회귀분석 및 실습 HW14

서울대학교 통계학과 2017-11362 박건도

2021년 06월 04일

IRIS data

```
# data processing
iris_df <- iris
iris_df[,"Species"] <- as.factor(ifelse(iris_df[, "Species"] == "setosa", 1, 0))

#split train - test set
idx <- 1:150 %in% sample.int(150, 105) # 70:30
iris_train <- iris_df[idx, ]
iris_test <- iris_df[!idx, ]

# logistic regression
result <- glm(Species ~ ., data = iris_train, family = "binomial",
              control = glm.control(maxit = 30))
```

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

```
summary(result)
```

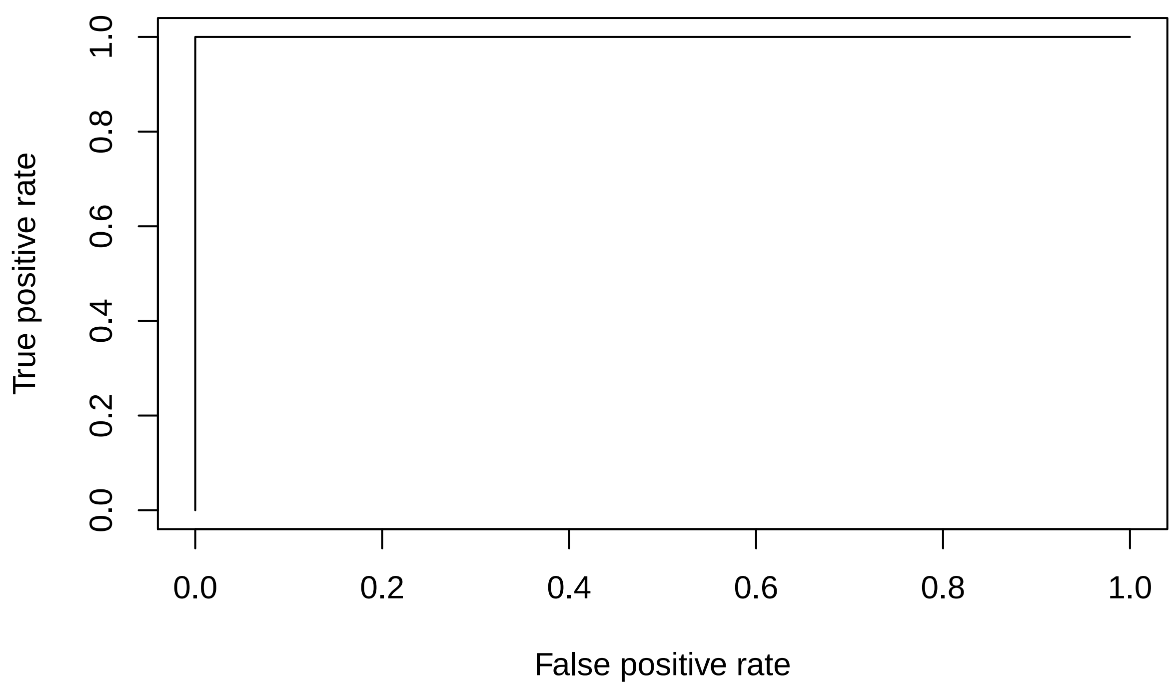
```
##
## Call:
## glm(formula = Species ~ ., family = "binomial", data = iris_train,
##     control = glm.control(maxit = 30))
##
## Deviance Residuals:
##       Min         1Q     Median         3Q        Max
## -1.08e-05  -2.11e-08  -2.11e-08   2.11e-08   9.43e-06
##
## Coefficients:
##                  Estimate Std. Error z value Pr(>|z|)
```

```
## (Intercept)     -45.564 973906.559       0        1
## Sepal.Length     20.536 355984.894       0        1
## Sepal.Width       6.034 229839.883       0        1
## Petal.Length    -21.199 342109.766       0        1
## Petal.Width     -31.115 528120.846       0        1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1.3367e+02  on 104  degrees of freedom
## Residual deviance: 2.7413e-10  on 100  degrees of freedom
## AIC: 10
##
## Number of Fisher Scoring iterations: 27
```

```
# train accuracy with cutoff = 0.5
prob <- predict(result, iris_train, type = "response")
pred <- prediction(prob, iris_train$Species)
perf <- performance(pred, measure = "acc")
perf@y.values[[1]][max(which(perf@x.values[[1]] >= 0.5))]
```

