ML HW2

Runze Wang

```
library(r02pro)     #INSTALL IF NECESSARY
library(tidyverse)  #INSTALL IF NECESSARY
library(MASS)
my_sahp <- sahp %>%
  na.omit() %>%
  mutate(expensive = sale_price > median(sale_price)) %>%
  dplyr::select(gar_car, liv_area, oa_qual, expensive)
my_sahp_train <- my_sahp[1:100, ]
my_sahp_test <- my_sahp[-(1:100), ]
```

0.1 Q1

```
fit_gar <- glm(expensive ~ gar_car, data = my_sahp_train, family='binomial')
fit_liv <- glm(expensive ~ liv_area, data = my_sahp_train, family='binomial')
fit_oa <- glm(expensive ~ oa_qual, data = my_sahp_train, family='binomial')
train_pred_gar<- predict(fit_gar, newdata = my_sahp_train, type = 'response')
# train error for gar
pred_train_prob_gar <- predict(fit_gar, type = 'response')
pred_train_label_gar <- ifelse(pred_train_prob_gar > 0.5, "TRUE", "FALSE")
mean(pred_train_label_gar!= my_sahp_train$expensive)
```

```
## [1] 0.29
```

```
# train error for liv
pred_train_prob_liv <- predict(fit_liv, type = 'response')
pred_train_label_liv <- ifelse(pred_train_prob_liv > 0.5, "TRUE", "FALSE")
mean(pred_train_label_liv != my_sahp_train$expensive)
```

```
## [1] 0.35
```

```
# train error for oa
pred_train_prob_oa  <- predict(fit_oa , type = 'response')
pred_train_label_oa  <- ifelse(pred_train_prob_oa  > 0.5, "TRUE", "FALSE")
mean(pred_train_label_oa  != my_sahp_train$expensive)
```

```
## [1] 0.23
```

```
# test error for gar
pred_test_prob_gar <- predict(fit_gar, newdata = my_sahp_test, type = 'response')
pred_test_label_gar <- ifelse(pred_test_prob_gar > 0.5, "TRUE", "FALSE")
mean(pred_test_label_gar != my_sahp_test$expensive)
```

```
## [1] 0.1774194
```

```
# test error for liv
pred_test_prob_liv <- predict(fit_liv, newdata = my_sahp_test, type = 'response')
```

```r
pred_test_label_liv <- ifelse(pred_test_prob_liv > 0.5, "TRUE", "FALSE")
mean(pred_test_label_liv != my_sahp_test$expensive)
```

```
## [1] 0.2580645
```

```r
# test error for oa
pred_test_prob_oa <- predict(fit_oa, newdata = my_sahp_test, type = 'response')
pred_test_label_oa <- ifelse(pred_test_prob_oa > 0.5, "TRUE", "FALSE")
mean(pred_test_label_oa != my_sahp_test$expensive)
```

```
## [1] 0.3064516
```

The oa_qual(0.23)has the smallest training error. The gar_car(0.1774194) has the smallest testing error.

##b

```
fit_all <- glm(expensive ~ gar_car + liv_area + oa_qual , data= my_sahp_train, family='binomial')
pred_train_prob_all  <- predict(fit_all  , type = 'response', newdata= my_sahp_train)
pred_train_label_all  <- ifelse(pred_train_prob_all > 0.5, "TRUE", "FALSE")
mean(pred_train_label_all != my_sahp_train$expensive)
```

```
## [1] 0.21
```

```
pred_test_prob_all <- predict(fit_all, newdata = my_sahp_test, type = 'response')
pred_test_label_all <- ifelse(pred_test_prob_all > 0.5, "TRUE", "FALSE")
mean(pred_test_label_all != my_sahp_test$expensive)
```

```
## [1] 0.1612903
```

The training error is 0.21 and the test error is 0.161. The training error is 0.21 smaller than the training error in three individual variables, so the overall model has better training performance. The test error is 0.161 smaller than the training error in three individual variables, so the overall model can capture the complexity of the data better.

But in the both cases, the test error is smaller than trainig error , which is not normal case. we have to recheck the data and take caution when using logistic regression to predicted model.

##Q2

```
lda.fit <- lda(expensive ~ gar_car + liv_area + oa_qual, data = my_sahp_train)
lda.predict_train <- predict(lda.fit, my_sahp_train)
lda.class_train<- lda.predict_train$class
mean(lda.class_train != my_sahp_train$expensive)
```

```
## [1] 0.17
```

```
#Test Error
lda.predict_test <- predict(lda.fit, my_sahp_test)
lda.class_test <- lda.predict_test$class
mean(lda.class_test != my_sahp_test$expensive)
```

```
## [1] 0.2419355
```

LDA: The training error is 0.17 and test error is 0.2419355.

```
qda.fit <- qda(expensive ~ gar_car + liv_area + oa_qual, data = my_sahp_train)
qda.predict_train<- predict(qda.fit, my_sahp_train)
qda.class_train<- qda.predict_train$class
mean(qda.class_train != my_sahp_train$expensive)
```

```
## [1] 0.16
```

```
#Test Error
qda.predict_test <- predict(qda.fit, my_sahp_test)
qda.class_test <- qda.predict_test$class
mean(qda.class_test != my_sahp_test$expensive)
```

```
## [1] 0.1774194
```

QDA: The training error is 0.16 and test error is 0.1774194.

