

# Statistics 452: Statistical Learning and Prediction

## Chapter 10, part 2: Cluster Analysis

Brad McNeney

# Clustering

- ▶ Clustering is a set of techniques for finding *subgroups*, or clusters, in a data set.
- ▶ A good set of clusters is a partition of the data such that observations are *similar* within clusters and *dissimilar* between clusters.
- ▶ Example: Suppose we have  $n$  observations of  $p$  features for cancer patients.
  - ▶ We can cluster to look for cancer sub-types.
  - ▶ However, these clusters will be highly dependent on how we measure similarity/dissimilarity.
  - ▶ Plus, we don't *know* that clusters correspond to cancer sub-types.

# Clustering as Dimension Reduction

- ▶ We can view clustering as an exploratory method to understand possible structure in our data.
- ▶ Similar in spirit to PCA, but a different mechanism
  - ▶ PCA finds a low-dimensional representation of observations explaining a good proportion of the variance
  - ▶ Clustering looks to find homogeneous sub-groups.

# Clustering Methods

- ▶ There are many.
- ▶ We focus on  $K$ -means/medoids and hierarchical clustering, and assume all features are quantitative.
- ▶ In  $K$ -means we partition into a pre-specified number of clusters.
- ▶ In hierarchical clustering we successively group observations.
  - ▶ Represent the nested partitions as a tree-like structure called a *dendrogram*.

# K-means Clustering

- ▶ Choose the desired number of clusters,  $K$ . The  $K$ -means clustering algorithm will assign each observation to *exactly one* of the  $K$  clusters.
- ▶ Let  $C_k$  be the set of indices for observations assigned to cluster  $k$ ,  $k = 1, \dots, K$ .
- ▶  $K$ -means clustering chooses clusters to solve

$$\min_{C_1, \dots, C_K} \sum_{k=1}^K W(C_k),$$

where  $W(C_k)$  is a measure of the amount by which observations within cluster  $C_k$  differ from one another.

## K-means Clustering: Choice of $W(C_k)$

- ▶ For quantitative variables, the standard choice for  $W(C_k)$  is the squared Euclidean distance,

$$W(C_k) = \frac{1}{|C_k|} \sum_{i,i' \in C_k} \sum_{j=1}^p (x_{ij} - x_{i'j})^2$$

where  $|C_k|$  denotes the number of observations in cluster  $k$ .

- ▶ It turns out that

$$\frac{1}{|C_k|} \sum_{i,i' \in C_k} \sum_{j=1}^p (x_{ij} - x_{i'j})^2 = 2 \sum_{i \in C_k} \sum_{j=1}^p (x_{ij} - \bar{x}_{kj})^2$$

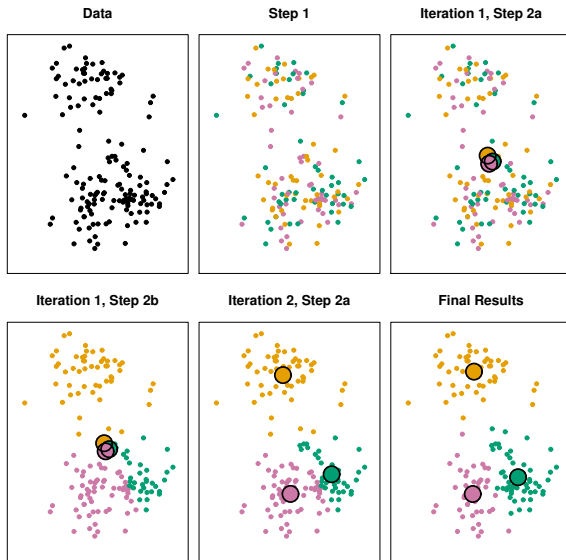
where the *cluster centroid*  $\bar{x}_{kj} = \frac{1}{|C_k|} \sum_{i \in C_k} x_{ij}$ .

- ▶ We can see where the name “K-means” comes from.

