Data Mining, Spring 2017: Homework1

Due April 16, 2017
(Please upload your report on e-class by the due date, and also turn in a hard copy in class on April 18.)

Name : 임희락 Department : 소프트웨어학과

Studen ID # : 201220940

1. In this problem, we consider KNN classifier based on Euclidean dis- tance. The following is our training data with one real-valued input (X) and one binary output (Y). For each question below, you may draw a horizontal line to present each data point and find/mark your answer.

	X	Y
1	-0.1	0
2	0.7	1
3	1.0	1
4	1.6	0
5	2	1
6	2.5	1
7	3.2	0
8	3.5	0
9	4.1	1
10	4.9	1

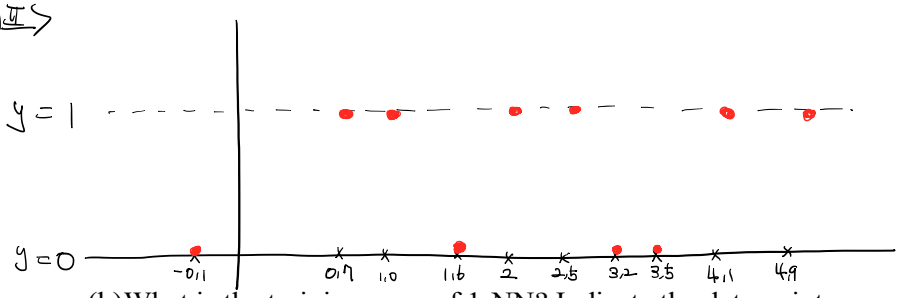
(a) What is the predicted output of a new test data with input value of $x = 1.5$ if you use 1-NN? What is the predicted value if you use 3-NN?

X(1.5)값에 대하여 training data 와의 Euclidean Distance 값 들을 나열 해 보면 다음과 같다.

	X	Y	ED
1	-0.1	0	1.6
2	0.7	1	0.8
3	1.0	1	0.5
4	1.6	0	0.1
5	2	1	0.5
6	2.5	1	1
7	3.2	0	1.7
8	3.5	0	2
9	4.1	1	2.6
10	4.9	1	2.4

따라서 1-NN 의 경우 index 4 번의 값만 고려하여 predicted output 은 0 이 나올 것이고, 3-NN 의 경우 index 3, 4, 5 를 고려하여 output 은 1 이 나올 것이다.

<그래프>



(b) What is the training error of 1-NN? Indicate the data points that are misclassified.

1-NN 의 경우 Euclidean Distance 가 가장 가까운 training data 는 자기 자신일 것이기 때문에 training data 에 대한 accuracy 는 100% 이고 error rate 는 0%이다

(c) Repeat the problem (b) with 3-NN.

각각의 training data 에 대한 predicted ouput 을 표로 나타내면 다음과 같다.

	X	Y	Predicted Output	Y == Predicted Output ?
1	-0.1	0	1	0
2	0.7	1	1	1
3	1.0	1	1	1
4	1.6	0	1	0
5	2	1	1	1
6	2.5	1	1	1
7	3.2	0	0	1
8	3.5	0	0	1
9	4.1	1	1	1
10	4.9	1	1	1

위 표에 따르면 3-NN 일 경우 training data 에 대한 accuracy 는 8/10 = 80% 이고 error rate 는 20%이다.

(d) What is the leave-one-out cross-validation error of 1-NN? Indicate the data points that are misclassified.

각각 한 개의 sample 을 training data 에서 제외하여 test data 로 사용하였을 경우 1-NN 에서는 다음과 같은 Accuracy 를 나타낸다.

i	*Accuracy(i)*
1	0
2	1
3	1
4	0
5	0
6	1
7	1
8	1
9	0
10	1

따라서, Final Accuracy 는 6/10 이므로 0.6 이고 Error rate 은 0.4 이다.

(e) Repeat the problem (d) with 3-NN.

각각 한 개의 sample 을 training data 에서 제외하여 test data 로 사용하였을 경우 3-NN 에서는 다음과 같은 Accuracy 를 나타낸다.

i	*Accuracy(i)*
1	0
2	0
3	1
4	0
5	1
6	0
7	0
8	0
9	0
10	0

따라서, Final Accuracy 는 2/10 이므로 0.2 이다.

2. In this problem, you will train the Naïve Bayes classifier using the following training samples. Color, Type, and Origin are all binary features, and Stolen? is either Yes or No.

	Color (X_1)	Type (X_2)	Origin (X_3)	Stolen? (Y)
1	Red	Sports	Domestic	Yes
2	Red	Sports	Domestic	No
3	Red	Sports	Domestic	Yes
4	Yellow	Sports	Domestic	No
5	Yellow	Sports	Imported	No
6	Yellow	SUV	Imported	No
7	Yellow	SUV	Imported	No
8	Yellow	SUV	Domestic	No
9	Red	SUV	Imported	No
10	Red	Sports	Imported	Yes

(a) Compute the prior probability for each class (Yes and No) in Y .

Class Y 에 대하여 각각의 사전 확률은 다음과 같다. P(Yes in Y) = 0.3 , P(No in Y)= 0.7

(b) Compute all the class-conditional probabilities(that is, $P(X_i = k|Y = j)$ for each i, each value k and j).

P(X1=Red | Y=Yes)= 0.3/0.3 = 1
P(X1=Yellow | Y=Yes)= 0= 0

P(X1=Red | Y=No) = 0.3/0.7 = 5/7
P(X1=Yellow | Y=No) = 2/7

P(X2=Sports | Y=Yes)= 0.3/0.3
P(X2=SUV | Y=Yes)= 0

P(X2=Sports | Y=No)= 0.3/0.7= 3/7
P(X2=SUV | Y=No)= 4/7

P(X3=Imported | Y=Yes)= 0.1/0.3 = 1/3
P(X3=Domestic | Y=Yes)= 2/3

P(X3=Imported | Y=No)= 0.4/0.7= 4/7
P(X3=Domestic | Y=No)= 3/7

(c) Now, suppose you have a Red Domestic SUV. Classify your car using the Naïve Bayes algorithm. You would need to compute the posterior probabilities using the probability values obtained above.

Naïve Bayes algorithm 에서는 각 사건을 독립으로 가정한다. 따라서 Posterior Probability 를 계산 해 보면 다음과 같다.

