CS 559: Gaussian Process & Supportive Vector Machine

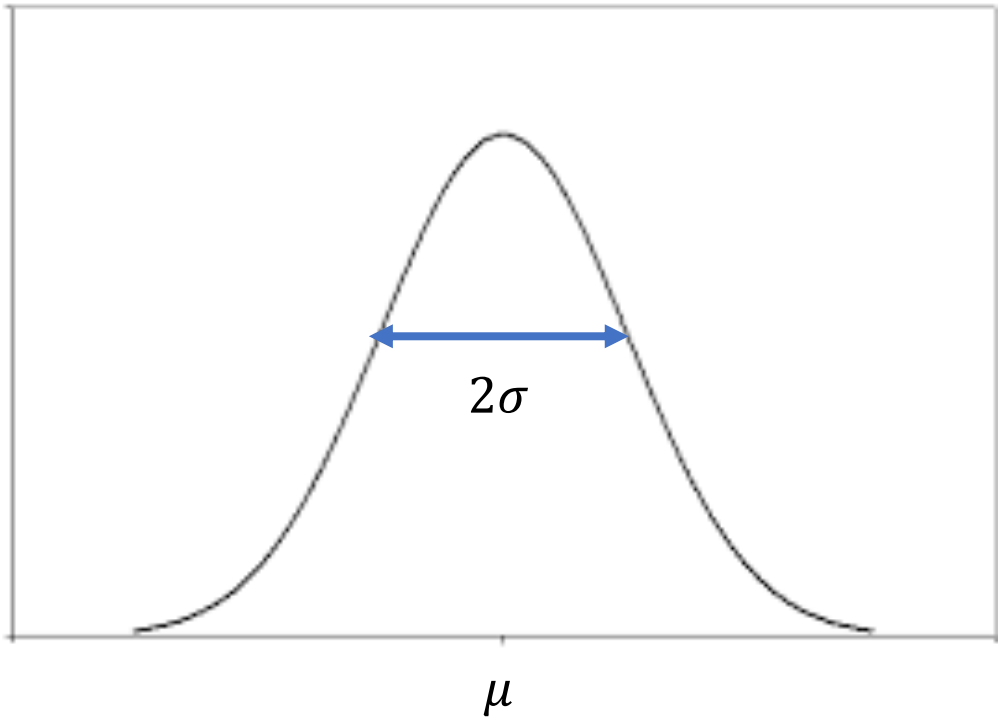
Lecture 5

Outline

- Gaussian Distribution
- Gaussian Process
- Supportive Vector Machine
- ML Demonstration

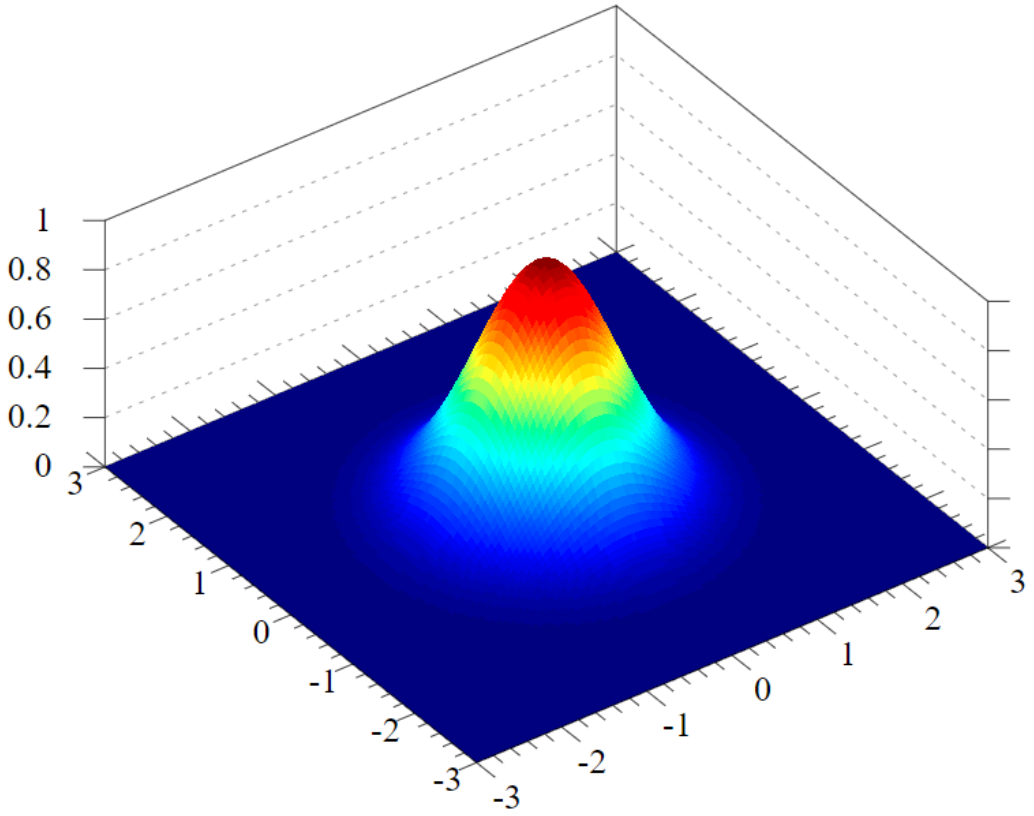
Gaussian Distributions

1-D Gaussian



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

Multivariate Gaussian



$$p(x|\mu,\Sigma) = \frac{1}{\sqrt{|2\pi\boldsymbol{\Sigma}|}} \exp\left\{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right\}$$

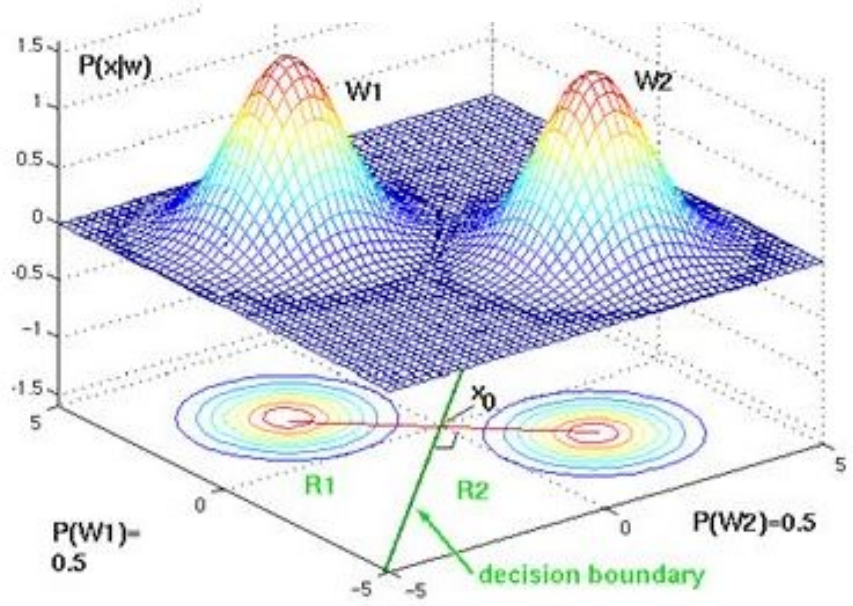
Gaussian Distributions

Multivariate Gaussian Example: A 2-D Gaussian

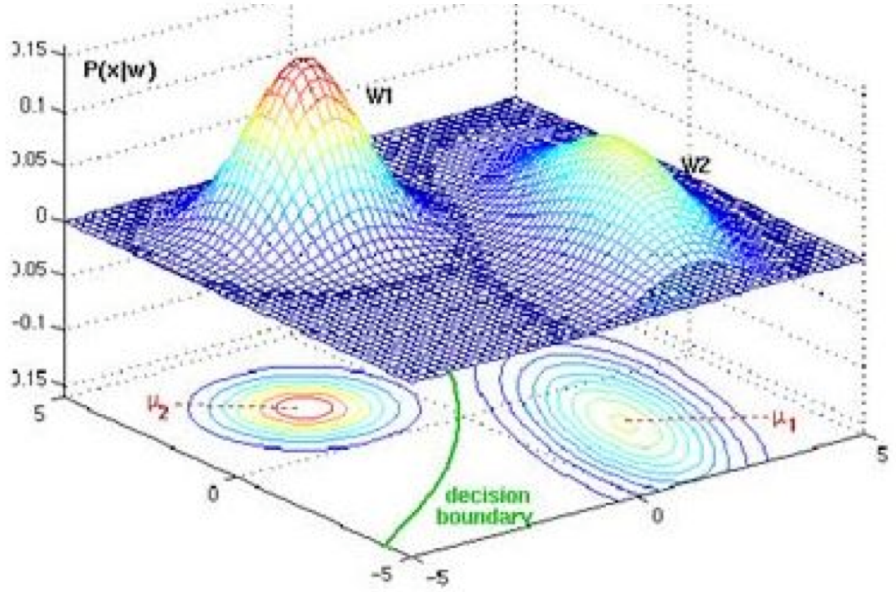
- A mean vector: $\mu = [\mu_1, \mu_2]$
- A covariance matrix: $\Sigma = \begin{bmatrix} \sigma_{11}^2 & \sigma_{21}^2 \\ \sigma_{12}^2 & \sigma_{22}^2 \end{bmatrix}$

where $\sigma_{ij}^2 = E\left[(x_i - \mu_i)(x_j - \mu_j)\right]$ is covariance and Σ is symmetric "positive semi-definite".

