Assignment 2

Question 3.1 (a)

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

Answer 3.1 (a)

- **Doing cross validation manually**

```
#Importing libraries
library(kknn)
library(ggplot2)

#Reading the data
credit_data<-read.table('credit_card_data-headers.txt', sep = "", header = TRUE)

#Viewing the data
head(credit_data)
```

```
##   A1    A2    A3   A8 A9 A10 A11 A12 A14 A15 R1
## 1  1 30.83 0.000 1.25  1   0   1   1 202   0  1
## 2  0 58.67 4.460 3.04  1   0   6   1  43 560  1
## 3  0 24.50 0.500 1.50  1   1   0   1 280 824  1
## 4  1 27.83 1.540 3.75  1   0   5   0 100   3  1
## 5  1 20.17 5.625 1.71  1   1   0   1 120   0  1
## 6  1 32.08 4.000 2.50  1   1   0   0 360   0  1
```

```
#Summarizing the data
str(credit_data)
```

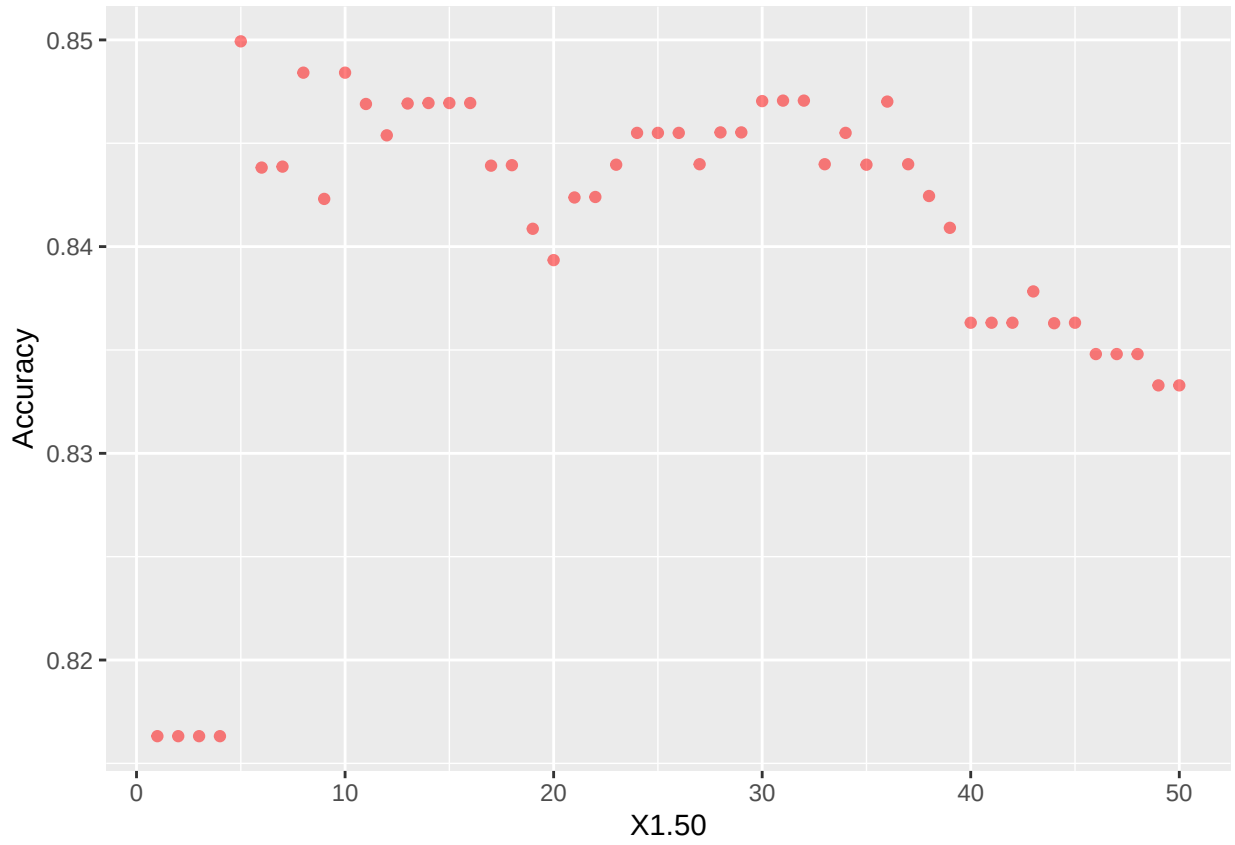
```
## 'data.frame':    654 obs. of  11 variables:
##  $ A1 : int  1 0 0 1 1 1 1 0 1 1 ...
##  $ A2 : num  30.8 58.7 24.5 27.8 20.2 ...
##  $ A3 : num  0 4.46 0.5 1.54 5.62 ...
##  $ A8 : num  1.25 3.04 1.5 3.75 1.71 ...
##  $ A9 : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ A10: int  0 0 1 0 1 1 1 1 1 1 ...
##  $ A11: int  1 6 0 5 0 0 0 0 0 0 ...
##  $ A12: int  1 1 1 0 1 0 0 1 1 0 ...
##  $ A14: int  202 43 280 100 120 360 164 80 180 52 ...
##  $ A15: int  0 560 824 3 0 0 31285 1349 314 1442 ...
##  $ R1 : int  1 1 1 1 1 1 1 1 1 1 ...
```

```r
#Setting the seed for reproducibility
set.seed(42)

#Shuffling the data randomly
credit_data<-credit_data[sample(nrow(credit_data)),]

#Creating 10 groups for cross validation. One can also user input to decide
#on the number of cross validation groups. Will be more automated.
group <- cut(seq(1,nrow(credit_data)),breaks=10,labels=FALSE)
```

