Yonsei University, Department of Applied Statistics

추가 자료 조사

신보현 2014122016

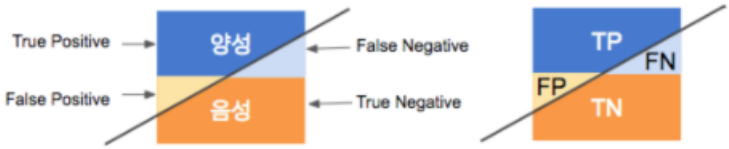
January 28, 2019

LDA를 공부하며 여러 궁금증이 생겼다. 우선 분류 문제에서 Confusion Matrix, ROC Curve가 많이 나오는 것 같고 이 개념은 다른 곳에서도 많이 쓰이는데 아직도 확실히 알지 못했다. 그래서 이번 기회에 확실히 정리를 하고 넘어가고자 한다. 또한 LDA, QDA가 각 클래스의 분포가 정규분포(또는 가우시안)라는 가정하에서 출발하는데, 그렇다면 이러한 정규 분포 가정을 어떻게 확인을 하는지가 궁금해졌다. 현재 알고있는 사항은 회귀분석에서 오차의 정규성을 알아볼 때, 잔차에 대하여 shapiro.test을 하는 것 정도인데, 또 어떤 것들이 있는지 궁금해졌다. 추가적으로 정규성 검정을 했는데 정규분포와 멀다는 결론이 났다면, 이러한 데이터를 어떻게 해서 정규 분포와 비슷하게 만드는지 궁금해졌다. 하나씩 살펴보자.

1. Confusion Matrix. 분류문제에서 많이 나오는 개념.

어떤 사람이 신용 불량자인지 아닌지를 판별하는 문제를 생각해보자. 이를 위해 모델을 설계했고 정확도가 100% 라면 아래 그림과 같을 것이고 여기서 양성으로 예측이 된 영역을 positive prediction, 음성으로 예측된 영역을 negative prediction이라고 한다.

하지만 실제 에서는 이렇게 완벽하게 분류해줄 수는 없을 것이다. 즉, 아래와 같이 틀린 부분이 발생할 수밖에 없다.



용어를 정리하면,
TP(True Positive): 양성인데 양성으로 제대로 검출된 것.
TN(True Negative): 음성인데 음성으로 제대로 검출된 것.
FN(False Negative): 양성인데 음성으로 잘못 검출된 것.
FP(False Positive): 음성인데 양성으로 잘못 검출된 것.

보통 아래 표와 같이 표현한다.

		Predicted		
		Positive	Negative	
Observed	Positive	TP	FN	P
	Negative	FP	TN	N

여기서 P는 양성인 전체 개수, 즉 이 안에는 양성인데 양성으로 제대로 검출된 것(TP)과 양성인데 음성으로 잘못 검출된 것이 있을 것(FN)이다.
같은 맥락으로 N은 음성인 전체 개수이며 이 안에는 음성인데 음성으로 제대로 검출된 것(TN)과 음성인데 양성으로 잘못 검출된 것(FP)가 있을 것이다.
쉽게 외우기 위해서 각 글자의 앞 글자, T와 N이 우리의 예측이 맞는지, 틀리는지를 나타내고 P와 F가 우리 양성 또는 음성으로 예측했는지를 나타내는 지표라고 생각하자.
여기서 우리는 두 개념을 확실히 구분해야 한다. 환자의 질병에 대해 양성인지 음성인지에 대한 정답(실제 label)은 이미 올바른 진단법(이를 gold standard라고 한다.)에 의해 알려져 있는 상태고, 새로운 진단법이 이를 양성인지, 음성인지를 예측하는 상황이다. 즉, 이미 올바른 진단법이 있는 상황에서 새롭게 개발된 진단법이 그에 준하는, 또는 그보다 더 성능이 좋은지 판단하기 위해서 위의 개념들이 만들어진 것이다.

이 값을 기반으로 지표를 만들어 분류 모델에 대한 평가를 할 때 사용한다.

- Accuracy
  전체 데이터 중에서, 제대로 분류된 데이터의 비율이다.

$$ACC = \frac{TP + TN}{P + N}$$

  모델이 얼마나 정확하게 분류를 하는지 보여준다.

