Marcus Crowder
STA 3920
Professor Lawrence Tatum
LN-6

6.1

By using the head() function I was able to find which Rows contained Sepal length and Sepal Width. With that Data I was able to run the NaiveBayes program from the R e1071 Package. The program failed to predict 32 of 150 Iris flowers. This is a less desirable result when compared to the result when using NaiveBayes along with the Sepal Length, Sepal Width, Petal Length, and Petal Width which only produced 6 of 150 errors. The explanation for this undesirable result lies in the amount of data that was used in both experiments. For my experiment, I only used the Sepal Length and Width while the other experiment added Petal Length and Width. With more data points and variations to classify the plants a better prediction was made on the other experiment.

```
> head(iris)
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
4          4.6         3.1          1.5         0.2  setosa
5          5.0         3.6          1.4         0.2  setosa
6          5.4         3.9          1.7         0.4  setosa
> classifier <- naiveBayes(iris[,1:2], iris[,5])
> table(predict(classifier, iris[,-5]), iris[,5])

             setosa versicolor virginica
  setosa         49          0         0
  versicolor      1         37        19
  virginica       0         13        31
```

6.2

For this exercise, I used the Cross Validation program on the Sepal Length and Sepal Width with similar results to the problem above. The results of this exercise produced more errors when only using Sepal.Length and Sepal.Width when compared to using Sepal.Length, Sepal.Width, Petal.Length, Petal.Width. This all but confirms that the Petal Width and Petal Length have a great prediction impact when it comes to classifying the species of flowers. There were 12 incorrect classifications.

```
> x.train = iris[train, 1:2]
> y.train = iris[ train, 5]
> head(x.train)
    Sepal.Length Sepal.Width
15           5.8         4.0
23           4.6         3.6
36           5.0         3.2
43           4.4         3.2
142          6.9         3.1
86           6.0         3.4
> x.test = iris[ -train, 1:2]
> y.test = iris[ -train, 5]
> > model = train(x.train,y.train,'nb',trControl=trainControl(method='cv',number=10))
Error: unexpected '>' in ">"
> model = train(x.train,y.train,'nb',trControl=trainControl(method='cv',number=10))
> table(predict(model$finalModel,x.test)$class,y.test)
            y.test
             setosa versicolor virginica
  setosa         17          0         0
  versicolor      0         10         9
  virginica       0          3        11
```

The first step was to choose a random sample for our data with the sample() function. For this exercise we used 100 points for training and 50 points for testing the classification of our flowers.

> train = sample(150,100)
> head(train)
[1] 41 93 103 113 122 72
> train = sample(150,100)

6.3

I decided to use my The Stock Market data for MKC stock. I created the Lag1 and Lag2 ranges in excel then stripped the first two values in the NaïveBayes because they were NA. I learned a neat slicing trick to remove the first two values at once by using MKCMarket[-(1:2)] as you can see it took trial and error to get the commands down but I was satisfied with the end result because there was a total of 2517 Data points between High and Low Risk.

In total the program was able to classify correctly 1768 of 2517 data points which is a modest 70% success rate but I feel happy that it worked. I will have to keep working to understand why there are 749 miss classified points.

```
> MKCMarket <- read.csv("~/Downloads/MKC.csv")
> head(MKCMarket)
     Date  Open  High   Low Close Adj.Close Volume    Range Lag1Range Lag2Range    Risk  X Median.Range
1  9/4/07 35.93 36.09 35.50 35.79  28.54640 618700 0.590000        NA        NA LowRisk NA     0.730004
2  9/5/07 35.52 35.72 35.27 35.29  28.14758 538500 0.450001  0.590000        NA LowRisk NA           NA
3  9/6/07 35.45 35.75 35.29 35.34  28.18746 382100 0.459999  0.450001  0.590000 LowRisk NA           NA
4  9/7/07 35.08 35.41 34.93 34.99  27.90830 537900 0.480000  0.459999  0.450001 LowRisk NA           NA
5 9/10/07 35.22 35.26 34.55 34.79  27.74878 470900 0.709999  0.480000  0.459999 LowRisk NA           NA
6 9/11/07 34.82 35.38 34.82 35.33  28.17948 613100 0.560001  0.709999  0.480000 LowRisk NA           NA
> classifier <- naiveNayes(MKCMarket[-(1:2),9:10], iris[,11])
Error in naiveNayes(MKCMarket[-(1:2), 9:10], iris[, 11]) :
  could not find function "naiveNayes"
> classifier <- naiveBayes(MKCMarket[-(1:2),9:10], iris[,11])
Error in `[.data.frame`(iris, , 11) : undefined columns selected
> classifier <- naiveBayes(MKCMarket[-(1:2),9:10], MKCMarket[,11])
Error in tapply(var, y, mean, na.rm = TRUE) :
  arguments must have same length
> classifier <- naiveBayes(MKCMarket[-(1:2),9:10], MKCMarket[-(1:2),11])
> table(predict(classifier, MKCMarket[-(1,2), 9:10]) MKCMarket[-(1:2),11])
Error: unexpected ',' in "table(predict(classifier, MKCMarket[-(1,"
> table(predict(classifier, MKCMarket[-(1,2), 9:10]), MKCMarket[-(1:2),11])
Error: unexpected ',' in "table(predict(classifier, MKCMarket[-(1,"
> table(predict(classifier, MKCMarket[-(1:2), 9:10]), MKCMarket[-(1:2),11])

          HIghRisk LowRisk
  HIghRisk      670     158
  LowRisk       591    1098
```