- **Building the Model**

```r
pre<-list() #Creating an empty list to store predictions
accu<-list() #Creating an empty list to store accuracy
for (j in 1:50){   #Loop to test different k between 1-50
  acc1=0           #Setting Accuracy to 0 for new value of k
  for (i in 1:10){ #Loop to test k-fold cross validation
    ind<-which(group==i, arr.ind = FALSE) #This create a set using the group
                                          #defined above.
    model_knn = kknn(R1~.,
                     train=credit_data[-ind,],
                     test=credit_data[ind,],
                     k = j,
                     scale = TRUE)
    pred_knn = fitted(model_knn)
    pre[[i]]<-ifelse(pred_knn>0.5,1,0)
    acc1=acc1+sum(pre[[i]] == credit_data[ind,11])/nrow(credit_data[ind,])
  }
  accu[[j]]=acc1/10 # Average accuracy for the jth value representing k
}
```

```r
#Plotting k vs accuracy
dat<-data.frame(k=list(1:50), Accuracy=unlist(accu))
ggplot(data=dat,aes(x=X1.50,y=Accuracy))+geom_point(alpha=0.5,color='red')
```




```
#Sorting the dataframe to get best k

dat_sort<-dat[order(-dat$Accuracy),]
best_k<-dat_sort[1,1]

message('Best k: ',best_k)
```

```
## Best k: 5
```

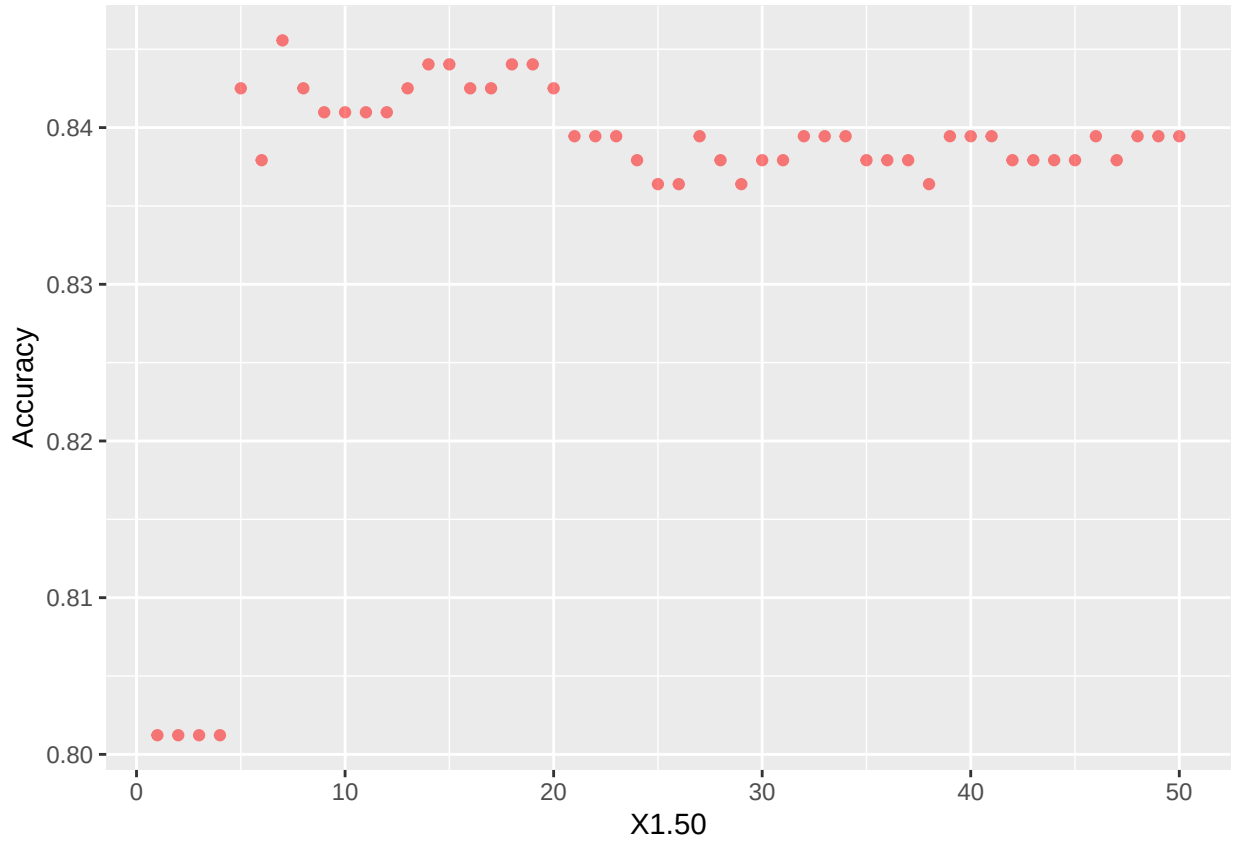
```
message('Accuracy of the model using the best k: ', dat_sort[1,2])
```

```
## Accuracy of the model using the best k: 0.84993006993007
```

