Hw-2

Q1

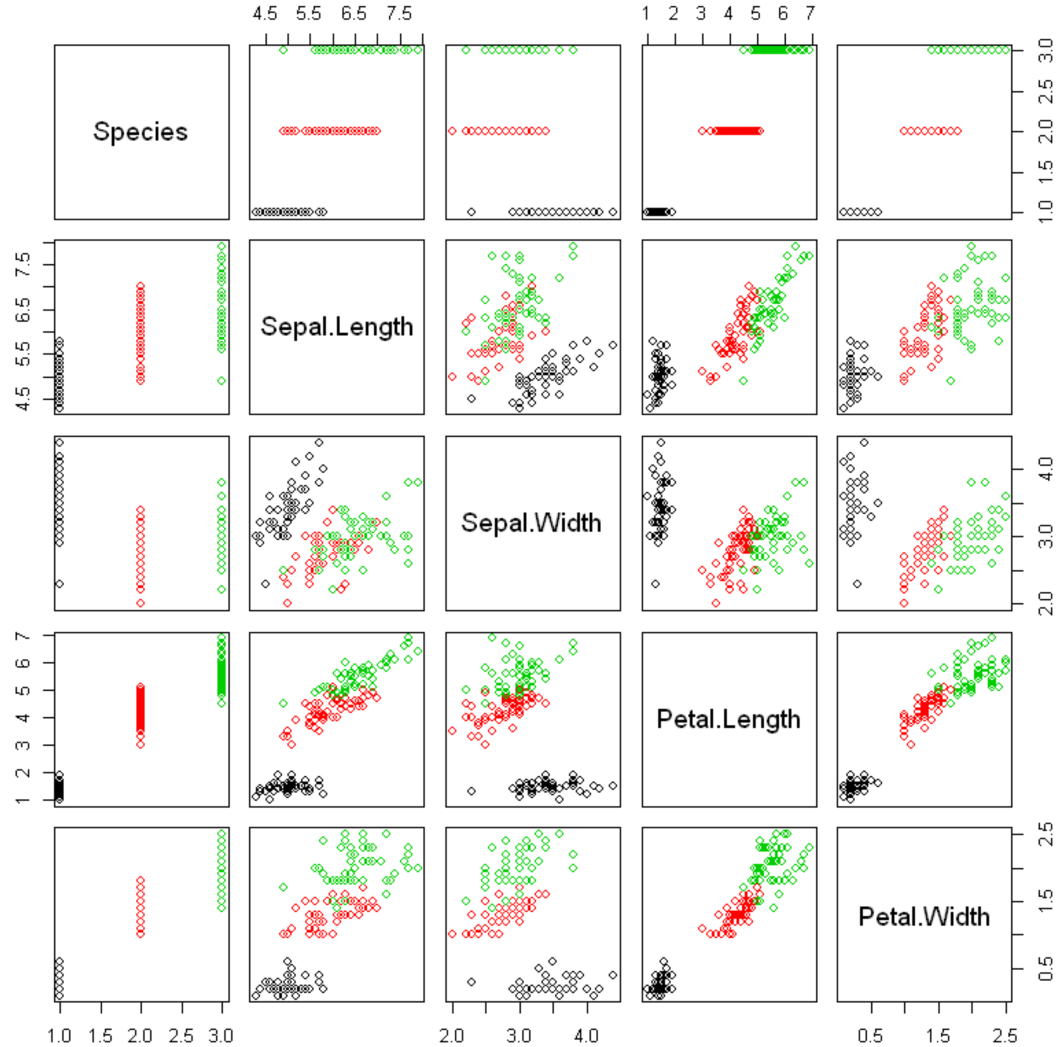
I like netflix. My understanding is that recommender systems work by clustering myself with other users and then making recommendations from their viewing.

Q2

In [1]:

```
# Load data
iris <- read.table('iris.txt', header=TRUE)
head(iris,5)

# There are 6 variables one is just an id , the species is the classification

# we can see some clear structure in this data
# I will begin by running a kmeans on all the data then drop the sepal values as there
 is more overlap here
pairs(Species~., data=iris[,2:6], col=iris$Species)
```

Num	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



We can see that there is more overlay of the classifcations on some pairs then others, in particular it looks like petal.length vs petal.width is clean and possibly petal.width vs sepal.width. We will run all features in a cluster and then see if we can get better results from a subset.

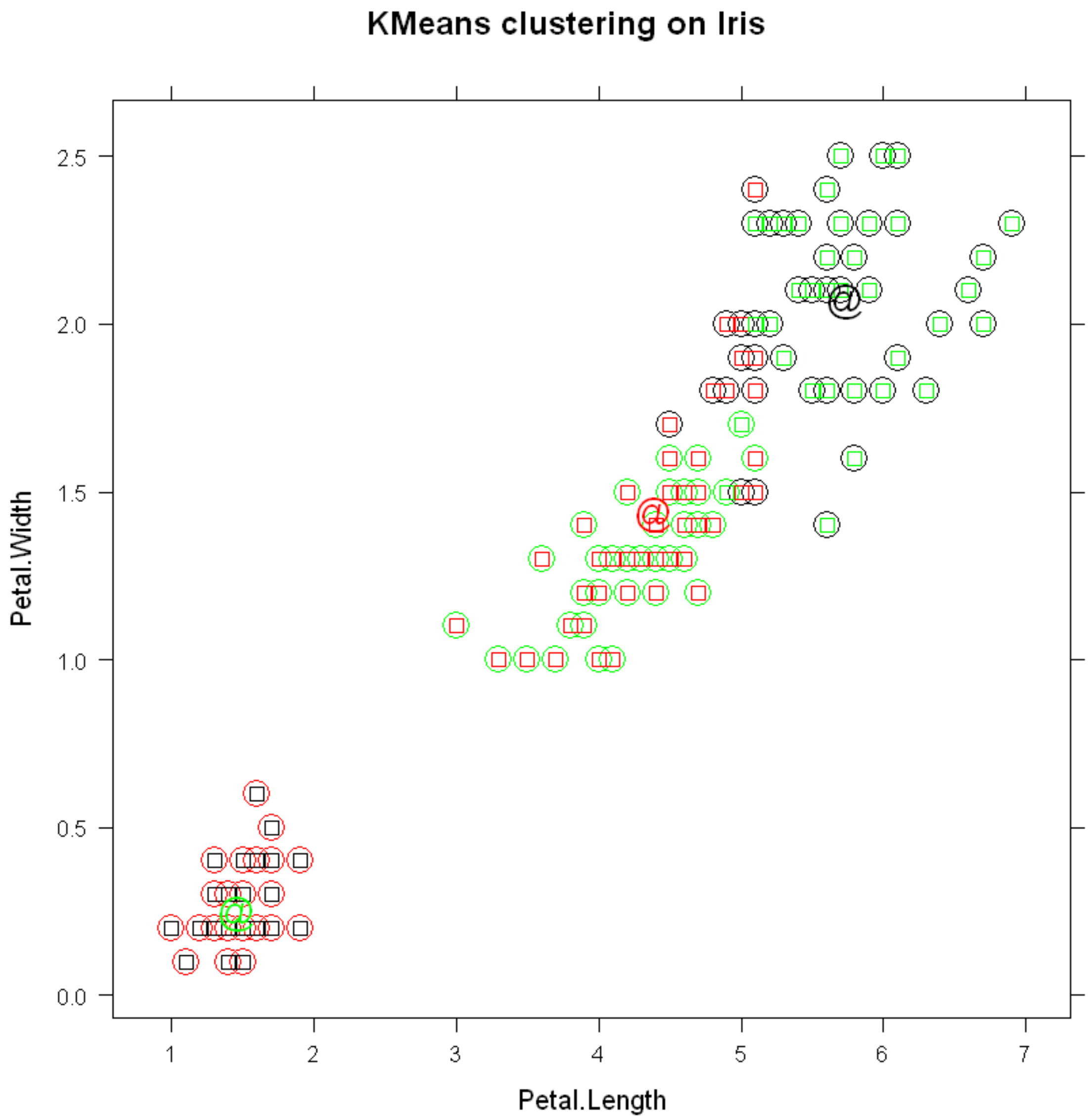
In [35]:

```
# attach lattice for the plotting
library('latticeExtra')
# set seed for repeatability
set.seed(42)

# this functions runs the kmeans on the data, and then outputs the classification accur
acy
# I will call it as I drop elements from the set I am computing the means on
run <- function(data, iris){

  results <- kmeans(data, 3, iter.max = 10, nstart = 1, algorithm = "Hartigan-Wong")
  print(results)

  data$Species <- data.frame(unlist(results[1]))[,1]

  A <- xyplot(Petal.Width ~ Petal.Length, group = Species, data = data, auto.key = 
list(space = "right"),
              par.settings = list(superpose.symbol = list(pch = 0, cex = 1, col = c("re
d", "green", "black"))))
  B <- xyplot(Petal.Width ~ Petal.Length, group = Species, data = iris, auto.key = 
list(space = "rigth"),
              par.settings = list(superpose.symbol = list(pch = 1, cex = 2, col = c("re
d", "green", "black"))), main = "KMeans clustering on Iris")
  C <- xyplot(results$centers[,c("Petal.Width")] ~ 
results$centers[,c("Petal.Length")],pch = "@", cex = 2,
              col = c("red", "black", "green"), auto.key = list(space = "right"))
  D <- B + as.layer(A + C)


  results <- data.frame(table(iris$Species, results$cluster))
  names(results) <- c("Real.Value", "Assigned.Cluster", "Frequency")
  accuracy <- (max(results$Frequency[1:3])+max(results$Frequency[4:6])+max(results$Freq
uency[7:9]))/150

  print(D)
  print(results)
  print(accuracy)
}
```