P(X1=Red, X2=SUV, X3=Domestic | Y=Yes)
= P(X1=Read | Y=Yes) * P(X2=SUV | Y=Yes) * P(X3=Domestic | Y=Yes) * P(Y = Yes)
= 1 * 0 * 2/3 * 3/10 = 0

P(X1=Red, X2=SUV, X3=Domestic | Y=No)

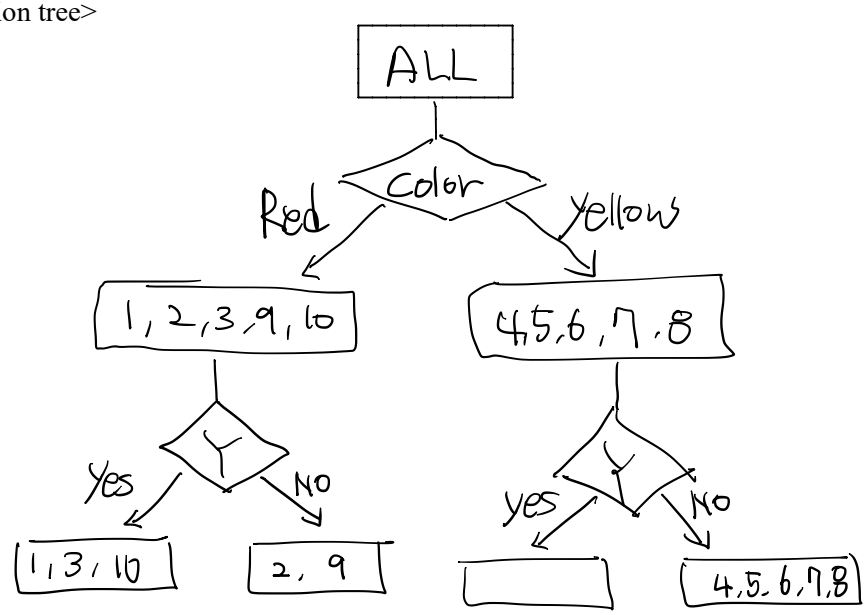
= P(X1=Read | Y=No) * P(X2=SUV | Y=No) * P(X3=Domestic | Y=No) * P(Y = No)

= 2/7 * 4/7 * 3/7 * 7/10 = 12/245

3. Consider the same dataset shown in Problem 2. We want to build a decision tree for predicting Y given the features.

   (a) Suppose you selected the feature *Color* as a root node. Let the training samples in Red color be assigned to the left subtree and those in *Yellow* to the right subtree. Compute the entropy of each subtree. (Hint: the samples on the left subtree are 1,2,3,9,and 10, which have three samples with label *Yes* and two samples with label *No*. So $P(Y = Yes) = 3/5$, $P(Y = No) = 2/5$ in case of the left subtree. The entropy is defined as $E = -\sum_i P(Y = i) \log P(Y = i)$.)

<decision tree>



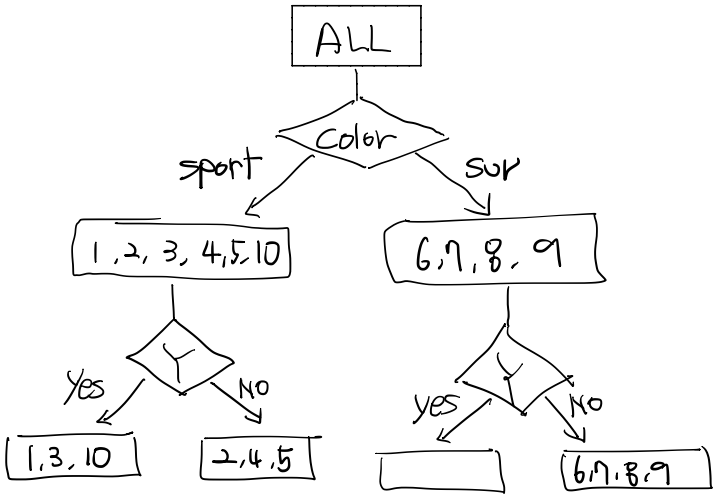
Left subtree(Red): -(3/5)*log(3/5) − (2/5) ∗ log(2/5) ≒ 0.971 (로그 의 밑은 2)

Right subtree(Yellow): 0*log 0 − 1 ∗ log 1 = 0 (로그의 밑은 2)

Weighted Average Entropy of children: (5/10) * 0.971 + (5/10) * 0 = 0.4855

(b) Repeat (a) with each feature *Type* and *Origin*, respectively. Discuss which among the three features is more suitable as a root node of Decision tree classifier.

<decision tree>

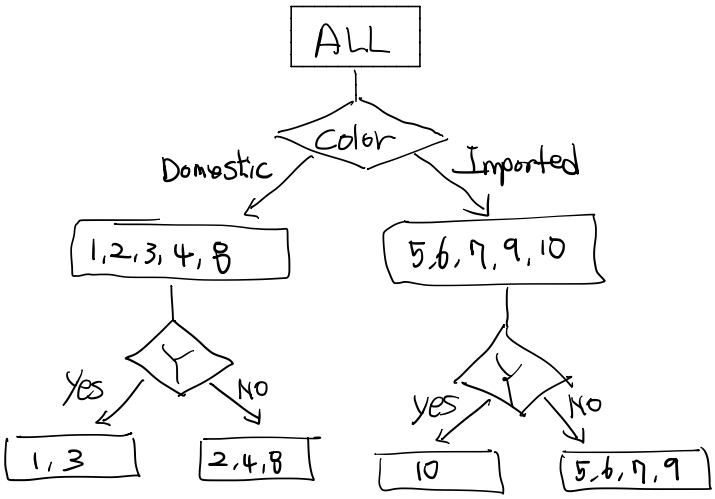


Left subtree(Sports): -(3/6) * log(3/6) − (3/6) ∗ log(3/6) = 1 (로그 의 밑은 2)

Right subtree(SUV): 0 * log 0 − 1 ∗ log 1 = 0 (로그의 밑은 2)

Weighted Average Entropy of children: (6/10) * 1 + (4/10) * 0 = 0.6

<decision tree>



Left subtree(Domestic): -(2/5) * log(2/5) − (3/5) ∗ log(3/5) ≒ 0.971…. (로그 의 밑은 2)

Right subtree(Imported): -(1/5) * log(1/5) − (4/5) ∗ log(4/5) ≒ 0.722…. (로그의 밑은 2)

Weighted Average Entropy of children: (5/10) * 0.971 + (5/10) * 0.722 = 0.8465

Color, Type, Origin 세 가지 feature 중에서 가장 information gain 이 큰 feature 를 root node 로 정하는 것이 합리적이다. Information gain 은 Parent Entropy 에서 Child Entropy 를 뺀 값이고, 세 경우 모두 Parent Entropy 는 같으므로 Child Entropy 가 가장 큰 Color feature 를 root node 로 하는 것이 가장 적절하다.