- Error Rate
  전체 데이터 중에서 잘못 분류한 데이터의 비율이다.

$$ERR = \frac{FN + FP}{P + N}$$

- Sensitivity(Recall or True positive Rate) = TPR(True Positive Rate)
  양성 데이터 중 양성으로 잘 분류된 데이터의 비율이다.

$$SN = \frac{TP}{TP + FN}$$

  이는 모델이 얼마나 정확하게 양성을 잘 찾아내는 지를 보여준다.

- Precision
  양성으로 예측한 것 중에 실제로 양성인 데이터의 비율이다.

$$PREC = \frac{TP}{TP + FP}$$

- TPR(True Positive Rate) = Sensitivity
  실제 양성인 것들 중에서 양성으로 분류한 비율이다.

$$TPR = \frac{TP}{TP + FN}$$

- TNR(True Negative Rate) = Specificity
  실제 음성인 것들 중에서 음성으로 분류한 비율이다.

$$TNR = \frac{TN}{TN + FP}$$

- FPR(False Positive Rate)
  실제로 음성인데 양성으로 예측한 비율

$$FPR = \frac{FP}{FP + TN}$$

- FNR(False Negative Rate)
  실제로 양성인데 음성으로 예측한 비율

$$FNR = \frac{FN}{FN + TP}$$

민감도와 특이도는 서로 상충 관계에 있다. 민감도를 줄이려고 하면 특이도는 올라가고 특이도를 줄이려고 하면 민감도는 올라간다. 극단적인 예를 생각해보자. 모든 환자를 양성으로 분류하는 진단법을 생각해보자. 이러한 진단법의 민감도는 1이고 특이도는 0이다. 또는 모든 환자를 음성으로 분류하는 진단법을 생각해보자. 이러한 진단법의 민감도는 0이고 특이도는 1일 것이다. 이렇게 서로 상충 관계에 있는 민감도와 특이도 사이에 적절한 지점을 찾아야 한다.
상황에 따라 민감도와 특이도는 조절이 될 수 있다. 예를 들어 심각한 질병의 유무를 판단하는 진단법에서는, 질병이 없는 사람을 질병이 있다고 판단하는 비율을 높여서라도, 질병이 있는 사람을 정상이라고 판단하는 비율을 최대한 줄이려고 할 것이다. 즉, 다시 말해서 민감도를 높이고 특이도를 줄이는 것이다. 병원에서 1차 진단을 받았는데 2차 검사까지 해야 한다고 나오고, 그 결과 음성 판정이 나왔다고 생각해보자. 병원 입장에서는 질병의 유무를 신중하게 판단해야 하기 때문에 민감도를 높게 잡은 것이며, 그렇게 함으로써 질병이 없는 사람도 질병이 있다고 판단되더라도, 2차 검사때 보다 정밀하게 그들을 분류하면 되기 때문에 민감도를 높게 잡은 것이다.
이제 위의 지표들로 모델을 어떻게 평가하는지 알아보자.

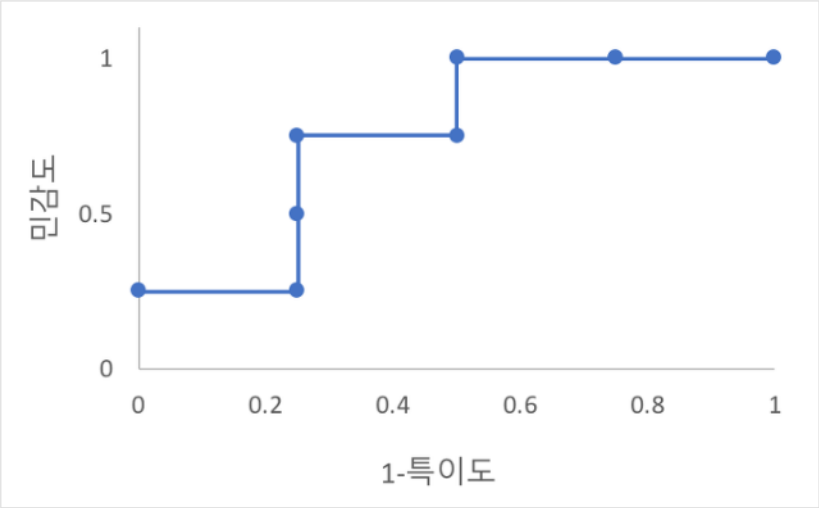
2. ROC Curve

ROC Curve는 Receiver-Operating Characteristic curve의 줄임말로, 특정 진단 방법의 민감도와 특이도가 어떤 관계를 갖고 있는지를 표현한 그래프이다. 민감도와 특이도가 어떤 의미를 가지는지 잠깐 상기해보자. 민감도는 실제 양성(질병이 있는)인 환자들 중 양성으로 판단하는 비율이고 특이도는 실제 음성(정상)인 환자들 중 음성으로 판단하는 비율이다.