In [36]:

```
# remove id and label
iris_trim <- iris[,2:5]

#run cluster with all features
results <- run(iris_trim, iris)
```

```
K-means clustering with 3 clusters of sizes 62, 38, 50

Cluster means:
  Sepal.Length Sepal.Width Petal.Length Petal.Width
1     5.901613    2.748387     4.393548    1.433871
2     6.850000    3.073684     5.742105    2.071053
3     5.006000    3.428000     1.462000    0.246000

Clustering vector:
  [1] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
 3 3 3
 [38] 3 3 3 3 3 3 3 3 3 3 3 3 3 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1
 [75] 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 2 2 2 2 1 2
 2 2 2
[112] 2 2 1 1 2 2 2 2 1 2 1 2 1 2 2 1 1 2 2 2 2 2 1 2 2 2 2 1 2 2 2 1 2 2
 2 1 2
[149] 2 1

Within cluster sum of squares by cluster:
[1] 39.82097 23.87947 15.15100
 (between_SS / total_SS =  88.4 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withi
nss"
[6] "betweenss"    "size"         "iter"         "ifault"
  Real.Value Assigned.Cluster Frequency
1     setosa                1         0
2 versicolor                1        48
3  virginica                1        14
4     setosa                2         0
5 versicolor                2         2
6  virginica                2        36
7     setosa                3        50
8 versicolor                3         0
9  virginica                3         0
[1] 0.8933333
```

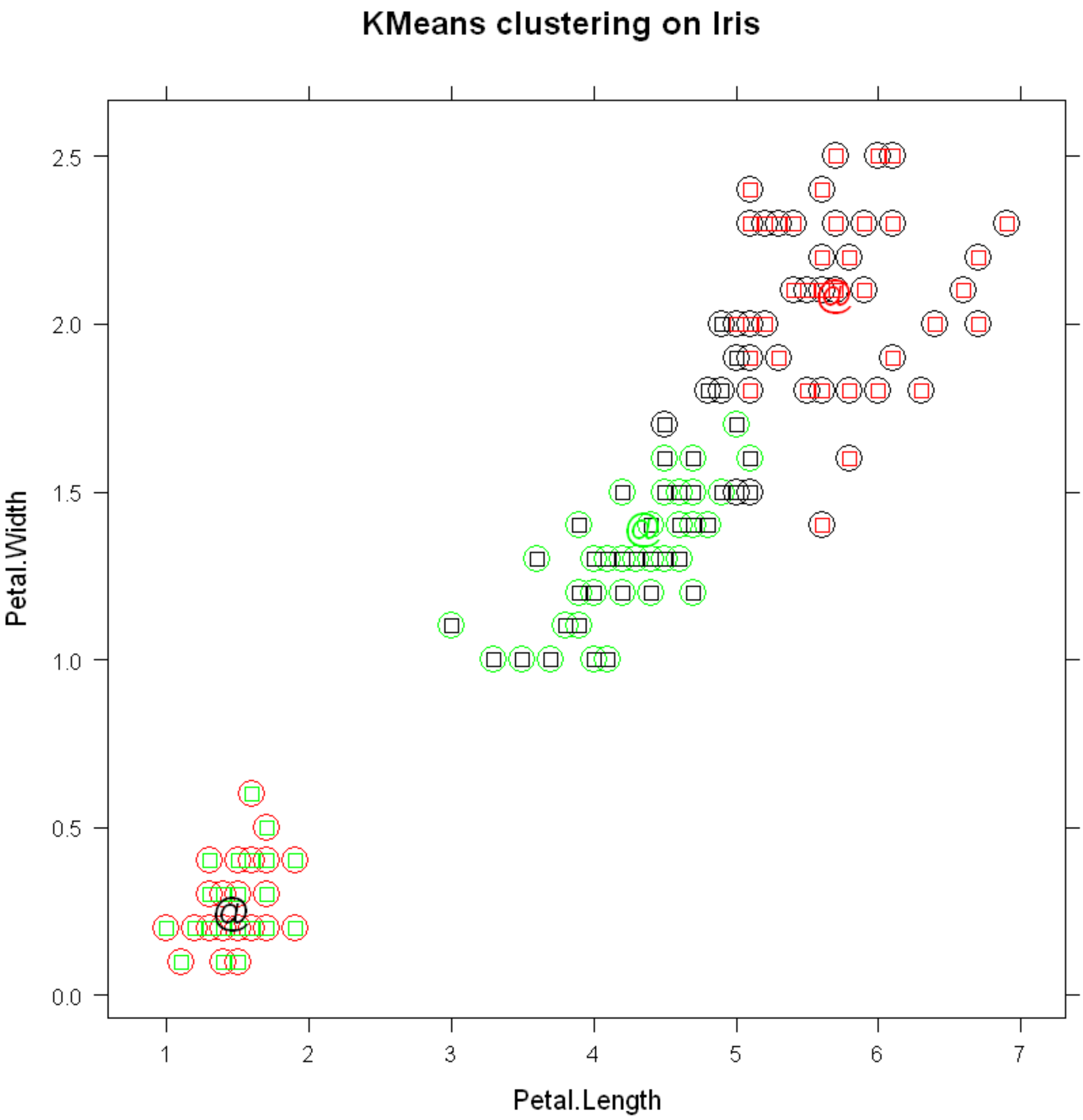


89.3% accuracy. So this cluster is quite good but we may improve by removing features.

In [37]:

```
# drop sepal.length
iris_trim<-subset(iris_trim, select = -Sepal.Length)
results <- run(iris_trim, iris)
```

```
K-means clustering with 3 clusters of sizes 41, 50, 59

Cluster means:
  Sepal.Width Petal.Length Petal.Width
1    3.034146     5.700000    2.085366
2    3.428000     1.462000    0.246000
3    2.759322     4.354237    1.391525

Clustering vector:
  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  1
9  20
  2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
 2   2
 21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  3
9  40
  2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
 2   2
 41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  5
9  60
  2   2   2   2   2   2   2   2   2   2   3   3   3   3   3   3   3   3
 3   3
 61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  7
9  80
  3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3
 3   3
 81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  9
9 100
  3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3
 3   3
101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 11
9 120
  1   1   1   1   1   1   3   1   1   1   1   1   1   1   1   1   1   1
 1   3
121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 13
9 140
  1   3   1   3   1   1   3   3   1   1   1   1   1   3   1   1   1   1
 3   1
141 142 143 144 145 146 147 148 149 150
  1   1   1   1   1   1   3   1   1   1

Within cluster sum of squares by cluster:
[1] 16.10341  9.06280 22.91458
 (between_SS / total_SS =  91.7 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withi
nss"
[6] "betweenss"    "size"         "iter"         "ifault"
  Real.Value Assigned.Cluster Frequency
1     setosa                1         0
2 versicolor                1         0
3  virginica                1        41
4     setosa                2        50
5 versicolor                2         0
6  virginica                2         0
7     setosa                3         0
8 versicolor                3        50
9  virginica                3         9
[1] 0.94
```

