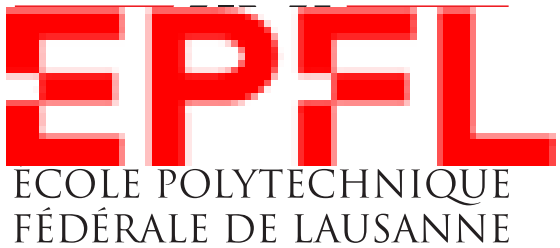
Course overview + exam instructions



Overview

Review the presented ML methods

Exam preparation

ML methods in this course

Structure Discovery

- PCA, ICA
- Clustering

Classification

- Bayes Rule and GMM
- k-Nearest Neighbor
- Support Vector Machine
- Neural Networks

Regression

- Ordinary Least Square Regression
- Weighted Least Square Regression
- Locally Weighted Regression
- Support Vector Regression
- Gaussian Mixture Regression

Model Evaluation

- Unsupervised
- Semi-supervised
- Supervised

Structure Discovery

Find structure by projecting or grouping the data from the original space into another space of lower dimension

Data Projection $\rightarrow$ PCA -> Determine correlations
Data Projection $\rightarrow$ ICA $\rightarrow$ Determine latent stati. indep. sources

Projected space highlights particular features common to subsets of datapoints

PCA / ICA are usually a preprocessing step

Group Datapoints $\rightarrow$ Clustering $\rightarrow$ Dimensionality reduction

PCA – Definition

Two equivalent definitions:

- the orthogonal projection of the data onto a lower dimensional linear space such that the variance of the projected data is maximized
- the linear projection that minimizes the reconstruction error, defined as the mean squared distance between the data points and their projections

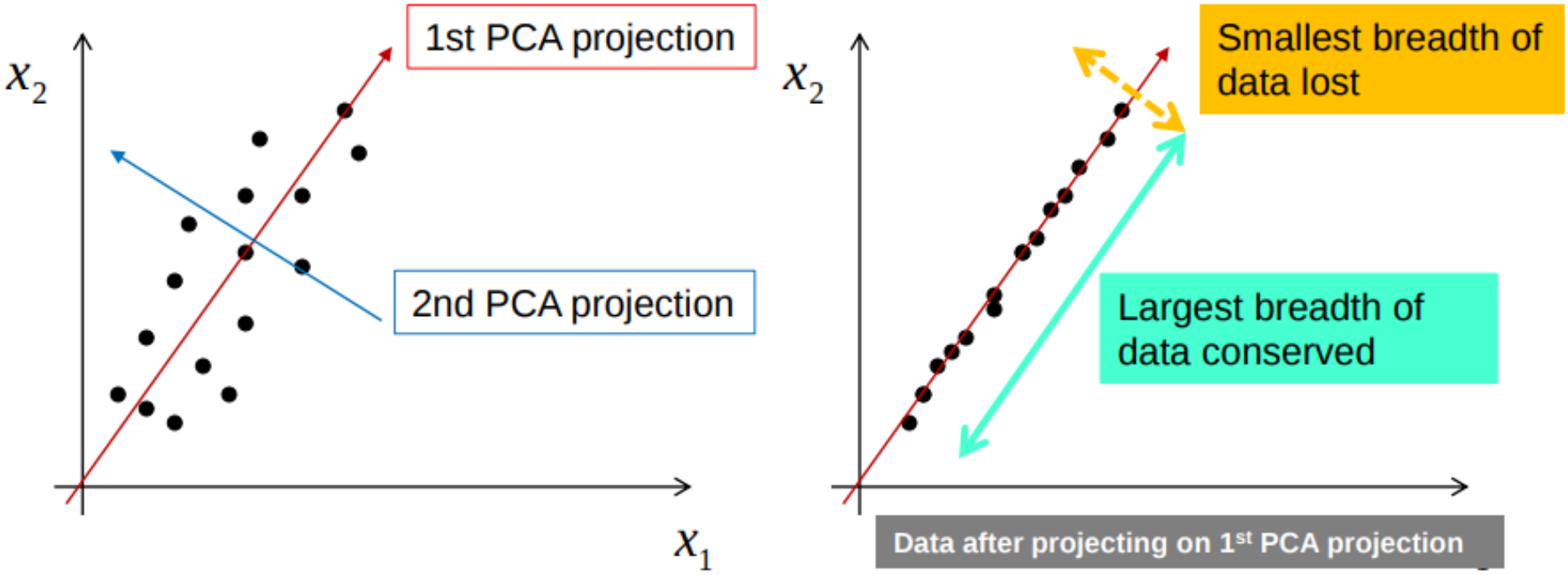
Given a set $X \in \Re^{m \times n}$ of m datapoints with n dimensions. PCA finds a mapping $A \in \Re^{n \times n}$ such as: $Y = AX$

$$\min_A \left|\left|A^{-1}y - x\right|\right| \tag{1}$$

Solutions are eigenvectors of the centered data covariance matrix C :

$$Ce = \lambda e$$

PCA – Eigenvectors



1st eigenvector points towards the direction with largest variance.

PCA – Eigenvalues

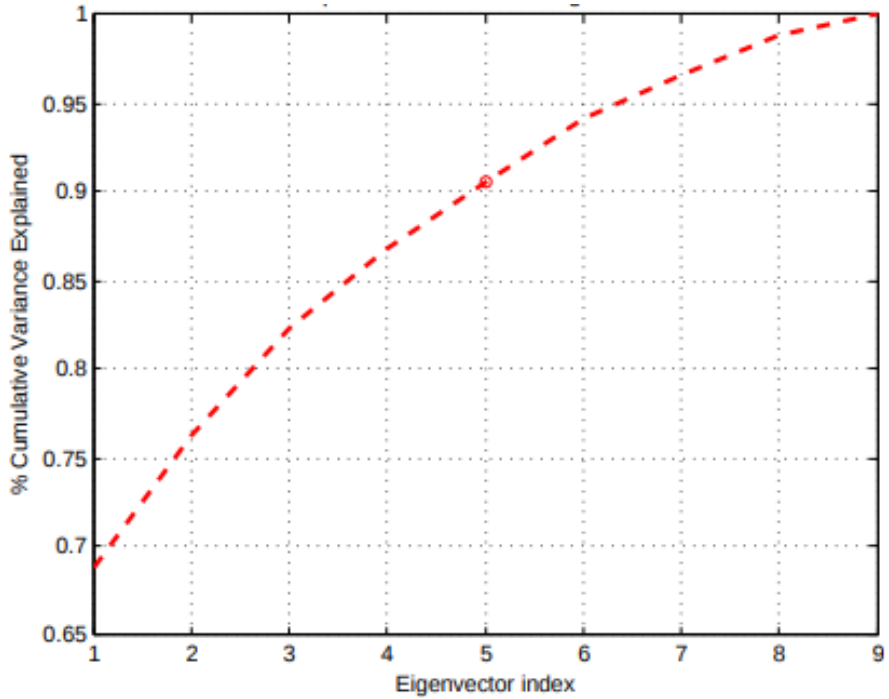
Each eigenvector has an associated eigenvalue
The eigenvalues provide a relative measurement of the variance along each eigenvector

$$\% \text{ of explaned variance along } e_i = \frac{\lambda_i}{\sum_i^n \lambda_i} \tag{2}$$

Dimensionallity reduction is performed based on the cumulative percentage of explained variance.

PCA – Dimensionality reduction

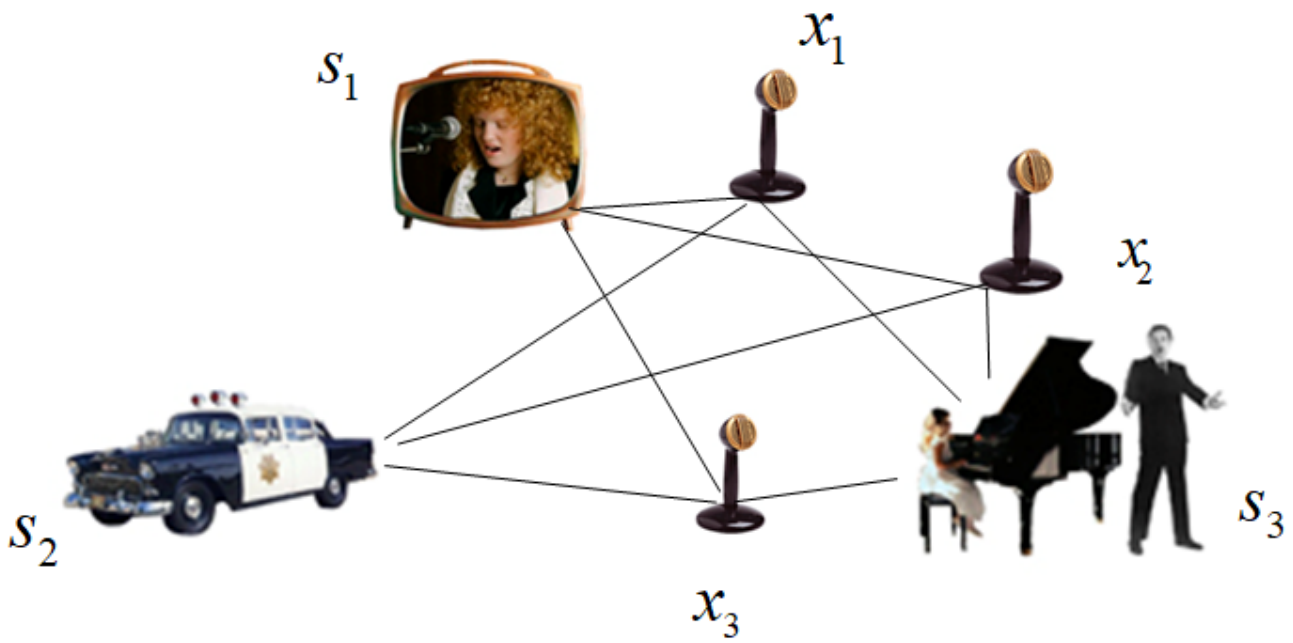
Find the first q eigenvectors that explain a desired percentage of the variance



$$A \in \Re^{n \times n} \xrightarrow[\text{dimensionality}]{\text{reduced}} A \in \Re^{q \times q}$$

$$Y = AX \ , \ Y \in \Re^{m \times q} \text{ where } q \ll n$$

ICA – Dimensionality reduction



N-dimensional observation vector $x \in \mathbb{R}^N$, $N = 3$.

x was generated by a linear combination of N sources, $s \in \mathbb{R}^N$.

