Clustering by GMM and DBSCAN

Group 1: Pariente Antonio, Bosch Guillem, Ebner Lena

11/10/2023

Clustering by GMM and DBSCAN

Introduction

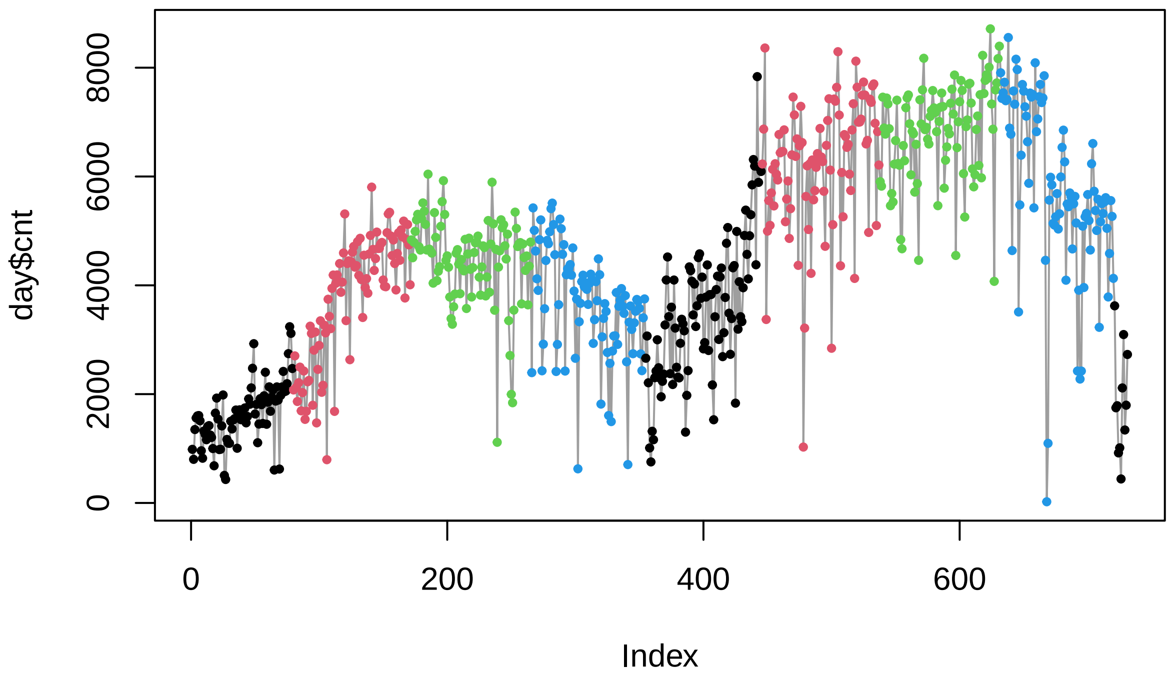
The file `BikeDay.Rdata` contains information on the bike-sharing rental service in Washington D.C., USA, corresponding to years 2011 and 2012. This file contains only one data frame, `day`, with 731 rows (one for each day of years 2011 and 2012, that was a leap year) and 16 columns:

- `instant` row index, going from 1 to 731
- `dteday` date
- `season` (1:spring, 2:summer, 3:fall, 4:winter)
- `yr` year (0: 2011, 1:2012)
- `mnth` 1 for January, until 12 for December
- `holiday` whether day is holiday or not
- `weekday` day of the week (0 Sunday to 6 Saturday)
- `workingday` if day is neither weekend nor holiday is 1, otherwise is 0
- `weathersit`:
  - 1: Clear, Few clouds, Partly cloudy, Partly cloudy
  - 2: Mist + Cloudy, Mist + Broken clouds, Mist + Few clouds, Mist
  - 3: Light Snow, Light Rain + Thunderstorm + Scattered clouds, Light Rain + Scattered clouds
  - 4: Heavy Rain + Ice Pallets + Thunderstorm + Mist, Snow + Fog
- `temp` Normalized temperature in Celsius. The values are divided to 41 (max)
- `atemp` Normalized feeling temperature in Celsius The values are divided to 50 (max)
- `hum` Normalized humidity. The values are divided to 100 (max)
- `windspeed` Normalized wind speed. The values are divided to 67 (max)
- `casual` count of rental bikes by causal users (not registered)
- `registered` count of rental bikes by registered users.
- `cnt` count of total rental bikes (casual + registered)

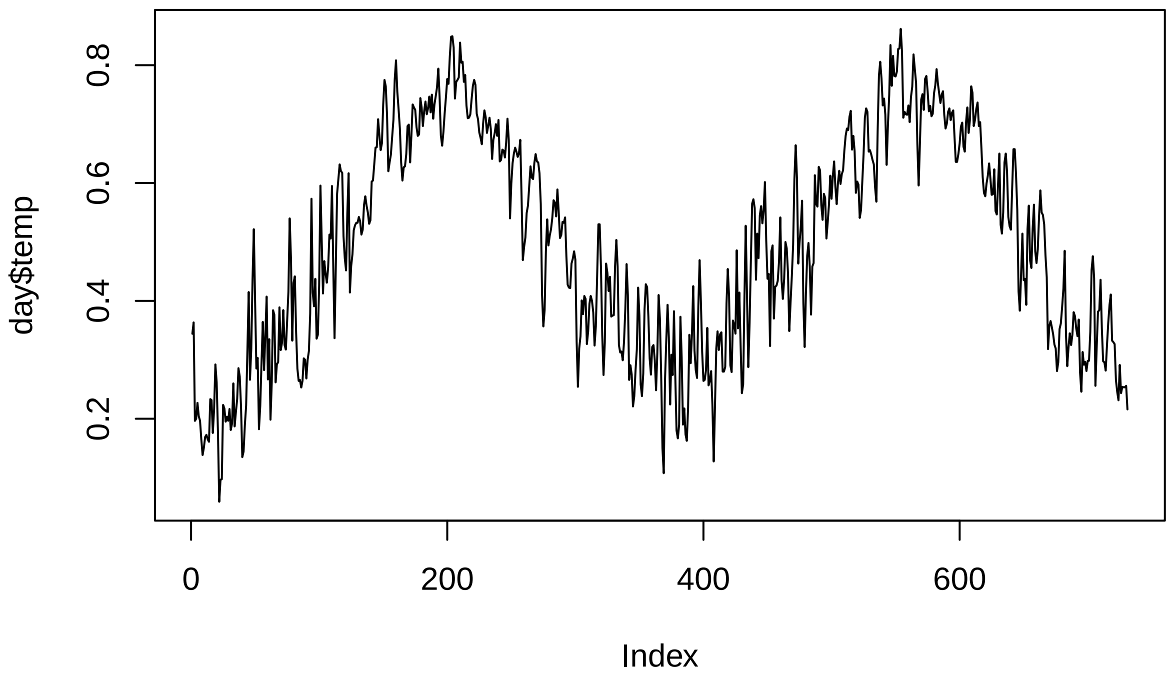
In particular we are interested in the joint distribution of `temp` and `casual` for year 2012:

Plots

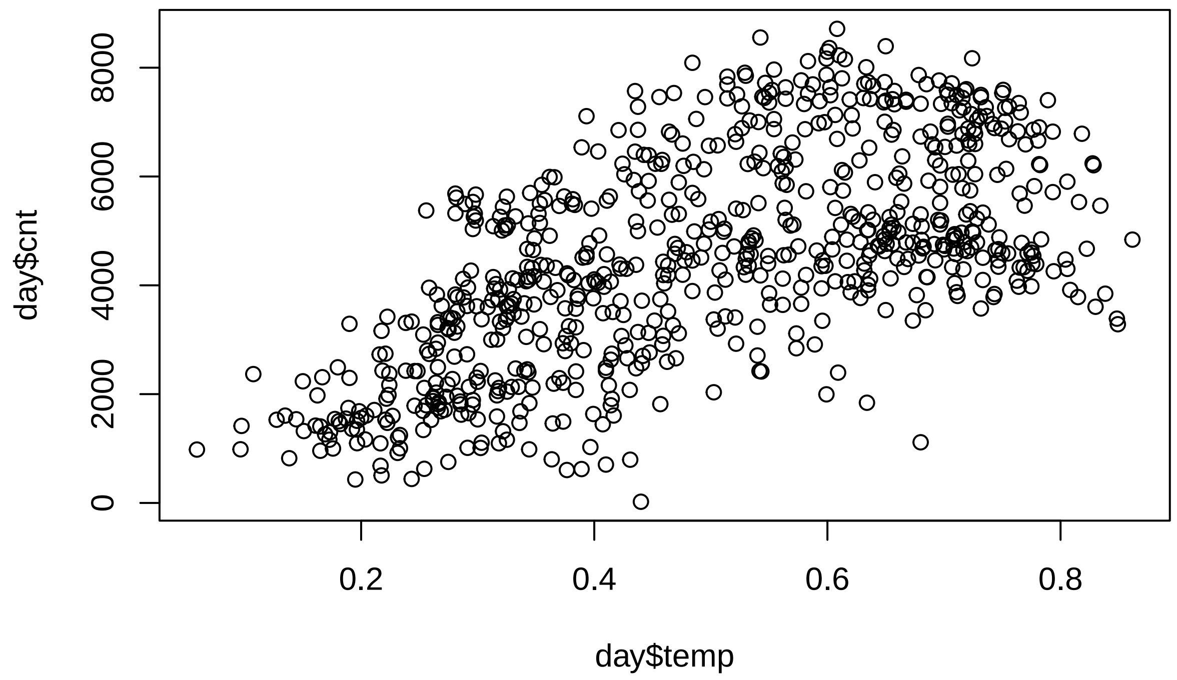
```r
plot(day$cnt, type="l", col=8) # timeseries plot
points(day$cnt, col=day$season, pch=19, cex=.5)
```



