Homework 2

Joe Diaz

9/3/2019

Question 3.1

Using the same data set (credit_card_data.txt or credit_card_data-headers.txt) as in Question 2.2, use the ksvm or kknn function to find a good classifier:

(a) using cross-validation (do this for the k-nearest-neighbors model; SVM is optional); and

https://www.rdocumentation.org/packages/kknn/versions/1.3.1/topics/train.kknn

Using the train.kknn function in the kknn library, we can perform leave-one-out crossvalidation easily.

```
library(kknn)
set.seed(0)
data <- read.table("credit_card_data.txt", stringsAsFactors = FALSE, header = FALSE)

kmax <- 30
accuracy<- c()
for (k in 1:kmax) {

    model <- cv.kknn(V11~.,data,kcv=10, k=k, scale=TRUE)
    predicted <- round(model[[1]][,2])
    accuracy[k] <- sum(predicted == data$V11)/nrow(data)
}

cat('Acheived best accuracy of: '
    , round(max(accuracy),2)*100,'%'
    , ' with k=',which.max(accuracy),sep='')
```

```
## Acheived best accuracy of: 86% with k=23
```

(b) splitting the data into training, validation, and test data sets (pick either KNN or SVM; the other is optional).

The code below shows how the train, test and validation split is made.

The data split is as follows:

Data	Percentage
Training	70%
Test	15%
Validation	15%

```
idx_train <- sample(nrow(data), size = floor(nrow(data) * .7))
train <- data[idx_train,]


rest <- data[-idx_train,]
```

```
idx_val <-sample(nrow(rest), size = floor(nrow(rest)*.5))


validation = rest[idx_val,]
test = rest[-idx_val, ]
```

Now that we have properly segmented the data, we can now train the data using the training dataset. We will then measure model performance with the test dataset. I find it better to store predictions and inspect manually rather than overwriting one variable.

```
accuracy <- c()
for (k in 1:20) {

     model <- kknn(V11~.,train,test,k=k,scale=TRUE)

     pred <- round(fitted(model))

     accuracy[k] = sum(pred == test$V11) / nrow(test)
}



cat('Acheived best accuracy of: '
    , round(max(accuracy),2)*100,'%'
    , ' with k=',which.max(accuracy),sep='')
```

```
## Acheived best accuracy of: 88% with k=8
```

Now it's time to test it with the validation set using the best model.

```
best_model <- kknn(V11~.,train,validation,
                k=which.max(accuracy),
                scale=TRUE)

     pred <- round(fitted(best_model))

cat("Validation performance: ",round(sum(pred == validation$V11) / nrow(validation),2)*100,"%",sep="")
```

```
## Validation performance: 86%
```

Question 4.1

Describe a situation or problem from your job, everyday life, current events, etc., for which a clustering model would be appropriate. List some (up to 5) predictors that you might use.

One possible use case for clustering is customer segmentation. For context, I work in an e-commerce company. Grouping like customers might provide us with some insight regarding our customer base.

Some features I would consider are:

1. Order history (Which products did they buy? How often etc.)
2. LTV
3. Income
4. Family size
5. Education

Question 4.2

The iris data set iris.txt contains 150 data points, each with four predictor variables and one categorical response. The predictors are the width and length of the sepal and petal of flowers and the response is the type of flower. The data is available from the R library datasets and can be accessed with iris once the library is loaded. It is also available at the UCI Machine Learning Repository (https://archive.ics.uci.edu/ml/datasets/Iris). The response values are only given to see how well a specific method performed and should not be used to build the model. Use the R function kmeans to cluster the points as well as possible. Report the best combination of predictors, your suggested value of k, and how well your best clustering predicts flower type.