민감도와 특이도는 서로 상충 관계에 있기 때문에 이를 변화시켜가며 그래프에 그려보는 것이 좋은 방법이다. 예를 들어 (혈압, 심근경색여부)에 대한 데이터가 다음과 같다고 하자.
(80/0), (92,0), (93,1), (98,0), (102,1), (110,1), (112,0), (119,1)
이 때, 1이면 심근경색이 있는 것(양성)이고 0이면 없는 것(음성)이다. 위의 경우, '혈압이 N이상이면 심근경색이다'라고 한다면 N을 90으로 잡으면 민감도가 1이 되나 특이도가 0이 된다. 따라서 N을 변화시키면서 각각의 값을 구해 보면 다음과 같다.

혈압	class	TP	TN	FP	FN	P(=TP+FN)	N(=TN+FP)	특이도	1-특이도	민감도
80	0	4	0	4	0	4	4	0	1	1
92	0	4	1	3	0	4	4	0.25	0.75	1
93	1	4	2	2	0	4	4	0.5	0.5	1
98	0	3	2	2	1	4	4	0.5	0.5	0.75
102	1	3	3	1	1	4	4	0.75	0.25	0.75
110	1	2	3	1	2	4	4	0.75	0.25	0.5
112	0	1	3	1	3	4	4	0.75	0.25	0.25
119	1	1	4	0	3	4	4	1	0	0.25

위의 테이블을 그래프로 그려보면 다음과 같다.



x축을 1-특이도로 한 이유는 이렇게 해야 우리가 익히 보아 오던 그래프 형태와 비슷하기 때문이다. 위 그래프를 보면, 민감도가 1이 되었을 때 특이도는 계속 줄어들어 결국 특이도가 0이 되는 형태를 확인할 수 있다.

AUC 구하는 방법

진단값을 오름차순으로 정렬한 후, 1-특이도가 변하는 지점에서의 sensitivity 합계를 전체 정상(negative) 수로 나누어 주면 된다. 이 때 낮은 진단값이 정상이라 가정한다. 반대일 경우 그에 맞게 적당히 변경해서 사용하면 된다. 작은 진단값에서부터 시작하여 큰 값으로 이동하면서 negative (즉, 이 경우 정상(0)) 을 만나는 점에서의 민감도값만 남겨 준다.

혈압	class	TP	TN	FP	FN	P(=TP+FN)	N(=TN+FP)	특이도	1-특이도	민감도
80	0	4	0	4	0	4	4	0	1	1
92	0	4	1	3	0	4	4	0.25	0.75	1
93	1	4	2	2	0	4	4	0.5	0.5	1
98	0	3	2	2	1	4	4	0.5	0.5	0.75
102	1	3	3	1	1	4	4	0.75	0.25	0.75
110	1	2	3	1	2	4	4	0.75	0.25	0.5
112	0	1	3	1	3	4	4	0.75	0.25	0.25
119	1	1	4	0	3	4	4	1	0	0.25

위의 경우 민감도 중 노랑색으로 칠한 값들의 합인 3 이 되겠다. 이 값을 전체 negative 수 (정상 수, 이 경우 4명)로 나누어 준다. 그러면 AUC가 0.75 가 나온다.

R로 ROC Curve를 그려보자.

```
library('pROC')
```

```
## Warning: package 'pROC' was built under R version 3.5.2
```

```
library(MASS)
head(Pima.te)
```

```
##   npreg glu bp skin  bmi   ped age type
## 1     6 148 72   35 33.6 0.627  50  Yes
## 2     1  85 66   29 26.6 0.351  31   No
## 3     1  89 66   23 28.1 0.167  21   No
## 4     3  78 50   32 31.0 0.248  26  Yes
## 5     2 197 70   45 30.5 0.158  53  Yes
## 6     5 166 72   19 25.8 0.587  51  Yes
```

데이터는 당뇨 데이터이고 type이 당뇨에 대한 확진 결과이다. 즉, type이라는 변수를 기존의 진단 방법이라고 생각하고 나머지 변수가 당뇨를 진단하는데 쓰일 새로운 지표라고 생각하면 된다. 이 새로운 지표들 중 어떤 것이 가장 좋을지 판단하기 위해서 각 변수에 대해 auc 값을 계산하여 내림차순으로 정렬한다.

