Stat 437 HW4

Nam Jun Lee (11606459)

General rule

Please show your work and submit your computer codes in order to get points. Providing correct answers without supporting details does not receive full credits. This HW covers

- Bayes classifier
- kNN classifier
- Discriminant analysis

For an assignment or project, you DO NOT have to submit your answers or reports using typesetting software. However, your answers must be well organized and well legible for grading. Please upload your answers in a document to the course space. Specifically, if you are not able to knit a .Rmd/.rmd file into an output file such as a .pdf, .doc, .docx or .html file that contains your codes, outputs from your codes, your interpretations on the outputs, and your answers in text (possibly with math expressions), please organize your codes, their outputs and your answers in a document in the format given below:

```
Problem or task or question ...
Codes ...
Outputs ...
Your interpretations ...
```

It is absolutely not OK to just submit your codes only. This will result in a considerable loss of points on your assignments or projects.

Conceptual exercises: I (Bayes classifier)

1. This exercise is on Bayes theorem and Bayes classifier.

1.1) State clearly the definition of the 0-1 loss function. Can this function be used in multi-class classification problems?

Yes, 0-1 loss L is classified as 0 only if the observation is classified correctly, otherwise L is classified as 1. Therefore, it can be used in multi-class classification problems to measure the performance of the classifier.

1.2) Let Y be the random variable for the class label of a random vector X, such that $Y \in \mathcal{G} = \{1, \ldots, K\}$ where $K \geq 2$ is the number of classes. Let $\hat{Y}$ be the estimated class label for X. Given the prior $\Pr(Y = k) = \pi_k, k \in \mathcal{G}$ on Class k and the conditional density $f_k(x)$ of X when it comes from Class k. Provide the formula to obtain the posterior $\Pr(Y = k|X = x)$, which is essentially the Bayes theorem. What is the Bayes classifier and how does it classify a new observation x_0 from X? Is the decision boundary of the Bayes classifier linear or quadratic in X? Explain (but do not have to mathematically prove) why the Bayes classifier minimizes the expected 0-1 loss. Note the a proof of the fact that the Bayes classifier minimizes the expected 0-1 loss is given in "LectureNotes4_notes.pdf". You should not copy and paste the proof. Instead, please provide the explanation based on your understanding of the proof.

Posterior formula: $Pr(Y_i = j|x_i) = \frac{f_j(x_i|Y_i=j)Pr(Y_i=j)}{f(x_i)}$
The Bayes classifier uses a Bayes model that uses prior, conditional, marginal, and posterior probabilities.
The Bayes classifier minimizes the expected value using the **0-1 loss function** to classify the new observation `x_0` at X.
The decision boundary of the Bayes classifier is linear in X.
The reason why the Bayes classifier minimizes the expected 0-1 loss is that the Bayes classifier can identify the misclassification by obtaining a class that maximizes the posterior probability. In order to calculate the posterior probability in this process, errors can be minimized using the 0-1 loss function, thereby creating a better model.

1.3) If $K = 2$ in subquestion 1.2), what is the threshold value on $\Pr(Y = 1|X = x_0)$ that is used by the Bayes classifier to determine the class label for x_0? Suppose you use a different threshold value on $\Pr(Y = 1|X = x_0)$ to classify x_0, is the corresponding classifier still the Bayes classifier, and is the corresponding loss function still the 0-1 loss? Explain your answer. Provide a scenario where to classify an observation a different threshold value is more sensible than the threshold value used by the Bayes classifier.

When K=2, the threshold value of $Pr(Y = 1|X = x_0)$ is **0.5**.

If you use a threshold different from 0.5 to classify x_0, it is different from using a Bayes classifier in case of 0-1 loss. For example in "LectureNotes5a_notes.pdf", if it is difficult to predict the default status of non-payers and delinquents in classifying non-payers, low thresholds such as 0.2 can be used instead of 0.5. This class provides a high error rate when setting the threshold to 0.5, so using the threshold of 0.2 reduces the error rate of the delinquent delinquent class, but increases the error rate of the non-defaulter class. So when determining the threshold, domain knowledge is first required.

1.4) If $K = 2$ in subquestion 1.2), $\pi_1 = 0.6$, $f_1(x) \sim \text{Gaussian}(0, 1)$ and $f_2(x) \sim \text{Gaussian}(2, 1)$ and $x_0 = 1.5$. Compute $\Pr(Y = 1|X = x_0)$ and use the Bayes classifier to classify x_0.

Find $\Pr(Y = 1|X = x_0)$: we know $K = 2$, $\pi_1 = 0.6$, $f_1(x) \sim \text{Gaussian}(0, 1)$ and $f_2(x) \sim \text{Gaussian}(2, 1)$ and $x_0 = 1.5$.
First, set S1 = merginal density for X: $f(x) = 0.6 * \text{Gaussian}(0, 1) + 0.6 * \text{Gaussian}(2, 1)$
Therefore, Bayes classifier: $Pr(Y = 2|X = 1.5) = \frac{0.6_2(1.5)f_2(1.5)}{\text{S1}}$

Conceptual exercises: II (k-NN classifier)

2. Given the training set $\mathcal{T}$ of n observations $(x_1, y_1), \ldots, (x_n, y_n)$, where y_i is the class label of observation x_i and $y_i \in \mathcal{G} = \{1, \ldots, K\}$ for $K \geq 2$, consider k-NN classifier, where k is the neighborhood size.

2.1) Describe how the decision boundary (such as its smoothness and shape) of k-NN classifier changes as k changes.

As k of the k-NN classifier increases, $\bar{\mathbf{Y}}(x)$ is the average of quite a lot of y_j's, and the corresponding x_j's of $N_k(x)$ is likely to span a large subset of feature spaces. Thus, if the two feature observations x and x_0 are not far apart, $N_k(x)$ and $N_k(x_0)$ will not contain significantly different x_j, which is a relatively smooth "insufficiently flexible" decision boundary.

2.2) Explain why the training error of 1-NN classifier is 0. Provide an estimator of the test error of a classifier and explain why it can be used as such an estimator. Is it true that a large k leads to a k-NN classifier with smaller test error? Can the test error of a k-NN classifier be equal to the test error of the Bayes classifier? When k is large and k/n is small, what is a k-NN classifier approximately estimating?

Under certain conditions, the expected 0-1 loss can be well approximated by a training error or a test error, so the training error of the 1-NN classifier is 0.
The test error of the classifier is a preferred measure for the performance of the classifier, and the better classifier has a smaller test error, so the test error should estimate the expected 0-1 loss.
No, it is not true that a large k leads to a smaller k-NN classifier.
Yes, the test error of the k-NN classifier may be the same as the test error of the Bayes classifier.
If k is large and k/n is small, $\hat{g}_j(x) \approx Pr(Y = j|X = x)$, and which estimates the similarity of all classifiers.

2.3) When there are $K \geq 2$ classes, how does a k-NN classifier classify a test observation x_0?

If there is a $K \geq 2$ class, the k-NN classifier classifies the test observation x_0 based on a smaller test error.

2.4) When should data be standardized before applying a k-NN classifier? When standardizing data, do we standardize each observation or each feature?

Before applying the k-NN classifier, data must be standardized before data is divided into training and test data.
When standardizing data, standardize each feature (if the scale is quite different).

2.5) Using your understanding of Example 3 in "LectureNotes4b_notes.pdf", provide a step-by-step guide on how to choose an optimal k for k-NN classifier using cross-validation. You can provide such as guide in the form of "pseudocode" (see, e.g., https://en.wikipedia.org/wiki/Pseudocode for some details on pseudocode). Suppose the training set has few observations, can you still perform cross-validation in order to choose an optimal k? Explain your answer. (Hint: for the 2nd part, think about if having more observations helps better estimate test error.)

Step-by-step guide for m-fold cross-validation for kNN classifiers:
1. Randomly split n observations randomly into m folds of similar size.
2. Select Fold S as "Test Set", set the rest of Fold as "Training Set", apply the kNN classifier, and obtain test error E.
3. For each $Sof1,.,m$, perform item 2. and obtain E of m.
4. Calculate the sample mean $\hat{\mu}(k,m)$ and the sample standard deviation $\hat{\sigma}(k,m)$ for m E.
5. Use $\hat{\mu}(k,m)$ as an estimate of the test error of the classifier.
Assuming that there are few observations in the training set referring to the second part, it is not very reasonable to attempt cross-validation to select the optimal neighbor size k.

Conceptual exercises: III (Discriminant analysis)

3. Exercise 2 of Section 4.7 of the Text, which starts with "It was stated in the text that classifying an observation to the class for which (4.12) is largest is equivalent to classifying an observation to the class for which (4.13) is largest. Prove that this is the case." (Helpful information on how to prove this is contained in the lecture video on LDA and "LectureNotes5b_notes.pdf".)

