2a. $P(y=1|X) = \frac{1}{1+e^{-\beta' x}} > P(y=0|X) = 1 - \frac{1}{1+e^{-\beta' x}}$

$$\ln(1) - \ln(1+e^{-\beta' x}) > \ln\left(1 - \frac{1}{1+e^{-\beta' x}}\right)$$

$$-\ln(1+e^{-\beta' x}) > \ln\left(1 - \frac{1}{1+e^{-\beta' x}}\right)$$

$$-\ln(1+e^{-\beta' x}) - \ln\left(1 - \frac{1}{1+e^{-\beta' x}}\right)$$

$$\ln\left(\frac{\frac{1}{1+e^{-\beta_1 x}}}{1 - \frac{1}{1+e^{-\beta_1 x}}}\right) > 0$$

$$\ln\left(\frac{1}{\frac{1}{e^{-\beta' x}} - \cancel{1}}\right) > 0$$

$$\ln\left(\frac{1}{e^{-\beta' x}}\right) > 0$$

$$\ln\left(\frac{1}{e^{-\beta' x}}\right) > 0$$

$$\ln\left(e^{\beta' x}\right) > 0$$

$$\beta' x > 0$$

$$\boxed{X \text{ s.t. } 1 + 2x_1 + 2x_2 > 0}$$

2b) $\frac{1}{1+e^{-\beta' x}} > 0.8$

$1 > 0.8 + 0.8e^{-\beta' x}$

$0.2 > 0.8e^{-\beta' x}$

$0.25 > e^{-\beta' x}$

$\ln(0.25) > -\beta_1' x$

$$\boxed{-1-2x_1-2x_2 < \ln(0.25)}$$

2c) $-1-2x_1-2x_2 < \ln\left(\frac{1}{4}\right) = -\ln(4)$

$-1-2x_1-1 < -\ln(4)$

$-2x_1-2 < -\ln(4)$

$-2x_1 < 2-\ln(4)$

so, x_1 s.t. $\boxed{x_1 > \frac{\ln 4 - 2}{2}}$

4a) $\frac{\partial z_i}{\partial \beta_0} = 1$, $\frac{\partial z_i}{\partial \beta_1} = x_{1i}$, $\frac{\partial z_i}{\partial \beta_2} = x_{2i}$

4b) $\frac{\partial J(\beta)}{\partial \beta_0} = \sum \frac{\partial}{\partial \beta_0} \ln(1+e^{-\beta_0+\beta_1 x_{1i}+\beta_2 x_{2i}}) - \frac{\partial}{\partial \beta_0} y_i \cdot (\beta_0+\beta_1 x_{1i}+\beta_2 x_{2i})$

$$= \boxed{\sum_{i=1}^{N} \left(\frac{1}{1+e^{\beta_0+\beta_1 x_{1i}+\beta_2 x_{2i}}}\right) - y_i}$$

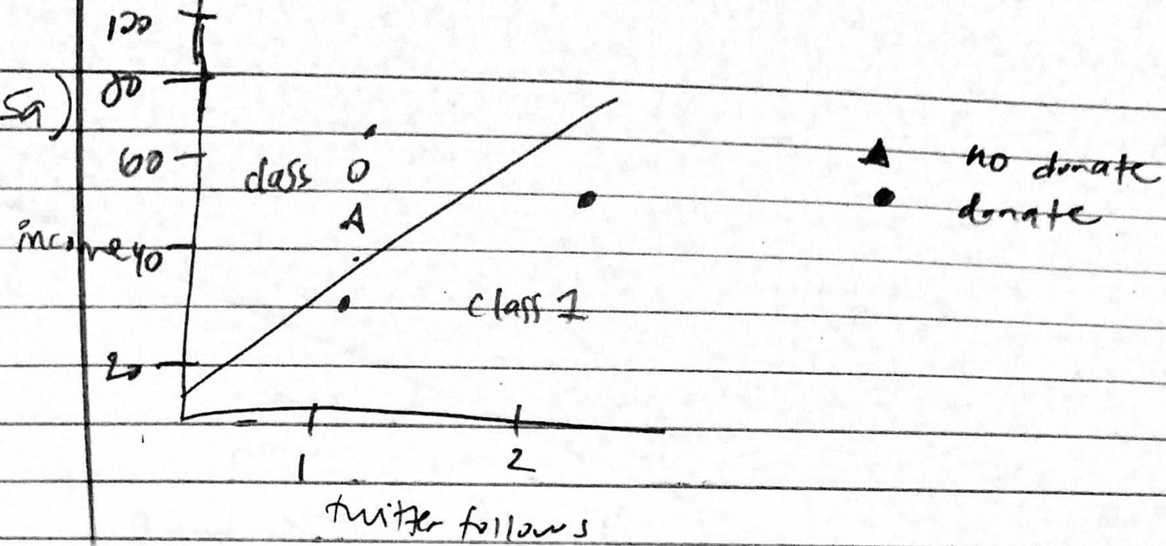
$\frac{\partial J(\beta)}{\partial \beta_1} = \sum \frac{\partial}{\partial \beta_1} \ln\left(1+e^{\beta_0+\beta_1 x_{1i}+\beta_2 x_{2i}}\right) - \frac{\partial}{\partial \beta_1} y_i (\beta_0+\beta_1 x_{1i}+\beta_2 x_{2i})$

$$= \boxed{\sum \left(\frac{x_{1i}}{1+e^{\beta_0+\beta_1 x_{1i}+\beta_2 x_{2i}}}\right) - y_i x_{1i}}$$

$$\frac{\partial J(\beta)}{\partial \beta_2} = \boxed{\sum \left(\frac{x_{2i}}{1+e^{\beta_0+\beta_1 x_{1i}+\beta_2 x_{2i}}}\right) - y_i x_{2i}}$$

no closed form solutions exist for the parameters.