# K-means Clustering Algorithm

1. Randomly assign a number, from  $1, \dots, K$  to each of the observations. They serve as the initial cluster assignments for the observations.
2. Iterate the following steps until the cluster assignments stop changing:
  - (a) For each of the  $K$  clusters, compute the cluster centroid.  
The  $k$ th cluster centroid is the vector of the  $p$  feature means for the observations in the  $k$ th cluster for continuous variables.
  - (b) Assign each observation to the cluster whose centroid is closest in terms of Euclidean distance.

# K-means Clustering Algorithm, Simulated Data Example



► Text, Figure 10.6. Progress of the  $K$ -means algorithm with  $K = 3$ .

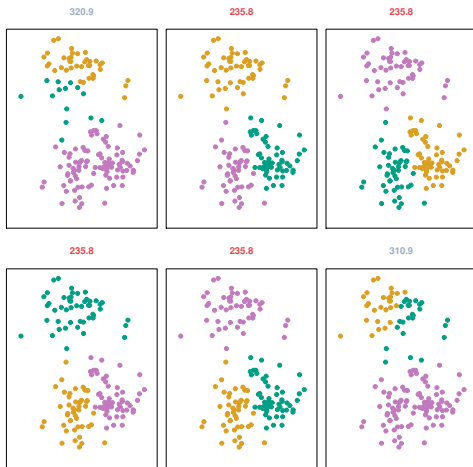


- ▶ See the video demo at  
<http://www.youtube.com/watch?v=zaKjh2N8jN4>.

# K-means Clustering and Local Minima

- ▶ The algorithm is guaranteed to decrease the value of  $\sum_k W(C_k)$  until no further improvement is possible, resulting in a *local* minimum.
- ▶ However, the local min depends on the random initialization. Hence, in practice we re-run the algorithm several times and keep the solution that achieves the lowest overall criterion value.

# K-means Clustering with Different Initialization



Text, Fig 10.7.  $K$ -means clustering performed six times on the same data with  $K = 3$  different random initializations. Above each plot is the value of the objective function. Three different local optima were found. The overall minimum, found four times, is 235.8. Note: Cluster labels different in 3 of 4.

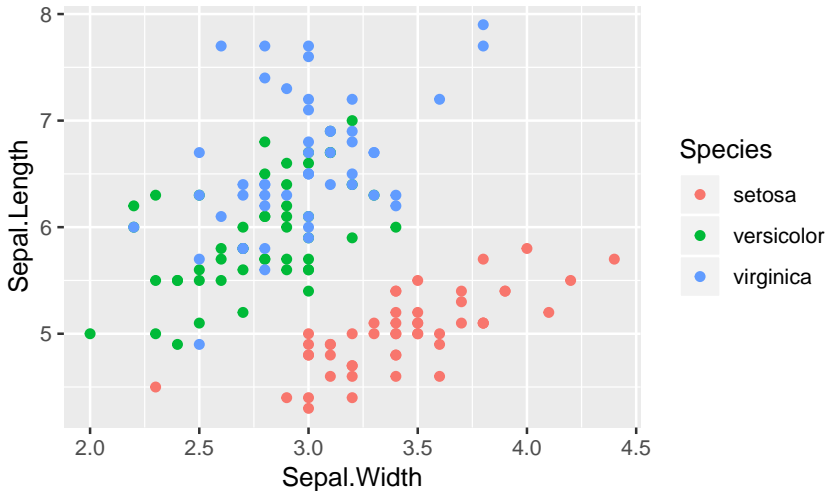
## K Means Clustering of the Iris Data

- We know there are three species of iris. Ignore the species labels and do the clustering.

```
library(ggplot2)
data(iris) # help(iris)
head(iris) # plot sepal length vs width with labels
```