$\Sigma = \sigma^2 I$



$\Sigma =$ arbitrary

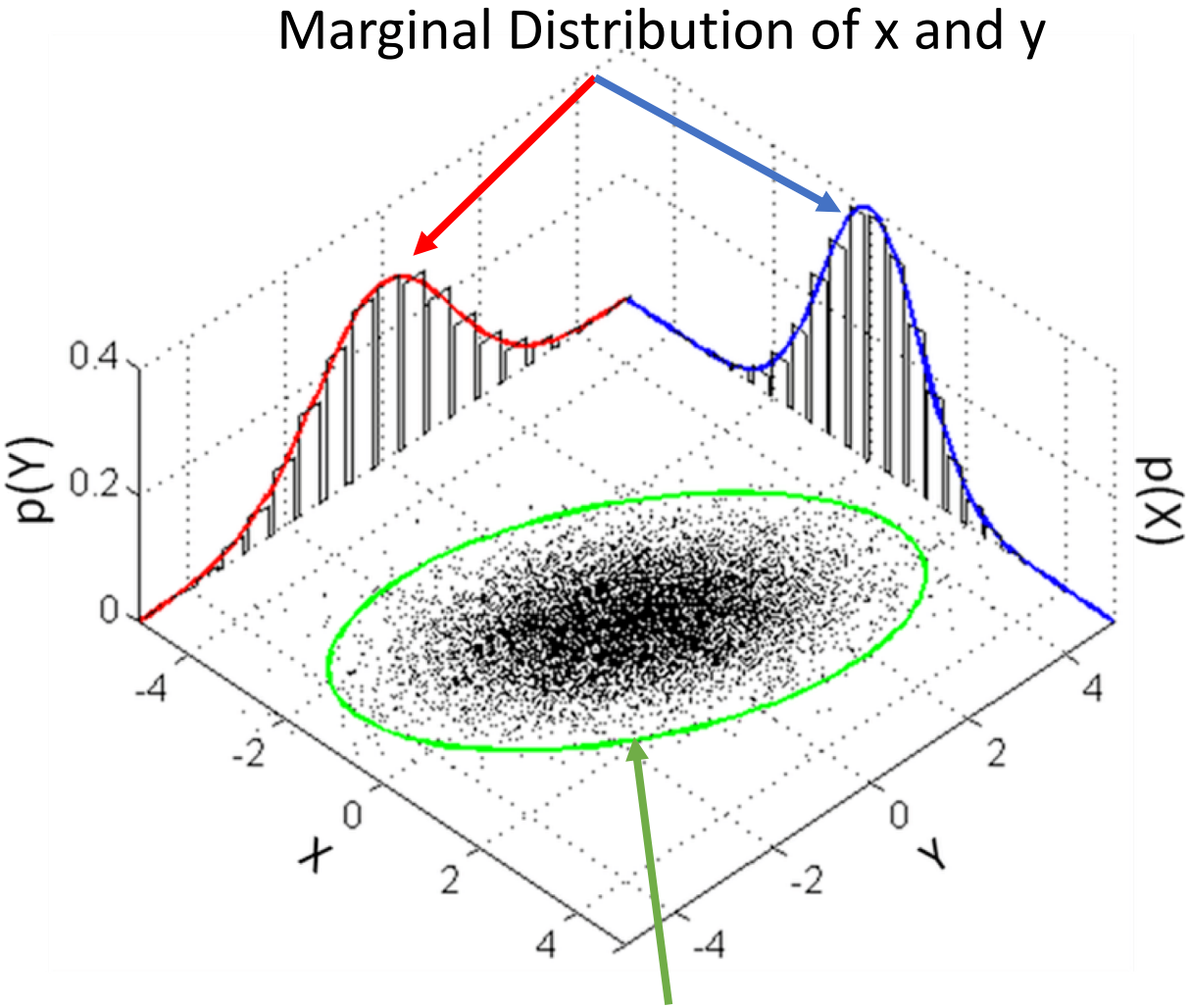


Gaussian Distributions

Marginal Distributions of Gaussians

- Given: $\boldsymbol{x} = (x_a, x_b), \boldsymbol{\mu} = (\mu_a, \mu_b), \Sigma = \begin{pmatrix} \Sigma_{aa} & \Sigma_{ab} \\ \Sigma_{ba} & \Sigma_{bb} \end{pmatrix}$
- Marginal distributions of Gaussians are Gaussian
- Marginal distribution - probability of variable w/o reference to other variables.

$$p(x_a) = \mathcal{N}(x_a | \mu_a, \Sigma_{aa})$$



Gaussian Distributions

Block Matrix Inversion

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix}^{-1} = \begin{bmatrix} (A - BD^{-1}C)^{-1} & -A^{-1}B(D - CA^{-1}B)^{-1} \\ -D^{-1}C(A - BD^{-1}C)^{-1} & (D - CA^{-1}B)^{-1} \end{bmatrix} = \begin{bmatrix} S_D^{-1} & -A^{-1}BS_A^{-1} \\ -D^{-1}CS_D^{-1} & S_A^{-1} \end{bmatrix}$$

Schur complements:

$$S_A = D - CA^{-1}B$$
$$S_D = A - BD^{-1}C$$

Conditional Distribution

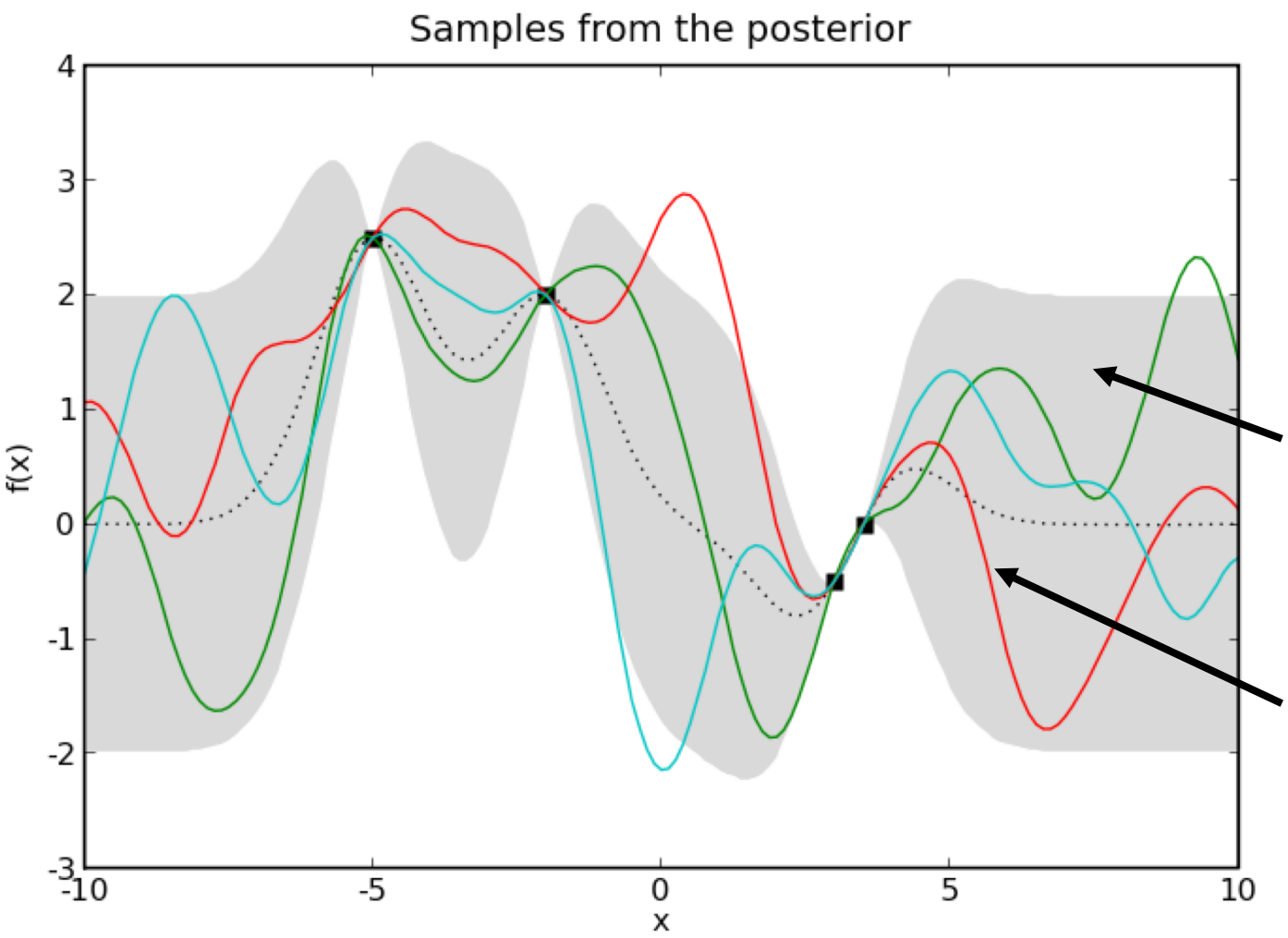
Notation: $\Sigma = \begin{pmatrix} \Sigma_{aa} & \Sigma_{ab} \\ \Sigma_{ba} & \Sigma_{bb} \end{pmatrix}$ $\Lambda = \Sigma^{-1} = \begin{pmatrix} \Lambda_{aa} & \Lambda_{ab} \\ \Lambda_{ba} & \Lambda_{bb} \end{pmatrix}$

