Olufemi Babalola

MSDS692 DATA SCIENCE PRACTICUM 1

MALARIAL PARASITES DETECTION AND CLEARANCE RATES – BAYESIAN HIERARCHICAL REGRESSION MODELLING

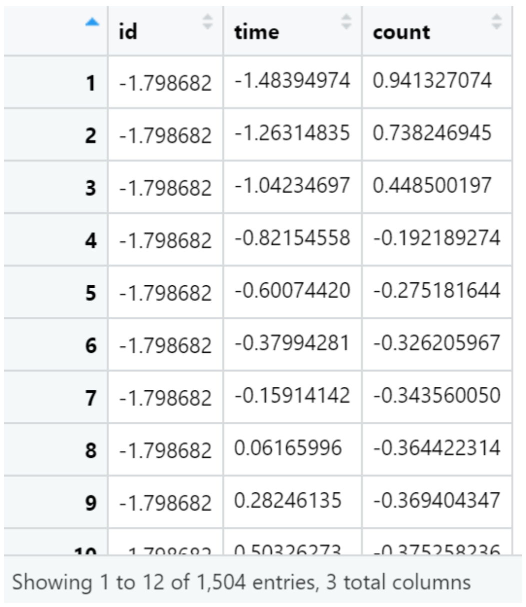
```
> install.packages("C:/Users/Moyosore/Downloads/bhrcr_1.0.3.tar.gz", repos = NULL, type = "source")
WARNING: Rtools is required to build R packages but is not currently installed. Please download and
 install the appropriate version of Rtools before proceeding:

https://cran.rstudio.com/bin/windows/Rtools/
Installing package into ‘C:/Users/Moyosore/Documents/R/win-library/3.6’
(as ‘lib’ is unspecified)
* installing *source* package 'bhrcr' ...
** package 'bhrcr' successfully unpacked and MD5 sums checked
** using staged installation
** R
** data
** demo
** inst
** byte-compile and prepare package for lazy loading
** help
*** installing help indices
  converting help for package 'bhrcr'
    finding HTML links ... done
    bhrcr-package                           html
    calculatePCE                            html
    clearanceEstimatorBayes                 html
    diagnostics                             html
    plot.bhrcr                              html
    posterior                               html
    print.bhrcr                             html
    pursat                                  html
    pursat_covariates                       html
    summary.bhrcr                           html
** building package indices
** installing vignettes
** testing if installed package can be loaded from temporary location
*** arch - i386
*** arch - x64
** testing if installed package can be loaded from final location
*** arch - i386
*** arch - x64
** testing if installed package keeps a record of temporary installation path
* DONE (bhrcr)

> library(bhrcr)
> library(kohonen)
>
> data(pursat)
```

The main function of the bhrcr package is clearanceEstimatorBayes, a function that returns the WWARN (Worldwide Antimalarial Resistance Network), PCE estimates as well as the estimates from the Bayesian hierarchical model.

The kohonen package function is to train self-organizing maps (SOMs). Also, it allows for interrogation of the maps and supports prediction using trained maps.

The Pursat data consists of Plasmodium falciparum clearance profiles of 110 patients, along with individual level covariates, measured in 2009 and 2010 in Pursat province of Western Cambodia. Parasite densities were measured every 6 hours. The parasites were divided into two genetically different groups, labeled group 1 and group 2. All 110 individuals were observed until no parasites were detected in their blood.

```
> View(pursat)
```



	id	time	count
1	-1.798682	-1.48394974	0.941327074
2	-1.798682	-1.26314835	0.738246945
3	-1.798682	-1.04234697	0.448500197
4	-1.798682	-0.82154558	-0.192189274
5	-1.798682	-0.60074420	-0.275181644
6	-1.798682	-0.37994281	-0.326205967
7	-1.798682	-0.15914142	-0.343560050
8	-1.798682	0.06165996	-0.364422314
9	-1.798682	0.28246135	-0.369404347
10	-1.798682	0.50326273	-0.375258236

Showing 1 to 12 of 1,504 entries, 3 total columns

```
> dim(pursat)
[1] 1504    3
```

```
> str(pursat)
'data.frame':   1504 obs. of  3 variables:
 $ id   : int  1 1 1 1 1 1 1 1 1 1 ...
 $ time : int  0 6 12 18 24 30 36 42 48 54 ...
 $ count: int  63662 53879 39921 9057 5059 2601 1765 760 520 238 ...
```

```
> summary(pursat)
       id              time            count       
 Min.   :  1.00   Min.   :  0.00   Min.   :     0  
 1st Qu.: 30.00   1st Qu.: 18.00   1st Qu.:   112  
 Median : 58.00   Median : 36.00   Median :  1352  
 Mean   : 56.53   Mean   : 40.32   Mean   : 18315  
 3rd Qu.: 82.25   3rd Qu.: 60.00   3rd Qu.: 12329  
 Max.   :110.00   Max.   :126.00   Max.   :546461  
```

There are no missing data.

```
> set.seed(123)
```

Above, is an optional user-specified number used to initialize a pseudorandom number generator, with a default value of 1234.The seed argument helps users to reproduce their results.

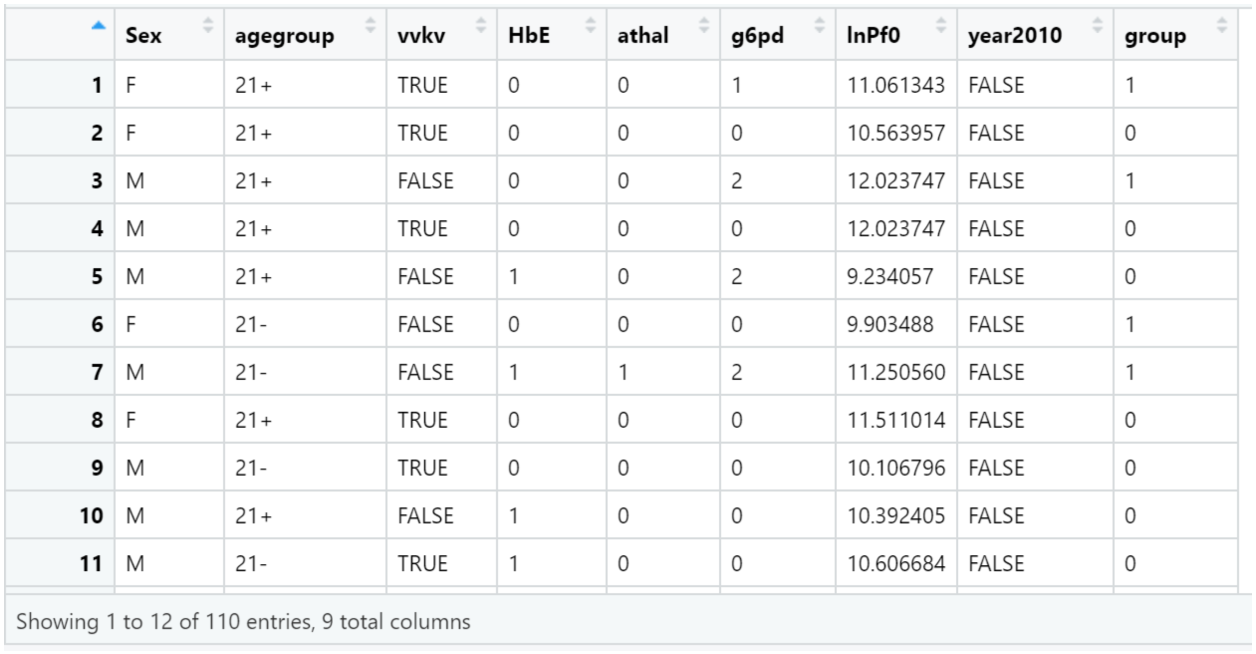
```
> data("posterior")
> plot(posterior)
[1] "all plots are saved under ./plots"
```