$x = As$, mixing matrix $A : N \times N$

ICA uncovers both A and s.

ICA – Dimensionality reduction

The distribution of s is non-Gaussian.

Find sources how joint distribution optimizes
a measure of non-gaussianity
Negentropy or Kurtosis

ICA – Sources identification

ICA Estimation proceeds through two preprocessing steps, following by an iterative procedure:

Preprocessing steps:

- Centering (mean=0)
- Whitening (variance = 1)

Iterative procedure:

- Estimate each component iteratively by gradient descent on non-gaussianity measure.
- Proceed to a decorrelation to ensure that each component is distinct.

Clustering

Unsupervised Learning - unsupervised classification:
We know neither the number of classes, nor
which class the datapoints belong to.

- Goal: Assign similar samples to the same groups (clusters)
- Methods: k-Means, Soft k-Means, DBSCAN, GMM
- Evaluation RSS, BIC, AIC and F1-Measure

What makes a good cluster?
Maximize inner-cluster similarity while minimize inter-cluster similarity

Clustering – Methods

	k-Means	Soft k-means	DBSCAN
hyper-parameters	k Lp-norm	k, β Lp-norm	ϵ,min samples in cluster Lp-norm
Training	EM	EM	Samples' Density based
Clusters	Globular	Globular	Arbitrary Shaped
Samples' Assignment	Hard	Soft	Hard
Comp. Complexity	O(K*M)	O(K*M)	O(M*log(M))
Noise Detection	No	No	Yes

Clustering – k-Means

$$\underline{\text{Goal:}}\ \text{Minimize}\ J = \sum_{k=1}^{K} \sum_{x_i \in c_k} ||x_i - \mu_k||_2$$

Find center of each cluster μ_k

Unknown samples that belong to each cluster ($x_i \in c_k$)

Solution: Expectation Maximization algorithm

Expectation: Assign each sample to a cluster ($x_i \in c_k$)

Maximization: Update cluster means μ_k

Repeat until convergence

Clustering – Expectation Maximization (k-means)

Expectation: Assign each sample to a cluster ($x_i \in c_k$)

K-Means:

$$r_i^k = \begin{cases} 1 \text{ if } k_i = k \\ 0 \text{ otherwise} \end{cases}$$

$$k_i = \underset{k}{\text{argmin}} \left\{ d\left(x_i, \mu_k\right) \right\}$$

soft K-Means:

$$r_i^k = \frac{exp(-\beta \cdot d\left(\mu_k, x_i\right))}{\sum_{k'} exp(-\beta \cdot d\left(\mu_{k'}, x_i\right))}$$

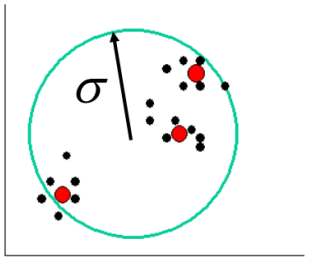
$$r_i^k \in [0,1], \beta \in \Re$$

Maximization: Update cluster means μ_k

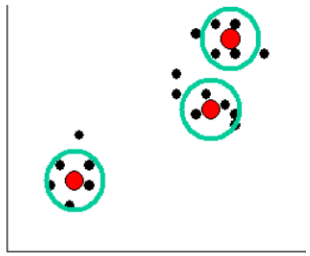
$$\mu_k = \frac{\sum_i r_i^k x_i}{\sum_i r_i}$$

Repeat until $\mu_k^t \approx \mu_k^{t-1}$

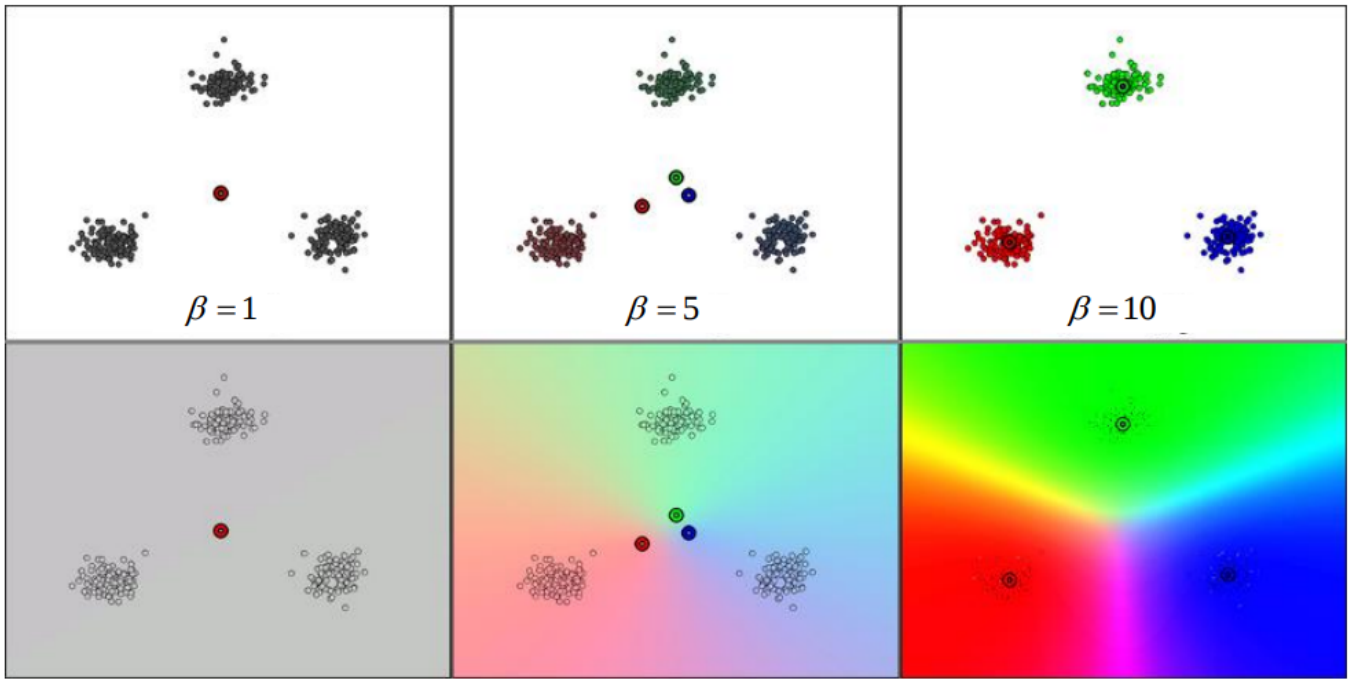
Clustering – Soft K-means, impact of β



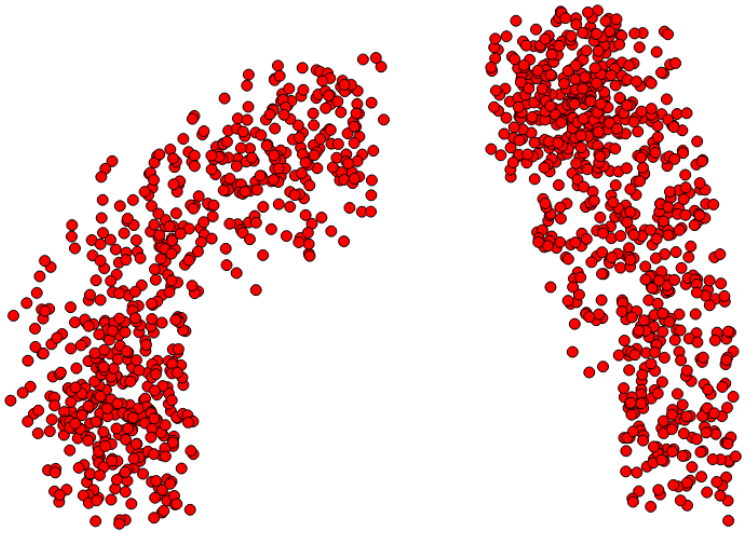
(a) small β



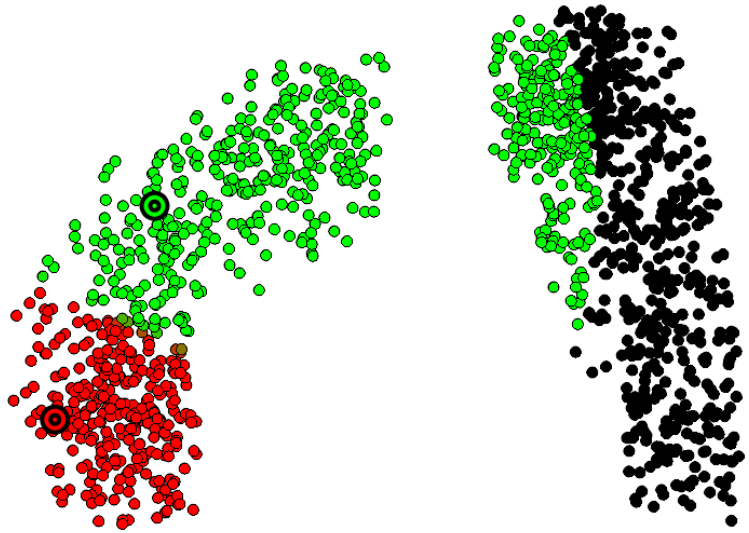
(b) large β



Clustering – Soft K-means, EM



(c) Dataset



(d) Initialization of soft k-means

How will the clusters centers move with iterations of EM algorithm?