4. **R Programming** You are given a Pima Indians Diabetes dataset which is provided in R library MASS. Given several clinical informations for each patient, build a supervised classification model to predict whether a patient is diabetic or not. You will use Pima.tr and Pima.te which is a training and test set for training the model. For details, please refer to the document for the dataset (https://stat.ethz.ch/ R-manual/R-devel/library/MASS/html/Pima.tr.html). **For each problem, copy-and-paste your R codes in your report.**

   Input variables(*X*) Patient information(number of pregnancies, body mass index, age, etc.).

   Output variable(*y*) Is the patient diabetic? (yes/no)

   (a) Data preparation

   Read the training and test data using the R code below.

```
library(MASS)

#trainingdataset
trainingdata <- Pima.tr

#testdataset
testdata <- Pima.te
```

   i. How many samples and features are in each dataset (training set, test set)?

Training data set 과 test data set 은 각각 200 개, 332 개의 sample 을 갖고 있으며, 두 data set 의 feature 는 다음과 같이 동일한 feature 를 갖는다.

npreg: number of pregnancies.
glu: plasma glucose concentration in an oral glucose tolerance test.
bp: diastolic blood pressure (mm Hg).
skin: triceps skin fold thickness (mm).
bmi: body mass index (weight in kg/(height in m)₩^2).

ped: diabetes pedigree function.

age: age in years.

type: Yes or No, for diabetic according to WHO criteria

여기서 type 은 categorical data 로서 당뇨병의 유무를 나타내며 본 data set 의 레퍼런스(https://stat.ethz.ch/ R-manual/R-devel/library/MASS/html/Pima.tr.html)를 참고 해 보았을 때, Type 은 output 이라고 생각할 수 있으므로 feature 는 경우에 따라서 7 개로 볼 수도 있다.

Code & Result

```
# 2017-1 Ajou univ. Data mining Assignment #1-4
# Heerak Lim, lrocky1229@gmail.com

library (MASS)

trainingData = Pima.tr
testData = Pima.te

# number of samples and features in dataset
nrow(trainingData)

## [1] 200

nrow(testData)

## [1] 332

length(trainingData)

## [1] 8

length(testData)

## [1] 8
```

ii.Show the summary statistics of each feature of the data.

Code & Result

```
# summary of statistics of each feature of the dataset

# trainingData
summary(trainingData$npreg)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.00    1.00    2.00    3.57    6.00   14.00

summary(trainingData$glu)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    56.0   100.0   120.5   124.0   144.0   199.0
```

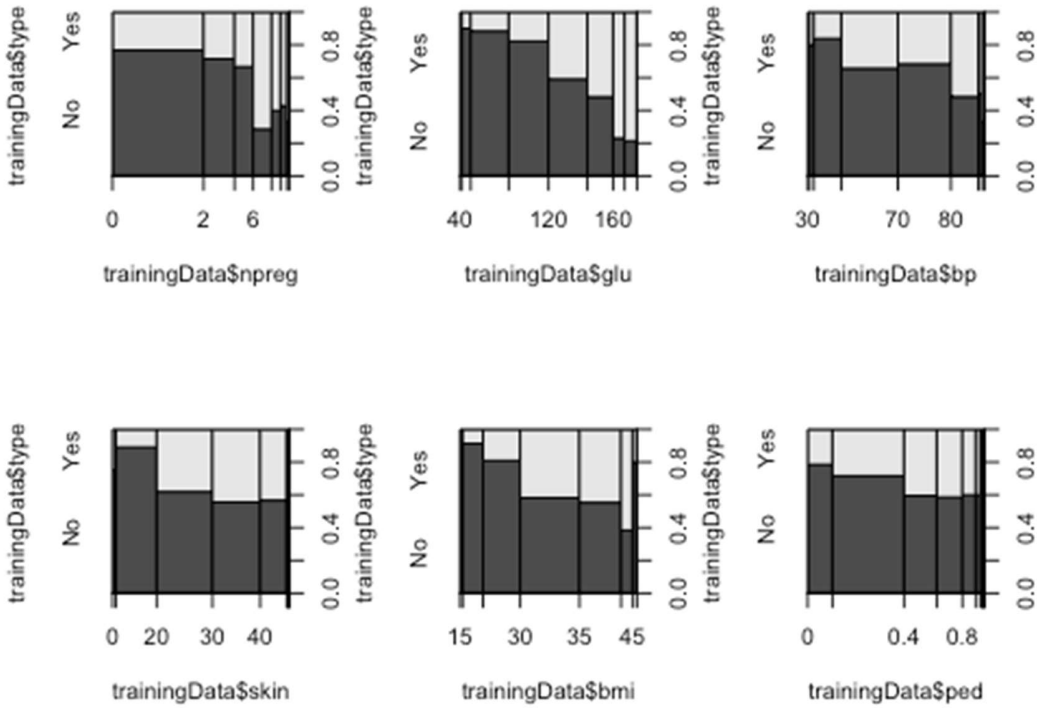
```r
summary(trainingData$bp)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   38.00   64.00   70.00   71.26   78.00  110.00

summary(trainingData$skin)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    7.00   20.75   29.00   29.22   36.00   99.00

summary(trainingData$bmi)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   18.20   27.58   32.80   32.31   36.50   47.90

summary(trainingData$ped)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0850  0.2535  0.3725  0.4608  0.6160  2.2880

summary(trainingData$age)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   21.00   23.00   28.00   32.11   39.25   63.00

summary(trainingData$type)

##  No Yes
## 132  68

# testData
summary(testData$npreg)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   1.000   2.000   3.485   5.000  17.000

summary(testData$glu)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    65.0    96.0   112.0   119.3   136.2   197.0

summary(testData$bp)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   24.00   64.00   72.00   71.65   80.00  110.00

summary(testData$skin)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    7.00   22.00   29.00   29.16   36.00   63.00

summary(testData$bmi)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   19.40   28.17   32.90   33.24   37.20   67.10

summary(testData$ped)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0850  0.2660  0.4400  0.5284  0.6792  2.4200

summary(testData$age)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   21.00   23.00   27.00   31.32   37.00   81.00

summary(testData$type)

##  No Yes
## 223 109
```

