## **Eltecon Data Science Course by Emarsys**

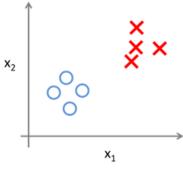
**Unsupervised Learning** 

Gábor Kocsis

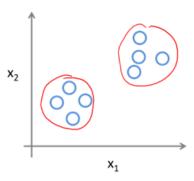
December 4, 2019

# Supervised vs unsupervised learning

# What is the difference between Supervised and Unsupervised learning?



Supervised Learning



Unsupervised Learning

# **Supervised learning**

- For each observation of the predictor measurements (X) there is an associated response measurement (Y).
- The goal is to fit a model that predicts the amount or the label of the response.
- Eg. linear regression, logistic regression, classification etc.

## **Unsupervised learning**

- We observe measurements (X), but no associated response variable (Y), so we cannot fit any regressions.
- The goal is to find relationship or structure among the measurements.

# Goals of unsupervised learning

- Find patterns in the features of the data by dimensionality reduction
  - Ex 1. instead of using both humidity and rainfall in a classification problem, they can be collapsed into just one underlying feature, since both of them are strongly correlated
- Find homogenous subgroups (clusters) within a population
  - Ex 1. segmenting consumers based on demographics and purchasing history
  - Ex 2. find similar movies based on features of each movie and reviews of the movies

# **Dimensionality reduction with PCA**

## Dimensionality reduction with PCA

 This section is based on James, G., Witten, D., Hastie, T., and Tibshirani, R. An Introduction to Statistical Learning with Applications in R. Pages 373-385.

# Dimensionality reduction with PCA

- Principal components allow us to summarize a large set of correlated variables with a smaller number of representative variables that collectively explain most of the variablility in the original set.
- Principal component analysis (PCA) is simply reducing the number of variables of a data set, while preserving as much information as possible.
- Reducing the number of variables comes at the expense of accuracy, so with PCA we trade a little accuracy for simplicity.

## What are principal components?

- Try to visualize n observations with measurements of p features by two-dimensional scatterplots (with p = 10 there are 45 plots!).
- Instead we'd like to find a low-dimensional representation of the data that captures most of the information.
- Imagine that each of the *n* observations lives in a *p*-dimensional space, but not all of these dimensions are equally *interesting*.
- Interesting is measured by the amount that the observations vary along each dimension.
- Each of the dimensions (principal components) found by PCA is a linear combination of the p features.

# **Steps of PCA**

## **Steps of PCA**

• This section is based on Jaadi, Z. A step by step explanation of principal component analysis.

# **Step 1: Standardization**

- The aim is to standardize the range of the continuous initial variables so that each one of them contributes equally to the analysis.
- We do standardization because PCA is quite sensitive regarding the variances of the initial variables (variables with larger ranges dominate over those with small ranges leading to biased results).
- Standardization can be done by **subtracting the mean and dividing by the standard deviation** for each value of each variable.

## Step 2: Covariance matrix computation

- The aim is to understand how the variables of the input data set are varying from the mean with respect to each other, i.e. to see if there is any relationship between them.
- We compute the covariance matrix in order to identify highly correlated variables.

# Step 2: Covariance matrix computation (cont.)

$$\begin{bmatrix} Cov(x,x) & Cov(x,y) & \cdots & Cov(x,p) \\ Cov(y,x) & Cov(y,y) & \cdots & Cov(y,p) \\ \vdots & \vdots & \ddots & \vdots \\ Cov(p,x) & Cov(p,y) & \cdots & Cov(p,p) \end{bmatrix}$$

- The main diagonal gives the variances of each variable, moreover the entries of the covariance matrix are symmetric with respect to it.
- If the sign of a covariance is
  - positive: the two variables increase or decrease together (correlated)
  - if negative: one increases when the other decreases (inversely correlated)

# Step 3: Compute the eigenvectors and eigenvalues of the covariance matrix to identify the PCs

- Every eigenvector has an eigenvalue and their number is equal to the number of dimensions of the data.
- Eigenvectors of the covariance matrix are actually the directions of the axes where there is the most variance and that we call principal components.
- Eigenvalues are simply the coefficients attached to eigenvectors, which give the amount of variance carried in each principal component.
- By ranking eigenvectors in order of their eigenvalues, highest to lowest, we get the principal components in order of significance.
- To compute the percentage of variance explained by each component, we divide the eigenvalue of each component by the sum of eigenvalues.