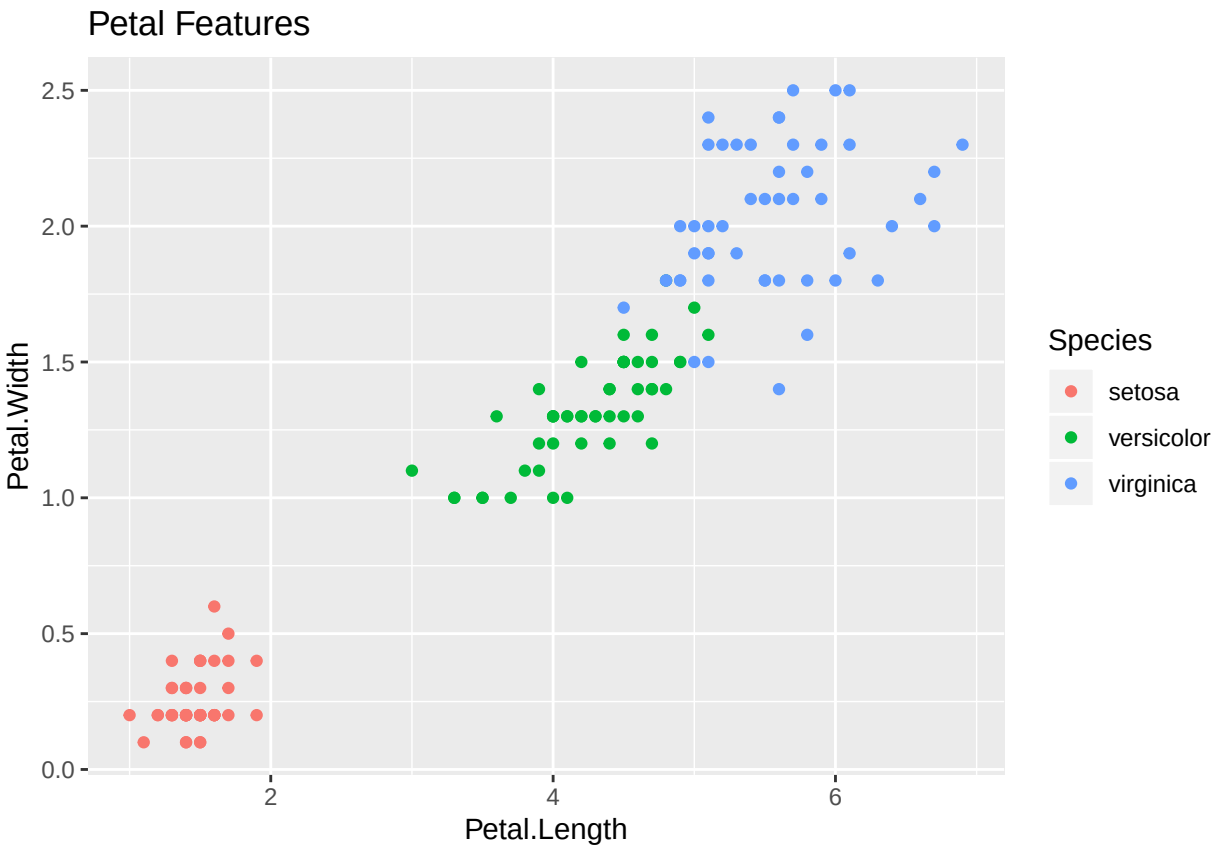
```
iris <- read.table("iris.txt", header = TRUE)
head(iris)
```

```
##   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
```

