Clustering

Data Sets

Attrition

```
attrition <- attrition %>% mutate_if(is.ordered, factor, order = F)

churn <- initial_split(attrition, prop = .7, strata = "Attrition")

churn_train <- training(churn)
churn_test <- testing(churn)

rm(churn)
```

Ames, Iowa housing data.

```
ames <- AmesHousing::make_ames()

set.seed(123)

ames_split <- initial_split(ames, prop =.7, strata = "Sale_Price")

ames_train <- training(ames_split)
ames_test <- testing(ames_split)

rm(ames_split)

response <- "Sale_Price"
predictors <- setdiff(colnames(ames_train), response)
```

Market Basket:

```
url <- "https://koalaverse.github.io/homlr/data/my_basket.csv"

my_basket <- readr::read_csv(url)
```

```
Parsed with column specification:
cols(
  .default = col_double()
)
```

```
See spec(...) for full column specifications.
```

```
dim(my_basket)
```

```
[1] 2000   42
```

MNIST:

```
mnist <- dslabs::read_mnist()
```

```
data(geyser, package = 'MASS')
```

K-means Clustering

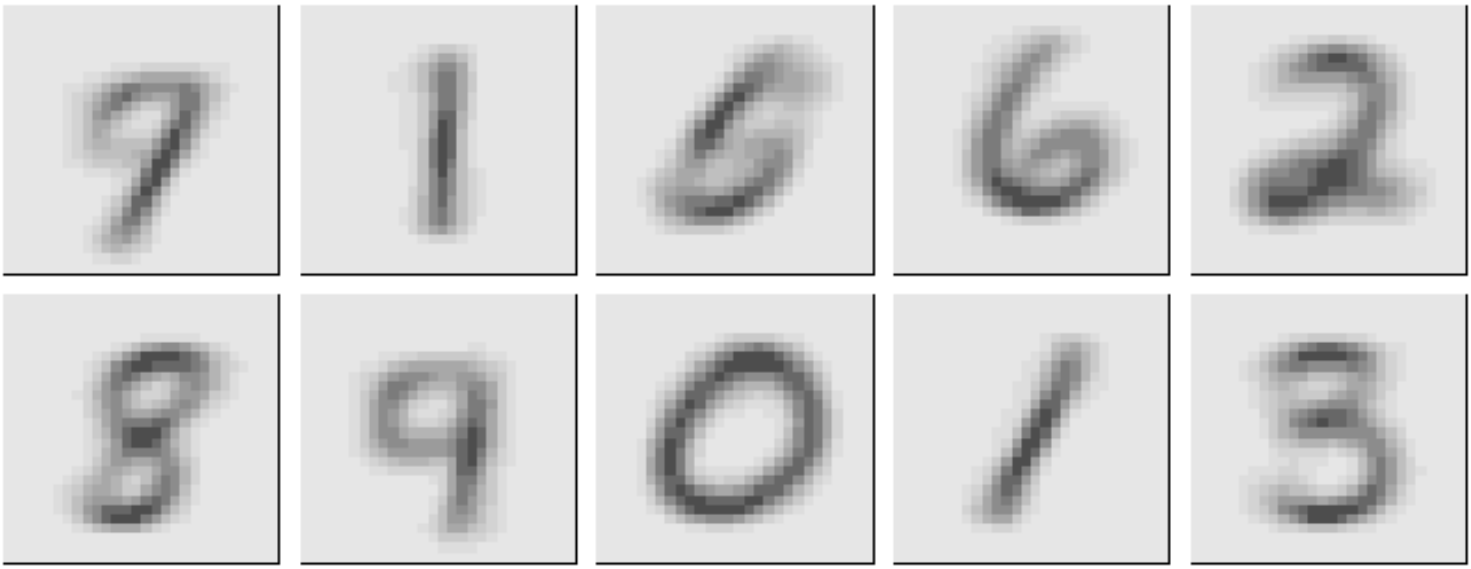
```
features <- mnist$train$images

# Use k-means model with 10 centers and 10 random starts
mnist_clustering <- kmeans(features, centers = 10, nstart = 10)
```

```
Warning: Quick-TRANSfer stage steps exceeded maximum (= 3000000)

Warning: Quick-TRANSfer stage steps exceeded maximum (= 3000000)

Warning: Quick-TRANSfer stage steps exceeded maximum (= 3000000)
```

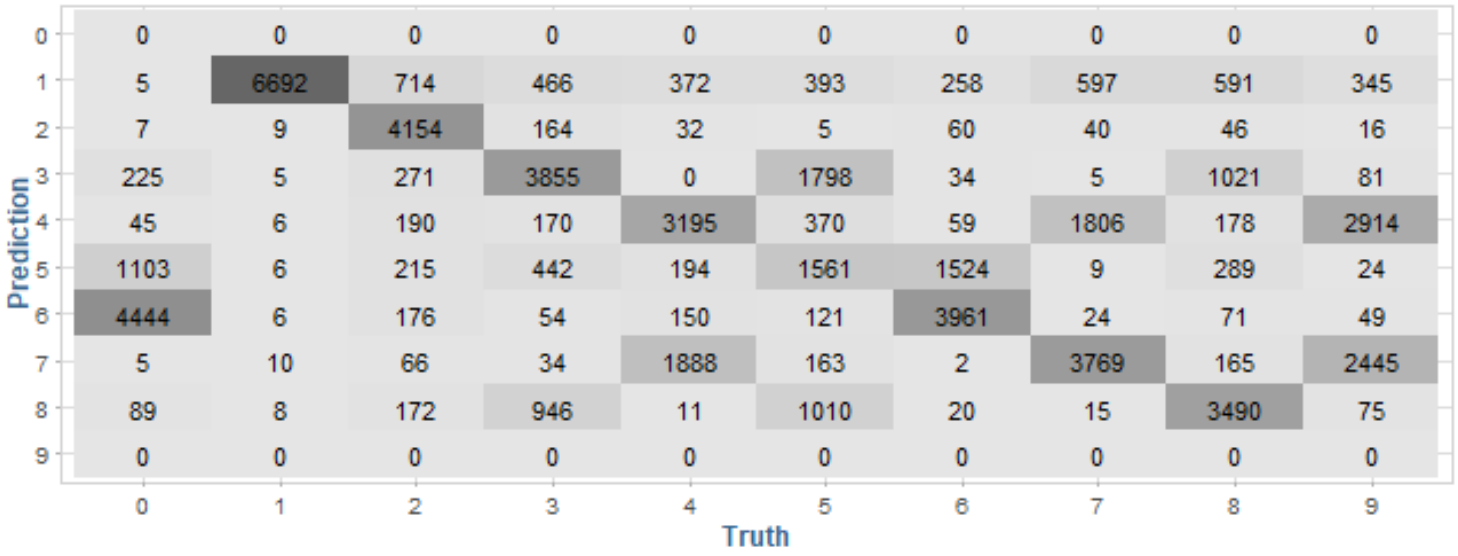
```
# Print contents of the model output
str(mnist_clustering)
```

```
List of 9
 $ cluster     : int [1:60000] 2 6 4 8 1 9 3 2 3 1 ...
 $ centers     : num [1:10, 1:784] 0 0 0 0 0 0 0 0 0 0 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:10] "1" "2" "3" "4" ...
  .. ..$ : NULL
 $ totss       : num 205706725984
 $ withinss    : num [1:10] 19548357874 15947348981 9086223869 24096706393 14910446731 ...
 $ tot.withinss: num 153001872957
 $ betweenss   : num 52704853027
 $ size        : int [1:10] 8547 5836 5929 8933 5367 4549 4507 4504 4533 7295
 $ iter        : int 10
 $ ifault      : int 0
 - attr(*, "class")= chr "kmeans"
```