- **Now trying the same thing using cv.kknn**

```
accu_new<-list() #Creating an empty list to store accuracy

#Model fit
for (m in 1:50){ #Testing different k's
  set.seed(42)
  kNN_fit<-cv.kknn(R1~.,data=credit_data,
                   scale=TRUE, k = m, kcv = 10)
  pred<-ifelse(kNN_fit[[1]][,2]>0.5,1,0)
  accu_new[[m]]=sum(pred==credit_data[,11])/nrow(credit_data)
```

```
}

#Plotting k vs accuracy
dat_new<-data.frame(k=list(1:50), Accuracy=unlist(accu_new))
ggplot(data=dat_new,aes(x=X1.50,y=Accuracy))+geom_point(alpha=0.5,color='red')
```




```
#Sorting the dataframe to get best k

dat_sort_new<-dat_new[order(-dat_new$Accuracy),]
best_k_new<-dat_sort_new[1,1]

message('Best k: ',best_k_new)
```

```
## Best k: 7
```

```
message('Accuracy of the model using the best k: ', dat_sort_new[1,2])
```

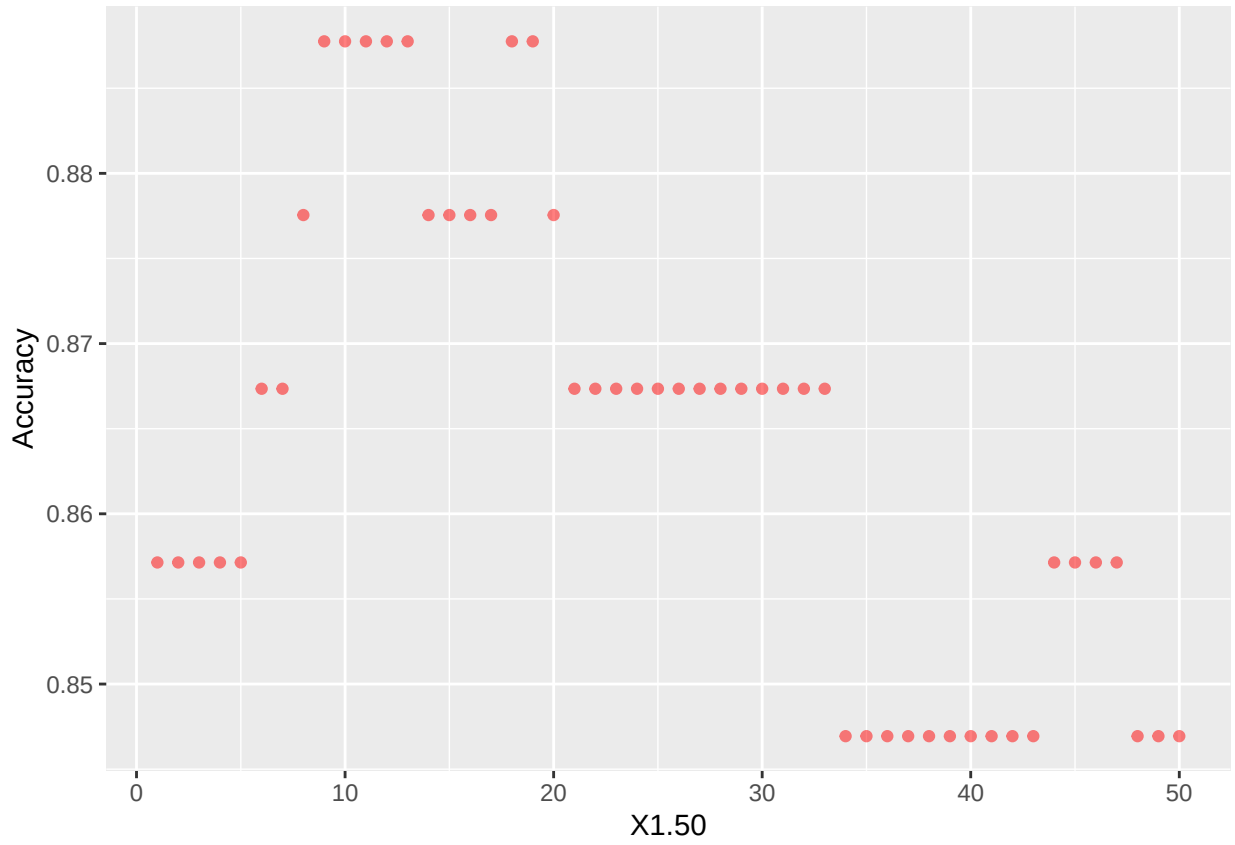
```
## Accuracy of the model using the best k: 0.845565749235474
```

Answer 3.1 (b)