5a)



5b) $x_{i2} = 60x_{i1}$

$0 = 60x_{i1} - x_{i2}$

$(1, 70)$

$$\boxed{w = \begin{bmatrix} 60 \\ -1 \end{bmatrix}, \quad b = [0 \quad 0]}$$

$z_i = 60x_{i2} - x_{i2}$

if $z_i < 0$, class 0
if $z_i > 0$, class 1

c) $z_1 = -30 \rightarrow \frac{1}{1+e^{30}}$

$z_2 = 60-50 = 10 \rightarrow \frac{1}{1+e^{10}}$

$z_3 = 60-70 = -10 \rightarrow \frac{1}{1+e^{-10}}$

$z_4 = 120-80 = 40 \rightarrow \frac{1}{1+e^{40}}$

$z_5 = 60-100 = -40 \rightarrow \frac{1}{1+e^{-40}}$

z_5 has smallest likelihood of being in class 1 (largest denominator). $(1, 100) \leftarrow$ (twitter, income)

5d) in part c, $P(y_i | X_i)$ increases for $y_i = 1$ and decreases for $y_i = 0$

$z_i = w' X_i + b$ since $d > 0$

$\hat{y}$ can change based on d value. class 0 boundary is narrower when z_i increases. points could change based on shift size.

Homework 3

Kitu Komya

May 22, 2018

Question 1

part a

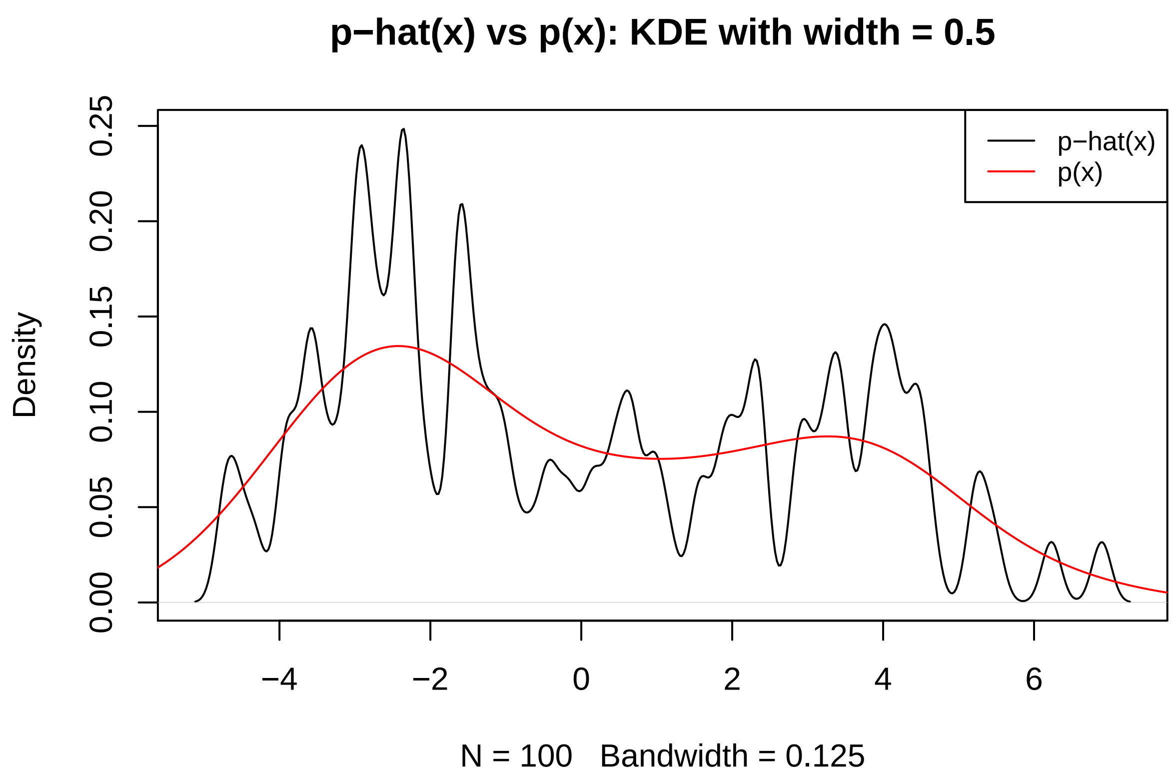
```r
set.seed(2018)

# sample from two mixed models
x <- rnorm(n = 60, mean = -2, sd = 1.5)
x <- append(x, rnorm(n = 40, mean = 3, sd = 1.5))
```