Clustering – DBSCAN

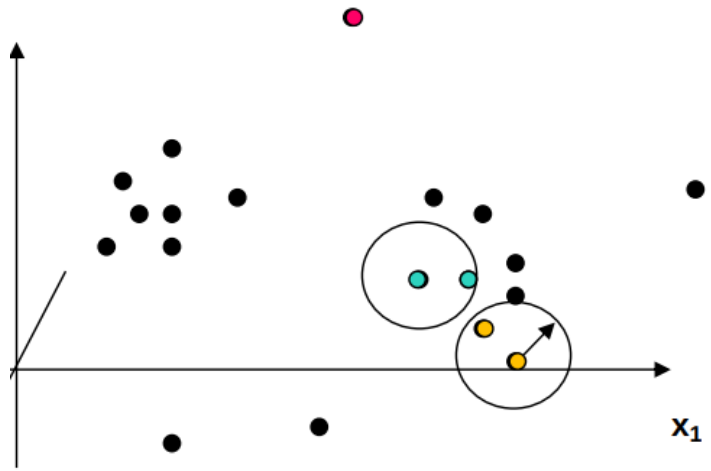
Hyper-parameters ϵ, min samples in cluster ($\min_s$)

Pick un-clustered sample

Find neighbors within ϵ

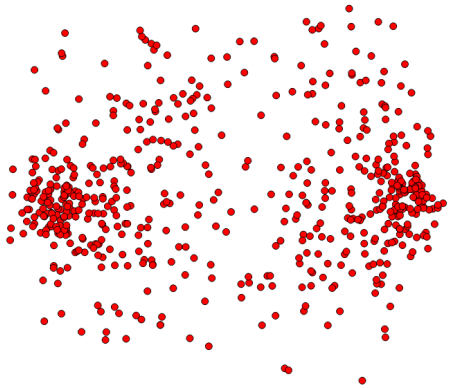
- If num neighbor $<$ $\min_s$ then outlier
- else Assign neighbors in cluster
  - If neighbor cluster exists then merge

Repeat until all assigned to clusters or noise

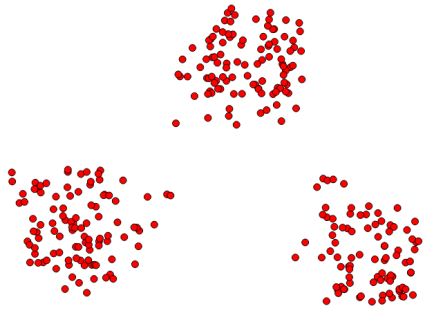


Clustering – Algorithm selection

Which algorithm would you select for each of those datasets?



(e)



(f)

(g)

Clustering – Gaussian Mixture Model

Probabilistic alternative to k-Means

$$p(x) = \sum_{k=1}^{K} \alpha_k \mathcal{N}(x; \mu_k, \Sigma_k)$$

where a_k mixing coefficients $\sum_k = 1$
and μ_k, Σ_k parameters of each component

Maximize the log-likelihood $\ln(\mathcal{L}(\Theta|X))$:

$$\ln(\mathcal{L}(\Theta|X)) = \sum_{m=1}^{M} \ln \left\{ \sum_{k=1}^{K} \alpha_k \mathcal{N}(x; \mu_k, \Sigma_k) \right\}$$

No closed form solution, apply Expectation-Maximization algorithm

Clustering – GMM – EM algorithm

1. k-Means initialization of μ_k,random initialization of Σ_k and α_k
2. **E step:** Evaluate responsibilities

$$r_i^k = \frac{\alpha_k N\left(x_i; \mu_k, \Sigma_k\right)}{\sum_{k'} \alpha_{k'} N\left(x_i; \mu'_k, \Sigma'_k\right)}$$

3. **M step:** Given current estimate, maximize the log-likelihood

$$\mu_k = \frac{1}{M_k} \sum_{i=1}^{M} r_i^k x_i \qquad \alpha_k = \frac{M_k}{M}$$

$$\Sigma_k = \frac{1}{M_k} \sum_{i=1}^{M} r_i^k \left(x_i - \mu_k\right)\left(x_i - \mu_k\right)^T$$

$$\text{where: } M_k = \sum_{i=1}^{M} r_k^i$$

Iterate E-M until convergence

Clustering – GMM – EM algorithm

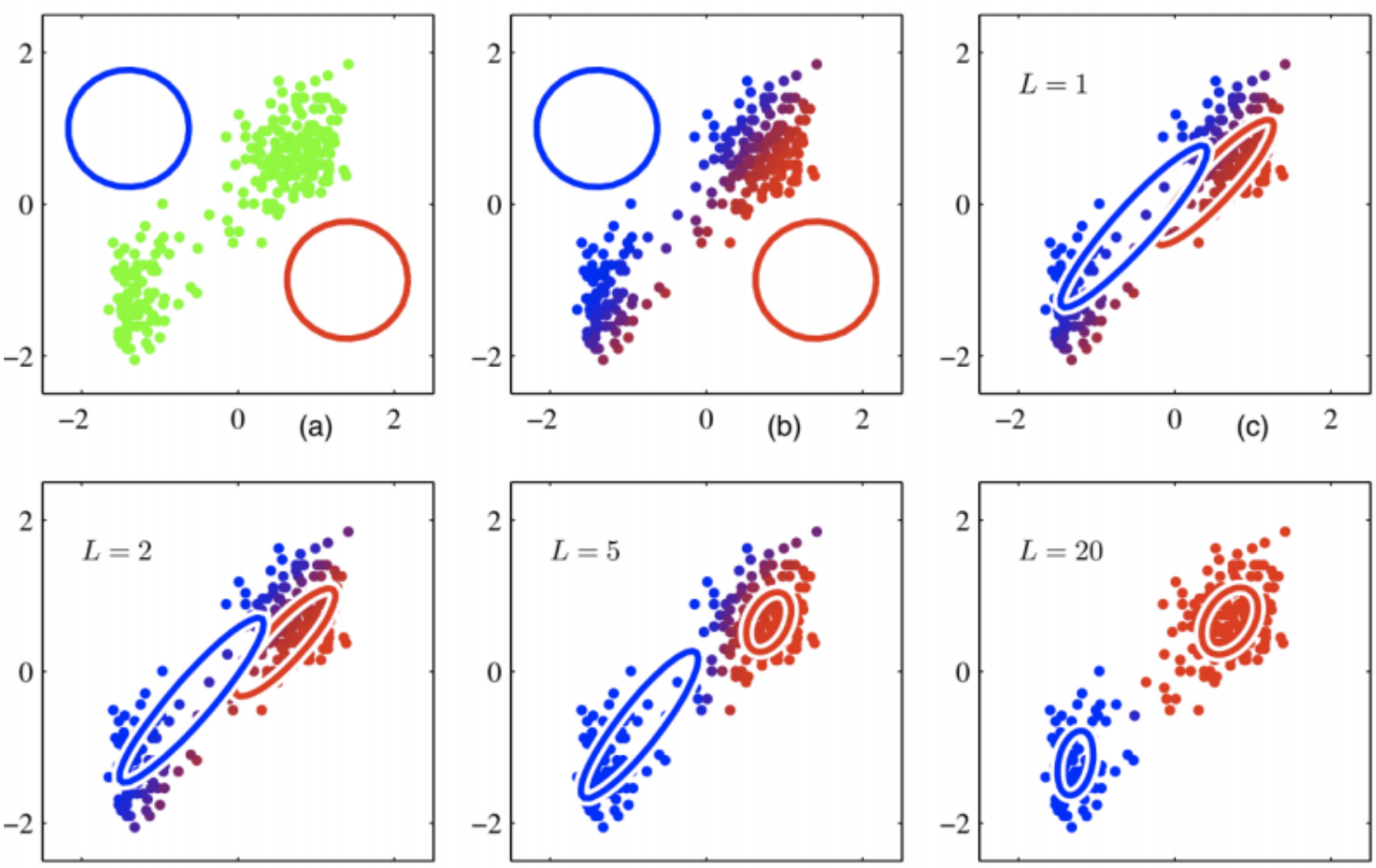
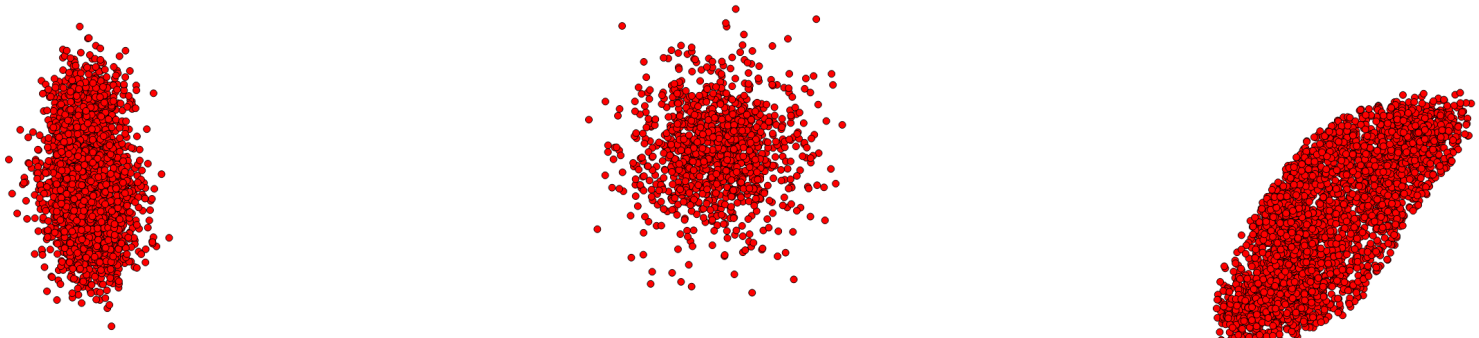


Figure: Step of E-M algorithm for GMM.Pattern Recognition and Machine Learning, C.Bishop p.[437]

Clustering – GMM – type of Covariance

Hyper-parameters: Number of Gaussian components, type of covariance Matrix (Spherical, Diagonal, Full)



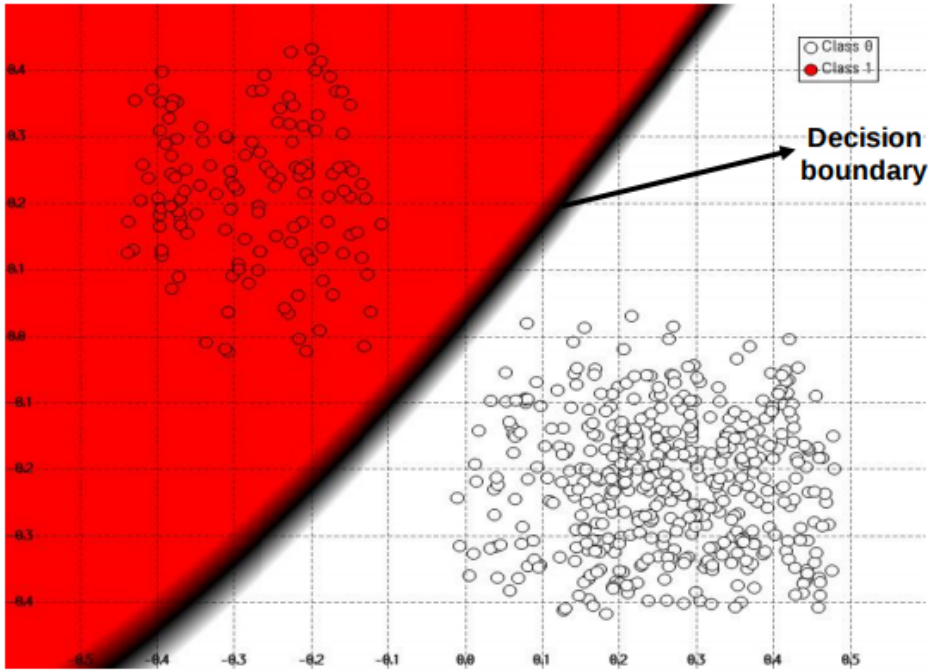
(a) (b) (c)

Which type of covariance would lead to largest likelihood ?