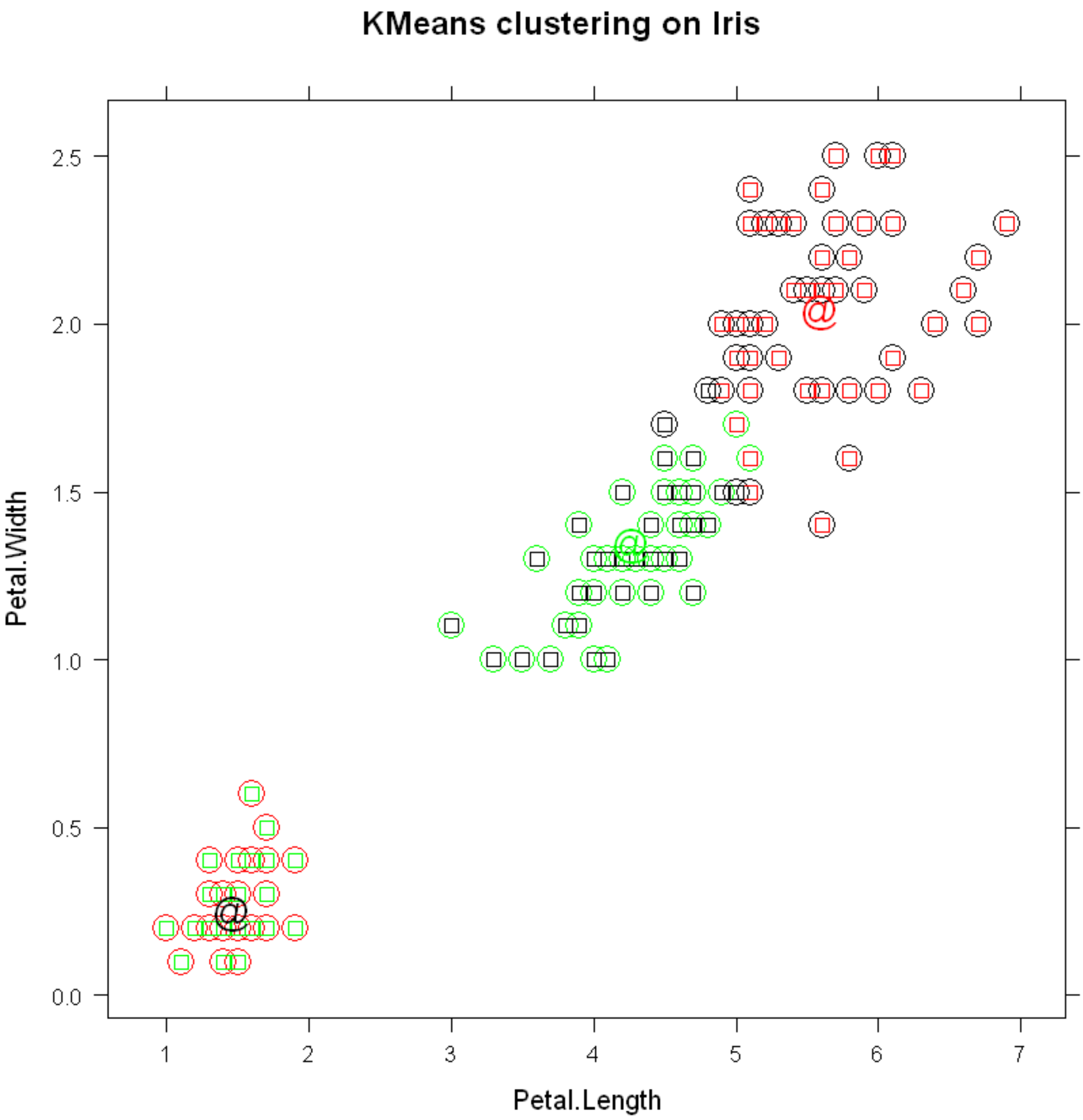


94% accuracy is not bad, lets try with only the petal information.

In [38]:

```
iris_trim<-subset(iris_trim, select = -Sepal.Width)
results <- run(iris_trim, iris)
```

```
K-means clustering with 3 clusters of sizes 48, 50, 52

Cluster means:
  Petal.Length Petal.Width
1     5.595833    2.037500
2     1.462000    0.246000
3     4.269231    1.342308

Clustering vector:
  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  1
9  20
  2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   
 2   2
 21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  3
9  40
  2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   
 2   2
 41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  5
9  60
  2   2   2   2   2   2   2   2   2   2   3   3   3   3   3   3   3   3   
 3   3
 61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  7
9  80
  3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   1   
 3   3
 81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  9
9 100
  3   3   3   1   3   3   3   3   3   3   3   3   3   3   3   3   3   3   
 3   3
101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 11
9 120
  1   1   1   1   1   1   3   1   1   1   1   1   1   1   1   1   1   1   
 1   3
121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 13
9 140
  1   1   1   1   1   1   3   1   1   1   1   1   1   1   1   1   1   1   
 3   1
141 142 143 144 145 146 147 148 149 150
  1   1   1   1   1   1   1   1   1   1

Within cluster sum of squares by cluster:
[1] 16.29167  2.02200 13.05769
 (between_SS / total_SS =  94.3 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withi
nss"
[6] "betweenss"    "size"         "iter"         "ifault"
  Real.Value Assigned.Cluster Frequency
1     setosa                1         0
2 versicolor                1         2
3  virginica                1        46
4     setosa                2        50
5 versicolor                2         0
6  virginica                2         0
7     setosa                3         0
8 versicolor                3        48
9  virginica                3         4
[1] 0.96
```



96% is going to be hard to beat. We will try a final clustering with the width values

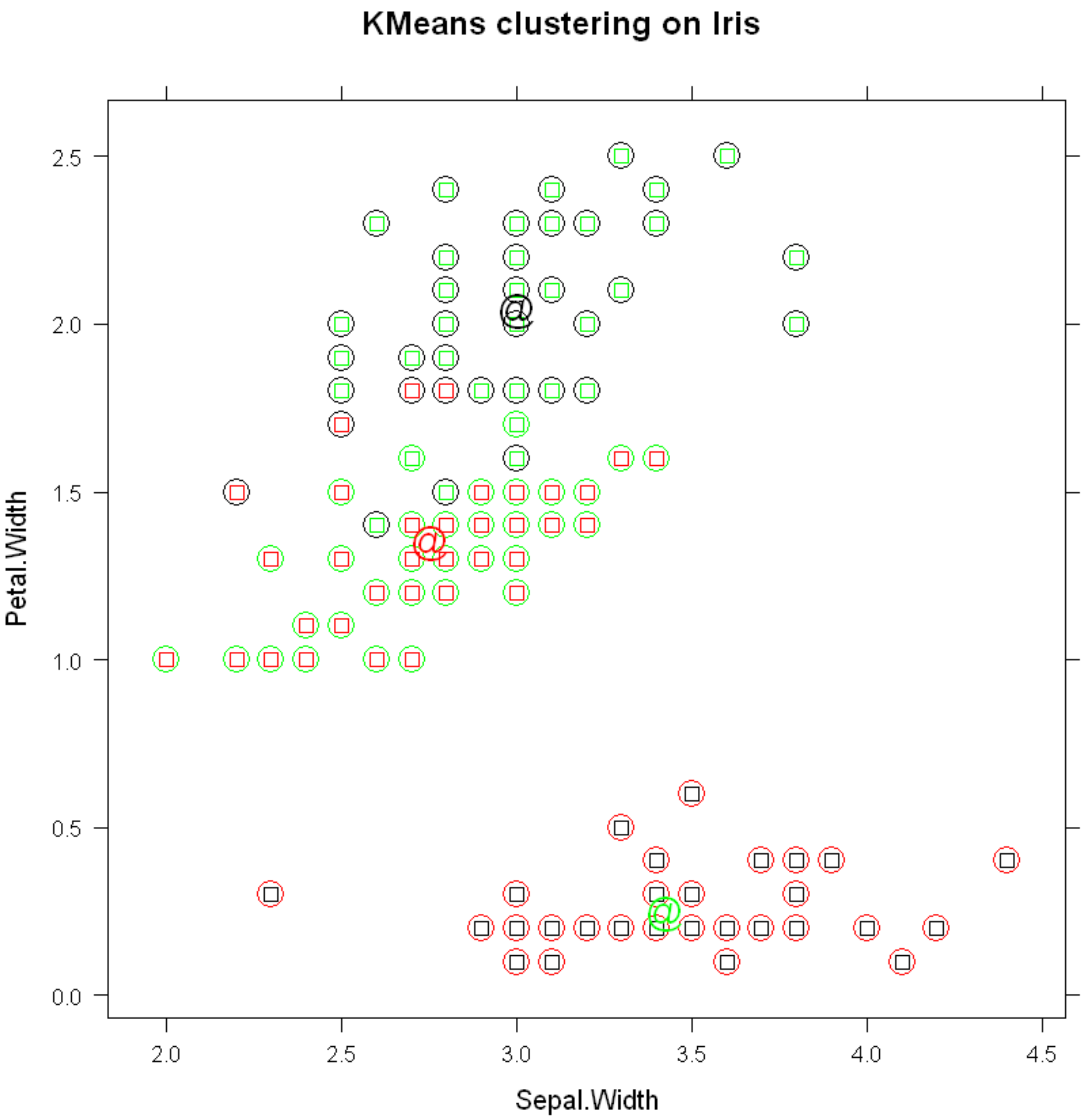
In [39]:

```
# attach lattice for the plotting
library('latticeExtra')
# set seed for repeatability
set.seed(42)

# this functions runs the kmeans on the data, and then outputs the classification accur
acy
# I will call it as I drop elements from the set I am computing the means on
run <- function(data, iris){

  results <- kmeans(data, 3, iter.max = 10, nstart = 1, algorithm = "Hartigan-Wong")
  print(results)

  data$Species <- data.frame(unlist(results[1]))[,1]

  A <- xyplot(Petal.Width ~ Sepal.Width, group = Species, data = data, auto.key =
list(space = "right"),
              par.settings = list(superpose.symbol = list(pch = 0, cex = 1, col = c("re
d", "green", "black"))))
  B <- xyplot(Petal.Width ~ Sepal.Width, group = Species, data = iris, auto.key =
list(space = "rigth"),
              par.settings = list(superpose.symbol = list(pch = 1, cex = 2, col = c("re
d", "green", "black"))), main = "KMeans clustering on Iris")
  C <- xyplot(results$centers[,c("Petal.Width")] ~ results$centers[,c("Sepal.Width")],p
ch = "@", cex = 2,
              col = c("red", "black", "green"), auto.key = list(space = "right"))
  D <- B + as.layer(A + C)


  results <- data.frame(table(iris$Species, results$cluster))
  names(results) <- c("Real.Value", "Assigned.Cluster", "Frequency")
  accuracy <- (max(results$Frequency[1:3])+max(results$Frequency[4:6])+max(results$Freq
uency[7:9]))/150

  print(D)
  print(results)
  print(accuracy)
}
```