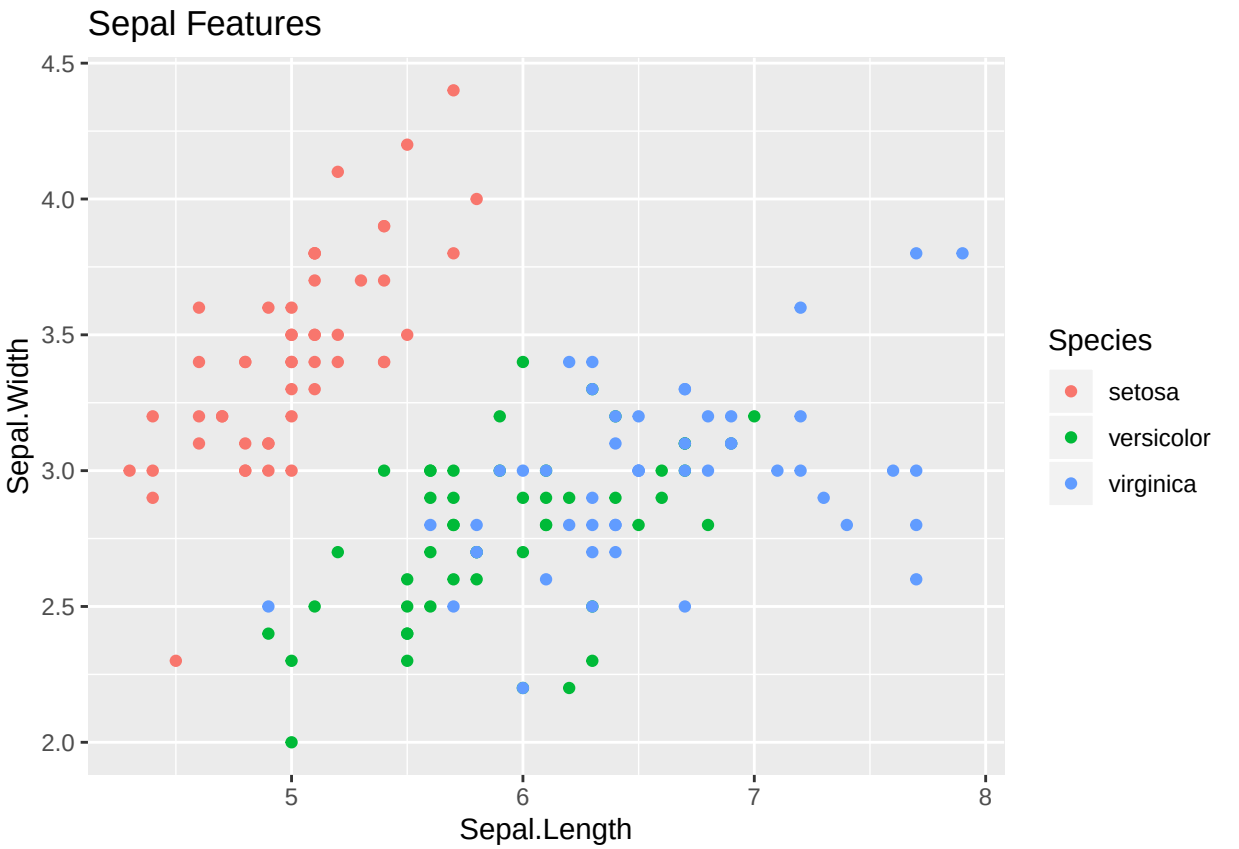
The data shows 4 features. 2 being Sepal features and the other 2 being Petal features.

Let's inspect these by doing a scatter plot. I find ggplot to be a good plotting package.

```
library(ggplot2)
ggplot(iris, aes(Petal.Length, Petal.Width, color = Species)) + geom_point() + ggtitle('Petal Features')
```




```
ggplot(iris, aes(Sepal.Length, Sepal.Width, color = Species)) + geom_point() + ggtitle('Sepal Features')
```

