UNIVERSITÄT DUISBURG ESSEN

Open-Minded

Mahalanobis Distance

Neuroinformatics Tutorial 4

Duc Duy Pham[1]

[1]Intelligent Systems, Faculty of Engineering,
University of Duisburg-Essen, Germany

Content

- Revision: Naive Bayes Classifer
- Revision: Lecture
- Mahalanobis Distance

Content

- Revision: Naive Bayes Classifer
- Revision: Lecture
- Mahalanobis Distance

Revision: Naive Bayes Classifier

- What is the main goal of the Naive Bayes Classifier

Revision: Naive Bayes Classifier

- What is the naive assumption of the Naive Bayes Classifier

Revision: Naive Bayes Classifier

- Which values need to be calculated during training for the Gaussian Naive Bayes Classifier?

Naive Bayes Classifier

- Given a labeled training set, how do we get these probabilities?

$$\underset{k=0,\ldots,K}{\operatorname{argmax}}[P(C_k) \cdot P(x_1|C_k) \cdot P(x_2|C_k) \cdots P(x_n|C_k)]$$

Prior of class C_k:
Number of class occurences in data set divided by number of all samples in data set

Likelihoods of all features, given class C_k

For each feature/class combination, we need a (gaussian) distribution model!
This way we can calculate the probability during inference!

Estimation of Likelihood

- In practice quite important:
  - Estimation of Likelihood P(feature | class)
  - Easy for categorical features:
  - But what about continuous features?

  - P(Temp=19.5°C | Class = Rainy) = 0% ?

  - => Need to estimate underlying distribution!
  - Assume Gaussian
    => Mean Temp (Given Class = Rainy) = 19°C
    => Variance (Given Class = Rainy) = $\frac{2}{3}$

    => P(Temp=19.5°C | Class = Rainy) = $\frac{1}{\sqrt{2\pi\frac{2}{3}}} e^{-\frac{(19.5-19)^2}{2\cdot\frac{2}{3}}}$

$$\frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

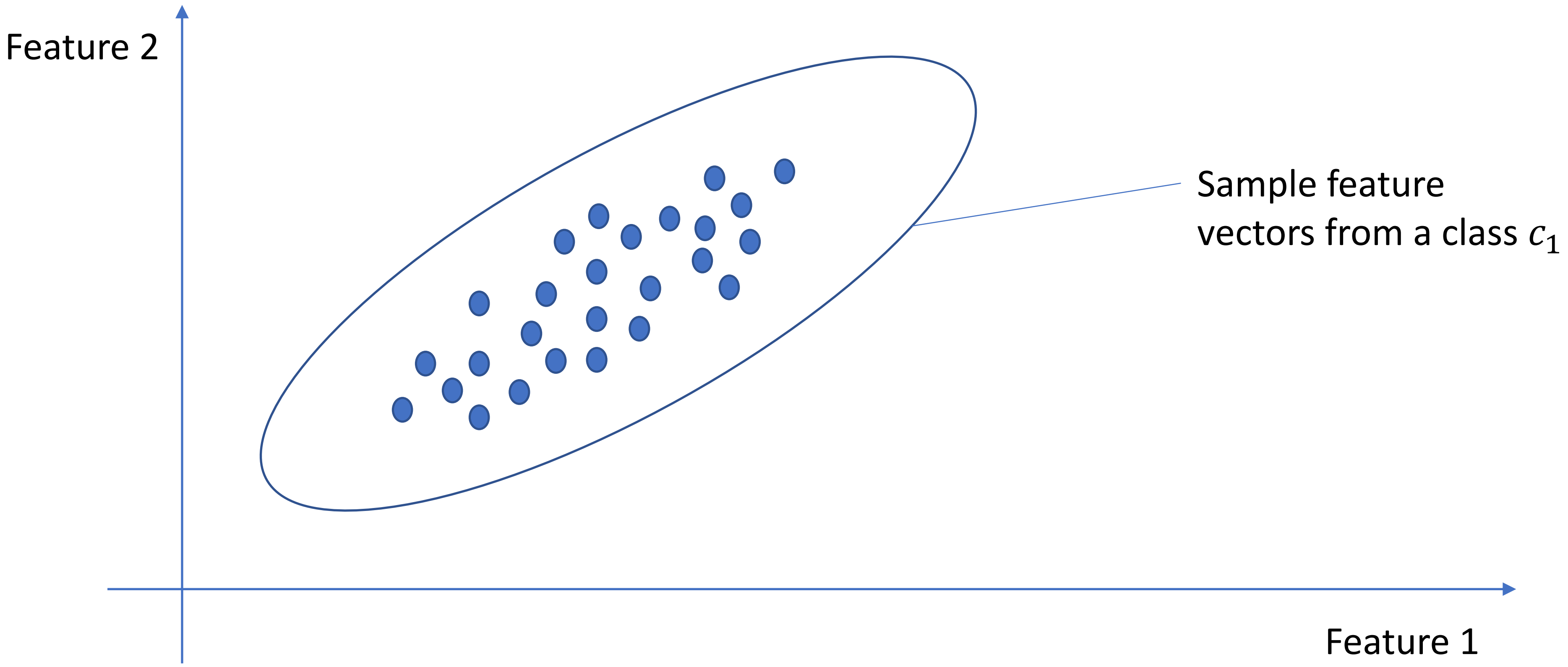
Temp.	Class
19°C	Rainy
18°C	Rainy
20°C	Rainy
21°C	Sunny
22°C	Sunny
24°C	Sunny