part b

```r
# estimating p-hat(x)
density(x, kernel = "gaussian", width = 0.5)
```

```
##
## Call:
##  density.default(x = x, kernel = "gaussian", width = 0.5)
##
## Data: x (100 obs.);  Bandwidth 'bw' = 0.125
##
##        x                 y
##  Min.   :-5.116   Min.   :0.000378
##  1st Qu.:-2.019   1st Qu.:0.037949
##  Median : 1.078   Median :0.073091
##  Mean   : 1.078   Mean   :0.080642
##  3rd Qu.: 4.174   3rd Qu.:0.110306
##  Max.   : 7.271   Max.   :0.248450
```

```r
# plot p-hat(x) and p(x)
plot(density(x, kernel = "gaussian", width = 0.5), main = "p-hat(x) vs p(x): KDE with width = 0.5")
lines(density(x), col = "red")
legend("topright", legend=c("p-hat(x)", "p(x)"),
       col = c("black", "red"), lty = 1, cex = 0.8)
```

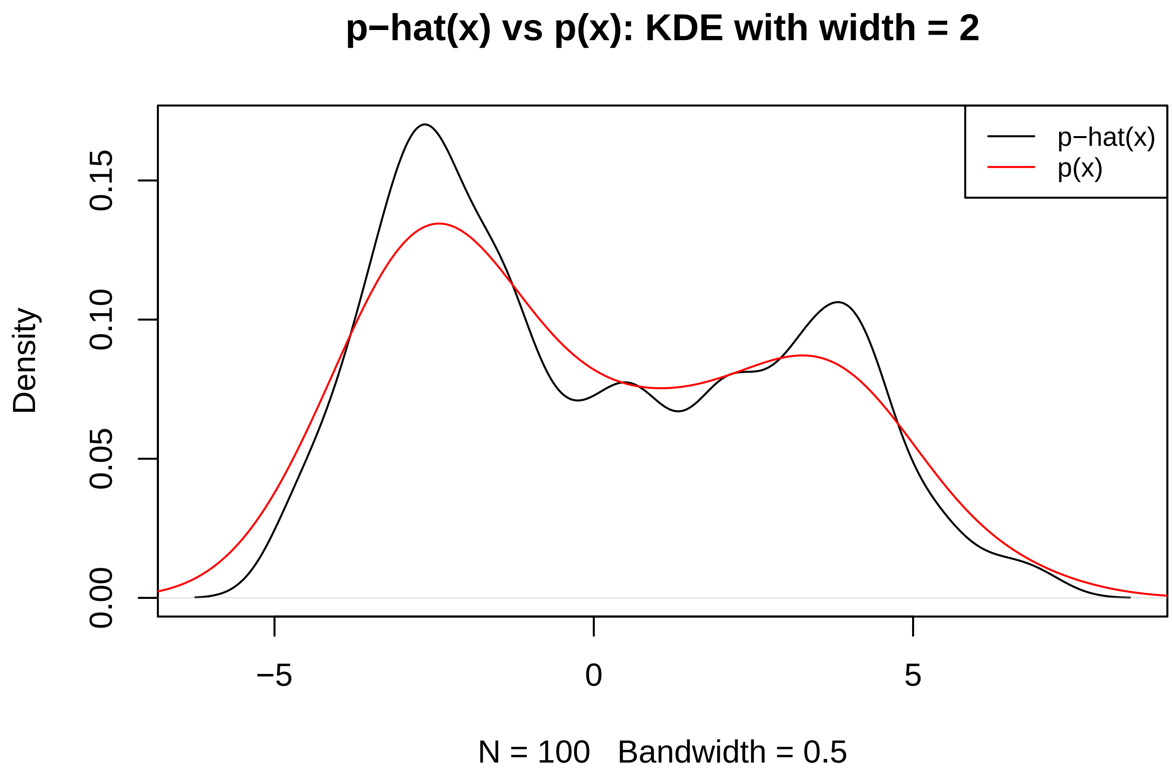


part c

```r
# h = 2
# estimating p-hat(x)
density(x, kernel = "gaussian", width = 2)
```

```
##
## Call:
##  density.default(x = x, kernel = "gaussian", width = 2)
##
## Data: x (100 obs.);  Bandwidth 'bw' = 0.5
##
##        x                 y
##  Min.   :-6.241   Min.   :9.043e-05
##  1st Qu.:-2.582   1st Qu.:2.041e-02
##  Median : 1.078   Median :7.313e-02
##  Mean   : 1.078   Mean   :6.825e-02
##  3rd Qu.: 4.737   3rd Qu.:9.728e-02
##  Max.   : 8.396   Max.   :1.701e-01
```