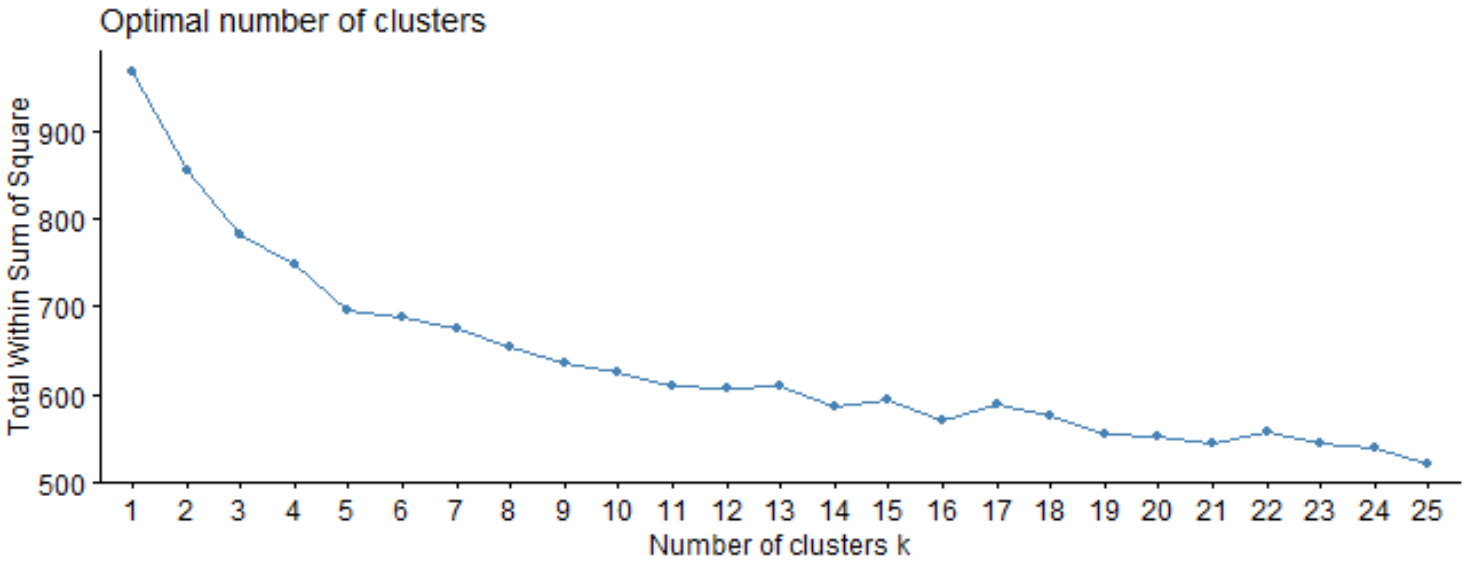
```
# Extract cluster centers
mnist_centers <- mnist_clustering$centers

# Plot typical cluster digits
par(mfrow = c(2, 5), mar=c(0.5, 0.5, 0.5, 0.5))
layout(matrix(seq_len(nrow(mnist_centers)), 2, 5, byrow = FALSE))
for(i in seq_len(nrow(mnist_centers))) {
  image(matrix(mnist_centers[i, ], 28, 28)[, 28:1],
        col = gray.colors(12, rev = TRUE), xaxt="n", yaxt="n")
}
```



```
# Create mode function
mode_fun <- function(x){
  which.max(tabulate(x))
}

mnist_comparison <- data.frame(
  cluster = mnist_clustering$cluster,
  actual = mnist$train$labels
) %>%
  group_by(cluster) %>%
  mutate(mode = mode_fun(actual)) %>%
  ungroup() %>%
  mutate_all(factor, levels = 0:9)

# Create confusion matrix and plot results
yardstick::conf_mat(
  mnist_comparison,
  truth = actual,
  estimate = mode
) %>%
  autoplot(type = 'heatmap')
```




```
fviz_nbclust(
  my_basket,
  kmeans,
  k.max = 25,
  method = "wss",
  diss = get_dist(my_basket, method = "spearman")
)
```