Classification

Supervised learning with nominal target values

Goal: Find decision boundary the separates the classes
Methods: GMM with Bayes rule, k-NN and SVM
Evaluation: Cross-validation
Performance Metrics: Classification error, F-Measure.



Classification: Bayes Rule

Model: $p(C = c|x)$ with Bayes Rule:

$$p(C = c|x) = \frac{p(C = c)p(x|C = c)}{p(x)}$$

$p(C = c) \rightarrow$ probability of class C

$p(x|C = c) \rightarrow$ How samples are distributed within class c

$p(x) \rightarrow$ Distribution of data (independent of class)

$$p(C = c|x) \propto p(C = c)p(x|C = c)$$

Decision Boundary:

$$p(C = 1|x_*) = p(C = 2|x_*)$$

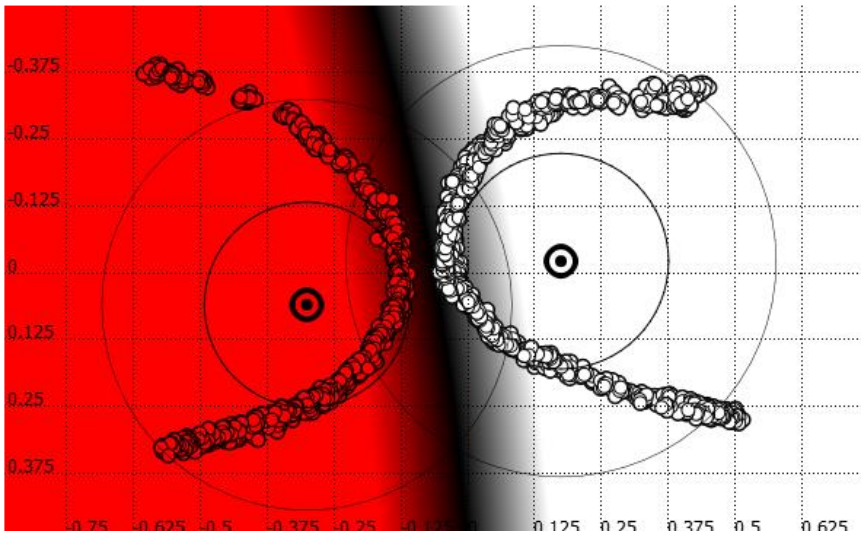
if $p(C = 1) = p(C = 2)$ then $p(x_*|C = 1) = p(x_*|C = 1)$

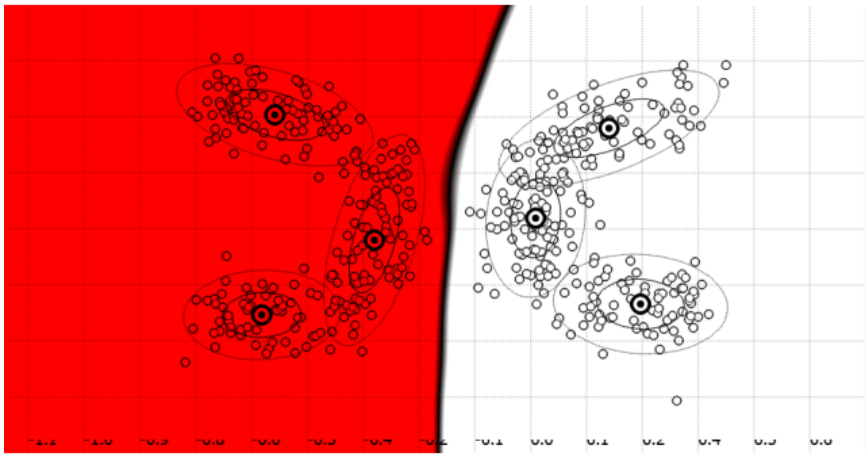
Classification – Bayes Rule and GMM

Model: $p(x|C = c)$ as GMM :

$$p\left(x|C = c\right) = \sum_{k=1}^{K} \alpha_k^c \mathcal{N}\left(x; \mu_k^c, \Sigma_k^c\right)$$

For each class fit a GMM using EM algorithm.





GMM classification with different number of components

Classification – k-NN

$$p(C = c|x_*, K) = \frac{N_{c,K}}{K}$$

$N_{c,K} \rightarrow$ Number of samples of class c included in K nearest neighbors

$K \rightarrow$ Number of nearest neighbors (hyper-parameter)

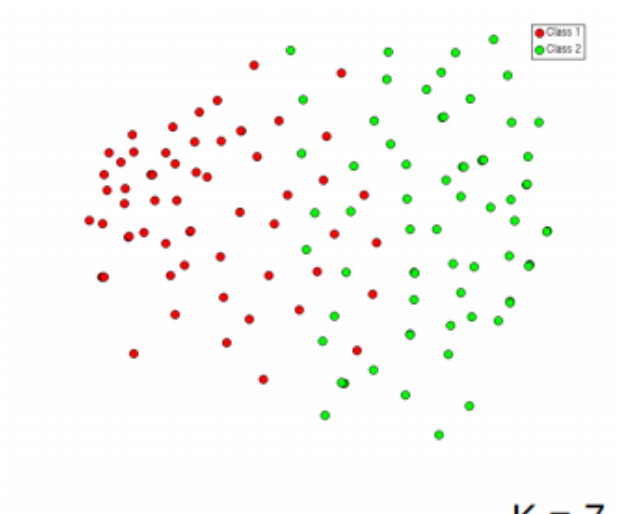
Advantages

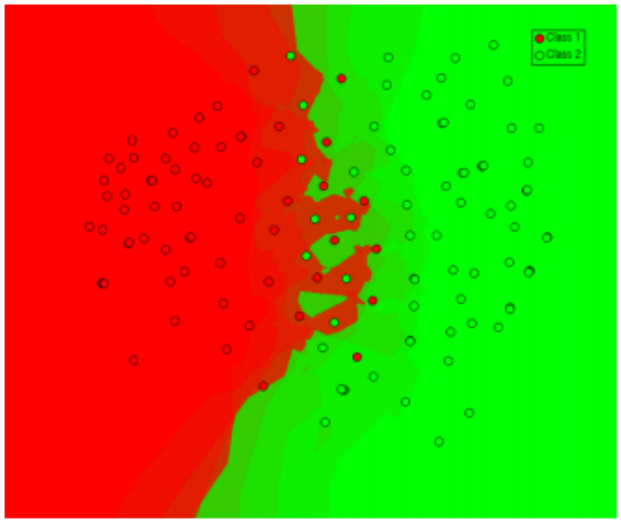
- Simplicity
- No assumptions regarding the distribution of data

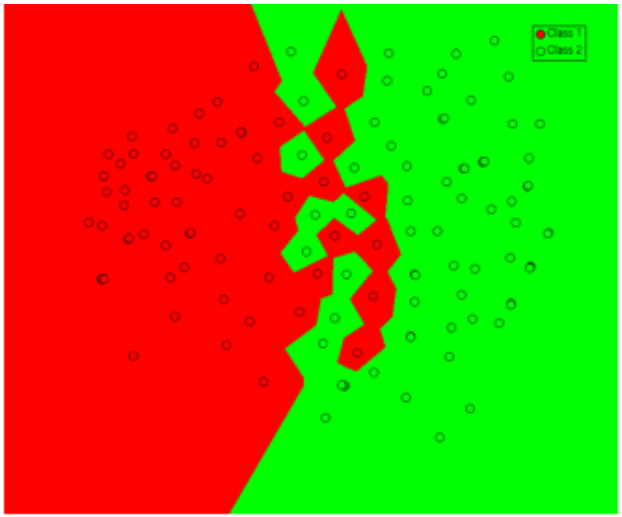
Disadvantages

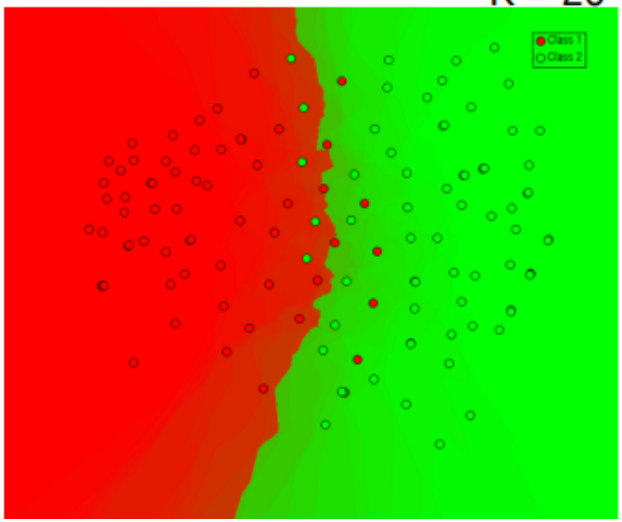
- Curse of dimensionality
- High computational complexity in testing
- Large Impact of K

Classification – k-NN









Classification – SVM

Model:

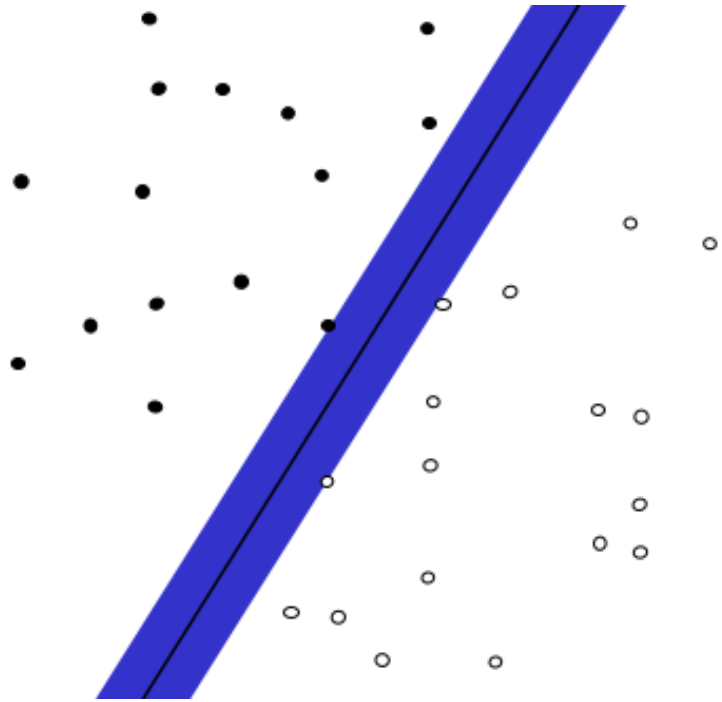
$$y = f(x; w, b) = \text{sgn}\left(w^T x + b\right)$$
$$y_i = \{-1, 1\}$$