Prove (4.12) and (4.13):

Function (4.12)

$$p_k(x) = \frac{\pi_k \frac{1}{\sqrt{2\pi\sigma}} exp(-\frac{1}{2\sigma^2}(x-\mu_k)^2)}{\sum_{l=1}^{K} \pi l \frac{1}{\sqrt{2\pi\sigma}} exp(-\frac{1}{2\sigma^2}(x-\mu l)^2)}$$

Removing all terms of the class k from Bayes classifier and denominator:

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

And, Function (4.13)

$$\delta_k(x) = x * \frac{\mu_k}{\sigma^2} - \frac{\mu_k^2}{2\sigma^2} + log(\pi_k)$$

This shows that discriminant function contains equal the Bayes classifier with k. Hence, largest function (4.12) will be the largest function (4.13).

4. Exercise 3 of Section 4.7 of the Text, which starts with "This problem relates to the QDA model, in which the observations within each class are drawn from a normal distribution with a class specific mean vector and a class specific covariance matrix. We consider the simple case where p = 1; i.e. there is only one feature." (Helpful information on how to prove this is contained in the lecture video on QDA and "LectureNotes5b_notes.pdf".)

Exercise 3 of Section 4.7 of the Textbook, prove that in (4.11), the Bayes' classifier is not linear.
Function (4.11)

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

Using Bayes classifier

$$P(Y=k|X=x) = \frac{P(X=x|Y=k)P(Y=k)}{\sum_{k=1}^{K} P(X=x|Y=k)P(Y=K)} = \frac{\pi_k f_k(x)}{\sum \pi_k f_k(x)}$$

So, Apply Function (4.13) and (4.12)

$$\delta_k(x) = log\pi_k - (\frac{x^2}{2\mu^2} - \frac{x\mu_k}{\mu^2} + \frac{\mu_k^2}{2\mu^2}) = x * \frac{\mu_k}{\sigma^2} - \frac{\mu_k^2}{2\sigma^2} + log(\pi_k)$$

So, $\delta_k(x)$ is not linear in x. It relates the QDA model.

5. Exercise 5 of Section 4.7 of the Text, which starts with "We now examine the differences between LDA and QDA." (Hint: for this question, you may also use information from Figures 4.9, 4.10 and 4.11 in the Text.)

- (a) If the Bayes decision boundary is linear, do we expect LDA or QDA to perform better on the training set? On the test set?

If the Bayes decision boundary is linear, QDA will perform better in the training set. This is because QDA may be more suitable for higher flexibility. Conversely, LDA will perform better than QDA in the test set. Because QDA overfitting linearity at Bayes decision boundaries, LDA will perform better in the test set.

- (b) If the Bayes decision boundary is non-linear, do we expect LDA or QDA to perform better on the training set? On the test set?

If the Bayes decision boundary is non-linear, QDA will perform better on both training and test sets.

- (c) In general, as the sample size n increases, do we expect the test prediction accuracy of QDA relative to LDA to improve, decline, or be unchanged? Why?

As sample size n increases, the test prediction accuracy of QDA for LDA will not change. This is because QDA has higher flexibility than LDA in the training set, so increasing sample size variance does not affect test prediction accuracy.

- (d) True or False: Even if the Bayes decision boundary for a given problem is linear, we will probably achieve a superior test error rate using QDA rather than LDA because QDA is flexible enough to model a linear decision boundary. Justify your answer.

False. If the Bayesian decision boundary is linear, the QDA is flexible enough to model the linear decision boundary, but this can exceed linearity, which will adversely affect the test error rate.
6. Let Y be the random variable for the class label of a random vector $X \in \mathbb{R}^p$ (where p is the number of features), such that $Y \in \mathcal{G} = \{1, \dots, K\}$ and $\Pr(Y = k) = \pi_k$ for Class k with $k \in \mathcal{G}$, where $K \geq 2$ is the number of classes. Consider the Gaussian mixture model such that the conditional density of X when it comes from Class k is $f_k(x) \sim \text{Gaussian}(\mu_k, \Sigma_k)$. Given the training set $\mathcal{T}$ of n observations $(x_1, y_1), \dots, (x_n, y_n)$ on (X, Y), where y_i is the class label of observation x_i, do the following:

6.1) Provide the MLEs of π_k, μ_k and Σ_k for each $k \in \mathcal{G}$ respectively for the case where all Σ_k's are equal and for the case where not all Σ_k's are equal. When $p > n$, is the MLE of Σ_k still accurate? If not, recommend a different estimator for estimating Σ_k and provide details on this estimator.

In general, MLE works well if p is less than the sample size n. However, when p>n, it is usually not accurate and plug-in $\hat{\delta_k}$ will not be performed well for classification. So, when p>n, a regularized DA is recommended.
The regularized DA is an appropriate method if p is larger than the sample, and the equation for obtaining the regularized estimate is:

$$\hat{\Sigma}_{k,a} = \alpha\hat{\Sigma}_k + (1 - \alpha)\hat{\Sigma}, \alpha \in [0, 1]$$

6.2) Assume $p = 2$ and $K = 2$ and $k = 1$. For the density $f_k(x) \sim \text{Gaussian}(\mu_k, \Sigma_k)$, what shape do its contours take, and how does Σ_k control the shape of these contours? How do you check if the conditional density of X given that it comes from Class k is Gaussian?

In the case of $p = 2$, $K = 2$, $k = 1$. Density $f_k(x) \sim \text{Gaussian}(\mu_k, \Sigma_k$, the contour is circular, and Σ_k controls the shape of this contour. If $\Sigma_k = I$, the contour is circular, and if $\Sigma_k! = I$, the contour

is elliptical.
If p > 1, we can check the conditional density of X given that it comes from Class k as Gaussian:

$$f_k(x) \sim \text{Gaussian}(\mu_k, \Sigma_k) = \frac{1}{\sqrt{2\pi}\sigma_k} exp(-\frac{1}{2\sigma_k^2}(x-\mu_k)^2)$$

6.3) Is it true that discriminant analysis will perform badly if the Gaussian assumption is violated? (Hint: for this question, you may also use the information provided by Figures 4.10 and 4.11 of the Text.) Let $X = (X_1, \ldots, X_p)^P$, i.e., X_1 up to X_p are the feature variables. Can discriminant analysis be applied to observations of X when some of $X_j, j = 1\ldots, p$ is a discrete variable (such as a categorical variable)? Explain your answer.

Yes, if the Gaussian assumption is violated, discriminant analysis may not be performed properly. In this case, logistic regression analysis is better than LDA or QDA analysis. In addition, discriminant analysis can be applied to observations of X when some of $X_j, j = 1\ldots, p$ even if some of them are discrete variables.

6.4) What is a ROC curve, and what is AUC? How is AUC used to gauge the performance of a classifier? If you apply the same classifier, say, LDA or QDA under the same Gaussian mixture model, to two data sets that are independently generated from the same data generating process, i.e., that are independently generated from (X, Y) for classification problems, and obtain two ROC curves, would the two ROC curves be quite different? Explain your answer. When there are 3 or more classes, are the codes provided in the lecture notes able to obtain ROC curves and their AUC's for LDA and QDA?

The ROC curve is a graph that is widely used to display two types of errors simultaneously for all possible thresholds.
AUC is the base area of the ROC curve. It is a value that can be a numerical criterion in performance evaluation, and the closer to 1, the better the model.
Yes, for the classification problem, the two ROC curves are different when applied to two LDA and QDA ROC curves independently generated at (X, Y). LDA assumes the same component covariance matrix, but since it is not QDA, LDA has lower flexibility and less variance than QDA. Therefore, the two rock curves are different.
No, when there are more than three classes, the code provided in the lecture note does not obtain the ROC curve and AUC for LDA or QDA.

6.5) Describe the key similarities and differences, respectively, between LDA and logistic regression. Provide a situation where discriminant analysis can still be sensibly applied but logistic regression is not well-defined.

Similarities between Logistic regression and LDA: Both methods use linear modeling equations for logarithmic calculations.

Difference between Logistic regression and LDA:
* (1) Logistic regression does not employ a mixture model, and it takes as the conditional probability of $X = x$ when it is generated using Class k conditional density f_k. However, LDA employs a Gaussian mixture model and takes as the posterior probability of $X = x$ in Class k after x is obtained.
* (2) Logistic regression does not require the ambient distribution F of X and maximizes the conditional likelihood $Pr(Y = k|X)$ given the observation of X. However, since LDA aims to obtain maximum posterior $Pr(Y = g|X)$ when classifying X as class g, we implicitly use F to maximize the likelihood of binding to (X, Y).