In [40]:

```
#iris_trim<-subset(iris_trim, select = -Petal.Length)
iris_trim$Sepal.Width <- iris$Sepal.Width
results <- run(iris_trim, iris)
```

```
K-means clustering with 3 clusters of sizes 53, 47, 50

Cluster means:
  Petal.Length Petal.Width Sepal.Width
1     4.281132    1.350943    2.754717
2     5.610638    2.042553    3.004255
3     1.462000    0.246000    3.428000

Clustering vector:
  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  1
9  20
  3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3
 3   3
 21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  3
9  40
  3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3
 3   3
 41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  5
9  60
  3   3   3   3   3   3   3   3   3   3   1   1   1   1   1   1   1   1
 1   1
 61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  7
9  80
  1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   2
 1   1
 81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  9
9 100
  1   1   1   2   1   1   1   1   1   1   1   1   1   1   1   1   1   1
 1   1
101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 11
9 120
  2   2   2   2   2   2   1   2   2   2   2   2   2   2   2   2   2   2
 2   1
121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 13
9 140
  2   2   2   1   2   2   1   2   2   2   2   2   2   2   2   2   2   2
 1   2
141 142 143 144 145 146 147 148 149 150
  2   2   2   2   2   2   2   2   2   2

Within cluster sum of squares by cluster:
[1] 18.86491 19.93872  9.06280
 (between_SS / total_SS =  91.7 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withi
nss"
[6] "betweenss"    "size"         "iter"         "ifault"
  Real.Value Assigned.Cluster Frequency
1     setosa                1         0
2 versicolor                1        48
3  virginica                1         5
4     setosa                2         0
5 versicolor                2         2
6  virginica                2        45
7     setosa                3        50
8 versicolor                3         0
9  virginica                3         0
[1] 0.9533333
```



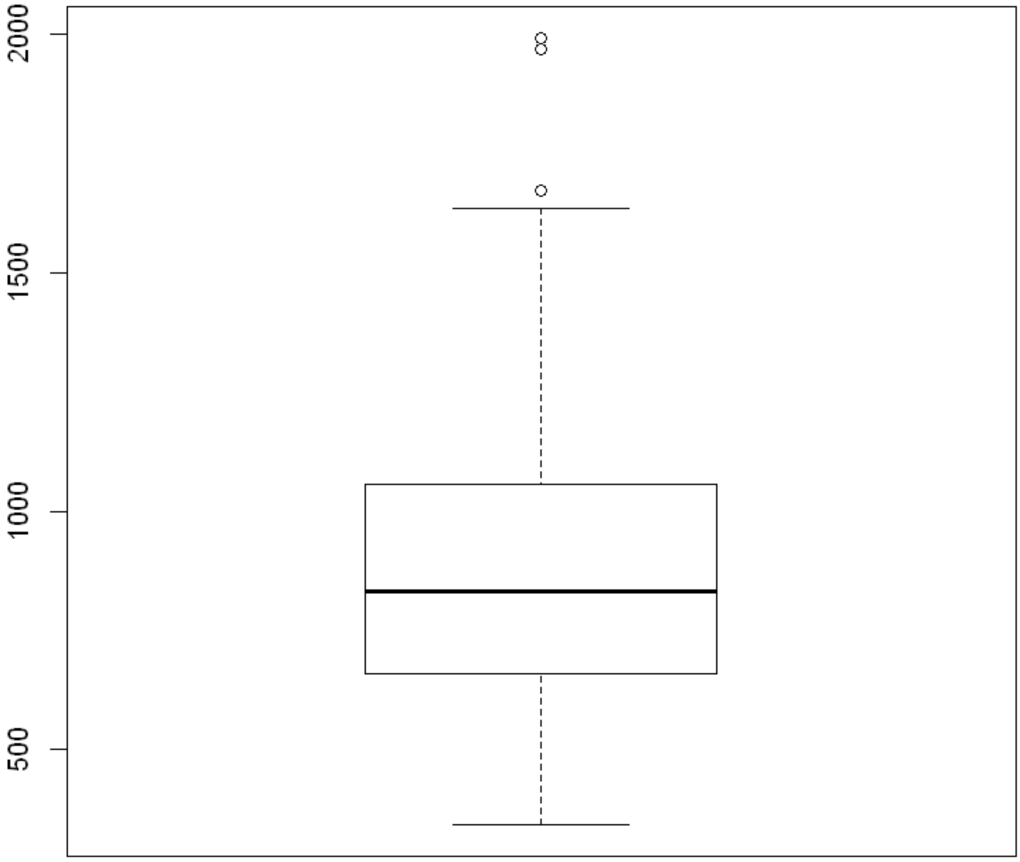
Accuracy is 92.6%, 96% will be as high as we get today.