R Studio automatically saved the hourly time interval of the patients over log parasite.

```
> data(pursat_covariates)
```

covariates: a data frame (with no missing values), ordered according to patients' order in data and contains individual level covariates. Additionally, it concludes that the covariates in the analysis can increase statistical power and improve precision of the treatment effect.

```
> View(pursat_covariates)
```



	Sex	agegroup	vvkv	HbE	athal	g6pd	lnPf0	year2010	group
1	F	21+	TRUE	0	0	1	11.061343	FALSE	1
2	F	21+	TRUE	0	0	0	10.563957	FALSE	0
3	M	21+	FALSE	0	0	2	12.023747	FALSE	1
4	M	21+	TRUE	0	0	0	12.023747	FALSE	0
5	M	21+	FALSE	1	0	2	9.234057	FALSE	0
6	F	21-	FALSE	0	0	0	9.903488	FALSE	1
7	M	21-	FALSE	1	1	2	11.250560	FALSE	1
8	F	21+	TRUE	0	0	0	11.511014	FALSE	0
9	M	21-	TRUE	0	0	0	10.106796	FALSE	0
10	M	21+	FALSE	1	0	0	10.392405	FALSE	0
11	M	21-	TRUE	1	0	0	10.606684	FALSE	0

Showing 1 to 12 of 110 entries, 9 total columns

The clearanceEstimatorBayes function is the principal function in the bhrcr package that analyzes the input data set in the Bayesian framework.

```
> results <- clearanceEstimatorBayes(data = pursat,covariates = pursat_covariates,detect.limit = 1
5, burnin=50, niteration=100, thin=10)
[1] "Progress starts. It may take a while..."
[1] "Calculating WWARN PCE Estimates..."
[1] "Conducting Bayesian Analysis..."
[1] "Progress: 50 out of 150"
[1] "Progress: 100 out of 150"
[1] "Progress: 150 out of 150"
```

We want to test the hypothesis that red blood cell polymorphisms—including Hemoglobin E (HbE), thalassemia (athal), and G6PD deficiency (g6pd)—may act to strengthen the pro-oxidant activity of parasite defenses against artemisinin, hence resulting in lower clearance rates.

```
> summary(results)

clearanceEstimatorBayes(data = pursat, covariates = pursat_covariates,
    detect.limit = 15, niteration = 100, burnin = 50, thin = 10)

Posterior Estimates and Intervals for the Effect of Covariates on log Clearance Rates

               Mean  Median  CI 2.5% CI 97.5%
(Intercept)  -1.5036 -1.6151 -2.1281  -0.6761
SexM         -0.1648 -0.1508 -0.3060  -0.0755
agegroup21+   0.0002 -0.0163 -0.0866   0.0674
vvkvTRUE      0.0227  0.0295 -0.0567   0.0985
HbE          -0.0898 -0.0961 -0.2017   0.0201
athal         0.0348  0.0608 -0.1307   0.1263
g6pd          0.0168  0.0222 -0.0579   0.0814
lnPf0        -0.0356 -0.0175 -0.1162   0.0140
year2010TRUE -0.0465 -0.0488 -0.1213   0.0306
group        -0.1532 -0.1522 -0.2418  -0.0734
---
Posterior Estimates and Intervals for the Effect of Covariates on log half-lives

               Mean  Median  CI 2.5% CI 97.5%
(Intercept)   1.1371  1.2486  0.3096   1.7616
SexM          0.1648  0.1508  0.0755   0.3060
agegroup21+  -0.0002  0.0163 -0.0674   0.0866
vvkvTRUE     -0.0227 -0.0295 -0.0985   0.0567
HbE           0.0898  0.0961 -0.0201   0.2017
athal        -0.0348 -0.0608 -0.1263   0.1307
g6pd         -0.0168 -0.0222 -0.0814   0.0579
lnPf0         0.0356  0.0175 -0.0140   0.1162
year2010TRUE  0.0465  0.0488 -0.0306   0.1213
group         0.1532  0.1522  0.0734   0.2418
---
Detect Limit:  15
```

Data is allowed to have the predicted WWARN PCE estimates stored in another column named Predicted.

detect.limit: detection limit of the parasite density in blood (parasites per microliter). The default value is 40.

burnin: length of the burn-in period. The default value is 500.

niteration: total number of simulations after the burn-in period, with a default value of 100,000.

thin: step size of the thinning process. The default value is 50.

The summary function allows us to perform an analysis of the covariates of interest. One point of interest was whether there is evidence of resistance to artemisinin developing over time. Thus, the indicator variable year2010TRUE for the year of data collection was included. According to the results produced, the parasite clearance half-life increased over time (positive mean and median) however this effect is not significant since its 95% credible interval contains zero.

```
> diagnostics(results)
[1] "all diagnostic plots are saved under ./mcmcDiagnostics"
```

```
> plot(results)
[1] "all plots are saved under ./plots"
```

From the results of the summary none of these factors has a significant positive impact on log half-lives since the 95% credible intervals all contain 0.We therefore reject our earlier hypothesis that red blood cell polymorphisms—including Hemoglobin E (HbE), thalassemia (athal), and G6PD deficiency (g6pd)—may act to strengthen the pro-oxidant activity of parasite defenses against artemisinin, hence resulting in lower clearance rates.

outlier.detect: indicator of whether or not to use the WWARN PCE outlier detection method .The default value is TRUE and it is recommended to set outlier.detect = TRUE if data is missing the Predicted column.

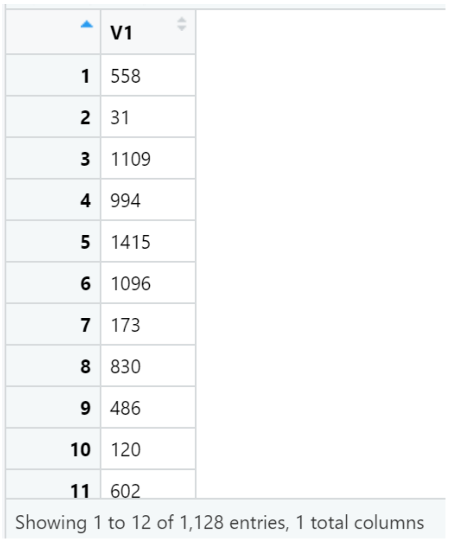
conf.level: required confidence level for reporting estimates' credible intervals, with a default value of 0.95.

We can calculate the posterior mean, median, and 95% credible interval of everyone's clearance rate.

```
> id <- c(2, 4, 15, 33)
> a <- .025

> results$clearance.mean[id]
        2         4        15        33 
0.1380624 0.2539274 0.1219033 0.1498263 
> results$clearance.median[id]
        2         4        15        33 
0.1386563 0.2542850 0.1239022 0.1475840 
```

Create the SOM Model

```
> pursat<-scale(pursat)
> smp_siz = floor(0.75*nrow(pursat))
> smp_siz
[1] 1128

> train_ind = sample(seq_len(nrow(pursat)),size = smp_siz)

> View(train_ind)
```

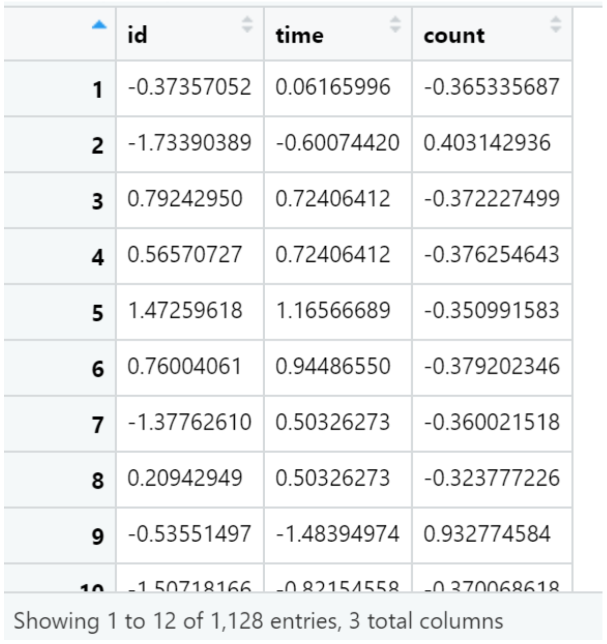


	V1
1	558
2	31
3	1109
4	994
5	1415
6	1096
7	173
8	830
9	486
10	120
11	602

Showing 1 to 12 of 1,128 entries, 1 total columns

Randomly identifies the rows equal to sample size from all the rows of Pursat dataset and stores the row number in train_ind.

```
> train = pursat[train_ind,]
> View(train)
```