Distribution: $p(x_a|x_b) = \mathcal{N}\left(x_a \middle| \mu_{(a|b)}, \Lambda_{aa}^{-1}\right)$

where $\mu_{(a|b)} = \mu_a - \Lambda_{aa}^{-1}\Lambda_{ab}(x_b - \mu_a) = \mu_a - \Sigma_{ab}\Sigma_{bb}^{-1}(x_b - \mu_a)$

and Schur complement $\Lambda_{aa}^{-1} = \Sigma_{aa} - \Sigma_{ab}\Sigma_{bb}^{-1}\Sigma_{ba}$

Regression



Linear Regression:

$$y = w^T x + \epsilon$$

$$y = w^T \phi(x) + \epsilon$$

Linear regression gives a predictive model for one particular parameter w.

How can we do regression?

- We would like a method that could provide confidence during the estimation.

- How about kernelize linear regression?

Gaussian Process

Definition:

- Probability distribution indexed by an arbitrary set.
- Each element gets a Gaussian distribution over the reals with mean μ
- Distributions are dependent/correlated as defined by k(x,z)
- Any finite subset of indices defines a multivariate Gaussian distribution
- $f \sim GP(\mu, k)$

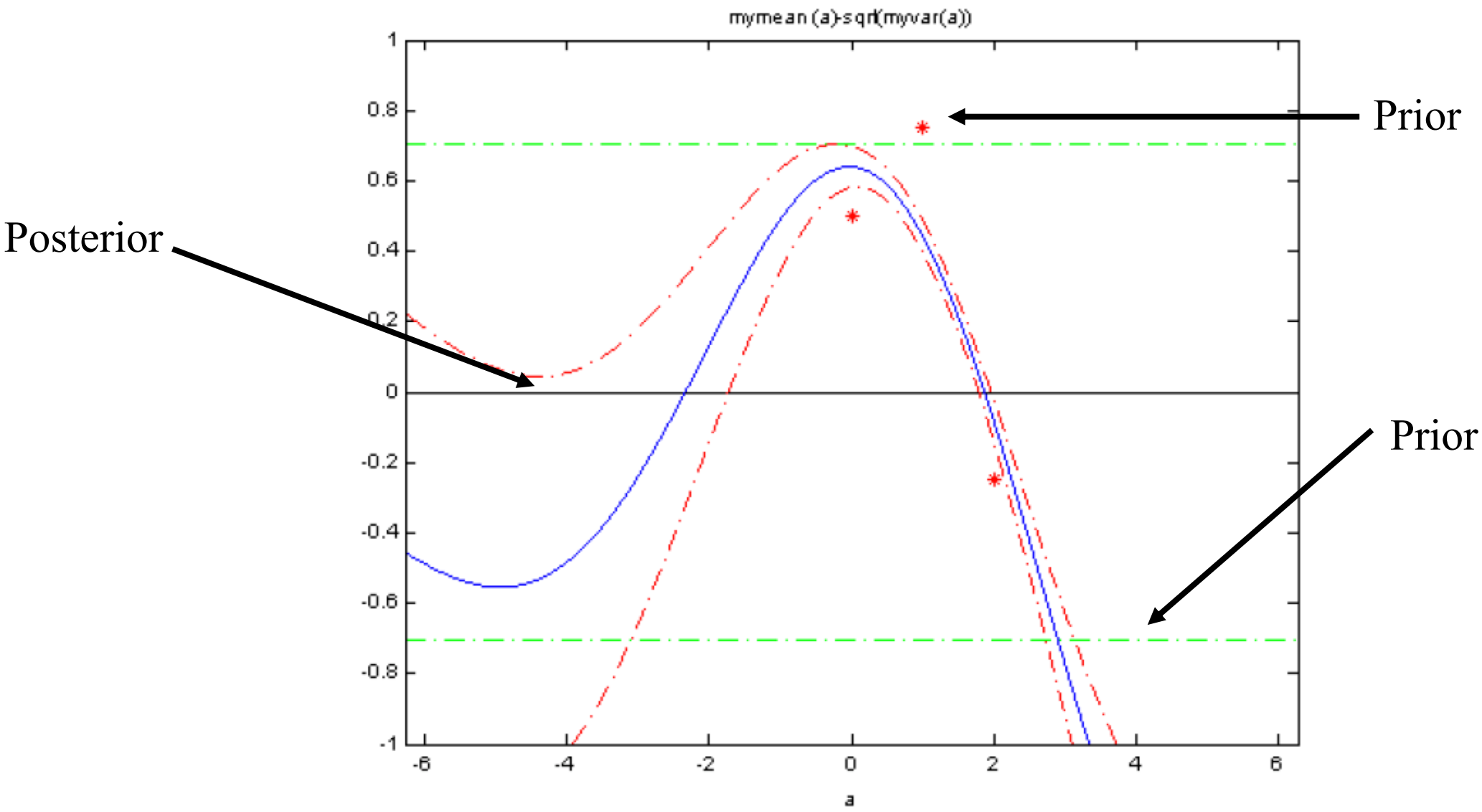
What we get…

- Distribution over functions and functions for any domain index sets.

General Procedure:

- Start with GP prior
- Get some data
- Compute a posterior

Gaussian Process



GP – Weight Space View

GP = Bayesian ridge regression in feature space + kernel trick to carry out computations

Training Set: $D = \{(x_i, y_i) | i = 1, \dots, n\}$

- x_i is the input vector of D-dimension
- y_i is a scalar output or target

Training Data: X and Y

Design matrix $X = [x_1 | \cdots | x_n] \in \mathbb{R}^{D \times n}$

Target vector: $Y = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix} \in \mathbb{R}^n$

$f(x) = x^T w \in \mathbb{R}, x, w \in \mathbb{R}^D$

$y = f(x) + \epsilon = x^T w + \epsilon \in \mathbb{R}$ where $\epsilon \sim \mathcal{N}(0, \sigma^2) \in \mathbb{R}$