* (3) If observations can be completely separated into two classes in hyperplane, the maximum likelihood estimate of the regression parameter is not defined in the case of logistic regression, but the coefficients can be well defined and implemented in the case of LDA.
When measuring equally correlated Gaussian feature variables across component bivariate Gaussian density; logistic regression analysis is not well defined.

Applied exercises: I (k-NN classifier)

7. Please refer to the NYC flight data `nycflights13` that has been discussed in the lecture notes and whose manual can be found at https://cran.r-project.org/web/packages/nycflights13/index.html. We will use `flights`, a tibble from `nycflights13`.

Please use `set.seed(123)` for the whole of this exercise. Randomly select from `flights` for each of the 3 `carrier` "UA", "AA" or "DL" 500 observations for the 3 features `dep_delay`, `arr_delay` and `distance`. Let us try to see if we can use the 3 features to identify if an observation belongs a specific carrier. The following tasks and questions are based on the extracted observations. Note that you need to remove rows with `na`'s from the extracted observations.

```r
# select 3 carrier and 3 features in flights dataset
dat <- nycflights13::flights %>%
   dplyr::select(carrier, dep_delay, arr_delay, distance) %>%
   dplyr::filter(carrier %in% c("UA", "AA", "DL"))
# remove rows with na from the observations
dat = na.omit(dat)
# check na in data
sum(is.na(dat))
```

```
## [1] 0
```

```r
# 500 observations for 3 features
set.seed(123)
dat1 <- dat[sample(nrow(dat), 500, replace = F), ]
```

After filtering the dataset, the missing values were processed and 500 random observations were generated in the variable `dat1`.

7.1) First, you need to standardize the features since they are on very different scales. Then randomly split the observations into a training set that contains 70% of the observations and a test set that contains the remaining observations.

```r
# standardize the feature
dat11 <- scale(dat1[,2:4])
set.seed(123)
# 70 % of training set and 30% of test set and set lables
train1 = base::sample(1:nrow(dat11), 0.7*nrow(dat11))
test1 = (1:nrow(dat11))[-train1]
trainSet1 = dat11[train1,]
```

```r
testSet1 = dat11[test1,]
trainLabel1 = dat1$carrier[train1]
testLabel1 = dat1$carrier[test1]
```

Features were standardized and training set and test set were established.

7.2) Consider the observations as forming 3 classes that are determined by `carrier`. To the training set, apply 10 fold cross-validation to k-NN classifier with features `arr_delay`, `dep_delay`, and `distance` to determine the optimal k from the values $\{1, \ldots, 15\}$. Apply the optimal k-NN to the test set, provide the classification table and the overall error rate, and provide visualization of the classification results. Do you think the error rate is reasonable? Explain your answer. (Hint: you can follow the strategy provided by Example 3 in "LectureNotes4b_notes.pdf".)

```r
set.seed(123)
# Step 1
# 10 fold
f = 10
# 10 fold random split of training set
fold = sample(1:f, nrow(trainSet1), replace = TRUE)
# check number of observation in each fold
table(fold)
```

```
## fold
##  1  2  3  4  5  6  7  8  9 10
## 31 31 33 30 34 34 38 31 43 45
```

```r
# 10-fold cv for 2-NN classfier
k = 2
# store test error for each fold
testErr1 <- double(f)
for (x in 1:f) {
   trainTmp <- trainSet1[fold != x,]
   testTmp <- trainSet1[fold == x,]
   trainLabelTmp <- trainLabel1[fold != x]
   testLabelTmp <- trainLabel1[fold == x]
   knn2 = knn(trainTmp, testTmp, trainLabelTmp, k)
   numOfMissObservation <- sum(1-as.numeric(knn2 == testLabelTmp))
   tErr = numOfMissObservation / length(testLabelTmp) # test error
   testErr1[x] = tErr
}
```

```r
# Step 2
# 10 fold cv for KNN classifiers for k = 1,...,15
kmax = 15
testErr <- matrix(0, nrow=2, ncol=kmax)
for (k in 1:kmax) {
   testErr1 <- double(f)
```

```r
  for (x in 1:f) {
    trainTmp = trainSet1[fold != x,]
    testTmp = trainSet1[fold == x,]
    trainLabelTmp = trainLabel1[fold != x]
    testLabelTmp = trainLabel1[fold == x]
    knn1to15 = knn(trainTmp, testTmp, trainLabelTmp, k)
    numOfMissObservation = sum(1-as.numeric(knn1to15 == testLabelTmp))
    tErr = numOfMissObservation / length(testLabelTmp)
    testErr1[x] = tErr
  }
  testErr[,k]=c(mean(testErr1), sd(testErr1))
}

# optimal k chosen by 10-fold cv for k = 1,...,15
colnames(testErr) = paste("k=", 1:kmax, sep="")
rownames(testErr) = c("mean(TestError)", "sd(TestError)")
testErr = as.data.frame(testErr)
as.numeric(testErr[1,])
```

```
##  [1] 0.5746586 0.5897821 0.5695861 0.5618785 0.5443093 0.5289542 0.5425410
##  [8] 0.5323567 0.5309836 0.5415908 0.5358920 0.5441429 0.5490828 0.5571488
## [15] 0.5563864
```

```r
# find optimal k
hatk = which(testErr[1,]==min(testErr[1,]))
hatk
```

```
## [1] 6
```

Each k = 1,. . . ,15, it shows the result value of the 10-fold cross-validation for the KNN classifier, and it can be seen that it is appropriate to **set k to 6**.

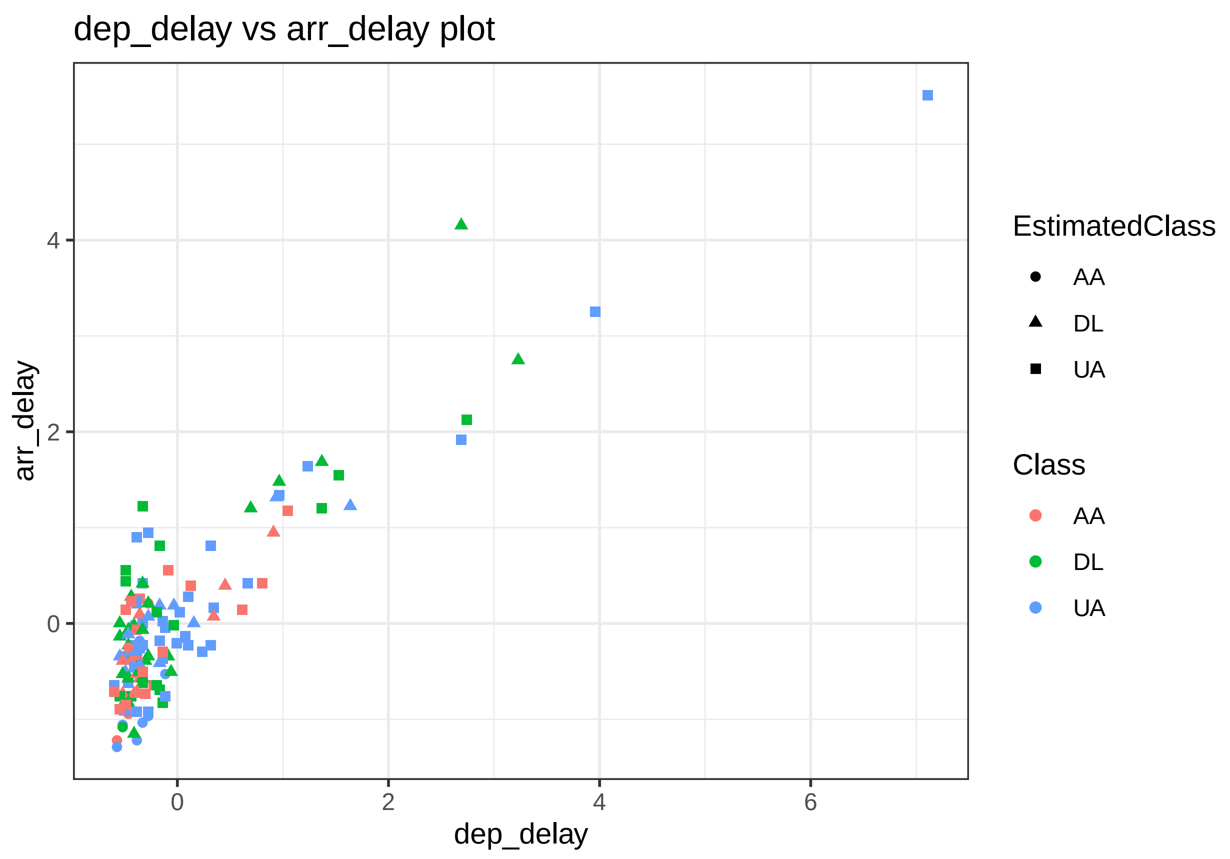
```r
# Apply optimal kNN classifier
knnOpt1 <- knn(trainSet1, testSet1, trainLabel1, hatk)
numOfMissObservation1 <- sum(1-as.numeric(knnOpt1==testLabel1))
# test error
terrorOpt <- numOfMissObservation1/length(testLabel1)
terrorOpt
```

```
## [1] 0.5133333
```

