Inteligencia Artificial

UTN – FRVM

5º Año Ing. en Sistemas de Información

Agenda

- Supervised Learning
  - Naive Bayes
- Unsupervised Learning
  - PCA
  - Clustering
    - *K-Means*

Naive Bayes Classifier

- Abstractly, naive Bayes is a conditional probability model: given a problem instance to be classified, represented by a vector $\mathbf{x} = (x_1, \ldots, x_n)$ representing some *n* features (independent variables), it assigns to this instance probabilities

$$p(C_k \mid x_1, \ldots, x_n)$$

for each of *K* possible outcomes or *clases* C_k

Naive Bayes Classifier

- The problem with the above formulation is that if the number of features *n* is large or if a feature can take on a large number of values, then basing such a model on probability tables is infeasible. We therefore reformulate the model to make it more tractable. Using Bayes' theorem, the conditional probability can be decomposed as

Joint Probability

$$p(C_k \mid \mathbf{x}) = \frac{p(C_k)\, p(\mathbf{x} \mid C_k)}{p(\mathbf{x})} \quad \text{posterior} = \frac{\text{prior} \times \text{likelihood}}{\text{evidence}}$$

Naive Bayes Classifier

- In practice, there is interest only in the numerator of that fraction, because the denominator does not depend on C and the values of the features x_i are given, so that the denominator is effectively constant. The numerator is equivalent to the joint probability model $p(C_k, x_1, \ldots, x_n)$ which can be rewritten as follows, using the chain rule for repeated applications of the definition of conditional probability:

$$
\begin{aligned}
p(C_k, x_1, \ldots, x_n) &= p(x_1, \ldots, x_n, C_k) \\
&= p(x_1 \mid x_2, \ldots, x_n, C_k) p(x_2, \ldots, x_n, C_k) \\
&= p(x_1 \mid x_2, \ldots, x_n, C_k) p(x_2 \mid x_3, \ldots, x_n, C_k) p(x_3, \ldots, x_n, C_k) \\
&= \ldots \\
&= p(x_1 \mid x_2, \ldots, x_n, C_k) p(x_2 \mid x_3, \ldots, x_n, C_k) \ldots p(x_{n-1} \mid x_n, C_k) p(x_n \mid C_k) p(C_k)
\end{aligned}
$$

Naive Bayes Classifier

- Now the "naive" conditional independence assumptions come into play: assume that each feature x_i is conditionally independent of every other feature x_j for $j \neq i$ given the category C_k
- This means that $p(x_i \mid x_{i+1}, \ldots, x_n, C_k) = p(x_i \mid C_k)$

$$
\begin{aligned}
p(C_k \mid x_1, \ldots, x_n) &\propto p(C_k, x_1, \ldots, x_n) = \\
&= p(C_k)\, p(x_1 \mid C_k)\, p(x_2 \mid C_k)\, p(x_3 \mid C_k)\, \cdots \\
&= p(C_k) \prod_{i=1}^{n} p(x_i \mid C_k)\,.
\end{aligned}
$$

Constructing a classifier from the probability model

- The discussion so far has derived the independent feature model, that is, the naive Bayes probability model. The naive Bayes classifier combines this model with a decision rule. One common rule is to pick the hypothesis that is most probable; this is known as the *maximum a posteriori* or *MAP* decision rule. The corresponding classifier, a Bayes classifier, is the function that assigns a class label

$\hat{y} = C_k$

$$\hat{y} = \underset{k \in \{1,\dots,K\}}{\operatorname{argmax}} \; p(C_k) \prod_{i=1}^{n} p(x_i \mid C_k).$$

Example

- ***Problem***: classify whether a given person is a male or a female based on the measured features. The features include height, weight, and foot size.

Person	height (feet)	weight (lbs)	foot size(inches)
male	6	180	12
male	5.92 (5'11")	190	11
male	5.58 (5'7")	170	12
male	5.92 (5'11")	165	10
female	5	100	6
female	5.5 (5'6")	150	8
female	5.42 (5'5")	130	7
female	5.75 (5'9")	150	9

The classifier created from the training set using a Gaussian distribution assumption would be (given variances are *unbiased* sample variances):

Person	mean (height)	variance (height)	mean (weight)	variance (weight)	mean (foot size)	variance (foot size)
male	5.855	3.5033*10-02	176.25	1.2292*10+02	11.25	9.1667*10-01
female	5.4175	9.7225*10-02	132.5	5.5833*10+02	7.5	1.6667

Example

- **Testing:** Below is a sample to be classified as male or female.