##	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

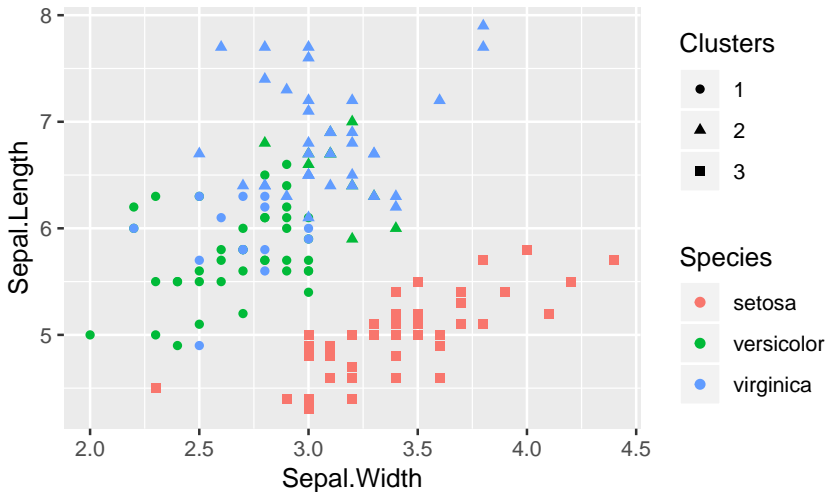
```
ggplot(iris,aes(x=Sepal.Width,y=Sepal.Length,color=Species)) +  
  geom_point()
```



```
library(dplyr)
set.seed(1)
irisX <- iris %>% select(-Species) %>% scale()
kout3 <- kmeans(irisX,centers=3,nstart=10)
iris <- data.frame(iris,Clusters=factor(kout3$cluster))
with(iris,table(Species,Clusters))
```

```
##           Clusters
## Species      1  2  3
##   setosa      0  0 50
##   versicolor 39 11  0
##   virginica  14 36  0
```

```
ggplot(iris,  
  aes(x=Sepal.Width,y=Sepal.Length,color=Species,shape=Clusters)) +  
  geom_point()
```

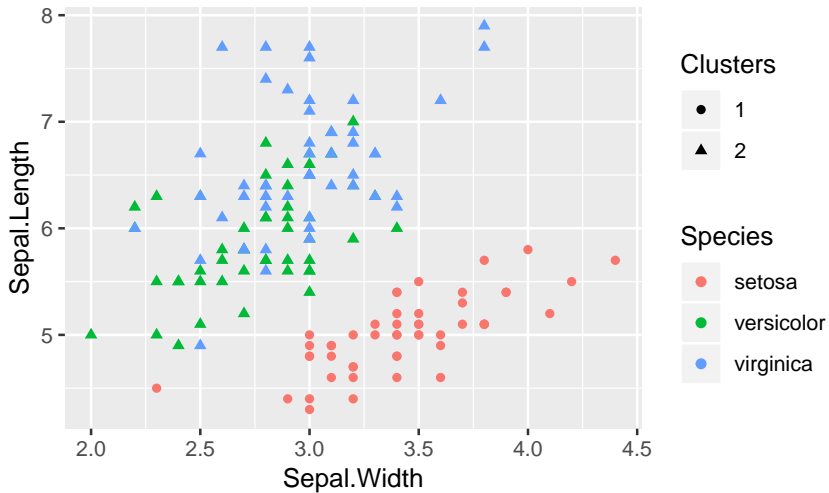


```
kout2 <- kmeans(irisX,centers=2,nstart=10)
iris$Clusters <- factor(kout2$cluster)
with(iris,table(Species,Clusters))
```

```
##           Clusters
## Species      1  2
##   setosa     50  0
##   versicolor  0 50
##   virginica   0 50
```



```
ggplot(iris,  
  aes(x=Sepal.Width,y=Sepal.Length,color=Species,shape=Clusters)) +  
  geom_point()
```



# Scaling Variables

- ▶ Whether or not to scale variables depends on the context, but usually we will.

## Choosing $K$

- ▶ Like PCA, there is no “best” method for choosing  $K$ .
- ▶ Cross-validation is not an option because there is no outcome.
- ▶ For small  $p$ , can visualize the clusters, but this becomes difficult as  $p$  grows.
- ▶ The silhouette plot is a graphical approach.
  - ▶ Discussed after the PAM algorithm below.