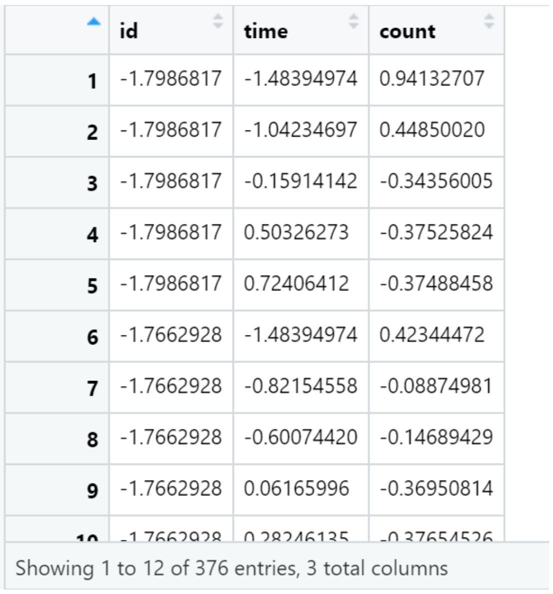


	id	time	count
1	-0.37357052	0.06165996	-0.365335687
2	-1.73390389	-0.60074420	0.403142936
3	0.79242950	0.72406412	-0.372227499
4	0.56570727	0.72406412	-0.376254643
5	1.47259618	1.16566689	-0.350991583
6	0.76004061	0.94486550	-0.379202346
7	-1.37762610	0.50326273	-0.360021518
8	0.20942949	0.50326273	-0.323777226
9	-0.53551497	-1.48394974	0.932774584
10	-1.50718166	-0.82154558	-0.370068618

Showing 1 to 12 of 1,128 entries, 3 total columns

Creates the training dataset with row numbers stored in train_ind.

```
> test= pursat[-train_ind,]
> View(test)
```



	id	time	count
1	-1.7986817	-1.48394974	0.94132707
2	-1.7986817	-1.04234697	0.44850020
3	-1.7986817	-0.15914142	-0.34356005
4	-1.7986817	0.50326273	-0.37525824
5	-1.7986817	0.72406412	-0.37488458
6	-1.7662928	-1.48394974	0.42344472
7	-1.7662928	-0.82154558	-0.08874981
8	-1.7662928	-0.60074420	-0.14689429
9	-1.7662928	0.06165996	-0.36950814
10	-1.7662928	0.28246135	-0.37654526

Showing 1 to 12 of 376 entries, 3 total columns

Creates the test dataset excluding the row numbers mentioned in train_ind.

```
> set.seed(123)
```

Create the SOM Grid - you generally have to specify the size of the training grid prior to training the SOM.

```
> som_grid <- somgrid(xdim = 20, ydim=20, topo="hexagonal")
```

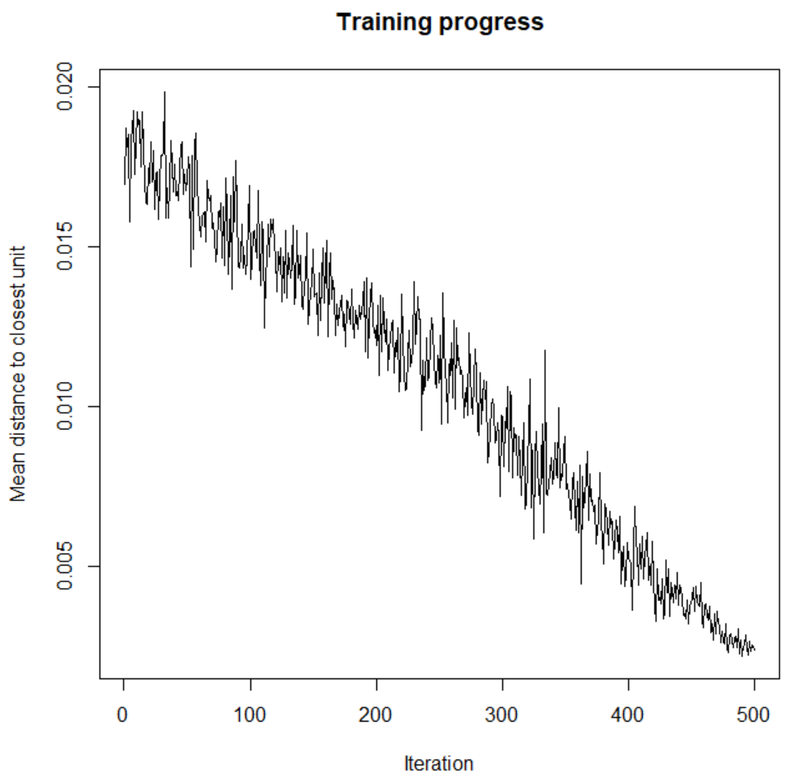
Finally, train the SOM, options for the number of iterations, the learning rates, and the neighborhood are available.

```
> set.seed(123)

> som_grid <- somgrid(xdim = 20, ydim=20, topo="hexagonal")

> som_model <- som(train,
+     grid=som_grid,
+     rlen=500,
+     alpha=c(0.05,0.01),
+         keep.data = TRUE )
```

Visualization

Training progress for SOM

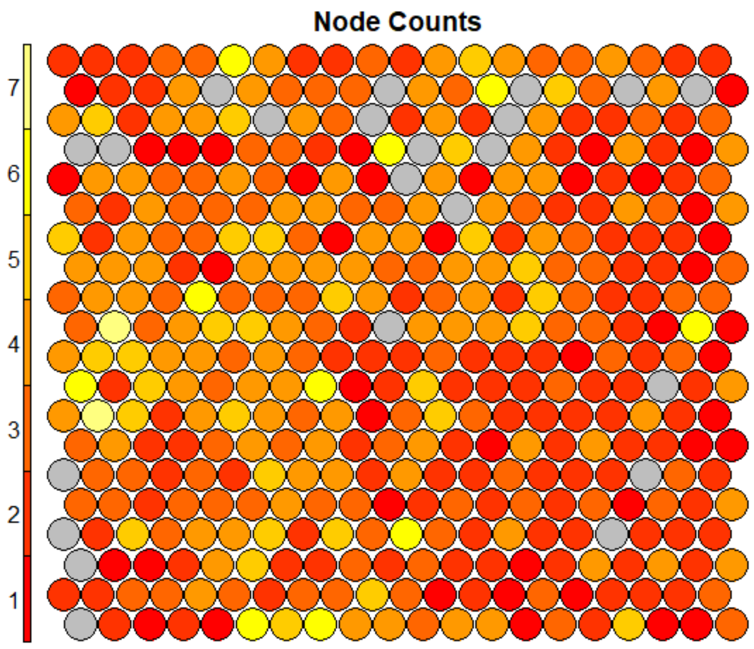
```
> set.seed(123)

> X11()
> plot(som_model, type="changes")
```



Node count plot

The Kohonen packages allows us to visualize the count of how many samples are mapped to each node on the map.

```
> set.seed(123)
> X11()
> plot(som_model, type="count", main="Node Counts")
```