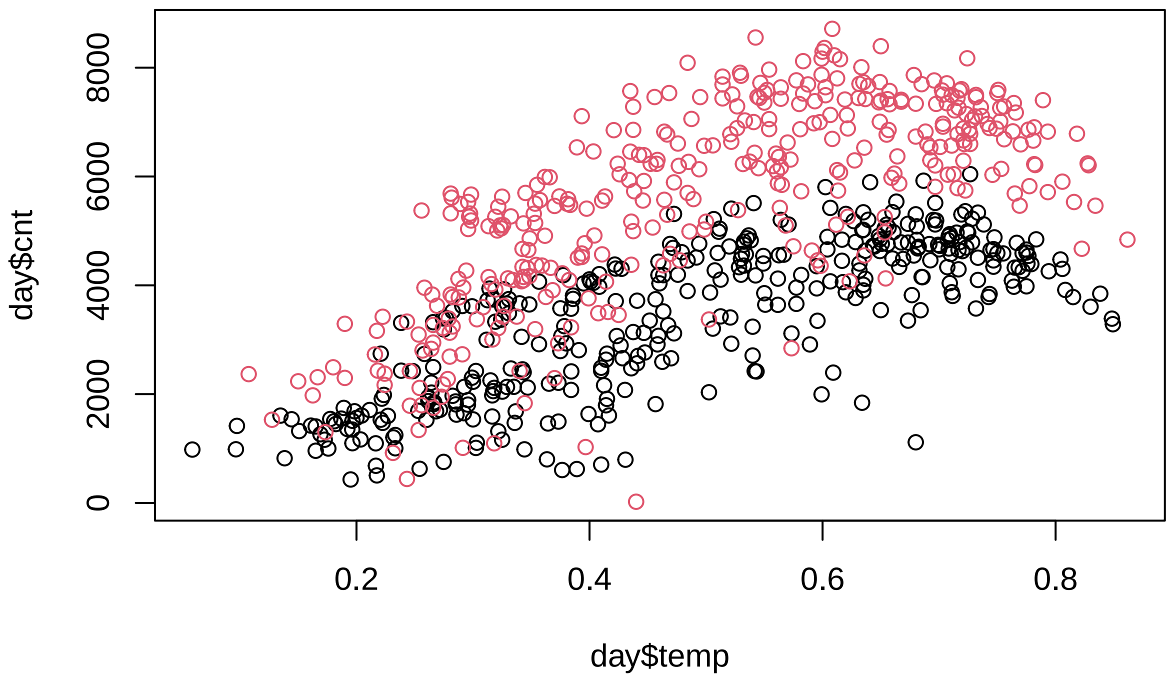
```
plot(day$temp, type="l")
```



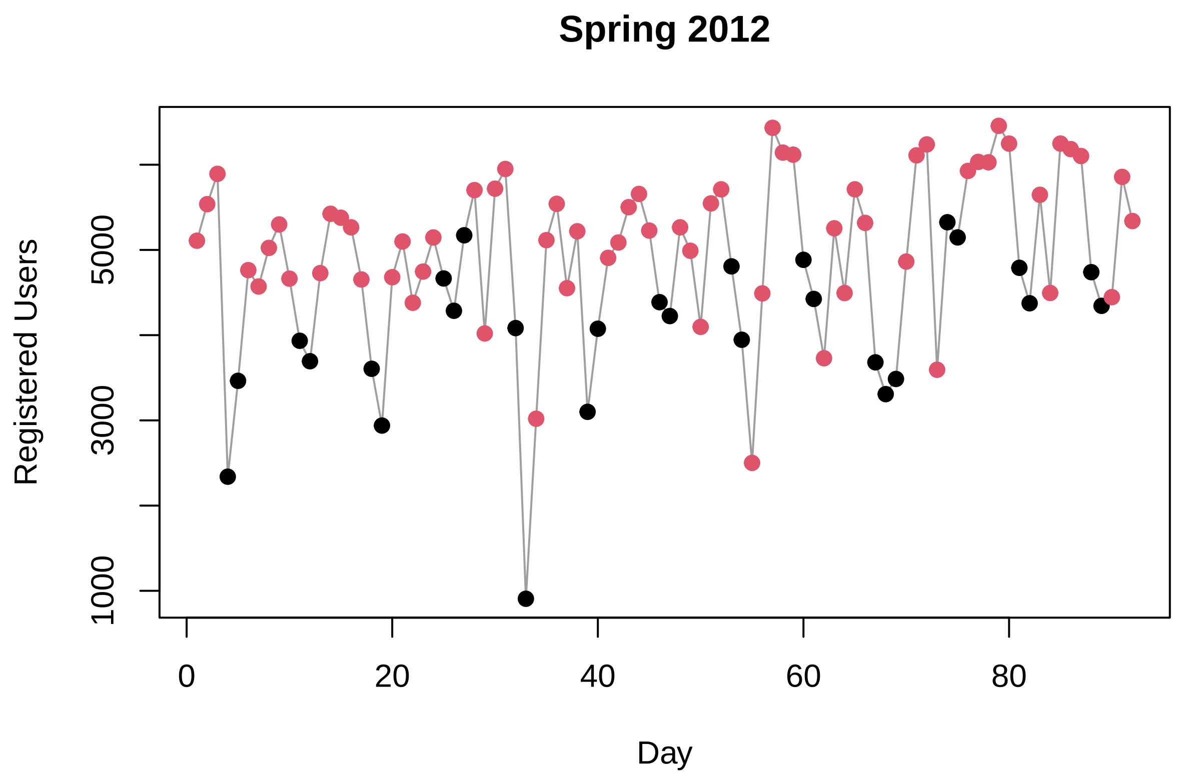
```
plot(day$temp, day$cnt)
```



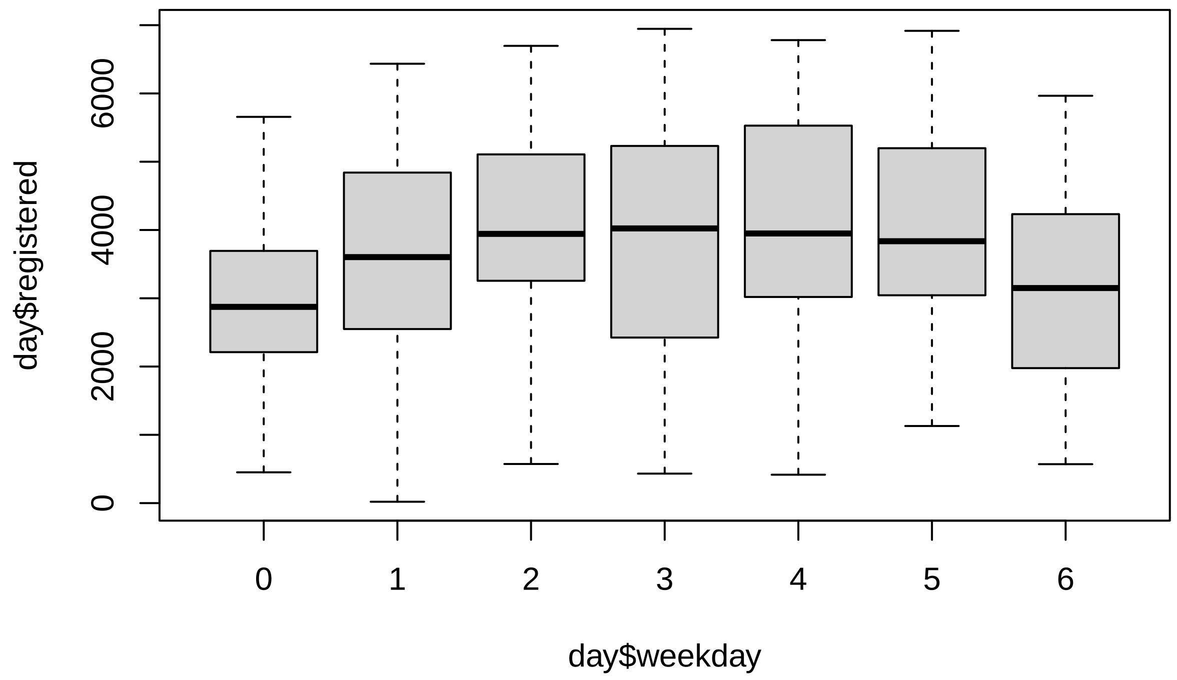
```r
plot(day$temp, day$cnt, col=day$yr+1)
```



```
selected <- which((day$season==2) &(day$yr==1))
plot(day$registered[selected], type="l", col=8,
    main="Spring 2012", ylab="Registered Users", xlab="Day")
points(day$registered[selected], col=day$workingday[selected]+1, pch=19)
```



```r
boxplot(day$registered~day$weekday)
```