# Sensitivity to Outliers

- ▶ Means and Euclidean distances are sensitive to outliers and  $K$ -means depends on both
  - ▶ Means are cluster centroids.
  - ▶ Criterion to minimize is a sum of squared Euclidean distances.

# K-Medoids Clustering

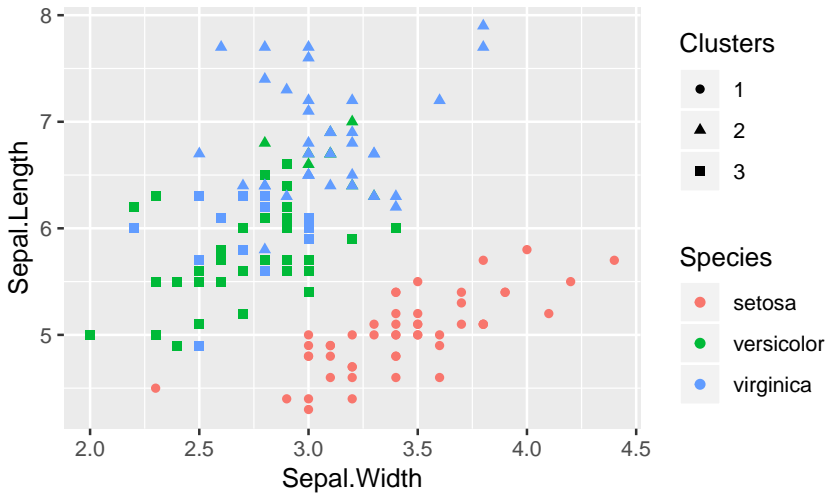
- ▶ An alternative to  $K$ -means is  $K$ -medoids clustering.
- ▶ Cluster centres are medoids, which are observations chosen to represent each cluster.
- ▶ There is flexibility in choosing the dissimilarity measure
  - ▶ Can choose a more robust measure, such as  $\ell_1$  (so-called Manhattan) distance.
- ▶ An implementation of  $K$ -medoids is the PAM (partitioning around medoids) algorithm.

# PAM on the Iris Data

```
library(cluster)
pout3 <- pam(irisX,k=3)
iris$Clusters <- factor(pout3$cluster)
with(iris,table(Species,Clusters))
```

```
##           Clusters
## Species      1  2  3
##  setosa      50  0  0
##  versicolor  0  9 41
##  virginica   0 36 14
```

```
ggplot(iris,  
  aes(x=Sepal.Width,y=Sepal.Length,color=Species,shape=Clusters)) +  
  geom_point()
```

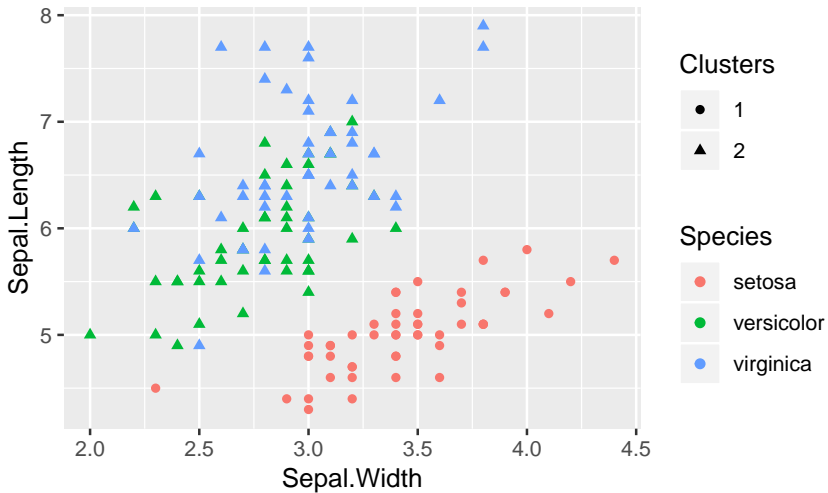


```
pout2 <- pam(irisX,k=2)
iris$Clusters <- factor(pout2$cluster)
with(iris,table(Species,Clusters))
```

```
##           Clusters
## Species      1  2
##   setosa     50  0
##   versicolor  0 50
##   virginica   0 50
```



```
ggplot(iris,  
  aes(x=Sepal.Width,y=Sepal.Length,color=Species,shape=Clusters)) +  
  geom_point()
```