Person	height (feet)	weight (lbs)	foot size(inches)
sample	6	130	8

- We wish to determine which posterior is greater, male or female.

$$\text{posterior (male)} = \frac{P(\text{male})\, p(\text{height} \mid \text{male})\, p(\text{weight} \mid \text{male})\, p(\text{foot size} \mid \text{male})}{evidence}$$

$$\text{posterior (female)} = \frac{P(\text{female})\, p(\text{height} \mid \text{female})\, p(\text{weight} \mid \text{female})\, p(\text{foot size} \mid \text{female})}{evidence}$$

$$\begin{aligned}\text{evidence} = {} & P(\text{male})\, p(\text{height} \mid \text{male})\, p(\text{weight} \mid \text{male})\, p(\text{foot size} \mid \text{male}) \\ & + P(\text{female})\, p(\text{height} \mid \text{female})\, p(\text{weight} \mid \text{female})\, p(\text{foot size} \mid \text{female})\end{aligned}$$

Example

- However, given the sample, the evidence is a constant and thus scales both posteriors equally. It therefore does not affect classification and can be ignored. We now determine the probability distribution for the sex of the sample.

$$P(\text{male}) = 0.5$$

$$p(\text{height} \mid \text{male}) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{-(6-\mu)^2}{2\sigma^2}\right) \approx 1.5789$$

$\mu = 5.855$

$\sigma^2 = 3.5033 \cdot 10^{-2}$

$$p(\text{weight} \mid \text{male}) = 5.9881 \cdot 10^{-6}$$
$$p(\text{foot size} \mid \text{male}) = 1.3112 \cdot 10^{-3}$$
$$\text{posterior numerator (male)} = \text{their product} = 6.1984 \cdot 10^{-9}$$
$$P(\text{female}) = 0.5$$
$$p(\text{height} \mid \text{female}) = 2.2346 \cdot 10^{-1}$$
$$p(\text{weight} \mid \text{female}) = 1.6789 \cdot 10^{-2}$$
$$p(\text{foot size} \mid \text{female}) = 2.8669 \cdot 10^{-1}$$
$$\text{posterior numerator (female)} = \text{their product} = 5.3778 \cdot 10^{-4}$$

Unsupervised Learning

- Unsupervised learning is often much more challenging that the supervised variant.
- It tends to be more subjective, and there is no simple goal for the analysis, such as prediction of a response.
- Unsupervised learning is often performed as part of an exploratory data analysis.
- Furthermore, it can be hard to assess the results obtained from unsupervised learning methods, since there is no universally accepted mechanism for performing cross-validation or validating results on an independent data set.

Unsupervised learning

- Techniques for unsupervised learning are of growing importance in a number of fields.
- A **cancer researcher** might assay gene expression levels in 100 patients with breast cancer. He or she might then look for subgroups among the breast cancer samples, or among the genes, in order to obtain a better understanding of the disease.
- An **online shopping** site might try to identify groups of shoppers with similar browsing and purchase histories, as well as items that are of particular interest to the shoppers within each group. Then an individual shopper can be preferentially shown the items in which he or she is particularly likely to be interested, based on the purchase histories of similar shoppers.
- A **search engine** might choose what search results to display to a particular individual based on the click histories of other individuals with similar search patterns. These statistical learning tasks, and many more, can be performed via unsupervised learning techniques.

PCA

- *Principal component analysis* (PCA) refers to the process by which principal components are computed, and the subsequent use of these components in understanding the data.
- PCA is an unsupervised approach, since it involves only a set of features $X_1, X_2, \ldots, X_p$, and no associated response Y.
- Apart from producing derived variables for use in supervised learning problems, PCA also serves as a tool for data visualization (visualization of the observations or visualization of the variables)

PCA

- In particular, we would like to find a low-dimensional representation of the data that captures as much of the information as possible.
- For instance, if we can obtain a two-dimensional representation of the data that captures most of the information, then we can plot the observations in this low-dimensional space.

PCA

- PCA provides a tool to do just this. It finds a low-dimensional representation of a data set that contains as much as possible of the variation.
- The idea is that each of the *n* observations lives in *p*-dimensional space, but not all of these dimensions are equally interesting.
- PCA seeks a small number of dimensions that are as interesting as possible, where the concept of *interesting* is measured by the amount that the observations vary along each dimension.
- Each of the dimensions found by PCA is a linear combination of the *p* features.