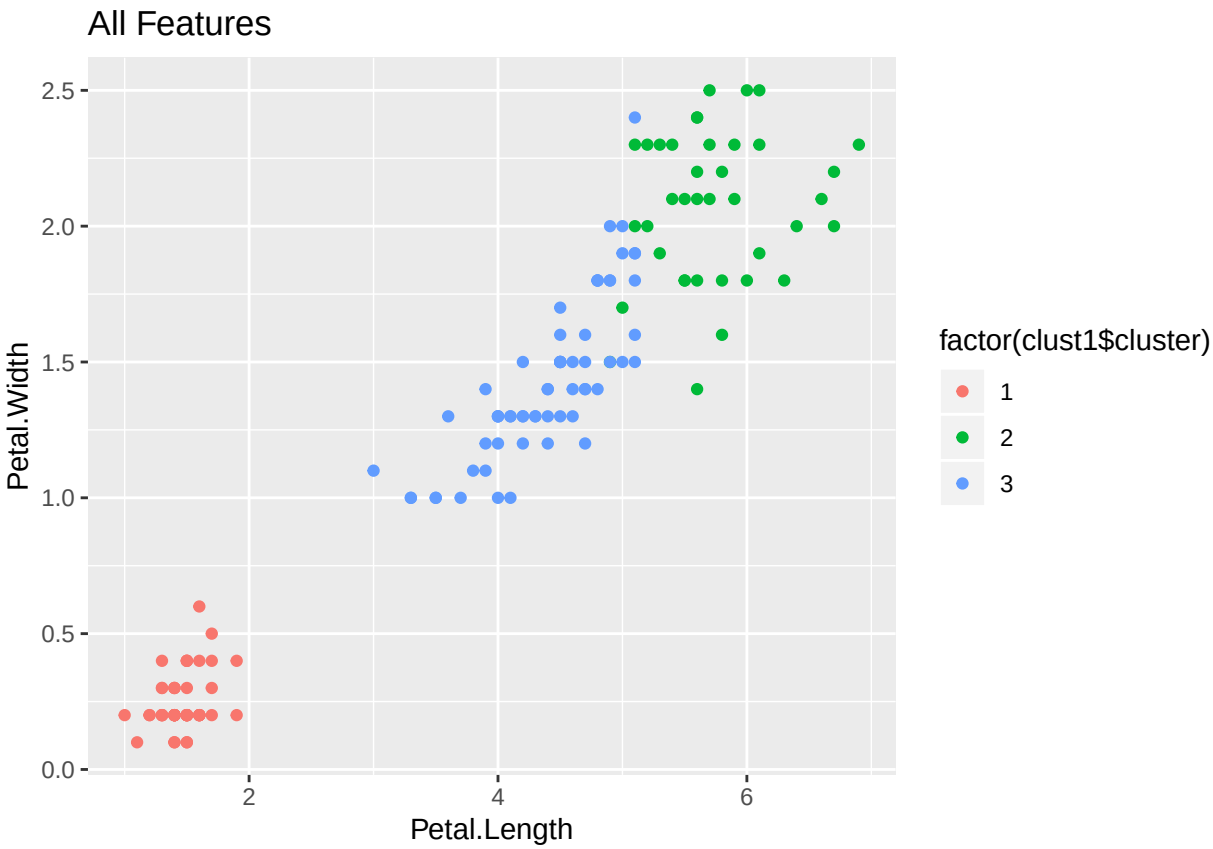


Looks like Petal metrics separates the data easily while the Sepal metrics not so much. We will try a few combinations to see which is the best.