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

```
# train auc
perf <- performance(pred, measure = "tpr", x.measure = "fpr")
plot(perf)
```



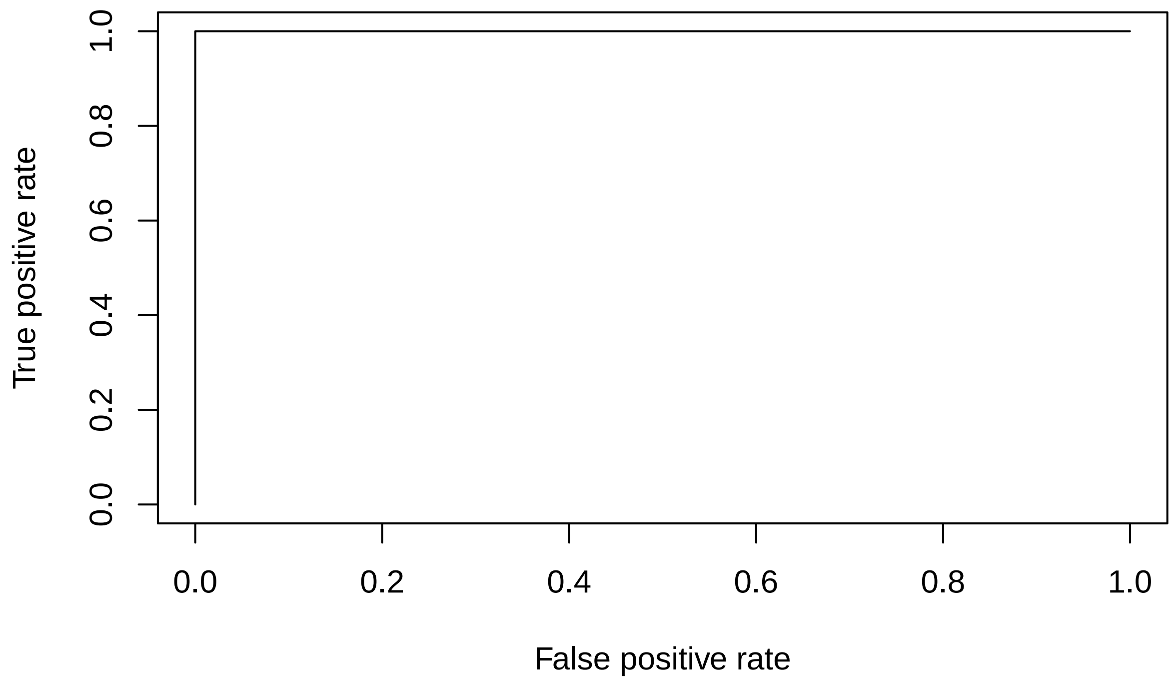
```r
performance(pred, measure = "auc")@y.values[[1]]
```

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

```r
# test accuracy
prob <- predict(result, iris_test, type = "response")
pred <- prediction(prob, iris_test$Species)
perf <- performance(pred, measure = "acc")
perf@y.values[[1]][max(which(perf@x.values[[1]] >= 0.5))]
```

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

```r
# test auc
perf <- performance(pred, measure = "tpr", x.measure = "fpr")
plot(perf)
```




```r
performance(pred, measure = "auc")@y.values[[1]]
```

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

2. num of Awards

```r
# data processing
p <- read.csv("https://stats.idre.ucla.edu/stat/data/poisson_sim.csv")
p <- within(p, {
  prog <- factor(prog, levels=1:3, labels=c("General", "Academic", "Vocational"))
```

```r
  id <- factor(id)
})

# split train - test set
idx <- 1:200 %in% sample.int(200, 140) # 70 : 30
p_train <- p[idx,]
p_test <- p[!idx,]

# poisson regression
m1 <- glm <- glm(num_awards ~ prog + math, family = "poisson", data = p_train)
summary(m1)
```

```
##
## Call:
## glm(formula = num_awards ~ prog + math, family = "poisson", data = p_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.1284  -0.8367  -0.5178   0.2979   2.3711
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)    -4.87485    0.75991  -6.415 1.41e-10 ***
## progAcademic    1.40623    0.43393   3.241  0.00119 **
## progVocational  0.54371    0.54001   1.007  0.31401
## math            0.05953    0.01193   4.989 6.06e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 197.82  on 139  degrees of freedom
## Residual deviance: 128.06  on 136  degrees of freedom
## AIC: 270.32
##
## Number of Fisher Scoring iterations: 5
```

```r
# train accuracy with cutoff = 0.5
prob <- predict(m1, p_train, type = "response")
sum(round(prob) == p_train$num_awards) / length(prob)
```

```
## [1] 0.5857143
```

```r
# train MSE
sum((prob - p_train$num_awards)^2) / length(prob)
```

```
## [1] 0.7393426
```