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

```r
#We will use the caret library to do the splitting
library(caret)
library(kknn)
library(ggplot2)
```

```r
#Getting the data
credit_data<-read.table('credit_card_data-headers.txt', sep = "", header = TRUE)

#The data will be divided with 70% Train, 15% Validation and 15% Test data
set.seed(42)

#Using cresteDataPartion
train_index<-createDataPartition(y=credit_data$R1, p=0.7,
                                 times = 1, list = FALSE)
#Training data
train_data<-credit_data[train_index,]

#Validation and testing data, this data is further divided into two sets
rest_data<-credit_data[-train_index,]
rest_index<-createDataPartition(y=rest_data$R1, p=0.5, times = 1,list = FALSE)

#Validation data
val_data<-rest_data[rest_index,]

#Testing data
test_data<-rest_data[-rest_index,]

#Using kkNN
acc<-list() #Creating an empty list to store accuracy

#Testing the kknn method for different k values (1-50)

for (i in 1:50){
  model_kknn<-kknn(R1~., train = train_data, test = val_data,
                   k=i, scale = TRUE)
  pred<-fitted(model_kknn)
  pre<-ifelse(pred>0.5,1,0)
  acc[[i]]=sum(pre==val_data[,11])/nrow(val_data)
}

#Choosing the best value of k from the validation accuracy. To this we put both
#accuracy and k in a data frame and plot it

score<-data.frame(k=list(1:50), Accuracy=unlist(acc))
ggplot(data = score, aes(x=X1.50, y=Accuracy))+geom_point(alpha=0.5,color='red')
```




```
#Finding the best model from the previous data frame
score_sort<-score[order(-score$Accuracy),]

message('Best k: ',score_sort[1,1])
```

```
## Best k: 9
```

```
message('Accuracy of the model using the best k: ', score_sort[1,2])
```

```
## Accuracy of the model using the best k: 0.887755102040816
```

```
#Now testing this model  on the test data
model_test<-kknn(R1~.,train = train_data, test = test_data, k=score_sort[1,1],
                 scale = TRUE)
test_predict<-ifelse(fitted(model_test)>0.5,1,0)
test_accuracy<-sum(test_predict==test_data[,11])/nrow(test_data)

message('Accuracy of the model on the test using the best k:', test_accuracy)
```

```
## Accuracy of the model on the test using the best k:0.836734693877551
```

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.

Answer:

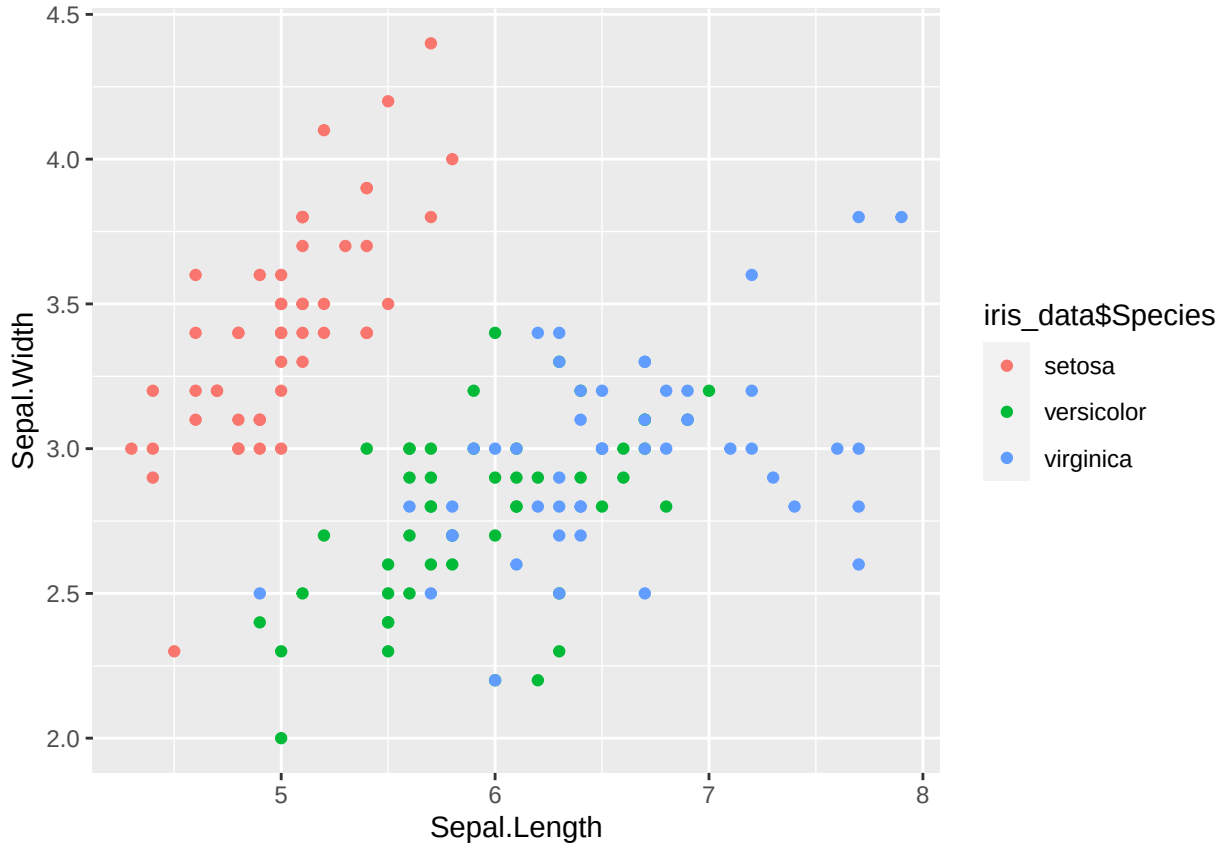
In my job as an engineer, I investigate well performance and decide which well may need any intervention or predict which well might go down. We interpret the signature of the well to predict what could be the potential problem. We use the following predictors for this investigation.