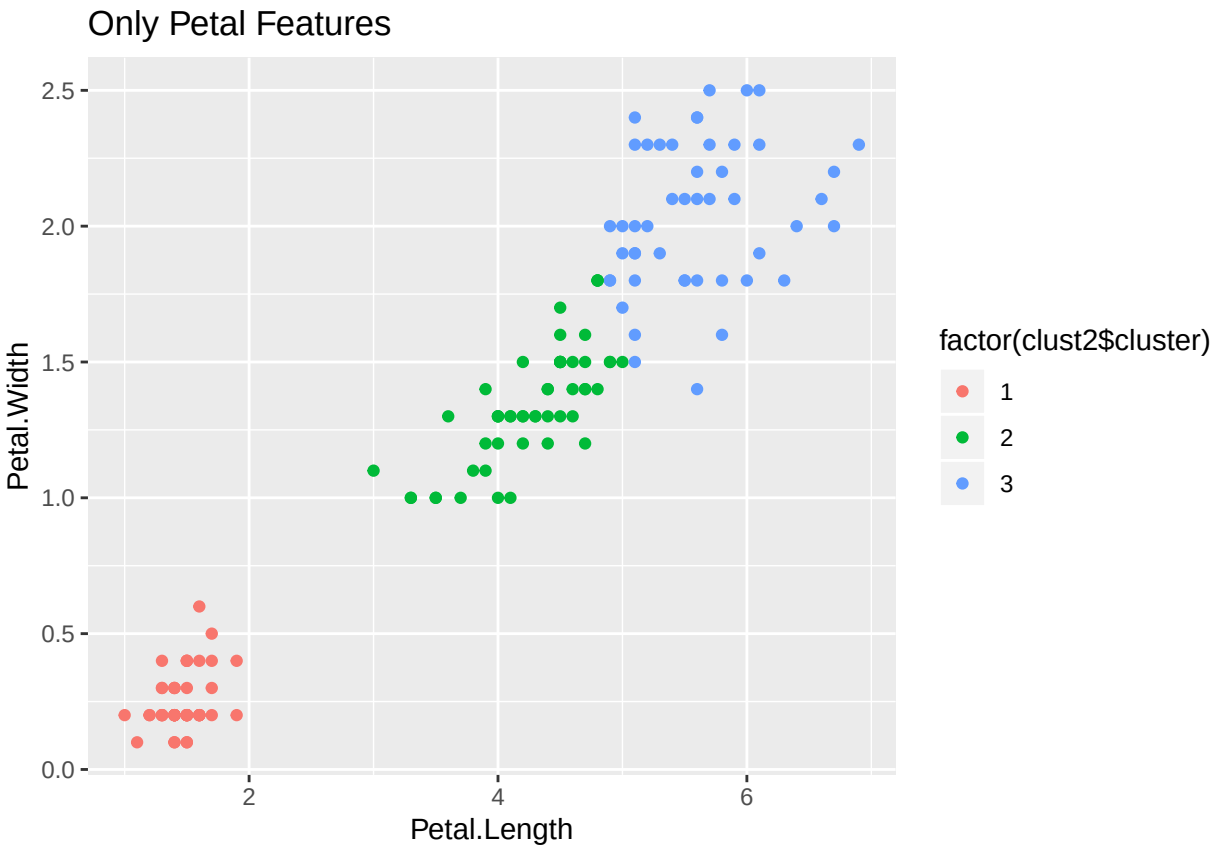
```
clust1 <- kmeans(iris[,-5], 3, nstart = 1000)
clust2 <- kmeans(iris[,3:4], 3, nstart = 1000)
clust3 <- kmeans(iris[,1:2], 3, nstart = 1000)
```

Now that we have a few different kmeans models, let's inspect how they do. Luckilym we are given their true groupings. We can use this to measure model performance.

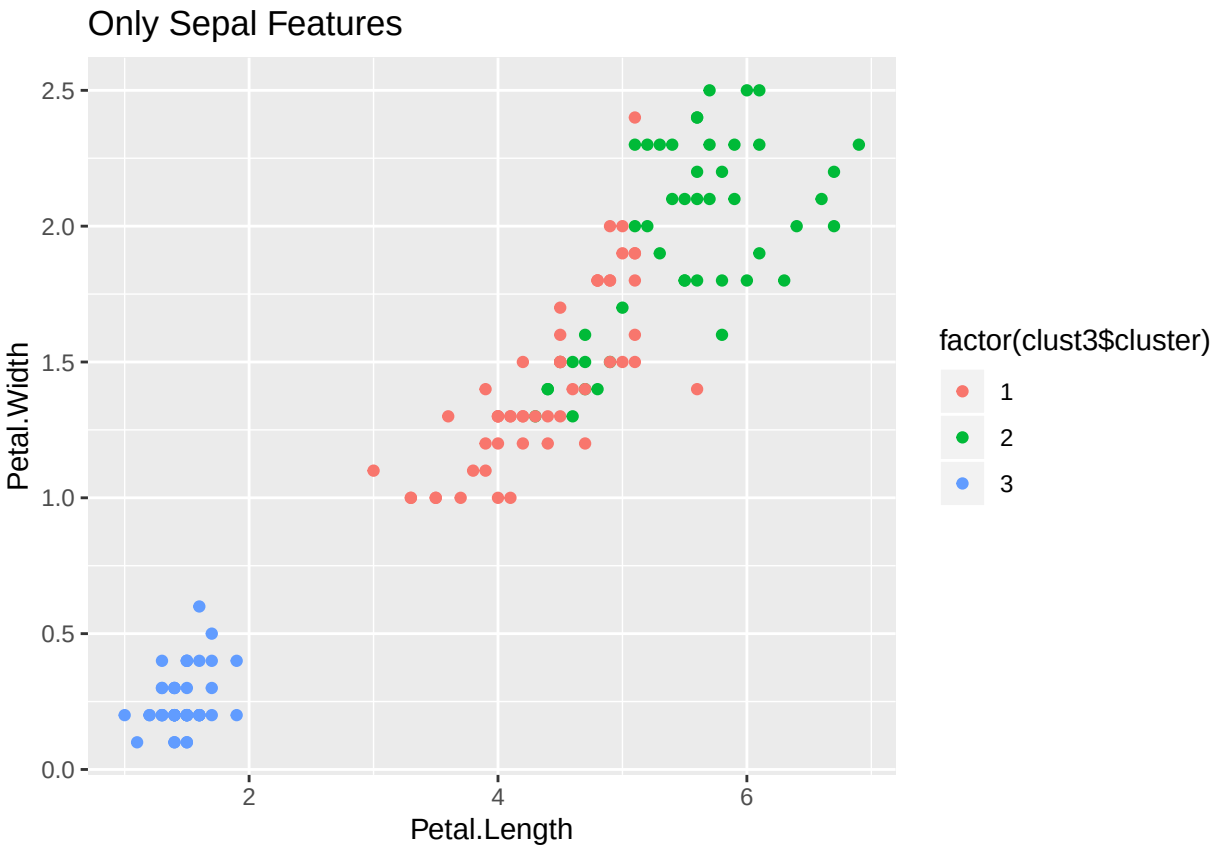
```
ggplot(iris, aes(Petal.Length, Petal.Width, color = factor(clust1$cluster))) + geom_point() + ggtitle("
```




```
ggplot(iris, aes(Petal.Length, Petal.Width, color = factor(clust2$cluster))) + geom_point() + ggtitle("
```




```
ggplot(iris, aes(Petal.Length, Petal.Width, color = factor(clust3$cluster))) + geom_point() + ggtitle("
```