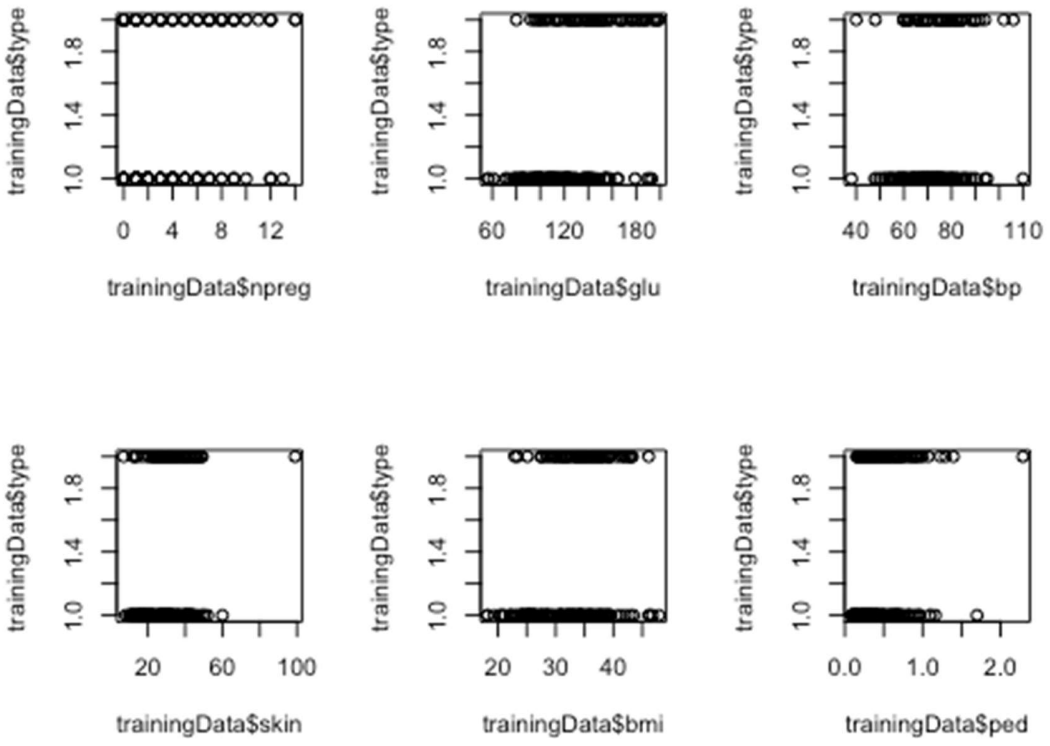
iii. Show at least one plot that shows the output variable (y) is correlated with some of the input variables X.

Code & Result

```
# Show at least one plot that shows the output variable (y)
# is correlated with some of the input variables X.

par(mfrow=c(2,3))
plot(trainingData$type~trainingData$npreg)
plot(trainingData$type~trainingData$glu)
plot(trainingData$type~trainingData$bp)
plot(trainingData$type~trainingData$skin)
plot(trainingData$type~trainingData$bmi)
plot(trainingData$type~trainingData$ped)
```



```r
# or another plot
plot(trainingData$npreg,trainingData$type)
plot(trainingData$glu,trainingData$type)
plot(trainingData$bp,trainingData$type)
plot(trainingData$skin,trainingData$type)
plot(trainingData$bmi,trainingData$type)
plot(trainingData$ped,trainingData$type)
```

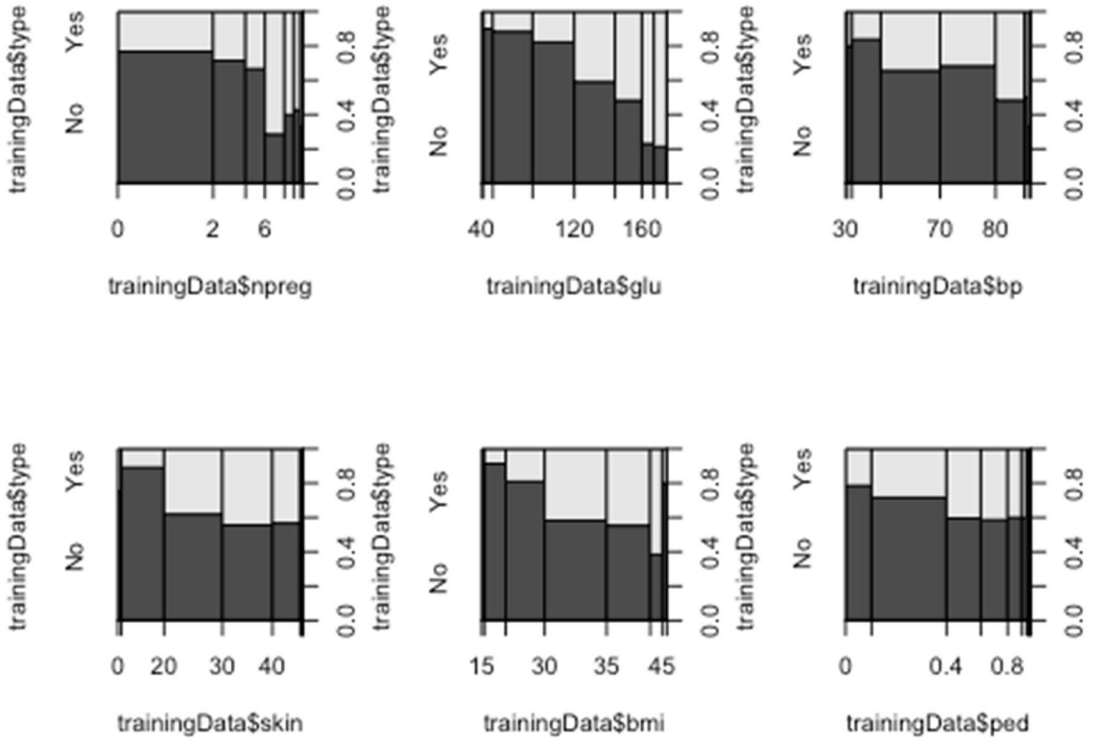


(b) **Exploratory Data Analysis** Before training the model with the dataset, you would explore the data first.

