Seoul National University
Dept. of Civil and Environmental Engineering

Instructor: Junho Song
junhosong@snu.ac.kr

M1586.002500 Information Engineering for CE Engineers

In-Class Material: Class 11

Classification (ISL Chapter 4)

1 Quadratic Discriminant Analysis (QDA)

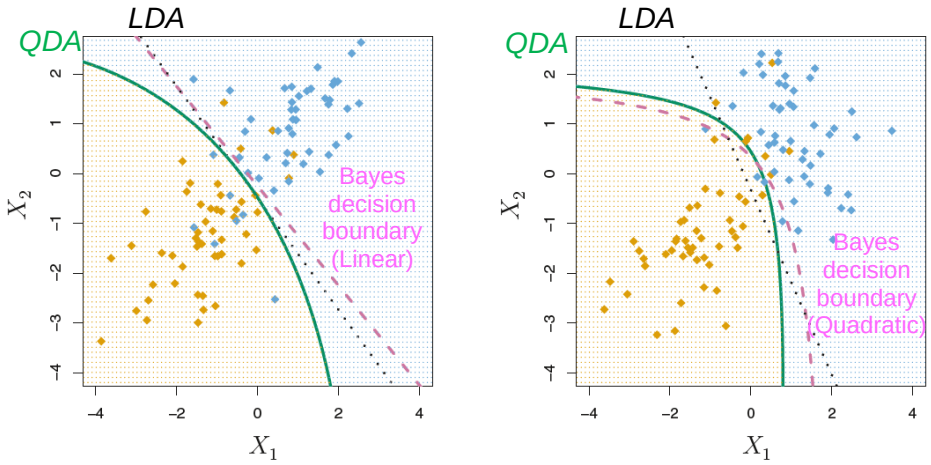
(a) **Assumptions**: the data within each class are normally distributed with different covariances $\mathbf{\Sigma}_k$

$$f_k(x) = \frac{1}{(2\pi)^{p/2}|\mathbf{\Sigma}_k|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu_k)^T \mathbf{\Sigma}_k^{-1}(x-\mu_k)\right)$$

(b) Discriminant functions

$$\delta_k(x) = -\frac{1}{2}x^T\mathbf{\Sigma}_k^{-1}x + \mu_k^T\mathbf{\Sigma}_k^{-1}\mu_k - \frac{1}{2}\mu_k^T\mathbf{\Sigma}_k^{-1}\mu_k - \frac{1}{2}log|\mathbf{\Sigma}_k| + log(\pi_k)$$

Note: "Discriminant functions" are "q_____" functions of **x**



(c) QDA performs better than LDA when:
- Common-covariance assumption is violated
- Training data set is very large (flexible but high variance → bias-variance trade-off)

```
library(ISLR) # ISLR library for stock market data
library(MASS)
attach(Smarket) # add variables in 'Smarket' to the search path
train = (Year < 2005)
Smarket.2005 = Smarket[!train, ] # use only dataset from 2005
Direction.2005 = Direction[!train]
qda.fit=qda(Direction~Lag1+Lag2,data=Smarket,subset=train)
   # Fit the Gaussian distribution and find QDA boundary
qda.fit # Show information about the model
qda.class=predict(qda.fit,Smarket.2005)$class
   # Predicted results for train set (UP/DOWN)
table(qda.class,Direction.2005) # Confusion matrix
mean(qda.class==Direction.2005) # Overall accuracy (LDA: 56%)
```

Instructor: Junho Song
junhosong@snu.ac.kr

2 Comparison between Classification Methods

(a) Two-class problem for which, $p_2(x) = 1 - p_1(x)$

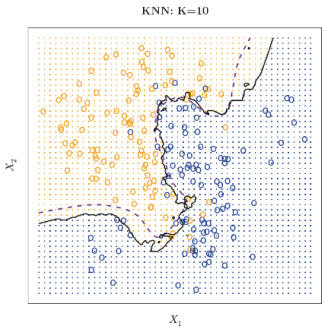
LDA: (Recall $p_k(x) = \pi_k \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(x-\mu_k)^2}{2\sigma^2}\right) / \sum_{l=1}^{K} \pi_l \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(x-\mu_l)^2}{2\sigma^2}\right)$)

$$\log\left(\frac{p_1(x)}{1-p_1(x)}\right) = c_0 + c_1 x$$

logistic regression:

$$\log\left(\frac{p_1(x)}{1-p_1(x)}\right) = \beta_0 + \beta_1 x$$

→ both produce the linear decision boundaries, only **the fitting methods** are different