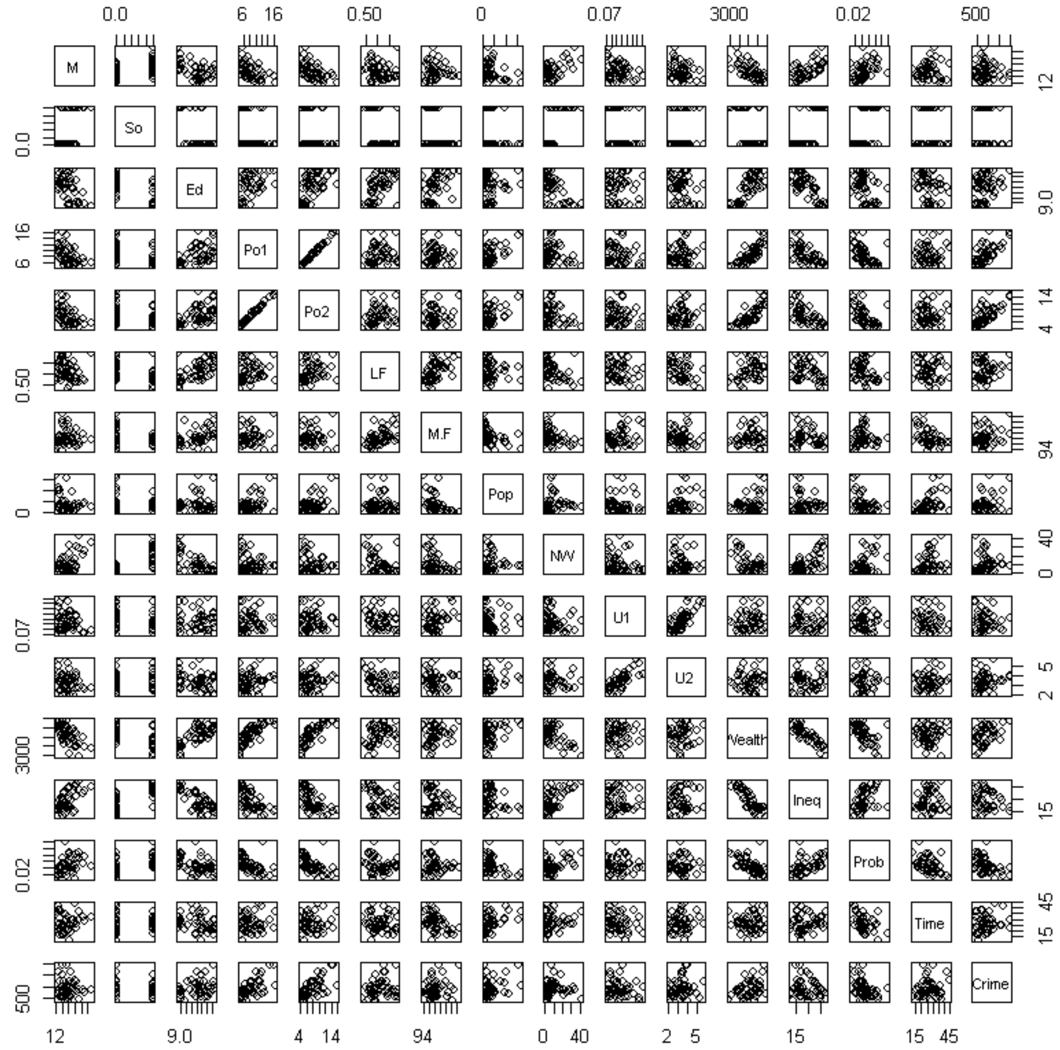
Q3

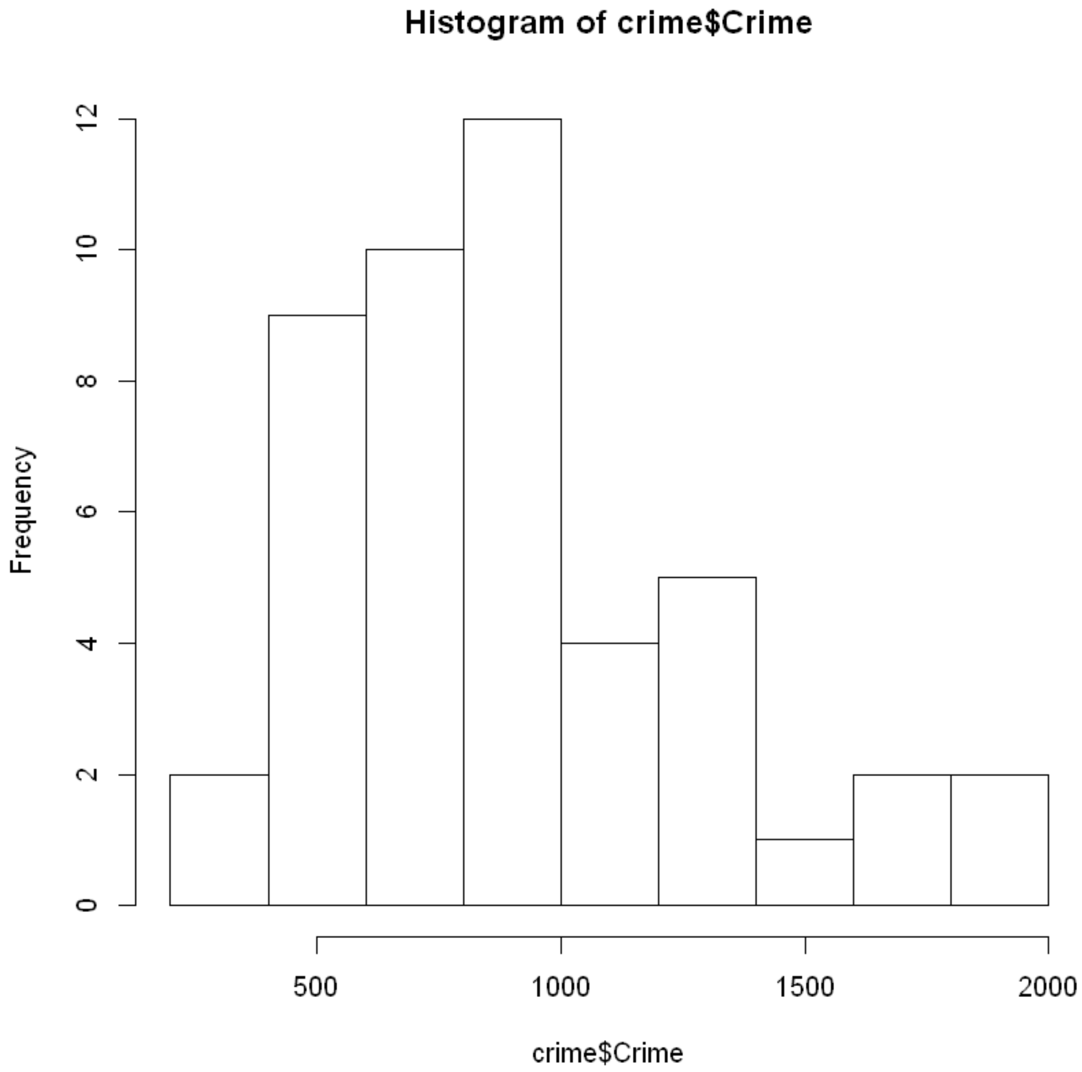
In [21]:

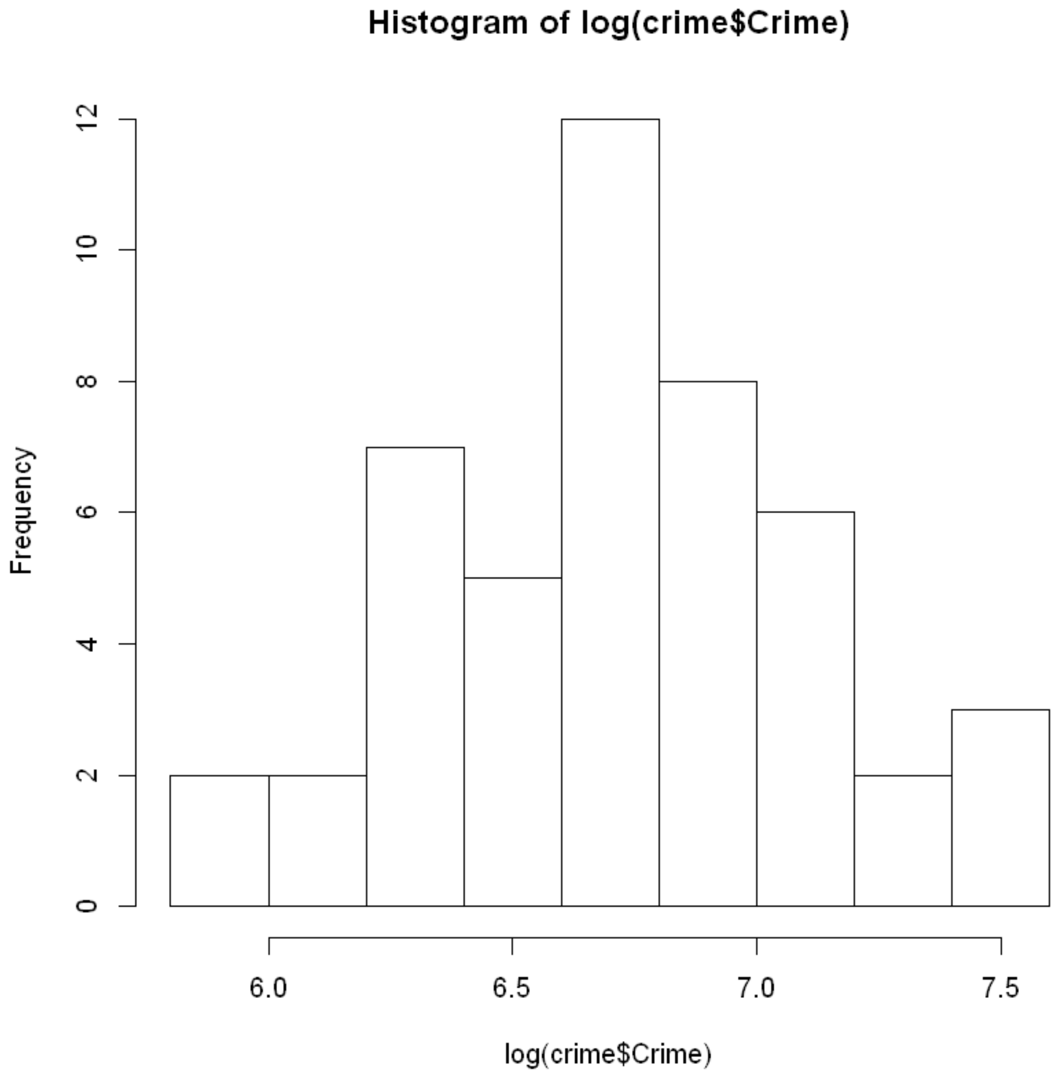
```
# load data
crime <- read.table('uscrime.txt', header=TRUE)
head(crime,5)

#bocplot show 2 datapoints someway higher
boxplot(crime$Crime)
pairs(crime)

# check normal distribution
hist(crime$Crime)
hist(log(crime$Crime))
x <- log(crime$Crime)
boxplot(x)
shapiro.test(crime$Crime)
shapiro.test(log(crime$Crime))
qqnorm(crime$Crime)
qqline(crime$Crime)
qqnorm(x)
qqline(x)
```