> classifier <- naiveBayes(MKCMarket[-(1:2),9:10], MKCMarket[-(1:2),11])
> table(predict(classifier, MKCMarket[-(1:2), 9:10]), MKCMarket[-(1:2),11])

I decided to use the same Lag1 and Lag2 ranges for the cross validation by selecting a random sample of data points to test against the rest of my data points. It turned out it was not the best of ideas to sample only 150 and use that to test against the rest of my over 2000+ values of data but this is a good learning experience. I may have used too little data points because of the 839 miss classifications. Which is less than the 70% success rate above but still a decent success rate 67%. Then again it could also be because of the random sample anyway it was fun to experiment with.

```
> MKCMarket = MKCMarket[-(1:2),]
> head(MKCMarket)
     Date  Open  High   Low Close Adj.Close Volume    Range Lag1Range Lag2Range    Risk  X Median.Range
3  9/6/07 35.45 35.75 35.29 35.34  28.18746 382100 0.459999  0.450001  0.590000 LowRisk NA           NA
4  9/7/07 35.08 35.41 34.93 34.99  27.90830 537900 0.480000  0.459999  0.450001 LowRisk NA           NA
5 9/10/07 35.22 35.26 34.55 34.79  27.74878 470900 0.709999  0.480000  0.459999 LowRisk NA           NA
6 9/11/07 34.82 35.38 34.82 35.33  28.17948 613100 0.560001  0.709999  0.480000 LowRisk NA           NA
7 9/12/07 35.30 35.70 35.11 35.40  28.23532 539600 0.590000  0.560001  0.709999 LowRisk NA           NA
8 9/13/07 35.70 35.73 35.40 35.58  28.37889 386600 0.329998  0.590000  0.560001 LowRisk NA           NA
> train=sample(150,100)
> x.train = MKCMarket[train, 9:10]
> y.train = MKCMarket[train, 11]
> head(x.train)
    Lag1Range Lag2Range
3    0.450001  0.590000
13   0.710000  0.790001
141  0.460003  0.870003
68   0.509998  0.479999
93   0.400002  0.829998
113  0.590001  1.070000
> x.test = MKCMarket[ -train, 9:10]
> y.test = MKCMarket[ -train, 11]
> model = train(x.train,y.train,'nb',trControl=trainControl(method='cv',number=10))
> table(predict(model$finalModel,x.test)$class,y.test)
         y.test
          HIghRisk LowRisk
  HIghRisk      503     116
  LowRisk       723    1075
```

I used my Lag1Range and Lag2Range Values to train my model and used the Risk to test the points.

> train=sample(150,100)
> x.train = MKCMarket[train, 9:10]
> y.train = MKCMarket[train, 11]
> x.test = MKCMarket[-train, 9:10]
> y.test = MKCMarket[-train, 11]
> model = train(x.train,y.train,'nb',trControl=trainControl(method='cv',number=10))

6.4
Suppose the disease occurs at a rate of 1/500 in the population. A test has a false negative rate of 2% and a false positive rate of 4%. Given a positive test, find the probability the disease is present.
I used common sense for this problem.
Disease = 1/500
False negative = 2%
False positive = 4%
Considering a population of 50,000 there will be 100 cases of the disease
On average .98 * 100 = 98 will be detected while 2 will be missed
There will be 49,900 cases without the disease
Of those .04 * 49,900 = 1,996 will be a false positive
98 + 1,996 = 2094
98/2094 = 0.0468 = 4.68%
there is a 4.68% chance that the disease will be present.

6.5
P(mu = 0.6| k=3)
This problem is a continuation from LN6
The first thing to be done is finding
P(k=3|mu=.6) = 5C3 * .06^3 * .04^2 = 0.3456

Then we use the denominator which was provided to use by the professor
Denominator = . 30025
P(mu=.6) = .25 (chance of bag with 6 black marbles)

P(k=3|mu=.6)*P(mu=.6) / .30025 = .2877602 = 28.78%

P(mu=0.6|k=3) = 28.78%

6.6
The prior distribution for all of these variables equates to
P=(mu=7/16) = P(mu=8/16) = P(mu=9/16) = 1/3
A bag contains 16 marbles with 6,7 or 8 black marbles
This is termed non informative prior

Denominator = .265025397

(a)

6.7
What would be meant by the symbol mu122(Virginica)?
This would represent the probability of the item in the 122nd row belonging to the Virginica Specimen.