## **Step 4: Feature vector**

- Choose whether to keep all components or discard those of lesser significance (of low eigenvalues), and form with the remaining ones a matrix of vectors that we call feature vector.
- The feature vector is simply a matrix that has as columns the eigenvectors of the components that we decide to keep.
- Discarding a component will reduce dimensionality, and will consequently cause a loss of information in the final data set.

## **Step 5: Score vector**

- The aim is to use the feature vector to reorient the data from the original axes to the ones represented by the principal components.
- Recasting can be done by multiplying the transpose of the original data set by the transpose of the feature vector.

 $ScoreVector = FeatureVector^T * StandardizedOriginalDataSet^T$ 

 This section is based on James, G., Witten, D., Hastie, T., and Tibshirani, R. An Introduction to Statistical Learning with Applications in R. Pages 401-403.

#### head(USArrests)

##		Murder	Assault	UrbanPop	Rape
##	Alabama	13.2	236	58	21.2
##	Alaska	10.0	263	48	44.5
##	Arizona	8.1	294	80	31.0
##	Arkansas	8.8	190	50	19.5
##	${\tt California}$	9.0	276	91	40.6
##	Colorado	7.9	204	78	38.7

##

##

```
apply(USArrests, 2, mean)
## Murder Assault UrbanPop Rape
## 7.788 170.760 65.540 21.232
apply(USArrests, 2, var)
```

18.97047 6945.16571 209.51878 87.72916

UrbanPop

Murder Assault

Rape

##

##

```
pca_output <- prcomp(USArrests, scale = TRUE)

pca_output$center

## Murder Assault UrbanPop Rape
## 7.788 170.760 65.540 21.232

pca_output$scale</pre>
```

• center and scale are the *means* and *standard deviations* of the

4.355510 83.337661 14.474763 9.366385

 center and scale are the means and standard deviations of the variables that were used for scaling prior to implementing PCA

UrbanPop

Rape

Murder Assault

#### pca\_output\$rotation

```
## PC1 PC2 PC3 PC4
## Murder -0.5358995 0.4181809 -0.3412327 0.64922780
## Assault -0.5831836 0.1879856 -0.2681484 -0.74340748
## UrbanPop -0.2781909 -0.8728062 -0.3780158 0.13387773
## Rape -0.5434321 -0.1673186 0.8177779 0.08902432
```

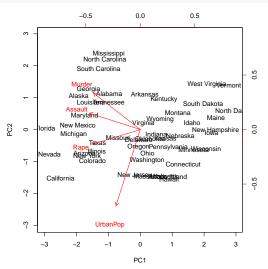
 rotation is the matrix whose columns contain the eigenvectors (also called loadings)

dim(pca output\$x)

```
## [1] 50
head(pca_output$x)
                     PC1
                                PC2
                                             PC3
##
                                                          PC4
              -0.9756604
                         1.1220012 -0.43980366
                                                  0.154696581
  Alabama
              -1.9305379 1.0624269
                                      2.01950027 - 0.434175454
  Alaska
              -1.7454429 -0.7384595
                                      0.05423025 - 0.826264240
  Arizona
  Arkansas
               0.1399989
                          1.1085423
                                      0.11342217 -0.180973554
## California -2.4986128 -1.5274267
                                      0.59254100 -0.338559240
  Colorado
              -1 4993407 -0 9776297
                                      1.08400162
                                                  0.001450164
```

matrix x has as its columns the principal component score vectors

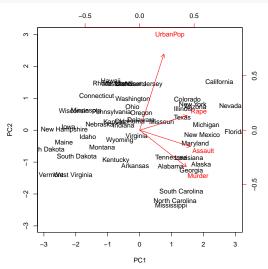
biplot(pca\_output, scale = 0)



```
pca_output$rotation <- -pca_output$rotation
pca_output$x <- -pca_output$x</pre>
```