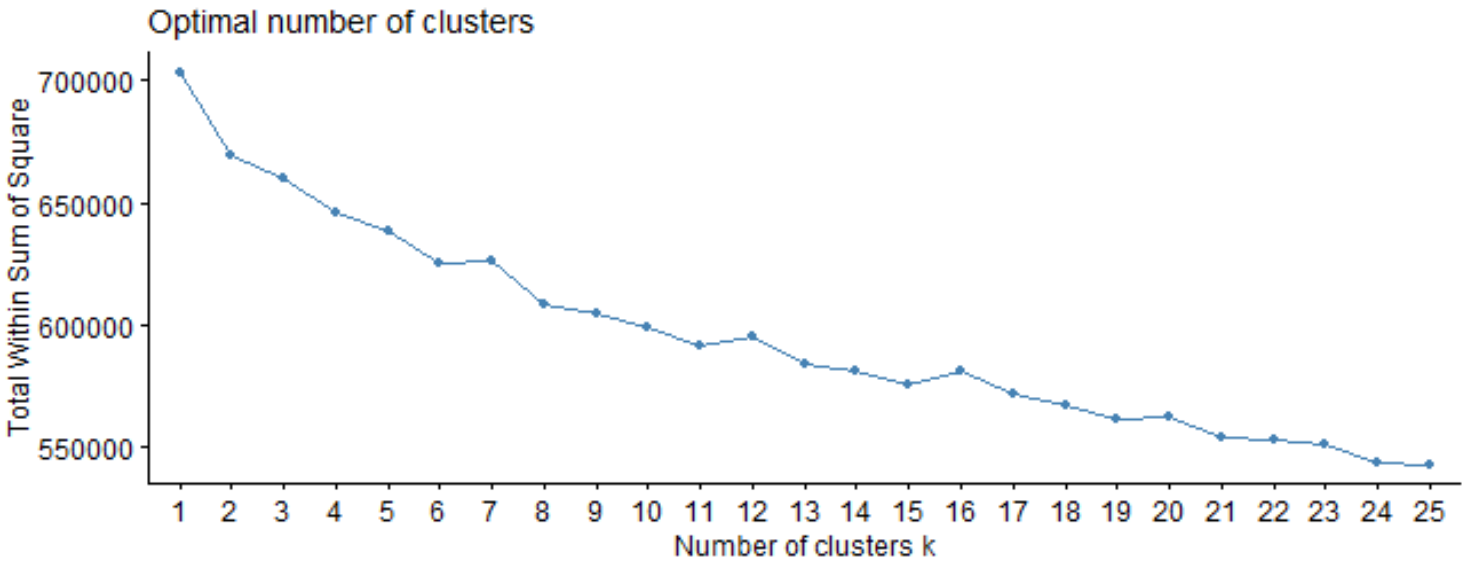


Mixed Data

```
# Full ames data set --> recode ordinal variables to numeric
ames_full <- AmesHousing::make_ames() %>%
  mutate_if(str_detect(names(.), 'Qual|Cond|QC|Qu'), as.numeric)

# One-hot encode --> retain only the features and not sale price
full_rank  <- caret::dummyVars(Sale_Price ~ ., data = ames_full,
                               fullRank = TRUE)
ames_1hot <- predict(full_rank, ames_full)

# Scale data
ames_1hot_scaled <- scale(ames_1hot)

# New dimensions
dim(ames_1hot_scaled)
```

```
[1] 2930  240
```

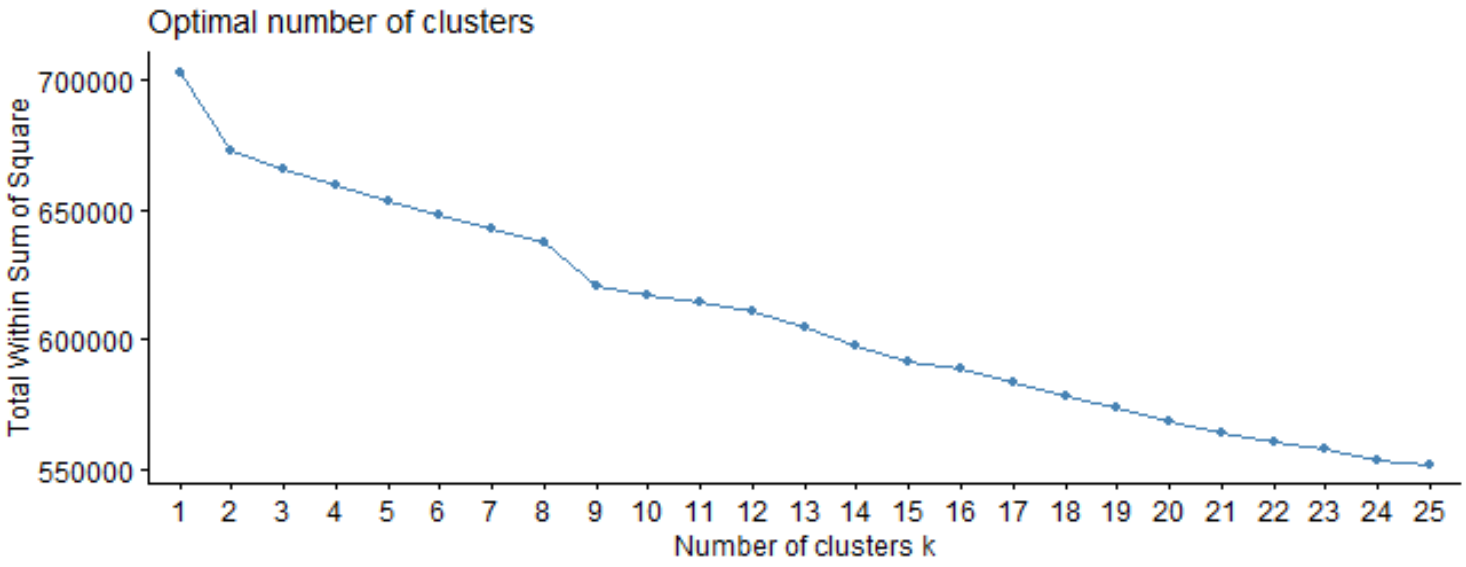
```
set.seed(123)

fviz_nbclust(
  ames_1hot_scaled,
  kmeans,
  method = "wss",
  k.max = 25,
  verbose = FALSE
)
```




```r
# Original data minus Sale_Price
ames_full <- AmesHousing::make_ames() %>% select(-Sale_Price)

# Compute Gower distance for original data
gower_dst <- daisy(ames_full, metric = "gower")
```

```r
# You can supply the Gower distance matrix to several clustering algos
pam_gower <- pam(x = gower_dst, k = 8, diss = TRUE)
diana_gower <- diana(x = gower_dst, diss = TRUE)
agnes_gower <- agnes(x = gower_dst, diss = TRUE)
```

```r
fviz_nbclust(
  ames_1hot_scaled,
  pam,
  method = "wss",
  k.max = 25,
  verbose = FALSE
)
```




```
# k-means computation time on MNIST data
system.time(kmeans(features, centers = 10))
```

```
   user  system elapsed 
  13.97    0.14   14.11 
```

```
# CLARA computation time on MNIST data
system.time(clara(features, k = 10))
```

```
   user  system elapsed 
  40.83    0.04   40.86 
```