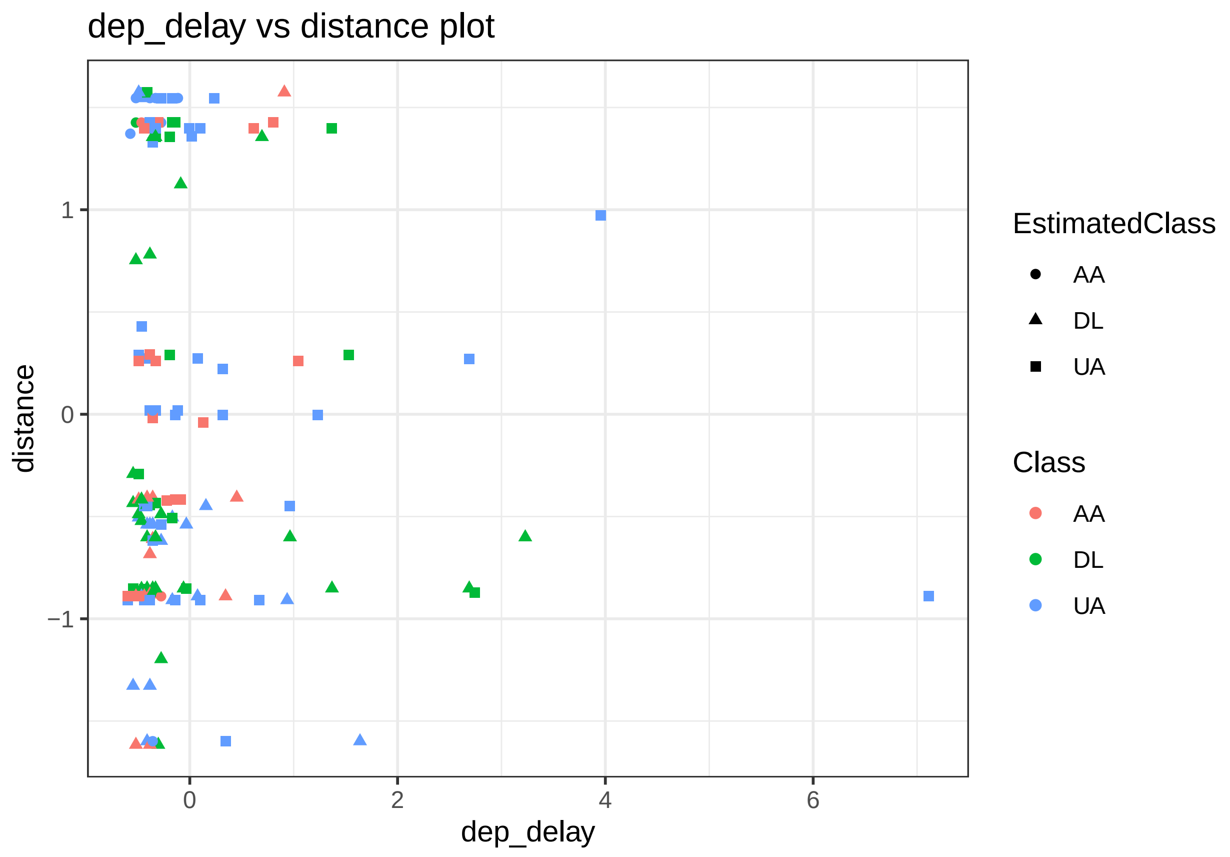
```r
# show table
table(knnOpt1, testLabel1)
```

```
##          testLabel1
## knnOpt1 AA DL UA
##       AA  4  2  9
##       DL 13 29 19
##       UA 16 18 40
```

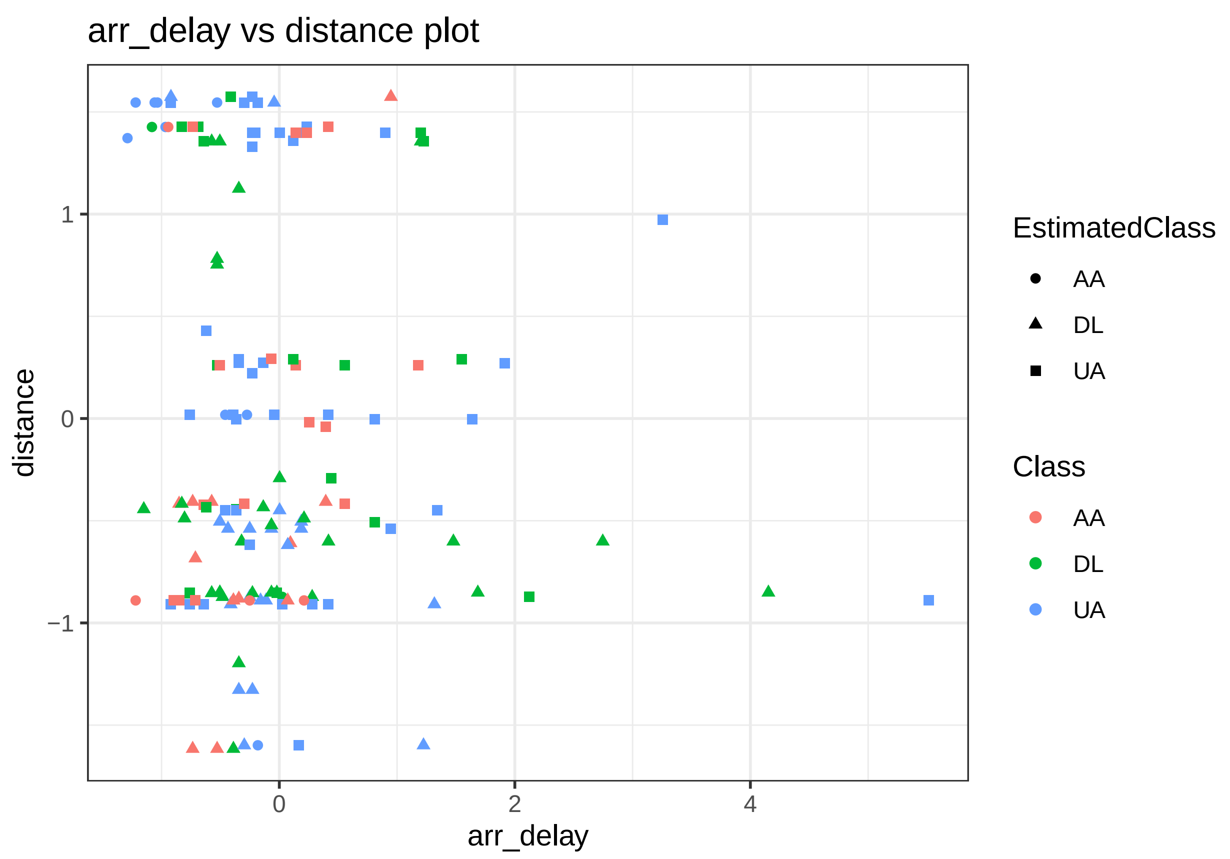
```
# visualization via 2 features
testSet11 <- as.data.frame(testSet1)
testSet11$Class = as.factor(testLabel1)
testSet11$EstimatedClass = knnOpt1
plot1 = ggplot(testSet11, aes(dep_delay, arr_delay)) +
   geom_point(aes(shape=EstimatedClass, color=Class)) +
   theme_bw() +
   labs(title = "dep_delay vs arr_delay plot")
plot1
```



```
plot2 = ggplot(testSet11, aes(dep_delay, distance)) +
   geom_point(aes(shape=EstimatedClass, color=Class)) +
   theme_bw() +
   labs(title = "dep_delay vs distance plot")
plot2
```



```
plot3 = ggplot(testSet11, aes(arr_delay, distance)) +
   geom_point(aes(shape=EstimatedClass, color=Class)) +
   theme_bw() +
   labs(title = "arr_delay vs distance plot")
plot3
```