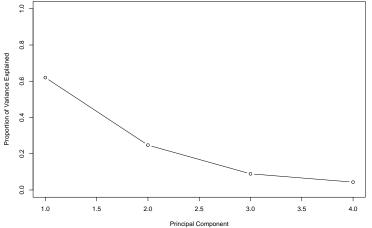
• principal components are only unique up to a sign change

biplot(pca\_output, scale = 0)

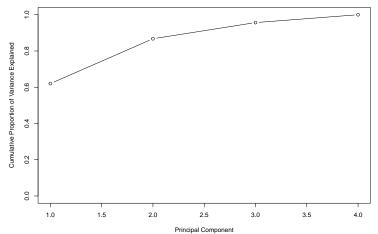


```
pca output$sdev
## [1] 1.5748783 0.9948694 0.5971291 0.4164494
pca_var <- pca_output$sdev^2</pre>
pca_var
## [1] 2.4802416 0.9897652 0.3565632 0.1734301
pca_var_expl <- pca_var / sum(pca_var)</pre>
pca var expl
## [1] 0.62006039 0.24744129 0.08914080 0.04335752
```

```
plot(pca_var_expl, xlab = "Principal Component",
    ylab = "Proportion of Variance Explained",
    ylim = c(0, 1), type = "b")
```



```
plot(cumsum(pca_var_expl), xlab = "Principal Component",
    ylab = "Cumulative Proportion of Variance Explained",
    ylim = c(0, 1), type = "b")
```



# **Clustering methods**

# **Clustering methods**

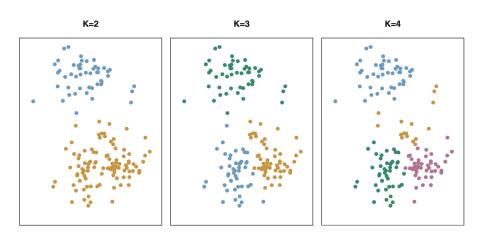
 This section is based on James, G., Witten, D., Hastie, T., and Tibshirani, R. An Introduction to Statistical Learning with Applications in R. Pages 385-401.

# K-means clustering

## K-means theory

- Clustering the observations of a data set means partitioning them into groups so that observations within each group are quite *similar*, while observations in different groups are quite *different* from each other.
- Each observation should belong to exactly one cluster.
- To perform K-means clustering, we must first specify the desired number of clusters K; then the K-means algorithm will assign each observation to exactly one of the K clusters.

# K-means clustering with different values of K



#### K-means optimization problem

- A good clustering is one for which the within-cluster variation is as small as possible.
- If the within-cluster variation for cluster  $C_k$  is a measure  $W(C_k)$ , then we want to solve:

$$\underset{C_1,...,C_K}{\text{minimize}} \left\{ \sum_{k=1}^K W(C_k) \right\}$$

The most common choice of measure is the squared Euclidean distance:

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

Thus the optimization problem is:

$$\underset{C_1,...,C_K}{\text{minimize}} \left\{ \sum_{k=1}^K \frac{1}{|C_k|} \sum_{i,i' \in C_k} \sum_{j=1}^p (x_{ij} - x_{i'j})^2 \right\}$$

#### K-means algorithm

- **Q** Randomly assign a number, from 1 to K, to each of the observations. These serve as initial cluster assignments for the observations.
- 2 Iterate until the cluster assignments stop changing:
  - (a) For each of the K clusters, compute the cluster *centroid*. The kth cluster centroid is the vector of the p feature means for the observations in the kth cluster.
  - **(b)** Assign each observation to the cluster whose centroid is closest (where *closest* is defined using Euclidean distance).

#### K-means algorithm (cont.)

- The results will depend on the initial (random) cluster assignment in Step 1, thus we need to run the algorithm multiple times from different random initial configurations.
- We select the *best* solution, for which the objective is the smallest.