- Tubing/Casing size
- Reservoir Pressure
- Temperature
- Casing/Tubing pressure
- Flow rate

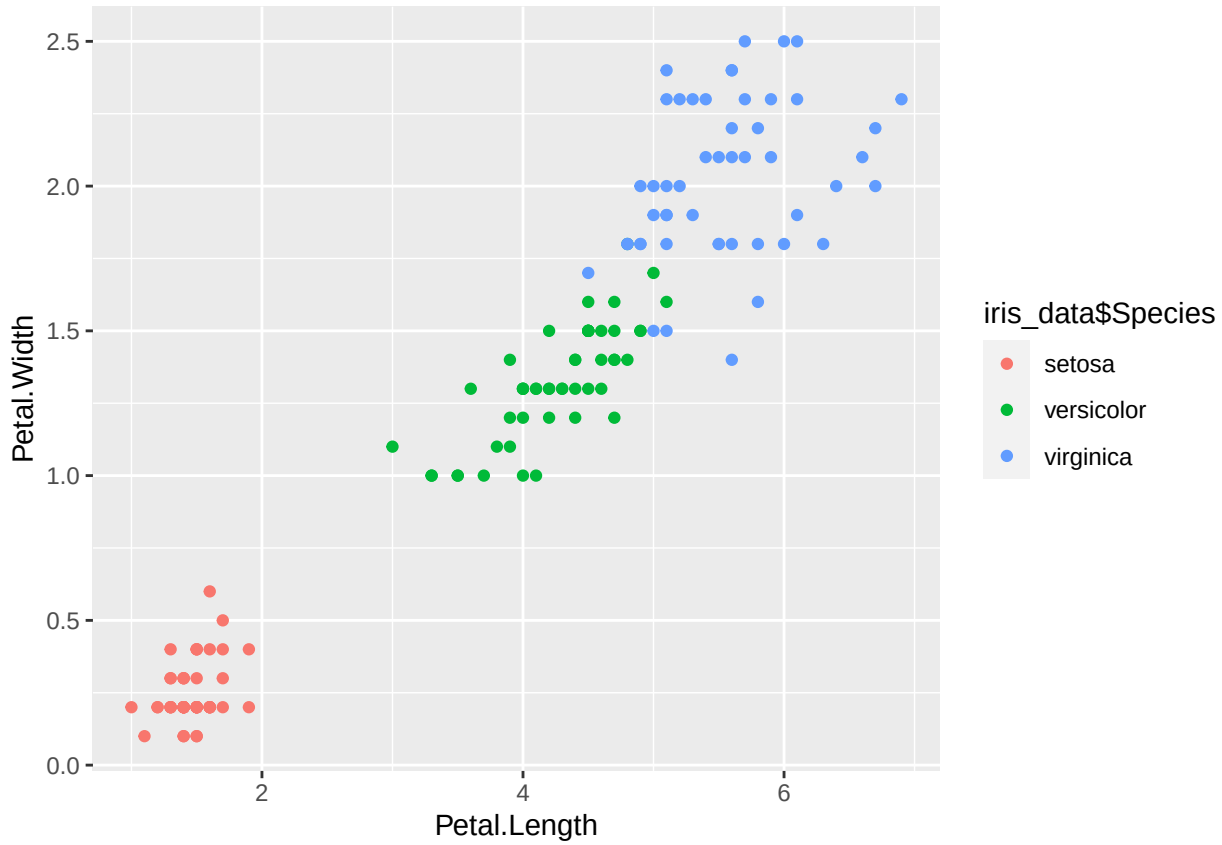
I can see myself developing a clustering model where I can feed these parameters and see what kind of problem cluster the well end up getting into and thus proactively take necessary actions.

Question 4.2

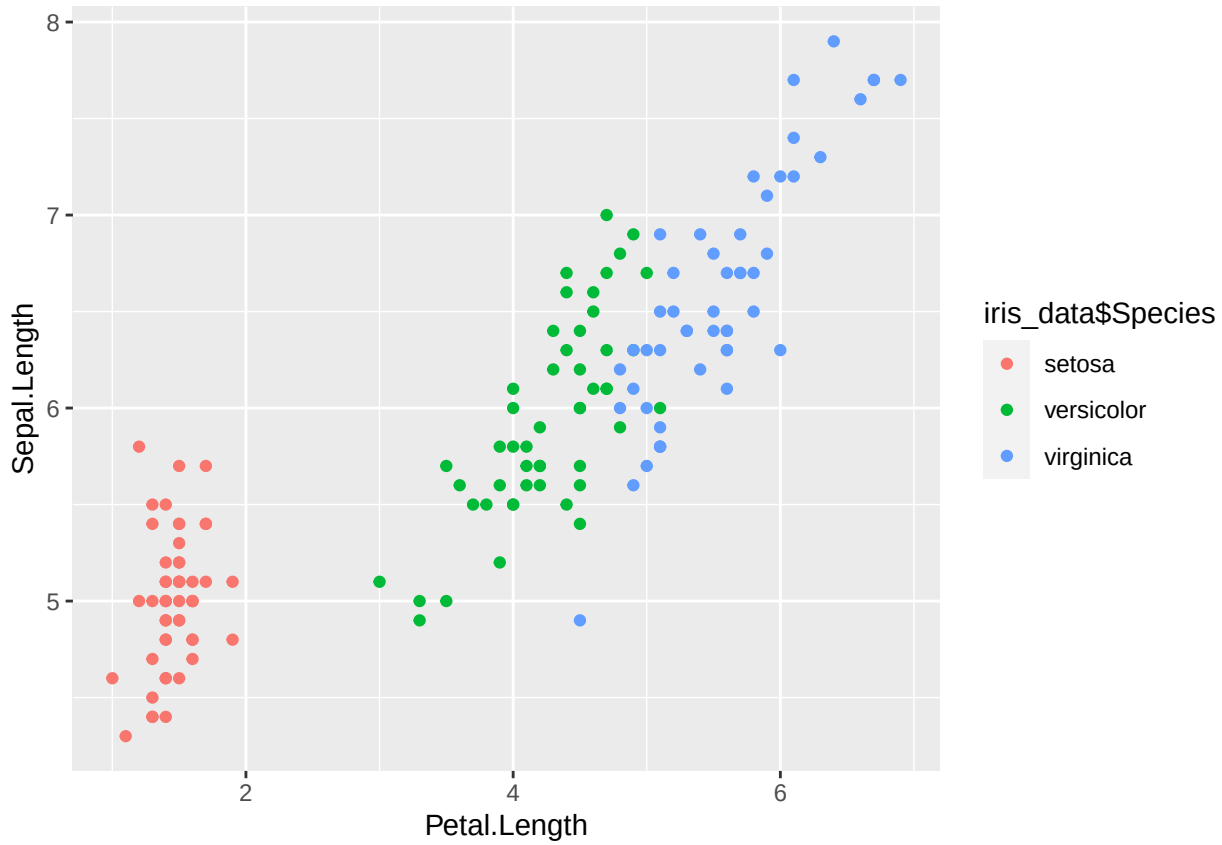
```
#Getting the data
iris_data<-read.table('iris.txt', sep = "", header = TRUE)

#Visualizing the data using various combination to get some intuition of the
#data distribution
ggplot(data = iris_data, aes(Sepal.Length,
                             Sepal.Width,
                             color=iris_data$Species))+
  geom_point(alpha=1) #The plot shows two clusters with sertosa having its own
```