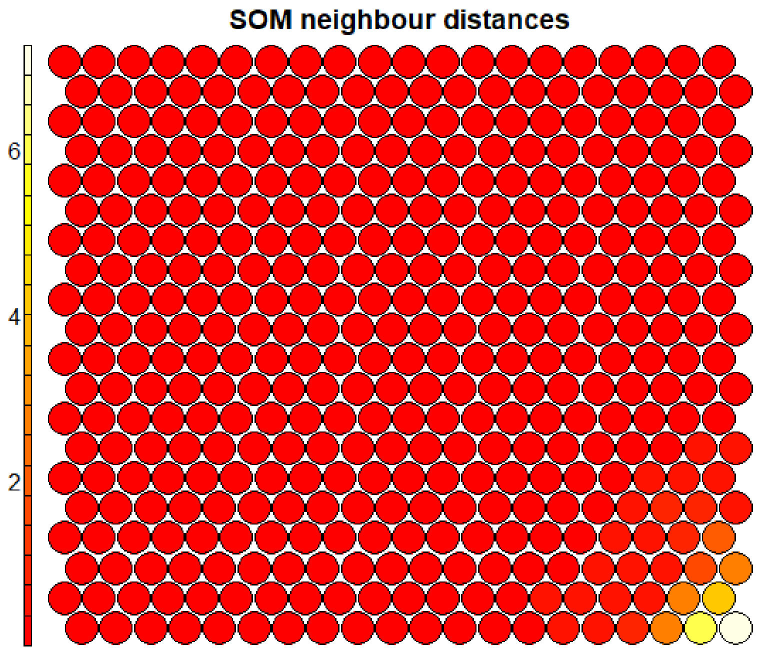


Neighbor Distance

Often referred to as the “U-Matrix”, this visualization is of the distance between each node and its neighbors. The U-Matrix can be used to identify clusters within the SOM map.

U-matrix visualization

```
> set.seed(123)

> X11()
> plot(som_model, type="dist.neighbours", main = "SOM neighbour distances")
```

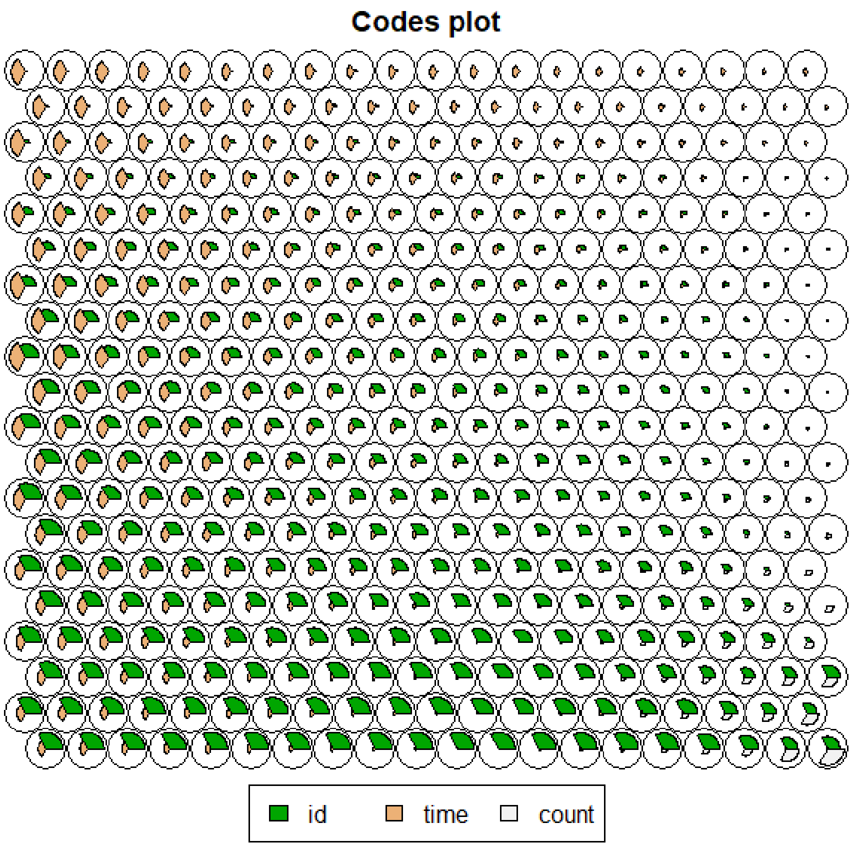


Codes / Weight vectors

The node weight vectors, or “codes”, are made up of normalized values of the original variables used to generate the SOM. Each node’s weight vector is representative / similar of the samples mapped to that node. By visualizing the weight vectors across the map, we can see patterns in the distribution of samples and variables.

Weight Vector View

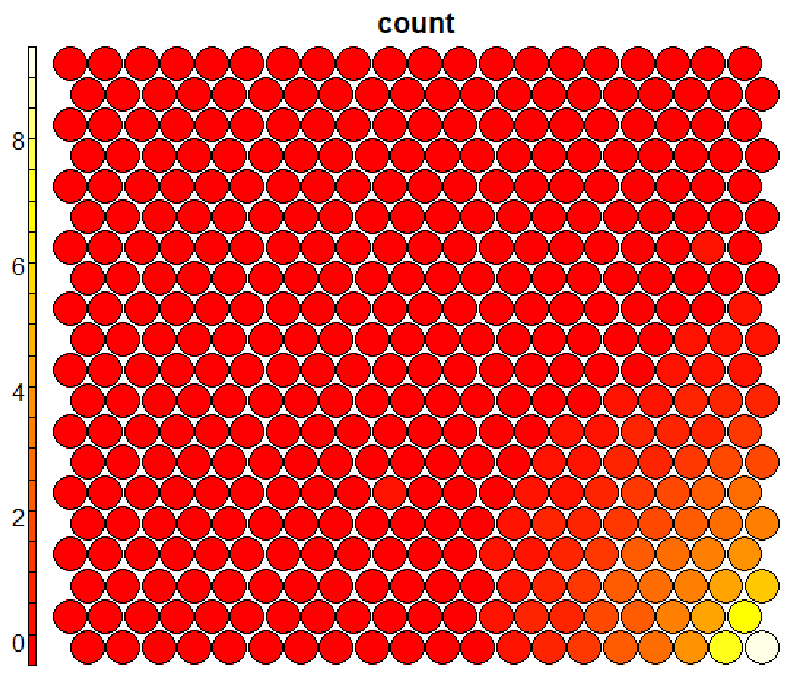
```
> set.seed(123)
> X11()
> plot(som_model, type="codes")
```



Heatmaps

Kohonen Heatmap creation

```
> set.seed(123)
> X11()
>
> plot(som_model, type = "property", property = getCodes(som_model)[,3], main=colnames(getCodes
(som_model))[3], palette(rainbow(6)))
```

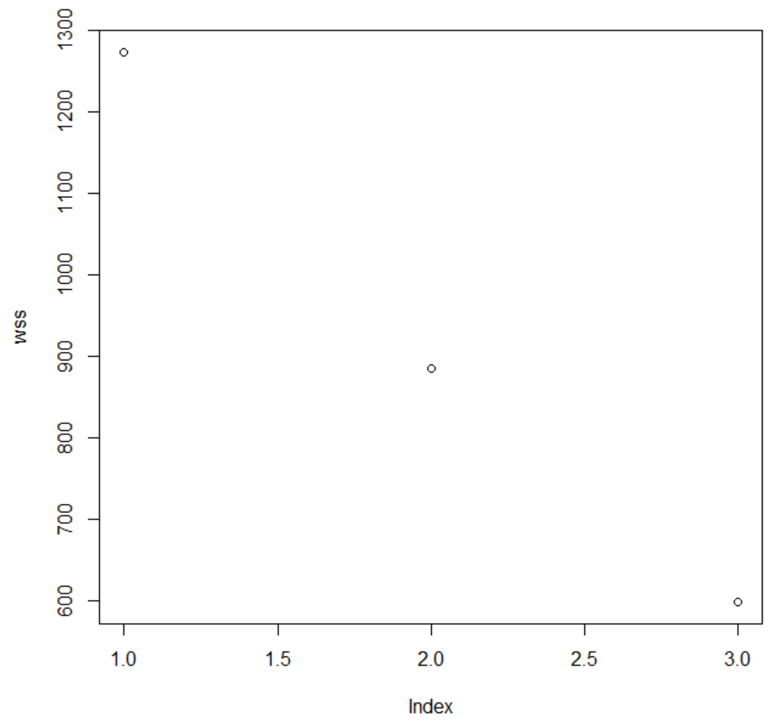


Clustering and Segmentation on top of Self-Organizing Map

Clustering can be performed on the SOM nodes to isolate groups of samples with similar metrics. The results of the clustering can be visualized using the SOM plot function again.