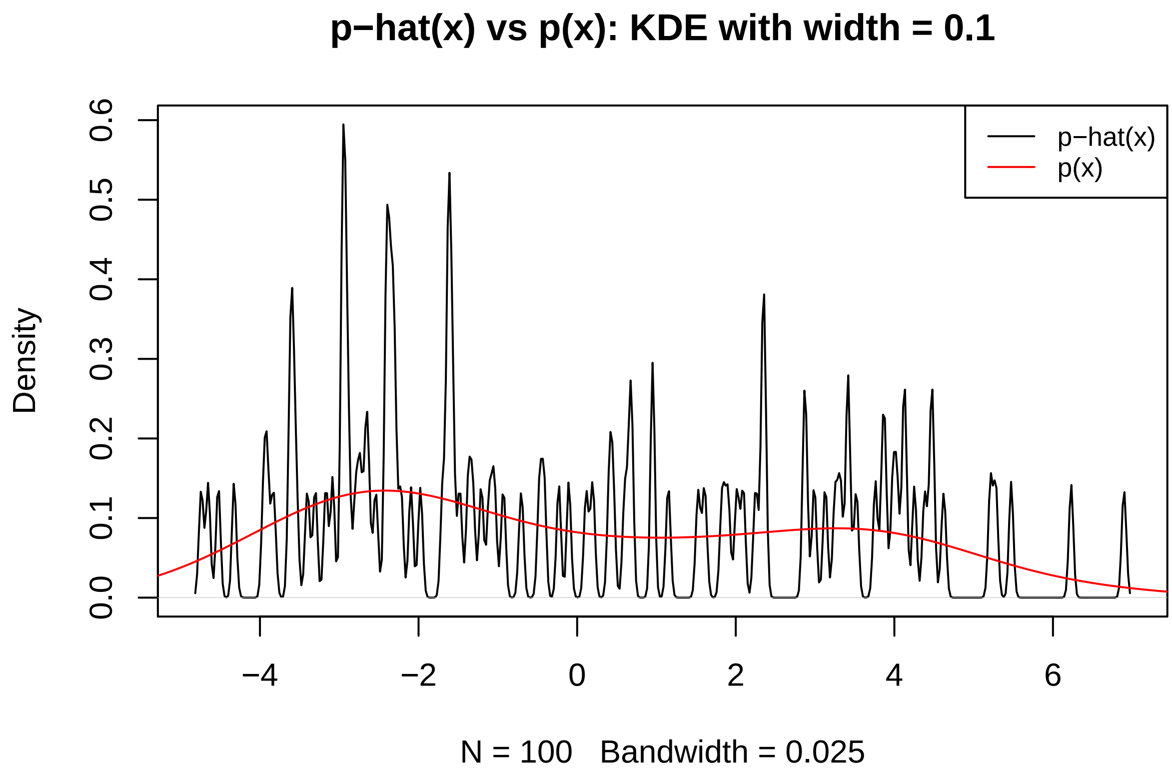
```r
# plot p-hat(x) and p(x)
plot(density(x, kernel = "gaussian", width = 2), main = "p-hat(x) vs p(x): KDE with width = 2")
lines(density(x), col = "red")
legend("topright", legend=c("p-hat(x)", "p(x)"),
       col = c("black", "red"), lty = 1, cex = 0.8)
```




```r
# h = 0.1
# estimating p-hat(x)
density(x, kernel = "gaussian", width = 0.1)
```

```
##
## Call:
##  density.default(x = x, kernel = "gaussian", width = 0.1)
##
## Data: x (100 obs.);  Bandwidth 'bw' = 0.025
##
##        x                 y
##  Min.   :-4.816   Min.   :0.000000
##  1st Qu.:-1.869   1st Qu.:0.001618
##  Median : 1.078   Median :0.064976
##  Mean   : 1.078   Mean   :0.084751
##  3rd Qu.: 4.024   3rd Qu.:0.131961
##  Max.   : 6.971   Max.   :0.594620
```

```r
# plot p-hat(x) and p(x)
plot(density(x, kernel = "gaussian", width = 0.1), main = "p-hat(x) vs p(x): KDE with width = 0.1")
lines(density(x), col = "red")
legend("topright", legend=c("p-hat(x)", "p(x)"),
       col = c("black", "red"), lty = 1, cex = 0.8)
```



When width = 2, the estimate is under-fit, and when width = 0.1, the estimate is over-fit. This makes intuitive sence since larger widths makes an average over more samples and thus we would expect it to be more "smooth" and under-fit by not capturing detailed variations in the true density.