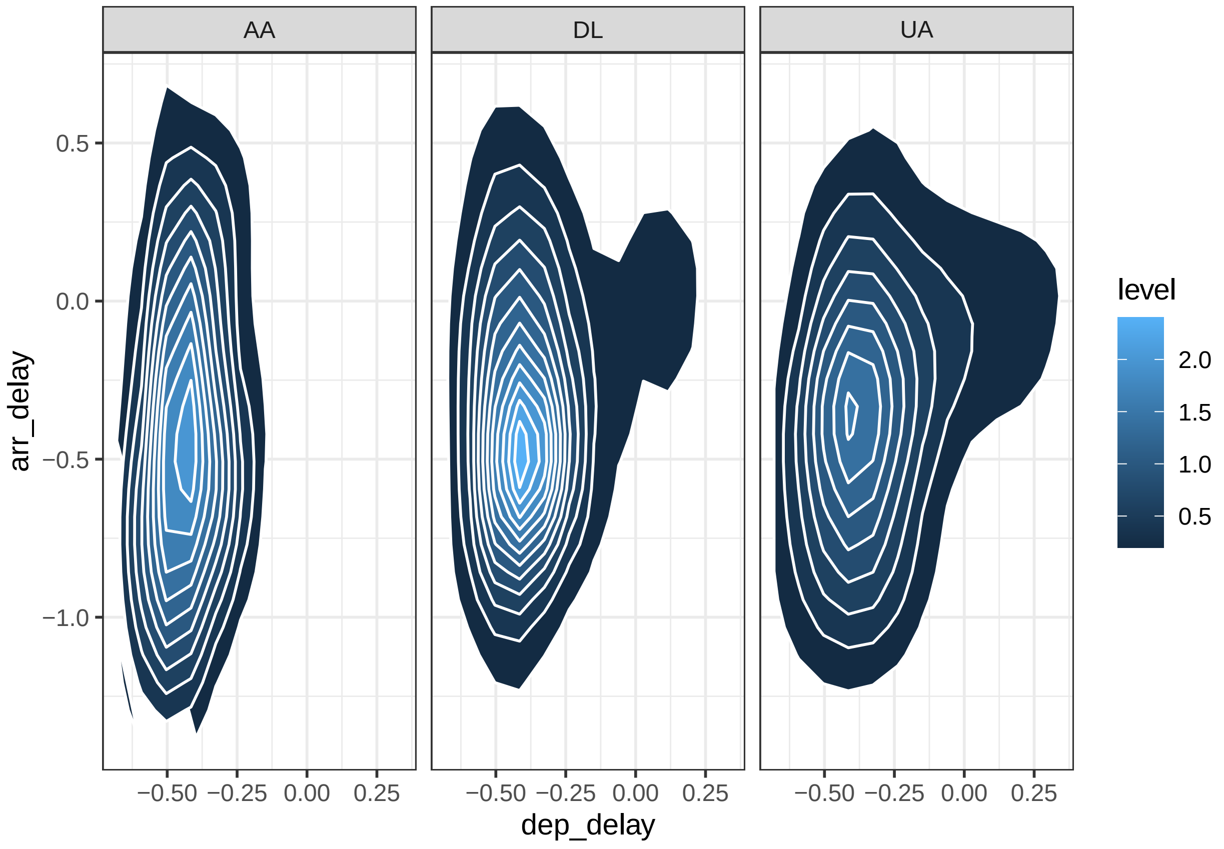


Applying the optimal k = 6 obtained by 10-fold cross-validation on training set and checking the error rate: **0.51333**
As a result of classification, when checking the plots and classification tables, it is confirmed that each class is not well grouped. So, it is judged reasonable that the error rate was **51.33 %**.

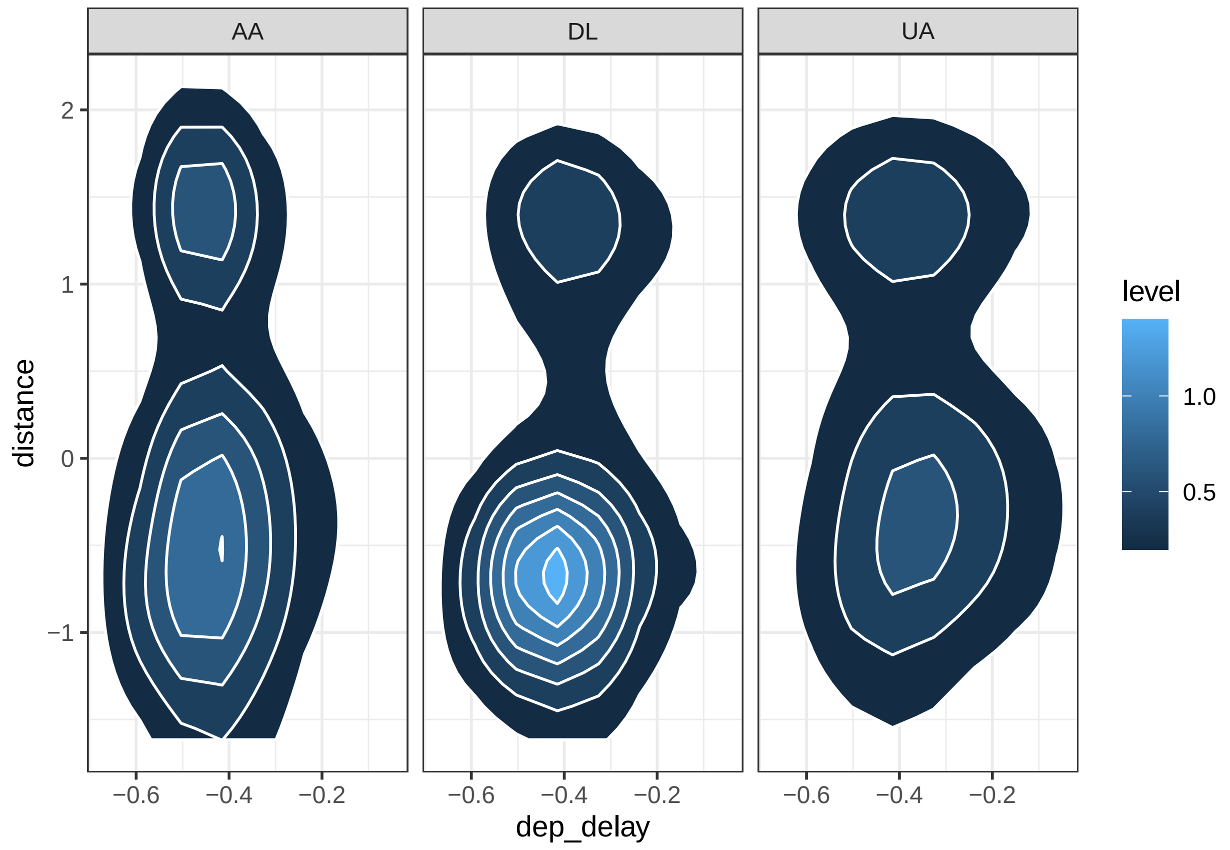
7.3) Note that your have standardized the features `arr_delay`, `dep_delay`, and `distance`. However, with the unstandardized features, you would surely know that none of them follows a Gaussian distribution no matter with respect to which class (i.e., `carrier`) you look at their observations since these features have non-negative values (whereas a Gaussian distribution can generate negative values). Again, the 3 classes are determined by `carrier`. So, you will apply QDA based on the 3 standardized the features to the training set to train the model, and then apply the trained model to the test set (that contains the 3 standardized the features) to classify its observations.

(7.3a) First, check if the Gaussian assumption is satisfied. For this, note that if the standardized features `arr_delay`, `dep_delay`, and `distance` follow a trivariate Gaussian distribution for each individual class, then any pair among the 3 standardized features follows a bivariate Gaussian distribution for each individual class.