# The Silhouette Plot

- ▶ For each observation we compute a measure of cluster certainty, called the silhouette width, that takes values in  $[-1, 1]$ .
  - ▶ Values near 1 indicate certainty that the observation is in the right cluster and
  - ▶ Values near  $-1$  indicate that the observation is in the wrong cluster.
- ▶ Formula for silhouette width is given in on the next slide.
- ▶ If the silhouette values in a cluster are, say, below average, this suggests the cluster can be merged with another.

# Silhouette Widths

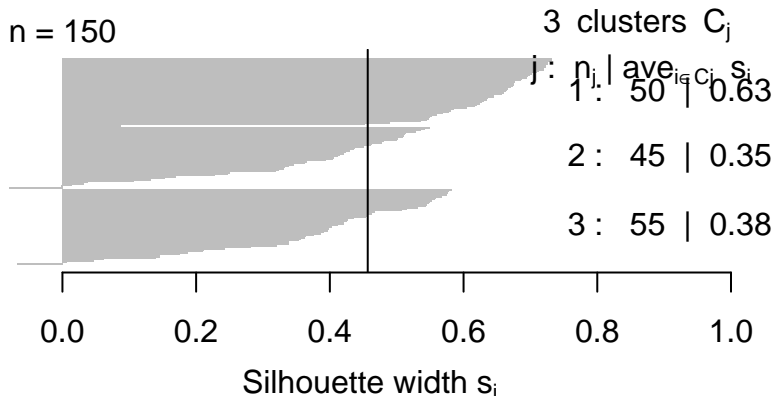
- ▶ For observation  $i$  in cluster  $k$ , the silhouette width  $s(i)$  is defined as follows.
- ▶ Let  $a(i)$  be the average dissimilarity between  $i$  and all other observations in **cluster**  $k$ .
- ▶ For **any other cluster**  $C$  ( $i$  not in  $C$ ), let  $d(i, C)$  be the average dissimilarity between  $i$  and all observations in cluster  $C$ .
- ▶ Let  $b(i) = \min_C d(i, C)$  (minimum over all  $C$  such that  $i$  not in  $C$ ).
- ▶ Then

$$s(i) = \frac{b(i) - a(i)}{\max(a(i), b(i))}$$

- ▶ Note:  $a(i)$  is a *dissimilarity* measure, and should be  $\ll b(i)$  if  $i$  confidently in cluster  $k$ .

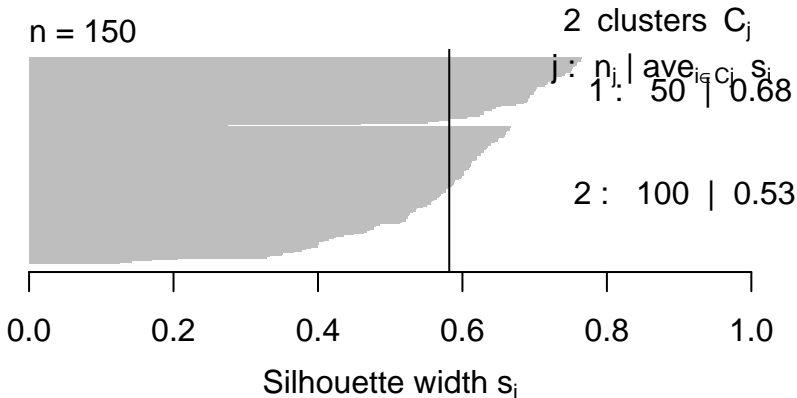
# Silhouette Plots for the Iris Data

```
sil3 <- silhouette(pout3)
plot(sil3,main=""); abline(v=mean(sil3[, "sil_width"]))
```