Viewing WCSS for kmeans

```
> set.seed(123)
> mydata <- som_model$codes
> mydata <- som_model$codes[[1]]

> wss <- (nrow(mydata)-1)*sum(apply(mydata,2,var))
> for (i in 1:3) {
+   wss[i] <- sum(kmeans(mydata, centers=i)$withinss)
+ }
>

> X11()
> plot(wss)
```



An estimate of the number of clusters that would be suitable can be ascertained using a kmeans algorithm and examining for an "elbow-point" in the plot of "within cluster sum of squares". In this case, the estimate is 2 clusters.

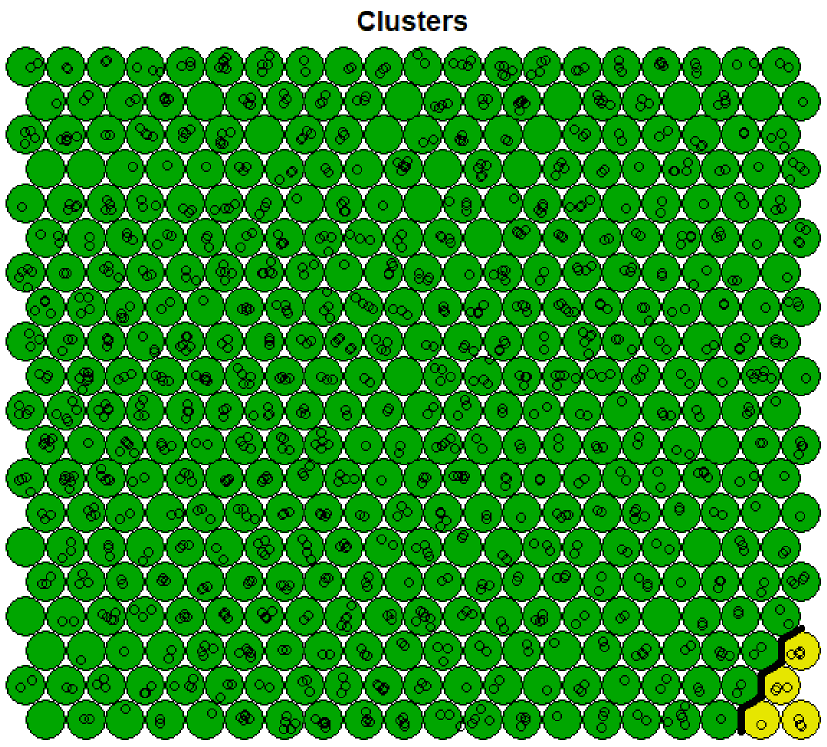
Visualizing cluster results

Use hierarchical clustering to cluster the codebook vectors

```
> set.seed(123)
> som_cluster <- cutree(hclust(dist(som_model$codes[[1]])), 2)
```

Plot these results

```
> x11()
> plot(som_model, type="mapping", bgcol = terrain.colors(5)[som_cluster], main = "Clusters")
>
> add.cluster.boundaries(som_model, som_cluster)
```



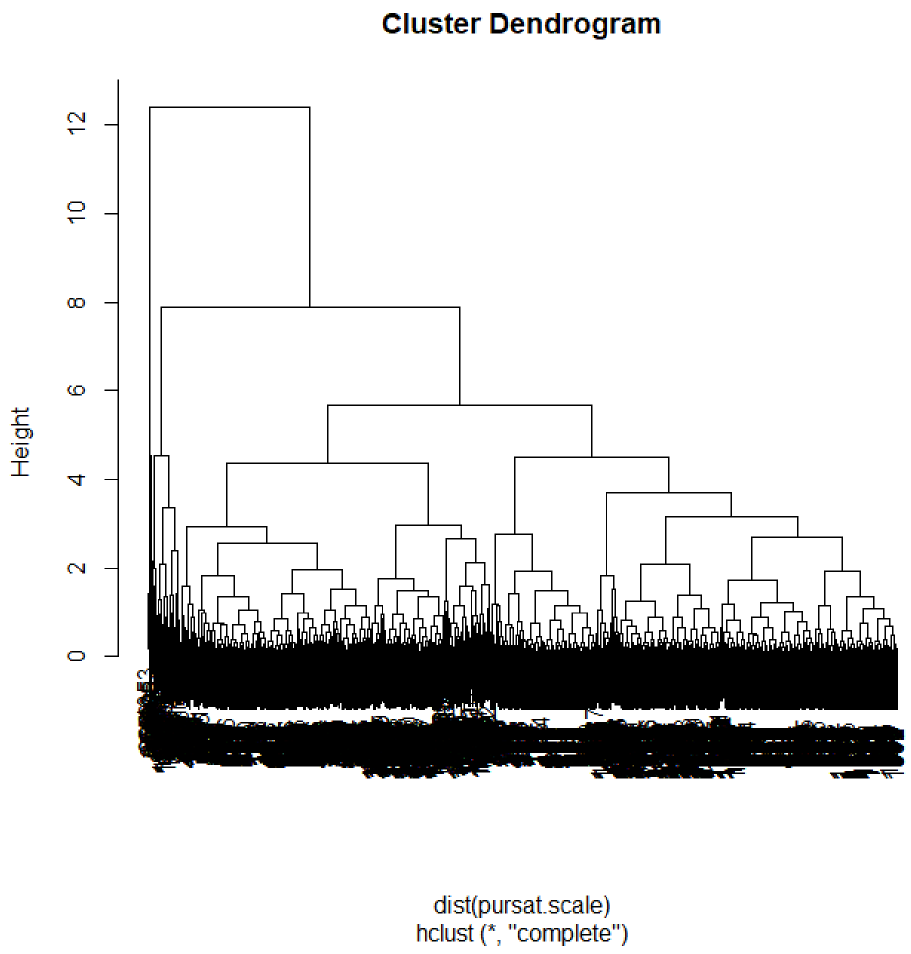
In general, the SOM has organized these data into well-defined clusters

Create hierarchical clustering model: hclust.out

```
> pursat.scale<-scale(pursat)
> hclust.out <- hclust(dist(pursat.scale))
```

Inspect the result

```
> summary(hclust.out)
            Length Class  Mode     
merge       3006   -none- numeric  
height      1503   -none- numeric  
order       1504   -none- numeric  
labels         0   -none- NULL     
method         1   -none- character
call           2   -none- call     
dist.method    1   -none- character
> X11()
> plot(hclust.out)
```