Goal: Find w, b that maximize the margin and classify all instances correctly (Hard-margin SVM)

$$\min_{w,b} \frac{1}{2} ||w||^2$$

subject to:

$$\left(y_i \left(w^T x_i + b\right)\right) \geq 1 \forall i$$



Classification – SVM – Dual problem

Use Lagrange duality for computational efficiency
Ease applicability of the kernel trick for nonlinear problem

Primal:

$$\min_{w,b} \frac{1}{2} ||w||^2$$

subject to:

$$\left(y_i \left(w^T x_i + b\right)\right) \geq 1$$

Lagrangian:

$$L(w, b, a) = \frac{1}{2} ||w||^2 - \sum_{i=1}^{M} \alpha_i \left(y_i \left(w^T x_i + b\right) - 1\right)$$

subject to:

$$\alpha_i \geq 0, i = 1..M$$

$$\text{Dual problem} : \left(\max_{a \geq 0} \left(\min_{w,b} L(w, b, a)\right)\right)$$

Classification – SVM – Dual problem

$$\text{Dual problem} : \left(\max_{a \geq 0} \left(\min_{w,b} L(w, b, a) \right) \right)$$

$$\min_{w,b} L(w, b, a) \rightarrow \frac{\partial L}{\partial w} = 0 \text{ and } \frac{\partial L}{\partial b} = 0$$

$$\frac{\partial L}{\partial w} = 0 \rightarrow w = \sum_{i=1}^{M} \alpha_i y_i x_i$$

$$\frac{\partial L}{\partial b} = 0 \rightarrow \sum_{i=1}^{M} \alpha_i y_i = 0$$

Replacing w at the Lagrangian we get the dual problem

Classification – SVM – Dual problem

Use Lagrange duality for computational efficiency
Ease applicability of the kernel trick for nonlinear problem

Primal:

$$\min_{w,b} \frac{1}{2} ||w||^2$$

subject to:

$$\left(y_i \left(w^T x_i + b\right)\right) \geq 1$$

Dual:

$$\max_{a \geq 0}(W) = \sum_{i=1}^{M} \alpha_i - \sum_{i=1}^{M} \sum_{j=1}^{M} \alpha_i \alpha_j y_i y_j (x_i^T x_j)$$

subject to:

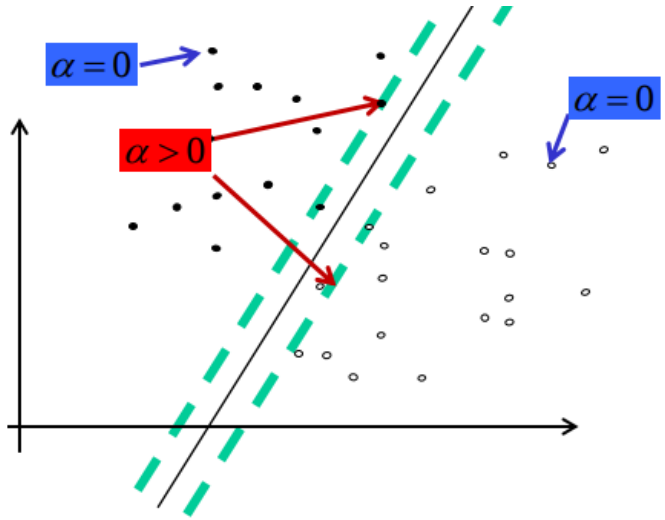
$$\alpha_i \geq 0 \text{ and } \sum_{i=1}^{n} \alpha_i y_i = 0, i = 1..M$$

Find optimal α with Sequential Minimal Optimization (SMO) algorithm

Classification – SVM – KKT conditions

Given optimal solutions α_* the KKT conditions apply

$\frac{\partial L}{\partial w} = 0$
$\frac{\partial L}{\partial b} = 0$
$\alpha_i \left(y_i \left(w^T x_i + b\right) - 1\right) = 0 \rightarrow$ Defines when a vector is a support vector
$y_i \left(w^T x_i + b\right) - 1 = 0 \rightarrow$ Requires all the samples to be correctly classified
$\alpha_i \geq 0$



Classification – Soft margin SVM

No solution of hard-margin SVM for overlapping classes

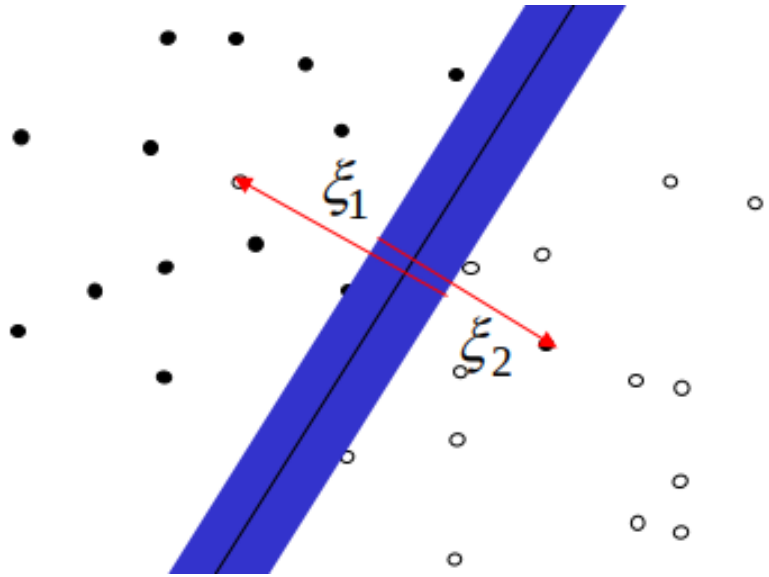
Introduce some slack on the constraints This allows SVMs to misclassify some samples or reduce the margin of error (data points are allowed to be on the wrong side of or inside the margin)

Primal:

$$\min_{w,b,\xi} \frac{1}{2}||w||^2 + \frac{C}{M}\sum_{i=1}^{M}\xi_i$$

subject to:

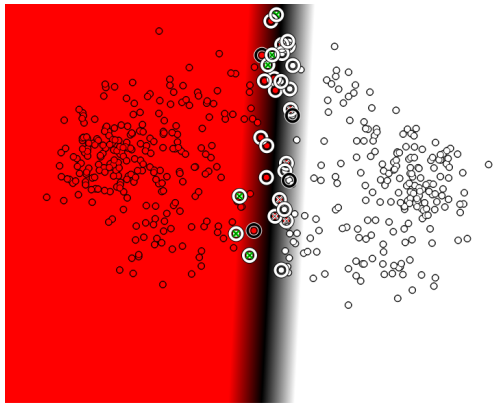
$$\left(y_i\left(w^T x_i + b\right)\right) \geq 1 - \xi_i$$



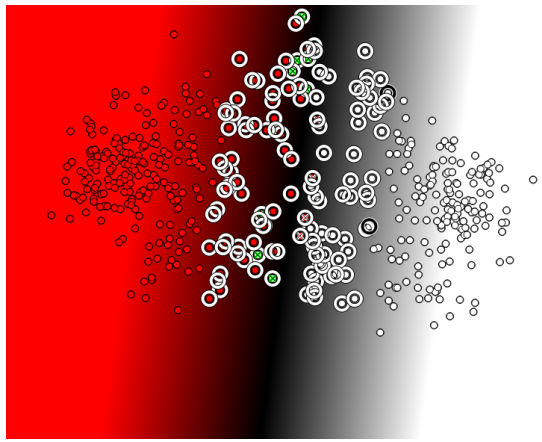
C is the cost of miss-classification. Defines a trade-off between the maximization of margin and minimization of slack

Classification – Soft margin SVM

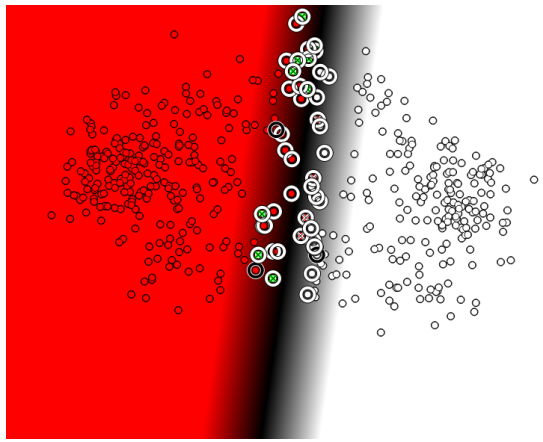
Sort the following models in decreasing C



(f)



(g)



(h)

Classification – Kernel SVM

Linear SVM cannot deal with non-linearly separable classes

$$y = f(x; w, b) = \text{sgn}\left(\sum_{i=1}^{M} \alpha_i y_i x_i^T x + b\right)$$

Linearly inseparable problem become linearly separable in higher dimension space.

Apply kernel trick.