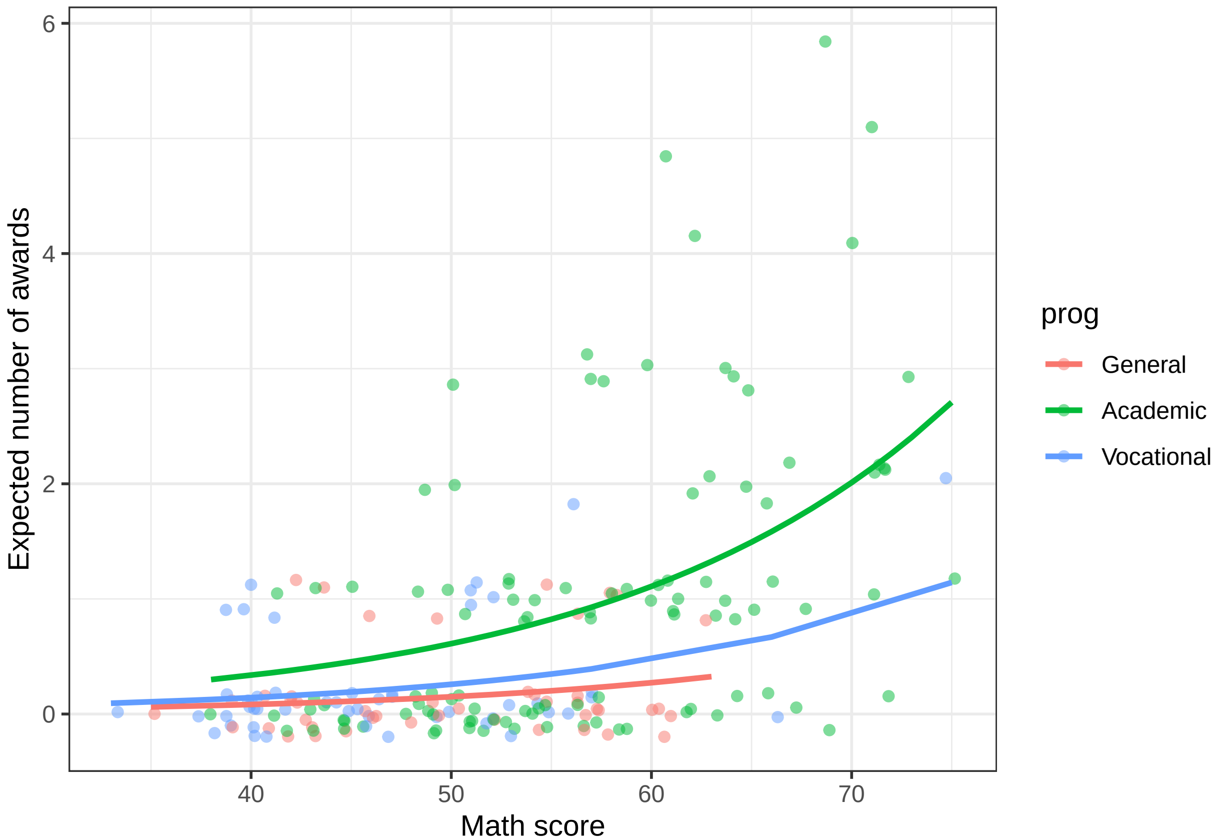
```r
# test accuracy with cutoff = 0.5
prob <- predict(m1, p_test, type = "response")
sum(round(prob) == p_test$num_awards) / length(prob)
```

```
## [1] 0.5833333
```

```r
# test MSE
sum((prob - p_test$num_awards)^2) / length(prob)
```

```
## [1] 0.788173
```

```r
#plot
p$phat <- predict(m1, p, type = "response")
ggplot(p, aes(x = math, y = phat, color = prog)) +
  geom_point(aes(y = num_awards), alpha = 0.5, position = position_jitter(h=.2))+
  geom_line(size = 1)+
  theme_bw()+
  labs(x = "Math score", y = "Expected number of awards")
```



Expected number of awards
6
4
2
0
40
50
60
70
Math score
prog
General
Academic
Vocational