PCA

- The *first principal component* of a set of features $X_1, X_2, \ldots, X_p$ is the normalized linear combination of the features that has the largest variance.

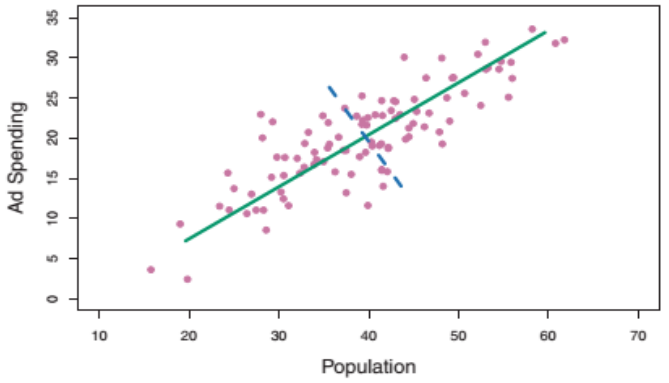
$$Z_1 = \phi_{11} X_1 + \phi_{21} X_2 + \ldots + \phi_{p1} X_p$$

- By *normalized*, we mean that $\sum_{j=1}^{p} \phi_{j1}^2 = 1$
- We refer to $\phi_{11}, \ldots, \phi_{p1}$ as the *loadings* of the first principal component.
- Principal components vector $\phi_1 = (\phi_{11} \ \phi_{21} \ \ldots \ \phi_{p1})^T$

PCA

- Given a $n \times p$ data set **X**, how do we compute the first principal component?
- Since we are only interested in variance, we assume that each of the variables in **X** has been centered to have mean zero (that is, the column means of **X** are zero).
- We then look for the linear combination of the sample feature values of the form $z_{i1} = \phi_{11} x_{i1} + \phi_{21} x_{i2} + \ldots + \phi_{p1} x_{ip}$ that has largest sample variance.
- In other words, the first principal component loading vector solves the optimization problem.

$$\underset{\phi_{11},\ldots,\phi_{p1}}{\text{maximize}} \left\{ \frac{1}{n} \sum_{i=1}^{n} \left(\sum_{j=1}^{p} \phi_{j1} x_{ij} \right)^2 \right\} \text{ subject to } \sum_{j=1}^{p} \phi_{j1}^2 = 1.$$



- Hence the objective that we are maximizing in is just the sample variance of the *n* values of *zi1*
- We refer to *z*11, . . . , *zn*1 as the *scores* of the first principal component.

PCA

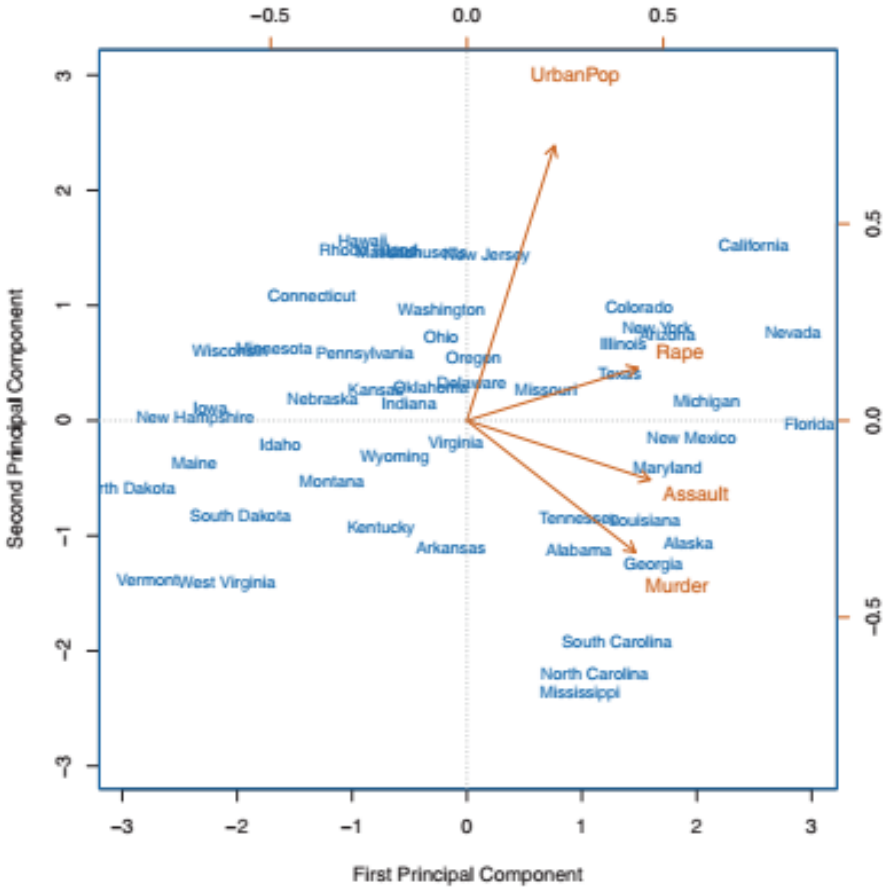
- After the first principal component Z_1 of the features has been determined, we can find the second principal component Z_2. The second principal component is the linear combination of $X_1, \ldots, X_p$ that has maximal variance out of all linear combinations that are *uncorrelated* with Z_1 (is orthogonal).