Question 3

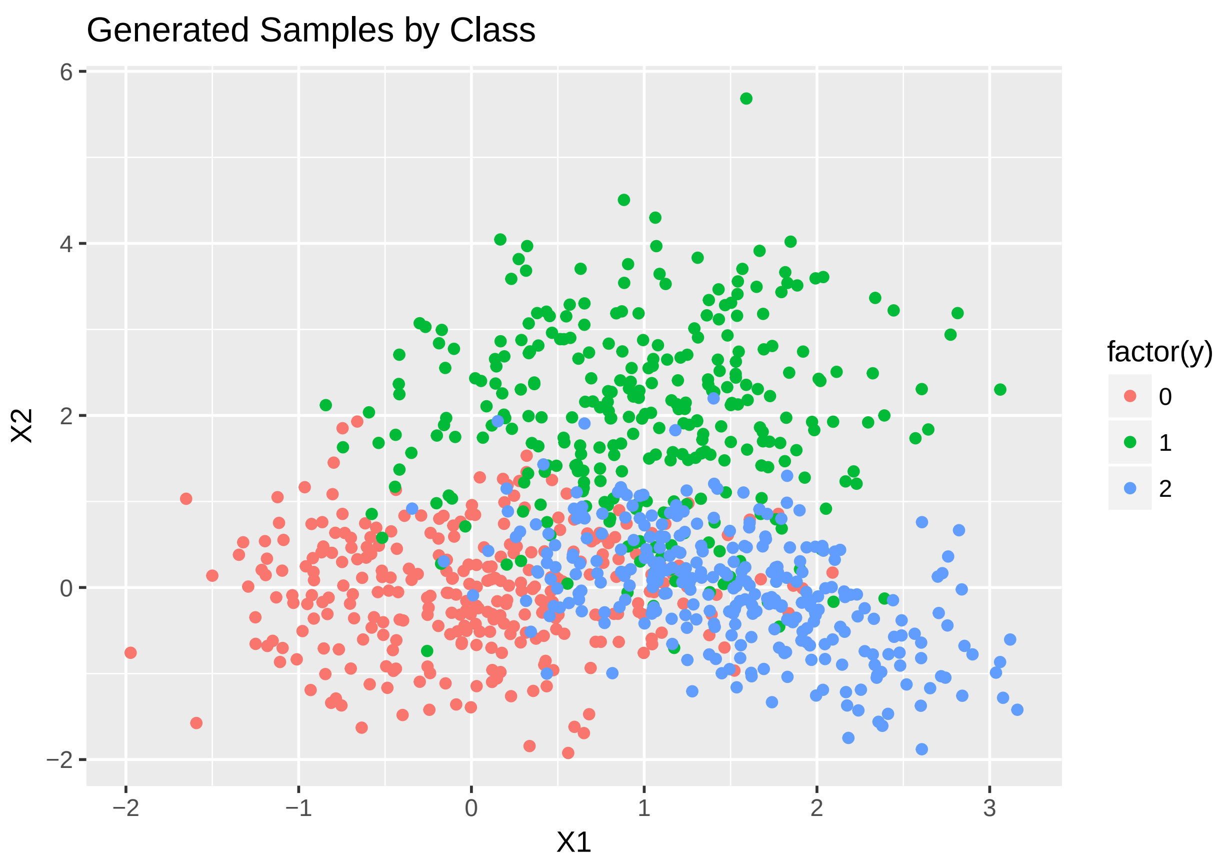
part a

```
set.seed(2018)

# covariance matrices
S0 <- matrix(c(0.5, 0, 0, 0.5), nrow = 2, ncol = 2)
S1 <- matrix(c(0.5, 0, 0, 1), nrow = 2, ncol = 2)
S2 <- matrix(c(0.5, -0.25, -0.25, 0.5), nrow = 2, ncol = 2)

library(mvtnorm)

# generate data
mew0 <- rmvnorm(n = 300, mean = c(0, 0), sigma = S0)
mew1 <- rmvnorm(n = 300, mean = c(1, 2), sigma = S1)
mew2 <- rmvnorm(n = 300, mean = c(1.5, 0), sigma = S2)

# put into dataframe
data <- data.frame(rbind(mew0, mew1, mew2))
```

```r
# add class
data$y <- rep(c(0, 1, 2), each = 300)

# plot the points
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 3.4.3
```

```r
ggplot(data, aes(x = X1, y = X2, col = factor(y))) + geom_point() +
  ggtitle("Generated Samples by Class")
```



part b and part c

```r
# training and testing dataframes
train <- data[sample(nrow(data), 540), ]
test <- data[!data$X1 %in% train$X1, ]

library(class)
```