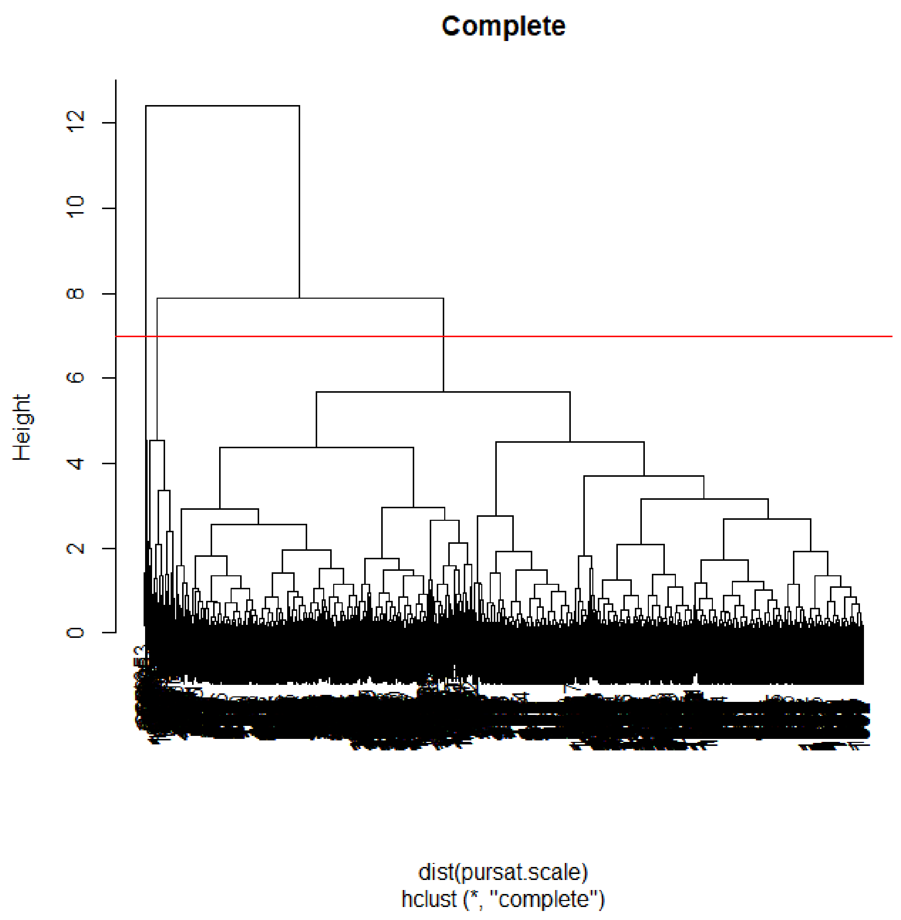


Cluster using complete linkage: hclust.complete

```
> hclust.complete <- hclust(dist(pursat.scale), method = "complete")
```

Plot dendrogram of hclust.complete

```
> X11()
> plot(hclust.complete, main = "Complete")
> abline(h = 7, col = "red")
```

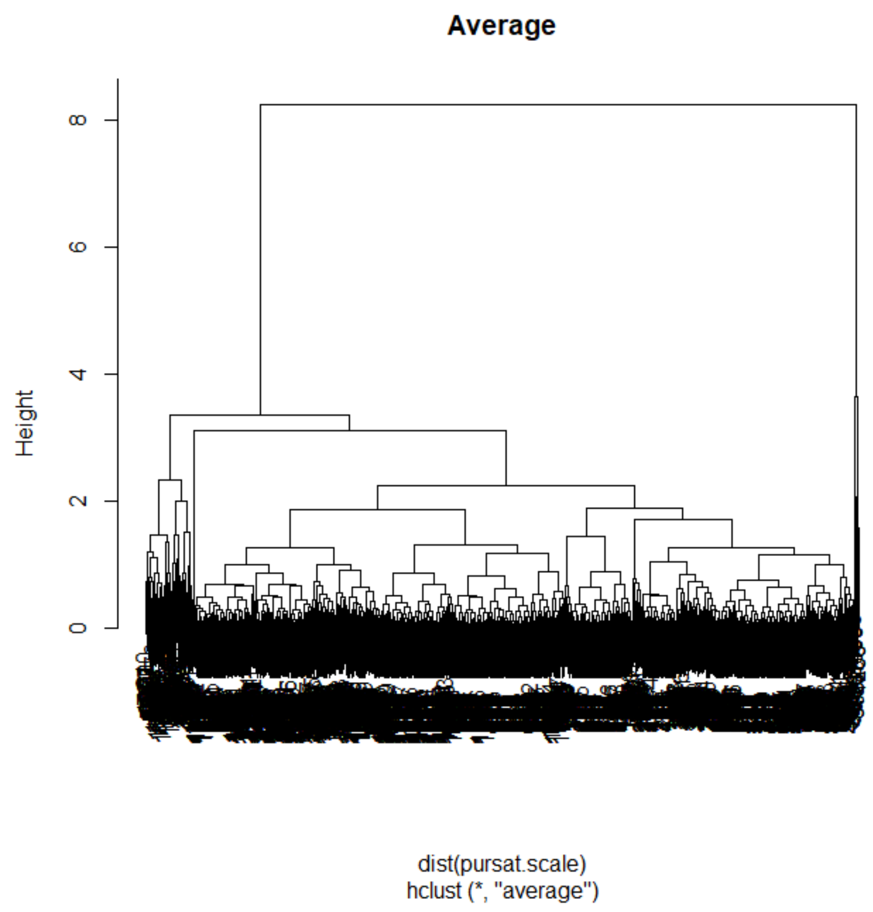


Cluster using average linkage: hclust.average

```
> hclust.average <- hclust(dist(pursat.scale), method = "average")
```

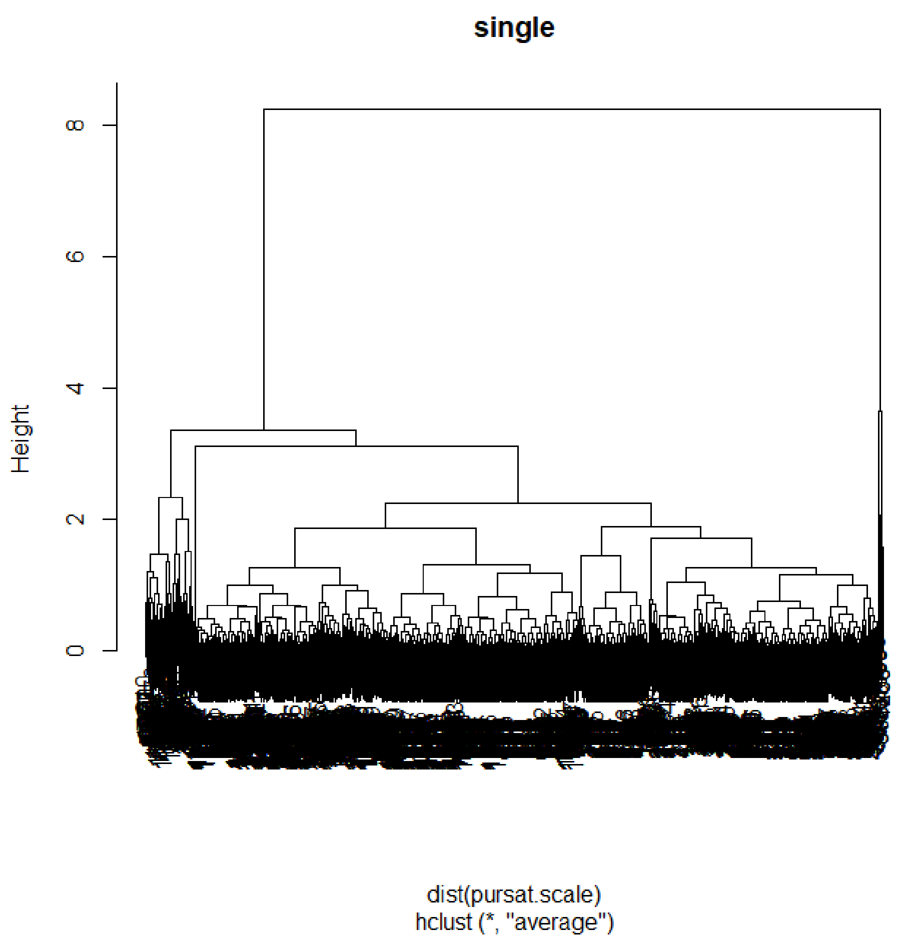
Plot dendrogram of hclust.average

```
> X11()
> plot(hclust.average, main = "Average")
```



Cluster using single linkage: hclust.single

```
> hclust.single <- hclust(dist(pursat.scale), method = "single")
> X11()
> plot(hclust.average, main = "single")
```



Apply cutree() to pursat.scale: cut.pursat

```
cut.pursat <- cutree(hclust.out, k =3)
```

Create the k-means model: km.out

```
> km.out <- kmeans(pursat, centers = 3, nstart = 20)
```