(b) (Recall) **K-nearest neighbor (KNN)**

Probability that an observation x belongs to the kth class :

$$\Pr(Y = k|X = x) = \frac{1}{K} \sum_{i \in \mathcal{N}_0} I(y_i = j)$$

```
library(class) # class library for knn function
train.X=cbind(Lag1,Lag2)[train,] # Train data set: variables
test.X=cbind(Lag1,Lag2)[!train,] # Test data set: variables
  # NOTE: KNN is only for the  "test" data set

train.Direction=Direction[train] # Train data set: responses

set.seed(1)
knn.pred=knn(train.X,test.X,train.Direction,k=3)
   # Predicted results for the test set (UP/DOWN) with K=3
table(knn.pred,Direction.2005) # Confusion matrix
mean(knn.pred==Direction.2005) # Overall accuracy
```

(c) Comparison of Classification methods

No method will dominate the others in every situation

→ Depends on the true decision boundaries

	Logistic regression	**LDA**	**QDA**	**KNN**
Decision Boundary	Linear	Linear	Quadratic	Arbitrary → *Flexible*
Assumption	$p(x)$ is logistic function	$f_k(x)$ are Gaussian with common variance	$f_k(x)$ are Gaussian distributions	
Fitting method	Maximum likelihood	Gaussian parameter estimation	Gaussian parameter estimation	Non-parametric

Seoul National University
Dept. of Civil and Environmental Engineering

Instructor: Junho Song
junhosong@snu.ac.kr

M1586.002500 Information Engineering for CE Engineers

In-Class Material: Class 12

Resampling Methods (ISL Chapter 5)

Problem: The size of test set is not large enough to estimate the test error rate directly.

Question: How can we estimate the test error rate with the available training data set?

→ **"Resampling Methods"** e.g. cross-validation, the bootstrap methods

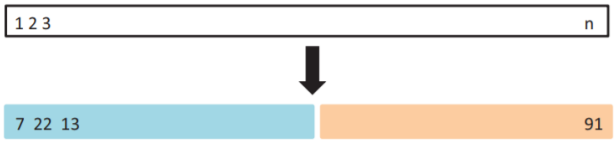
1. Validation Set Approach

(a) A very simple strategy validation for estimating the test error.

(b) **Randomly dividing** the available set of observations into two parts, ***a training set*** and ***a validation set*** or *hold-out set*.

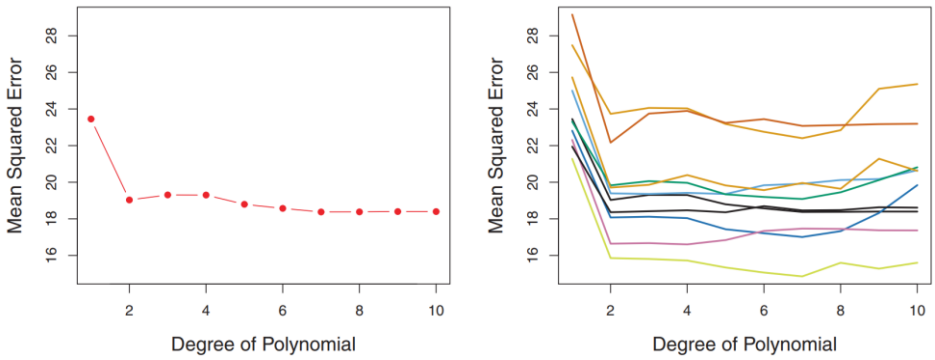
- Training set: The model is fit on the training set.
- Validation set (Hold-out set): The fitted model is used to predict the responses for the observations in the validation set.

→ The resulting validation set error rate provides an estimate of the test error rate.



(c) Two potential drawbacks:

- The validation estimate of the test error rate can be highly v_____, depending on the observations which are divided into the "training set" and "validation set."
- Since only a subset of the observations is used to fit the model, the validation set error rate may tend to o_____ the test error rate as compared to the model fit on the entire data set (★)



