Question One

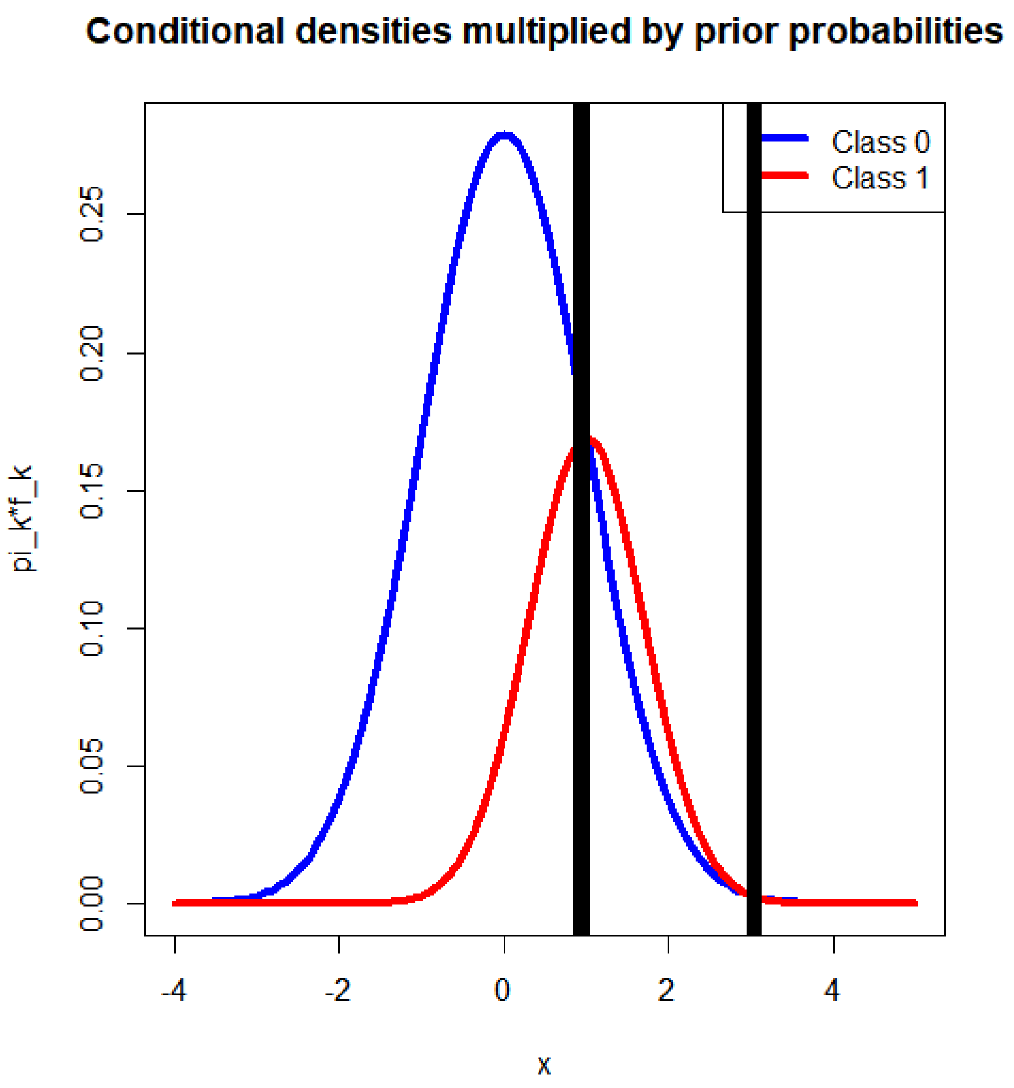
An advantage that a more flexible approach has is that it provides the opportunity to have a low variance and low bias in the model, compared to a more flexible model.

The disadvantage to a more flexible approach is that overfitting of the training data (low training error) can occur which can cause a higher level of testing error if the data has a greater error.

A less flexible approach is preferred when the model is more linear as a flexible approach will add more unnecessary complexity (principle of parsimony). This makes it also harder to explain to someone that does not understand statistics as well.