- To find $\boldsymbol{\varphi 2}$, we solve a problem similar to the shown before with $\varphi 2$ replacing $\varphi 1$, and with the additional constraint that $\boldsymbol{\varphi 2}$ is orthogonal to $\boldsymbol{\varphi 1}$.

- Once we have computed the principal components, we can plot them against each other in order to produce low-dimensional views of the data.

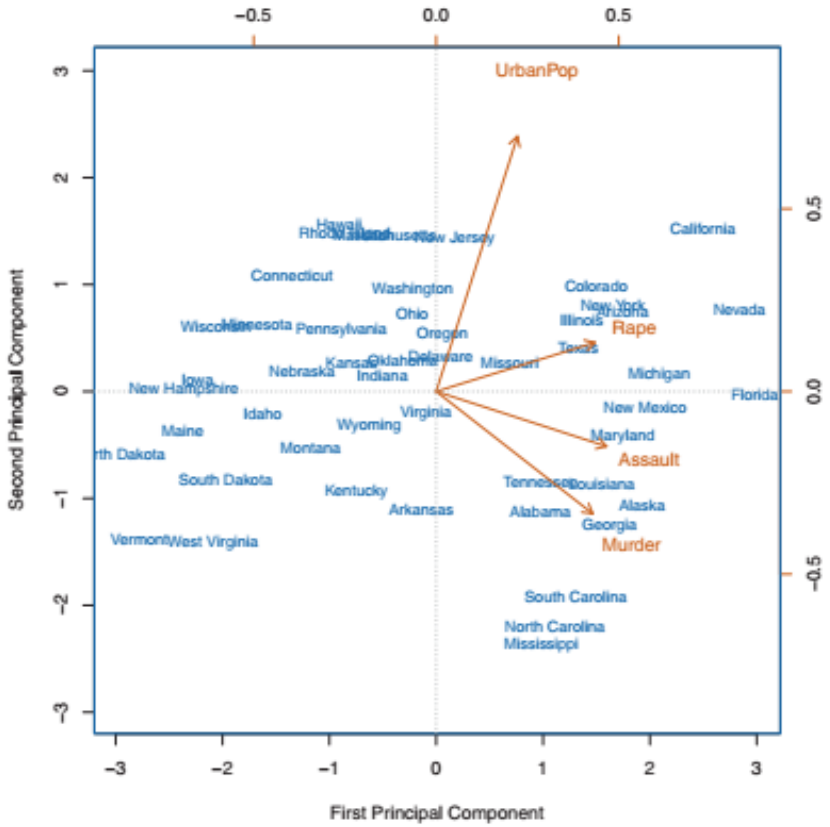
- Geometrically, this amounts to projecting the original data down onto the subspace spanned by $\boldsymbol{\varphi 1}$, $\boldsymbol{\varphi 2}$, and $\boldsymbol{\varphi 3}$, and plotting the projected points.

PCA



- *The first two principal components for the* USArrests *data.*
- *The blue state names represent the scores for the first two principal components. The orange arrows indicate the first two principal component loading vectors (with axes on the top and right).*
- *For example, the loading for* Rape *on the first component is* 0.54*, and its loading on the second principal component* 0.17 *(the word* Rape *is centered at the point* (0.54, 0.17)*). This figure is known as a biplot, because it displays both the principal component scores and the principal component loadings.*

PCA



- We see that the first loading vector places approximately equal weight on **Assault**, **Murder**, and **Rape**, with much less weight on **UrbanPop**. Hence this component roughly corresponds to a measure of overall rates of serious crimes.