Hierachial Clustering

```
ames_scale <- AmesHousing::make_ames() %>%
  select_if(is.numeric) %>%  # select numeric columns
  select(-Sale_Price) %>%    # remove target column
  mutate_all(as.double) %>%  # coerce to double type
  scale()                    # center & scale the resulting columns
```

```
# For reproducibility
set.seed(123)

# Dissimilarity matrix
d <- dist(ames_scale, method = "euclidean")

# Hierarchical clustering using Complete Linkage
hc1 <- hclust(d, method = "complete" )
```

```
# For reproducibility
set.seed(123)

# Compute maximum or complete linkage clustering with agnes
hc2 <- agnes(ames_scale, method = "complete")

# Agglomerative coefficient
hc2$ac
```

```
[1] 0.926775
```

```
# methods to assess
m <- c( "average", "single", "complete", "ward")
names(m) <- c( "average", "single", "complete", "ward")

# function to compute coefficient
ac <- function(x) {
  agnes(ames_scale, method = x)$ac
}

# get agglomerative coefficient for each linkage method
purrr::map_dbl(m, ac)
```

```
  average    single  complete      ward
0.9139303 0.8712890 0.9267750 0.9766577
```

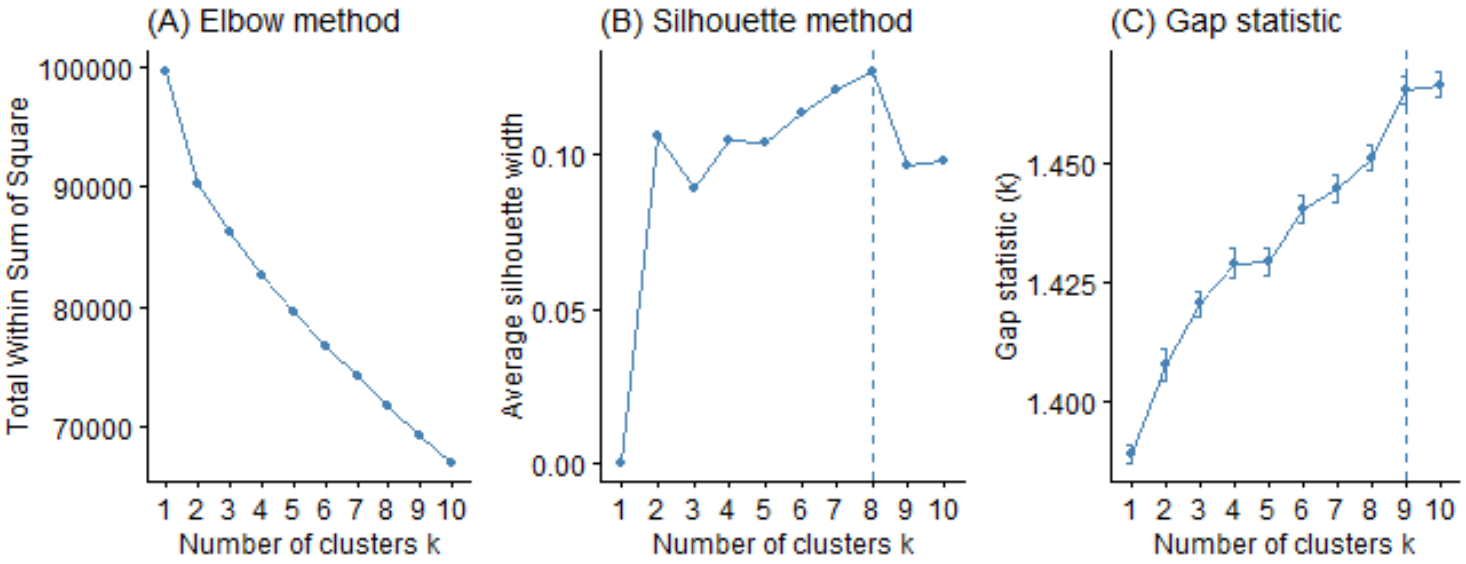
```
# compute divisive hierarchical clustering
hc4 <- diana(ames_scale)

# Divise coefficient; amount of clustering structure found
hc4$dc
```

```
[1] 0.9191094
```