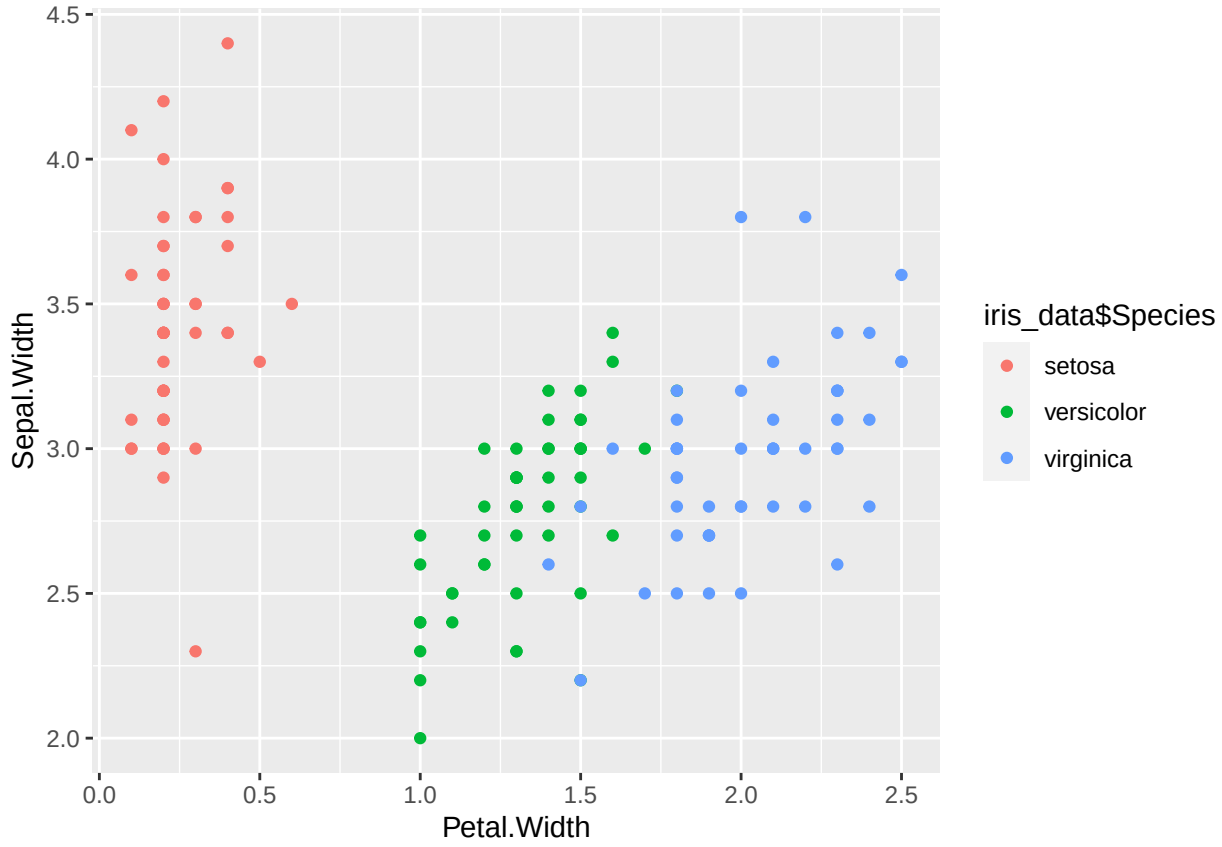

```
                    #clear cluster.

ggplot(data = iris_data, aes(Petal.Length,
                             Petal.Width,
                             color=iris_data$Species))+
  geom_point(alpha=1) #Shows three good clusters
```