M	So	Ed	Po1	Po2	LF	M.F	Pop	NW	U1	U2	Wealth	Ineq	Prob
15.1	1	9.1	5.8	5.6	0.510	95.0	33	30.1	0.108	4.1	3940	26.1	0.084602
14.3	0	11.3	10.3	9.5	0.583	101.2	13	10.2	0.096	3.6	5570	19.4	0.029599
14.2	1	8.9	4.5	4.4	0.533	96.9	18	21.9	0.094	3.3	3180	25.0	0.083401
13.6	0	12.1	14.9	14.1	0.577	99.4	157	8.0	0.102	3.9	6730	16.7	0.015801
14.1	0	12.1	10.9	10.1	0.591	98.5	18	3.0	0.091	2.0	5780	17.4	0.041399

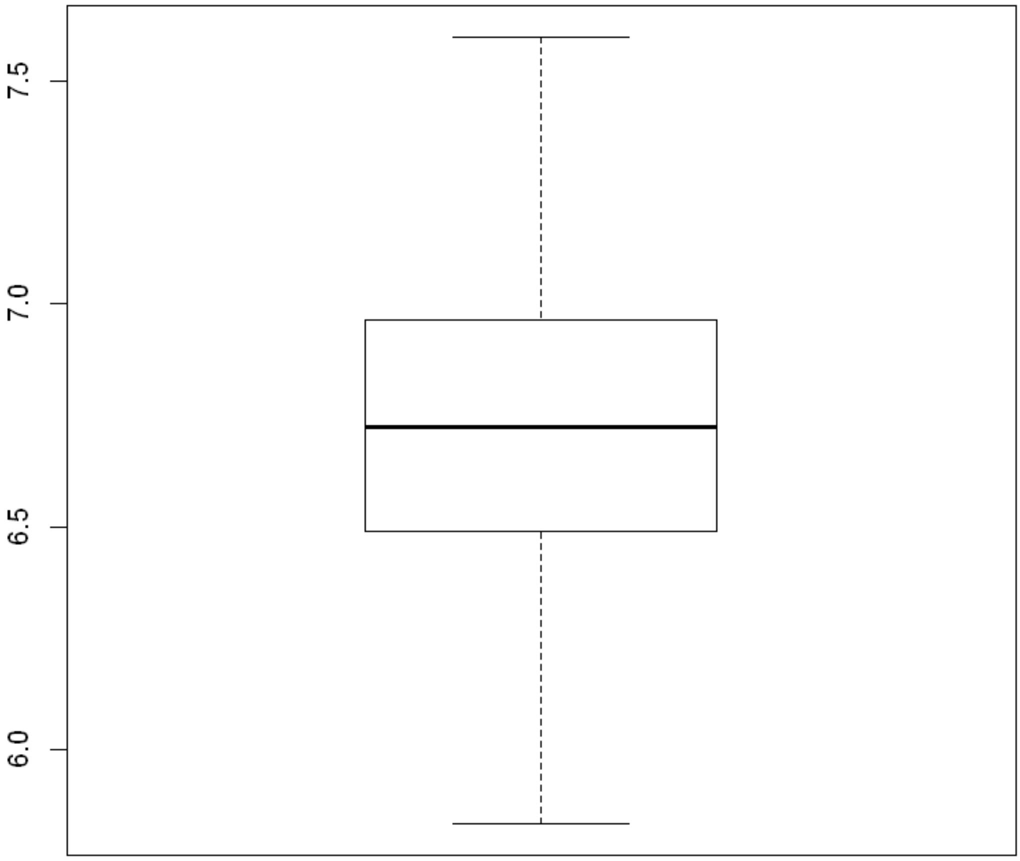



M
So
Ed
Po1
Po2
LF
M.F
Pop
NW
U1
U2
Wealth
Ineq
Prob
Time
Crime

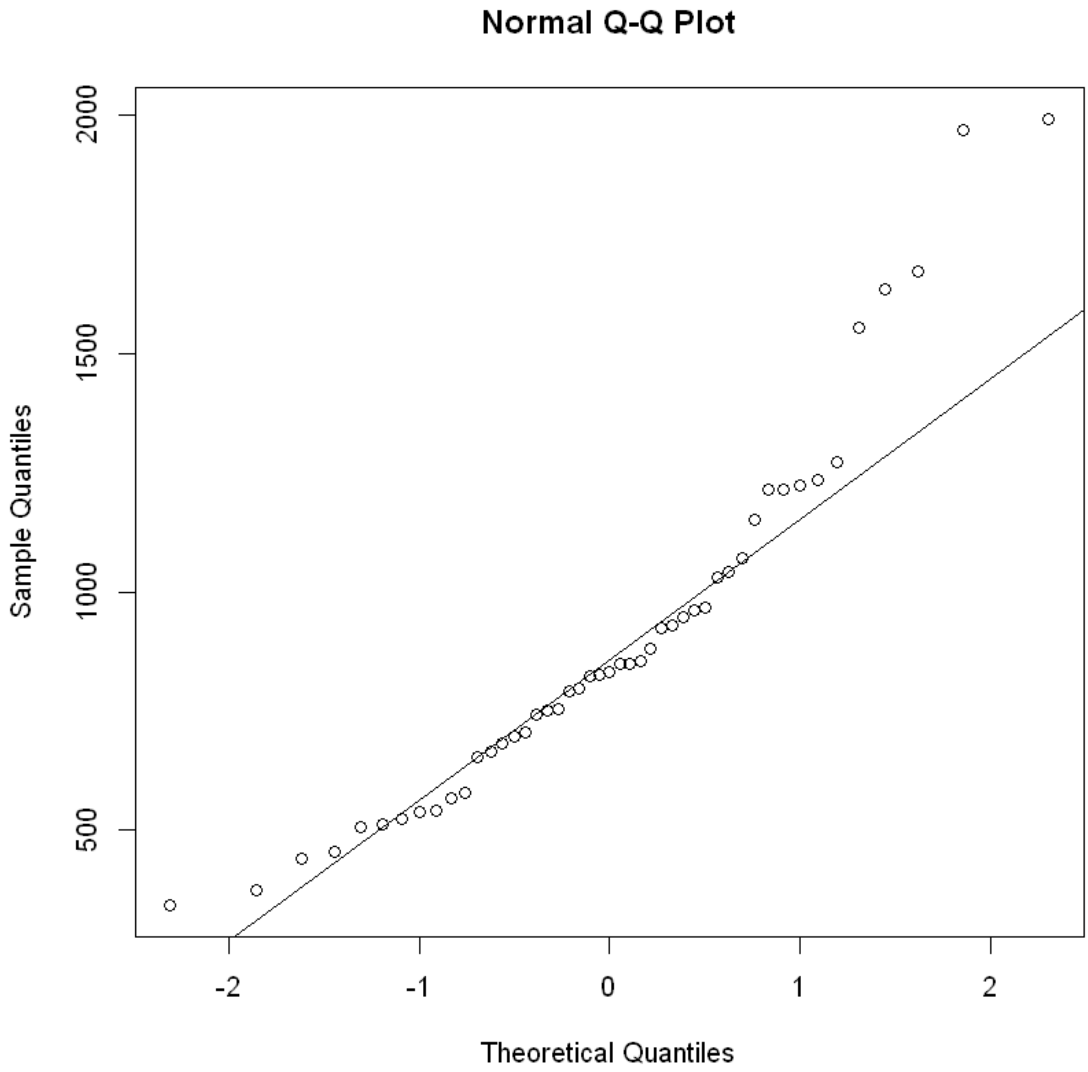

Histogram of crime$Crime
Frequency
0
2
4
6
8
10
12
500
1000
1500
2000
crime$Crime