GP – Weight Space View LR w/ Gaussian Noise

Calculate the Likelihood and put w prior over parameters w, $w \sim \mathcal{N}(0, \Sigma)$:

$$p(Y|X,w) = \prod_{i=1}^{n} p\left(y_i \middle| x_i^T w\right)$$

$$= \prod_{i=1}^{n} \mathcal{N}\left(x_i^T w, \sigma^2\right) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-\frac{\left(y_i - x_i^T w\right)^2}{2\sigma^2}\right]$$

$$= \frac{1}{(2\pi\sigma^2)^{\frac{n}{2}}} \exp\left[-\frac{1}{2\sigma^2}\left|\left|Y - X^T w\right|\right|^2\right]$$

$$= \mathcal{N}(X^T w, \sigma^2 I)$$

GP – Weight Space View LR w/ Gaussian Noise

The prior: $w \sim \mathcal{N}(0, \Sigma)$

The posterior:

$$p(w|X, Y) = \frac{p(Y|X, w)p(w)}{p(Y|X)} = \frac{p(Y|X, w)p(w)}{\int p(Y|X, w)dw}$$

$$= \frac{\mathcal{N}\left(X^T w, \sigma^2\right)\mathcal{N}(0, \Sigma)}{\int \mathcal{N}(X^T w, \sigma^2)\mathcal{N}(0, \Sigma)dw}$$

$$\sim \mathcal{N}\left(X^T w, \sigma^2\right)\mathcal{N}(0, \Sigma)$$

GP – Weight Space View LR w/ Gaussian Noise

Ridge Regression

$$p(w|X,Y) \sim \mathcal{N}\left(X^T w, \sigma^2\right)\mathcal{N}(0,\Sigma)$$

$$\sim \exp\left[-\frac{1}{2\sigma^2}\left(Y - X^T w\right)^T\left(Y - X^T w\right)\right]\exp\left[-\frac{1}{2}w^T\Sigma^{-1}w\right]$$

$$\sim \exp[-\frac{1}{2}(w-\bar{w})^T\underbrace{\left(\frac{1}{\sigma^2}XX^T + \Sigma^{-1}\right)}_{=A}(w-\bar{w})$$

$$\sim \mathcal{N}\left(\bar{w}, A^{-1}\right)$$

After "completing the square"

where $\bar{w} = \sigma^{-2}\underbrace{(\sigma^{-2}XX^T + \Sigma^{-1})^{-1}}_{A^{-1} \in \mathbb{R}^{D\times D}}XY \in \mathbb{R}^D$ and $A = (\sigma^{-2}XX^T + \Sigma^{-1}) \in \mathbb{R}^{D\times D}$

GP – Weight Space View LR w/ Gaussian Noise

Use the posterior to predict f in a test point x_*:

$f(x) = x^T w \in \mathbb{R}, x, w \in \mathbb{R}^D$

$y = f(x) + \epsilon = x^T w + \epsilon \in \mathbb{R}$ where $\epsilon \sim \mathcal{N}(0, \sigma^2) \in \mathbb{R}$

$$p(f_*|x_*, X, Y) = \int \underbrace{p(f_*|x_*, w)}_{\delta(f_*, x_*^T, w)} \underbrace{p(w|Y, X)}_{\mathcal{N}(\overline{w}, A^{-1})} dw$$

$$= N\left(x_*^T \overline{w}, x_*^T A^{-1} x_*\right)$$

GP – Explicit Feature Space

$$\phi(x) = [x_1, x_1 x_2^2, \dots]^T \in \mathbb{R}^N$$
$$\Phi(X) = [\phi(x_1)|\phi(x_2)|\cdots|\phi(x_n)] \in \mathbb{R}^{N \times n}$$

$\Phi(X)$ is the aggregation of columns $\phi(x)$ for all cases in the training set.

The model is $f(x) = \phi(x)^T w$ and $\phi(x), w \in \mathbb{R}^N$

$$y = f(x) + \epsilon = \phi(x)^T w + \epsilon \in \mathbb{R}$$

GP – Explicit Feature Space

The predictive distribution after feature map:

$$p(f_*|x_*, X, Y) = \mathcal{N}\left(\phi(x_*)^T \bar{w}, \phi(x_*)^T A^{-1} \phi(x_*)\right)$$

where $\bar{w} = \sigma^{-2}(\sigma^{-2}\Phi(X)\Phi(X)^T + \Sigma^{-1})\Phi(X)Y \in \mathbb{R}^N$ and

$A^{-1} \in R^{N\times N}$ $\in R^{N\times n}$ $\in R^{n\times 1}$

$$A = (\sigma^{-2}\Phi(X)\Phi(X)^T + \Sigma^{-1}) \in \mathbb{R}^{N\times N}$$

A problem is we need $N\times N$ matrix inversion…

GP – Explicit Feature Space

We can shorten this equation using

$$\phi_* = \phi(x) \in \mathbb{R}^N, \Phi = \Phi(X) \in \mathbb{R}^{N\times n},$$
$$\bar{w} = \sigma^{-2}(\sigma^{-2}\Phi\Phi^T + \Sigma^{-1})\Phi Y \in \mathbb{R}^N,$$
$$A = (\sigma^{-2}\Phi\Phi^T + \Sigma^{-1}) \in \mathbb{R}^{N\times N}$$

We only need to work with ***n*×*n* matrices**, not $N\times N$!

$$p(f_*|x_*, X, Y) = \mathcal{N}(\phi_*^T\bar{w}, \phi_*^T A^{-1}\phi)$$
$$= \mathcal{N}(\sigma^{-2}\phi_*^T[\sigma^{-2}\Phi\Phi^T + \Sigma^{-1}]^{-1}\Phi Y, \phi_*^T[\sigma^{-2}\Phi\Phi^T + \Sigma^{-1}]^{-1}\phi_*)$$

$\mathbb{R}^{N\times N}$ $\mathbb{R}$

Let $K = \Phi(X)^T\Sigma\Phi(X)$,

$$p(f_*|x_*, X, Y)$$
$$= \mathcal{N}\left(\phi_*^T\Sigma\Phi(K + \sigma^2 I)^{-1}Y, \phi_*^T\Sigma\phi_* - \phi_*^T\Sigma\Phi(K + \sigma^2 I)^{-1}\Phi^T\Sigma\phi_*\right)$$

$\mathbb{R}^{1\times n}$ $\mathbb{R}^{n\times n}$ $\mathbb{R}^{n\times 1}$ $\mathbb{R}^{1\times 1}$ $\mathbb{R}^{1\times n}$ $\mathbb{R}^{n\times n}$ $\mathbb{R}^{n\times 1}$

GP – Explicit Feature Space

$$p(f_*|x_*, X, Y)$$
$$= \mathcal{N}\left(\phi_*^T \Sigma \Phi (K + \sigma^2 I)^{-1} Y, \phi_*^T \Sigma \phi_* - \phi_*^T \Sigma \Phi (K + \sigma^2 I)^{-1} \Phi^T \Sigma \phi_*\right)$$

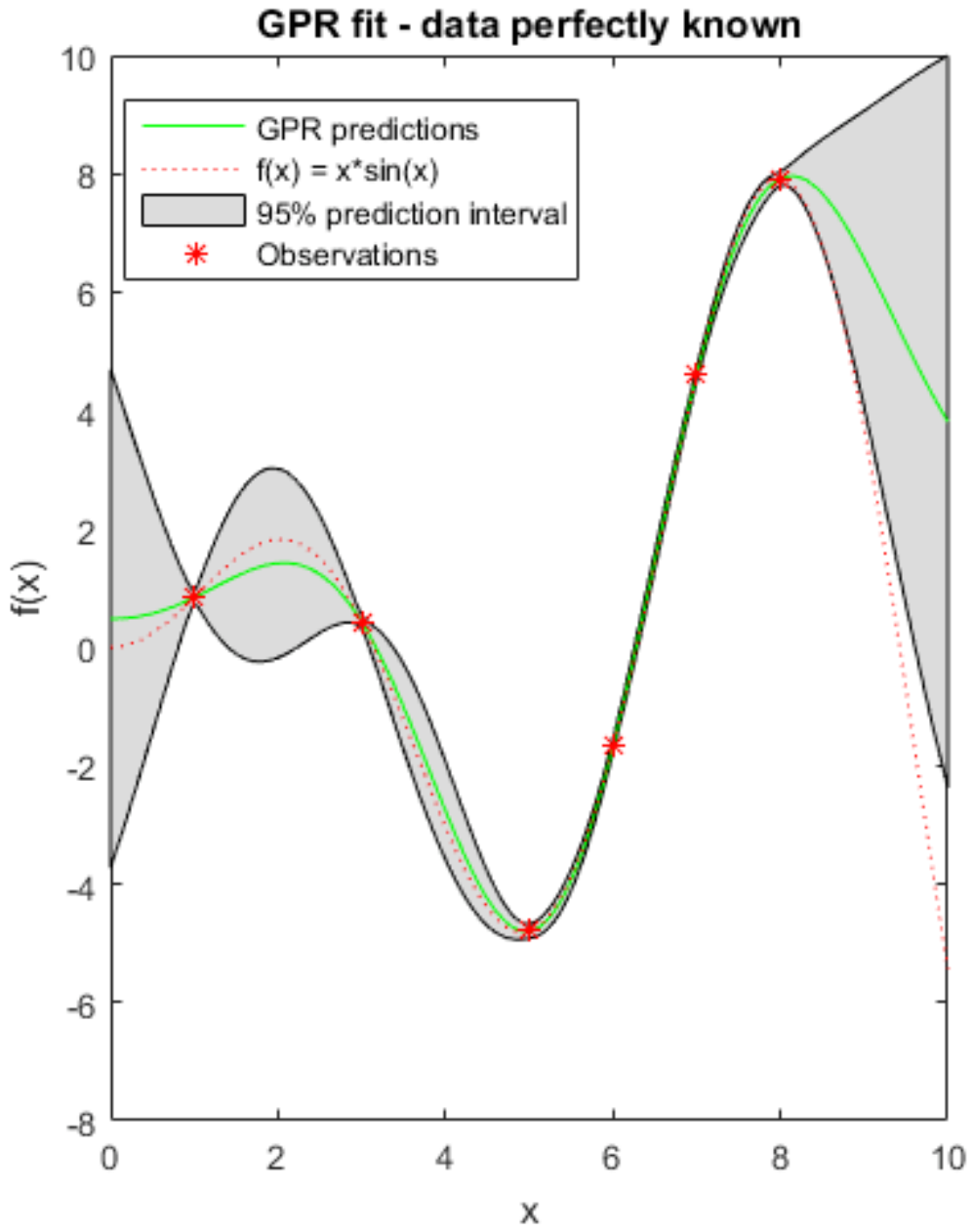
The feature space **always** enters in the form of