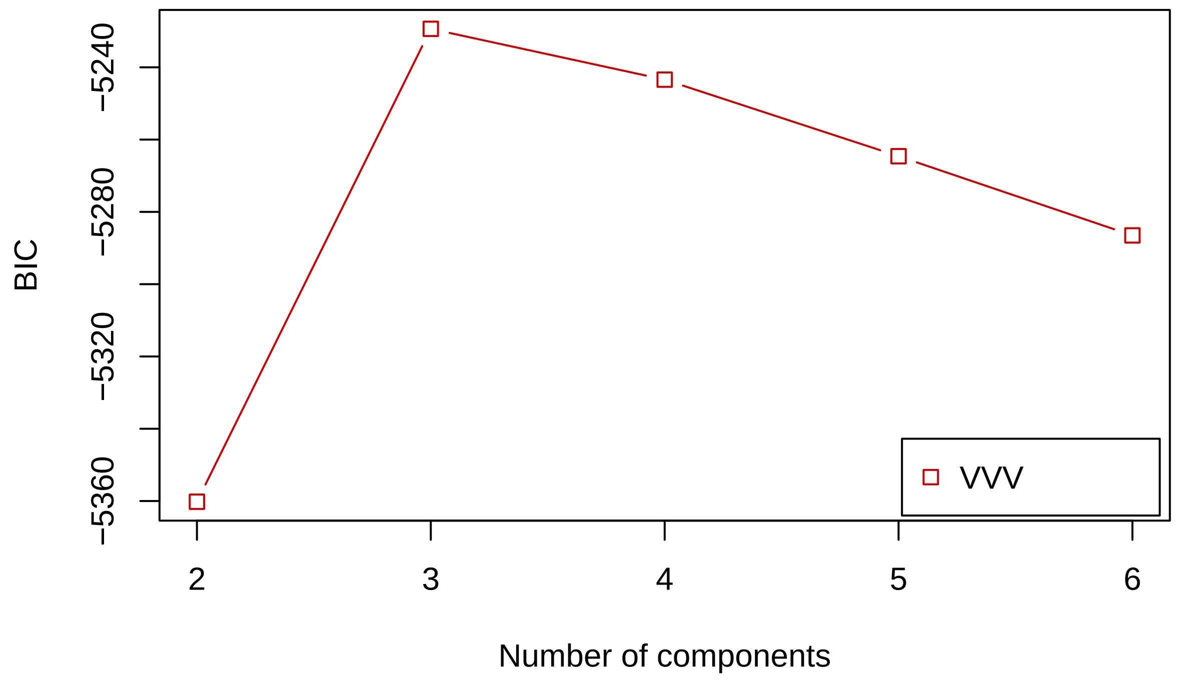


Implementation

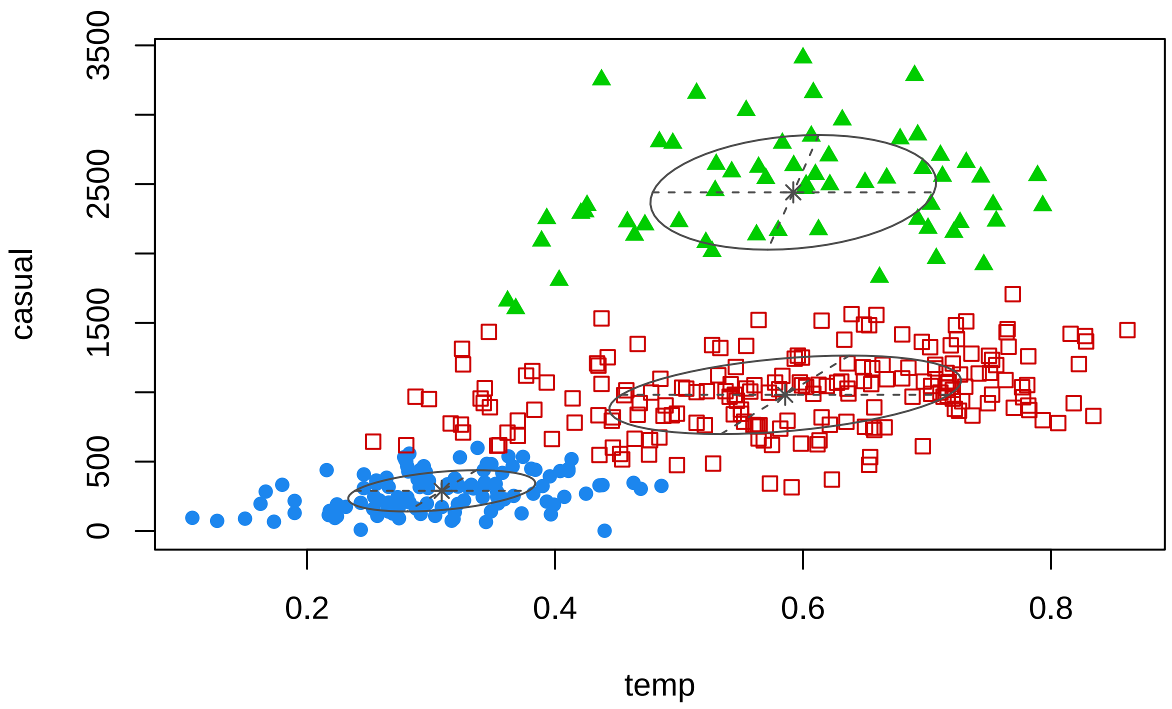
1. Use library `mclust` for the following task. Do a model based clustering of these data assuming a Gaussian Mixture Model, allowing varying volume, shape, and orientation for different components in the mixture. Choose k_{BIC}, the best number of clusters $k \in \{2, \ldots, 6\}$ according to BIC.
   Plot the resulting object from `Mclust` (do 4 different graphics: `BIC`, `classification`, `uncertainty` and `density`).

```
load("./BikeDay.Rdata")
X <- as.matrix(day[day$yr==1,c(10,14)])
```

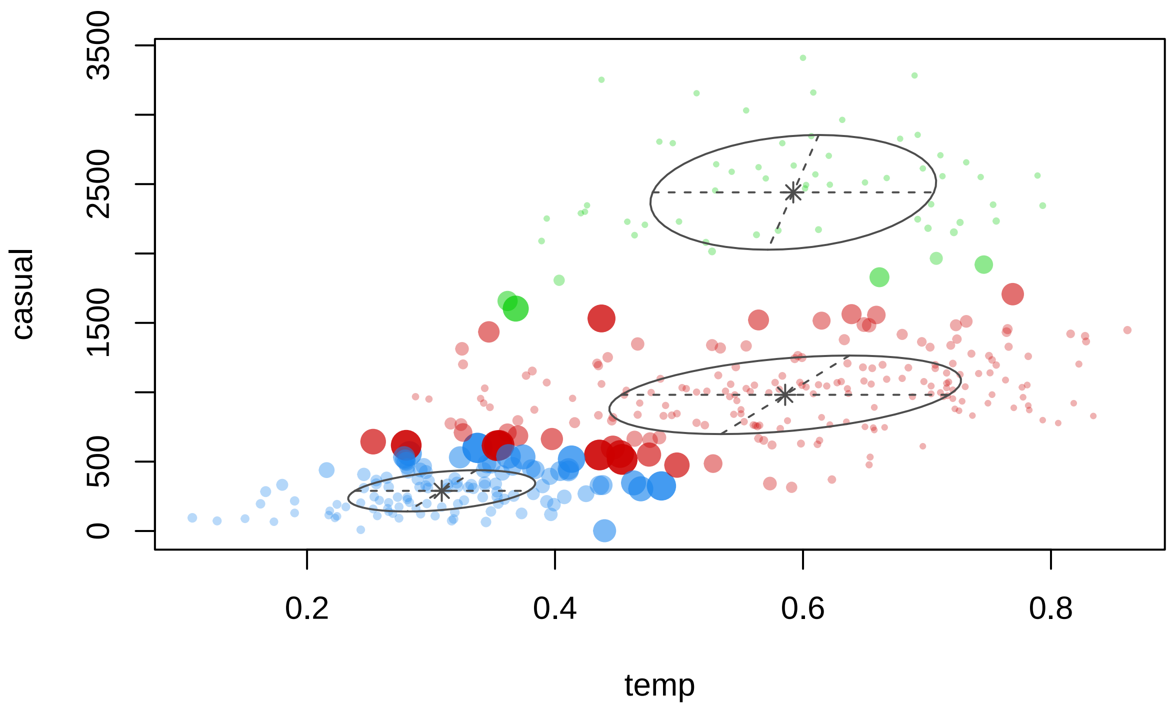
```
rng=2:6
GMM <- Mclust(X,G=rng, modelNames = "VVV")

plot(GMM, what="BIC")
```



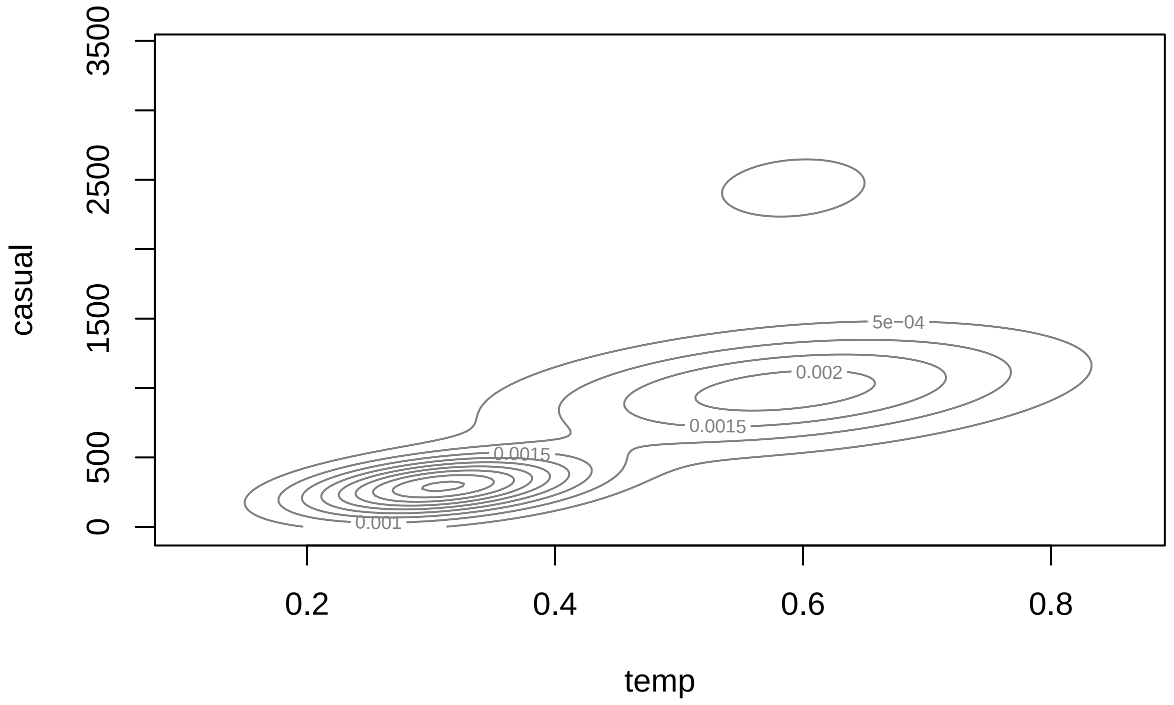
```r
plot(GMM, what="classification")
```



```
plot(GMM, what="uncertainty")
```



```r
plot(GMM, what="density")
```



```
# points(X)

k=rng[which.max(GMM$BIC)]
```

The best number of clusters with respect to the BIC criteria is $k = 3$.