All Features

```
table(clust1$cluster, iris$Species)
```

```
##
##     setosa versicolor virginica
##   1     50          0         0
##   2      0          2        36
##   3      0         48        14
```

Only Petal Features

```
table(clust2$cluster, iris$Species)
```

```
##
##     setosa versicolor virginica
##   1     50          0         0
##   2      0         48         4
##   3      0          2        46
```

Only Sepal Features

```
table(clust3$cluster, iris$Species)
```

```
##
##     setosa versicolor virginica
##   1      0         38        15
##   2      0         12        35
##   3     50          0         0
```

Just eyeballing the graphs, it seems to be better to just cluster off of Petal metrics. This is confirmed when looking at the cluster matrix. `Only Petal Features` beats out `All Features` mainly on grouping out virginica.

As a result, my solution will be a kmeans clustering model using only petal features.

Solution

```
clust2
```

```
## K-means clustering with 3 clusters of sizes 50, 52, 48
##
## Cluster means:
##   Petal.Length Petal.Width
## 1     1.462000    0.246000
## 2     4.269231    1.342308
## 3     5.595833    2.037500
##
## Clustering vector:
##   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
##  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
##  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   2   2   2   2
##  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
##  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90
##   2   2   2   2   2   3   2   2   2   2   2   3   2   2   2   2   2   2
##  91  92  93  94  95  96  97  98  99 100 101 102 103 104 105 106 107 108
##   2   2   2   2   2   2   2   2   2   2   3   3   3   3   3   3   2   3
## 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126
##   3   3   3   3   3   3   3   3   3   3   3   2   3   3   3   3   3   3
## 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144
##   2   3   3   3   3   3   3   3   3   3   3   3   2   3   3   3   3   3
## 145 146 147 148 149 150
##   3   3   3   3   3   3
##
## Within cluster sum of squares by cluster:
## [1]  2.02200 13.05769 16.29167
##  (between_SS / total_SS =  94.3 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"
## [5] "tot.withinss" "betweenss"    "size"         "iter"
## [9] "ifault"
```

```
ggplot(iris, aes(Petal.Length, Petal.Width, color = factor(clust2$cluster))) + geom_point() + ggtitle("
```