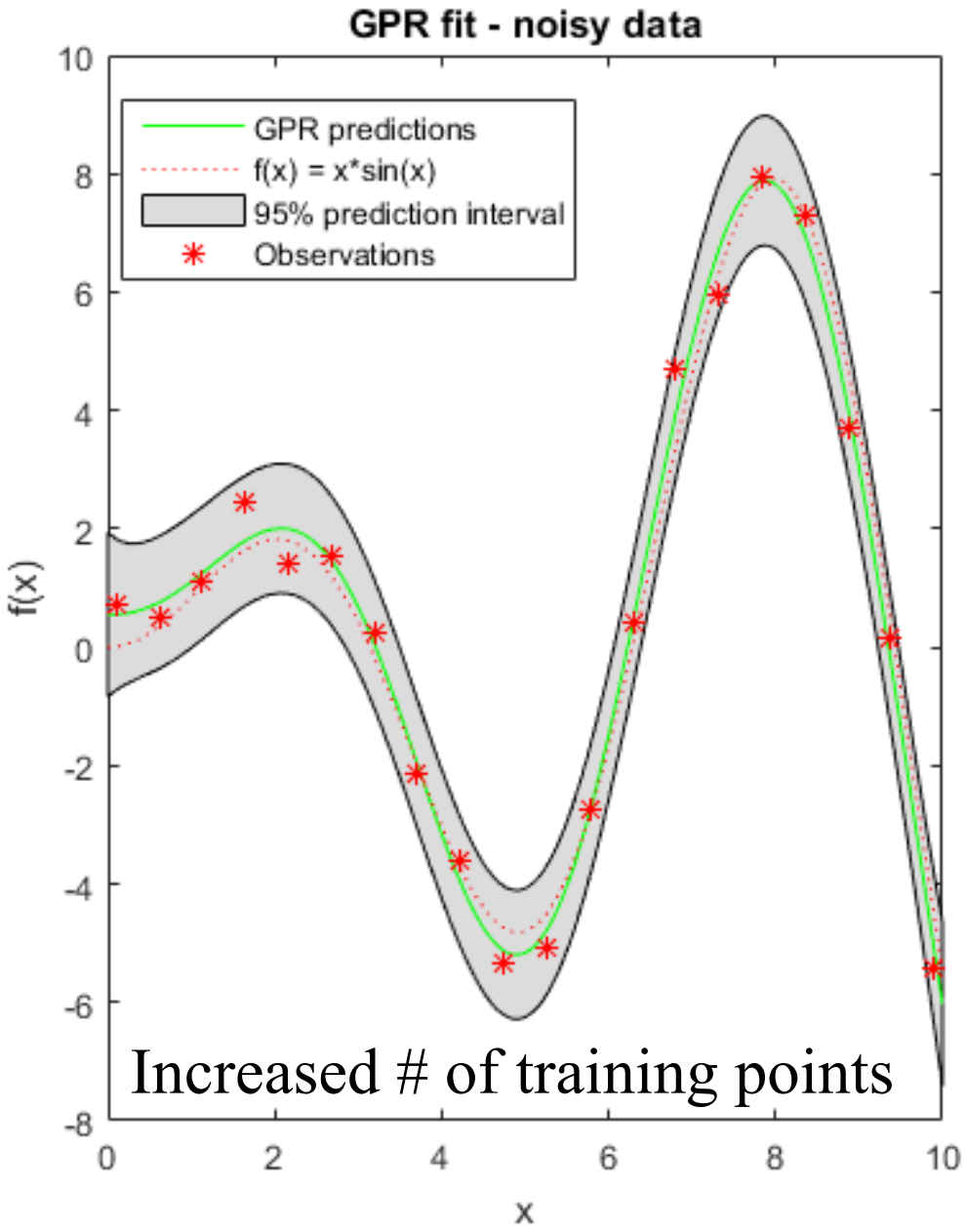
$$\phi_*^T \Sigma \phi_*, \phi_*^T \Sigma \phi, \phi^T \Sigma \phi \in \mathbb{R}^{n \times n}$$

Let form of $k = \phi(x)^T \Sigma \phi(x')$ - a covariance function or kernel.

Inner product in the feature space: $\psi(x) = \Sigma^{1/2} \phi(x)$

GP – Results


GPR fit - data perfectly known
GPR predictions
f(x) = x*sin(x)
95% prediction interval
Observations
f(x)
x


GPR fit - noisy data
GPR predictions
f(x) = x*sin(x)
95% prediction interval
Observations
f(x)
x
Increased # of training points

GP – Function Space View

Definition: GP is a collection of random variables, any finite number of which have a joint Gaussian distribution.

$$m(x) = E[f(x)] \qquad \text{(mean function)}$$

$$k(x, x') = E\left[\left(f(x) - m(x)\right)\left(f(x') - m(x')\right)^T\right] \in \mathbb{R} \qquad \text{(covariance func.)}$$

GP is completely specified by its mean function $m(x)$ and covariance function $k(x, x')$.

$$f(x) \sim GP\left(m(x), k(x, x')\right) \in \mathbb{R}, x \in \mathbb{R}^D$$

GP – Function Space View

For each $x \in \mathbb{R}^D$ we associate a Gaussian variable $f(x)$ such that $f(x) \in \mathcal{N}\big(m(x), k(x,x)\big)$ and its correlation with other $f(x')$ variables is $k(x,x')$.

$$\mathbb{R} \ni f(x) \sim \mathcal{N}\big(m(x), f(x,x)\big)$$

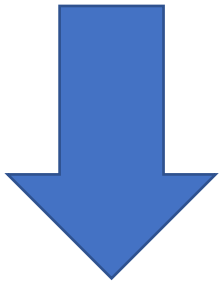
$$\begin{bmatrix} f(x) \\ f(x') \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} m(x) \\ m(x') \end{bmatrix}, \begin{bmatrix} k(x,x) & k(x',x) \\ k(x,x') & k(x',x') \end{bmatrix}\right)$$

GP – noise free observations

Training set: $D = \{(x_i, f_i) | i = 1, \dots, n\}$

$$X = \begin{bmatrix} x_1^T \\ \vdots \\ x_n^T \end{bmatrix} \in \mathbb{R}^{n \times D}, n \text{ training inputs.} \qquad X_* = \begin{bmatrix} x_{*1}^T \\ \vdots \\ x_{*m}^T \end{bmatrix} \in \mathbb{R}^{m \times D}, m \text{ test inputs}$$

$$f = \begin{bmatrix} f_1 \\ \vdots \\ f_n \end{bmatrix} \in \mathbb{R}^{n \times D}, n \text{ training inputs} \qquad f_* = \begin{bmatrix} f_{*1} \\ \vdots \\ f_{*m} \end{bmatrix} \in \mathbb{R}^{m \times D}, m \text{ test inputs}$$



$$\begin{bmatrix} f \\ f_* \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 0_n \\ 0_m \end{bmatrix}, \begin{bmatrix} k(X,X) & k(X,X_*) \\ k(X_*,X) & k(X_*,X_*) \end{bmatrix} \right)$$