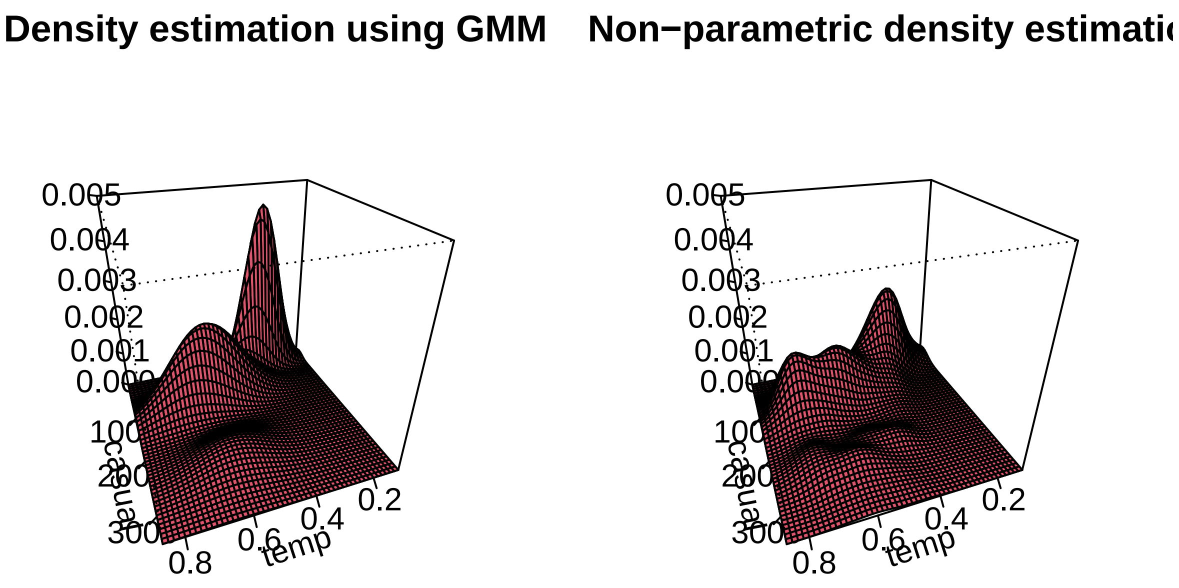
---

2. Compare the previous `density` plot with the non-parametric density estimation of (`temp`,`casual`) obtained when using the kernel estimator implemented in `sm::sm.density`, with the bandwidths proportional to the standard deviations in both dimensions: $h = a \cdot (StdDev(\texttt{temp}), StdDev(\texttt{casual}))$. *Use* $a = 0.25$.

To compare the density functions we will plot both of them in 3D

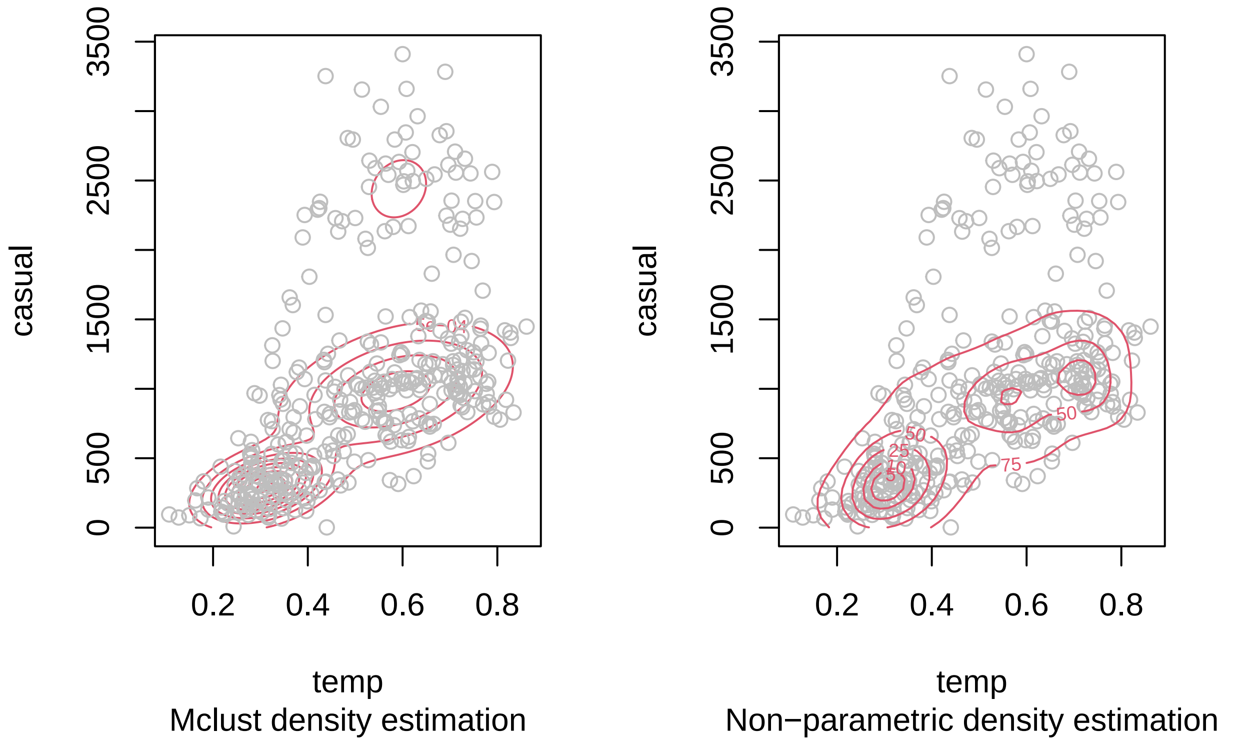
```
mixprobs = GMM$parameters$pro
mixmus = GMM$parameters$mean
mixsigmas = GMM$parameters$variance$sigma

par(mfrow = c(1, 2))

density_mixture_of_multivariate_gaussians <- function(X, means, covariances, probs) {
  num_components <- dim(means)[2]
  density <- rep(0, nrow(X))
```

```r
  for (i in 1:num_components) {
    density <- density + probs[i] * dmvnorm(X, mean = means[,i], sigma = covariances[,,i])
  }
  return(density)
}
ln = 50
temp_values <- seq(min(X[,"temp"]), max(X[,"temp"]), length.out = ln)
casual_values <- seq(min(X[,"casual"]), max(X[,"casual"]), length.out = ln)
mesh <- expand.grid(temp = temp_values, casual = casual_values)
mesh$mix_dens <- density_mixture_of_multivariate_gaussians(as.matrix(mesh), mixmus, mixsigmas, mixprobs
mix_dens_matrix <- matrix(mesh$mix_dens, nrow = ln, ncol = ln, byrow = FALSE)

persp(temp_values, casual_values, mix_dens_matrix, col = 2,
        xlab = "temp", ylab = "casual", zlab = "", main = "Density estimation using GMM", theta = 160, 

h = 0.25 * c(sd(X[, "temp"]), sd(X[, "casual"]))

mesh$np_dens <- sm.density(X,h=h,display="none",eval.grid=FALSE, eval.points=mesh[, c("temp", "casual")
np_dens_matrix <- matrix(mesh$np_dens, nrow = ln, ncol = ln, byrow = FALSE)

persp(temp_values, casual_values, np_dens_matrix, col = 2, xlab = "temp", ylab = "casual", zlab = "", m
```



```r
par(mfrow = c(1, 2))

plot(GMM, what="density", sub="Mclust density estimation", col=2)
```

```r
points(X, col = "grey")

plot(X, sub="Non-parametric density estimation", col = "grey")
sm.density(X, h=h, display="slice", col=2, add=TRUE, props = c(5,10,25,50,75))
```



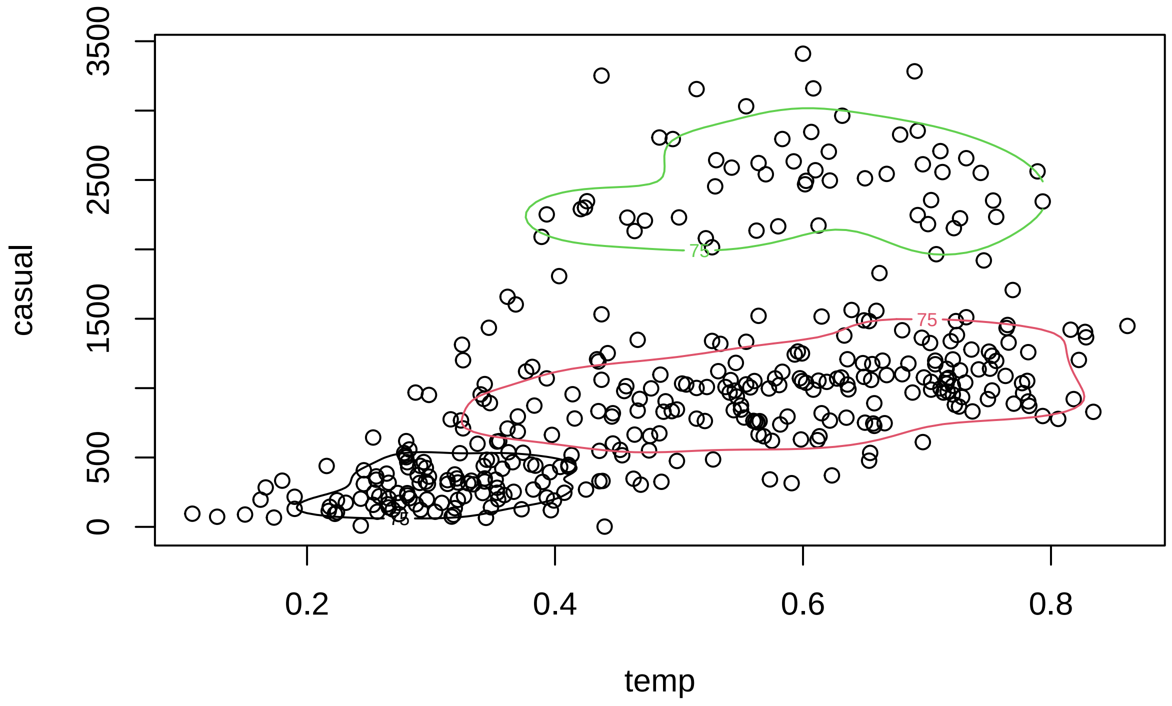
Comparison:
Comparing the 2 different density estimations, it is clear to see that the non-parametric density estimation is able to detect a more detailed version of the density inside the data. In this density each data point creates a small mountain in the density plot. Even if the points are isolated the mountain appears visible. On the other hand, the GMM density puts a stronger focus on the main clusters; for example, it also detects some density at ~ (0.6, 2500) but neglects the effect of isolated points. This second density is smoother in the sense that there are less peaks but the defined peaks are more remarkable.