#### K-means algorithm explained

Notice that:

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

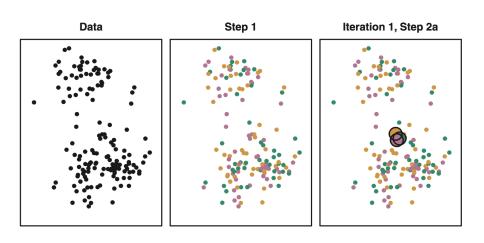
where

$$\bar{x}_{kj} = \frac{1}{|C_k|} \sum_{i \in C_k} x_{ij}$$

is the mean for feature j in cluster  $C_k$ 

- The above shows that in Step 2(a) the cluster means for each feature are the constants that minimize the sum-of-squared deviations, and by reallocation in Step 2(b) we can only improve.
- When the result no longer changes, a local optimum has been reached.

#### K-means clustering progression



#### K-means clustering progression (cont.)

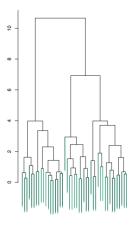
Iteration 1, Step 2b **Final Results** Iteration 2, Step 2a

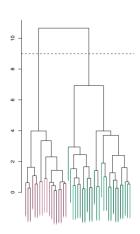
### Hierarchical clustering

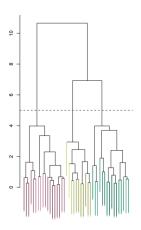
#### Hierarchical clustering

- Hierarchical clustering does not require choosing a particular K number of clusters.
- It results in a tree-based representation of the observations, called a dendogram.
- We focus on bottom-op or agglomerative clustering (vs top-down or divisive).

# The dendogram







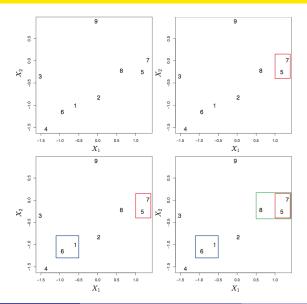
#### The dendogram

- Each *leaf* is an observation, as we move up the tree, leafs begin to *fuse* into branches based on their *similarity*.
- The height of fusion (on the vertical axis) indicates how different the observations are.
- Clusters are defined by cuting the dendogram horizontally, although where to make the cut is not so obvious.
- Hierarchical refers to that clusters obtained by cutting the tree at a given height are necessarily nested within the clusters obtained by cutting higher. However, it isn't always the case!

#### Hierarchical clustering algorithm

- We need to define a dissimilarity measure between each pair of observations (most often Euclidean distance).
- Starting from the bottom, each observation is treated as a separate cluster, then the two most similar are *fused*, next the two most similar clusters are fused, etc., until all observations belon to a cluster and the dendogram is complete.
- Dissimilarity between clusters depend on the selected linkage (average and complete linkage are the most preferred ones) and the dissimilarity measure.

# Hierarchical clustering algorithm (example)



#### Linkage types

- The linkage function tells you how to measure the distance between clusters.
- Single linkage: Minimal intercluster dissimilarity.

$$f = min(d(x, y))$$

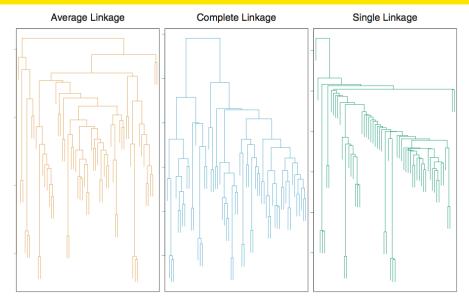
Complete linkage: Maximal intercluster dissimilarity.

$$f = max(d(x, y))$$

Average linkage: Mean intercluster dissimilarity.

$$f = average(d(x, y))$$

### Clustering with different linkages



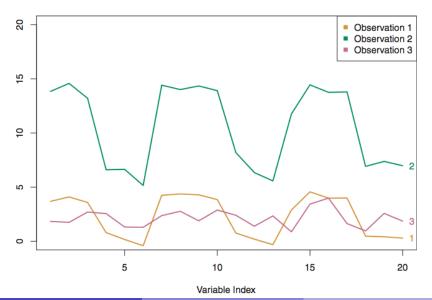
#### Dissimilarity measures

• Euclidean distance is the most common measure used.

$$\sqrt{\sum_i (a_i - b_i)^2}$$

- Correlation-based distance is also very useful, eg. it is used for gene expression.
  - It considers two observations to be similar if their features are highly correlated, even though the observed values may be far apart in terms of Euclidean distance.
  - The distance between two vectors is 0 when they are perfectly correlated.
  - It focuses on the shapes of observation profiles rather than their magnitudes.