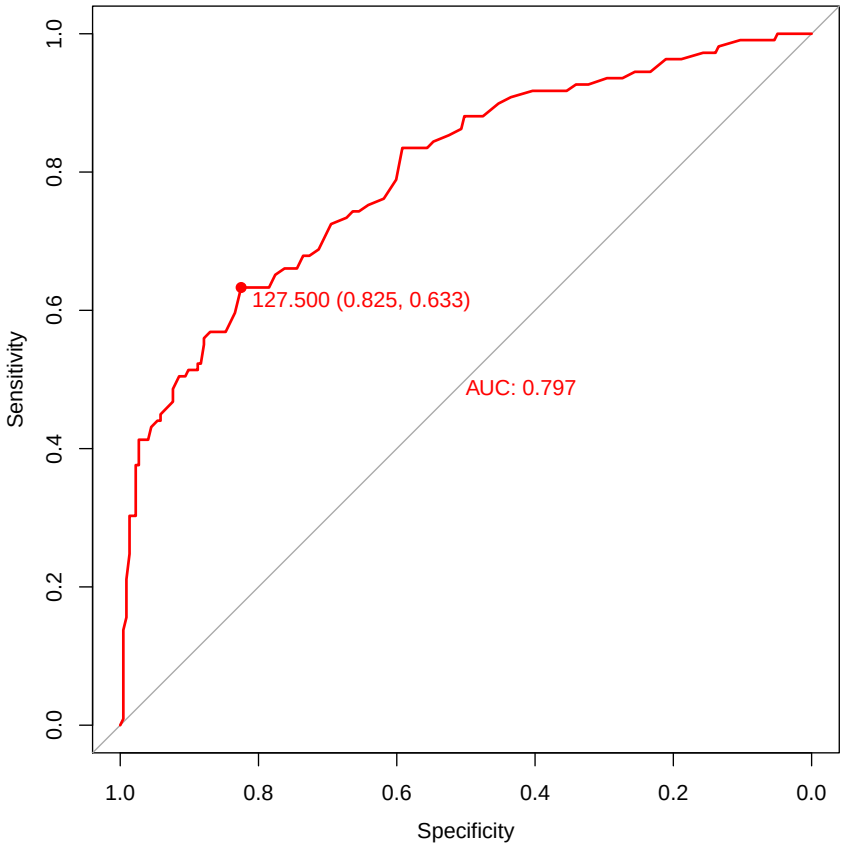
```
auc_df <- data.frame(Attribute=c(colnames(Pima.te)[1:7]), AUC=NA)
for (i in 1:nrow(auc_df)){
roc_value = roc(Pima.te$type,Pima.te[,colnames(Pima.te)[i]])
auc_df[i,"AUC"] = roc_value$auc
}
auc_df = auc_df[order(-auc_df$AUC),]
auc_df
```

```
##   Attribute       AUC
## 2       glu 0.7970543
## 7       age 0.7210886
## 5       bmi 0.6839799
## 4      skin 0.6656313
## 6       ped 0.6563541
```

```
## 1     npreg 0.6201094
## 3        bp 0.6097626
```