Naive Bayes Classifier: Jupyter

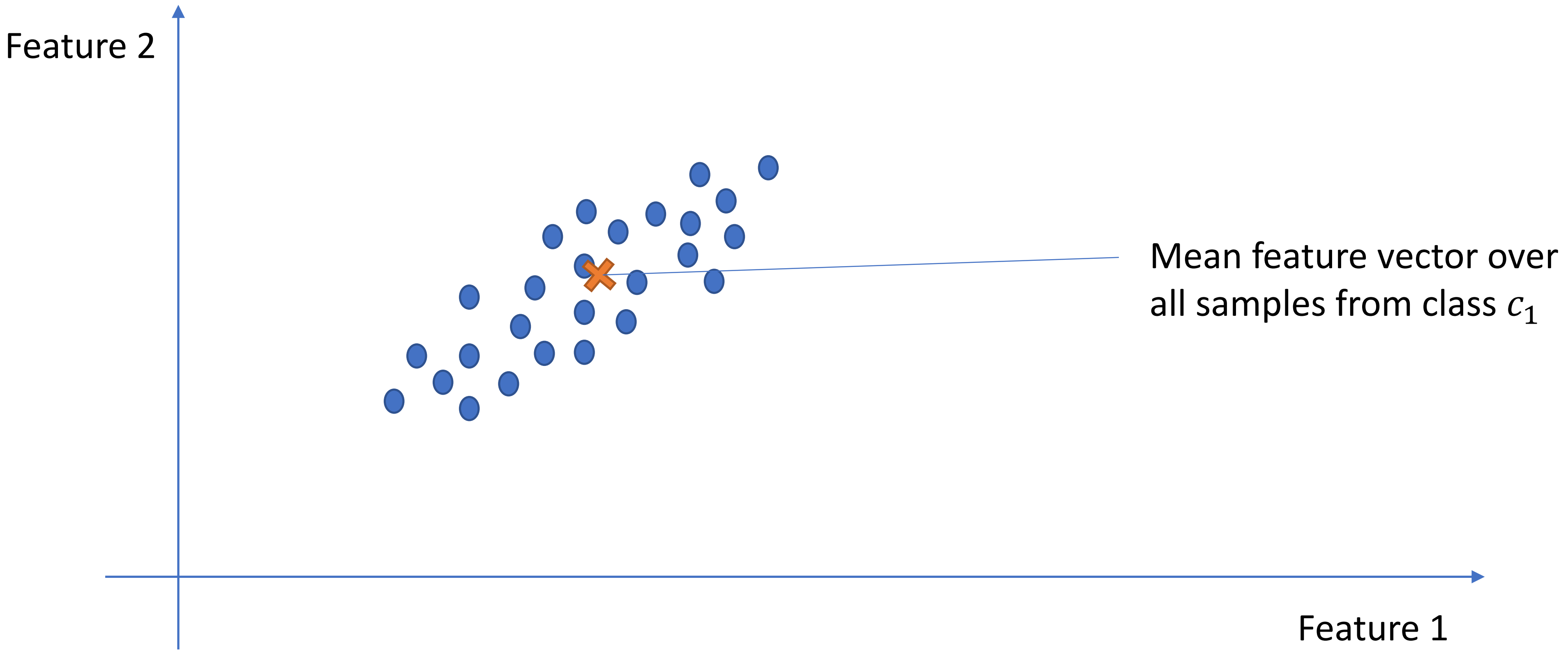
Revision: Lecture

- Which statements regarding discrimination functions are true in the context of classification?
    1. The discrimination function evaluates an input feature vector for a given class
    2. The discrimination function depends on the class
    3. A classifier chooses the class that maximizes the discrimination function
    4. The log-likelihood can be used as a discrimination function

