kmeans_with_prediction

Viraj @DataTasteMaker

March 29, 2016

k-means clustering and predicting:

Clear memory

```
rm(list=  ls())
```

Load *iris* data

```
set.seed(1) # to get same output everytime
ipdata <- iris[,-5]
head(ipdata)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width
## 1          5.1         3.5          1.4         0.2
## 2          4.9         3.0          1.4         0.2
## 3          4.7         3.2          1.3         0.2
## 4          4.6         3.1          1.5         0.2
## 5          5.0         3.6          1.4         0.2
## 6          5.4         3.9          1.7         0.4
```

```
## Original iris data
```

```
##   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
```

Run k-means with 3 clusters as *iris* data has 3 segments

```
km <- kmeans(ipdata,3)
```

See how the means/centers are for each cluster

```
km$centers
```

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

Create a list of new values to predict

```
# new data record values
myvalues <- c(5.1,3.5,1.4,0.2) # here I have used the first record values
```

now extract mean values for each cluster, then find the Euclidean distance of each values with the new values

```
model_cluster1 <- km$centers[1,]
model_cluster2 <- km$centers[2,]
model_cluster3 <- km$centers[3,]

ed_cluster1 <- sqrt(sum((model_cluster1 - myvalues)^2))
ed_cluster2 <- sqrt(sum((model_cluster2 - myvalues)^2))
ed_cluster3 <- sqrt(sum((model_cluster3 - myvalues)^2))
```

The distances for each clusters are as below

```
cat("The distances are", ed_cluster1, ed_cluster2, ed_cluster3)
```

```
## The distances are 0.1413506 5.059542 3.419251
```

Now find the minimum value of the 3 distances, the one with the minimum value is the cluster this new data will be falling into

```
cat("The new record withe values -->", myvalues, " will fall into cluster# = ",
which.min(as.list(c(ed_cluster1,ed_cluster2,ed_cluster3))))
```

```
## The new record withe values --> 5.1 3.5 1.4 0.2  will fall into cluster# =  1
```

To cross validate our model we can do it as below

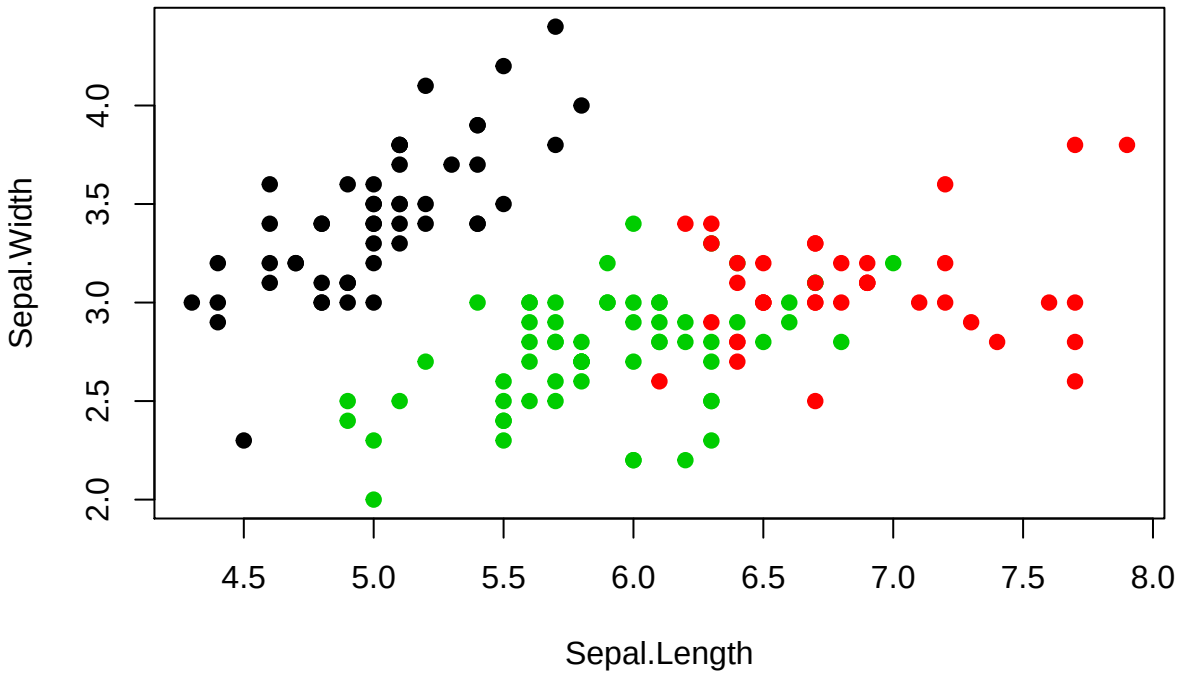
```
## As I had pulled the first record of the data file, let's see which segment it was marked in

iris[1,]
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
```

```
## as you see the cluster it is marked in is "setosa"
## and as per the table below, the kmeans model is setting Cluster 1 as Setosa
table(iris$Species,km$cluster)
```

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

```
plot(ipdata[c(1:2)], col=km$cluster, pch=19)
```


Sepal.Width
2.0
2.5
3.0
3.5
4.0
4.5
5.0
5.5
6.0
6.5
7.0
7.5
8.0
Sepal.Length