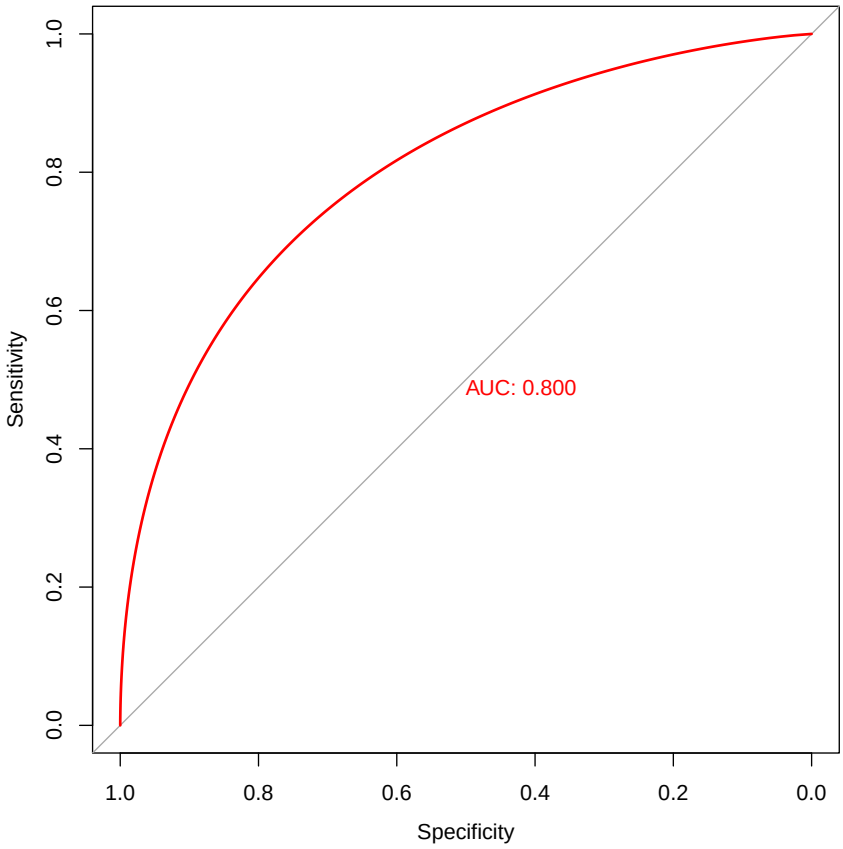
glu 변수가 AUC 값이 가장 높게 나왔다. 이제 ROC Curve를 그려보자.

```
glu_roc = roc(Pima.te$type,Pima.te$glu)
plot.roc(glu_roc,
col="red",
print.auc=TRUE,
max.auc.polyton=TRUE,
print.thres=TRUE, print.thres.pch=19, print.thres.col="red",
auc.polyton=TRUE, auc.polygon.col="#D1F2EB")
```




```
# Smooth ROC curve
plot.roc(smooth(glu_roc),
col="red",
print.auc=TRUE,
```

```
max.auc.polyton=TRUE,
auc.polyton=TRUE, auc.polygon.col="#D1F2EB")
```



plot.roc() 함수의 arguments 정보는 아래를 참조하면 된다.
https://cran.r-project.org/web/packages/pROC/pROC.pdf

3. 정규성 검정

LDA나 QDA를 시행하기 이전에, 확인해야할 가정이 있다. 각 클래스에서의 데이터 분포가 변수가 한개일 경우, 단변량 정규분포를 따라야하고, 변수가 여러개일 경우 다변량 정규분포를 따라야한다. 추가적으로 LDA는 각 클래스가 정규분포를 따를 때 동일한 분산을 가정하지만 QDA는 좀더 relax한 가정으로 서로 다른 분산을 가정한다. 따라서 LDA와 QDA을 시행하기 이전, 각 클래스에서의 데이터 분포가 정규분포를 따르는지, 그리고 등분산인지 확인을 해야 한다.

첫번째로 정규성 검정을 살펴보자. 단변수일때 사용되는 대표적인 정규성 검정을 살펴보자.

- Shapiro-Wilk test, 샤피로-윌크 검정
  귀무가설은 'H0: 정규분포를 따른다'는 것으로 p-value가 0.05보다 크면 정규성을 가정하게 된다. 다만 유의할

점은 여기서 귀무가설을 기각하지 못 했다는 것은 정규분포를 따르지 않는다고 말할 근거가 부족한 것일 뿐 100% 정규성이 만족된다는 뜻은 아니다. 참고하는 정도로 보는 것이 좋다.

- Kolmogorov-Smirnov test, 콜모고로프-스미노프 검정
  EDF, 즉 Empirical distribution function에 기반한 적합도 검정 방법이다. 자료의 평균/표준편차와 히스토그램을 표준정규분포와 비교하여 적합도를 검정한다. Shapiro-Wilk test와 마찬가지로 p-value가 0.05보다 크면 정규성을 가정하게 된다.
- Anderson-Darling tests: ad.test
- Cramer-Von Mises test: cvm.test
- Lilliefors (Kolmogorov-Smirnov) test: lillie.test
- Pearson chi-square: pearson.test
- Shapiro-Francia test: sf.test

```
library(nortest) # for univariate normality test
```

```
## Warning: package 'nortest' was built under R version 3.5.2
```

```
data(trees)
x = trees[,1]
shapiro.test(x)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  x
## W = 0.94117, p-value = 0.08893
```

```
ks.test(x,"pnorm",mean=mean(x),sd=sd(x))
```

```
## Warning in ks.test(x, "pnorm", mean = mean(x), sd = sd(x)): ties should not be present for
the Kolmogorov-Smirnov test
```

```
##
##  One-sample Kolmogorov-Smirnov test
##
```

```
## data:  x
## D = 0.14143, p-value = 0.5647
## alternative hypothesis: two-sided
```

```
ad.test(x)
```

```
##
##  Anderson-Darling normality test
##
## data:  x
## A = 0.7455, p-value = 0.04668
```

```
cvm.test(x)
```

```
##
##  Cramer-von Mises normality test
##
## data:  x
## W = 0.12828, p-value = 0.04353
```

```
lillie.test(x)
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  x
## D = 0.14143, p-value = 0.1179
```

```
pearson.test(x)
```

```
##
##  Pearson chi-square normality test
##
## data:  x
## P = 7.9677, p-value = 0.158
```

```r
sf.test(x)
```

```
##
##  Shapiro-Francia normality test
##
## data:  x
## W = 0.94622, p-value = 0.1113
```