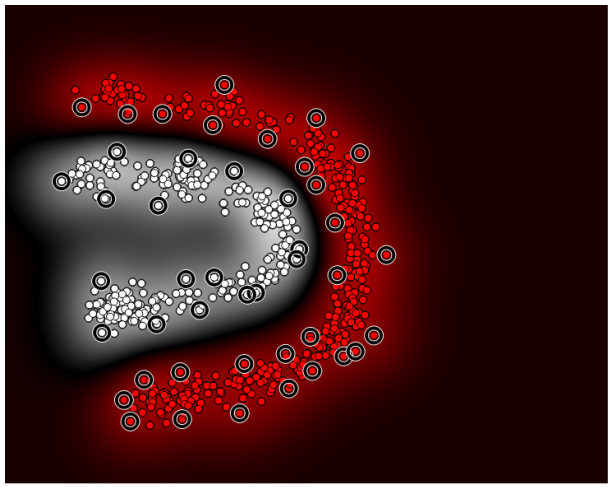
Decision function :

$$y = f(x_*; w, b) = \text{sgn}\left(\sum_{i=1}^{M} \alpha_i y_i k(x_i, x_*) + b\right)$$

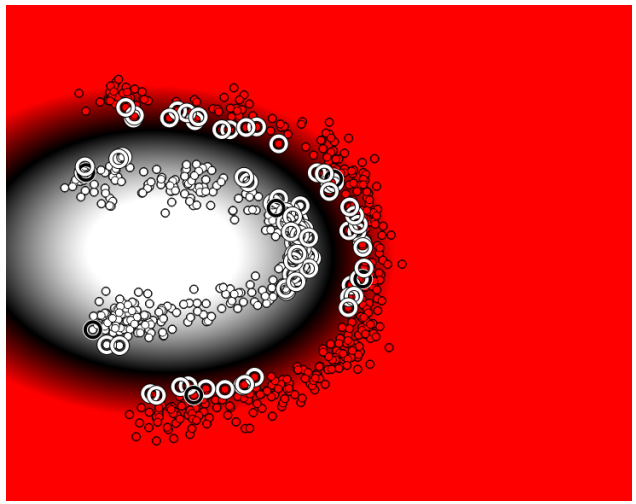
RBF (Gaussian) kernel: $k(xi, xj) = \exp\left(-\frac{||x_i - x_j||^2}{2\sigma^2}\right)$

Classification – Kernel SVM

Which model has largest kernel width?



(i)



(j)

Regression

Supervised learning with continuous target values y

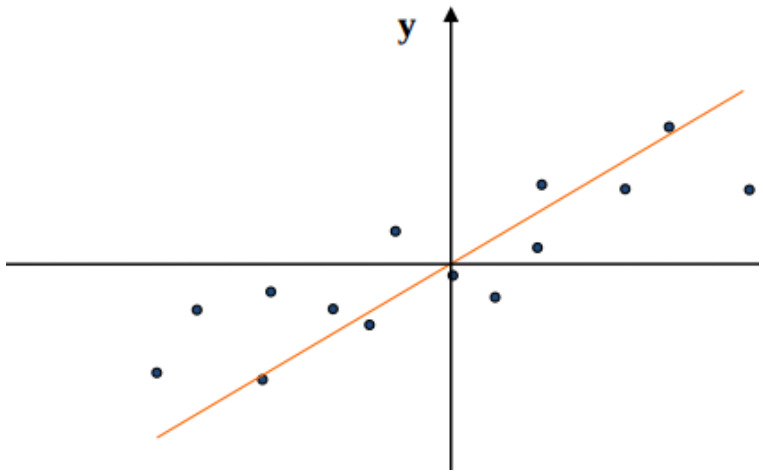
Goal: Find $f(\cdot)$ such as: $y = f(x) + \epsilon$
Methods: Linear, Weighted, Locally weighted, SVR and GMR
Evaluation: Cross-validation
Performance Metrics:Mean Squared error

Regression – Ordinary Least Squares

Model: $\hat{y} = w^T x + b$

Goal: Minimize Least Square Error $\rightarrow J = \sum_{i=1}^{M} \frac{1}{2} \left(w^T x_i - y_i \right)^2$

Solution: $\hat{w} = \left(X X^T \right)^{-1} X y$



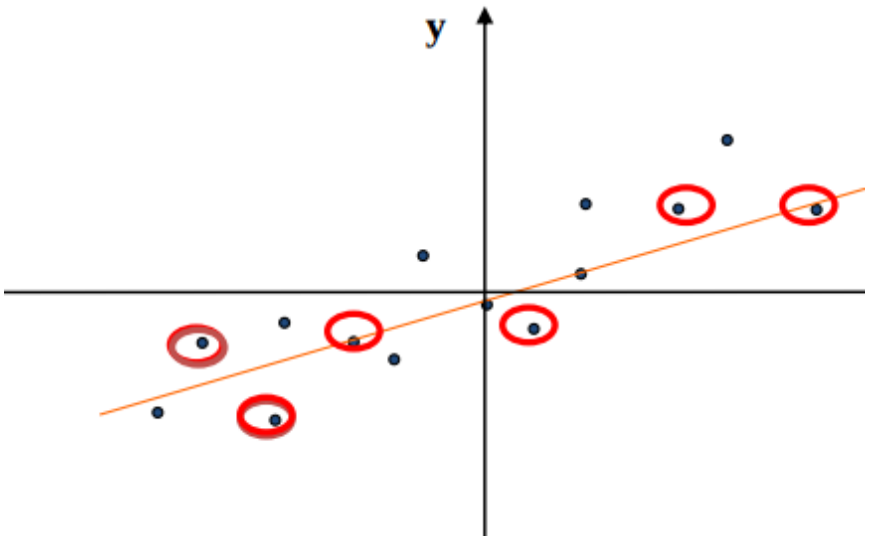
Regression – Weighted Least Squares

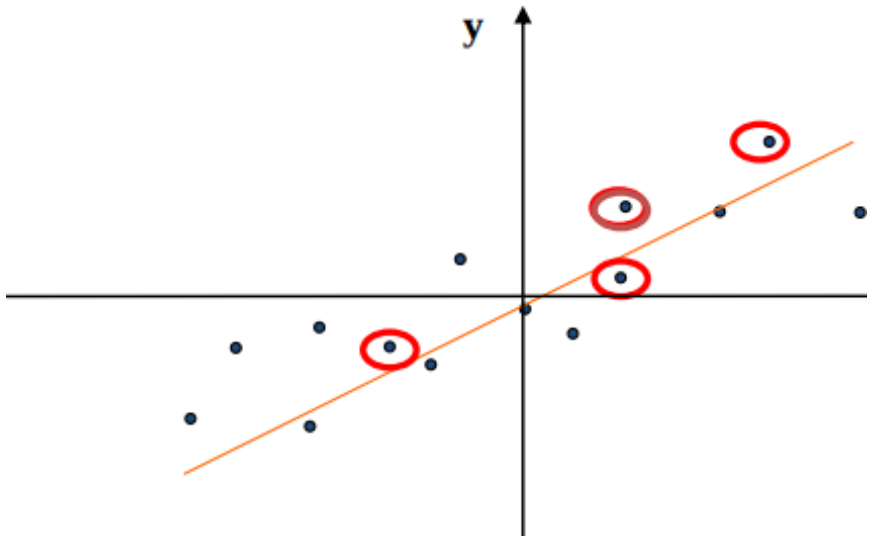
Introduce weight β_i to the error of samples

Model: $\hat{y} = w^T x + b$

Goal: Minimize Least Square Error $\rightarrow J = \sum_{i=1}^{M} \frac{1}{2} \beta_i \left(w^T x_i - y_i \right)^2$

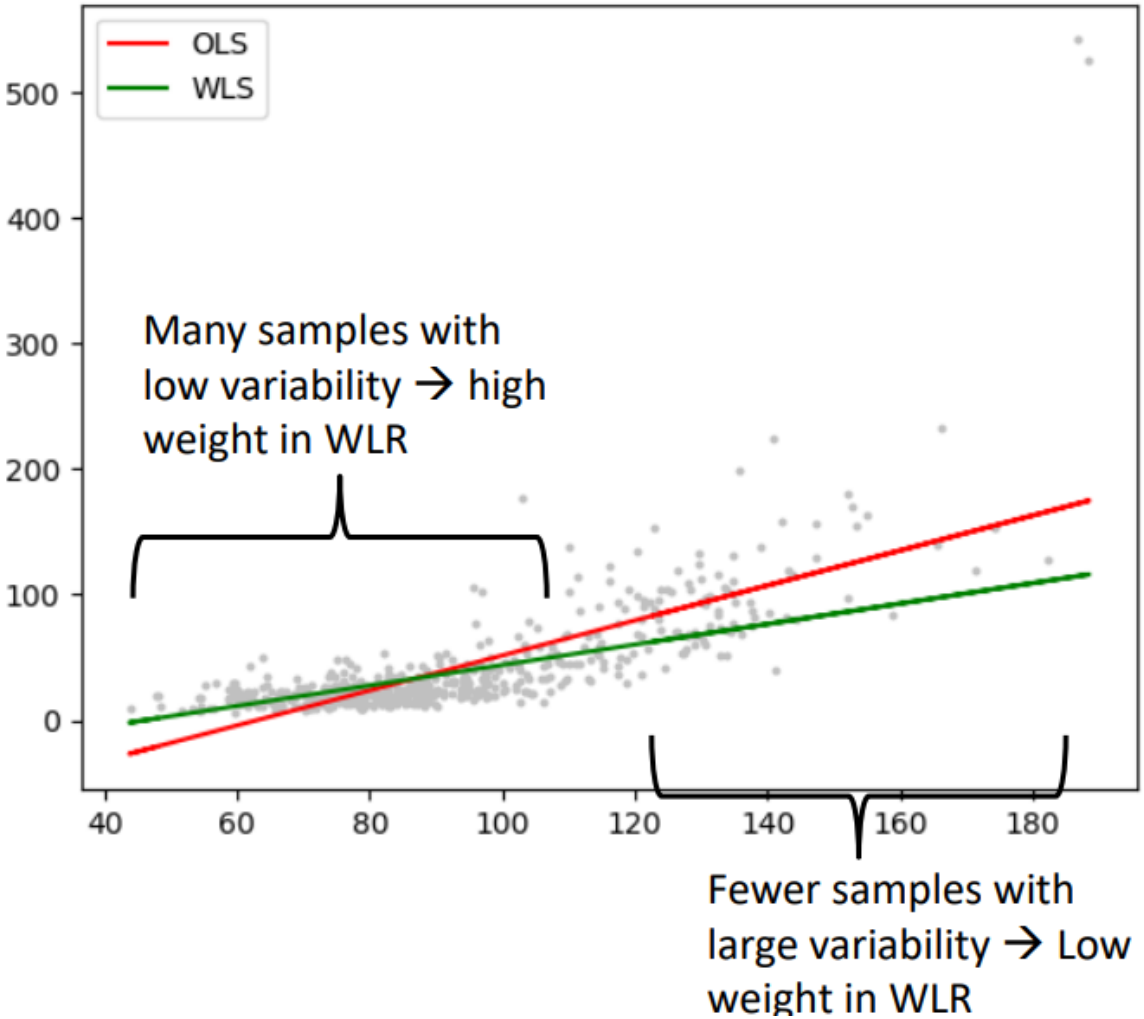
Solution: $\hat{w} = \left(Z Z^T \right)^{-1} Z v$, where: $Z = X B^{1/2}$ and $v = B^{1/2} y$





Prediction error of samples in circles have higher weight

Regression: OLS vs WLS



Regression – Locally Weighted Regression (LWR)

Estimates linear dependencies locally

LWR performs regression analysis by fitting models locally

Local models allow to estimate more complex regression functions.