- Overall, we see that the crime-related variables (**Murder**, **Assault**, and **Rape**) are located close to each other, and that the **UrbanPop** variable is far from the other three. This indicates that the crime-related variables are correlated with each other—states with high murder rates tend to have high assault and rape rates—and that the **UrbanPop** variable is less correlated with the other three.

- States with large positive scores on the first component, such as **California**, **Nevada** and **Florida**, have high crime rates, while states like **North Dakota**, with negative scores on the first component, have low crime rates. **California** also has a high score on the second component, indicating a high level of urbanization, while the opposite is true for states like **Mississippi**. States close to zero on both components, such as **Indiana**, have approximately average levels of both crime and urbanization.

PVE

- How much of the information in a given data set is lost by projecting the observations onto the first few principal components?

- That is, how much of the **variance** in the data is *not* contained in the first few principal components? More generally, we are interested in knowing the *proportion of variance explained* (PVE) by each proportion of variance explained principal component.

PVE

- The *total variance* present in a data set (assuming that the variables have been centered to have mean zero) is defined as

$$\sum_{j=1}^{p} \mathrm{Var}(X_j) = \sum_{j=1}^{p} \frac{1}{n} \sum_{i=1}^{n} x_{ij}^2.$$

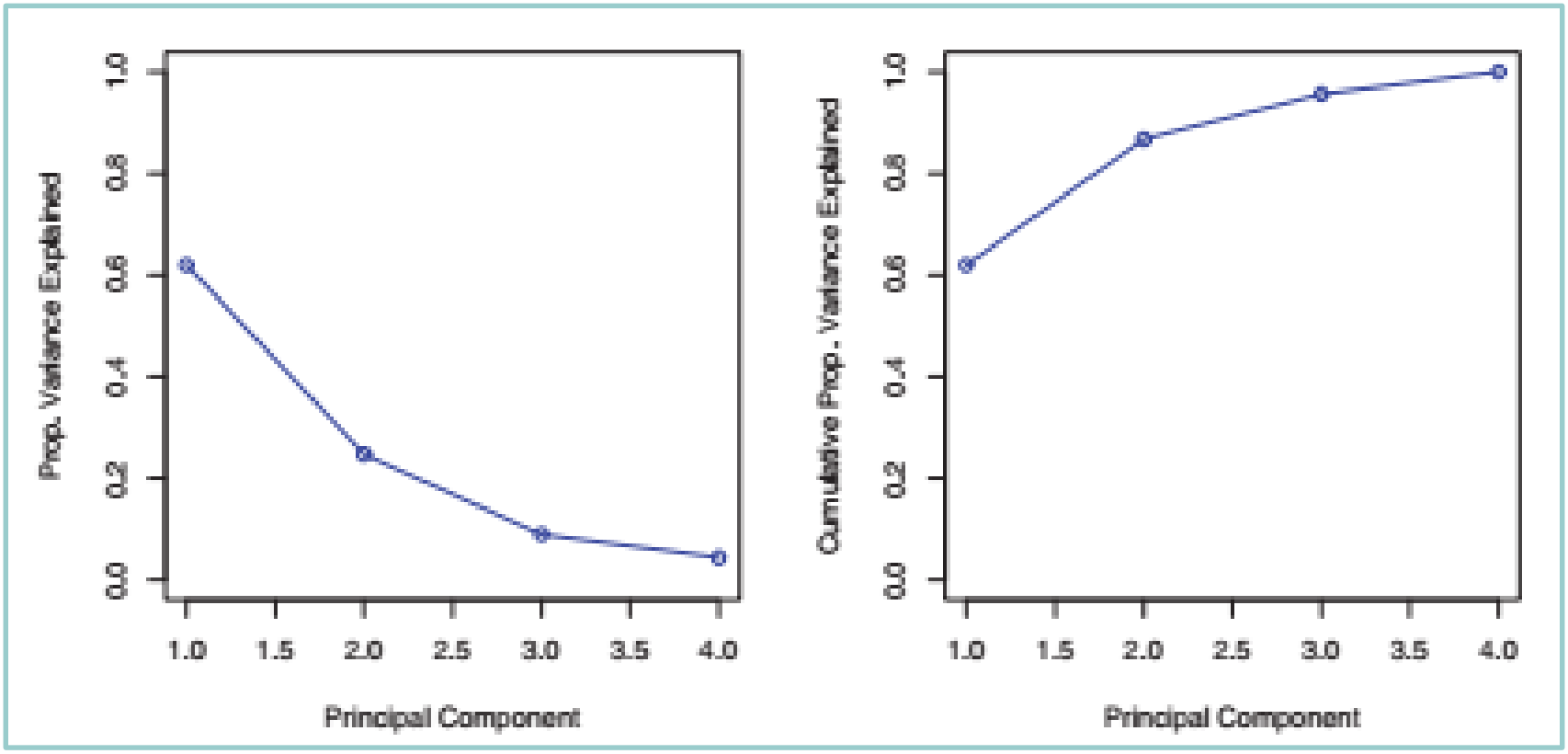
- m_{th} principal component

$$\frac{1}{n} \sum_{i=1}^{n} z_{im}^2 = \frac{1}{n} \sum_{i=1}^{n} \left(\sum_{j=1}^{p} \phi_{jm} x_{ij} \right)^2$$

- PVE m_{th} PC

$$\frac{\sum_{i=1}^{n} \left(\sum_{j=1}^{p} \phi_{jm} x_{ij} \right)^2}{\sum_{j=1}^{p} \sum_{i=1}^{n} x_{ij}^2}.$$

PVE


Prop. Variance Explained
0.0
0.2
0.4
0.6
0.8
1.0
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Principal Component
Cumulative Prop. Variance Explained
0.0
0.2
0.4
0.6
0.8
1.0
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Principal Component

Clustering

- ***Clustering*** refers to a very broad set of techniques for finding *subgroups*, or *clusters*, in a data set.
- When we cluster the observations of a data set, we seek to partition them into distinct groups so that the observations within each group are quite similar to each other, while observations in different groups are quite different from each other.
- Of course, to make this concrete, we must define what it means for two or more observations to be ***similar*** or ***different***.