GP – noise free observations

$$\begin{bmatrix} f \\ f_* \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 0_n \\ 0_m \end{bmatrix}, \begin{bmatrix} k(X,X) & k(X,X_*) \\ k(X_*,X) & k(X_*,X_*) \end{bmatrix}\right)$$

$$\Sigma = \begin{pmatrix} \Sigma_{aa} & \Sigma_{ab} \\ \Sigma_{ba} & \Sigma_{bb} \end{pmatrix} \Lambda = \Sigma^{-1} = \begin{pmatrix} \Lambda_{aa} & \Lambda_{ab} \\ \Lambda_{ba} & \Lambda_{bb} \end{pmatrix}$$

$$p(x_a|x_b) = \mathcal{N}\left(x_a \middle| \mu_{a|b}, \Lambda_{aa}^{-1}\right)$$

$$\mu_{a|b} = \mu_a - \Lambda_{aa}^{-1}\Lambda_{ab}(x_b - \mu_a) = \mu_a - \Sigma_{aa}\Sigma_{bb}^{-1}(x_b - \mu_a)$$

$$\Lambda_{aa}^{-1} = \Sigma_{aa} - \Sigma_{ab}\Sigma_{bb}^{-1}\Sigma_{ba}$$

$$p(f_*|X_*, X, f) = \mathcal{N}\left(k(X_*,X)k(X,X)^{-1}f, k(X_*,X_*) - k(X_*,X)k(X,X)^{-1}k(X,X_*)\right)$$

If $X_* = X \Longrightarrow f_* = f$ and the covariance is 0.

$$p(f_*|X_*, X, f) = \mathcal{N}\left(\phi_*^T\Sigma\Phi(K + \sigma^2 I)^{-1}f, \phi_*^T\Sigma\phi_* - \phi_*^T\Sigma\Phi(K + \sigma^2 I)^{-1}\Phi^{\mathrm{T}}\Sigma\phi_*\right)$$

GP – noise observations

$$y = f(x) + \epsilon \in \mathbb{R}$$
$$\epsilon \sim \mathcal{N}(0, \sigma^2) \in \mathbb{R}$$

$$\begin{bmatrix} f \\ f_* \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 0_n \\ 0_m \end{bmatrix}, \begin{bmatrix} k(X,X) & k(X,X_*) \\ k(X_*,X) & k(X_*,X_*) \end{bmatrix}\right)$$

$$\Rightarrow cov(y_p, y_q) = k(x_p, x_q) + \sigma^2 \delta_{p,q}$$ ← (Dirac Delta Function)

$$\Rightarrow cov([y_1, \dots, y_n]) = k(X,X) + \sigma^2 I \in \mathbb{R}^{n \times n}$$

$$\Rightarrow \begin{bmatrix} y \\ f_* \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 0_n \\ 0_m \end{bmatrix}, \begin{bmatrix} k(X,X) + \sigma^2 I & k(X,X_*) \\ k(X_*,X) & k(X_*,X_*) \end{bmatrix}\right)$$ ← (New joint distribution)

The posterior for the noisy observations:

$$p(f_*|X,Y,X_*) = \mathcal{N}\left(\bar{f}_*, cov(f_*)\right)$$

GP – noise observations

$$\Rightarrow \begin{bmatrix} y \\ f_* \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 0_n \\ 0_m \end{bmatrix}, \begin{bmatrix} k(X,X)+\sigma^2 I & k(X,X_*) \\ k(X_*,X) & k(X_*,X_*) \end{bmatrix}\right)$$

$$\Sigma = \begin{pmatrix} \Sigma_{aa} & \Sigma_{ab} \\ \Sigma_{ba} & \Sigma_{bb} \end{pmatrix} \quad \Lambda = \Sigma^{-1} = \begin{pmatrix} \Lambda_{aa} & \Lambda_{ab} \\ \Lambda_{ba} & \Lambda_{bb} \end{pmatrix}$$

$$p(x_a|x_b) = \mathcal{N}\left(x_a \middle| \mu_{a|b}, \Lambda_{aa}^{-1}\right)$$

$$\mu_{a|b} = \mu_a - \Lambda_{aa}^{-1}\Lambda_{ab}(x_b - \mu_a) = \mu_a - \Sigma_{aa}\Sigma_{bb}^{-1}(x_b - \mu_a)$$

$$\Lambda_{aa}^{-1} = \Sigma_{aa} - \Sigma_{ab}\Sigma_{bb}^{-1}\Sigma_{ba}$$

$$p(f_*|X,Y,X_*) = \mathcal{N}\left(\bar{f}_*, cov(f_*)\right)$$

where

$$\bar{f}_* = E[f_*|X,Y,X_*] = k(X_*,X)[k(X,X)+\sigma^2 I]^{-1}Y \in \mathbb{R}^m$$

$$cov(f_*) = k(X_*,X_*) - k(X_*,X)[k(X,X)+\sigma^2 I]^{-1}K(X,X_*) \in \mathbb{R}^{m\times m}$$

$$\mathcal{N}(\phi_*^T\Sigma\Phi(K+\sigma^2 I)^{-1}Y, \phi_*^T\Sigma\phi_* - \phi_*^T\Sigma\Phi(K+\sigma^2 I)^{-1}\Phi\Sigma\phi_*)$$ ← (Weight Space View)

If $k = \phi(x)^T\Sigma\phi(x')$

GP – noise observations

Short Notation for a single test point x_*:

$$K = k(X, X) \in \mathbb{R}^{n \times n}$$

$$K_* = k(X, X_*) \in \mathbb{R}^{n \times m}$$

$$k(x_*) = k_* = k(X, x_*) = \begin{bmatrix} k(x_1, x_*) \\ \vdots \\ k(x_n, x_*) \end{bmatrix} \in \mathbb{R}^n$$

$$\bar{f}_* = k_*^T [K + \sigma^2 I]^{-1} Y \in \mathbb{R}$$

$\mathbb{R}^{1 \times n}$ $\mathbb{R}^{n \times n}$ $\mathbb{R}^n$

$$cov(f_*) = k(x_*, x_*) - k_*^T [K + \sigma^2 I]^{-1} k_* \in \mathbb{R}$$

$\mathbb{R}$ $\mathbb{R}^{1 \times n}$ $\mathbb{R}^{n \times n}$ $\mathbb{R}^n$

GP – noise observations

Linear combination of observation Y - linear predictor

Let $\beta^T = k_*^T \left[K + \sigma^2 I\right]^{-1} \in \mathbb{R}^{1\times n}$

$$\bar{f}_* = \sum_{i=1}^{N} \beta_i^T y_i$$

$$\bar{f}_* = \underbrace{k_*^T \left[K + \sigma^2 I\right]^{-1}}_{\mathbb{R}^{\mathbf{1}\times \boldsymbol{n}}} \underbrace{Y}_{\mathbb{R}^n} \in \mathbb{R}$$