---

3. For each one of the k_{BIC} clusters obtained above, do the following tasks *(A unique plot should be done, at which the k* densities are represented simultaneously):

- Consider the bivariate data set of the points in this cluster.
- Estimate non-parametrically the joint density of `temp` and `casual`, conditional to this cluster, using the kernel estimator implemented in `sm::sm.density` with the bandwidths proportional to the standard deviations in both dimensions: $h = a \cdot (StdDev(\texttt{temp}), StdDev(\texttt{casual}))$. *Use* $a = 0.4$ and compute the standard deviations at each cluster.
- Represent the estimated bivariate density using the level curve that covers the 75% of the points in this cluster.

```r
indexes <-  GMM$classification
plot(X)
for (j in 1:k){
  cluster_j <- (indexes==j)
  h = 0.4 * c(sd(X[cluster_j, "temp"]), sd(X[cluster_j, "casual"]))
  sm.density(X[cluster_j,],h=h,
             display="slice",
             col=j,props=c(75), cex=4, add=TRUE)
}
```

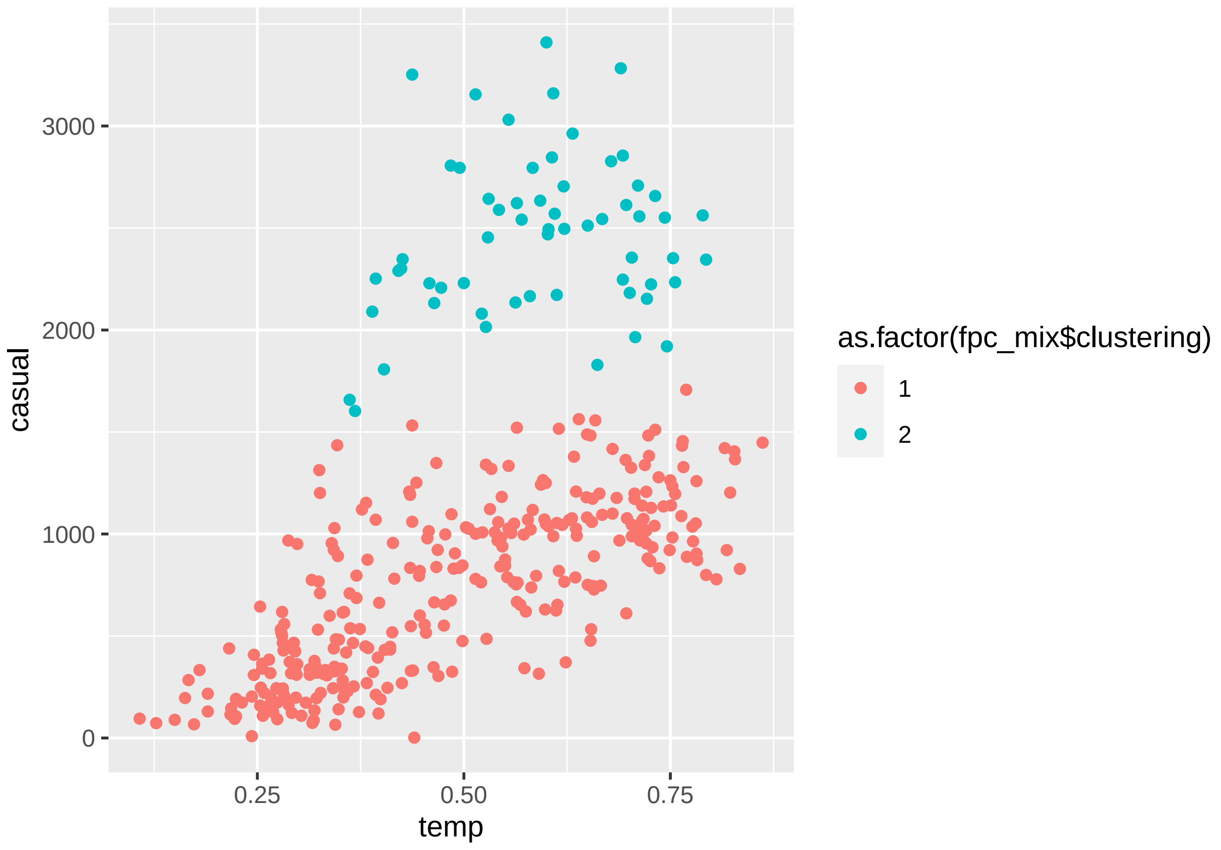


---

4. Use library `fpc` to check if it is possible to merge some of the components in the Gaussian Mixture Model previously estimated. Let k^* be the final number of clusters after the merging process. Do the scatterplot of (`temp`,`casual`) with colors according to the new k^* clusters.
   Indication: Use the function `mergenormals` with the option `method="bhat"`.

It is possible to merge two of the clusters defined in the previous section.

```r
scm=summary(mclustBIC(X, G=k, modelNames = "VVV"),X)
fpc_mix<- fpc::mergenormals(X,method="bhat",scm)
Y=as.data.frame(X)
ggplot(Y,aes(x=temp,y=casual,colour=as.factor(fpc_mix$clustering)))+geom_point()
```

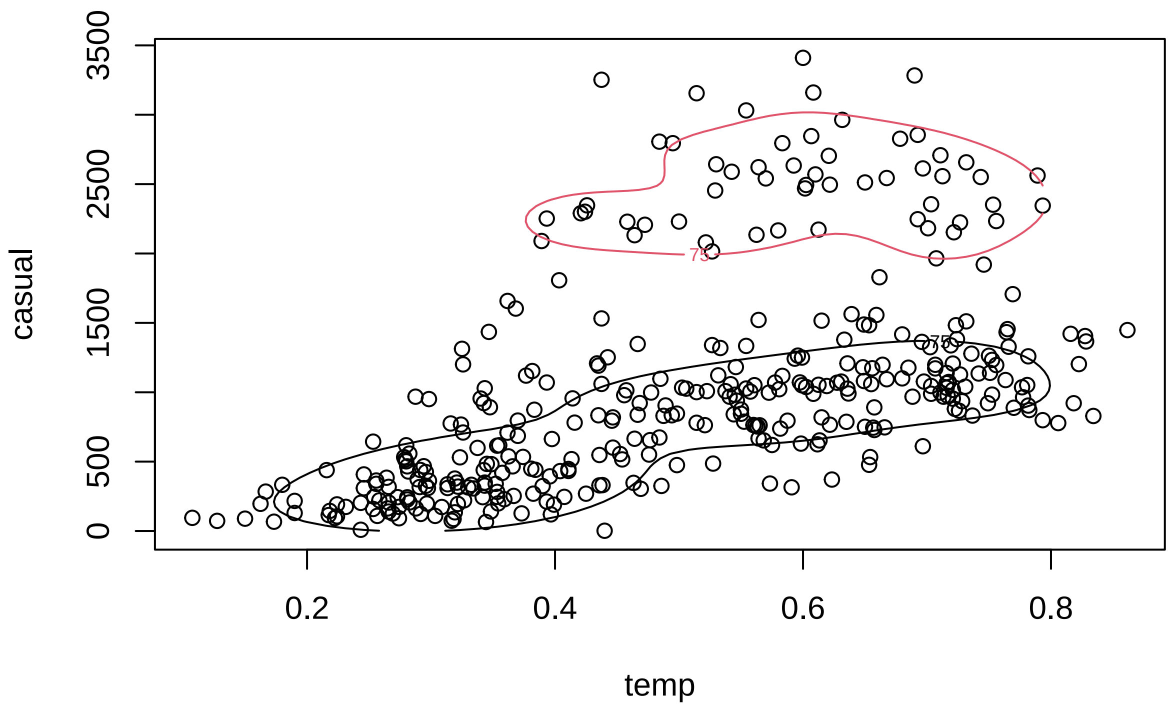


---

5. For each one of the k^* clusters obtained above, do the following tasks *(A unique plot should be done, at which the k* densities are represented simultaneously):

- Consider the bivariate data set of the points in this cluster.
- Estimate non-parametrically the joint density of (`temp`,`casual`), conditional to this cluster, using the kernel estimator implemented in `sm::sm.density` with the bandwidths proportional to the standard deviations in both dimensions: $h = a \cdot (StdDev(\texttt{temp}), StdDev(\texttt{casual}))$. *Use* $a = 0.4$ and compute the standard deviations at each cluster.
- Represent the estimated bivariate density using the level curve that covers the 75% of the points in this cluster.