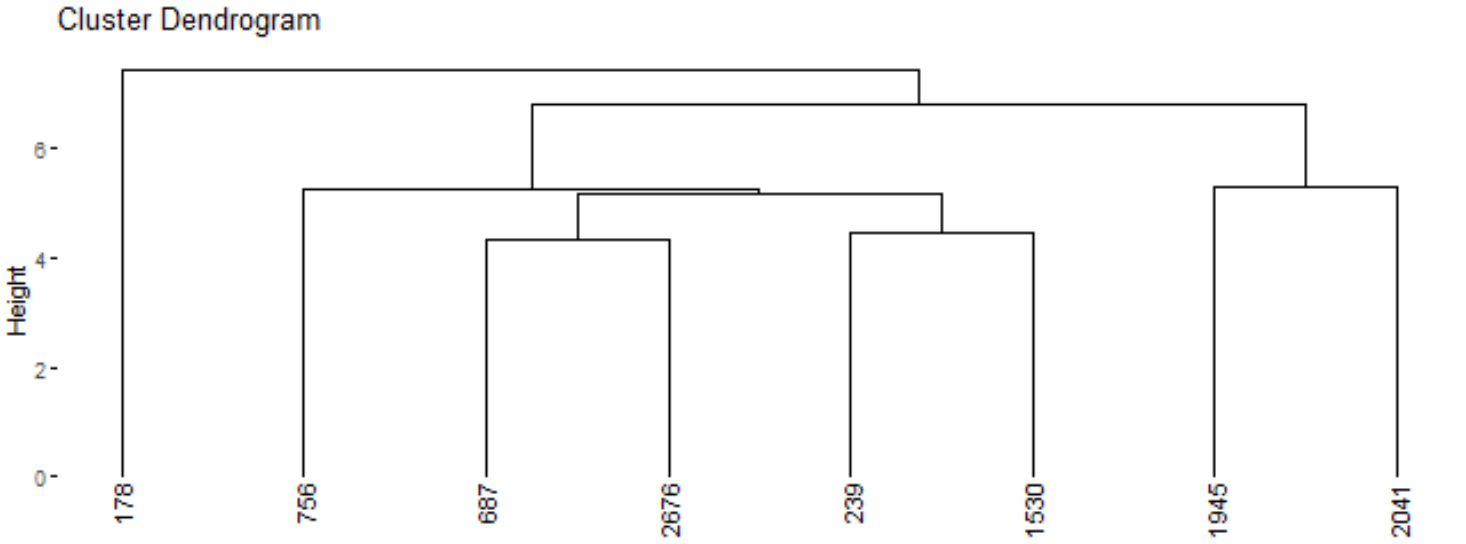
Optimal Clusters

```
# Plot cluster results
p1 <- fviz_nbclust(ames_scale, FUN = hcut, method = "wss",
                   k.max = 10) +
  ggtitle("(A) Elbow method")
p2 <- fviz_nbclust(ames_scale, FUN = hcut, method = "silhouette",
                   k.max = 10) +
  ggtitle("(B) Silhouette method")
p3 <- fviz_nbclust(ames_scale, FUN = hcut, method = "gap_stat",
                   k.max = 10) +
  ggtitle("(C) Gap statistic")
```

```
# Display plots side by side
gridExtra::grid.arrange(p1, p2, p3, nrow = 1)
```




```
# Construct dendorgram for the Ames housing example
hc5 <- hclust(d, method = "ward.D2" )
dend_plot <- fviz_dend(hc5)
dend_data <- attr(dend_plot, "dendrogram")
dend_cuts <- cut(dend_data, h = 8)
fviz_dend(dend_cuts$lower[[2]])
```

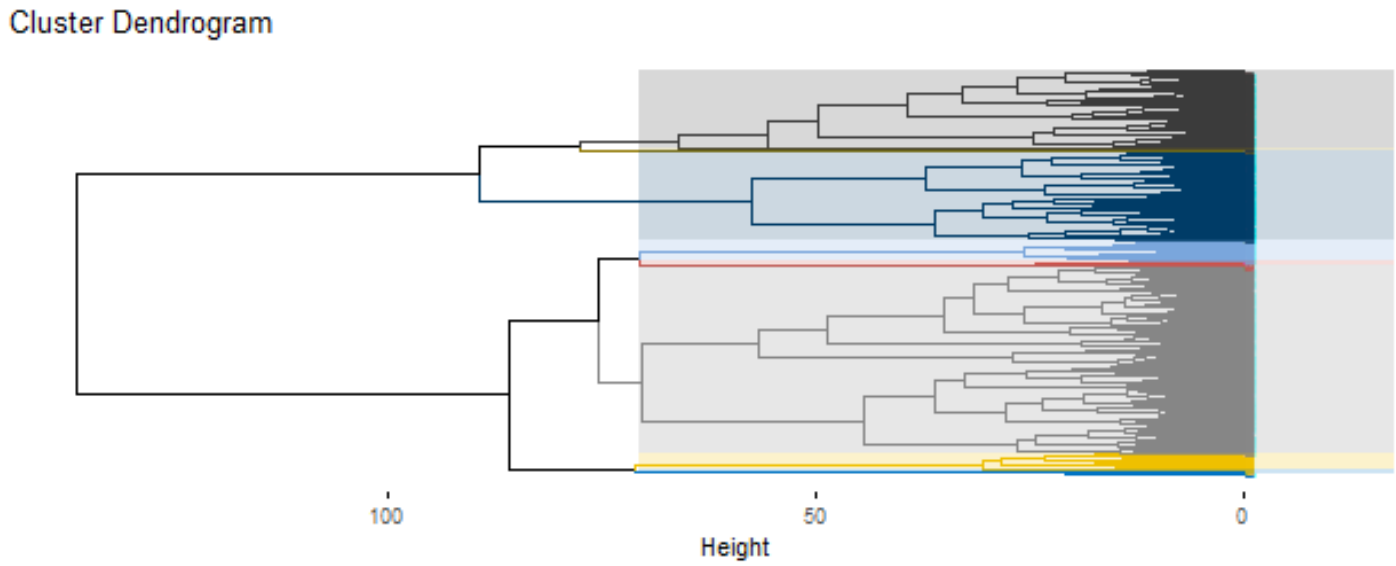



```
# Ward's method
hc5 <- hclust(d, method = "ward.D2" )
```

```
# Cut tree into 4 groups
sub_grp <- cutree(hc5, k = 8)

# Number of members in each cluster
table(sub_grp)
```