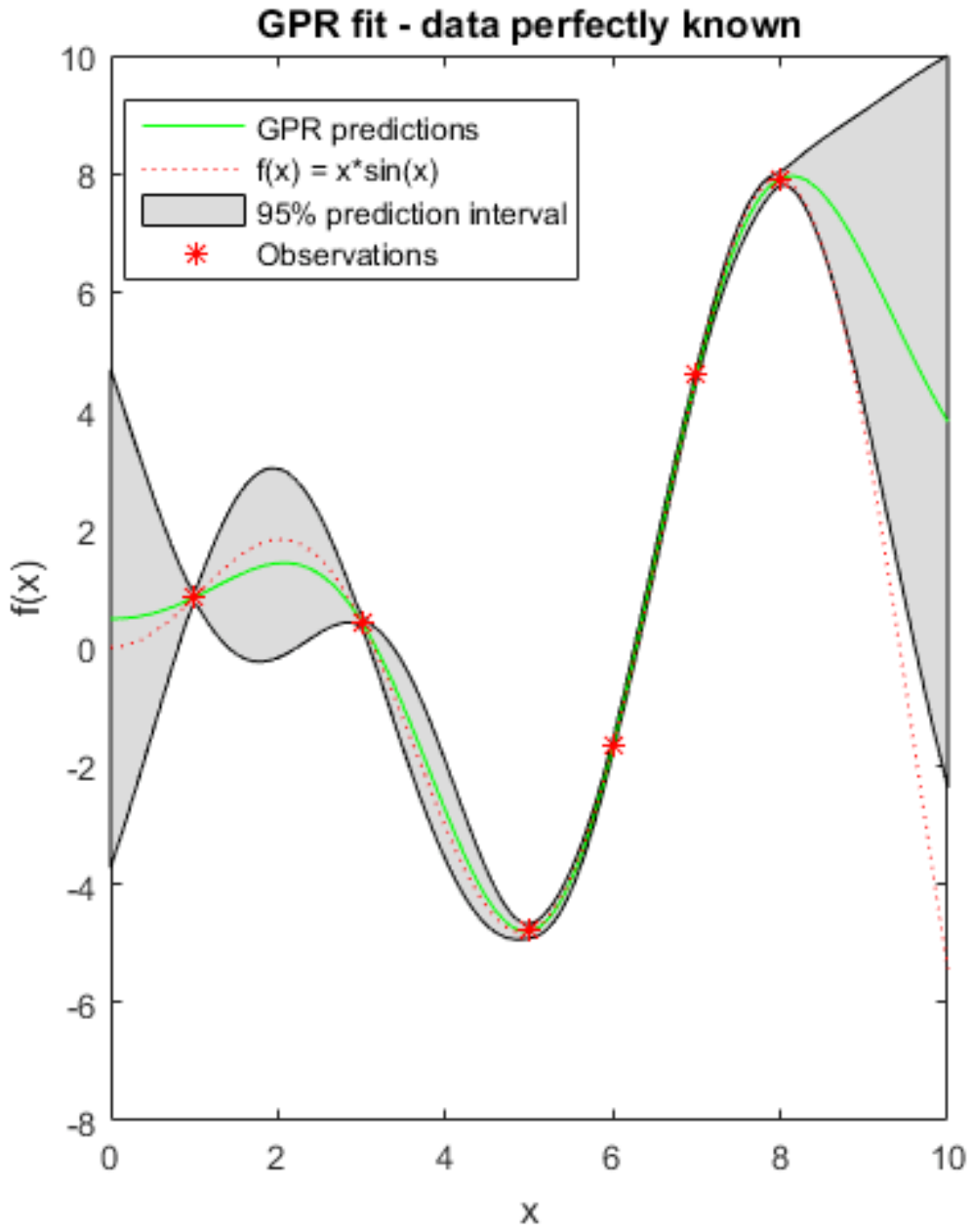
A linear combination of n kernel function - representer theorem

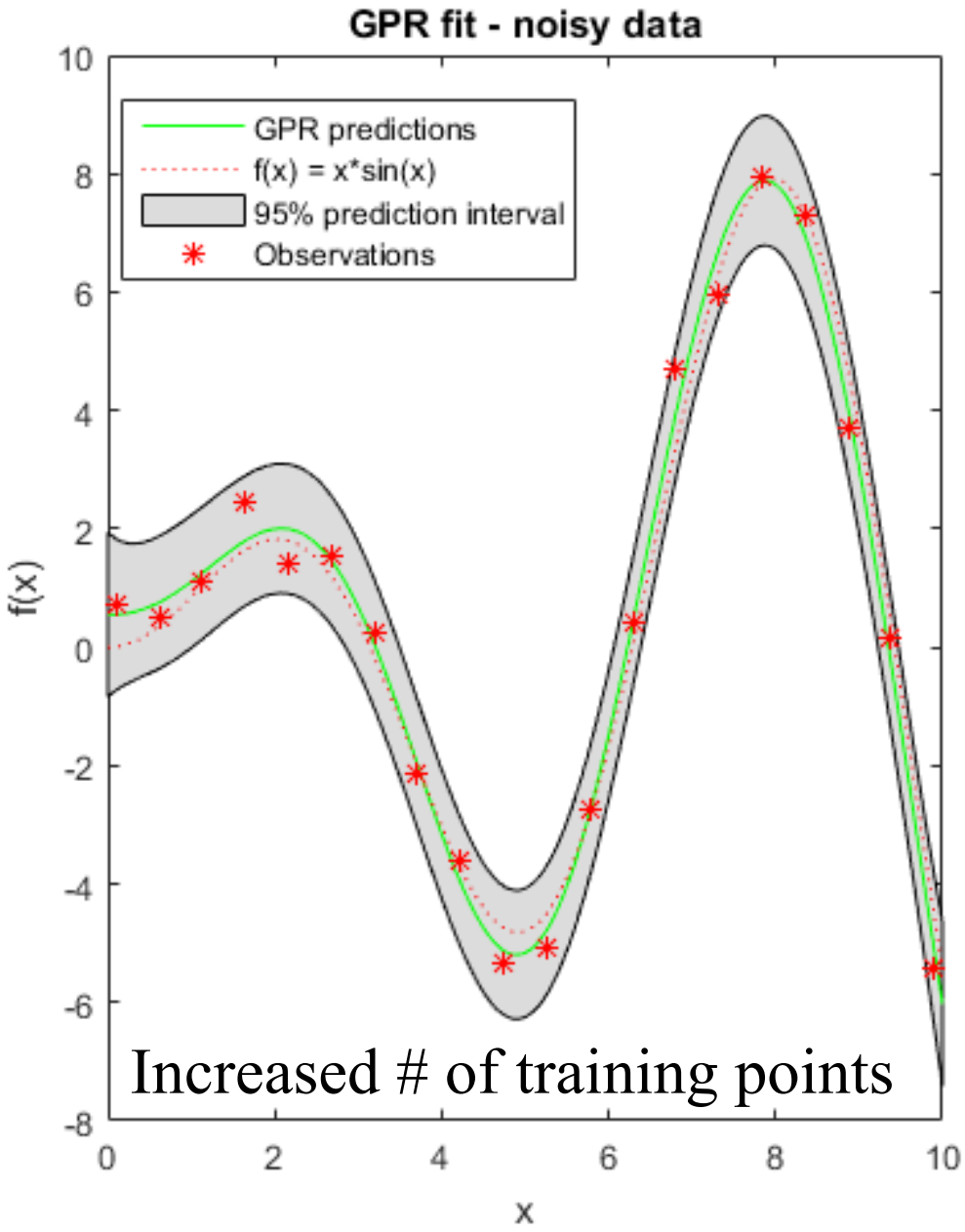
Let $\alpha = \left[K + \sigma^2 I\right]^{-1} Y$

$$\bar{f}_* = \sum_{i=1}^{n} \alpha_i^T k(x_i, x_*)$$

$$\bar{f}_* = \underbrace{\{\left[K + \sigma^2 I\right]^{-1} Y\}^T}_{\mathbb{R}^{\boldsymbol{n}\times \boldsymbol{n}}} \underbrace{k_*}_{\mathbb{R}^{n\times 1}} \in \mathbb{R}$$

GP – Results


GPR fit - data perfectly known
GPR predictions
f(x) = x*sin(x)
95% prediction interval
Observations
f(x)
x


GPR fit - noisy data
GPR predictions
f(x) = x*sin(x)
95% prediction interval
Observations
f(x)
x
Increased # of training points

Supportive Vector Machine

Maximum Margin Classifiers
Overlapping class distributions
Classification
Regression