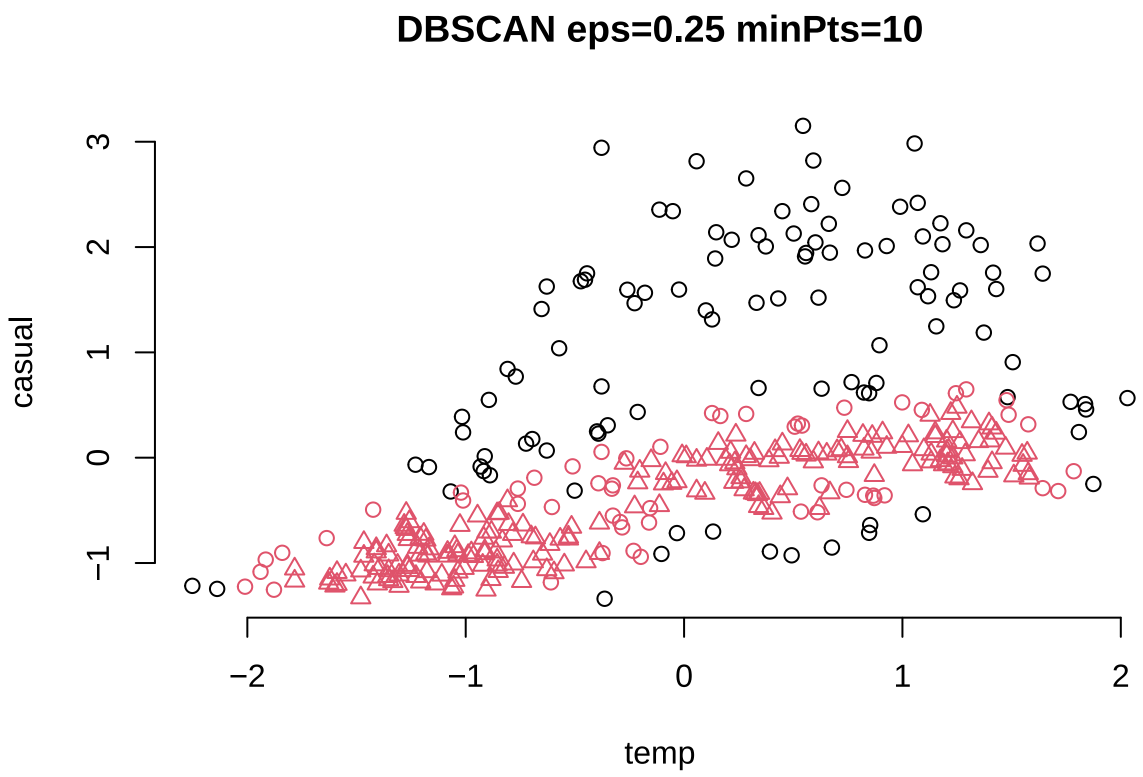
```
indexes<- fpc_mix$clustering
plot(X)
for (j in 1:length(fpc_mix$clusternumbers)){
  cluster_j <- (indexes==j)
  h = 0.4 * c(sd(X[cluster_j, "temp"]), sd(X[cluster_j, "casual"]))
  sm.density(X[cluster_j,],h=h,
             display="slice",
             col=j,props=c(75), cex=4, add=TRUE)
}
```