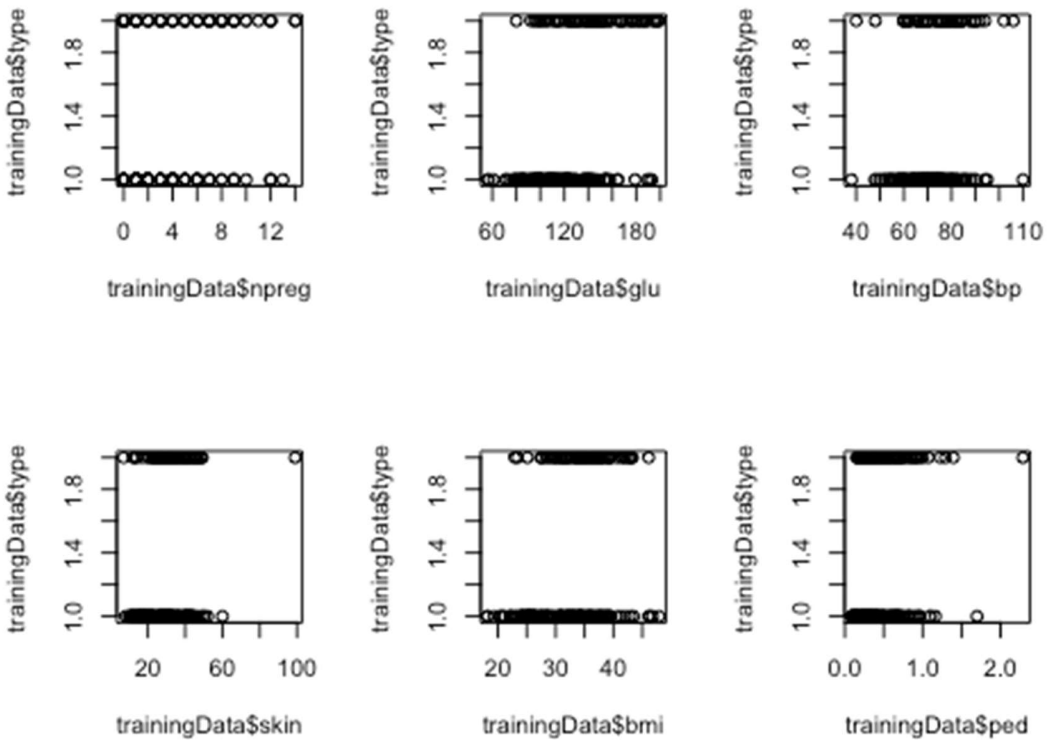
i. First, choose more than two features among them. At least one categorical variable and one continuous variable shoul be selected. Now, visualize the distribution of the selected features using histogram, boxplot, etc. (Please refer to `hist(.)` and `boxplot(.)` function) Briefly explain your findings.

Code & Result

```
# Show at least one plot that shows the output variable (y)
# is correlated with some of the input variables X.

par(mfrow=c(2,3))
plot(trainingData$type~trainingData$npreg)
plot(trainingData$type~trainingData$glu)
plot(trainingData$type~trainingData$bp)
plot(trainingData$type~trainingData$skin)
plot(trainingData$type~trainingData$bmi)
plot(trainingData$type~trainingData$ped)
```

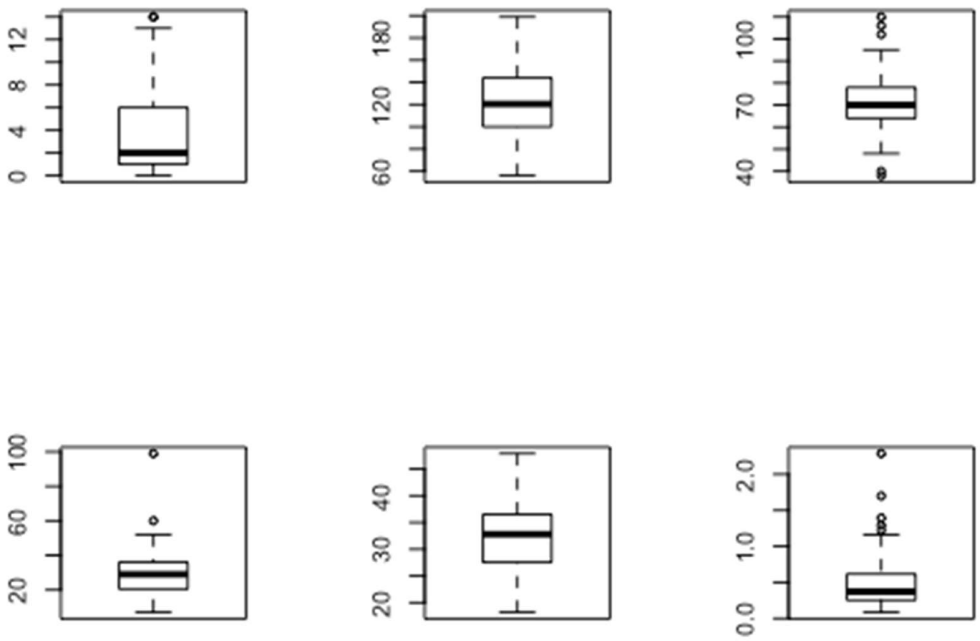



```
# or this
plot(trainingData$npreg,trainingData$type)
plot(trainingData$glu,trainingData$type)
plot(trainingData$bp,trainingData$type)
plot(trainingData$skin,trainingData$type)
plot(trainingData$bmi,trainingData$type)
plot(trainingData$ped,trainingData$type)
```

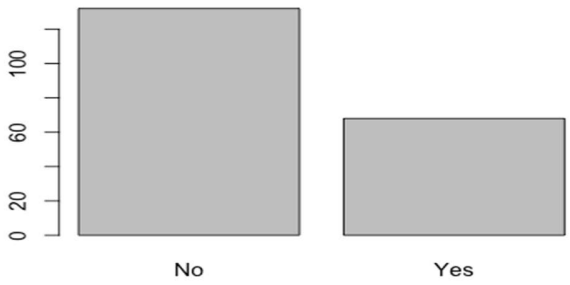



```r
# histogram of 6continuous varicables
hist(trainingData$npreg)
hist(trainingData$glu)
hist(trainingData$bp)
hist(trainingData$skin)
hist(trainingData$bmi)
hist(trainingData$ped)
```


```
v <- readline("please enter any key to next: ")

## please enter any key to next:

# boxblot of 6continuous varicables
boxplot(trainingData$npreg)
boxplot(trainingData$glu)
boxplot(trainingData$bp)
boxplot(trainingData$skin)
boxplot(trainingData$bmi)
boxplot(trainingData$ped)
```




```
# plot of categorical variable(type)

par(mfrow=c(1,1))
plot(trainingData$type)
```



Code & Result 에서 나타나는 첫 번째 그래프를 보았을 때, glu feature 의 정도가 증가함에

따라 type 은 yes 가 점점 더 많아 짐을 알 수 있다. 세 번째 그래프(히스토그램)를 보면, glu, bmi, bp, skin 데이터는 나타내는 빈도의 수가 어느정도 정규 분포와 비슷한 모습을 나타냄을 알 수 있고, 나머지 데이터는 점점 빈도수가 급격하게 감소하는 모습을 나타낸다.