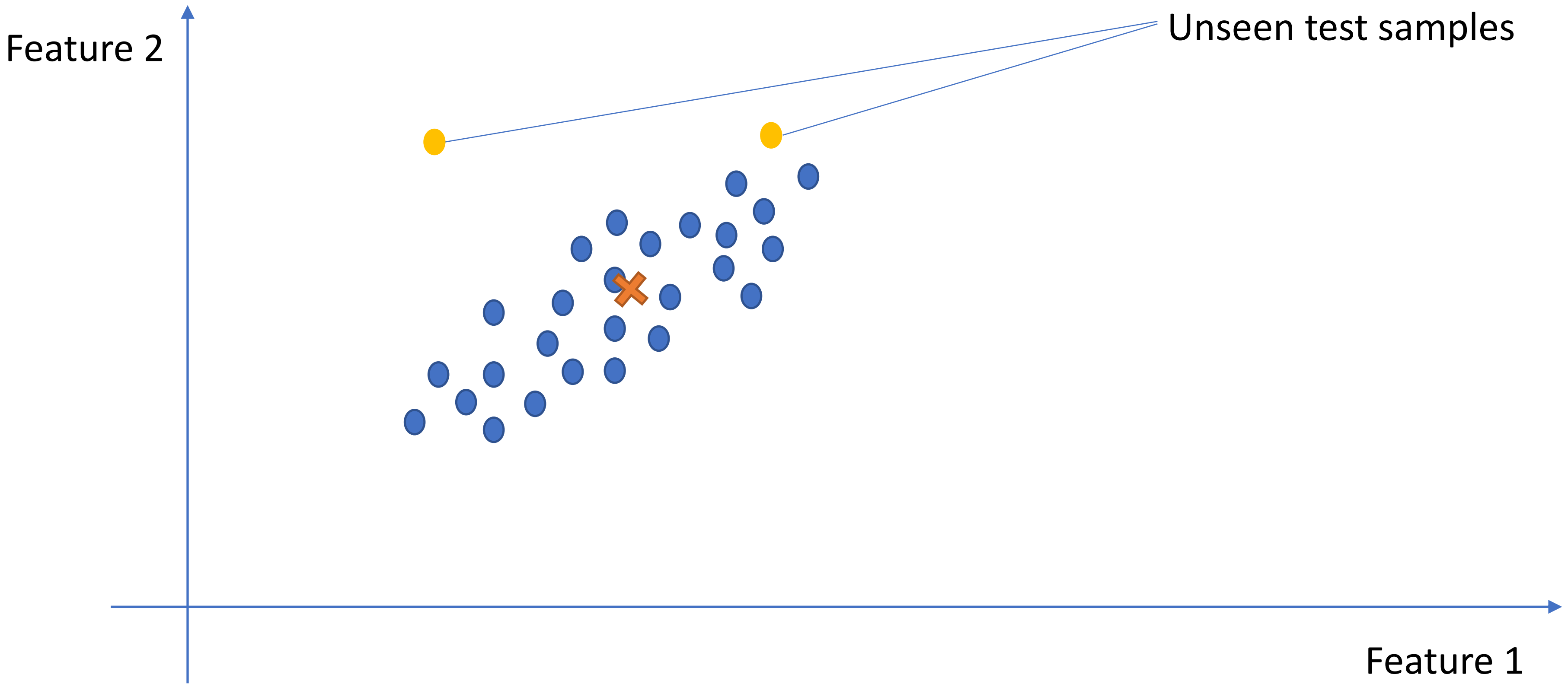
Mahalanobis Distance: Motivation



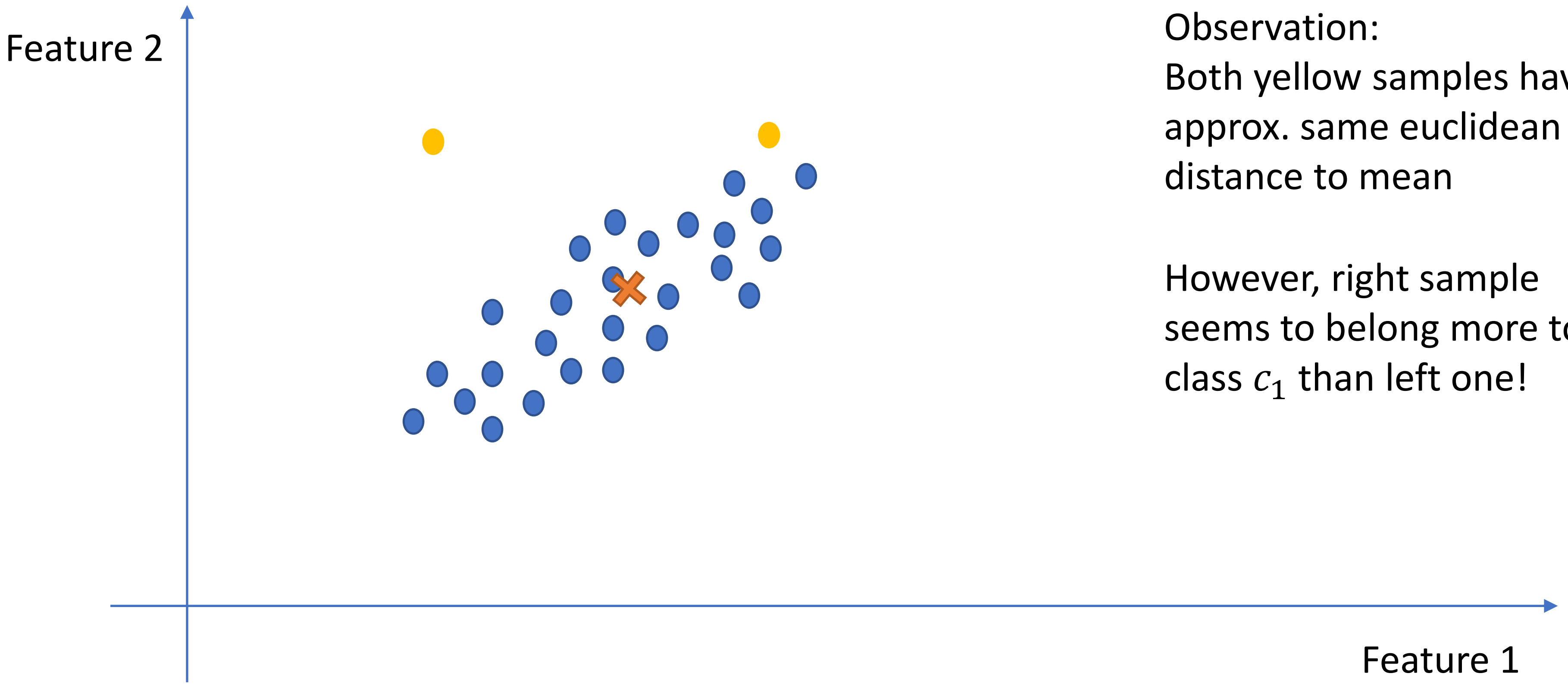
Mahalanobis Distance: Motivation



Mahalanobis Distance: Motivation



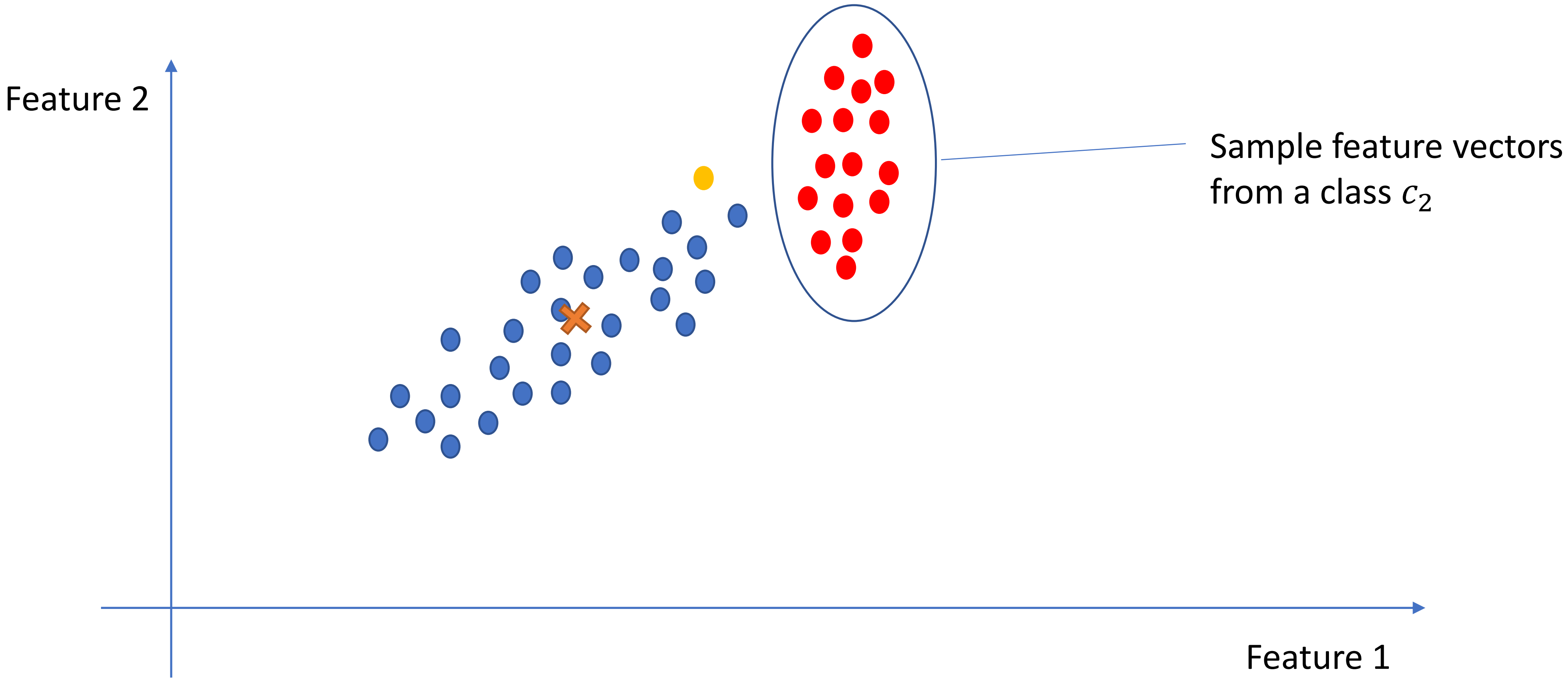
Mahalanobis Distance: Motivation



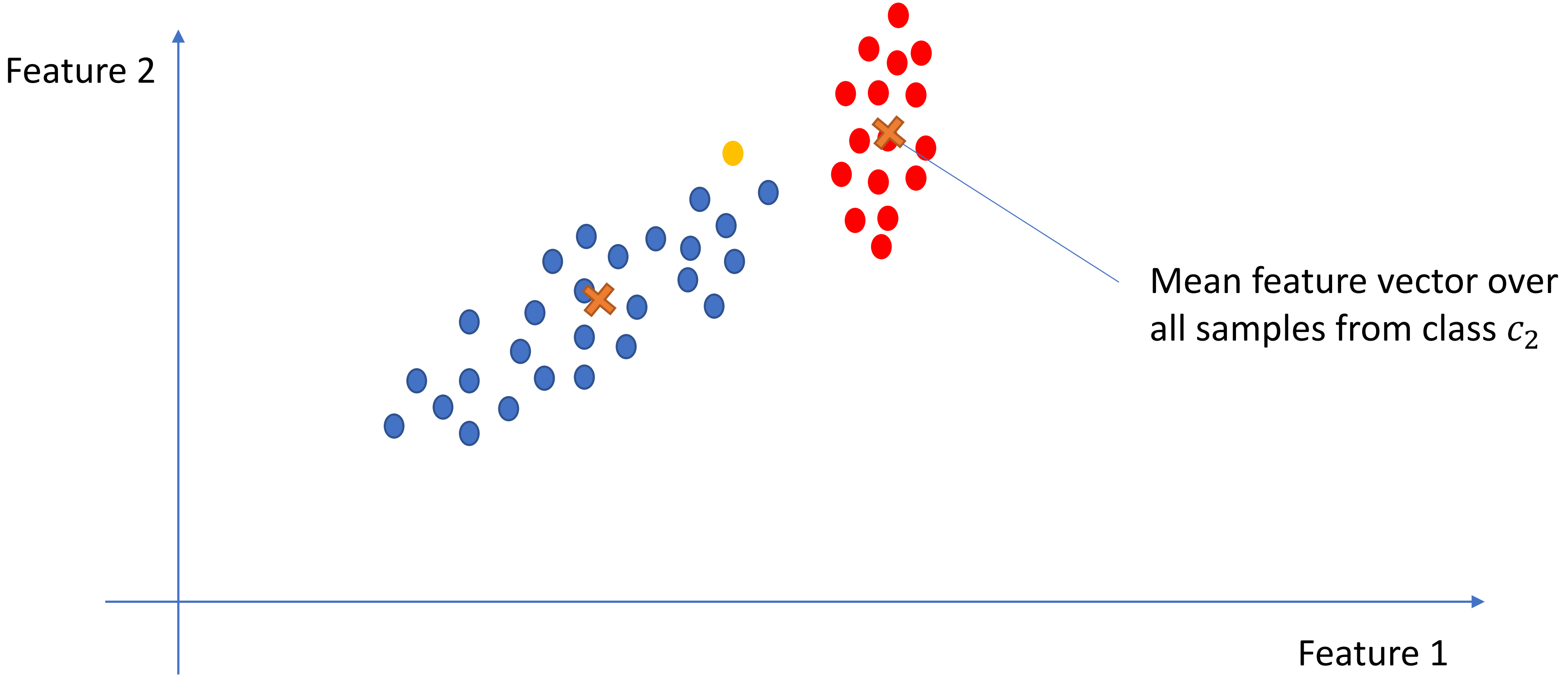
Observation:
Both yellow samples have approx. same euclidean distance to mean

However, right sample seems to belong more to class c_1 than left one!