- Indeed, this is often a domain-specific consideration that must be made based on knowledge of the data being studied.

Clustering

- An application of clustering arises in marketing.
- We may have access to a large number of measurements (e.g. median household income, occupation, distance from nearest urban area, and so forth) for a large number of people.
- Our goal is to perform *market segmentation* by identifying subgroups of people who might be more receptive to a particular form of advertising, or more likely to purchase a particular product.
- The task of performing market segmentation amounts to clustering the people in the data set.

K-Means Clustering

- ***K-means*** clustering is a simple and elegant approach for partitioning a data set into *K* distinct, non-overlapping clusters. 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

- Let $C_1, \ldots, C_K$ denote sets containing the indices of the observations in each cluster. These sets satisfy two properties:
  - $C_1 \cup C_2 \cup \ldots \cup C_K = \{1, \ldots, n\}$. In other words, each observation belongs to at least one of the K clusters.
  - $C_k \cap C_{k'} = \emptyset$ for all $k \neq k'$. In other words, the clusters are nonoverlapping: no observation belongs to more than one cluster.

K-Means

- The idea behind *K*-means clustering is that a *good* clustering is one for which the *within-cluster variation* is as small as possible.
- The within-cluster variation for cluster C_k is a measure $W(C_k)$ of the amount by which the observations within a cluster differ from each other.
- Hence we want to solve the problem

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

K-Means

- We want to partition the observations into *K* clusters such that the total ***within-cluster variation***, summed over all *K* clusters, is as small as possible.
- In order to make it actionable we need to define the ***within-cluster variation***. There are many possible ways to define this concept, but by far the most common choice involves *squared Euclidean distance*.
- That is, we define

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

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

K-Means

- This is in fact a very difficult problem to solve precisely, since there are almost K^n ways to partition *n* observations into *K* clusters.
- Fortunately, a very simple algorithm can be shown to provide a local optimum—a *pretty good solution*—to the *K*-means optimization problem.

K-Means Clustering

1. 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).