본 데이터에 대한 자세한 분석은 ii 에서 다루도록 한다.

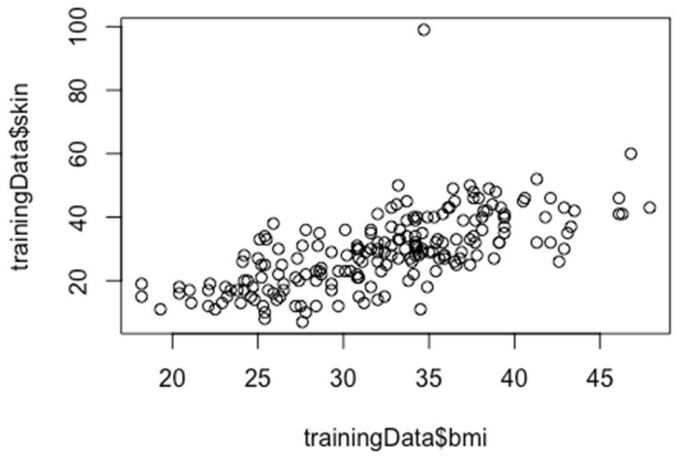
ii. Now, plot and analyze one interesting relationship between the selected two features. (e.g. correlation between age and triceps skin fold thickness.)

여러 data relationship 에 대하여 단순한 plot 을 만들어 봤을 때, skin feature 와 bmi 와의 관계가 의미 있다고 생각하여 두 feature 에 대하여 분석하였다. 두 feature 간의 관계를 나타내는 plot 을 보았을 때, 어느정도 선형적인 관계가 나타났고, bmi(체질량 지수)가 높아질수록 skin(삼두근 피부 두께)의 정도가 어느정도 증가하는 모습을 보여주었다. 단순 선형 회귀 모델(Y=bX+a, Y:skin, X:bmi)을 통해 분석 해 보았을 때, 귀무가설 H0(b 는 0 이다)에 대하여 상당히 낮은 p-value 를 갖는다 .따라서 선형 회귀 모델은 데이터를 상당히 잘 표현한다고 생각 할 수 있다. b 의 값이 약 1.2 이므로 bmi 가 1 정도 증가한다고 생각 했을 때, 삼두근의 피부 두께는 1.2(mm)정도 증가한다고 생각할 수 있다.

아래 실행 결과에서 첫 번째 plot 의 결과를 보면, bmi 가 약 35, skin 이 약 100 의 값을 나타내는 outlier 가 나타난다. 다른 데이터들과 상당히 동떨어진 값을 나타내므로 이 데이터를 없애는 것 이 바람직할 것이다. 기존의 데이터에서 outlier 에 해당하는 157 번 째 인덱스를 갖는 데이터를 지우고 같은 방법으로 선형 회귀 분석을 해 보았을 때, 두 변수간의 상관관계의 정도를 나타내는 R-squared 와 Adjusted R-squared 값이 각각 0.4343 에서 0.5107 로, 0.4315 에서 0.5082 로 증가했음을 볼 수 있다.

Code Result

```
# analyze one interesting relationship between the selected two features
# I choose the skin(triceps skin fold thickness) and bmi(body mass index)

#simple plot
plot(trainingData$skin~trainingData$bmi)
```



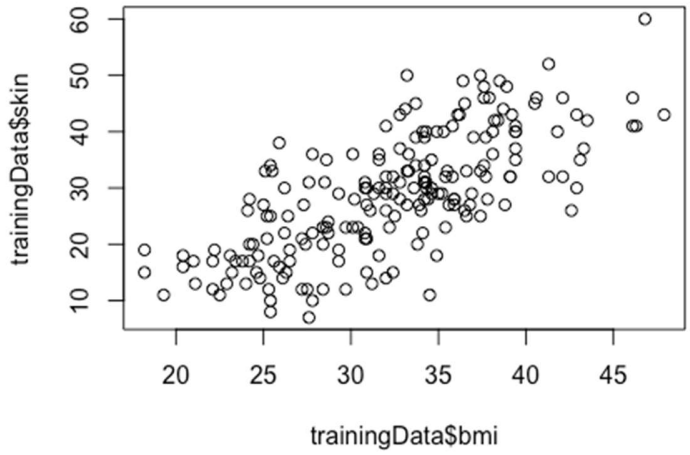

```r
#linear regression
fig = lm(trainingData$skin~trainingData$bmi)
summary(fig)
```

```
##
## Call:
## lm(formula = trainingData$skin ~ trainingData$bmi)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -20.975  -5.633  -0.913   4.558  66.772
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)      -11.5107     3.3616  -3.424  0.00075 ***
## trainingData$bmi   1.2605     0.1022  12.330  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.84 on 198 degrees of freedom
## Multiple R-squared:  0.4343, Adjusted R-squared:  0.4315
## F-statistic:   152 on 1 and 198 DF,  p-value: < 2.2e-16
```

```r
#remove outlier(index 157 data)
trainingData <- trainingData[-c(157),]
nrow(trainingData)
```

```
## [1] 199
```

```r
#plot and linear regression on the new dataset
plot(trainingData$skin~trainingData$bmi)
```




```
fig = lm(trainingData$skin~trainingData$bmi)
summary(fig)

##
## Call:
## lm(formula = trainingData$skin ~ trainingData$bmi)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -20.5926  -5.1018  -0.6373   4.8062  20.0181
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)      -11.1530     2.8404  -3.927 0.000119 ***
## trainingData$bmi   1.2390     0.0864  14.340  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.469 on 197 degrees of freedom
## Multiple R-squared:  0.5107, Adjusted R-squared:  0.5082
## F-statistic: 205.6 on 1 and 197 DF,  p-value: < 2.2e-16
```

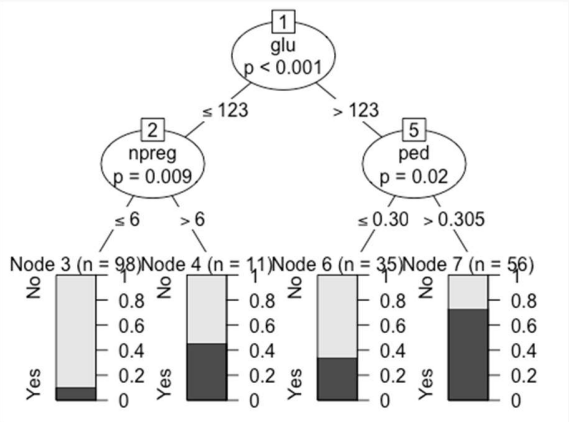
(c) **Classiftcation model** We have whole labeled dataset and training and test set are already separated following the R code above. Now, we want to train the classification models with the train- ing set using **Decision tree, Naive Bayes classiftcation, and K-NN classiftcation**. Then, the model is evaluated with the test set. Additionally, the optimal parameter selection will be performed using 5-fold cross validation in the training dataset.