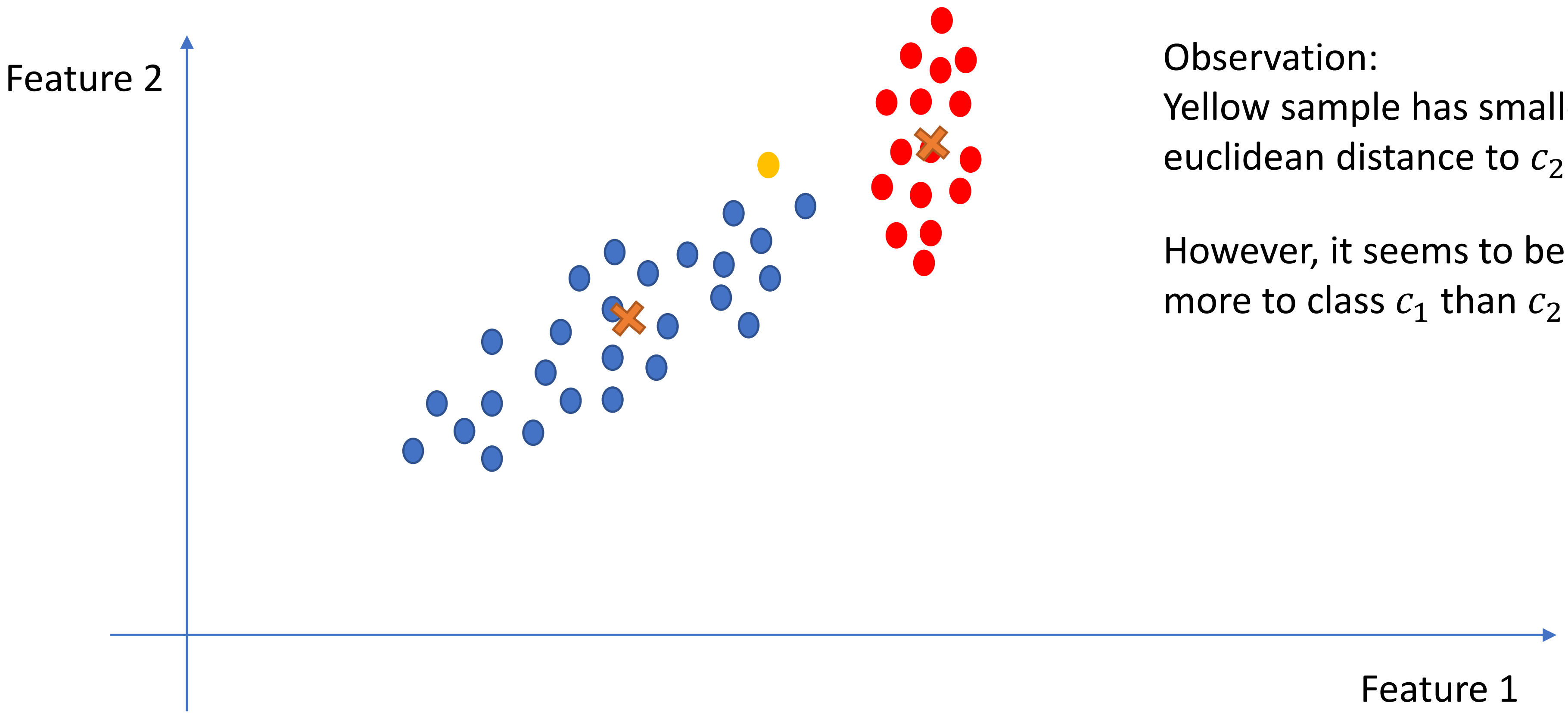
Mahalanobis Distance: Motivation



Mahalanobis Distance: Motivation



Mahalanobis Distance: Motivation



Observation:
Yellow sample has smaller euclidean distance to c_2 mean

However, it seems to belong more to class c_1 than c_2 !

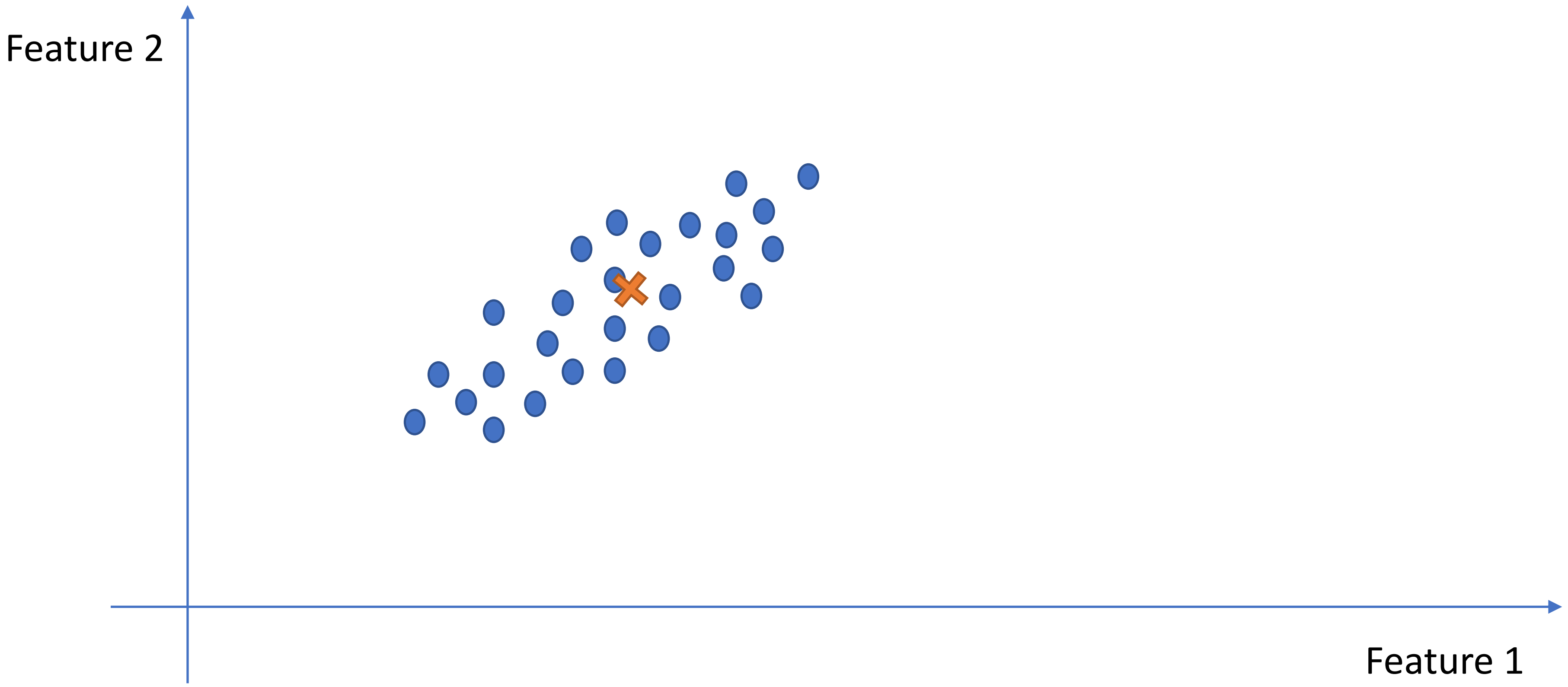
Prerequisites: Covariance

- Given two real valued random variables, the covariance is defindes as:

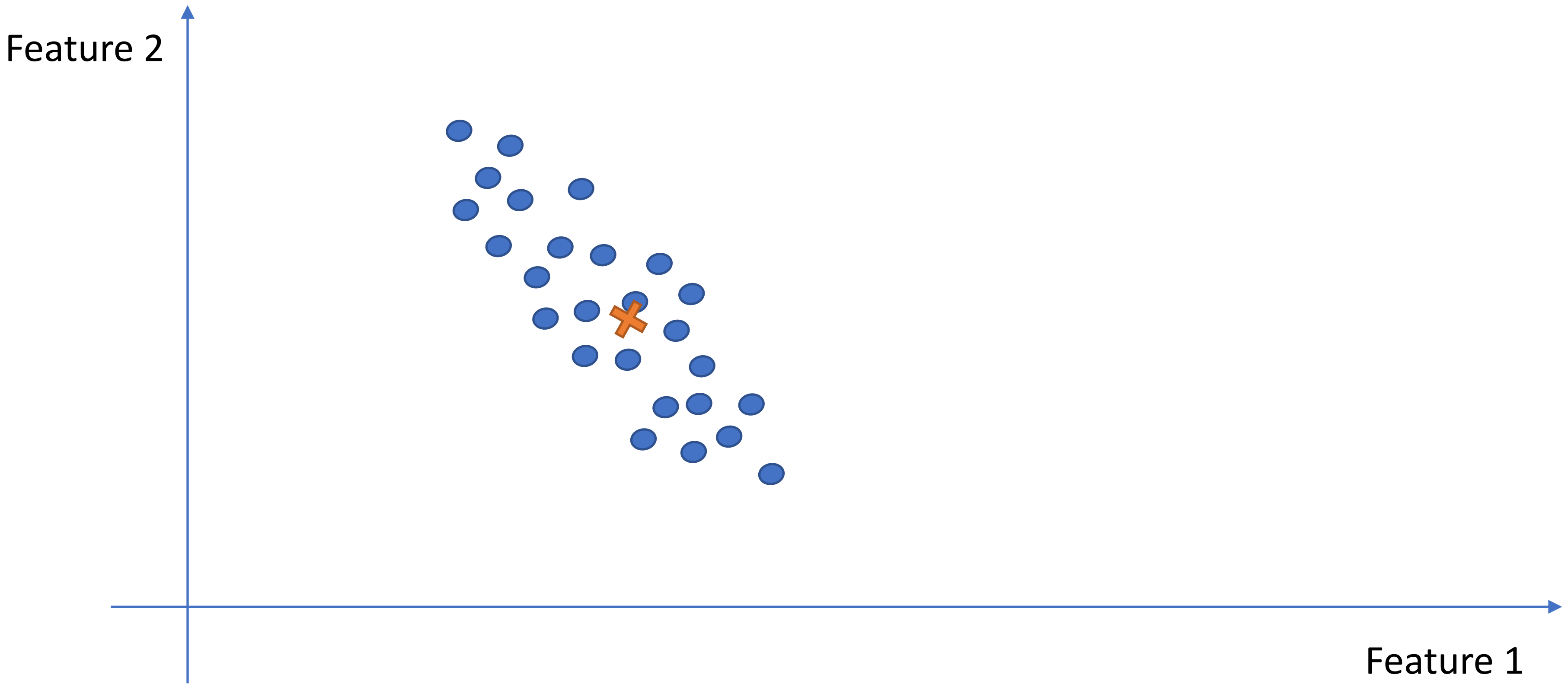
$$\mathrm{Cov}(X, Y) = \mathbb{E}[(X - \mathbb{E}[X])(Y - \mathbb{E}[Y])]$$

- Covariance is positive, if there is a positive linear dependency
- Covariance is negative, if there is a negative linear dependency
- Covariance is zero, if there is no linear dependency
  (But there can be a non-linear dependency!)
- Features/Measurements can be statistically represented by random variables!
  (Random variables map a value to probabilistic events!)

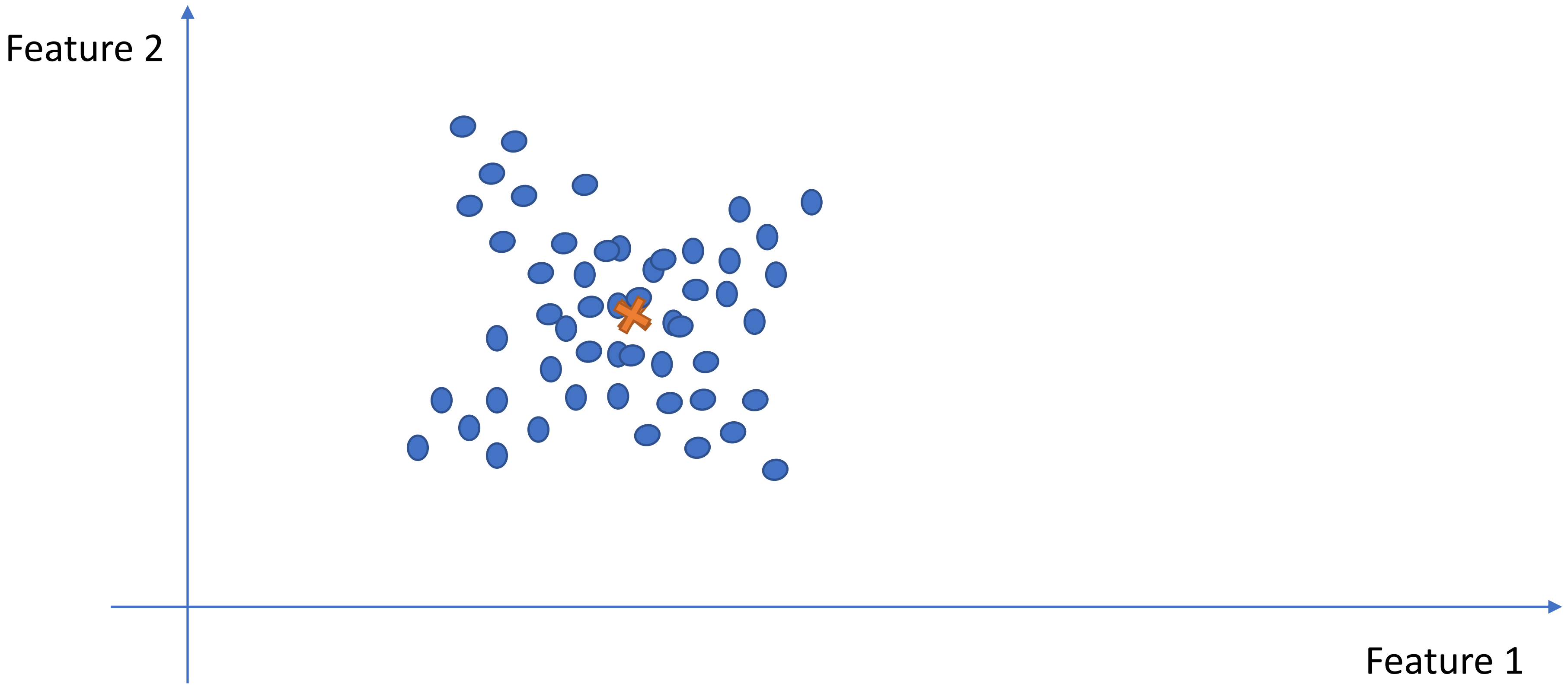
Positive Covariance



Negative Covariance



Zero Covariance



Excursion: Pearson Correlation Coefficient

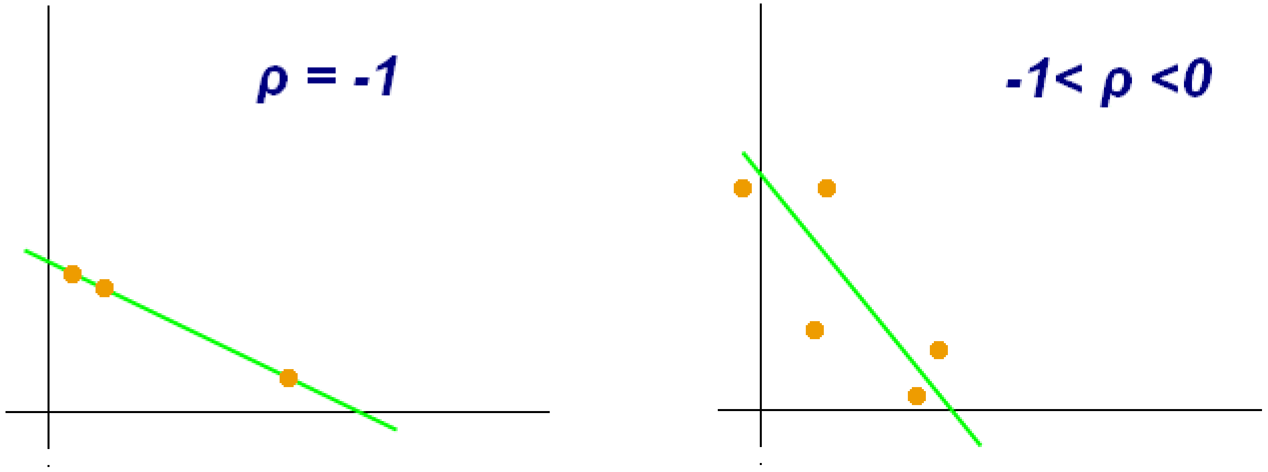
- Covariance does not yield information of the degree of dependency! – Only the direction!
- For information of degree, covariance needs to be normed!