다음으로 다변수일때 사용되는 검정을 살펴보자.

```r
library(MVN) # for multivariate normality test
```

```
## Warning: package 'MVN' was built under R version 3.5.2
```

```
## sROC 0.1-2 loaded
```

```r
mvn(trees,mvnTest="mardia")
```

```
## $multivariateNormality
##              Test          Statistic            p value Result
## 1 Mardia Skewness   20.9802564628517 0.0212316600572144     NO
## 2 Mardia Kurtosis -0.163815893754234  0.869876079301746    YES
## 3             MVN               <NA>               <NA>     NO
##
## $univariateNormality
##           Test  Variable Statistic   p value Normality
## 1 Shapiro-Wilk   Girth      0.9412    0.0889    YES
## 2 Shapiro-Wilk  Height      0.9655    0.4034    YES
## 3 Shapiro-Wilk  Volume      0.8876    0.0036    NO
##
## $Descriptives
##         n     Mean   Std.Dev Median  Min  Max  25th  75th       Skew
## Girth  31 13.24839  3.138139   12.9  8.3 20.6 11.05 15.25  0.5010559
## Height 31 76.00000  6.371813   76.0 63.0 87.0 72.00 80.00 -0.3568773
## Volume 31 30.17097 16.437846   24.2 10.2 77.0 19.40 37.30  1.0132739
##          Kurtosis
## Girth  -0.7109412
```

```
## Height -0.7233677
## Volume  0.2460393
```

```
mvn(trees,mvnTest="royston")
```

```
## $multivariateNormality
##      Test        H   p value MVN
## 1 Royston 8.331369 0.0170622  NO
##
## $univariateNormality
##           Test  Variable Statistic   p value Normality
## 1 Shapiro-Wilk   Girth      0.9412    0.0889    YES
## 2 Shapiro-Wilk  Height      0.9655    0.4034    YES
## 3 Shapiro-Wilk  Volume      0.8876    0.0036    NO
##
## $Descriptives
##         n     Mean   Std.Dev Median  Min  Max  25th  75th       Skew
## Girth  31 13.24839  3.138139   12.9  8.3 20.6 11.05 15.25  0.5010559
## Height 31 76.00000  6.371813   76.0 63.0 87.0 72.00 80.00 -0.3568773
## Volume 31 30.17097 16.437846   24.2 10.2 77.0 19.40 37.30  1.0132739
##          Kurtosis
## Girth  -0.7109412
## Height -0.7233677
## Volume  0.2460393
```

```
mvn(trees, mvnTest="hz")
```

```
## $multivariateNormality
##            Test      HZ    p value MVN
## 1 Henze-Zirkler 0.92118 0.03216314  NO
##
## $univariateNormality
##           Test  Variable Statistic   p value Normality
## 1 Shapiro-Wilk   Girth      0.9412    0.0889    YES
## 2 Shapiro-Wilk  Height      0.9655    0.4034    YES
## 3 Shapiro-Wilk  Volume      0.8876    0.0036    NO
##
```

```
## $Descriptives
##         n     Mean   Std.Dev Median  Min  Max  25th  75th       Skew
## Girth  31 13.24839  3.138139   12.9  8.3 20.6 11.05 15.25  0.5010559
## Height 31 76.00000  6.371813   76.0 63.0 87.0 72.00 80.00 -0.3568773
## Volume 31 30.17097 16.437846   24.2 10.2 77.0 19.40 37.30  1.0132739
##          Kurtosis
## Girth  -0.7109412
## Height -0.7233677
## Volume  0.2460393
```

4. 등분산 검정 for multivariate data

5. 정규성이 만족되지 않을 때 transformation to normality

많은 통계적 기법들은 정규분포를 가정하고 진행된다. 예를 들어, 회귀 분석을 시행할 때, 오차항에 대한 정규성이 보장되어야 신뢰구간, 가설 검정 등 통계적 추론을 진행할 수 있다. 4단원에서 배운 LDA와 QDA 또한 각 클래스에 대한 데이터 분포가 정규분포임을 가정하기 때문에 이 부분에 대한 확인이 필수적이고, 확인 시 정규분포를 따르지 않는다면 데이터 변환을 통해서 정규분포 형태로 만들 필요가 있다. Box-Cox 변환은 대표적인 정규화 변환이다.