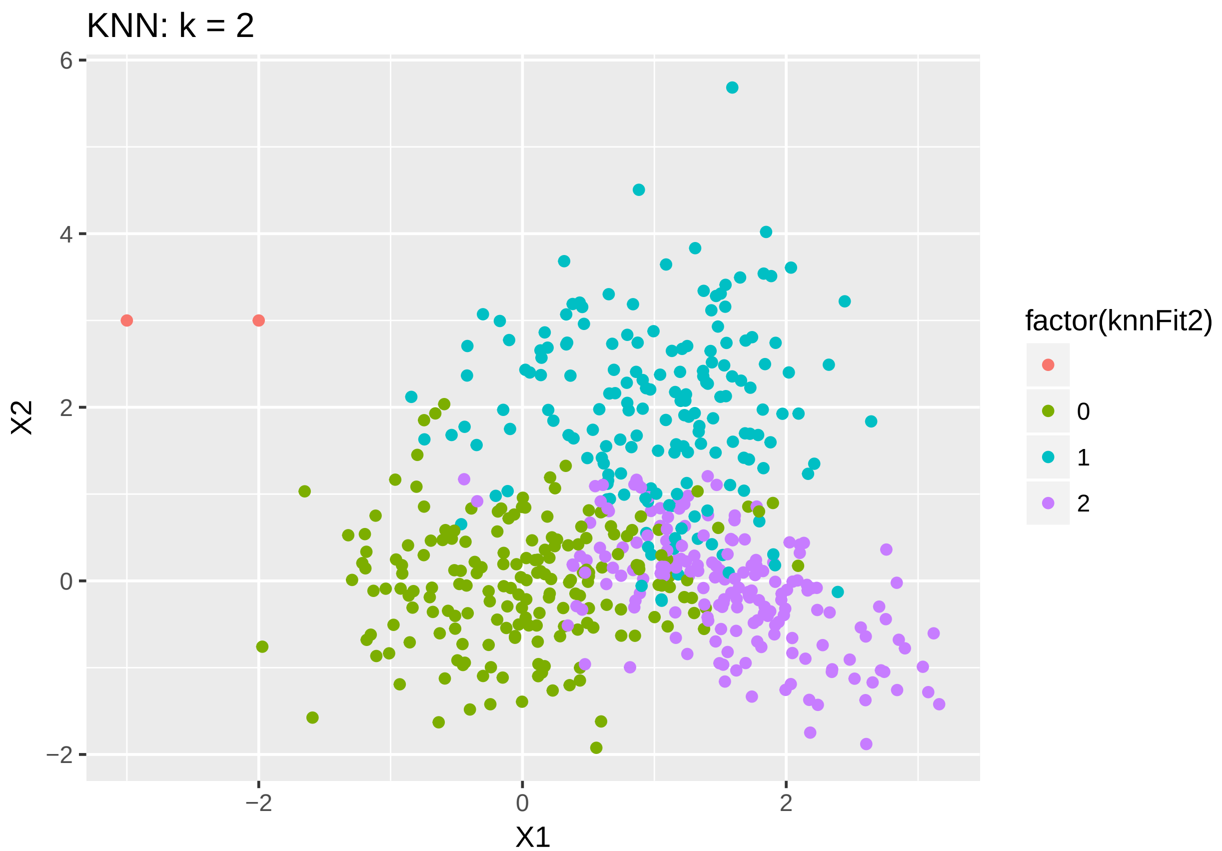
```
## Warning: package 'class' was built under R version 3.4.4
```

```r
# test points
x0 <- c(-3, 3, "")
x1 <- c(-2, 3, "")
```

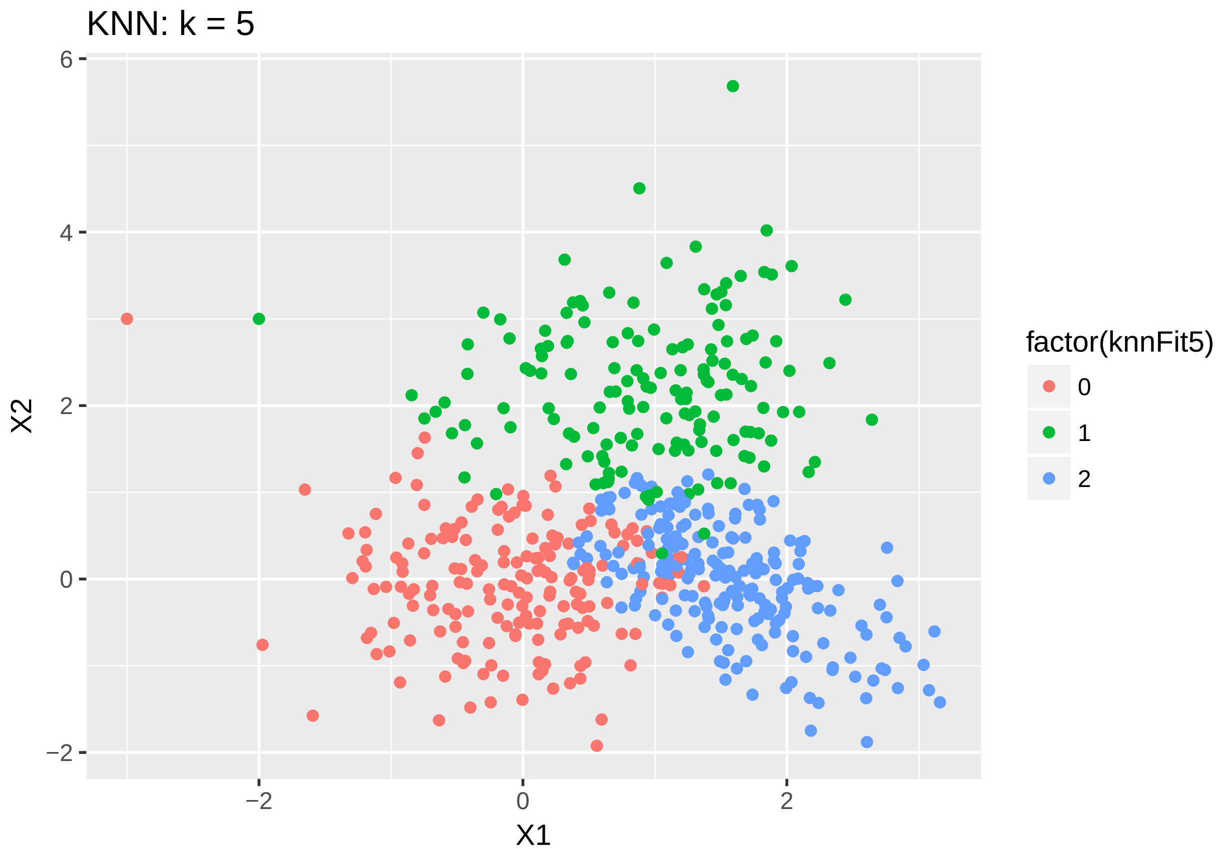
```r
# bind them to data
train <- rbind(train, x0)
train <- rbind(train, x1)

# make numeric
train$X1 <- as.numeric(train$X1)
train$X2 <- as.numeric(train$X2)

# KNN classification
train$knnFit2 <- knn(train = train[ , 1:2], test = train[ , 1:2], cl = train[ , 3], k = 2)
train$knnFit5 <- knn(train = train[ , 1:2], test = train[ , 1:2], cl = train[ , 3], k = 5)
train$knnFit15 <- knn(train = train[ , 1:2], test = train[ , 1:2], cl = train[ , 3], k = 15)

# classification
train[541:542, ]
```

```
##      X1 X2 y knnFit2 knnFit5 knnFit15
## 5411 -3  3                 0        1
## 542  -2  3                 1        1
```