Instructor: Junho Song
junhosong@snu.ac.kr

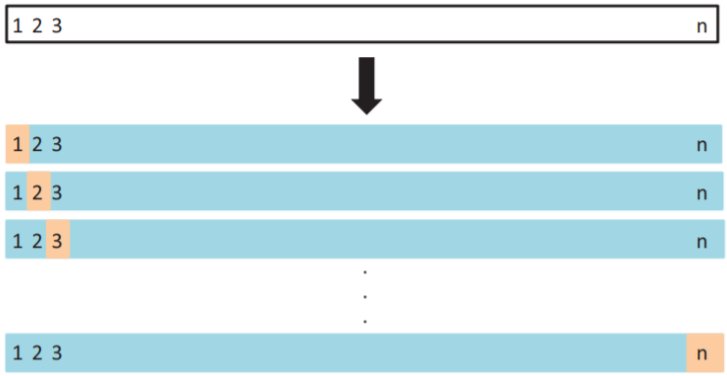
```
library(ISLR) # Load ISLR Library for Auto data
attach(Auto)
MSE = rep(0,10) # Mean Squared Error of 10 Validations
for (i in 1:10){
set.seed(i) # See number for random number generation
train=sample(392,196) # Randomly choose 196 among 392
lm.fit=lm(mpg~horsepower,data=Auto,subset=train)
MSE[i]=mean((mpg-predict(lm.fit,Auto))[-train ]^2) # Compute Test MSE
}
```

2. Leave-One-Out Cross Validation (LOOVC)

(a) LOOCV is a very general method, and can be used with any kind of predictive modeling (e.g. logistic regression, linear discriminant analysis). Closely related to the validation set approach, but it attempts to address the method's drawbacks.

(b) "a single" observation (x_1, y_1) is used for the validation set, and the remaining observations $\{(x_2, y_2), \ldots, (x_n, y_n)\}$ make up the training set.

- The statistical learning method is fit on the (n–1) training observations, and a prediction $\hat{y}_1$ is made for the excluded observation, using its value x_1.
- $\mathrm{MSE}_1 = (y_1 - \hat{y}_1)^2$ provides an "approximately unbiased" but "highly variable" estimate for the test error.



(c) Repeating this approach n times produces n squared errors, $\mathrm{MSE}_1, \ldots, \mathrm{MSE}_n$. The LOOCV estimate for the test MSE is the average of these n test error estimates:

$$\mathrm{CV}_{(n)} = \frac{1}{n}\sum_{i=1}^{n} \mathrm{MSE}_i$$

Note: If n is very large and if each individual model is slow to fit, LOOCV has the potential to be expensive to implement. For some of those cases (least squares linear or polynomial regression), the following formula could be applied as a shortcut:

$$\mathrm{CV}_{(n)} = \frac{1}{n}\sum_{i=1}^{n} \left(\frac{y_i - \hat{y}_i}{1 - h_i}\right)^2$$

Instructor: Junho Song
junhosong@snu.ac.kr

where $h_i = \frac{1}{n} + \frac{(x_i - \bar{x})^2}{\sum_{i'=1}^{n}(x_{i'} - \bar{x})^2}$ is the 'leverage' (textbook page.98). This is like the ordinary MSE except the $(1 - h_i)$ term. The cost of the shortcut is the same as that of a single fit.

(d) Two major advantages of LOOCV as compared to the validation set approach:

- First, it has far less bias. In LOOCV, we repeatedly fit the statistical learning method using training sets that contain (*n*–1) observations. Hence, the LOOCV approach tends *not* to overestimate the test error rate as much as the validation set approach.
- Second, performing LOOCV multiple times will always yield the same results: there is *no* randomness in the training/validation set splits.

```
# Leave-One-Out Cross Validation (LOOCV) for linear regression
library(boot) # load 'boot' library for cross validation (cv.glm)
glm.fit=glm(mpg~horsepower,data=Auto) # use 'glm' instead of 'lm' to use
'cv.glm'
cv.err=cv.glm(Auto,glm.fit) # no value for 'K' means LOOCV by default
cv.err$delta # cross validation estimates - raw and adjusted (correct bias
when NOT using LOOCV)

# Leave-One-Out Cross Validation (LOOCV) for polynomial regression
cv.error_LOOCV=rep(0,10)
for (i in 1:10){
  glm.fit=glm(mpg~poly(horsepower,i),data=Auto)
  cv.error_LOOCV[i]=cv.glm(Auto,glm.fit)$delta[1]}
# LOOCVs for up to 10th order polynomial regression
```

3. *k*-Fold Cross Validation

(a) An alternative to LOOCV is *k*-fold CV. This approach involves randomly dividing the set of observations into *k* groups, or *folds*, of approximately equal size. LOOCV is a special case of *k*-fold CV in which *k* is set to equal *n*.

(b) In practice, *k*-fold CV using *k* = 5 or 10 are often performed. As compared to LOOCV, *k*-fold CV have obvious computational advantage when *k* < *n*.