#### Euclidean vs correlation-based distance



#### Scaling variables before clustering

- If variables are scaled to have standard deviation one before dissimilarities are computed, then each variable will be given equal importance in the hierarchical clustering.
- We might also want to scale the variables to have standard deviation one if they are measured on different scales.

#### Practical issues in clustering

- There are tevhniques for validating clustering, although there is no single best approach.
- Outliers might distort the clustering. The mixture of models are used to treat this.
- Clustering methods generally are not very robust to perturbations to the data.

# **Summary**

#### **Summary**

- Supervised vs unsupervised learning
- PCA looks to find a low-dimensional representation of the observations that explain a good fraction of the variance.
- Clustering looks to find homogeneous subgroups among the observations.
  - In K-means clustering, we seeks to partition the observations into a pre-specified number of clusters.
  - In hierarchical clustering, we do not know in advance how many clusters
    we want; in fact, we end up with a tree-like visual representation of the
    observations, called a dendrogram, that allows us to view at once the
    clusterings obtained for each possible number of clusters, from 1 to n.

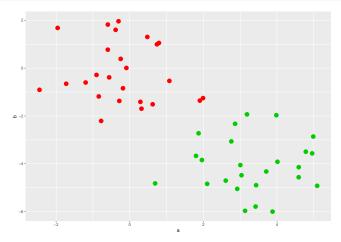
#### **Clustering lab**

#### **Clustering lab**

 This section is based on James, G., Witten, D., Hastie, T., and Tibshirani, R. An Introduction to Statistical Learning with Applications in R. Pages 404-407.

```
set.seed(2)
x <- data.table(a = rnorm(50), b = rnorm(50))
x[1:25, `:=`(a = a + 3, b = b - 4)]
km_output <- kmeans(x, centers = 2, nstart = 20)
km_output$cluster</pre>
```

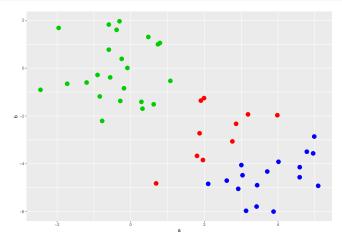
```
ggplot(x, aes(x = a, y = b)) +
    geom_point(colour = (km_output$cluster+1), size = 4)
```



```
K-means clustering with 2 clusters of sizes 25, 25
Cluster means:
          а
1 3.3339737 -4.0761910
2 -0.1956978 -0.1848774
Clustering vector:
                                          [37] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
Within cluster sum of squares by cluster:
[1] 63.20595 65.40068
(between_SS / total_SS = 72.8 %)
Available components:
[1] "cluster" "centers"
                               "totss"
                                              "withinss"
[5] "tot.withinss" "betweenss" "size"
                                              "iter"
[9] "ifault"
```

```
set.seed(4)
km_output <- kmeans(x, centers = 3, nstart = 20)</pre>
```

```
ggplot(x, aes(x = a, y = b)) +
    geom_point(colour = (km_output$cluster+1), size = 4)
```



# Different nstart initial cluster assignments result different total within-cluster sum of squares

```
set.seed(3)
km output <- kmeans(x, centers = 3, nstart = 1)
km output$tot.withinss
## [1] 104.3319
km_output <- kmeans(x, centers = 3, nstart = 20)</pre>
km_output$tot.withinss
## [1] 97.97927
km output <- kmeans(x, centers = 3, nstart = 50)
km output$tot.withinss
```

## [1] 97.97927

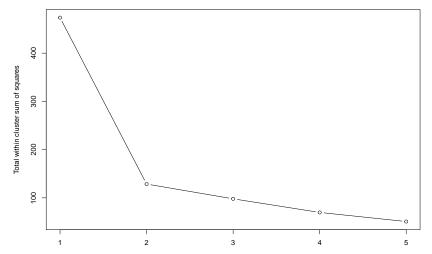
#### How to determine the number of clusters?