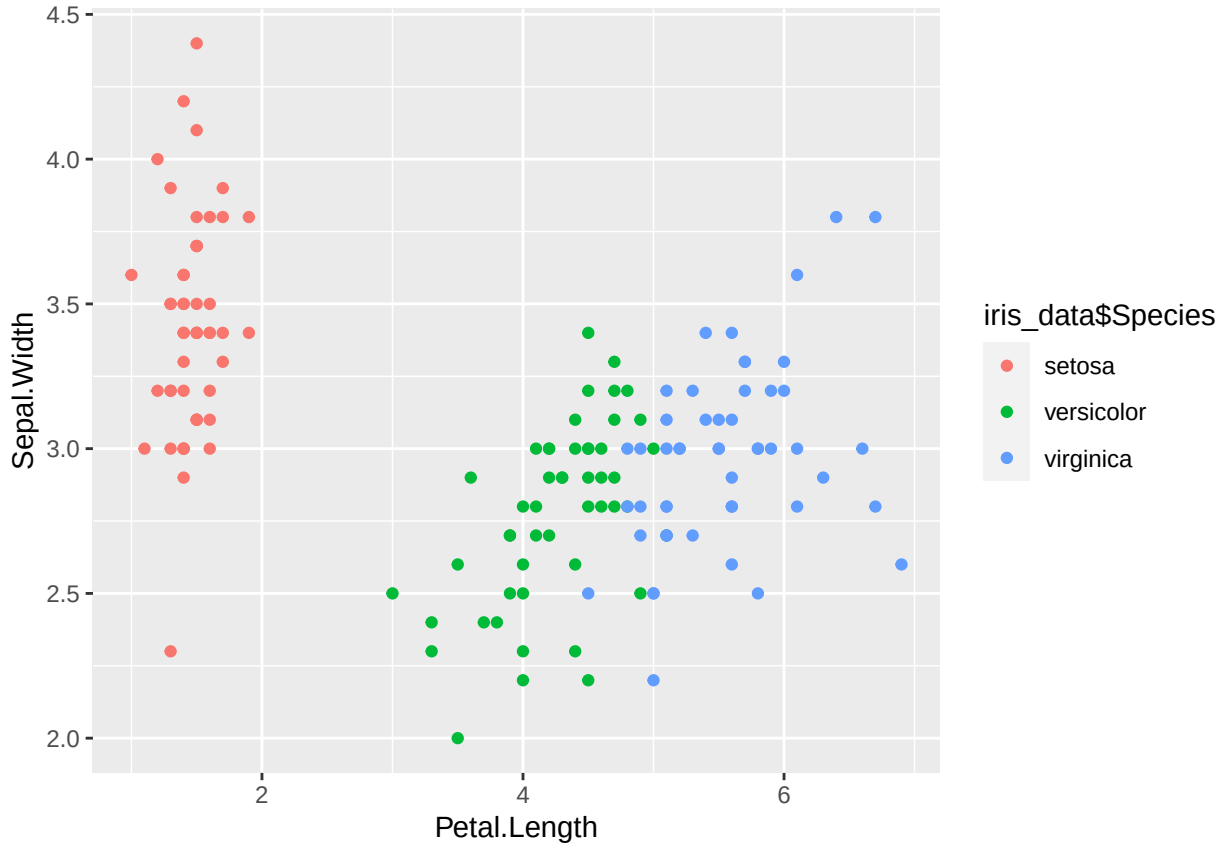

```
ggplot(data = iris_data, aes(Petal.Length,
                             Sepal.Length,
                             color=iris_data$Species))+
  geom_point(alpha=1) #Shows three ok clusters
```