단 변수일 때, Box-Cox 변환은 아래와 같이 이루어 진다.

$$y^{(\lambda)} = \begin{cases} \dfrac{y^{\lambda-1}}{\lambda} & \lambda \neq 0 \\ log(y) & \lambda = 0 \end{cases} \longrightarrow .$$

여기서 λ는 데이터로부터 estimate될 또 다른 parameter이다.

$$Define\, K_2 = \left(\prod_{i=1}^{n} Y_i \longrightarrow\right)^{1/n}, K_1 = \frac{1}{\lambda K_2^{\lambda-1}}$$

$$W_i = \begin{cases} K_1(Y_i^{\lambda} - 1) & \lambda \neq 0 \\ K_2 log(Y_i) & \lambda = 0 \end{cases} \longrightarrow .$$

Let $SSE(\lambda)$ be the SSE from the regression of W on X for given λ. grid search로 λ의 estimate을 정하게 된다. 즉, λ을 {-2, -1.75, $\cdots$,1.75, 2} 이런 식으로 정해서 $SSE(\lambda)$을 최소로 하는 λ을 λ에 대한 estimate로 한다. 원리는 이 정도로만 알아두고 실제 R에서 사용 해보자.

```
library(car)


## Loading required package: carData
```

```
powerTransform(cars)


## Estimated transformation parameters
##     speed      dist
## 0.8724383 0.4084646


attach(cars)
trans_cars = cbind(speed^0.87,dist^0.408)
mvn(trans_cars)


## $multivariateNormality
##              Test         Statistic           p value Result
## 1 Mardia Skewness  2.64039723126766  0.61968386668063    YES
## 2 Mardia Kurtosis -0.56984689077437 0.568781548745791    YES
## 3             MVN              <NA>              <NA>    YES
##
## $univariateNormality
##           Test  Variable Statistic   p value Normality
## 1 Shapiro-Wilk  Column1     0.9761    0.4006    YES
## 2 Shapiro-Wilk  Column2     0.9909    0.9645    YES
##
## $Descriptives
##    n      Mean  Std.Dev    Median      Min       Max     25th      75th
## 1 50 10.716069 3.262083 10.548711 3.340352 16.451587 8.687357 12.957333
## 2 50  4.419106 1.196647  4.314907 1.326845  7.051897 3.778410  5.167265
##         Skew   Kurtosis
## 1 -0.2194373 -0.5691548
## 2 -0.2042289 -0.1494201
```