Question Two

a)



b) $f_0(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}x^2}$

$\pi_0(x) = 0.69$

$f_1(x) = \frac{1}{\sqrt{\pi}} e^{(-(x-1)^2)}$

$\pi_1(x) = 0.31$

Bayes decision boundary:

$f_0(x) * \pi_0(x) = f_1(x) * \pi_1(x)$

$\frac{0.69}{\sqrt{2\pi}} e^{-\frac{1}{2}x^2} = \frac{0.31}{\sqrt{\pi}} e^{(-(x-1)^2)}$

$= \frac{0.69}{\sqrt{2}} e^{-\frac{1}{2}x^2} = 0.31e^{(-(x-1)^2)}$

$= \frac{0.69}{0.31\sqrt{2}} e^{-\frac{1}{2}x^2} = e^{(-(x-1)^2)}$

$= \ln\left(\frac{0.69}{0.31\sqrt{2}}\right) - \frac{1}{2}x^2 = -(x-1)^2$

$= \ln\left(\frac{0.69}{0.31\sqrt{2}}\right) = \frac{1}{2}x^2 - (x-1)^2$

$= \ln\left(\frac{0.69}{0.31\sqrt{2}}\right) = \frac{1}{2}x^2 - (x^2 - 2x + 1)$

$= \ln\left(\frac{0.69}{0.31\sqrt{2}}\right) = \frac{1}{2}x^2 - x^2 + 2x - 1$

$= \ln\left(\frac{0.69}{0.31\sqrt{2}}\right) = -\frac{x^2}{2} + 2x - 1$

$= -\frac{x^2}{2} + 2x - 1 - \ln\left(\frac{0.69}{0.31\sqrt{2}}\right) = 0$

$= x = 0.955$ or $x = 3.045$ (3 d.p)

c) $\frac{f_0(x)*\pi_0(x)}{f_0(x)*\pi_0(x)+f_1(x)*\pi_1(x)}$ and $X = 3$

$$= \frac{\frac{0.69}{\sqrt{2\pi}}e^{-\frac{1}{2}x^2}}{\frac{0.69}{\sqrt{2\pi}}e^{-\frac{1}{2}x^2} + \frac{0.31}{\sqrt{\pi}}e^{(-(x-1)^2)}}$$

$$\text{Class 0} = \frac{\frac{0.69}{\sqrt{2\pi}}e^{-\frac{1}{2}*3^2}}{\frac{0.69}{\sqrt{2\pi}}e^{-\frac{1}{2}*3^2} + \frac{0.31}{\sqrt{\pi}}e^{(-(3-1)^2)}} = 0.4884$$

$$\frac{f_1(x)*\pi_1(x)}{f_0(x)*\pi_0(x)+f_1(x)*\pi_1(x)}$$

$$\text{Class 1} = \frac{\frac{0.31}{\sqrt{\pi}}e^{(-(3-1)^2)}}{\frac{0.69}{\sqrt{2\pi}}e^{-\frac{1}{2}*3^2} + \frac{0.31}{\sqrt{\pi}}e^{(-(3-1)^2)}} = 0.5116$$