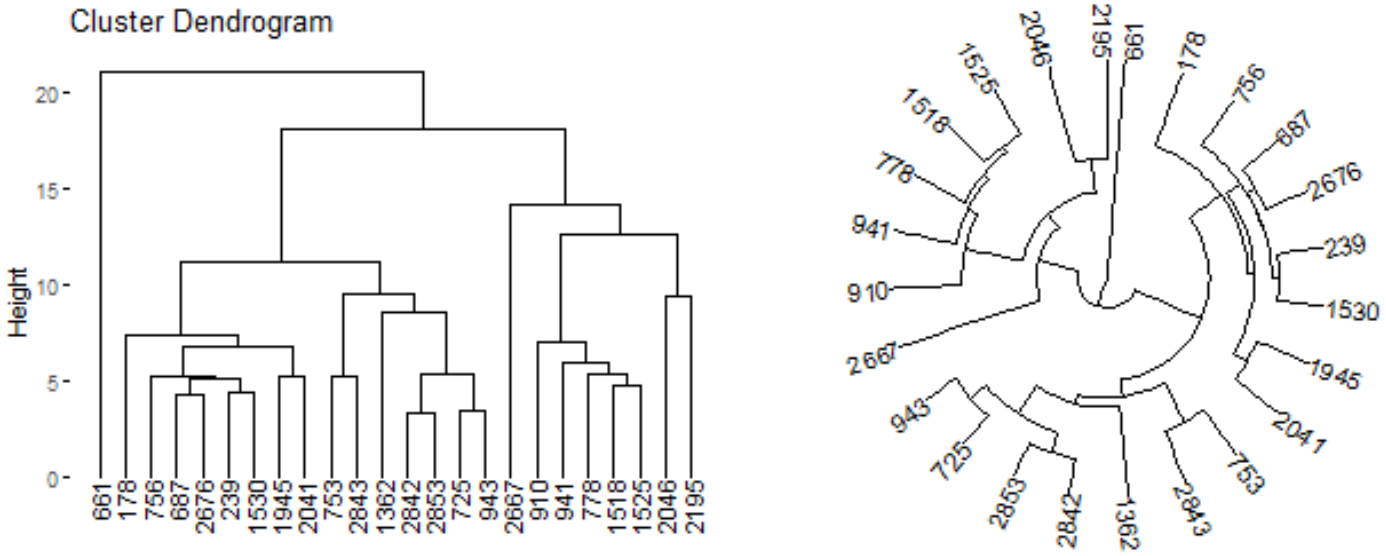
```
sub_grp
   1    2    3    4    5    6    7    8
1363  567  650   36  123  156   24   11
```

```
# Plot full dendogram
fviz_dend(
  hc5,
  k = 8,
  horiz = TRUE,
  rect = TRUE,
  rect_fill = TRUE,
  rect_border = "jco",
  k_colors = "jco",
  cex = 0.1
)
```



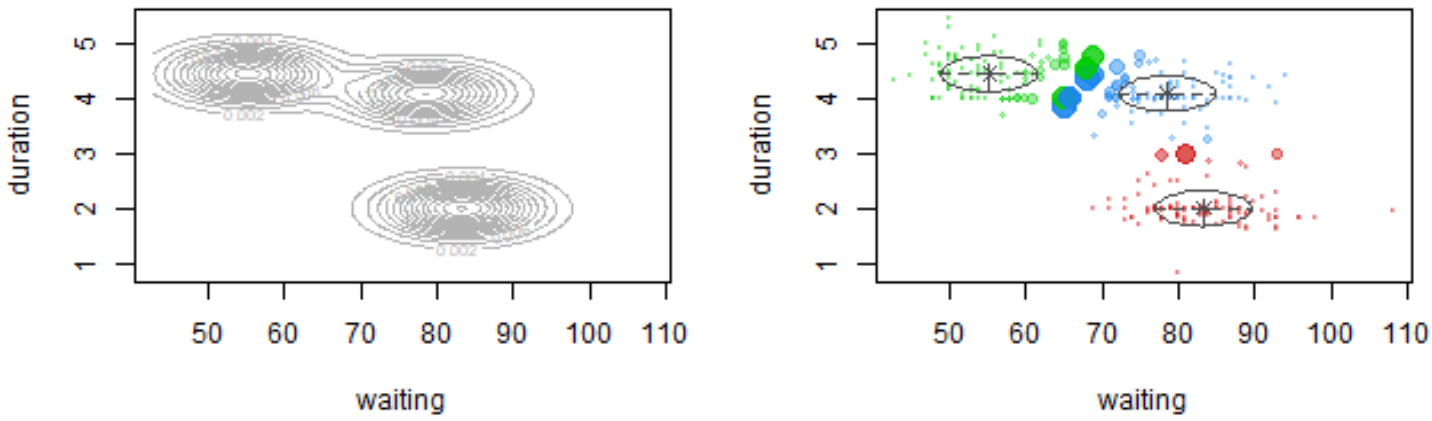

```
dend_plot <- fviz_dend(hc5)                # create full dendogram
dend_data <- attr(dend_plot, "dendrogram") # extract plot info
dend_cuts <- cut(dend_data, h = 70.5)      # cut the dendogram at
                                           # designated height
# Create sub dendrogram plots
p1 <- fviz_dend(dend_cuts$lower[[1]])
p2 <- fviz_dend(dend_cuts$lower[[1]], type = 'circular')
```

```
# Side by side plots
gridExtra::grid.arrange(p1, p2, nrow = 1)
```



Model-Based Clustering

```
data(geyser, package = 'MASS')

# Apply GMM model with 3 components
geyser_mc <- Mclust(geyser, G = 3)

par(mfrow = c(1, 2))
# Plot results
plot(geyser_mc, what = "density")
plot(geyser_mc, what = "uncertainty")
```




```
# Observations with high uncertainty
sort(geyser_mc$uncertainty, decreasing = TRUE) %>% head()
```

```
      187       211        85       285        28       206
0.4689087 0.4542588 0.4355496 0.4355496 0.4312406 0.4168440
```

```
summary(geyser_mc)
```

```
----------------------------------------------------
Gaussian finite mixture model fitted by EM algorithm
----------------------------------------------------

Mclust EEI (diagonal, equal volume and shape) model with 3 components:

 log-likelihood   n df      BIC       ICL
      -1371.823 299 10 -2800.65 -2814.577

Clustering table:
  1   2   3
 91 107 101
```

```
geyser_optimal_mc <- Mclust(geyser)

summary(geyser_optimal_mc)
```