Decision tree You will build a decision tree with the training set and predict the class label with the test set. (Please refer to `ctree(.)` function)

i. Plot the decision tree built with the training set.

Code & Result

```
# restore the original data
trainingData = Pima.tr

# make Decision Tree
training_data <- ctree(type ~ .,data=trainingData)
plot(training_data)
```



ii. Calculate the training error and test error in this classification model.

Code & Result

```
# Calculate training error by DT
pre_training = predict(training_data, trainingData)
trainingError = mean(pre_training!=trainingData$type)
print(trainingError)

## [1] 0.21

# Calculate test error by DT
pre_test = predict(training_data, testData)
testError = mean(pre_test!=testData$type)
print(testError)

## [1] 0.2710843
```

Naive Bayes classiftcation Train the Naive Bayes classifica- tion model with the training set and predict the class label with the test set. (Please refer to `naiveBayes(.)` function in `e1071` package)

i. Calculate the training error and test error. Show the confusion matrix for calculation. (Please refer to `table(.)` and `predict(.)` function)

Code & Result

```
# Naive Bayes Classification
training_data <- naiveBayes(type~.,data=trainingData)

# Calculate training error by NBC
pre_training = predict(training_data, trainingData)
trainingError = mean(pre_training!=trainingData$type)
print(trainingError)

## [1] 0.22

#confusion matrix of training data
confusionMatrix <- table(pre_training, trainingData$type)
print(confusionMatrix)

##
## pre_training  No Yes
##          No  110  22
##          Yes  22  46

# Calculate test error by NBC
pre_test = predict(training_data, testData)
testError = mean(pre_test!=testData$type)
print(testError)

## [1] 0.2439759



#confusion matrix of training data
confusionMatrix <- table(pre_test, testData$type)
print(confusionMatrix)

##
## pre_test  No Yes
##      No  185  43
##      Yes  38  66
```

K-NN classiftcation In K-NN classification, you will perform 5-fold cross validation to determine the best parameter *K* on your training set. You would calculate cross-validation error at each fold with varying *K*. (Please refer to knn(.) function in class package)

i. First, you will separate the training dataset into 5 folds to perform 5-fold cross validation.

Code & Result

```
# for cross validation


trainingAsNumeric <- trainingData
trainingAsNumeric$npreg <- as.numeric(as.character(trainingData$npreg))
trainingAsNumeric$glu <- as.numeric(as.character(trainingData$glu))
trainingAsNumeric$bp <- as.numeric(as.character(trainingData$bp))
trainingAsNumeric$skin <- as.numeric(as.character(trainingData$skin))
trainingAsNumeric$bmi <- as.numeric(as.character(trainingData$bmi))
trainingAsNumeric$ped <- as.numeric(as.character(trainingData$ped))
trainingAsNumeric$age <- as.numeric(as.character(trainingData$age))
trainingAsNumeric$type <- as.numeric(as.factor(trainingData$type))

normalize<-function(x){
  return((x-min(x))/(max(x)-min(x)))
}

nomalizedTrainingData <- as.data.frame(lapply(trainingData[1:7],normalize))
nomalizedTestData <- as.data.frame(lapply(testData[1:7],normalize))

set.seed(1)
# 5-fold cross validation
idx <- createFolds(trainingData$type, k=5)
print(sapply(idx, length))

## Fold1 Fold2 Fold3 Fold4 Fold5
##    39    41    40    39    41
```

ii. Calculate the average cross-validation error across the folds for each value of *K*. To find the optimal *K*, plot the result (x-axis: parameter *K*, y-axis: average cross-validation error). Explain how you determine the best parameter *K* in your experiments.

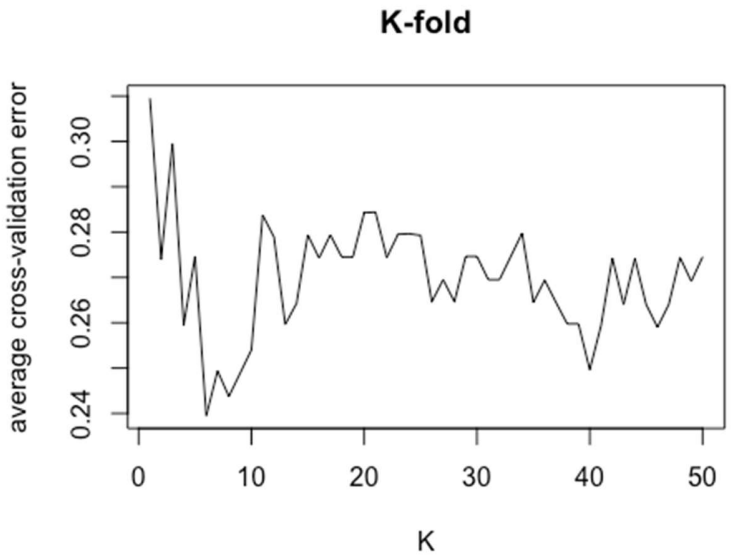
다음과 같은 Code 를 통해 optimal 한 K 값을 구할 수 있다. 아래 그래프에서 average cross-validation error 가 가장 작은 point 의 K 값이 optimal k 이며 which.min() 함수를 통해 간단하게 구할 수 있다.