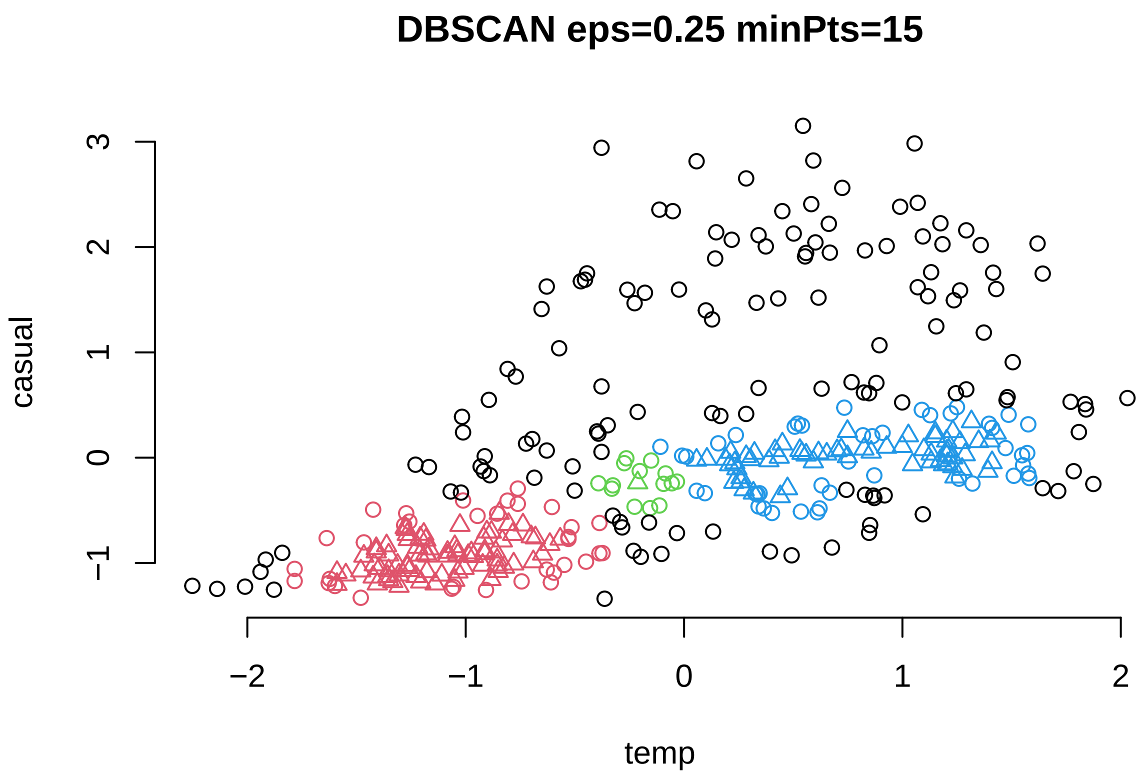


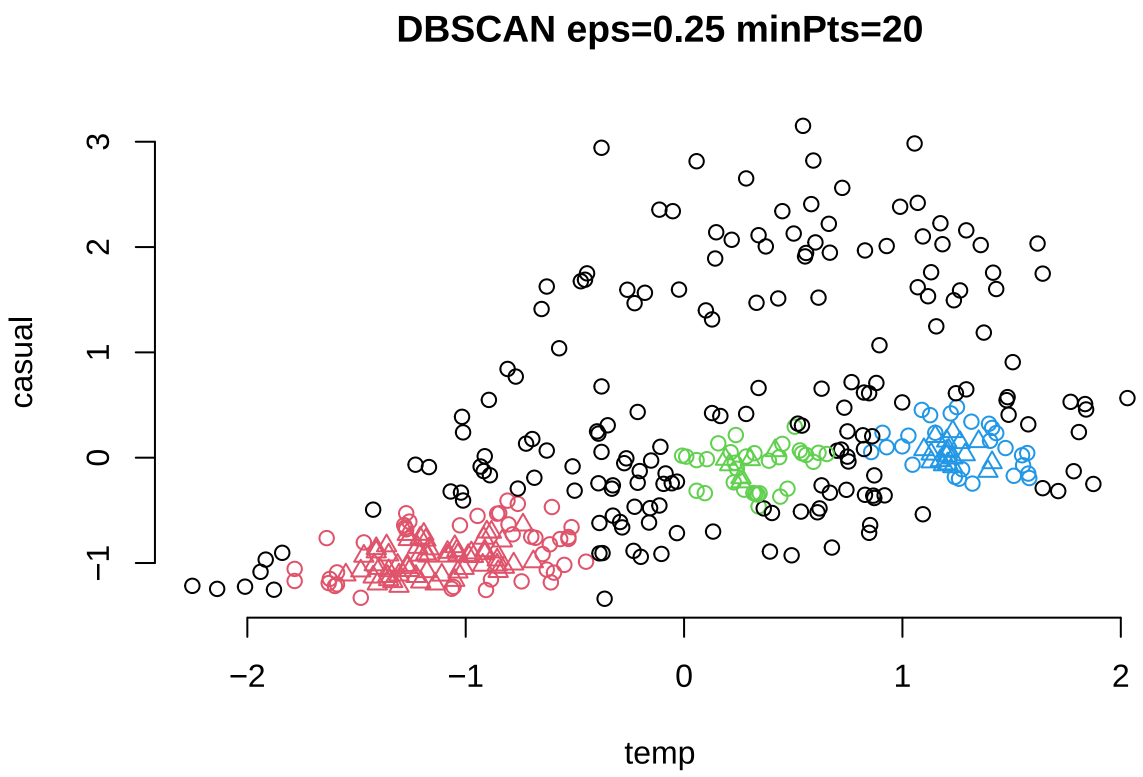
---

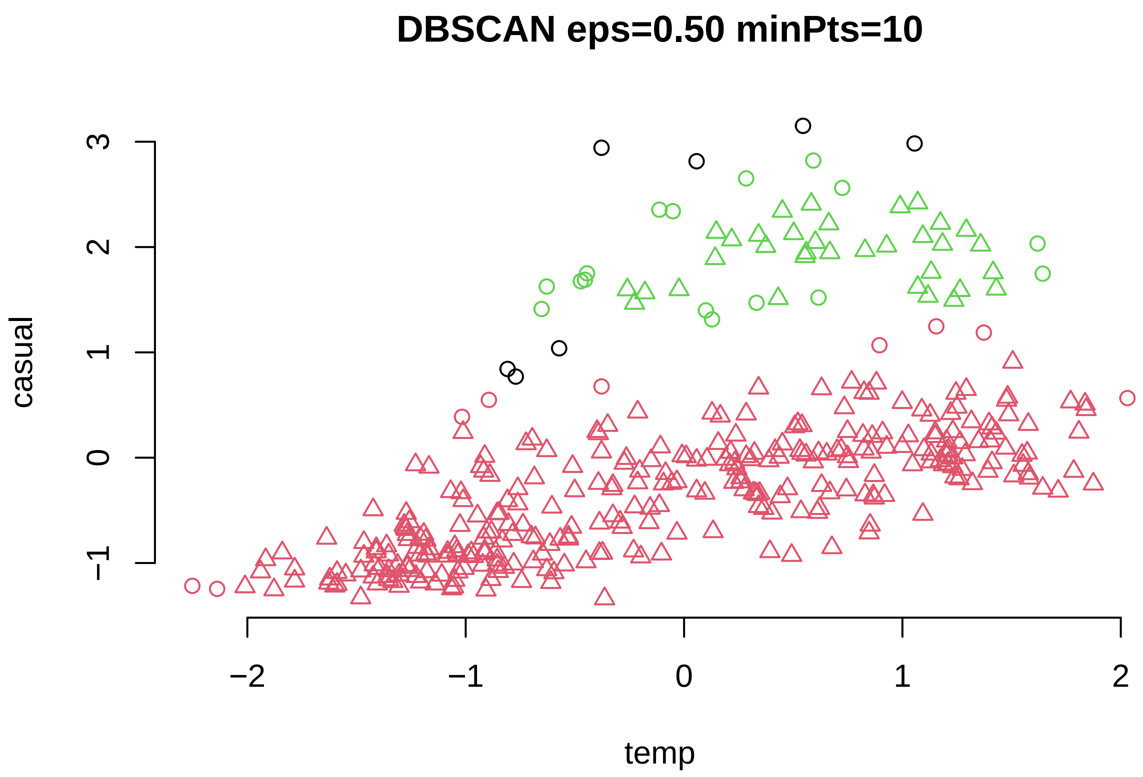
6. Use DBSCAN to find clusters (and outliers) in the data set (`temp`,`casual`), after **centering and scaling** both variables (do `Xs <- scale(X)`). Try $\varepsilon \in \{0.25, 0.5\}$ and `minPts` $\in \{10, 15, 20\}$. Which combination of the tuning parameters do you consider the *best one*?
   Compare the DBSCAN clustering corresponding to your favorite combination of tuning parameters with the results of `mergenormals` (print their cross-table).

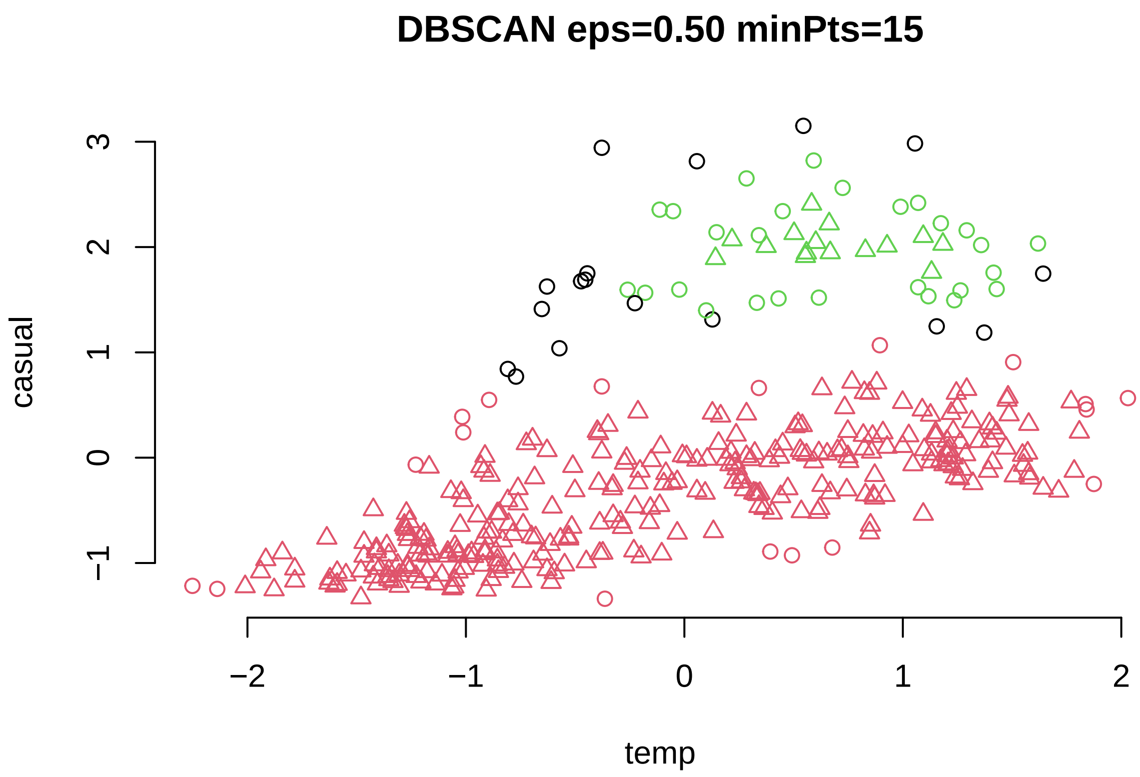
```r
Xs <- scale(X)

epsilons <- c(0.25, 0.5)
minPts_values <- c(10, 15, 20)

results <- list()

# perform DBSCAN clustering for different parameter combinations
for (eps in epsilons) {
  for (minPts in minPts_values) {
    dbscan_result <- fpc::dbscan(Xs, eps = eps, MinPts = minPts, showplot = 0)
    plot(dbscan_result,Xs, main = sprintf("DBSCAN eps=%.2f minPts=%d", eps, minPts), frame = FALSE)
    results[[paste("Eps_", eps,"_MinPts_", minPts, sep = "")]] <- dbscan_result
  }
}
```

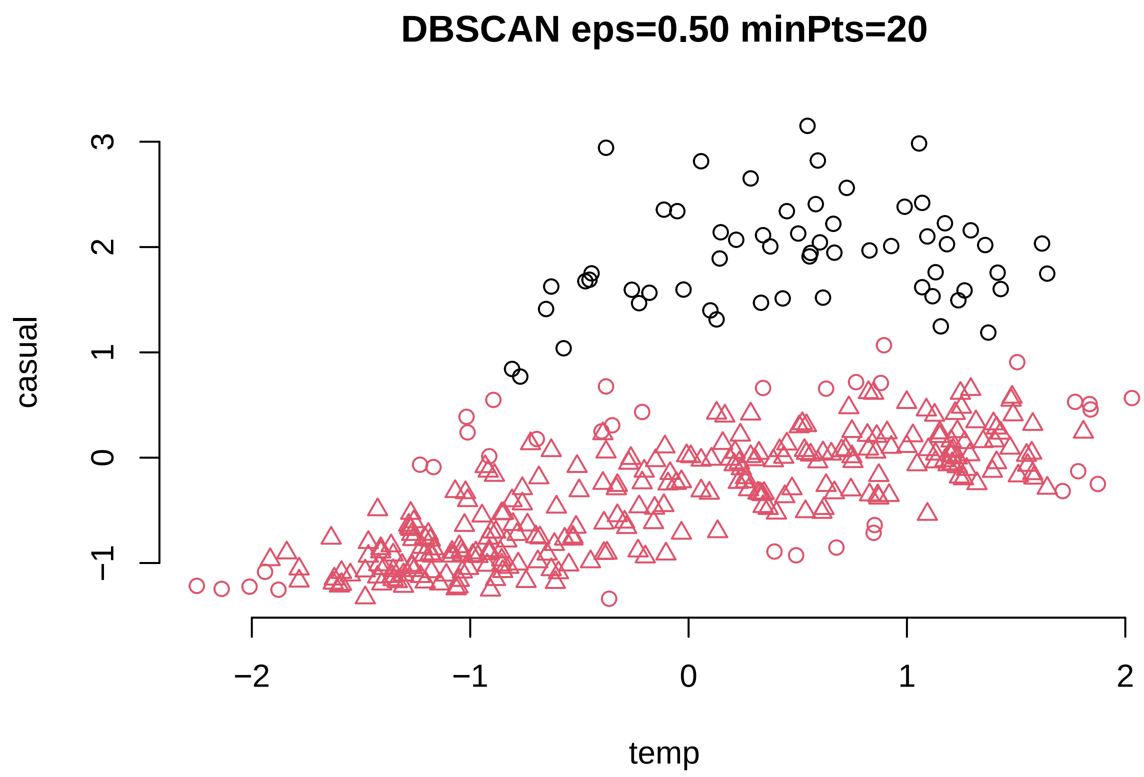

DBSCAN eps=0.25 minPts=10
casual
temp
3
2
1
0
−1
−2
−1
0
1
2


DBSCAN eps=0.25 minPts=15
casual
3
2
1
0
−1
temp
−2
−1
0
1
2


DBSCAN eps=0.25 minPts=20
casual
temp
−2
−1
0
1
2
3


DBSCAN eps=0.50 minPts=10
casual
3
2
1
0
−1
−2
−1
0
1
2
temp


DBSCAN eps=0.50 minPts=15
casual
temp
−2
−1
0
1
2
3



```
results
```

```
## $Eps_0.25_MinPts_10
## dbscan Pts=366 MinPts=10 eps=0.25
##          0   1
## border 105  54
## seed     0 207
## total  105 261
##
## $Eps_0.25_MinPts_15
## dbscan Pts=366 MinPts=15 eps=0.25
##          0   1  2   3
## border 135  34 14  44
## seed     0  76  1  62
## total  135 110 15 106
##
## $Eps_0.25_MinPts_20
## dbscan Pts=366 MinPts=20 eps=0.25
##          0   1  2  3
## border 178  42 32 27
## seed     0  62  6 19
## total  178 104 38 46
##
## $Eps_0.5_MinPts_10
## dbscan Pts=366 MinPts=10 eps=0.5
##        0   1  2
## border 7   9 16
## seed   0 300 34
## total  7 309 50
##
## $Eps_0.5_MinPts_15
## dbscan Pts=366 MinPts=15 eps=0.5
##         0   1  2
## border 17  18 27
## seed    0 289 15
## total  17 307 42
##
## $Eps_0.5_MinPts_20
## dbscan Pts=366 MinPts=20 eps=0.5
##         0   1
## border 59  35
## seed    0 272
## total  59 307
```

Choosing best model based on looking on the plots: the best fit seems to be with `eps` $= 0.5$ and `minPts` $= 10$ where we want to obtain 2 clusters. Also when looking at the statistics we see that in that case only 7 points where classified as noise.