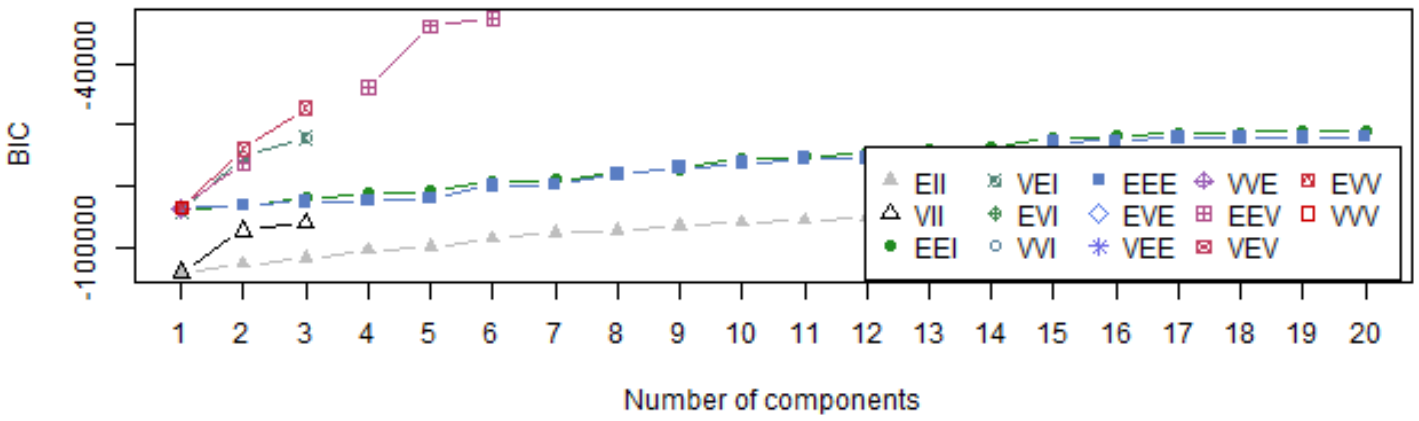
```
----------------------------------------------------
Gaussian finite mixture model fitted by EM algorithm
----------------------------------------------------

Mclust VVI (diagonal, varying volume and shape) model with 4 components:
```

```
 log-likelihood   n df       BIC       ICL
      -1330.13 299 19 -2768.568 -2798.746

Clustering table:
 1  2  3  4
90 17 98 94
```

```
my_basket_mc <- Mclust(my_basket, 1:20)

summary(my_basket_mc)
```

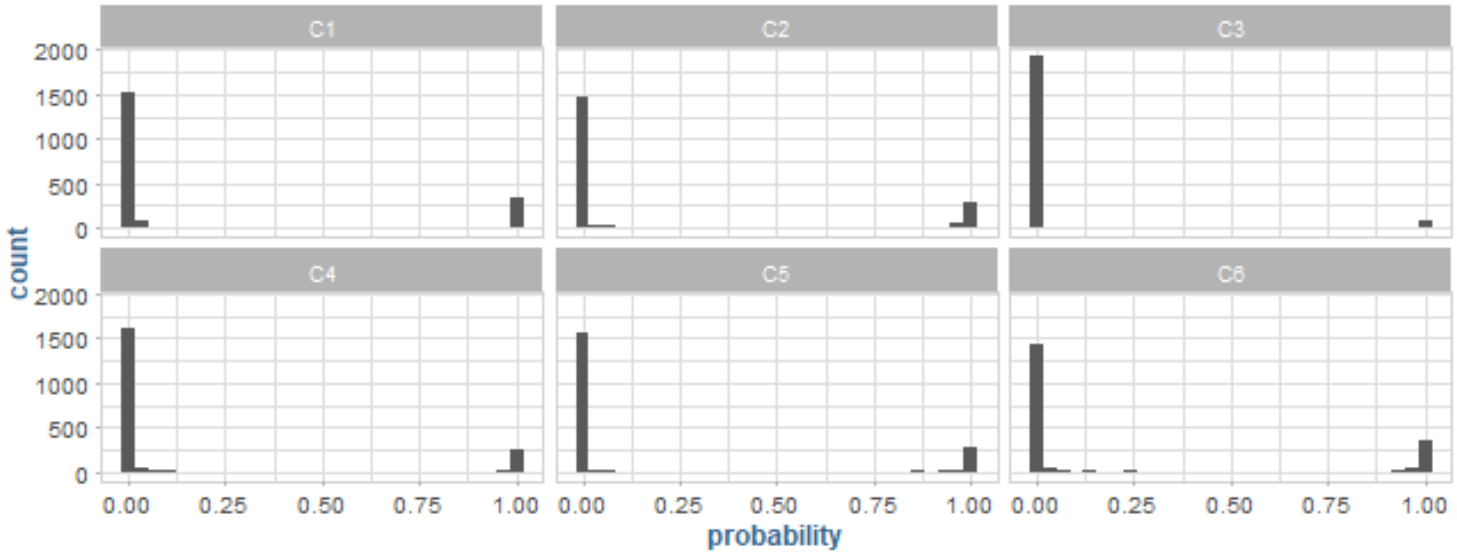
```
----------------------------------------------------
Gaussian finite mixture model fitted by EM algorithm
----------------------------------------------------

Mclust EEV (ellipsoidal, equal volume and shape) model with 6 components:

 log-likelihood    n   df      BIC       ICL
      8308.915 2000 5465 -24921.1 -25038.38

Clustering table:
  1   2   3   4   5   6
391 403  75 315 365 451
```

```
## ----------------------------------------------------
## Gaussian finite mixture model fitted by EM algorithm
## ----------------------------------------------------
##
## Mclust EEV (ellipsoidal, equal volume and shape) model with 6 components:
##
##  log-likelihood    n   df      BIC       ICL
##       8308.915 2000 5465 -24921.1 -25038.38
##
## Clustering table:
##   1   2   3   4   5   6
## 391 403  75 315 365 451

plot(my_basket_mc, what = 'BIC',
     legendArgs = list(x = "bottomright", ncol = 5))
```