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

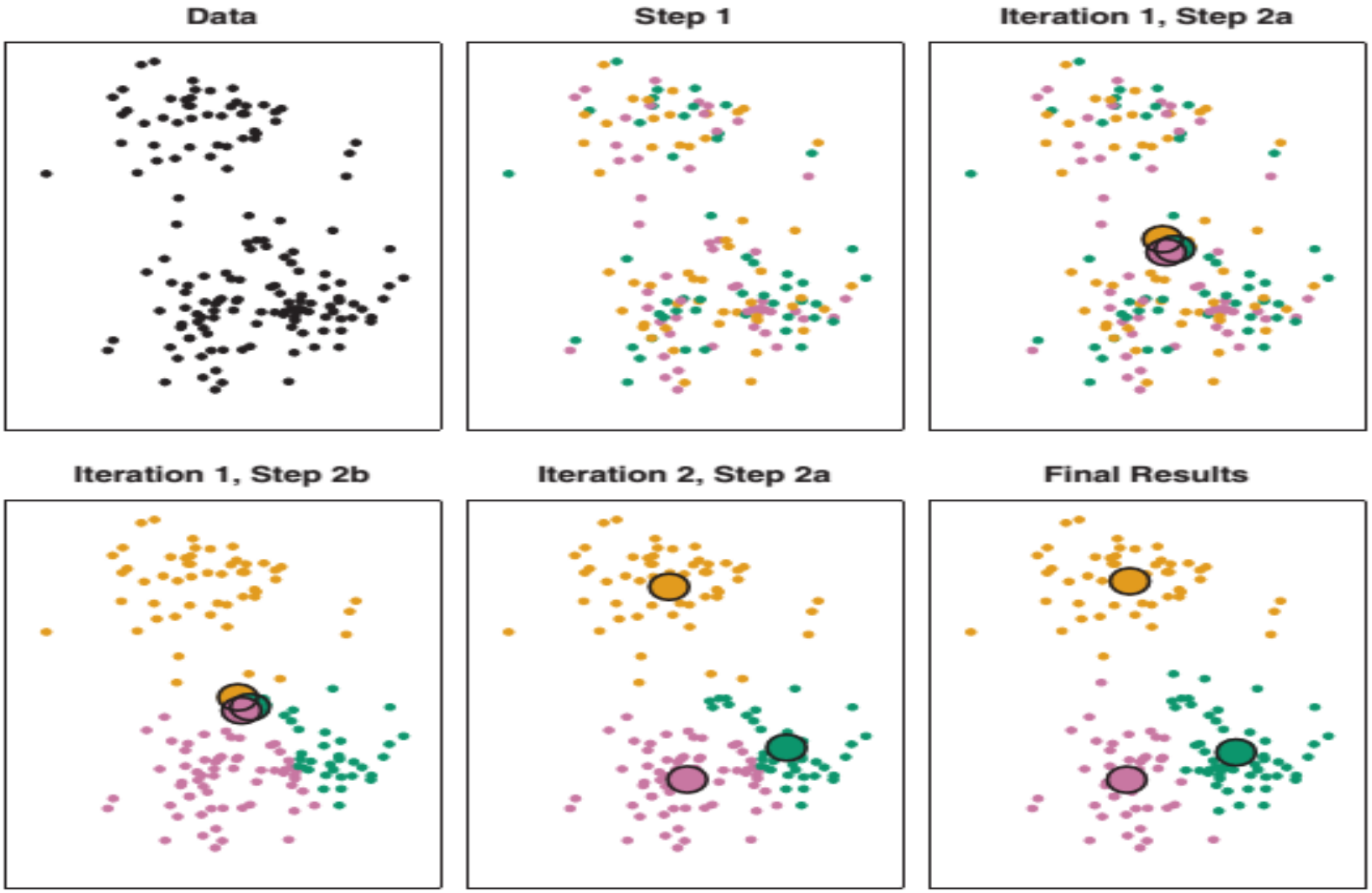
$$\bar{x}_{kj} = \frac{1}{|C_k|}\sum_{i \in C_k} x_{ij}$$

K-Means

- This means that as the algorithm is run, the clustering obtained will continually improve until the result no longer changes;

- The objective of the equation before will never increase. When the result no longer changes, a *local optimum* has been reached.

Example


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

K-means

- Because the *K*-means algorithm finds a local rather than a global optimum, the results obtained will depend on the initial (random) cluster assignment of each observation in Step 1 of Algorithm. For this reason, it is important to run the algorithm multiple times from different random initial configurations. Then one selects the *best* solution, i.e. that for which the objective is smallest