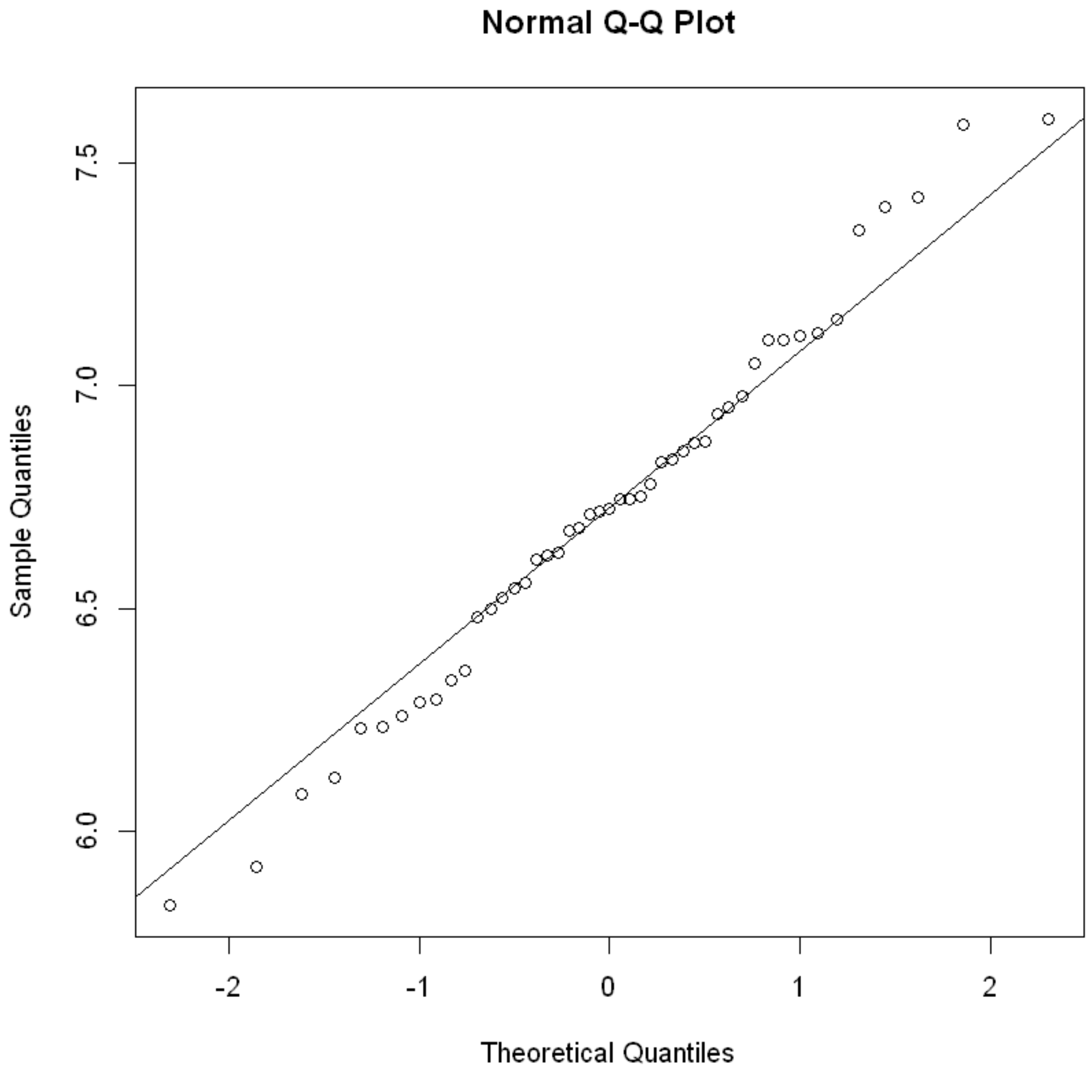



```
        Shapiro-Wilk normality test

data:  crime$Crime
W = 0.91273, p-value = 0.001882

        Shapiro-Wilk normality test

data:  log(crime$Crime)
W = 0.98709, p-value = 0.8778
```



7.5
7.0
6.5
6.0


Normal Q-Q Plot
Sample Quantiles
Theoretical Quantiles
2000
1500
1000
500
-2
-1
0
1
2



So by having a look at the data we suspect that the distribution is not normal so the grubbs test may not be appropraite, the log transformation makes the data look a lot more normal. Regardless lets do our grubbs test.

In [23]:

```
library(outliers)

grubbs.test(crime$Crime)
grubbs.test(x, opposite=TRUE)
```

```
        Grubbs test for one outlier

data:  crime$Crime
G = 2.81290, U = 0.82426, p-value = 0.07887
alternative hypothesis: highest value 1993 is an outlier
```

```
        Grubbs test for one outlier

data:  x
G = 2.12250, U = 0.89994, p-value = 0.712
alternative hypothesis: highest value 7.59739632021279 is an outlier
```

So the grubbs test suggests that depending on our tolerance the top score may be an outlier. In the log example we cannot reject the null, it does not appear to be an outlier.

In [24]:

```
grubbs.test(crime$Crime, opposite=TRUE)
grubbs.test(x)
```

```
	Grubbs test for one outlier

data:  crime$Crime
G = 1.45590, U = 0.95292, p-value = 1
alternative hypothesis: lowest value 342 is an outlier

	Grubbs test for one outlier

data:  x
G = 2.16540, U = 0.89585, p-value = 0.6329
alternative hypothesis: lowest value 5.8348107370626 is an outlier
```

Testing the other side of the data grubbs test suggest we can say with any signifigance that there is an outlier.

In [25]:

```
grubbs.test(crime$Crime, type=11)
```

```
	Grubbs test for two opposite outliers

data:  crime$Crime
G = 4.26880, U = 0.78103, p-value = 1
alternative hypothesis: 342 and 1993 are outliers
```

Again grubbs test suggest we cannot say that there is an ourlier on either side.

I wanted to test the alternate hypothesis there were outliers on both sides of the dist, but type 20 only seems to run on very small datasets. As such I tested each point as per below.

In [26]:

```
scores_test <- scores(crime$Crime, prob=0.95)
crime$Scores <- scores_test
crime[(crime$Scores==TRUE),16:17]
crime$Crime[order(crime$Crime, decreasing = TRUE)[1:5]]
```

	Crime	Scores
2	1635	TRUE
4	1969	TRUE
8	1555	TRUE
11	1674	TRUE
26	1993	TRUE

1993 1969 1674 1635 1555

In [27]:

```
log_scores_test <- scores(x, prob=0.95)
log_scores_test[(log_scores_test==TRUE)]
crime$log_score <- log_scores_test
crime[(crime$log_score==TRUE),16:18]
```

TRUE TRUE TRUE TRUE TRUE

	Crime	Scores	log_score
4	1969	TRUE	TRUE
11	1674	TRUE	TRUE
26	1993	TRUE	TRUE
27	342	FALSE	TRUE
31	373	FALSE	TRUE

The outlier tests had mixed results, I certainly would not be removing the top for entries as outliers if I was going to do an analysis. I would conclude that this data is not normally distributed and if I was a building a model I wold check to see the validity of a log transformation of the crime level.

Q4

For my work we often assess the ongoing performance of biometric engines. An overall change in biometric match scores could suggest a change in conditions or use that would be worth monitoring and control.

Q5

I also completed a spreadsheet, it is attached.

For part 1: In the spreadsheet I set the threshold to -2.5 as we want to detect the change as soon as we the temprature consistently declining.

For part 2: St reach a high of 7 and then declined. We would expect some further growth in St if temprature was raising so setting a threshold above this would be wise. We cannot detect a change in average tempratures at this time.