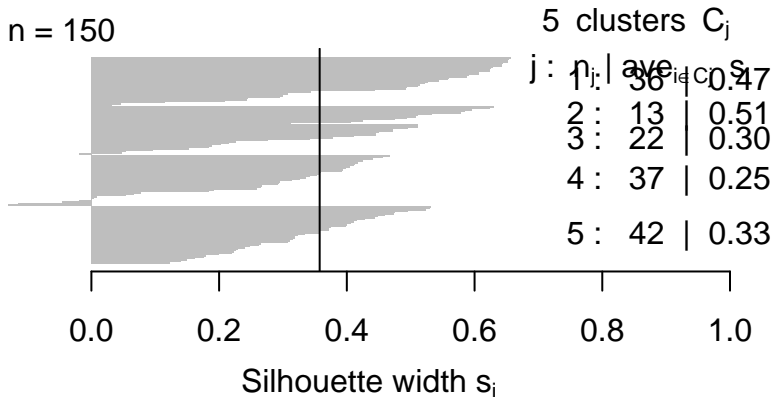
Average silhouette width : 0.46

```
sil2 <- silhouette(pout2)
plot(sil2,main=""); abline(v=mean(sil2[, "sil_width"]))
```



Average silhouette width : 0.58

```
sil5 <- silhouette(pam(irisX,k=5))
plot(sil5,main=""); abline(v=mean(sil5[, "sil_width"]))
```



Average silhouette width : 0.36

- Had to go to about  $K = 15$  before silhouettes suggested merging clusters (not shown).

# Clustering with categorical data.

## ► Use the HUI data

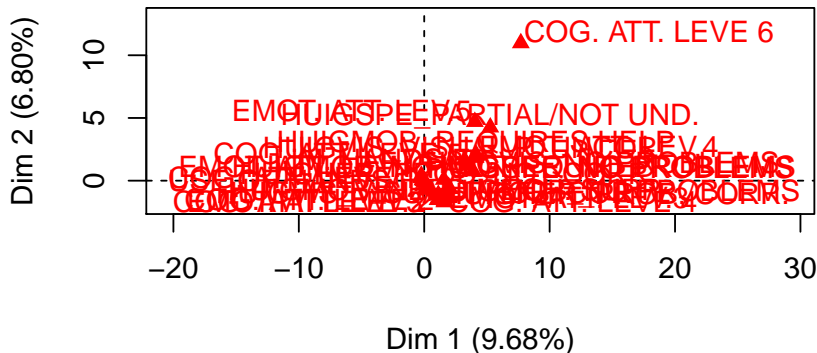
```
hui <- read.csv("HUI.csv.gz", na.strings = "NOT STATED")
hui[hui=="NA"] <- NA
hui <- na.omit(hui)
library(dplyr)
hsub <- select(hui, HUIDCOG, HUIGDEX, HUIDEMO, HUIGHER, HUIGMOB,
               HUIGSPE, HUIGVIS)
names(hsub)
```

```
## [1] "HUIDCOG" "HUIGDEX" "HUIDEMO" "HUIGHER" "HUIGMOB" "HUIGSPE" "HUIGVIS"
```

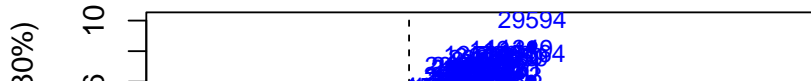
## Extract PCs from MCA

```
library(FactoMineR)
res.mca <- MCA(hsub, row.w = hui$WTS_M) #huiPCs <- res.mca$ind$coord
```