Inspect the result

```
> summary(km.out)
             Length Class  Mode
cluster      1504   -none- numeric
centers         9   -none- numeric
totss           1   -none- numeric
withinss        3   -none- numeric
tot.withinss    1   -none- numeric
betweenss       1   -none- numeric
size            3   -none- numeric
iter            1   -none- numeric
ifault          1   -none- numeric
```

Initialize total within sum of squares error: wss

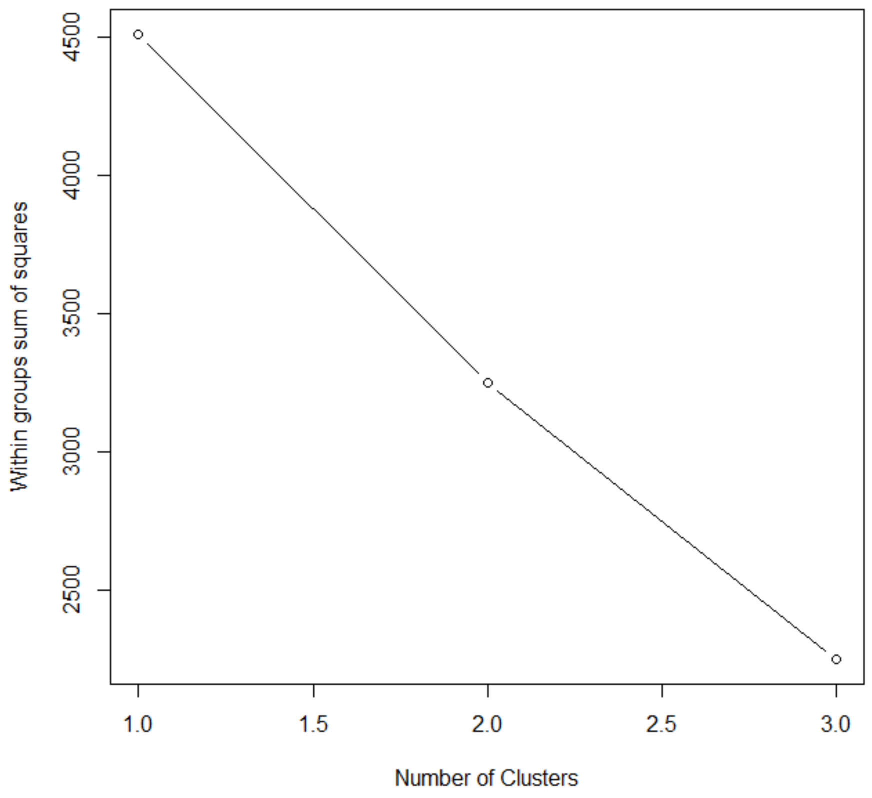
```
> wss <- 0
```

Look over 1 to 15 possible clusters

```
> for (i in 1:3) {
+   # Fit the model: km.out
+   km.out <- kmeans(pursat, centers = i, nstart = 20, iter.max = 50)
+   # Save the within cluster sum of squares
+   wss[i] <- km.out$tot.withinss
+ }
>
```

Produce a screen plot

```
> X11()
> plot(1:3, wss, type = "b",
+      xlab = "Number of Clusters",
+      ylab = "Within groups sum of squares")
>
```



Print the cluster membership component of the model

```
> km.out$cluster
   [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 2 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3
  [44] 3 3 2 2 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 2 3 3 3 3 3 3
  [87] 3 3 3 3 3 2 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
 [130] 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 2 3 3 3 3 3 3 3 3
 [173] 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 3 3 3 3 3 3
 [216] 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 2 3 3 3 3 3 3 3 3 3 3 3 3
 [259] 3 3 3 2 2 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 3
 [302] 3 3 3 3 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2 2 3 3 3 3 3 3 3 3 3 3 3 3 2 3 3 3 3
 [345] 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 3 3 3 3 3 3
 [388] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
 [431] 3 3 3 3 3 3 3 3 3 3 2 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
 [474] 2 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 1 3 3 3 3 3 3 3 3 3 3 3 2 3 3 3 3
 [517] 3 3 3 3 3 3 3 3 1 1 1 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 1 1 3 3 3 3 3 3 3 3 3
 [560] 3 3 3 1 1 1 1 3 3 3 3 3 3 3 3 3 3 3 1 3 3 3 3 3 3 3 3 3 3 3 1 1 1 1 1 1 2 2 3 3 3 3 3
 [603] 3 3 3 1 1 1 3 3 3 3 3 3 3 3 3 3 1 1 1 1 1 1 3 3 3 3 3 3 3 3 3 1 1 1 3 3 3 3 3 3 3 3 3
 [646] 1 1 1 1 1 1 1 2 3 3 3 3 3 3 3 1 1 1 1 1 1 1 1 1 2 2 3 3 3 3 3 3 1 1 1 1 1 1 1 1 1 2 2
 [689] 3 3 3 3 3 1 1 1 1 1 1 1 1 1 3 3 3 3 3 3 3 1 1 1 1 1 1 2 2 2 3 3 3 3 1 1 1 1 1 1 3 2 3
 [732] 2 3 3 1 1 1 1 1 1 3 2 3 3 3 3 1 1 1 1 1 1 1 1 1 1 1 2 2 3 3 3 1 1 1 1 1 2 2 2 3 3 1 1
 [775] 1 1 1 1 1 1 1 1 1 1 1 1 1 3 3 3 3 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 1 1 1 1 1 1 1 1
 [818] 1 1 1 2 2 2 3 1 1 1 1 1 1 1 1 1 3 3 3 1 1 1 1 1 1 1 1 1 1 1 1 3 2 1 1 1 1 1 1 1 1 1 1
 [861] 1 1 1 2 3 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1
 [904] 1 1 1 1 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 [947] 1 1 1 1 1 1 2 2 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 [990] 1 1 1 1 1 1 1 1 1 2 2
 [ reached getOption("max.print") -- omitted 504 entries ]
```