- **Pearson Correlation Coefficient:**

$$\rho_{X,Y} = \frac{\mathrm{Cov}(X,Y)}{\sigma_X \sigma_Y}$$

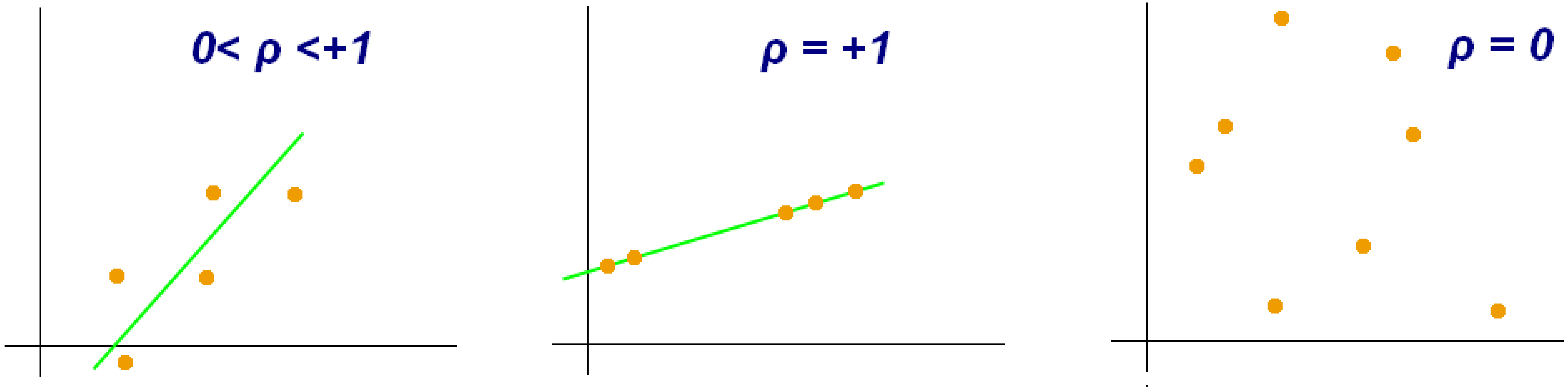
Standard deviations

Excursion: Pearson Correlation Coefficient



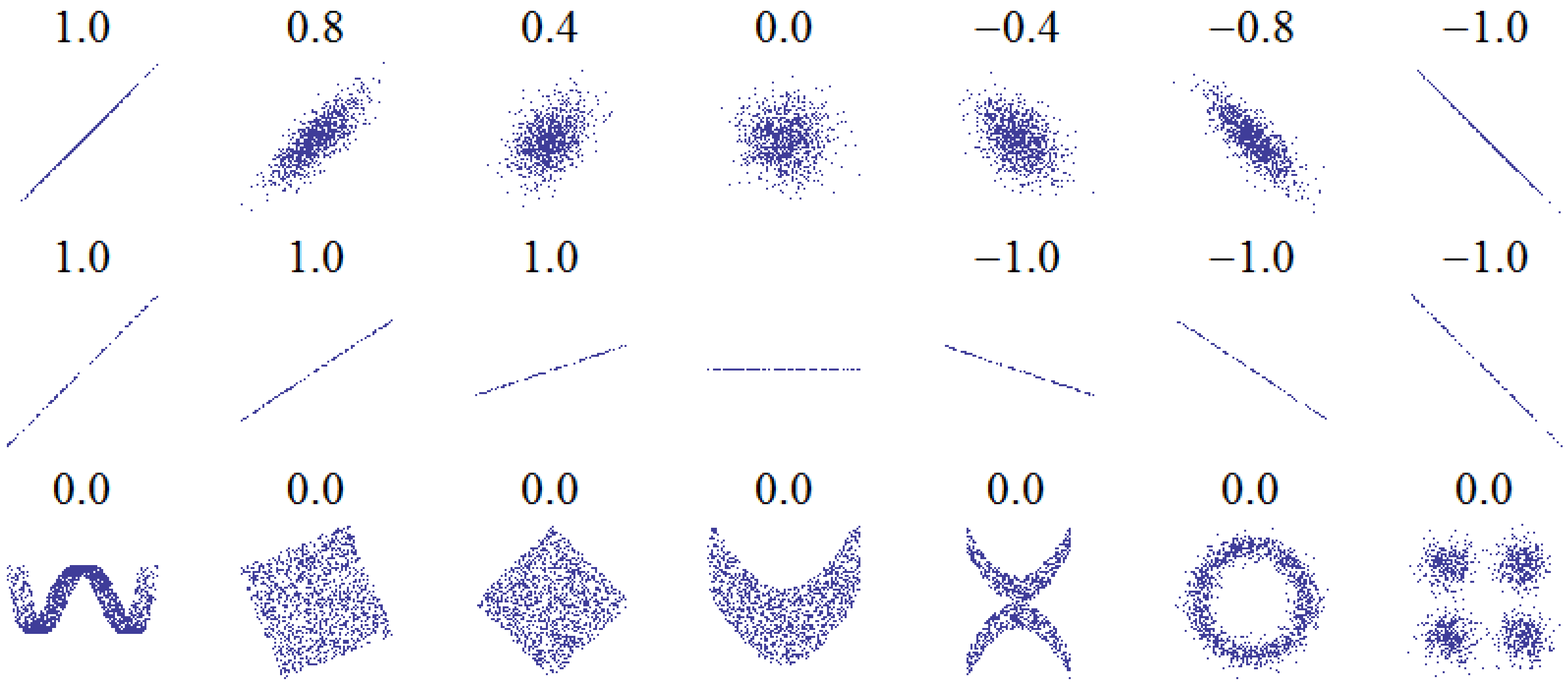
Caution:

No information about slope except whether positive or negative!



http://de.wikipedia.org/wiki/Korrelationskoeffizient

Excursion: Pearson Correlation Coefficient



http://de.wikipedia.org/wiki/Korrelationskoeffizient

Prerequisites: Covariance Matrix

- Multidimensional feature vectors can be represented by multivariate random variables!

$$\mathrm{X} := \begin{pmatrix} X_1 \\ X_2 \\ \cdots \\ X_N \end{pmatrix}$$

Prerequisites: Covariance Matrix

- For multivariate random variables the covariance can be determined for each combination of random variables within the vector
- The covariance matrix is definded as:

$$\mathrm{Cov}(\mathrm{X}) = \mathbb{E}[(X - \mathbb{E}[X])(X - \mathbb{E}[X])^T]$$

Prerequisites: Covariance Matrix

- How to get covariance matrix from training data set?
- Need covariance matrix for each class!
- Assuming equal distribution: Expected value = mean!

$$\mathrm{Cov}(\mathrm{X}) = \mathbb{E}[(X - \mathbb{E}[X])(X - \mathbb{E}[X])^T]$$

$$= \frac{1}{N_k} \sum_{i=1}^{N_k} (x_i - \mu_k)(x_i - \mu_k)^T$$

Number of samples belonging to class k

Sample feature vector from class k

Mean feature vector over all samples from class k

Prerequisites: Cholesky Matrix

- A covariance matrix Σ can be uniquely decomposed into:

$$\boldsymbol{\Sigma} = \boldsymbol{U}^{\boldsymbol{T}}\boldsymbol{U}$$

- U is an upper triangular matrix
- This matrix is called Cholesky matrix

Inverse Cholesky Transform

- Claim:
  Multiplying each sample point with the inverse transposed Cholesky matrix $(U^T)^{-1}$ of the covariance matrix uncorrelates the data set!