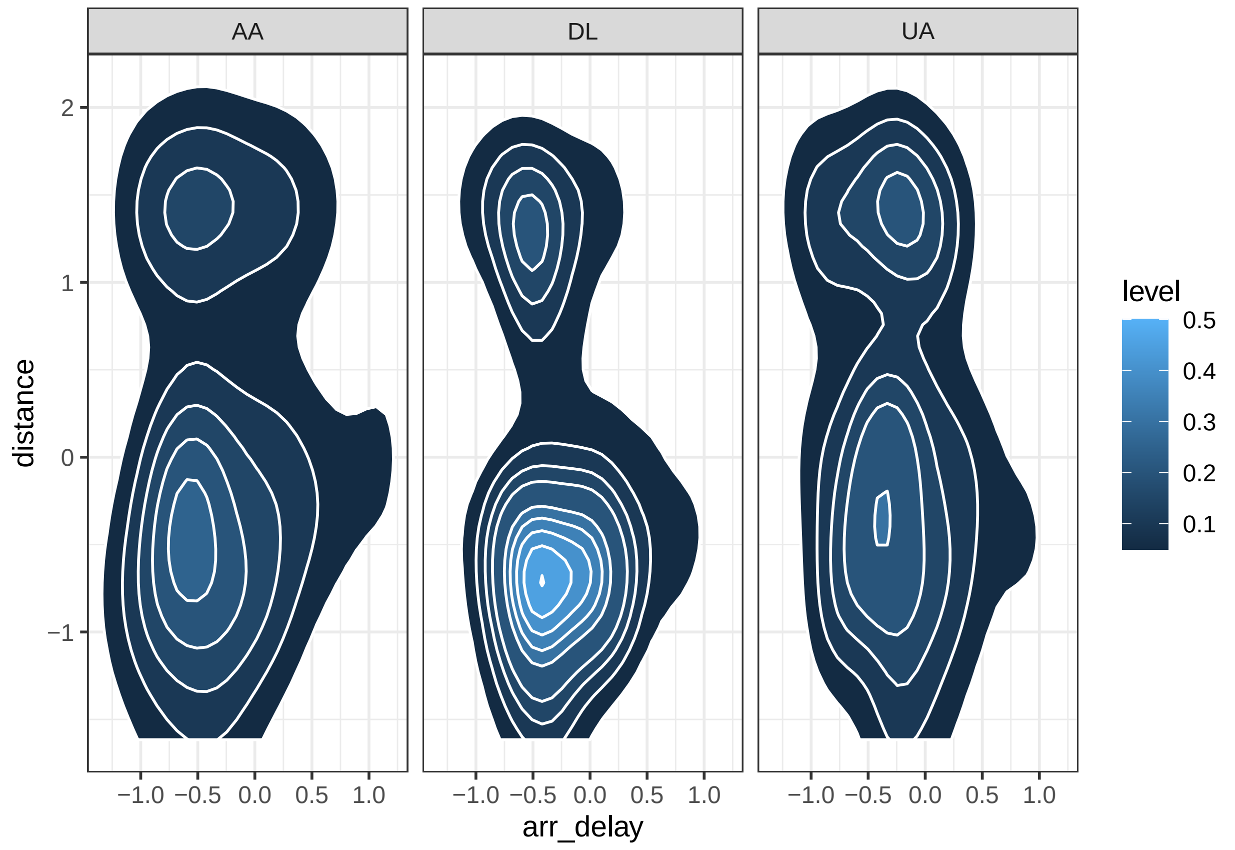
```r
# recall standardized set dat11 into data frame
dat111 <- as.data.frame(dat11)
dat111$Carrier <- as.factor(dat1$carrier)
# Show 2d density plots of bivariate gaussian distribution for each class
dplot1 <- ggplot(dat111, aes(dep_delay, arr_delay)) +
   theme_bw() +
   facet_grid(~Carrier, labeller = label_parsed) +
   stat_density_2d(aes(fill = ..level..),
                   geom = "polygon",
                   colour = "white") +
   theme(legend.position = "right", legend.direction = "vertical")
dplot1
```



```
dplot2 <- ggplot(dat111, aes(dep_delay, distance)) +
  theme_bw() +
  facet_grid(~Carrier, labeller = label_parsed) +
  stat_density_2d(aes(fill = ..level..),
                  geom = "polygon",
                  colour = "white") +
  theme(legend.position = "right", legend.direction = "vertical")
dplot2
```



```
dplot3 <- ggplot(dat111, aes(arr_delay, distance)) +
   theme_bw() +
   facet_grid(~Carrier, labeller = label_parsed) +
   stat_density_2d(aes(fill = ..level..),
                   geom = "polygon",
                   colour = "white") +
   theme(legend.position = "right", legend.direction = "vertical")
dplot3
```



As a result of checking whether it follows the Gaussian distribution through the 2d density plot, it can be confirmed that it does follow the bivariate Gaussian distribution by class for each feature.

(7.3b) Apply the estimated (i.e., trained) QDA model to the test set, provide the estimated mixing proportion and estimated mean vector for each class, and provide the classification table. If you randomly pick an observation on the 3 standardized features, is it approximately equally likely to belong to each of the 3 carriers? (You do not need to mathematically prove your answer. However, think along this line: we are testing the equality of 3 population proportions, and each estimated population proportion is based on around 350 observations, which approximately can be done via a z-test since the central limit theorem is in effect.) How is the performance of QDA on this test set? Explain your answers.

```
# change the train and test set to data frame
t1 <- data.frame(trainSet1)
te1 <- data.frame(testSet1)
# apply estimated trained QDA model
qda.fit.3ca <- qda(trainLabel1~., data=t1)
# Set QDA model to the test set
qda.class.3ca <- predict(qda.fit.3ca,te1)$class
# show estimated mixing proportion
qda.fit.3ca$prior
```

```
##         AA         DL         UA
## 0.1942857 0.3571429 0.4485714
```

```
# estimated mean
qda.fit.3ca$means
```

```
##        dep_delay    arr_delay   distance
## AA -0.1744714040 -0.208966511  0.1611379
## DL -0.0003235299  0.001568097 -0.2210485
## UA  0.0911770326  0.100422224  0.1358149
```

```
# show table QDA
table(qda.class.3ca, testLabel1)
```

```
##              testLabel1
## qda.class.3ca AA DL UA
##            AA  0  1  0
##            DL 22 23 37
##            UA 11 25 31
```

```
# accuary of QDA
mean(qda.class.3ca == testLabel1)
```

```
## [1] 0.36
```

```
# classification eror of QDA
mean(qda.class.3ca != testLabel1)
```

```
## [1] 0.64
```

As a result of checking the estimated mixing proportion for all three features in each class of the estimated trained QDA model, it is judged that AA is **0.1942857**, DL is **0.3571429**, UA is **0.445714**. When checking the estimated mean vector and classification table, it is unlikely that standardized observations belong to each of the three carriers.
In addition, as a result of correcting the accuracy and error rate, it is judged that the **QDA performance of this test set is not very good with 36% accuracy and 64% error rate**.

(7.3c) Extract observations that are for "UA" or "DL" from the training set and the test set, respectively, to form a new training set and a new subset, so that there are now 2 classes "UA" and "DL". Apply QDA to the new training set and then apply the trained model to the new test set. Report the overall error rate on the test set, provide the ROC curve, and calculate the AUC. How is the performance of QDA on this test set? Explain your answer.

```r
# extract observations only carrier are "UA" or "DL" from dat1
dat2 <- dat1 %>%
   dplyr::filter(carrier == c("UA", "DL"))
# standardize the feature
dat22 <- scale(dat2[,2:4])
set.seed(123)
# 70 % of training set and 30% of test set and set labels
train2 = base::sample(1:nrow(dat22), 0.7*nrow(dat22))
test2 = (1:nrow(dat22))[-train2]
trainSet2 = dat22[train2,]
testSet2 = dat22[test2,]
trainLabel2 = dat2$carrier[train2]
testLabel2 = dat2$carrier[test2]
# change the train and test sets to data frame
t2 <- data.frame(trainSet2)
te2 <- data.frame(testSet2)
# apply QDA to training set
qda.fit.2ft <- qda(trainLabel2~., data=t2)
# trained model to test set
qda.pred.2ft <- predict(qda.fit.2ft, te2)
qda.class.2ft <- qda.pred.2ft$class
# show new QDA table
table(qda.class.2ft, testLabel2)
```

```
##                testLabel2
## qda.class.2ft DL UA
##             DL 16 16
##             UA  9 19
```