Print the km.out object

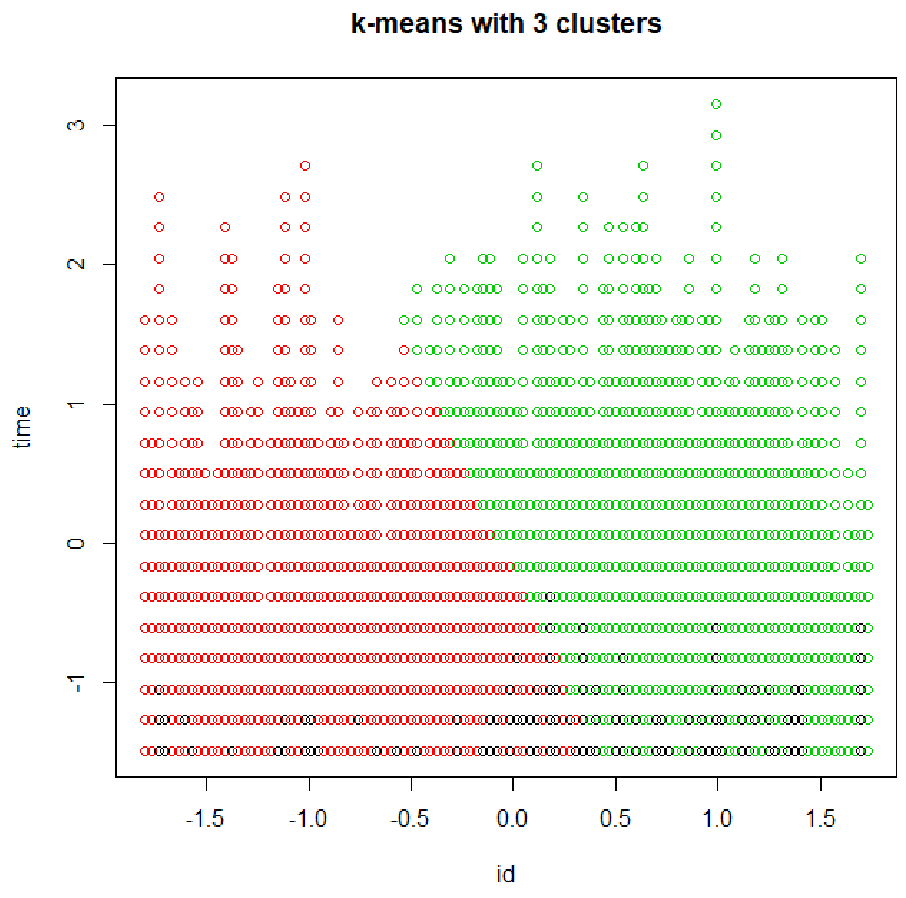
```
> km.out
```

```
K-means clustering with 3 clusters of sizes 751, 106, 647

Cluster means:
          id       time      count
1  0.7619814  0.3374412 -0.2472535
2  0.3001795 -1.2652314  2.9133735
3 -0.9336430 -0.1843954 -0.1903094

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 2 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3
  [44] 3 3 2 2 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 2 3 3 3 3 3 3
  [87] 3 3 3 3 3 2 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
 [130] 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 2 3 3 3 3 3 3 3 3
 [173] 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 3 3 3 3 3 3
 [216] 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 2 3 3 3 3 3 3 3 3 3 3 3 3
 [259] 3 3 3 2 2 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 3
 [302] 3 3 3 3 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 2 3 3 3
 [345] 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 3 3 3 3 3 3
 [388] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
 [431] 3 3 3 3 3 3 3 3 3 3 2 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
 [474] 2 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 1 3 3 3 3 3 3 3 3 3 3 3 2 3 3 3 3
 [517] 3 3 3 3 3 3 3 3 1 1 1 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 1 1 3 3 3 3 3 3 3 3 3
 [560] 3 3 3 1 1 1 1 3 3 3 3 3 3 3 3 3 3 3 1 3 3 3 3 3 3 3 3 3 3 3 1 1 1 1 1 1 2 2 3 3 3 3 3
 [603] 3 3 3 1 1 1 3 3 3 3 3 3 3 3 3 3 1 1 1 1 1 1 3 3 3 3 3 3 3 3 3 1 1 1 3 3 3 3 3 3 3 3 3
 [646] 1 1 1 1 1 1 1 2 3 3 3 3 3 3 3 1 1 1 1 1 1 1 1 1 2 2 3 3 3 3 3 3 1 1 1 1 1 1 1 1 1 2 2
 [689] 3 3 3 3 3 1 1 1 1 1 1 1 1 1 3 3 3 3 3 3 3 1 1 1 1 1 1 2 2 2 3 3 3 3 1 1 1 1 1 1 3 2 3
 [732] 2 3 3 1 1 1 1 1 1 3 2 3 3 3 3 1 1 1 1 1 1 1 1 1 1 1 2 2 3 3 3 1 1 1 1 1 2 2 2 3 3 1 1
 [775] 1 1 1 1 1 1 1 1 1 1 1 1 1 3 3 3 3 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 1 1 1 1 1 1 1 1
 [818] 1 1 1 2 2 2 3 1 1 1 1 1 1 1 1 1 3 3 3 1 1 1 1 1 1 1 1 1 1 1 1 3 2 1 1 1 1 1 1 1 1 1 1
 [861] 1 1 1 2 3 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1
 [904] 1 1 1 1 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 [947] 1 1 1 1 1 1 2 2 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 [990] 1 1 1 1 1 1 1 1 1 2 2
 [ reached getOption("max.print") -- omitted 504 entries ]

Within cluster sum of squares by cluster:
[1] 983.2131 505.3058 764.6601
 (between_SS / total_SS =  50.0 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss" "betweenss"
[7] "size"         "iter"         "ifault"
```

Visualizing and interpreting results of kmeans()

Scatter plot of Pursat

```
> X11()
> plot(pursat,
+   col = km.out$cluster,
+   main = "k-means with 3 clusters")
>
```



# Set k equal to the number of clusters corresponding to the elbow location

```
> k <- 2
```

# Build model with k clusters: km.out

```
> km.pursat<- kmeans(pursat, centers = k, nstart = 20, iter.max = 50)
```

View the resulting model

```
> km.pursat
K-means clustering with 2 clusters of sizes 666, 838

Cluster means:
           id       time      count
1 -0.02818630 -0.9246525  0.4440217
2  0.02240104  0.7348671 -0.3528860

Clustering vector:
   [1] 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 2 2 2 2 2 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2
  [44] 2 2 1 1 1 1 1 1 2 2 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 2 2 2 2 1 1 1 1 1 1 2 2
  [87] 2 2 2 2 2 1 1 1 1 1 1 2 2 2 2 2 2 1 1 1 1 1 1 2 2 2 2 2 2 2 1 1 1 1 1 1 2 2 2 2 1 1 1
 [130] 1 1 1 2 2 2 1 1 1 1 1 1 2 2 2 2 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 2 2 2
 [173] 2 2 2 2 2 2 2 2 1 1 1 1 1 1 2 2 2 2 2 2 2 2 1 1 1 1 1 1 2 2 2 2 1 1 1 1 1 1 2 2 2 2 2
 [216] 2 1 1 1 1 1 1 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 1 1 1 1 1 1 2 2 2 2 2 2 2
 [259] 2 2 2 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 2 2 2 2 2 2 2 1 1 1 1 1 1 2 2
 [302] 2 2 2 2 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 1 1 1 1
 [345] 1 1 2 2 2 2 2 1 1 1 1 1 1 2 2 2 2 1 1 1 1 1 1 2 2 2 2 2 2 1 1 1 1 1 1 2 2 2 2 2 2 2 2
 [388] 2 1 1 1 1 1 1 2 2 2 2 2 1 1 1 1 1 1 2 2 1 1 1 1 1 1 2 2 2 2 2 2 1 1 1 1 1 1 2 2 2 1 1
 [431] 1 1 1 1 2 2 2 2 2 2 1 1 1 1 1 1 2 2 2 2 2 2 2 1 1 1 1 1 1 2 1 1 1 1 1 1 2 2 2 2 2 2 2
 [474] 1 1 1 1 1 1 2 2 2 2 2 2 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 2 2 2 2 2 1 1 1 1 1
 [517] 1 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 2 2 2 1 1 1 1 1 1 2 2 2 2 2 2 2 2 1 1 1 1 1 1 2 2 2
 [560] 2 2 2 2 2 2 2 1 1 1 1 1 1 2 2 2 2 2 2 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 2
 [603] 2 2 2 2 2 2 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 2 2 2 2 2 2 1 1 1 1 1 1 2 2 2
 [646] 2 2 2 2 2 2 2 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 1 1
 [689] 1 1 1 1 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 2 2 2 2 2 2 2 1 1 1 1 1 1 2 2 2 2 2 2 2 1 1 1
 [732] 1 1 1 2 2 2 2 2 2 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 2 2 2 2 1 1 1 1 1 1 1
 [775] 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 2 2 2 2 2 2 2
 [818] 2 2 2 1 1 1 1 1 1 2 2 2 2 2 2 2 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 2 2 2 2 2 2
 [861] 2 2 2 1 1 1 1 1 1 2 2 2 2 2 2 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 2 2 2
 [904] 2 2 2 2 1 1 1 1 1 1 2 2 2 2 2 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 2 2 2 2 2
 [947] 2 2 2 2 2 2 1 1 1 1 1 1 2 2 2 2 2 2 2 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1
 [990] 2 2 2 2 2 2 2 2 2 1 1
 [ reached getOption("max.print") -- omitted 504 entries ]

Within cluster sum of squares by cluster:
[1] 2066.218 1184.209
 (between_SS / total_SS =  27.9 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss" "betweenss"
[7] "size"         "iter"         "ifault"
```

Compare methods

```
> table(km.pursat$cluster,cut.pursat)
   cut.pursat
       1   2   3
  1  600  58   8
  2  838   0   0
```

The scaled hierarchical cluster pretty much puts all observations in cluster 1