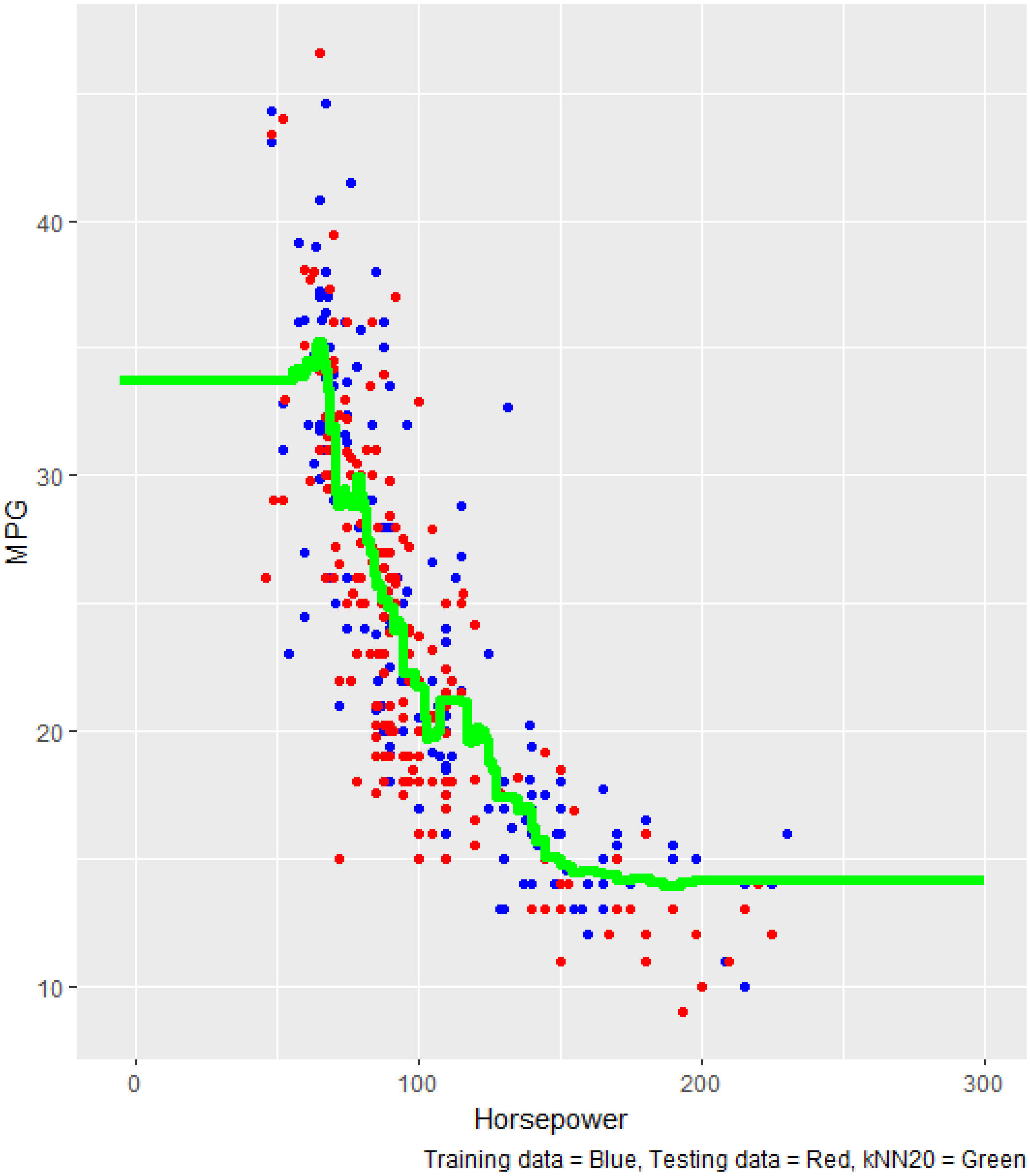
I would choose Class 1 as it has the higher probability of being in that class, with probability = 0.5116.
Also, looking at the graph in 2a, it is within the Bayes Decision Boundary which confirms its class.

d) $$\text{Class 1} = \frac{\frac{0.31}{\sqrt{\pi}}e^{(-(2-1)^2)}}{\frac{0.69}{\sqrt{2\pi}}e^{-\frac{1}{2}*2^2} + \frac{0.31}{\sqrt{\pi}}e^{(-(2-1)^2)}} = 0.5498$$

<u>Question 3</u>

a) R-Code is at the end.
b)
TestMSE2
22.86349
TestMSE5
19.56322
TestMSE10
18.62914
TestMSE20
17.31858
TestMSE30
17.93018
TestMSE50
19.57374
TestMSE100
26.31542
TestMSE100
26.31542

K = 20 performed the best as it had the lowest MSE for the testing data set with MSE = 17.31858 c)



d) Increasing the level of flexibility (1/K) by reducing the number of K neighbours will decrease the bias of the function however it will also increase the variance. In the example above, low levels of K had high levels of variance however low amounts of bias and by increasing the value of K we reduced our MSE until we reached the optimum amount at K = 20. Above K = 20, we saw the bias begin to increase faster than the reduction in variance, causing the MSE to increase.