하지만 위의 box cox 변환은 정의상 양수 값일 때만 사용이 가능하다. 음수 또는 0의 값에 대해 box-cox 변환을 하고 싶다면 Yeo-Johnson 변환을 이용하면 된다.

```
library(caret)


## Warning: package 'caret' was built under R version 3.5.2


library(ISLR)
```

```r
data.market = Smarket[,c(-1,-9)]
transparams = preProcess(data.market, method="YeoJohnson")
print(transparams)
```

```
## Created from 1250 samples and 7 variables
##
## Pre-processing:
##   - ignored (0)
##   - Yeo-Johnson transformation (7)
##
## Lambda estimates for Yeo-Johnson transformation:
## 0.96, 0.96, 0.96, 0.96, 0.95, -0.3, 0.96
```

```r
tras.market = predict(transparams, data.market)
summary(tras.market)
```

```
##       Lag1                Lag2                Lag3
##  Min.   :-5.16968   Min.   :-5.16915   Min.   :-5.15909
##  1st Qu.:-0.64689   1st Qu.:-0.64687   1st Qu.:-0.64709
##  Median : 0.03897   Median : 0.03897   Median : 0.03847
##  Mean   :-0.01404   Mean   :-0.01391   Mean   :-0.01548
##  3rd Qu.: 0.59034   3rd Qu.: 0.59035   3rd Qu.: 0.59061
##  Max.   : 5.43747   Max.   : 5.43807   Max.   : 5.44940
##       Lag4                Lag5               Volume            Today
##  Min.   :-5.15907   Min.   :-5.23672   Min.   :0.2909   Min.   :-5.17416
##  1st Qu.:-0.64709   1st Qu.:-0.64935   1st Qu.:0.7214   1st Qu.:-0.64702
##  Median : 0.03847   Median : 0.03846   Median :0.7761   Median : 0.03847
##  Mean   :-0.01556   Mean   :-0.01726   Mean   :0.7833   Mean   :-0.01505
##  3rd Qu.: 0.59061   3rd Qu.: 0.58893   3rd Qu.:0.8413   3rd Qu.: 0.59023
##  Max.   : 5.44943   Max.   : 5.36328   Max.   :1.1560   Max.   : 5.43244
```

```r
mvn(tras.market)
```

```
## $multivariateNormality
##              Test       Statistic               p value Result
## 1 Mardia Skewness  557.5333075808 5.40011955249151e-71     NO
## 2 Mardia Kurtosis 71.796197842233                    0     NO
```

```
## 3             MVN            <NA>                 <NA>     NO
##
## $univariateNormality
##           Test  Variable Statistic   p value Normality
## 1 Shapiro-Wilk   Lag1       0.9736  <0.001      NO
## 2 Shapiro-Wilk   Lag2       0.9736  <0.001      NO
## 3 Shapiro-Wilk   Lag3       0.9737  <0.001      NO
## 4 Shapiro-Wilk   Lag4       0.9737  <0.001      NO
## 5 Shapiro-Wilk   Lag5       0.9722  <0.001      NO
## 6 Shapiro-Wilk  Volume      0.9781  <0.001      NO
## 7 Shapiro-Wilk   Today      0.9736  <0.001      NO
##
## $Descriptives
##           n        Mean   Std.Dev     Median        Min      Max
## Lag1   1250 -0.01403673 1.1351043 0.03896782 -5.1696841 5.437469
## Lag2   1250 -0.01391358 1.1350871 0.03896789 -5.1691485 5.438071
## Lag3   1250 -0.01547913 1.1376463 0.03846993 -5.1590918 5.449404
## Lag4   1250 -0.01555996 1.1377191 0.03846994 -5.1590706 5.449428
## Lag5   1250 -0.01725870 1.1456256 0.03846045 -5.2367213 5.363282
## Volume 1250  0.78326894 0.1078223 0.77610854  0.2909375 1.156042
## Today  1250 -0.01504776 1.1351286 0.03846808 -5.1741605 5.432442
##              25th      75th        Skew Kurtosis
## Lag1   -0.6468896 0.5903414  0.03610739 2.242888
## Lag2   -0.6468740 0.5903548  0.03623308 2.243396
## Lag3   -0.6470902 0.5906063  0.03768070 2.225791
## Lag4   -0.6470896 0.5906068  0.03773408 2.224672
## Lag5   -0.6493501 0.5889268  0.03567170 2.289917
## Volume  0.7214198 0.8413212 -0.01950690 1.435105
## Today  -0.6470203 0.5902297  0.03501035 2.240856
```

하지만 변환을 해도 mvn 검정 결과를 보면 매우 유의하게 정규분포를 따르지 않음을 알 수 있다. 도대체 왜 그럴까? 많은 시간 생각해본 결과 애초에 데이터가 정규분포와 매우 다르게 생겼다면 아무리 변환을 하더라도 정규분포와는 거리가 먼 것 같았다. 즉, 위의 추천된 파라미터를 보면 거의 1에 가까움을 확인할 수 있다. 원래 정규분포를 따르지 않는 데이터에 거의 변화를 주지 않는 것과 다름 없으므로 변환을 시행하고 나서도 정규분포와 거리가 먼 것이다.

Reference

http://blog.naver.com/PostView.nhn?blogId=nife0719&logNo=220993392408 [ROC Curve 그리기]

https://adnoctum.tistory.com/121[Confusion Matrix & ROC Curve 개념]

https://www.isixsigma.com/tools-templates/normality/making-data-normal-using-box-cox-power-transformation/ [Box-Cox 변환에 대한 설명]

https://www.rdocumentation.org/packages/car/versions/2.1-3/topics/powerTransform [powerTransform에 대한 설명]

https://machinelearningmastery.com/pre-process-your-dataset-in-r/ [음수 값 box cox 변환에 대한 소스 코드]

https://cran.r-project.org/web/packages/bestNormalize/bestNormalize.pdf [단변수 정규변환일 때 가장 유용할 듯한 패키지]