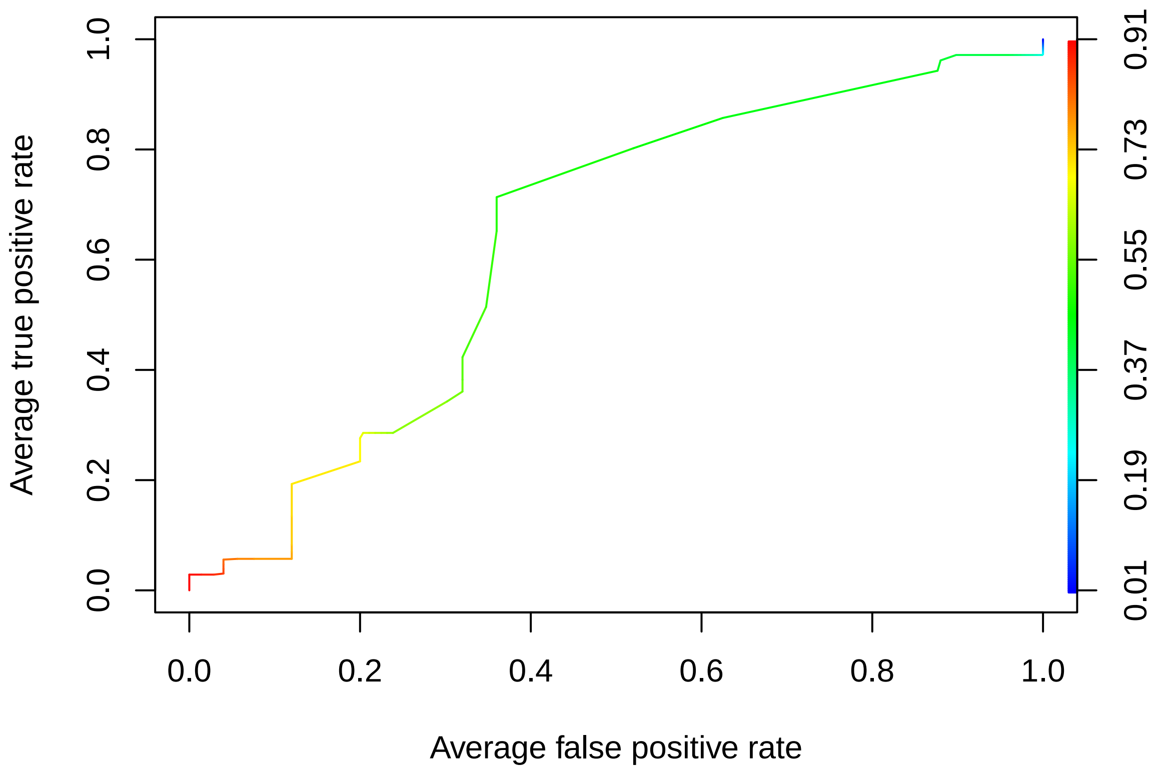
```r
# accuarcy of new QDA
mean(qda.class.2ft == testLabel2)
```

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

```r
# classfication error of new QDA
mean(qda.class.2ft != testLabel2)
```

```
## [1] 0.4166667
```

```r
# ROC curve plot
par(mfrow=c(1,1), mar=c(5,5,0,0.5), oma=c(1.5,1.5,0.5,1.5))
pred.qda.roc <- prediction(qda.pred.2ft$posterior[,2], testLabel2)
perf.qda.roc <- performance(pred.qda.roc, "tpr", "fpr")
plot(perf.qda.roc, avg="threshold",
     colorize = TRUE)
```



```r
# calculate the AUC
qda.auc <- ROCR::prediction(qda.pred.2ft$posterior[,2], testLabel2) %>%
   ROCR::performance(measure = "auc") %>%
   .@y.values
as.numeric(qda.auc)
```

```
## [1] 0.6388571
```

As a result of checking the accuracy and error rate of the observations for the two classes after QDA modeling, it can be confirmed that the accuracy is **58.33%** and the error rate is **41.67%**. Also, when looking at the ROC curve, it can be seen that the performance of this model is not so good, and the AUC is **0.6388571**. As a result, it is determined that the QDA model is not the optimal model for this test set.

Applied exercises: II (Discriminant analysis)

8. The following is on software commands:

(8.1) What is the main cause of the message "Warning in `lda.default`(x, grouping, . . .): variables are collinear"? What is the main cause of the message "Error in `qda.default`(x, grouping, . . .) : some group is too small for 'qda' "?

The main reason for the message `lda.default` (x, grouping, . . .): variables are collinear" is that, as the error message said, some variables are on the same line. In other words, it means that elements of one group are also in elements of another group.

The main reason for the message `qda.default` (x, grouping, . . .): some group is too small for 'qdq', it means not enough observations in each group to sensibly estimate the corresponding component parameters.

(8.2) Provide details on the `list` that `predict{MASS}` returns.

`predict{MASS}` returns a `list` with components:
* (1) class: it is the MAP classification.
* (2) posterior: poterior probailities for classes.

(8.3) The arguments `gamma` and `lambda` of `rda{klaR}` are usually determined by cross-validation. Can they be set manually?

Yes, If unspecified `gamma` and `lambda` in `rda{klaR`, either or both are deterimend by minimizing the estimated error rate.

9. We will use the human cancer microarray data that were discussed in the lectures and are provided by the R library `ElemStatLearn` (available at https://cran.r-project.org/src/contrib/Archive/ElemStatLearn/). Pick 3 cancer types "MELANOMA", "OVARIAN" and "RENAL", and randomly select the same set of 60 genes for each cancer type. Please use `set.seed(123)` for the whole of this exercise. Your analysis will be based on observations for these genes and cancer types.

```
# set human cancer microarray data
data(nci)
# get demensions
n = dim(nci)[2]
p = dim(nci)[1]
# randomly select the same set of 60 genes
set.seed(123)
rowSel = sample(1:p, size=60, replace = FALSE)
# pick 3 cancer types MELANOMA, OVARIAN, and RENAL
cct = colnames(nci) %in% c("MELANOMA", "OVARIAN", "RENAL")
# each column name is cancer type for obs
colSel = which(cct == TRUE)
# take genes from rows of nci and samples from column of nci
nci_set = nci[rowSel, colSel]
colnames(nci_set) = colnames(nci)[colSel]
```

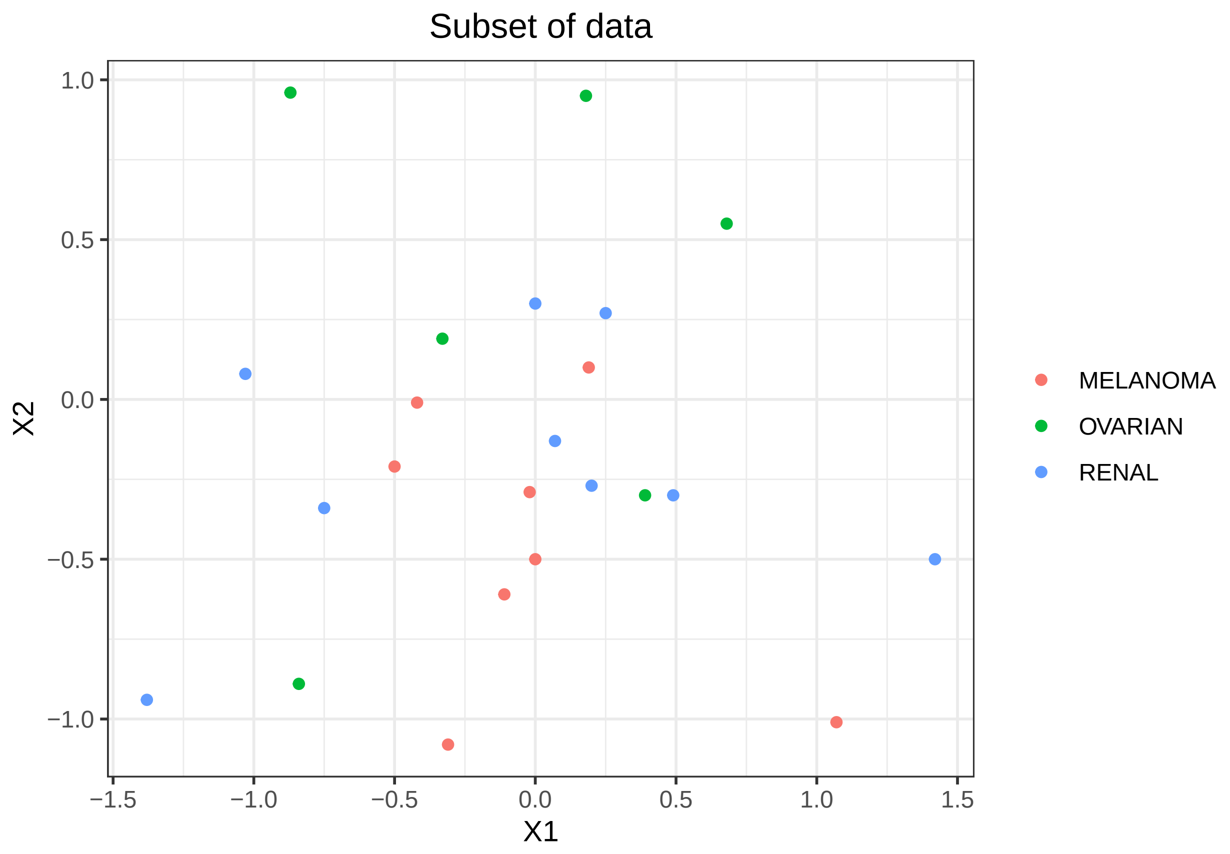
Three cancer types were selected, 60 random genes were extracted, and the nci_set variable was generated.