R-Code:

```
## Normal plots

x = seq(-4,5,length=100)
plot(x,
     0.7*dnorm(x, 0, sqrt(1)),
     pch=21,
     col="blue",
     cex=0.6,
     lwd = 4,
     type="l",
     xlab = "x",
     ylab = "pi_k*f_k",
     main = "Conditional densities multiplied by prior probabilities")
points(x,
       0.30*dnorm(x,1,sqrt(0.5)),
       pch=21,
       col="red",
       cex=0.6,
       lwd = 4,
       type="l")
legend("topright",
       legend = c("Class 0", "Class 1"),
       col = c("blue","red"),
       lwd = 4,
       text.col = "black",
       horiz = FALSE)

points(c(0.955,0.955),
       c(-0.1,0.3),
       lwd = 8,
       col = "black",
       type="l")

points(c(3.045,3.045),
       c(-0.1,0.3),
       lwd = 8,
       col = "black",
       type="l")

#########################################################################
library(ggplot2)

data = read.csv('AutoTrain.csv')
data2 = read.csv('AutoTest.csv')

## STAT318/462 kNN regression function

kNN <- function(k,x.train,y.train,x.pred) {
  #
  ## This is kNN regression function for problems with
  ## 1 predictor
  #
  ## INPUTS
  #
  # k       = number of observations in nieghbourhood
  # x.train = vector of training predictor values
  # y.train = vector of training response values
  # x.pred  = vector of predictor inputs with unknown
  #           response values
  #
```

```
  ## OUTPUT
  #
  # y.pred  = predicted response values for x.pred

  ## Initialize:
  n.pred <- length(x.pred);         y.pred <- numeric(n.pred)

  ## Main Loop
  for (i in 1:n.pred){
    d <- abs(x.train - x.pred[i])
    dstar = d[order(d)[k]]
    y.pred[i] <- mean(y.train[d <= dstar])
  }
  ## Return the vector of predictions
  invisible(y.pred)
}

kNN2<-kNN(2, data$horsepower, data$mpg, data2$horsepower)
kNN2
TrainMSE2 = mean((data$mpg - kNN2)^2)
TestMSE2 = mean((data2$mpg - kNN2)^2)
TrainMSE2
TestMSE2

kNN5<-kNN(5, data$horsepower, data$mpg, data2$horsepower)
kNN5
TrainMSE5 = mean((data$mpg - kNN5)^2)
TestMSE5 = mean((data2$mpg - kNN5)^2)
TrainMSE5
TestMSE5

kNN10<-kNN(10, data$horsepower, data$mpg, data2$horsepower)
kNN10
TrainMSE10 = mean((data$mpg - kNN10)^2)
TestMSE10 = mean((data2$mpg - kNN10)^2)
TrainMSE10
TestMSE10

kNN20<-kNN(20, data$horsepower, data$mpg, data2$horsepower)
kNN20
TrainMSE20 = mean((data$mpg - kNN20)^2)
TestMSE20 = mean((data2$mpg - kNN20)^2)
TrainMSE20
TestMSE20

kNN30<-kNN(30, data$horsepower, data$mpg, data2$horsepower)
kNN30
TrainMSE30 = mean((data$mpg - kNN30)^2)
TestMSE30 = mean((data2$mpg - kNN30)^2)
TrainMSE30
TestMSE30

kNN50<-kNN(50, data$horsepower, data$mpg, data2$horsepower)
kNN50
TrainMSE50 = mean((data$mpg - kNN50)^2)
TestMSE50 = mean((data2$mpg - kNN50)^2)
TrainMSE50
TestMSE50

kNN100<-kNN(100, data$horsepower, data$mpg, data2$horsepower)
kNN100
TrainMSE100 = mean((data$mpg - kNN100)^2)
TestMSE100 = mean((data2$mpg - kNN100)^2)
TrainMSE100
```

```
TestMSE100

TestMSE2
TestMSE5
TestMSE10
TestMSE20
TestMSE30
TestMSE50
TestMSE100




kNN20

x_sample = seq(-5, 300, length=10000)
knnx <- kNN(20, data$horsepower, data$mpg, x_sample)
knnx
k<-data.frame(x_sample, knnx)



ggplot() +
  geom_point(data = data, aes(x = horsepower, y = mpg), color = 'blue') +
  geom_point(data = data2, aes(x = horsepower, y = mpg), color = 'red') +
xlab('Horsepower') + ylab('MPG') +
  geom_line(data = k, aes(x = k[,1], y = k[,2]), color = "green", lwd=2) +
  scale_color_manual(values=c('Training Data' = 'blue', 'Testing
Data'='red', 'kNN20' = 'Green')) +
  labs(caption = 'Training data = Blue, Testing data = Red, kNN20 =
Green')
```