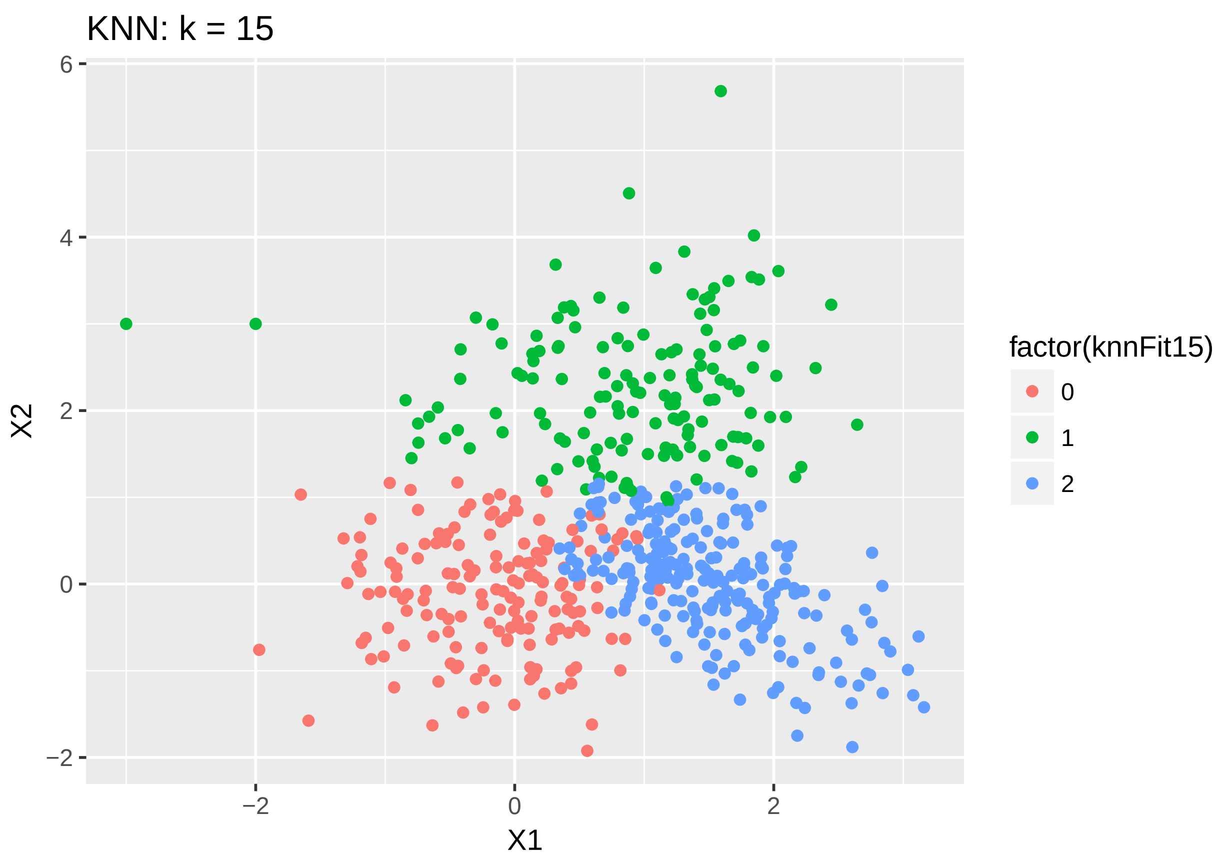
```r
# plot the points
ggplot(train, aes(x = X1, y = X2, col = factor(knnFit2))) + geom_point() +
  ggtitle("KNN: k = 2")
```




```r
ggplot(train, aes(x = X1, y = X2, col = factor(knnFit5))) + geom_point() +
  ggtitle("KNN: k = 5")
```




```r
ggplot(train, aes(x = X1, y = X2, col = factor(knnFit15))) + geom_point() +
  ggtitle("KNN: k = 15")
```




```r
# cross validation
library(caret)
```

```
## Loading required package: lattice
```

```r
library(e1071)
```

```
## Warning: package 'e1071' was built under R version 3.4.4
```

```r
grid = expand.grid(k = c(2, 5, 15))
train(y ~ ., method = "knn", data = train,
              trControl = trainControl(method = "cv", number = 2, search = "grid"),
              tuneGrid = grid)
```

```
## k-Nearest Neighbors
##
## 542 samples
##   5 predictor
##   4 classes: '', '0', '1', '2'
##
## No pre-processing
## Resampling: Cross-Validated (2 fold)
## Summary of sample sizes: 272, 270
## Resampling results across tuning parameters:
##
##   k   Accuracy   Kappa
##    2  0.7988290  0.6987138
##    5  0.8247141  0.7372031
```

```
##   15  0.8431917  0.7648527
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was k = 15.
```

```r
# K = 5 is best model since it has highest accuracy and lowest MSE.

# 2 to 100
grid = expand.grid(k = seq(2:100))
train(y ~ ., method = "knn", data = train,
              trControl = trainControl(method = "cv", number = 2, search = "grid"),
              tuneGrid = grid)
```

```
## k-Nearest Neighbors
##
## 542 samples
##   5 predictor
##   4 classes: '', '0', '1', '2'
##
## No pre-processing
## Resampling: Cross-Validated (2 fold)
## Summary of sample sizes: 270, 272
## Resampling results across tuning parameters:
##
##   k   Accuracy   Kappa
##    1  0.7842184  0.6772674
##    2  0.7584150  0.6383753
##    3  0.8026144  0.7041194
##    4  0.8155637  0.7234831
##    5  0.8155637  0.7234630
##    6  0.8266204  0.7400623
##    7  0.8302832  0.7455444
##    8  0.8339869  0.7511052
##    9  0.8376362  0.7564926
##   10  0.8357843  0.7537005
##   11  0.8450300  0.7675766
##   12  0.8431645  0.7648069
##   13  0.8412990  0.7620017
##   14  0.8394608  0.7592524
##   15  0.8394608  0.7592794
##   16  0.8394472  0.7592630
##   17  0.8357707  0.7537511
##   18  0.8394472  0.7592630
##   19  0.8394608  0.7592759
##   20  0.8394608  0.7592759
##   21  0.8394608  0.7592853
##   22  0.8413126  0.7620651
##   23  0.8394608  0.7592853
##   24  0.8376225  0.7565393
##   25  0.8394608  0.7592993
##   26  0.8394608  0.7592993
##   27  0.8394608  0.7592993
##   28  0.8394608  0.7592993
##   29  0.8394608  0.7592993
##   30  0.8394608  0.7592993
```