### MCA factor map

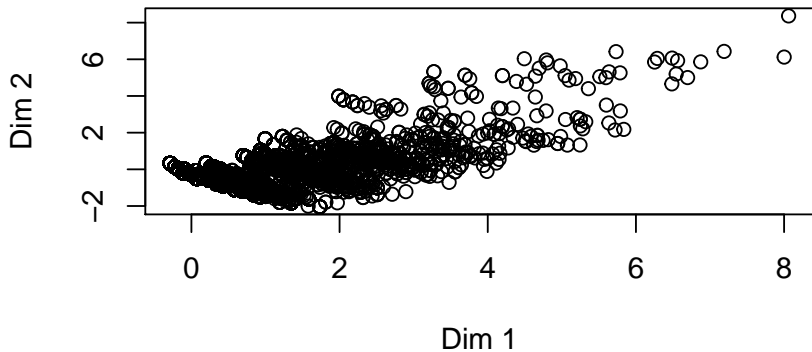


### MCA factor map





```
plot(huiPCs[,1:2])
```

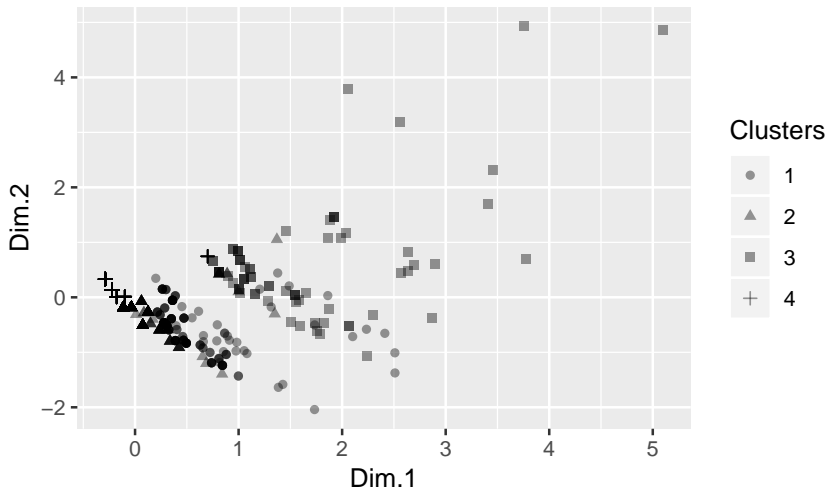


# Cluster on PCs

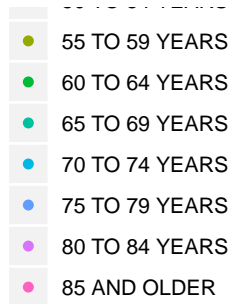
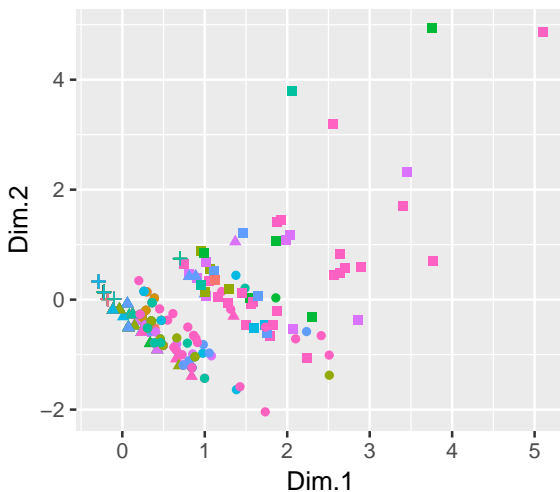
- ▶ Will work with a small subset (first 1000 people) to keep computation and overplotting down.

```
n <- 1000
hk <- kmeans(huiPCs,centers=4,nstart=10)
# plot for a small sample of the dataset
huiPCs <- data.frame(huiPCs[1:n,])
huiPCs$Clusters <- factor(hk$cluster[1:n])
huiPCs$age <- hui$DHHGAGE[1:n]
huiPCs$sex <- hui$DHH_SEX[1:n]
```

```
library(ggplot2)
ggplot(huiPCs,
       aes(x=Dim.1,y=Dim.2,shape=Clusters)) +
  geom_point(alpha=.4)
```



```
ggplot(huiPCs,
       aes(x=Dim.1,y=Dim.2,color=age,shape=Clusters)) +
  geom_point()
```



Clusters



```
ggplot(huiPCs,  
  aes(x=Dim.1,y=Dim.2,color=sex,shape=Clusters)) +  
  geom_point()
```