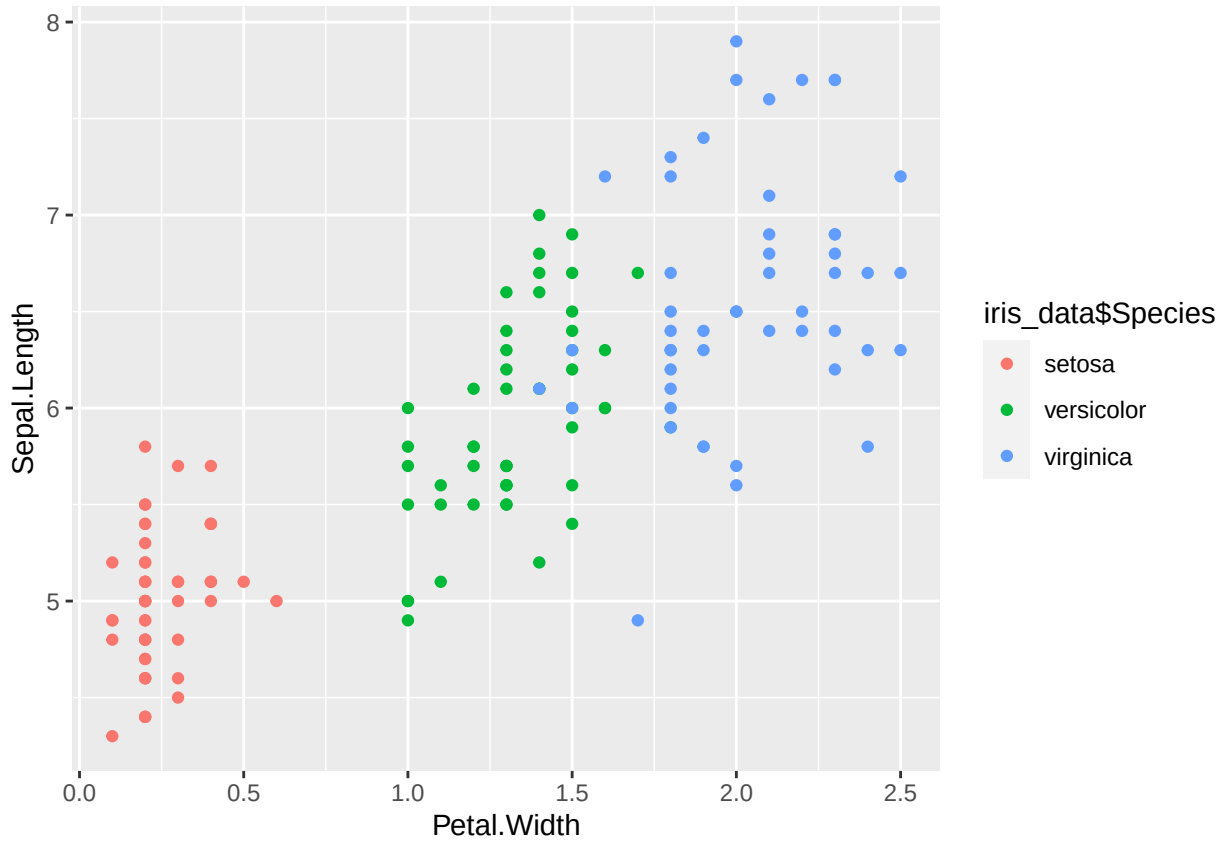

```
ggplot(data = iris_data, aes(Petal.Width,
                             Sepal.Width,
                             color=iris_data$Species))+
  geom_point(alpha=1) #Shows three good clusters
```




```
ggplot(data = iris_data, aes(Petal.Length,
                             Sepal.Width,
                             color=iris_data$Species))+
  geom_point(alpha=1) #Shows three clusters
```




```
ggplot(data = iris_data, aes(Petal.Width,
                             Sepal.Length,
                             color=iris_data$Species))+
  geom_point(alpha=1) #Shows three clusters
```




```
#This above method of trying to plot all comibnation of predictor variables will
#not be effective when their are a lot of predictors. The conclusion from this
#is that we need somewhere between 2 to 3 clusters, but we will test for
#upto 5 clusters

#######Now evaluating kmeans with different data combinations#################

#Scaling the data
scale_iris<-scale(iris_data[-5])

#Tesing the algorithm with all the predictors for clusters between (2-5)
table1<-list()
for (i in 2:5){
  model_all<-kmeans(x=scale_iris[,1:4], centers = i, nstart = 30)
  table1[[i]]=table(iris_data$Species,model_all$cluster)
  print(table1[[i]])
}
```

```
##
##                1  2
##   setosa       0 50
##   versicolor  50  0
##   virginica   50  0
##
##                1  2  3
##   setosa       0 50  0
```

```
##   versicolor 39  0 11
##   virginica  14  0 36
##
##               1  2  3  4
##   setosa     25 25  0  0
##   versicolor  0  0 39 11
##   virginica   0  0 14 36
##
##               1  2  3  4  5
##   setosa     22  0  0  0 28
##   versicolor  0  2 21 27  0
##   virginica   0 27  2 21  0
```

```
#Tesing the algorithm with Petal Length & Petal Width for clusters between (2-5)
table2<-list()
for (i in 2:5){
  model_all<-kmeans(x=scale_iris[,3:4], centers = i, nstart = 30)
  table2[[i]]=table(iris_data$Species,model_all$cluster)
  print(table2[[i]])
}
```

```
##
##               1  2
##   setosa      0 50
##   versicolor 50  0
##   virginica  50  0
##
##               1  2  3
##   setosa      0 50  0
##   versicolor 48  0  2
##   virginica   4  0 46
##
##               1  2  3  4
##   setosa      0  0  0 50
##   versicolor 42  8  0  0
##   virginica   0 23 27  0
##
##               1  2  3  4  5
##   setosa      0 50  0  0  0
##   versicolor 24  0  0 23  3
##   virginica   4  0 26  0 20
```

```
#Tesing the algorithm with Petal Width & Sepal Width for clusters between (2-5)
table3<-list()
for (i in 2:5){
  model_all<-kmeans(x=scale_iris[,c(2,4)], centers = i, nstart = 30)
  table3[[i]]=table(iris_data$Species,model_all$cluster)
  print(table3[[i]])
}
```

```
##
##               1  2
##   setosa      1 49
```

```
##   versicolor 50  0
##   virginica  50  0
##
##               1  2  3
##   setosa     49  0  1
##   versicolor  0 15 35
##   virginica   0 39 11
##
##               1  2  3  4
##   setosa     16  1 33  0
##   versicolor  0 35  0 15
##   virginica   0 11  0 39
##
##               1  2  3  4  5
##   setosa      0 16  1 33  0
##   versicolor 29  0 18  0  3
##   virginica  19  0  6  0 25
```

```
#By comparing all the tables its clear that 3 clusters can define the data well,
#with Petal length and Petal width providing the best classification with
#Sertosa in cluster 3 completely without any misclassification. For Versicolor
#most of them are in cluster 2 with only 2 in cluster 1. Similarly,for virginica
#most of them are in cluster 1 with only 4 in cluster 2. The conclusions are
#made using table 2 using k=3 clusters.
```