- **1** Run K-means with K=1, K=2, ..., K=n.
- 2 Record total within-cluster sum of squares for each value of K.
- **3** Choose *K* at the *elbow* position.

```
ks <- 1:5
tot_within_ss <- sapply(ks, function(k) {
    km_output <- kmeans(x, k, nstart = 20)
    km_output$tot.withinss
})</pre>
```

#### How to determine the number of clusters?

```
plot(ks, tot_within_ss, type = "b", xlab = "Values of K",
    ylab = "Total within cluster sum of squares")
```



#### Hierachical clustering using different linkage types

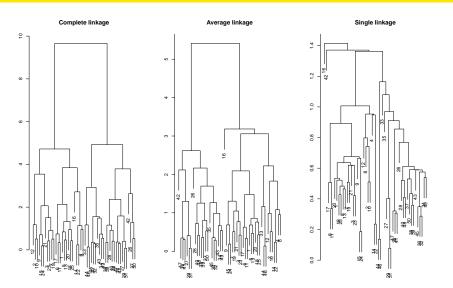
```
set.seed(2)
x <- data.table(a = rnorm(50), b = rnorm(50))
x[1:25, `:=`(a = a + 3, b = b - 4)]
hc_complete <- hclust(dist(x), method = "complete")
hc_average <- hclust(dist(x), method = "average")
hc_single <- hclust(dist(x), method = "single")</pre>
```

- Compute the inter-observation Euclidean distance matrix using dist()
- Set the linkage type in the method argument

#### Preparing the dendogram

```
par(mfrow = c(1, 3))
plot(hc_complete, main = "Complete linkage", xlab = "",
      ylab = "", sub = "")
plot(hc_average, main = "Average linkage", xlab = "",
      ylab = "", sub = "")
plot(hc_single, main = "Single linkage", xlab = "",
      ylab = "", sub = "")
```

#### Preparing the dendogram



# Determining cluster labels with cutree() by selecting k

```
[36] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
cutree(hc_average, k = 2)
  [36] 2 2 2 2 2 2 2 2 1 2 1 2 2 2 2
cutree(hc_single, k = 2)
```

cutree(hc complete, k = 2)

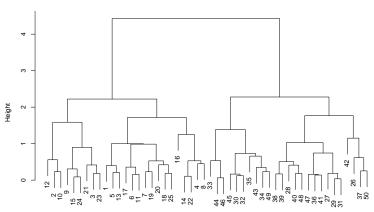
# Determining cluster labels with cutree() by selecting h

```
cutree(hc_average, h = 4)
 [36] 2 2 2 2 2 2 2 2 1 2 1 2 2 2 2
cutree(hc_single, h = 1.4)
```

cutree(hc complete, h = 5)

#### Scale variables with scale()

#### Hierarchical clustering with scaled features



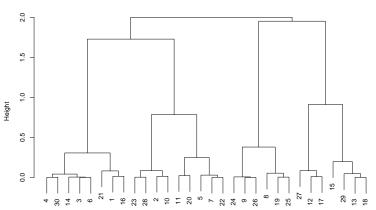
#### Compute correlation-based distance using as.dist()

```
set.seed(5)
x <- matrix(rnorm(30 * 3), ncol = 3)
dd <- as.dist(1 - cor(t(x)))</pre>
```

#### Compute correlation-based distance using as.dist()

```
plot(hclust(dd, method = "complete"),
    main = "Complete linkage with correlation-based distance"
    xlab = "", sub = "")
```

#### Complete linkage with correlation-based distance



#### Resources

#### Resources

- Uberoi, A. Introduction to Dimensionality Reduction.
- Jaadi, Z. A step by step explanation of principal component analysis.
- James, G., Witten, D., Hastie, T., and Tibshirani, R. An Introduction to Statistical Learning with Applications in R.
- Gatto, L. Chapter 4 Unsupervised Learning. In An Introduction to Machine Learning with R.
- Surles, W. Unsupervised Learning in R.