- This means the covariance matrix of the transformed data set should be the identity!

$$\mathrm{Cov(X)} = \mathbb{E}[(X - \mathbb{E}[X])(X - \mathbb{E}[X])^T]$$

$$= \frac{1}{N_k} \sum_{i=1}^{N_k} (x_i - \mu_k)(x_i - \mu_k)^T$$

Inverse Cholesky Transform

$$\frac{1}{N_k}\sum_{i=1}^{N_k}({U^T}^{-1}x_i - {U^T}^{-1}\mu_k)({U^T}^{-1}x_i - {U^T}^{-1}\mu_k)^T$$

$$=\frac{1}{N_k}\sum_{i=1}^{N_k}{U^T}^{-1}(x_i-\mu_k)(x_i-\mu_k)^T{{U^T}^{-1}}^T$$

$$=\frac{1}{N_k}\sum_{i=1}^{N_k}{U^T}^{-1}(x_i-\mu_k)(x_i-\mu_k)^T U^{-1}$$

$$={U^T}^{-1}[\frac{1}{N_k}\sum_{i=1}^{N_k}(x_i-\mu_k)(x_i-\mu_k)^T]U^{-1}$$

$$={U^T}^{-1}\Sigma U^{-1} \quad ={U^T}^{-1}(U^T U)U^{-1} \quad =({U^T}^{-1}U^T)(UU^{-1})$$

Inverse Cholesky Transform

$$\frac{1}{N_k}\sum_{i=1}^{N_k}({U^T}^{-1}x_i - {U^T}^{-1}\mu_k)({U^T}^{-1}x_i - {U^T}^{-1}\mu_k)^T$$

$$=\frac{1}{N_k}\sum_{i=1}^{N_k}{U^T}^{-1}(x_i-\mu_k)(x_i-\mu_k)^T{{U^T}^{-1}}^T$$

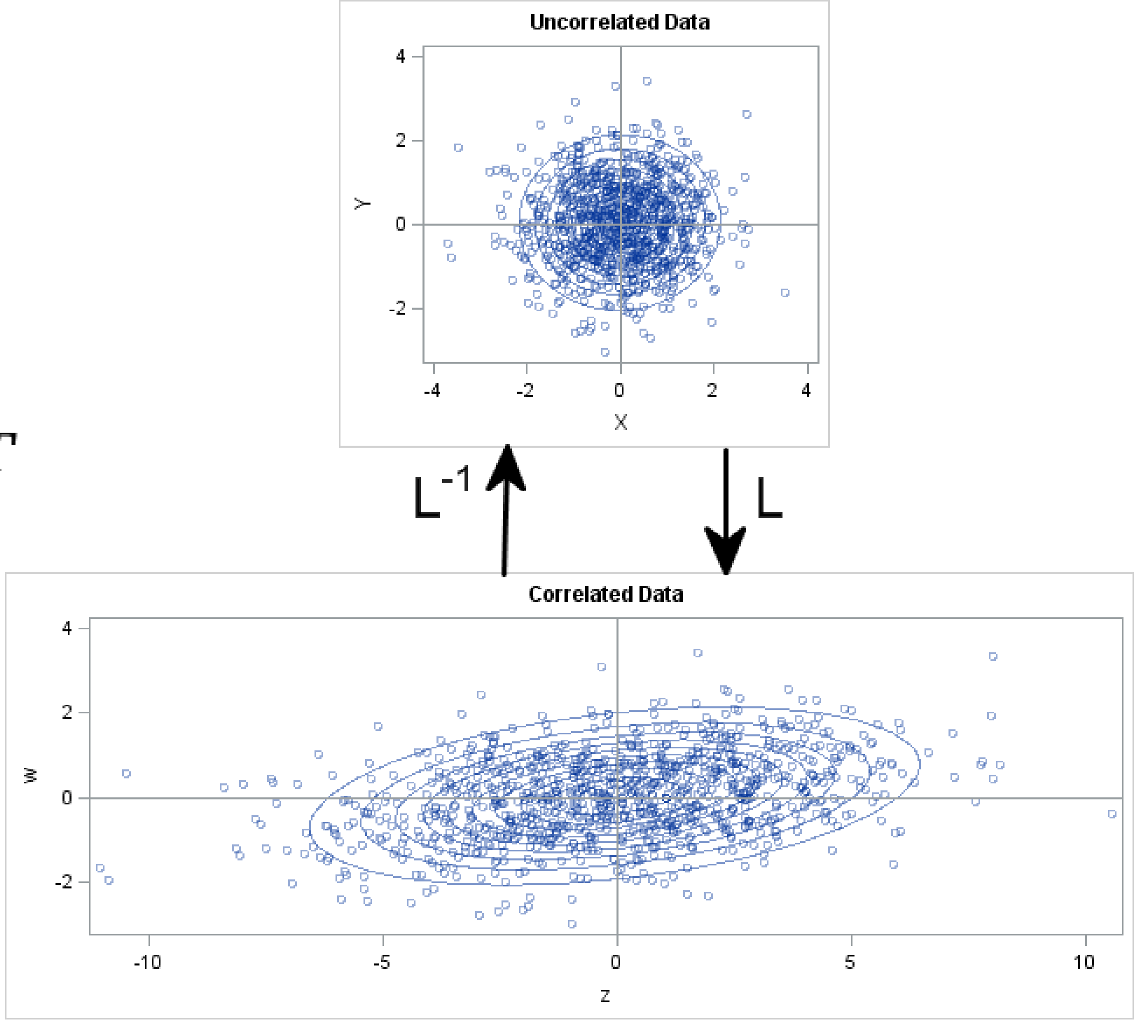
$$=\frac{1}{N_k}\sum_{i=1}^{N_k}{U^T}^{-1}(x_i-\mu_k)(x_i-\mu_k)^T U^{-1}$$

$$={U^T}^{-1}[\frac{1}{N_k}\sum_{i=1}^{N_k}(x_i-\mu_k)(x_i-\mu_k)^T]U^{-1}$$

$$={U^T}^{-1}\Sigma U^{-1} \quad ={U^T}^{-1}(U^T U)U^{-1} \quad =({U^T}^{-1}U^T)(UU^{-1}) =I$$

Inverse Cholesky Transform

$$L := U^T$$



Mahalanobis Distance

- Distance measure, that takes into account the correlations of the data set
- Main idea:
  - Transform data into uncorrelated space with inverse cholesky transform
  - Measure euclidean distance to mean in transformed space