It is memory based i.e. Requires the training data for predictions

Define weighting function $\beta_i = \exp\left(-\frac{||x_i - x_*||}{2\sigma^2}\right)$

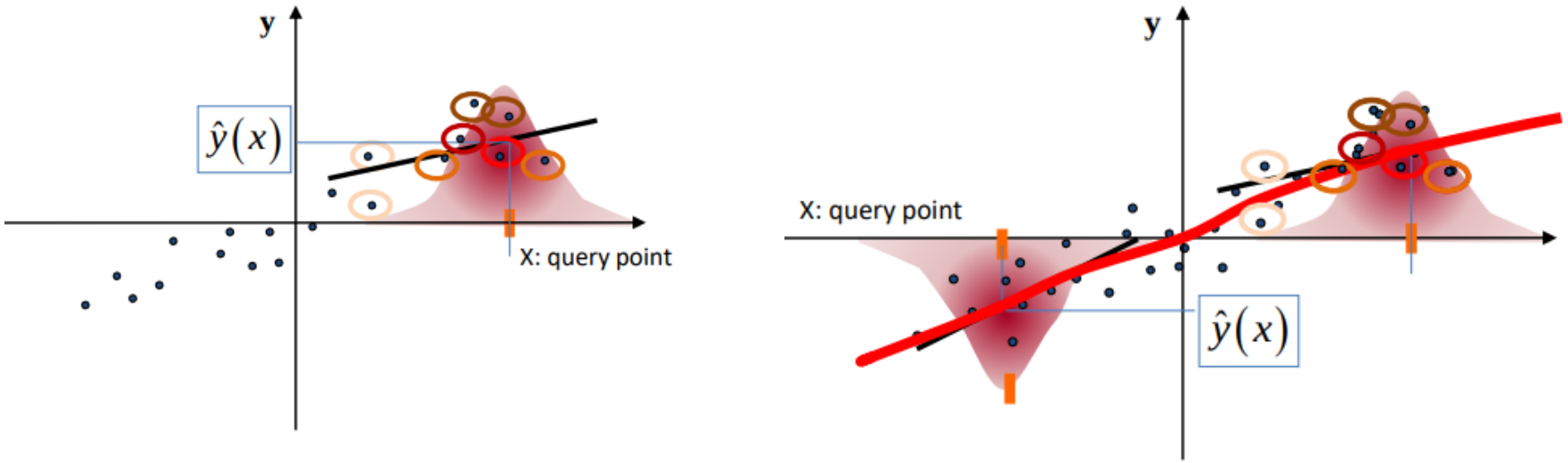
$$\hat{w}_{x_*} = \left(XBX^T\right) XBy$$

B is a diagonal matrix with elements b_i

$$\hat{y} = w_{x_*} x_*$$

Regression – Locally Weighted Regression (LWR)

The RBF defines the weight of each x_i on a regression model specific for x_*



A local weighted model is built for each query point x_*

Regression – Support Vector Regression

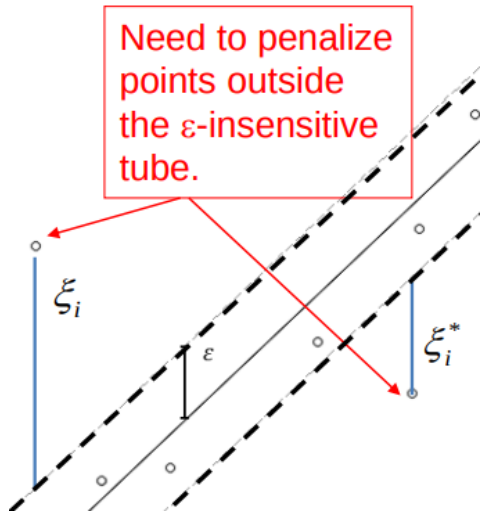
Model: $\hat{y} = f(x) = w^T x + b$

Goal: Minimize epsilon insensitive loss :

$$L = \begin{cases} 0 \text{ if } |y - f(x) \leq \epsilon| \\ |y - f(x)| - \epsilon \text{ otherwise} \end{cases}$$

Penalize only data points with prediction error more than ϵ (hyper-parameter)

$$\min \frac{1}{2}||w||^2 + \frac{C}{M}\sum_{i=1}^{M}(\xi_i + \xi_i^*)$$



subject to:

$$f(x_i) - y_i \leq \epsilon + \xi_i^*$$

$$y_i - f(x_i) \leq \epsilon + \xi_i, \xi_i, \xi_i^* \geq 0$$

Regression – Support Vector Regression

Solution is given by the dual optimization problem using the Lagrangian
At the optimal solution (KKT conditions):

$$\sum_{i=1}^{M} \alpha_i = \sum_{i=1}^{M} \alpha_i^*$$

Support vectors should be balanced in both sides of $\epsilon-$tube

$$w = \sum_{i=1}^{M} (\alpha_i - \alpha_i*) x_i$$

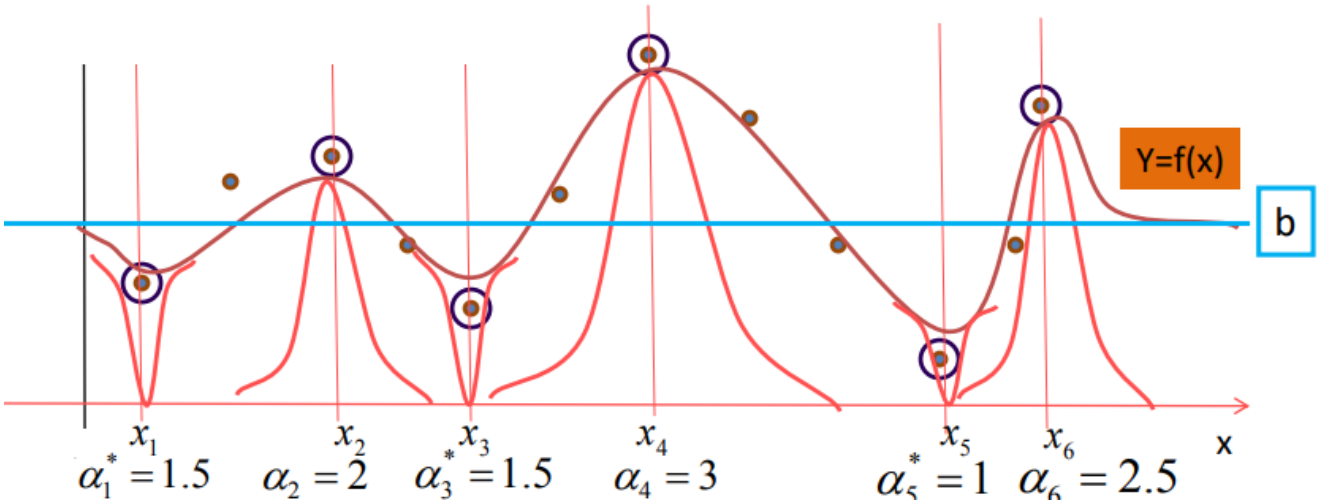
Linear combination of support vectors

Support vectors $\alpha \neq 0$ are located on or out of the epsilon tube.

Regression – SVR kernel trick

Apply kernel trick to achieve non-linear regression

$$y = f(x) = \sum_{i=1}^{M} (\alpha_i - \alpha_i*) k(x_i, x) + b$$

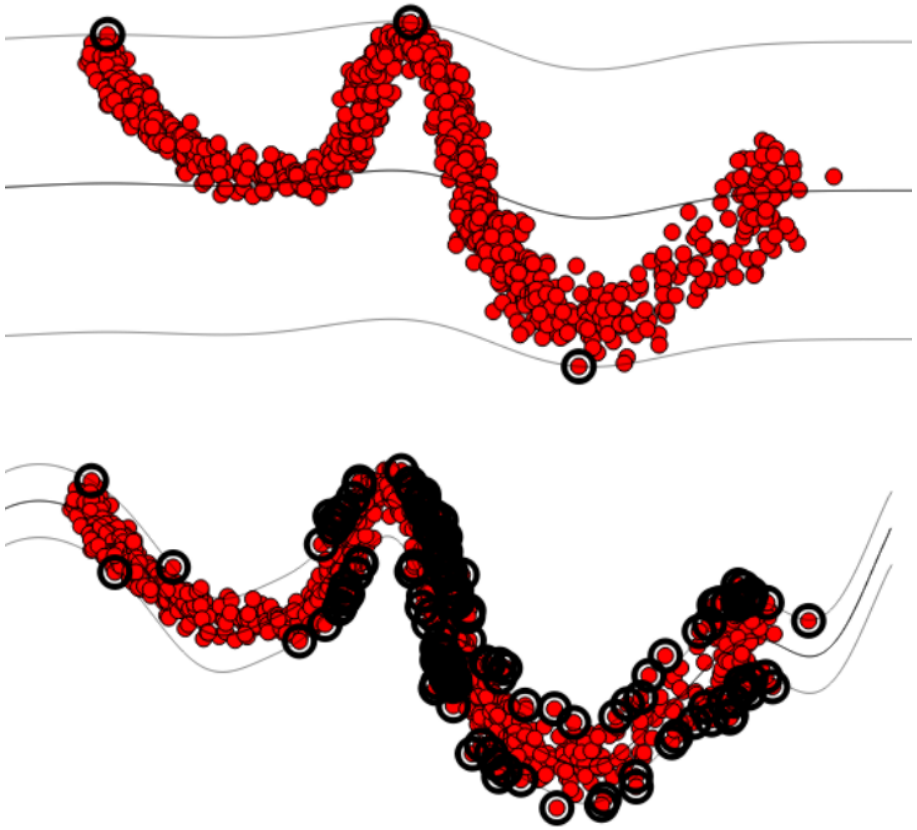


Away from support vectors prediction converges to b ($\bar{y}$)

$$b = \frac{1}{M} \sum_{j}^{M} \left(y_j - \sum_{i}^{M} (\alpha_i - \alpha_i*) k(x_j, x_i) \right)$$