I also completed this exercise in R as below:

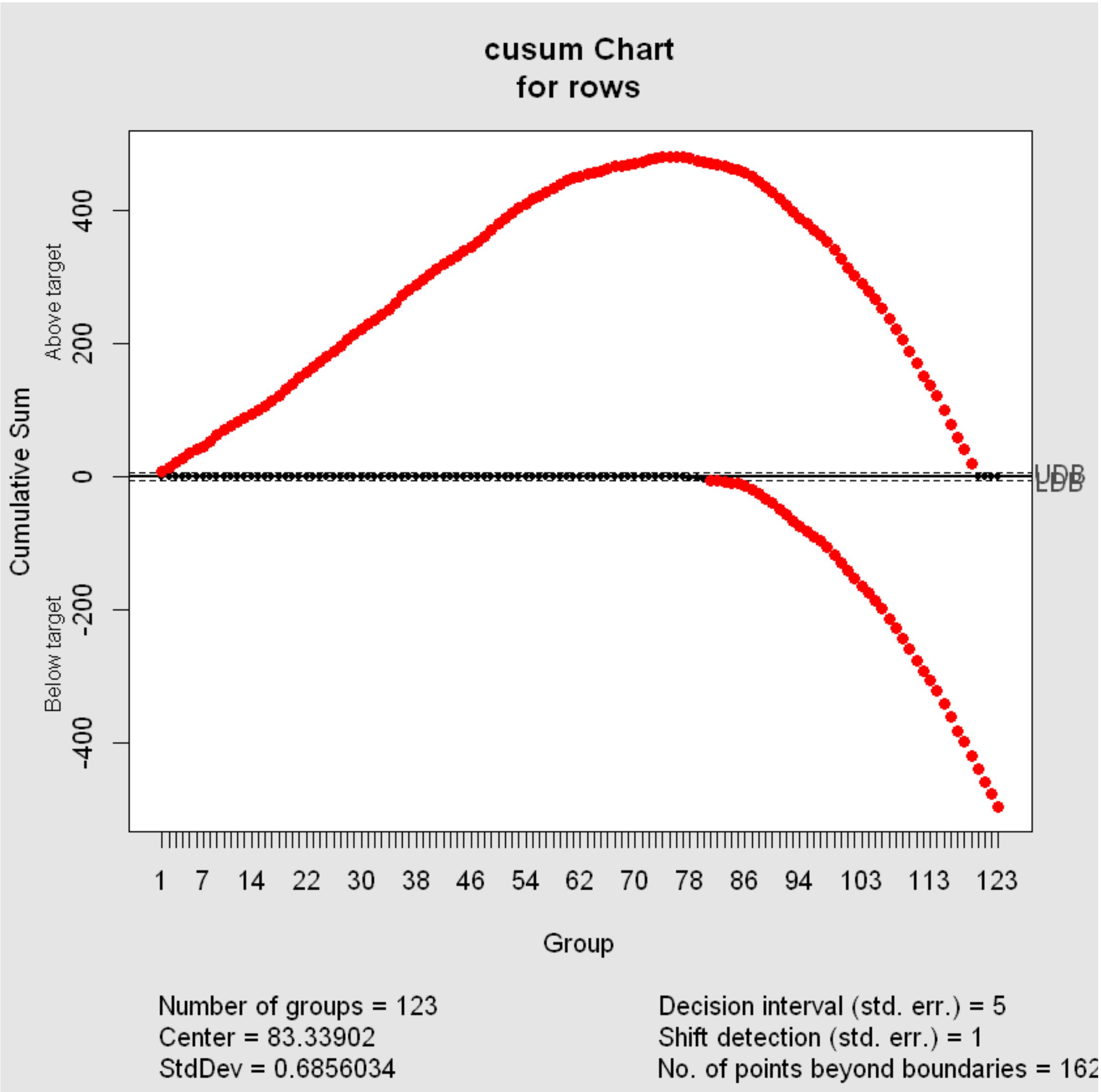
In [31]:

```
temps <- read.table('temps.txt', header = TRUE)
head(temps,5)
rows <- rowMeans(temps[2:21], na.rm=TRUE, dims = 1)
cols <- colMeans(temps[2:21], na.rm = TRUE, dims = 1)

library('qcc')
change_days <- cusum(rows)
change_days$violations$lower
temps$DAY[81]
```

DAY	X1996	X1997	X1998	X1999	X2000	X2001	X2002	X2003	X2004	...	X2006	X20
1-Jul	98	86	91	84	89	84	90	73	82	...	93	95
2-Jul	97	90	88	82	91	87	90	81	81	...	93	85
3-Jul	97	93	91	87	93	87	87	87	86	...	93	82
4-Jul	90	91	91	88	95	84	89	86	88	...	91	86
5-Jul	89	84	91	90	96	86	93	80	90	...	90	88

81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97
98 99 100 101 102 103 104 105 106 107 108 109 110 111
112 113 114 115 116 117 118 119 120 121 122 123

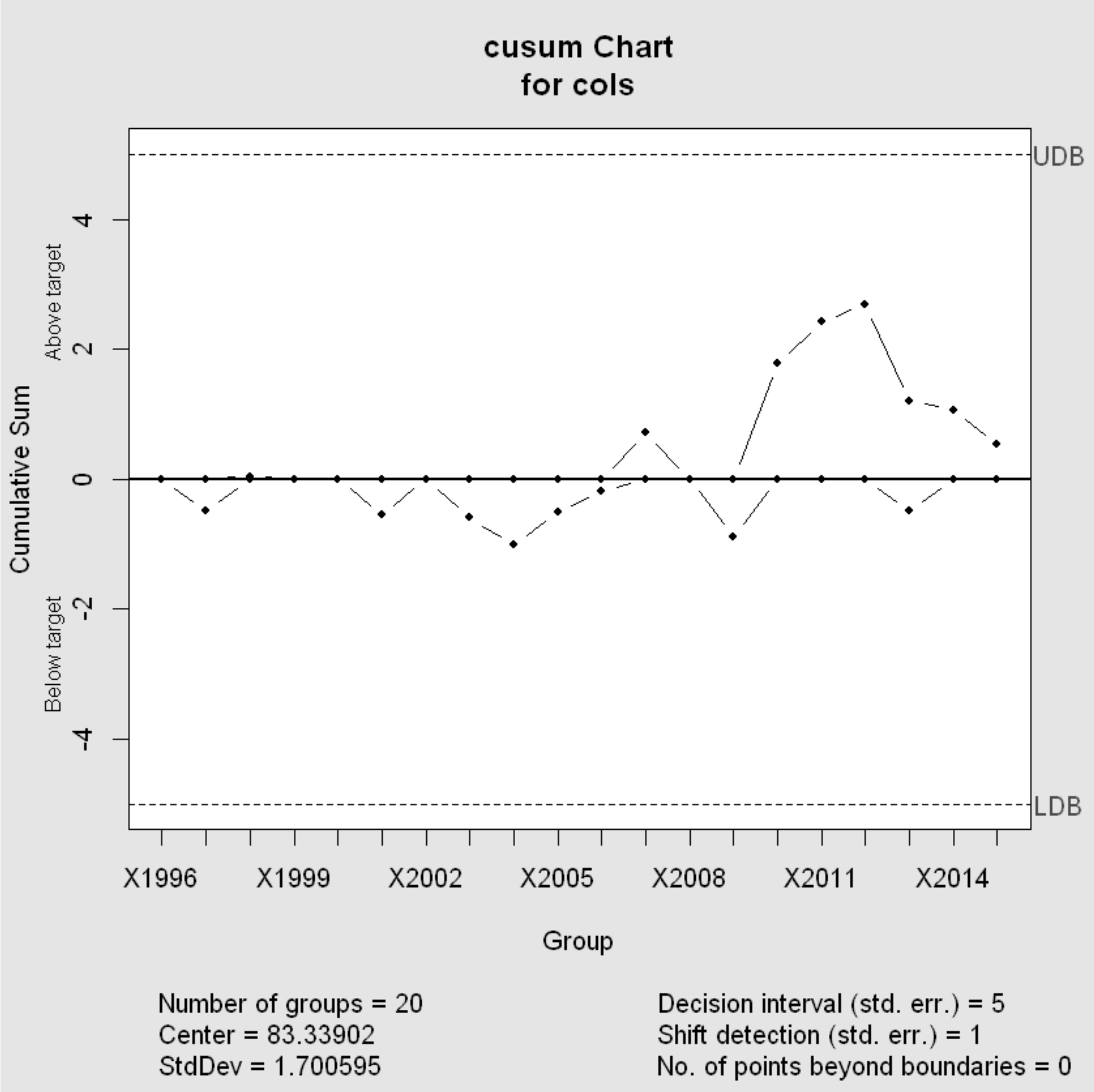
19-Sep



This suggest that 19 Sep is the detected change with std error set to 1.

In [30]:

```
change_years <- cusum(cols)
```



This suggest not change has been detected with std error 1.