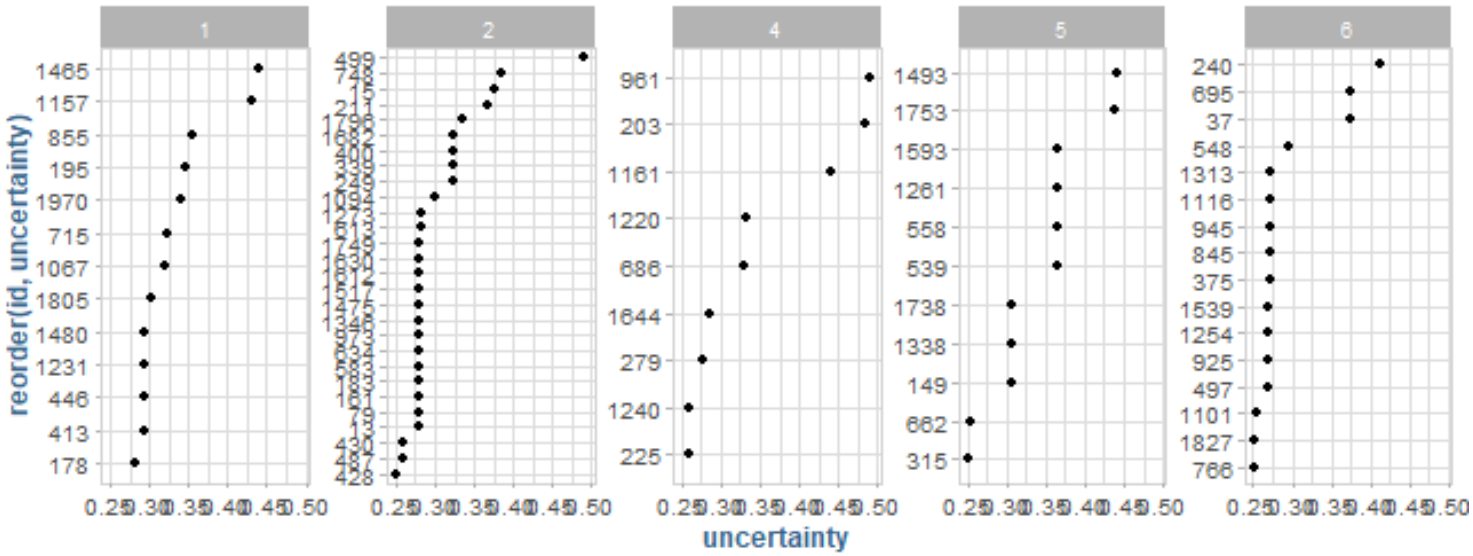



```
probabilities <- my_basket_mc$z
colnames(probabilities) <- paste0('C', 1:6)

probabilities <- probabilities %>%
  as.data.frame() %>%
  mutate(id = row_number()) %>%
  tidyr::gather(cluster, probability, -id)

ggplot(probabilities, aes(probability)) +
  geom_histogram() +
  facet_wrap(~ cluster, nrow = 2)
```

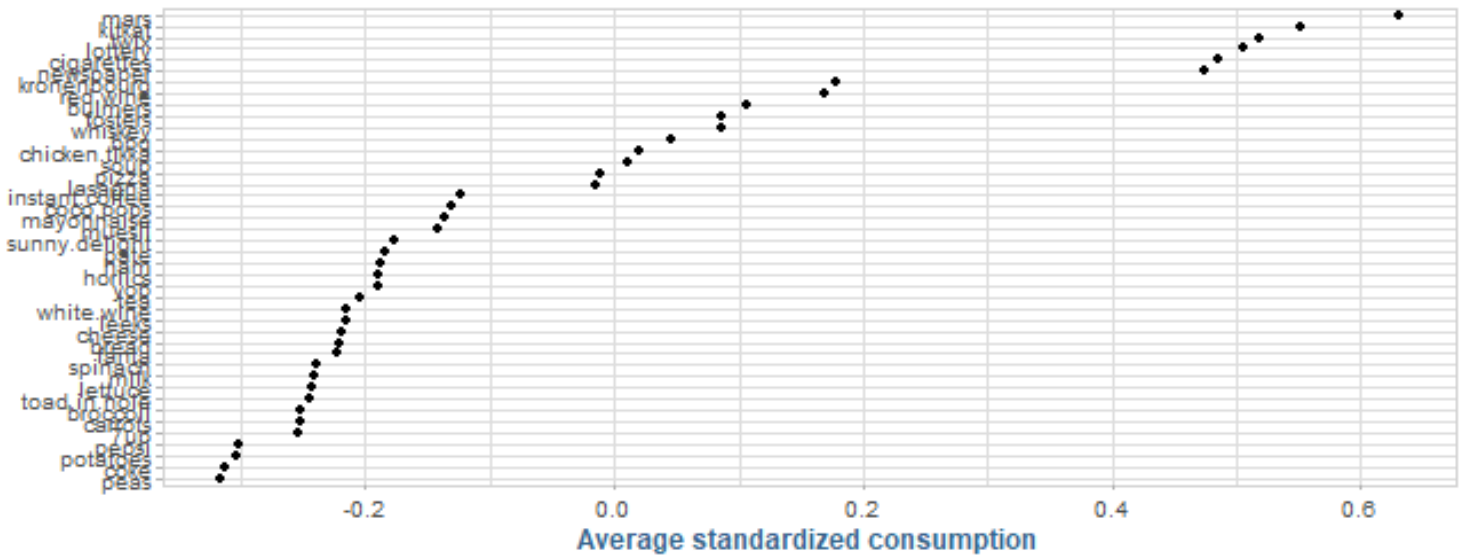
```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



```
uncertainty <- data.frame(
  id = 1:nrow(my_basket),
  cluster = my_basket_mc$classification,
  uncertainty = my_basket_mc$uncertainty
)

uncertainty %>%
  group_by(cluster) %>%
  filter(uncertainty > 0.25) %>%
  ggplot(aes(uncertainty, reorder(id, uncertainty))) +
  geom_point() +
  facet_wrap(~ cluster, scales = 'free_y', nrow = 1)
```




```
cluster2 <- my_basket %>%
  scale() %>%
  as.data.frame() %>%

  mutate(cluster = my_basket_mc$classification) %>%
  filter(cluster == 2) %>%
  select(-cluster)

cluster2 %>%
  tidyr::gather(product, std_count) %>%
  group_by(product) %>%
  summarize(avg = mean(std_count)) %>%
  ggplot(aes(avg, reorder(product, avg))) +
  geom_point() +
  labs(x = "Average standardized consumption", y = NULL)
```



Clean up

```
# clean up
rm(list = ls())
```