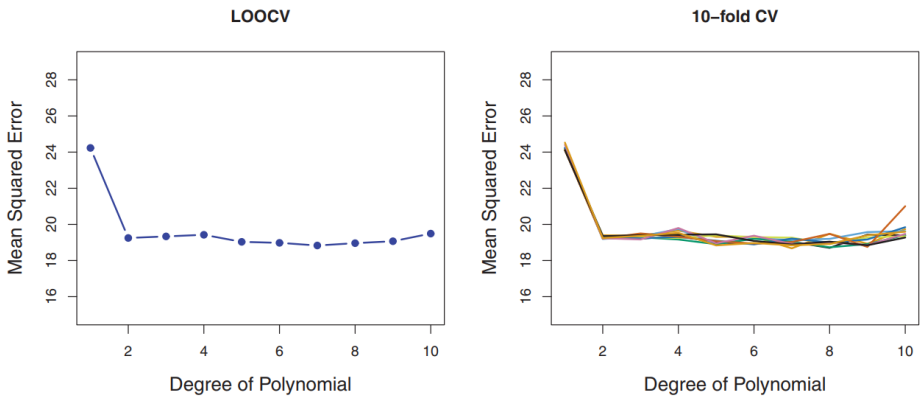


Figure illustrates the estimated MSE that results from the validation set approach using LOOCV and k-fold CV approach: (Left) The LOOCV error curve; (Right) 10-fold CV was run nine separate times, with a different random split.

Instructor: Junho Song
junhosong@snu.ac.kr

```
# k-Fold Cross Validation
set.seed(17)
cv.error_K10.10=rep(0,10)
for (i in 1:10){
  glm.fit=glm(mpg~poly(horsepower,i),data=Auto)
  cv.error_K10.10[i]=cv.glm(Auto,glm.fit,K=10)$delta[1]} # k-fold CV

# Comparison between LOOCV and k-Fold Cross Validation (k=10)
plot(1:10,cv.error_LOOCV,type='b',lty='dotdash',pch=1,ylab='CV',xlab='Order
of Polynomial Regression')
lines(1:10,cv.error_K10.10,type='b',pch=8,col='red')
```

4. Bias-Variance Trade-Off for *k*-Fold Cross Validation

In using a statistical learning procedure, we must consider not only "bias" but also procedure's "variance." Recall

$$\begin{aligned}\mathrm{E}\left[\left(y-\hat{f}\right)^2\right] &= \left(f-\mathrm{E}[\hat{f}]\right)^2+\mathrm{E}\left[\left(\hat{f}-\mathrm{E}[\hat{f}]\right)^2\right]+\mathrm{Var}(\epsilon)\\ &= \left[\mathrm{Bias}(\hat{f})\right]^2+\mathrm{Var}(\hat{f})+\mathrm{Var}(\epsilon)\end{aligned}$$

(a) Even though LOOCV uses larger training set, *k*-fold CV often gives *more accurate* estimates of test error rate than LOOCV. This is because of *bias-variance trade-off*. Test error rate estimated using *k*-fold CV (when *k*=5 or 10) suffer neither from excessively high *bias* nor from very high variance.

(b) "Bias" when using 2, *k* and *n* groups for cross validation (2 < *k* < *n*)

- Number of observations: Validation set approach < *k*-fold CV < LOOCV
- Bias from estimation procedure: Validation set approach > *k*-fold CV > LOOCV (same reasoning with ★)

(c) Then, why LOOCV has larger "variance" than *k*-fold CV?

- When we perform LOOCV, we are in effect averaging the outputs of *n* fitted models, each is trained on an almost identical and *highly correlated* set of observations.
- In contrast, when we perform k-fold CV with *k*<*n*, overlap between the training sets in each model is smaller than LOOCV.

➢ The mean of many *highly correlated* quantities has *higher variance* than does the mean of many quantities that are not as highly correlated.