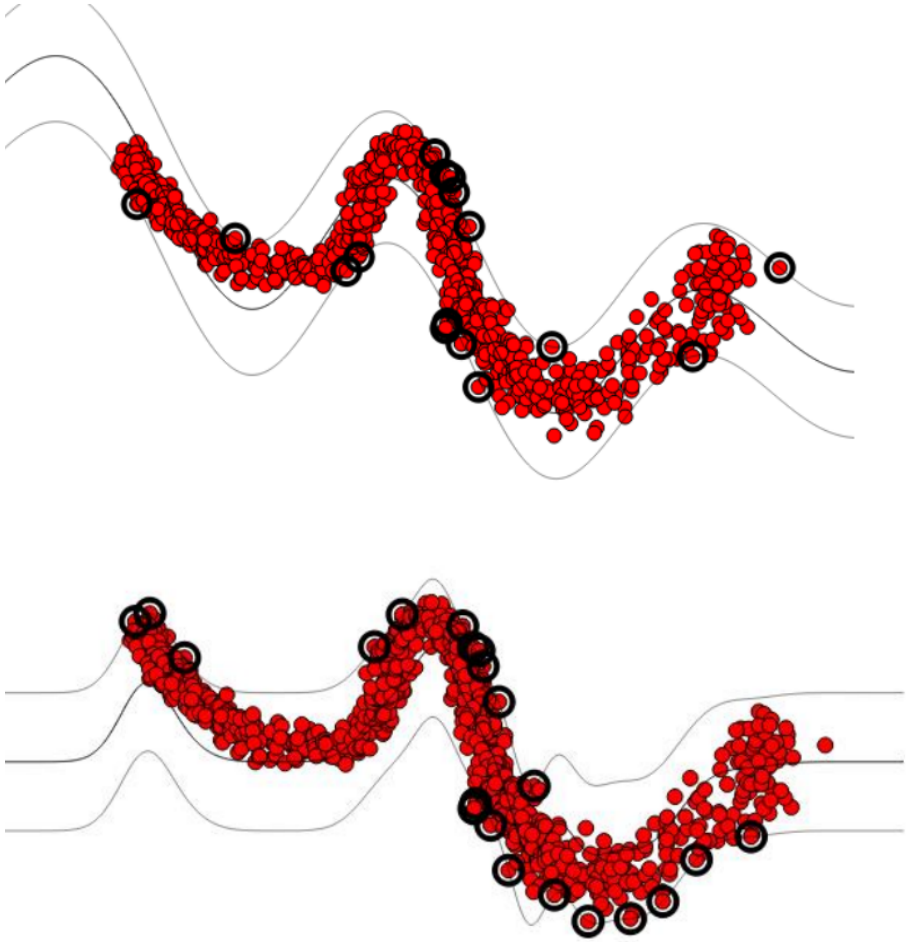
Regression – SVR hyper-parameters

Which hyper-parameter (ϵ, kernel width) has changed and how?



Regression – SVR hyper-parameters

Which hyper-parameter (ϵ, kernel width) has changed and how?



Regression – GMR

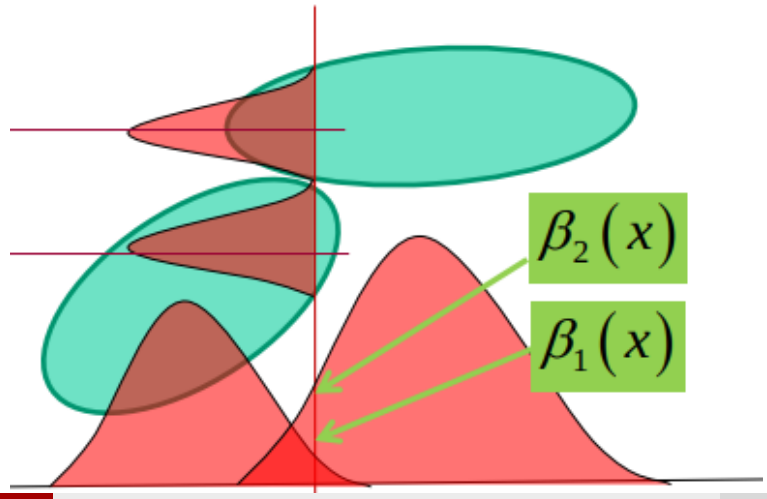
Fit a GMM in the whole dataset

Model: $p(x,y) \sim \sum_{k=1}^{K} \alpha_k \mathcal{N}(x,y;\mu_k,\Sigma_k)$

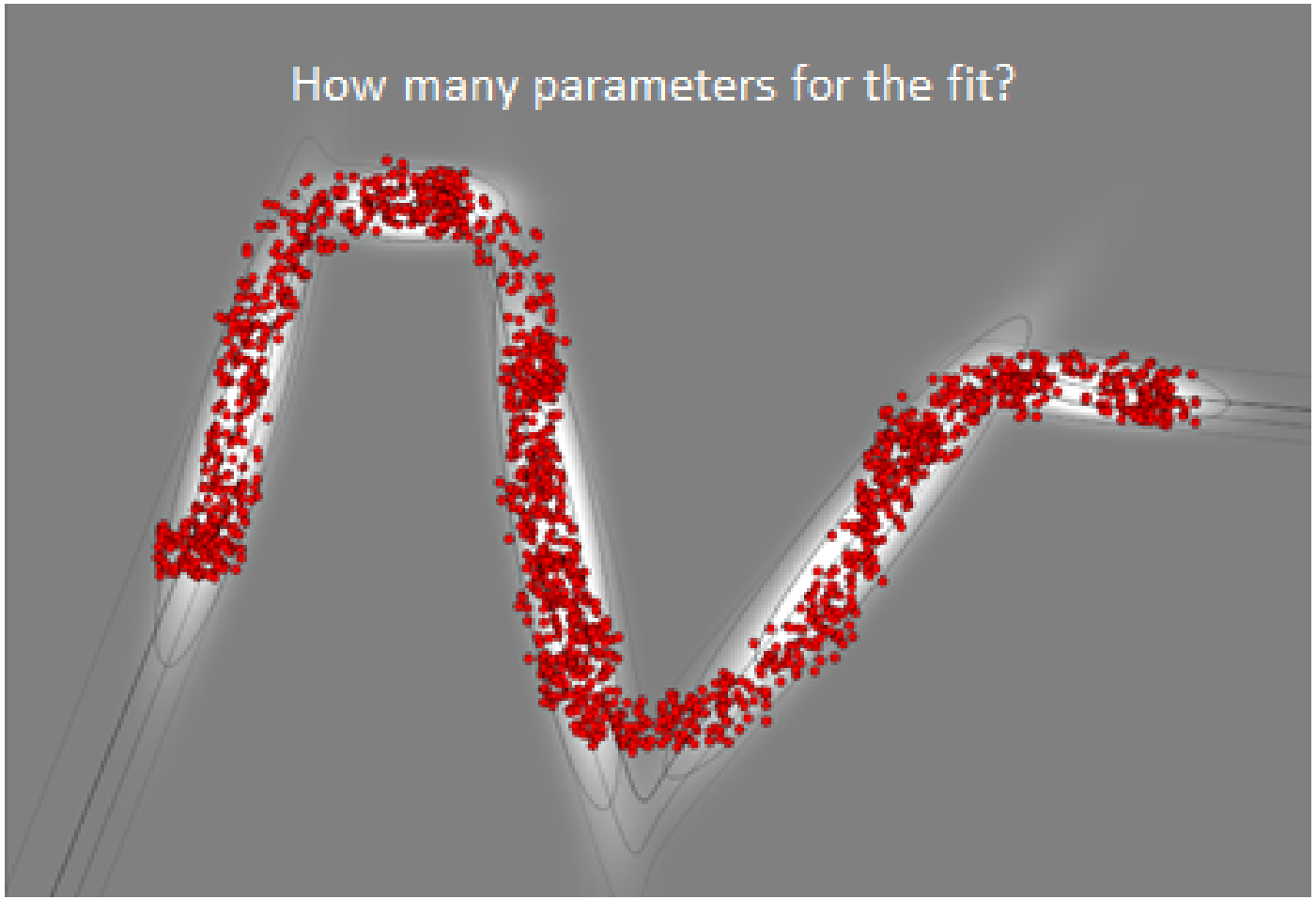
Goal: Estimate regressive signal $p(y|x) = \frac{p(x,y)}{p(x)}$

Prediction: : $y = E\{p(y|x)\} = \sum_{k=1}^{K} \beta_k(x)\, \mu_{y|x}^k(x)$

Variance: a weighted combination of the variances of the components around the weighted mean

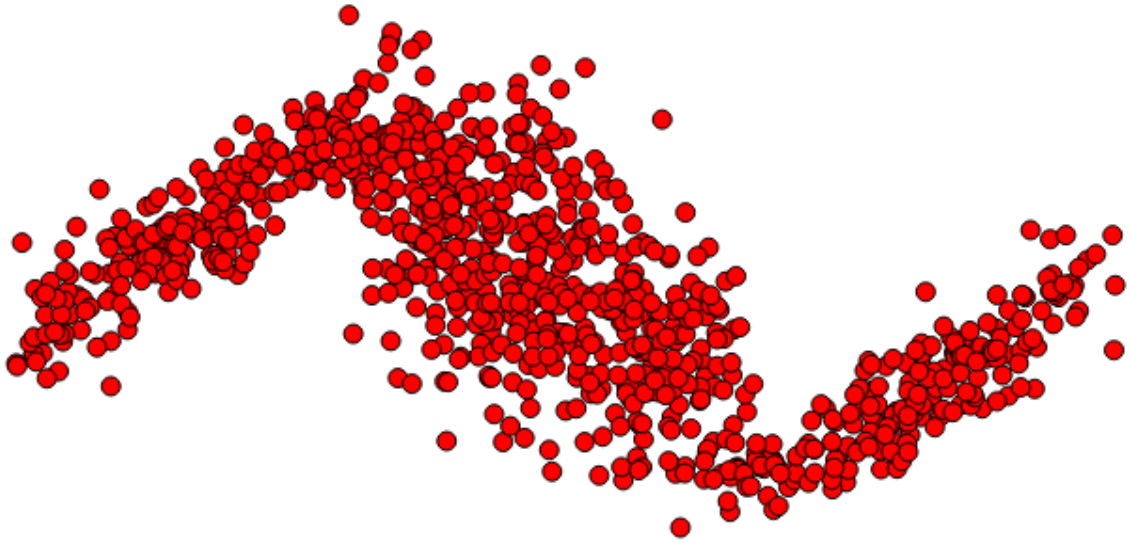


Regression – GMR



Regression – GMR vs SVR

GMR or SVR?



GMR better suited to encapsulate the variance of this dataset