Our favorite clustering separation is almost identical to the one coming from the `mergenormals` output. We think it is the most reasonable clustering on the given data points.

```
best_dbscan_result <- results[["Eps_0.5_MinPts_10"]]

GMM_BIC <- mclustBIC(X,G=2:6, modelNames = "VVV")
GMM_BIC_summary <- summary(GMM_BIC, X)

mergenormals_result <- mergenormals(Xs, GMM_BIC_summary, method="bhat")

cross_table <- table(best_dbscan_result$cluster, mergenormals_result$clustering)
cross_table
```

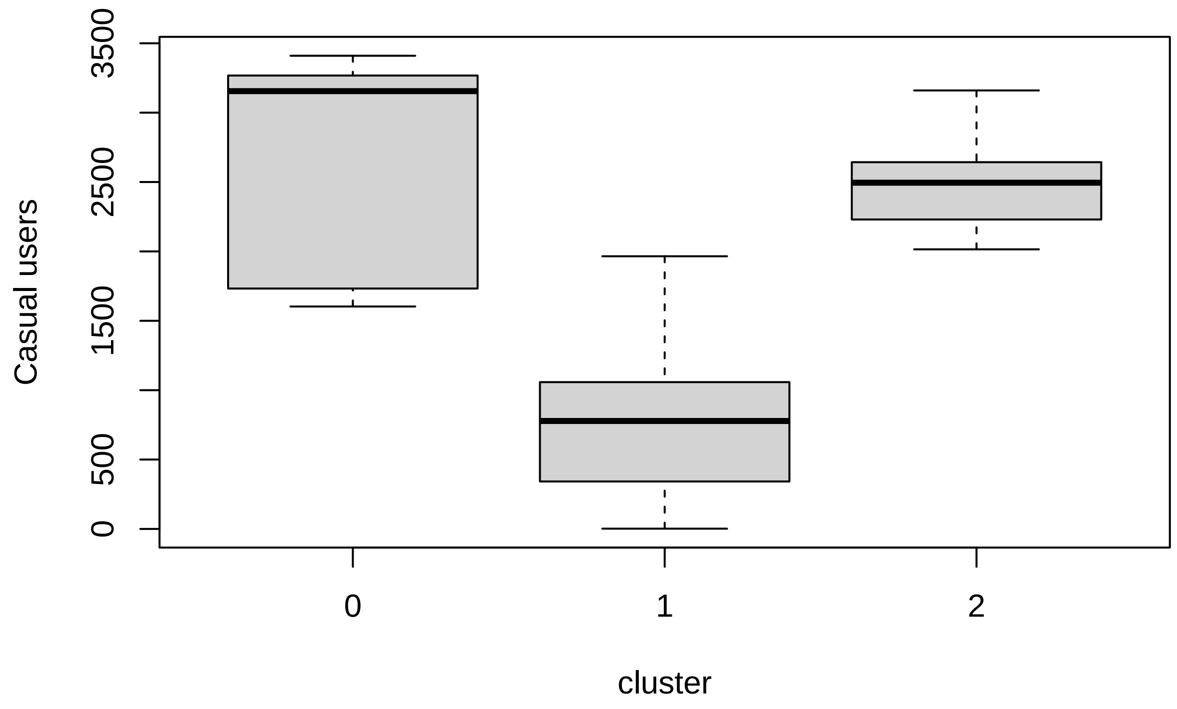
```
##
##       1   2
##   0   0   7
##   1 306   3
##   2   0  50
```

---

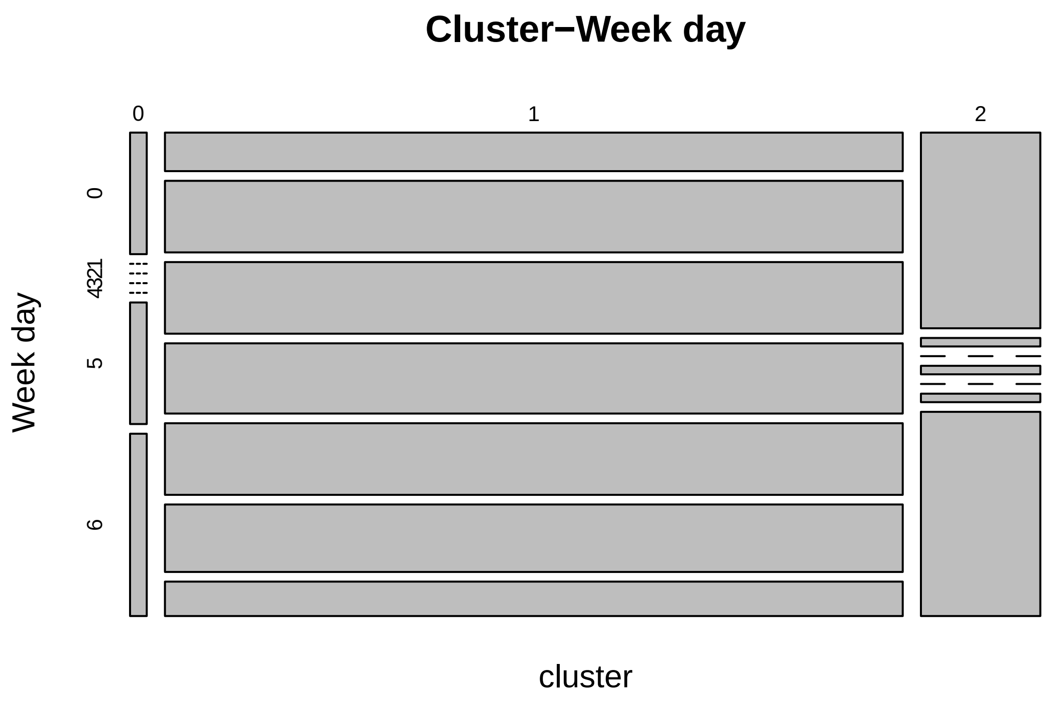
7. Give an interpretation (or explanation, or description) of the clusters you have found before. ***(Indication:** Other variables in the data set can help to describe the clusters).*

The first thing we can say is that the records belonging to the second cluster have more casual users than the first one. This is clearly visible in the previous figures. Additionally, in the following figures we can see that the second cluster has a high proportion of Sundays and Saturdays. Hence we can say that the second cluster represents the weekends while the first one represents mainly the working days. This is also visible in the figure with the variable `workingday`. By looking at figures comparing the clustering labels and other variables we can see other relations, but these are not as strong as the ones already mentioned and do not serve to characterize the clustering.

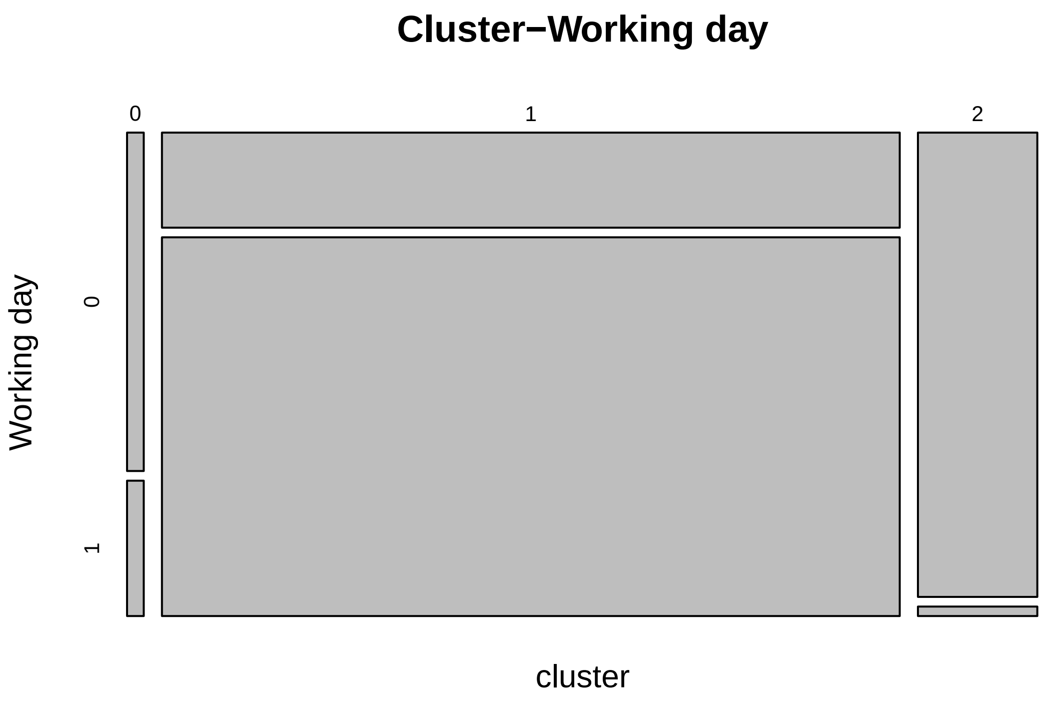
```r
interesting_day = day[day$yr==1,]

#for numerical vars
boxplot(interesting_day$casual~best_dbscan_result$cluster, xlab = "cluster", ylab = "Casual users")
```



```r
#for categorical vars
plot(table(best_dbscan_result$cluster, interesting_day$weekday), xlab = "cluster", ylab = "Week day", ma
```



```
plot(table(best_dbscan_result$cluster, interesting_day$workingday), xlab = "cluster", ylab = "Working da
```


Cluster−Working day
0
1
2
Working day
0
1
cluster