```
## 31 0.8394608 0.7592993
## 32 0.8394608 0.7592853
## 33 0.8394608 0.7592853
## 34 0.8413126 0.7620651
## 35 0.8413126 0.7620792
## 36 0.8413126 0.7620651
## 37 0.8413126 0.7620651
## 38 0.8413126 0.7620651
## 39 0.8413126 0.7620651
## 40 0.8413126 0.7620651
## 41 0.8413126 0.7620651
## 42 0.8413126 0.7620651
## 43 0.8413126 0.7620651
## 44 0.8413126 0.7620651
## 45 0.8413126 0.7620651
## 46 0.8413126 0.7620651
## 47 0.8413126 0.7620651
## 48 0.8413126 0.7620651
## 49 0.8413126 0.7620651
## 50 0.8413126 0.7620651
## 51 0.8413126 0.7620651
## 52 0.8413126 0.7620651
## 53 0.8413126 0.7620651
## 54 0.8431509 0.7648299
## 55 0.8431509 0.7648299
## 56 0.8431509 0.7648299
## 57 0.8431509 0.7648299
## 58 0.8431509 0.7648299
## 59 0.8394744 0.7592951
## 60 0.8413126 0.7620651
## 61 0.8431509 0.7648299
## 62 0.8413126 0.7620651
## 63 0.8413126 0.7620604
## 64 0.8413126 0.7620604
## 65 0.8413126 0.7620604
## 66 0.8431509 0.7648299
## 67 0.8413126 0.7620604
## 68 0.8413126 0.7620604
## 69 0.8413126 0.7620604
## 70 0.8413126 0.7620604
## 71 0.8413126 0.7620604
## 72 0.8413126 0.7620604
## 73 0.8413126 0.7620604
## 74 0.8413126 0.7620604
## 75 0.8413126 0.7620604
## 76 0.8413126 0.7620604
## 77 0.8413126 0.7620604
## 78 0.8413126 0.7620604
## 79 0.8413126 0.7620604
## 80 0.8413126 0.7620604
## 81 0.8413126 0.7620604
## 82 0.8413126 0.7620604
## 83 0.8413126 0.7620604
## 84 0.8413126 0.7620604
```

```
##    85  0.8413126  0.7620604
##    86  0.8413126  0.7620604
##    87  0.8413126  0.7620604
##    88  0.8413126  0.7620604
##    89  0.8413126  0.7620604
##    90  0.8413126  0.7620604
##    91  0.8413126  0.7620604
##    92  0.8413126  0.7620604
##    93  0.8413126  0.7620604
##    94  0.8413126  0.7620604
##    95  0.8413126  0.7620604
##    96  0.8413126  0.7620604
##    97  0.8413126  0.7620604
##    98  0.8413126  0.7620604
##    99  0.8413126  0.7620604
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was k = 11.
```

```
# K = 9 is best model with highest accuracy and lowest MSE.
```