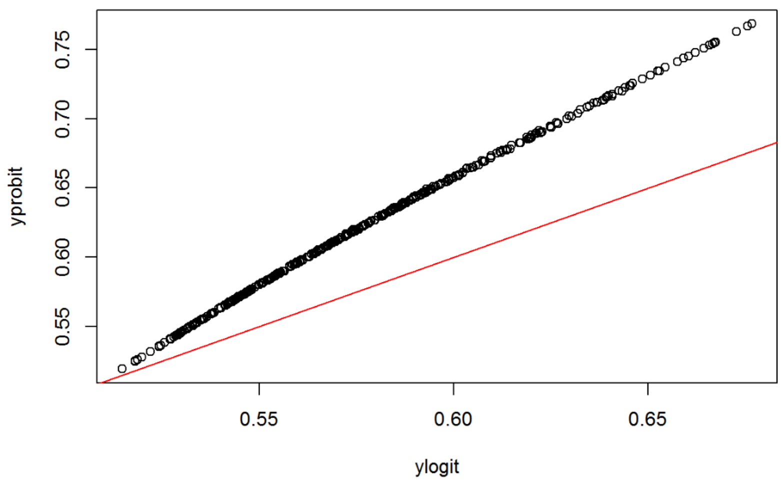
可见在两个模型下，预测值与变量gre之间的关系类似。

3.6.2 拟合值

```
library(aod)
```

```
## Warning: package 'aod' was built under R version 3.4.4
```

```
dat = read.table('pgaBinary.txt', header = TRUE)

dat$rank = factor(dat$rank)

mylogit = glm(admit ~ gre + gpa + rank, data = dat, family = "binomial")

myprobit = glm(admit ~ gre + gpa + rank, data = dat, binomial(link='probit'))

ylogit = plogis(mylogit$fitted.values)

yprobit = pnorm(myprobit$fitted.values)

plot(ylogit,yprobit)
abline(0,1,col = 'red')
```



可见对于该数据集，Logit模型和Probit模型的回归结果并不类似。