Variance from estimation procedure: *k*-fold CV < LOOCV

Alternative explanation: LOOCV relies more on the training data set than *k*-fold CV → Tries to capture the data-set-specific trends and characteristics more than *k*-fold CV

Instructor: Junho Song
junhosong@snu.ac.kr

5. Cross-Validation on Classification Problems

(a) Cross-validation can also be a very useful approach in the classification setting when *Y* is "qualitative." The number of misclassified observations is used to quantify test error. For instance, in the classification setting, the LOOCV error rate takes the form

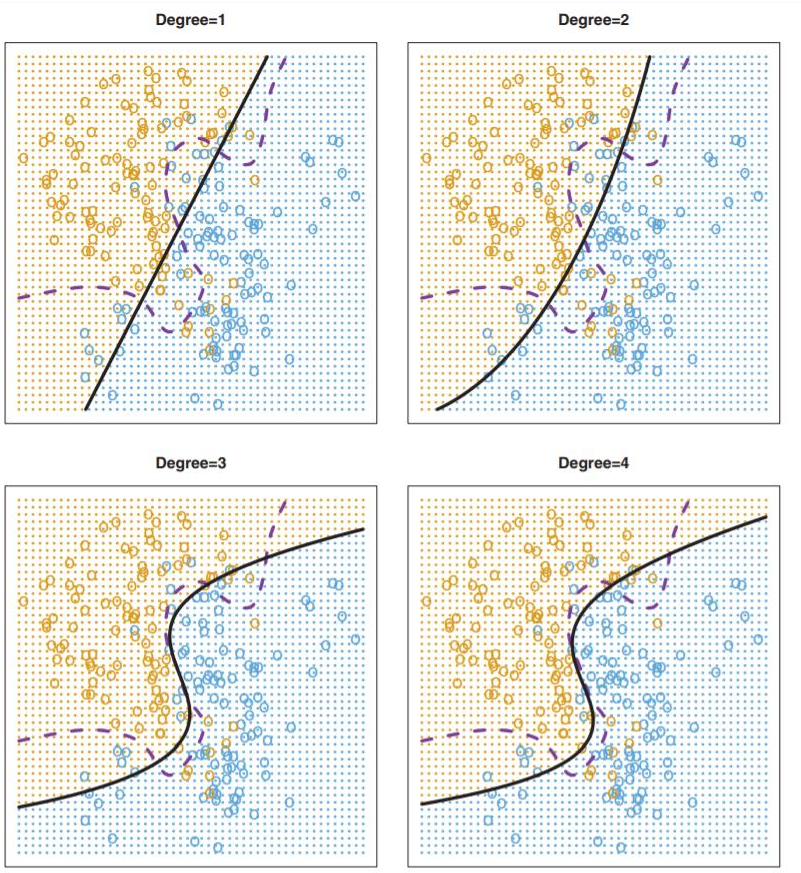
$$\mathrm{CV}_{(n)} = \frac{1}{n}\sum_{i=1}^{n} \mathrm{ERR}_i$$

where $\mathrm{ERR}_i = I\,(y_i \neq \hat{y}_i)$. The *k*-fold CV error rate and validation set error rates are defined similarly.

(b) In practice, for real data, the Bayes decision boundary and the test error rates are "*unknown*." So we can use "cross-validation" in order to decide better logistic regressions models for classification.

(c) For example, consider logistic regression models

- Logistic regression: $\log\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 X_1 + \beta_2 X_2$
- Quadratic logistic regression: $\log\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 X_1 + \beta_2 X_1^2 + \beta_3 X_2 + \beta_4 X_2^2$
- 3rd order, 4th order, and so on.



Instructor: Junho Song
junhosong@snu.ac.kr

10-fold CV error helps find the near-optimal level of fitting, i.e. the level giving near-minimal test error (without using a separate test data set)

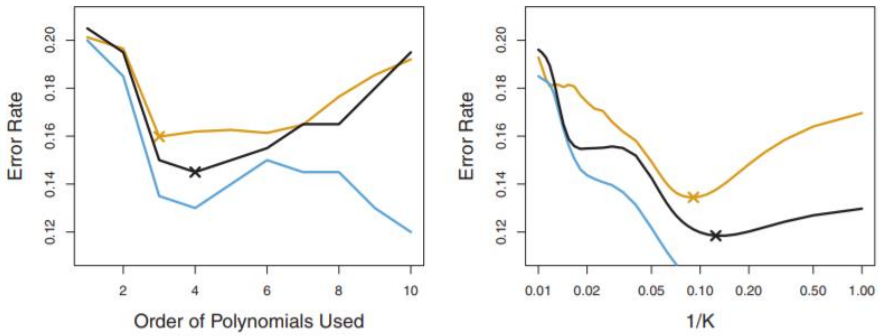


FIGURE 5.8. *Test error (brown), training error (blue), and 10-fold CV error (black) on the two-dimensional classification data displayed in Figure 5.7.* Left: *Logistic regression using polynomial functions of the predictors. The order of the polynomials used is displayed on the x-axis.* Right: *The KNN classifier with different values of K, the number of neighbors used in the KNN classifier.*