Mahalanobis Distance

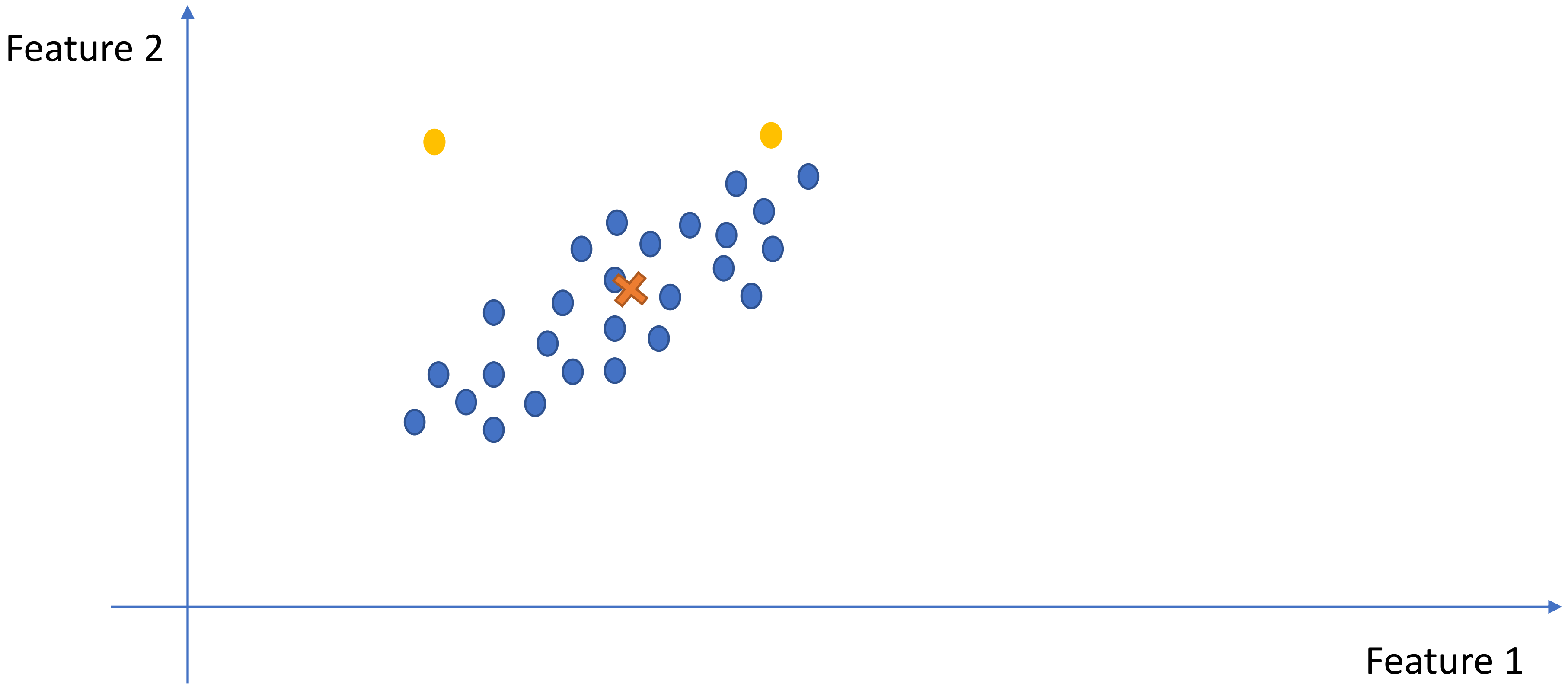
$$
\begin{aligned}
d(x,\mu) &:= \sqrt{(U^{T^{-1}}x - U^{T^{-1}}\mu)^T(U^{T^{-1}}x - U^{T^{-1}}\mu)} \\
&= \sqrt{(x-\mu)^T U^{T^{-1^T}} U^{T^{-1}} (x-\mu)} \\
&= \sqrt{(x-\mu)^T U^{-1} U^{T^{-1}} (x-\mu)} \\
&= \sqrt{(x-\mu)^T (U^T U)^{-1} (x-\mu)} \\
&= \sqrt{(x-\mu)^T \Sigma^{-1} (x-\mu)}
\end{aligned}
$$

Mahalanobis Distance

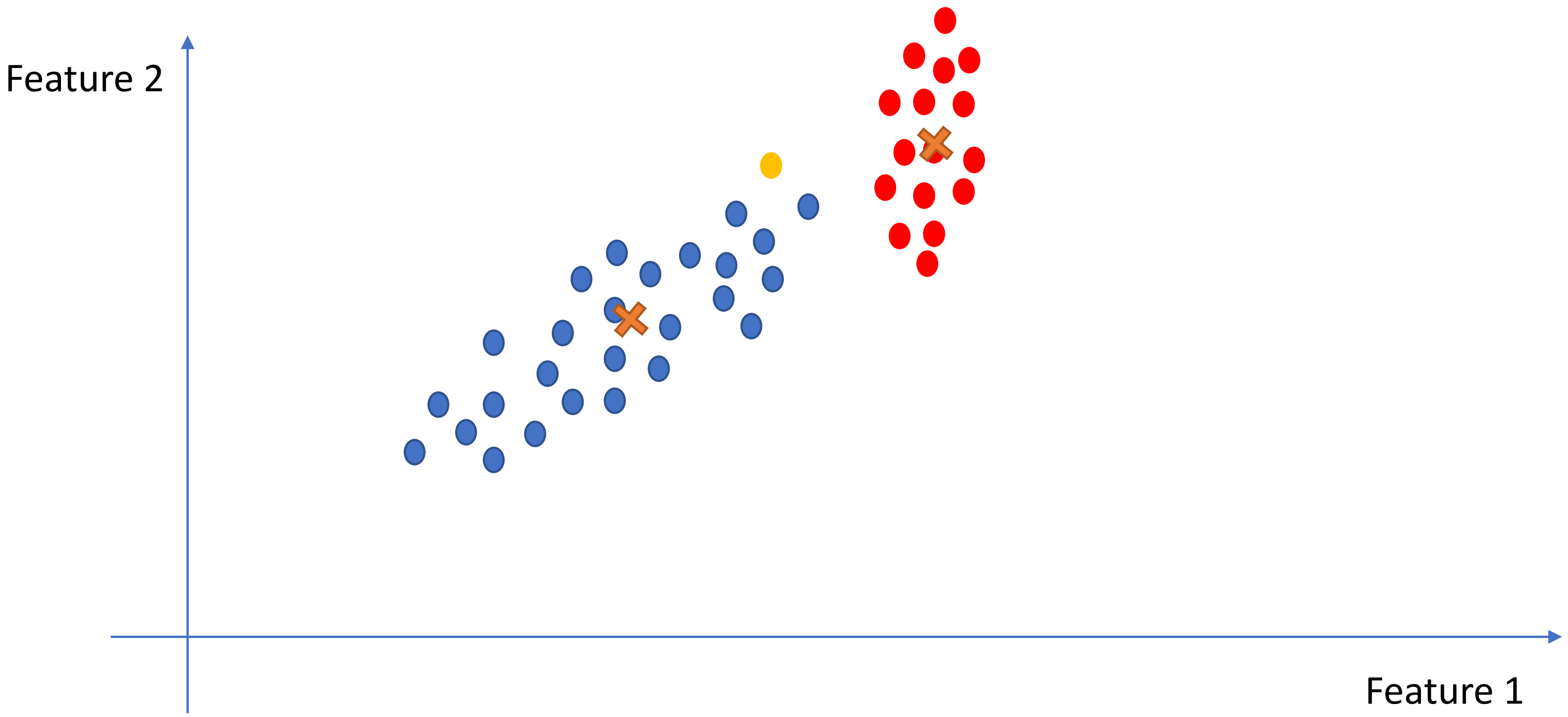
$$d(x, \mu) := \sqrt{(x-\mu)^T \Sigma^{-1} (x-\mu)}$$

- To calculate the distance:
  - No need for cholesky matrix
  - Only need inverse of covariance!

Likelihood Estimation



Classification via Likelihood Estimation



Mahalanobis Classifier

- „Training“:
  - Calculate mean feature vectors for each class
  - Calculate inverse covariance matrix for each class
- Inference:
  - For each class calculate (squared) Mahalanobis distance to its mean feature vector
  - Return class with lowest Mahalanobis distance

$$d(x, \mu) := \sqrt{(x - \mu)^T \Sigma^{-1} (x - \mu)}$$