9.1) Pick 2 features and visualize the observations using the 2 features. Do you think it is hard to classify the observations based on the amount of overlap among the 3 neighborhoods of observations in each of the 3 classes? Here "a neighborhood of observations in a class" is a "open disk that contains the observations in the class".

```r
# transposed subset data as temp
temp = data.frame(t(nci_set))
temp$Class = colnames(nci_set)
rownames(temp) = NULL
table(temp$Class)
```

```
##
## MELANOMA  OVARIAN    RENAL
##        8        6        9
```

```r
# visualize subset of data
plotOver <- ggplot(data=temp, aes(x=X1,y=X2)) +
   geom_point(aes(color = as.factor(Class))) +
   theme_bw() +
   theme(plot.title = element_text(hjust=0.5),
         legend.title = element_blank()) +
   labs(title = "Subset of data")
plotOver
```



As a result of visualization, it is not judged that it is difficult to classify observations because the degree of overlap between the three adjacent observations does not overlap.

9.2) Apply LDA and report the classwise error rate for each cancer type.

```
lda.cc <- lda(Class~., data=temp)
lda.cc.pred <- predict(lda.cc, temp)
TrueClassL.cc <- temp$Class
LDAEstClassL.cc <- lda.cc.pred$class
table(LDAEstClassL.cc, TrueClassL.cc)
```

```
##                TrueClassL.cc
## LDAEstClassL.cc MELANOMA OVARIAN RENAL
##        MELANOMA        6       0     0
##        OVARIAN         0       5     0
##        RENAL           2       1     9
```

```
# accuracy
mean(LDAEstClassL.cc==TrueClassL.cc)
```

```
## [1] 0.8695652
```

```
# error rate
mean(LDAEstClassL.cc!=TrueClassL.cc)
```

```
## [1] 0.1304348
```

As a result of checking the accuracy and error rate for each cancer type after generating the LDA model, it is determined that the model is suitable with **86.96 %** accuracy and **13.04 %** error rate.

9.3) Use the library `klaR`, and apply regularized discriminant analysis (RDA) by setting the arguments `gamma` and `lambda` of `rda{klaR}` manually so that the resulting classwise error rate for each cancer type is zero.

```
# apply rda using klaR; gamma = 0.05, lambda = 0.2 (manually)
rda.cc <- rda(Class~., data=temp, gamma = 0.05, lambda = 0.2)
rda.cc.pred <- predict(rda.cc, temp)
TrueClassL.cc.rda <- temp$Class
rDAEstimtedClassL.cc <- rda.cc.pred$class
table(rDAEstimtedClassL.cc, TrueClassL.cc.rda)
```

```
##                     TrueClassL.cc.rda
## rDAEstimtedClassL.cc MELANOMA OVARIAN RENAL
##             MELANOMA        8       0     0
##             OVARIAN         0       6     0
##             RENAL           0       0     9
```

```
# accuracy
mean(rDAEstimtedClassL.cc==TrueClassL.cc.rda)
```

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

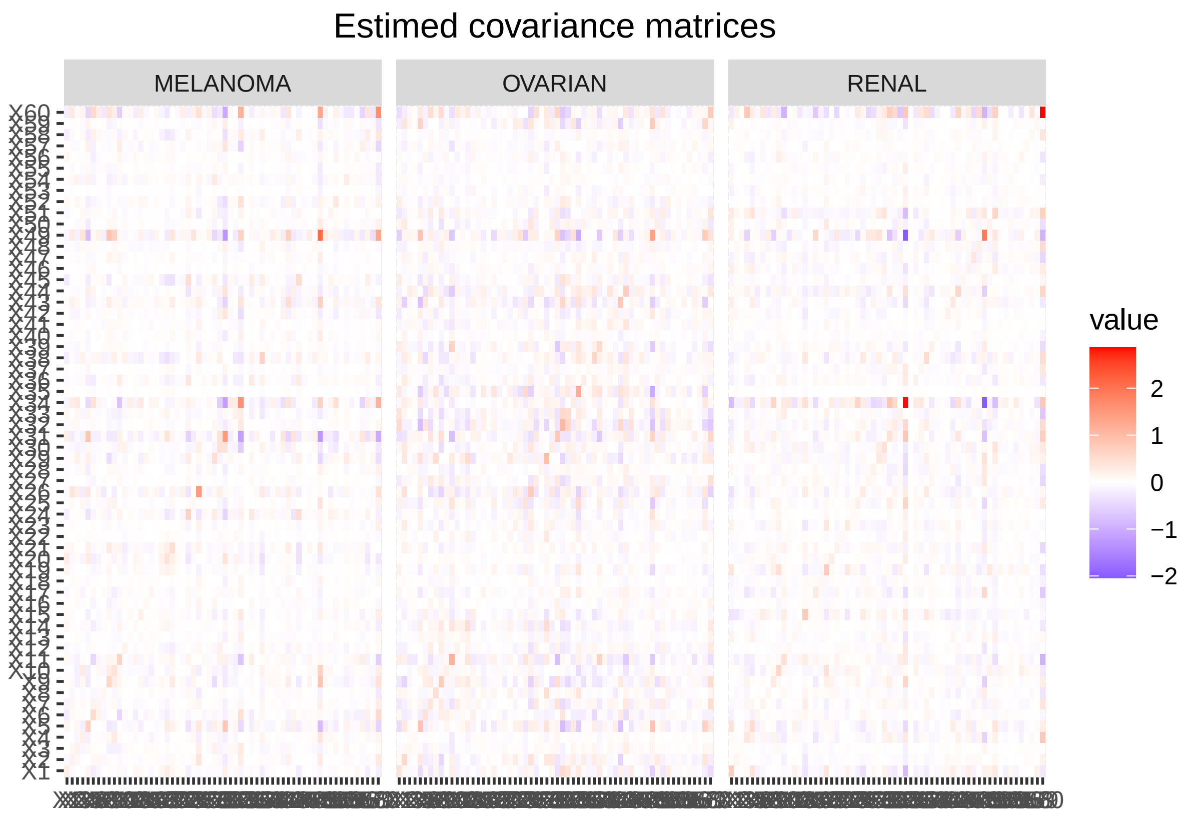
```
# error rate
mean(rDAEstimtedClassL.cc!=TrueClassL.cc.rda)
```

```
## [1] 0
```

It can be confirmed that the `gamma` and `lambda` arguments are manually set so that the error rate for each grade by cancer type is zero.

9.4) Obtain the estimated covariance matrices from the RDA and visualize them using the same strategy in Example 3 in "LectureNotes5c_notes.pdf". What can you say about the degree of dependence among these genes for each of the three cancer types? (Hint and caution: the class labels "MELANOMA", "OVARIAN" and "RENAL" will be ordered alphabetically by R. So, you need to keep track on which estimated covariance matrix is for which class. Otherwise, you will get wrong visualization.)

```
# extract estimated covariance matrices
hatSigma.cc1 = rda.cc$covariances[,,1]
hatSigma.cc2 = rda.cc$covariances[,,2]
hatSigma.cc3 = rda.cc$covariances[,,3]
# convert estimated covariacne matrices for plotting
melted_cc1 = melt(hatSigma.cc1)
melted_cc2 = melt(hatSigma.cc2)
melted_cc3 = melt(hatSigma.cc3)
EstSigma.cc <- rbind(melted_cc1,
                     melted_cc2,
                     melted_cc3)
# the class labels will be ordered alphabetically
EstSigma.cc$Cancer = rep(c("MELANOMA",
                           "OVARIAN",
                           "RENAL"),
                         each=nrow(melted_cc1))
# change cancer type to factor
EstSigma.cc$Cancer = factor(EstSigma.cc$Cancer)
# plot covariance matrices
cmplot <- ggplot(data=EstSigma.cc, aes(x=Var1, y=Var2, fill=value)) +
   geom_tile() +
   scale_fill_gradient2(low="blue", high="red") +
   facet_grid(~Cancer) +
   xlab("") +
   ylab("") +
   ggtitle("Estimed covariance matrices") +
   theme(plot.title = element_text(hjust = 0.5))
cmplot
```



As a result of visualizing the estimated covariance matrix, some gene expressions with a correlation higher than the absolute value 1 in `RENAL`. There are also some gene expressions in `MELANOMA` that are higher than the absolute value of 1. These gene expressions can be close to singularities, so it is considered good to remove them. All of the remaining `OVERIAN` gene expressions are most highly dependent on genes because the sample correlation exists between absolute values 1 than the other cancers.