Code & Result

```
set.seed(1)
ks <- 1:50
res <- sapply(ks, function(k) {
  res.k <- sapply(seq_along(idx), function(i) {
    pred <- knn(train = nomalizedTrainingData[ -idx[[i]] ,] , test = n
omalizedTrainingData[ idx[[i]], ], cl=trainingData$type[-idx[[i]] ], k
=k)
    mean(trainingData$type[idx[[i]]]!=pred)
  })
  mean(res.k)
})

print(res)
```

```
##  [1] 0.3095341 0.2740150 0.2995278 0.2595091 0.2746435 0.2394966 0.
2493871
##  [8] 0.2437523 0.2488806 0.2540088 0.2837774 0.2788993 0.2596373 0.
2642652
## [15] 0.2793934 0.2742652 0.2793934 0.2745153 0.2745153 0.2842714 0.
2843934
## [22] 0.2743871 0.2795153 0.2796373 0.2792652 0.2646310 0.2695091 0.
2646310
## [29] 0.2746373 0.2746373 0.2695091 0.2695091 0.2746373 0.2797655 0.
2645091
## [36] 0.2693871 0.2645091 0.2597592 0.2597592 0.2496310 0.2593871 0.
2742714
## [43] 0.2640150 0.2741432 0.2640150 0.2590150 0.2640150 0.2743996 0.
2691432
## [50] 0.2745216
```

```
plot(ks, res, type="l",ylab="average cross-validation error", xlab="K
", main="K-fold")
```




```
# What is the Best Parameter K in experiments?
bestK<-which.min(res)
print(bestK)

## [1] 6
```

iii. What is the final cross-validation error, and the final classification error on the test data using the best model you selected?

```
bestPredict <- knn(train=nomalizedTrainingData, test=nomalizedTestData, cl=trainingDat
a$type, k=bestK)
#final cross-validation error
min(res)

## [1] 0.2394966

#final classification error
mean(testData$type != bestPredict)

## [1] 0.2349398
```

Model Comparison Compare and analyze three different classification models with respect to the classification performance on this dataset. Briefly discuss the result.

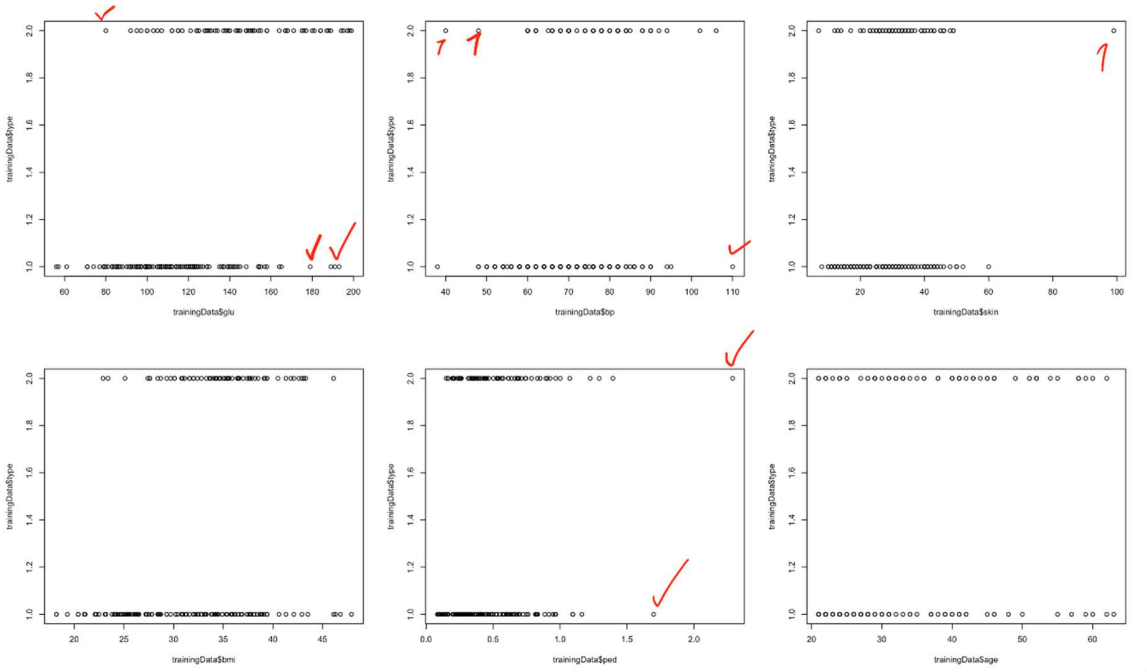
서로 다른 세 가지 classification model 을 error rate 관점에서 비교한다면 다음과 같다.

Classification model	Training data error	Test data error
Decision Tree	0.21	0.2710843
Naïve Bayse	0.22	0.2439759
5-fold cross validation	0.2394966	0.2349398

Model improvement Think of a way to improve the test error from any of the three algorithms above. Briefly explain your idea, implement/re-train the algorithm based on the idea, and report the improved test error. Briefly discuss your findings.

본 과제를 수행하면서, 총 3 가지 classification 방법을 시도 해 보았다. 하지만 세가지 모두 전처리과정을 수행하지 않았기 때문에, 적은 수의 outlier 가 모델 전체에 나쁜 영향을 준다 따라서 training data 와 test data 를 Pima.tr Pima.te 를 그대로 쓰지 않고 outlier 를 제거하는 전 처리 과정을 거쳐 새로운 data set 을 사용하였다.

각각의 데이터의 simple plot 을 참고하여 다음과 같이 outlier 를 선정하였다.



8, 11, 48, 104, 111, 132, 135, 157, 193 index 를 갖는 data 들을 다음과 같이 제거한 뒤 본 보고서에서 접근한 방법과 동일한 방법으로 세 가지 Classification model 의 accuracy 를 비교해 보았다.

Code

```
# removing outlier
trainingData <- trainingData[-c(8, 11, 48, 104, 111, 132, 135, 157, 193),]
```

<raw data set>

Classification model	Training data error	Test data error
Decision Tree	0.21	0.2710843
Naïve Bayse	0.22	0.2439759
5-fold cross validation	0.2394966	0.2349398

<after preconditioning>

Classification model	Training data error	Test data error
Decision Tree	0.1727749	0.2590361
Naïve Bayse	0.2041885	0.2379518
5-fold cross validation	0.2304993	0.2771084

두 표를 보면 Test data 의 5-fold cross validation 방법을 제외하고 모두 error rate 가 낮아졌음을 확인할 수 있다. 모든 곳에서 error rate 가 낮아지지 않은 이유는 outlier 를 선정하는 과정에서 매뉴얼 하게 선정했기 때문이라고 생각한다.

[첨부 1]

전체 R 코드 및 결과를 R studio 의 R markdown tool 을 사용하여 Auto Generation 된 문서로서 제출합니다.