The training error for LDA and QDA are both samller than logistic regression. QDA has more flexible than LDA, so it has the samllest training eror. The test error for QDA is silimar with the test error for overall model, LDA has the largest test error.

But in the Q1, the test error is smaller than trainig error and is not normal case. QDA show a balance between this three modl, whcih more prefer QDA.

##Q3 Fitted one dimensional normal Gaussian by theorem 4.15 (ISLRv2).

$$f_k(x) = \frac{1}{\sqrt{2\pi}\sigma_k} \exp\left(-\frac{(x-\mu_k)^2}{2\sigma_k^2}\right)$$

$$p_k(x) = \frac{\pi_k \frac{1}{\sqrt{2\pi}\sigma_k} \exp\left(-\frac{(x-\mu_k)^2}{2\sigma_k^2}\right)}{\sum_{l=1}^{K} \pi_l \frac{1}{\sqrt{2\pi}\sigma_l} \exp\left(-\frac{(x-\mu_l)^2}{2\sigma_l^2}\right)}$$

$$p_k(x) = \frac{\pi_k \frac{1}{\sqrt{\sigma_k}} \exp\left(-\frac{(x-\mu_k)^2}{2\sigma_k^2}\right)}{\sum_{l=1}^{K} \pi_l \frac{1}{\sqrt{\sigma_l}} \exp\left(-\frac{(x-\mu_l)^2}{2\sigma_l^2}\right)}$$

Taking the log

$$\log(p_k(x)) = \log\left(\frac{\pi_k \frac{1}{\sigma_k} \exp\left(-\frac{(x-\mu_k)^2}{2\sigma_k^2}\right)}{\sum_{l=1}^{K} \pi_l \frac{1}{\sigma_k} \exp\left(-\frac{(x-\mu_l)^2}{2\sigma_l^2}\right)}\right)$$

By the property of logarithm

$$\log(p_k(x)) = \log\left(\frac{\pi_k}{\sigma_k}\right) - \log\left(\sum_{l=1}^{K} \frac{\pi_l}{\sigma_l}\right) - \left(\frac{(x-\mu_k)^2}{2\sigma_k^2}\right) + \sum_{l=1}^{K}\left(\frac{(x-\mu_l)^2}{2\sigma_l^2}\right)$$

In this case, the summation term not contribute to the differentiation between classes in the optimization process, we can omit summation term.

$$\log(p_k(x)) = \log\left(\frac{\pi_k}{\sigma_k}\right) - \frac{x^2}{2\sigma_k^2} - \frac{\mu_k^2}{2\sigma_k^2} + \frac{\mu_k x}{\sigma_k^2}$$

$$\delta_k(x) = \log\left(\frac{\pi_k}{\sigma_k}\right) - \frac{x^2}{2\sigma_k^2} - \frac{\mu_k^2}{2\sigma_k^2} + \frac{\mu_k x}{\sigma_k^2}$$

In Conclusion, this Bayes classifier is quadratic not linear.

##Q4

$$p(X) = \frac{e^{\beta_0+\beta_1 X_1+\beta_2 X_2}}{1+e^{\beta_0+\beta_1 x_1+\beta_2 X_2}}$$

$$p(X) = \frac{e^{-6+0.05*40+1*3.5}}{1+e^{-6+0.05*40+1*3.5}}$$

$$p(X) = \frac{e^{-6+0.05*40+1*3.5}}{1+e^{-6+0.05*40+1*3.5}}$$

$$p(X) = 0.3775$$

The probability that a student who studies for 40 h and has an undergrad GPA of 3.5 gets an A in the class is 0.3775. ##b

$$p(X) = \frac{e^{-6+0.05*X_1+1*3.5}}{1+e^{-6+0.05*X_1+1*3.5}} = 0.5$$

$$e^{-6+0.05*X_1+1*3.5} = 0.5*(1+e^{-6+0.05*X_1+1*3.5})$$

$$e^{0.05*X_1-2.5} = 0.5+0.5*e^{0.05*X_1-2.5}$$

$$e^{0.05*X_1-2.5} = 1$$

Taking the log

$$0.05*X_1-2.5 = 0$$

$$X_1 = 50$$

student need to study 50 hours to have a 50 % chance of getting an A in the class.