SVM

$$y(x) = w^T \phi(x) + b$$

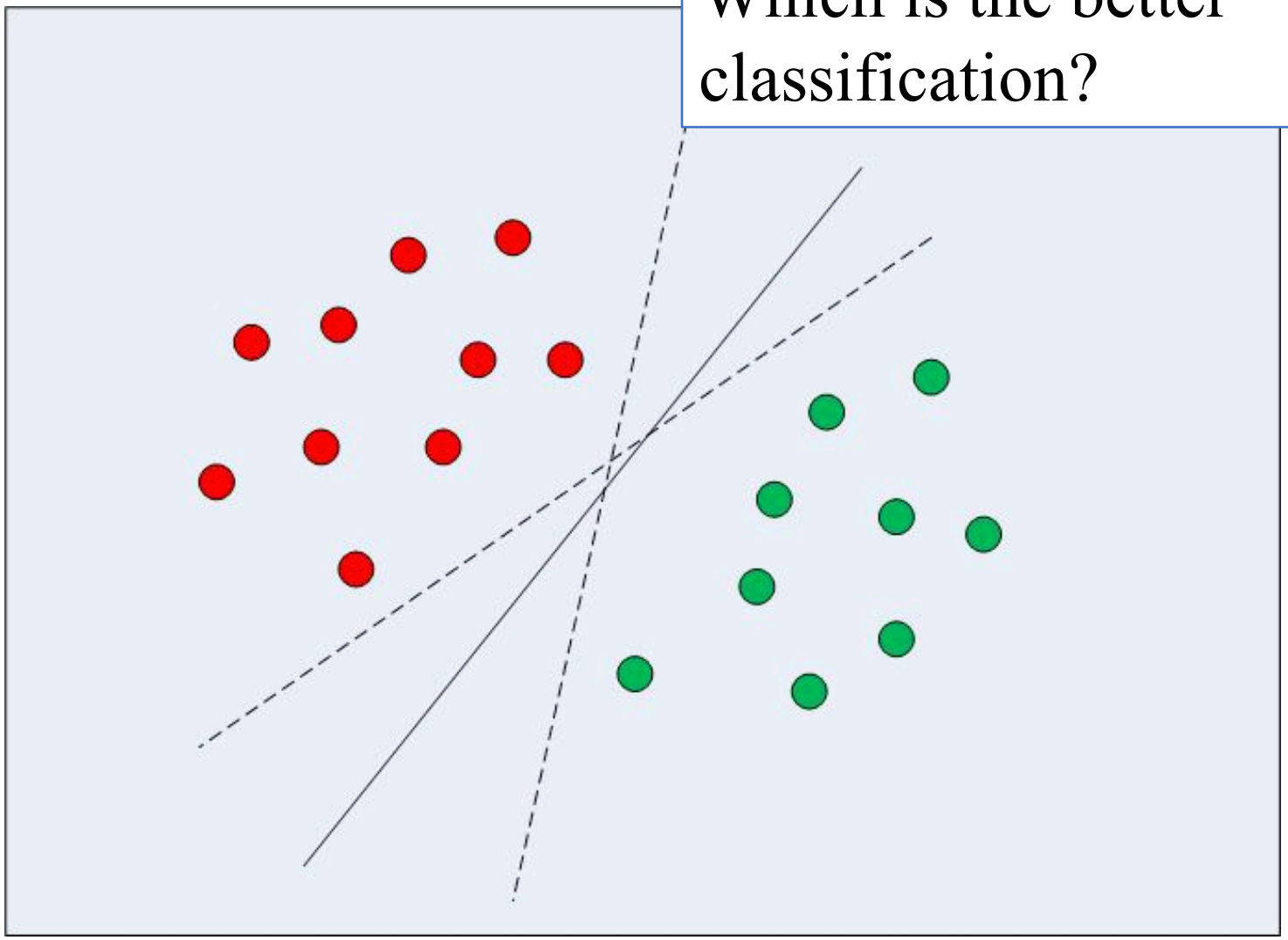
- $\phi(x)$ a fixed feature-space transformation
- b the bias parameter
- training set - linearly separable in feature space
- Data: $\{(x_i, t_i) | n = 1, \ldots, n\}$
- $t_i \in \{-1,1\}$

SVM vs. Perceptron Algorithm

Perceptron Algorithm

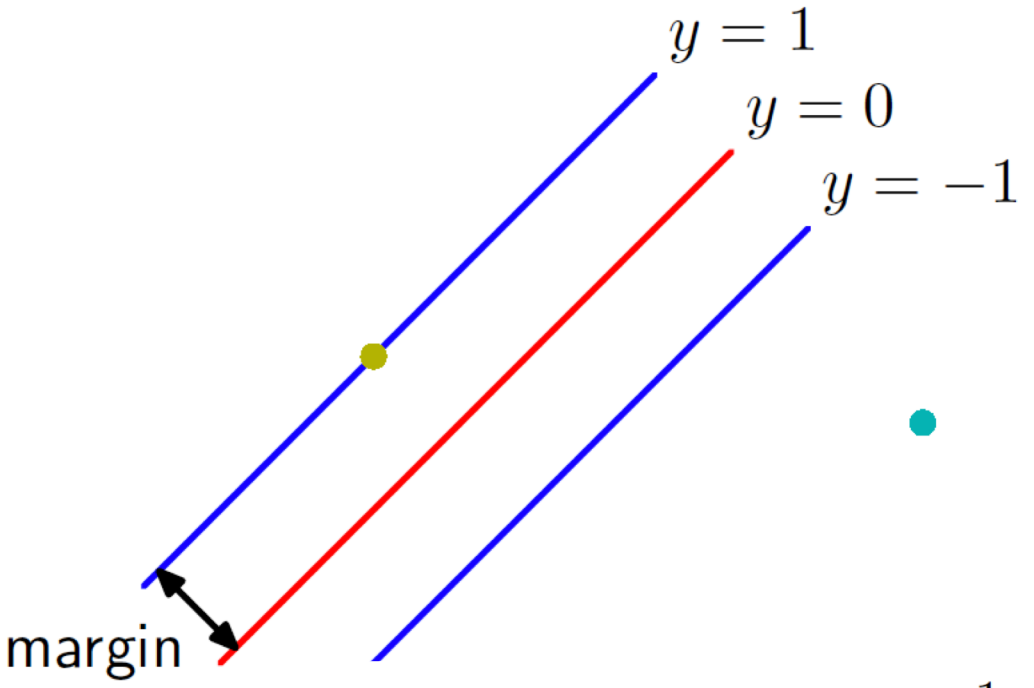
- guarantees to find a solution in a finite number of steps
- initial value for w and b starts from arbitrary chosen value..

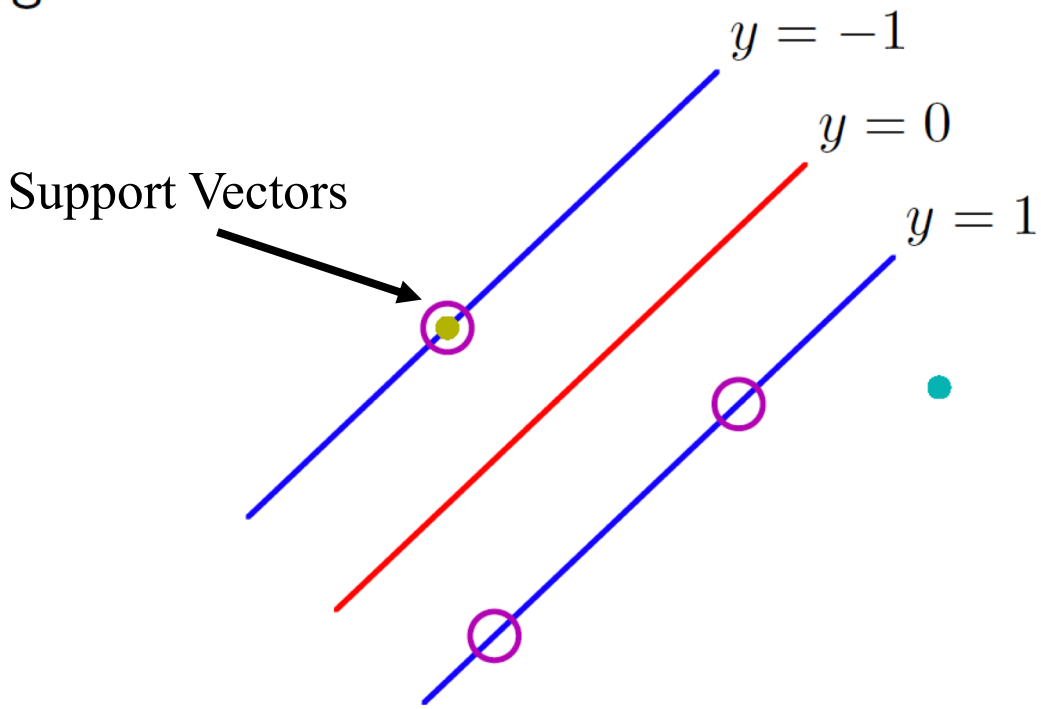
If there are multiple solutions of all of which classify the training data set exactly, then we should try to find the one that give **the smallest generalization error!**



SVM

- An extension of the Perceptron developed by Rosenblatt in 1958.
- Concept of the margin - the smallest distance between the decision boundary an any of the samples.
- Choose the max margin
- Using the optimal boundary, determine the best ***hyperplane*** by minimizing the probability of error relative to the learned density mode.
- hyperplane becomes independent of data points that are not ***support vectors (SV)***.





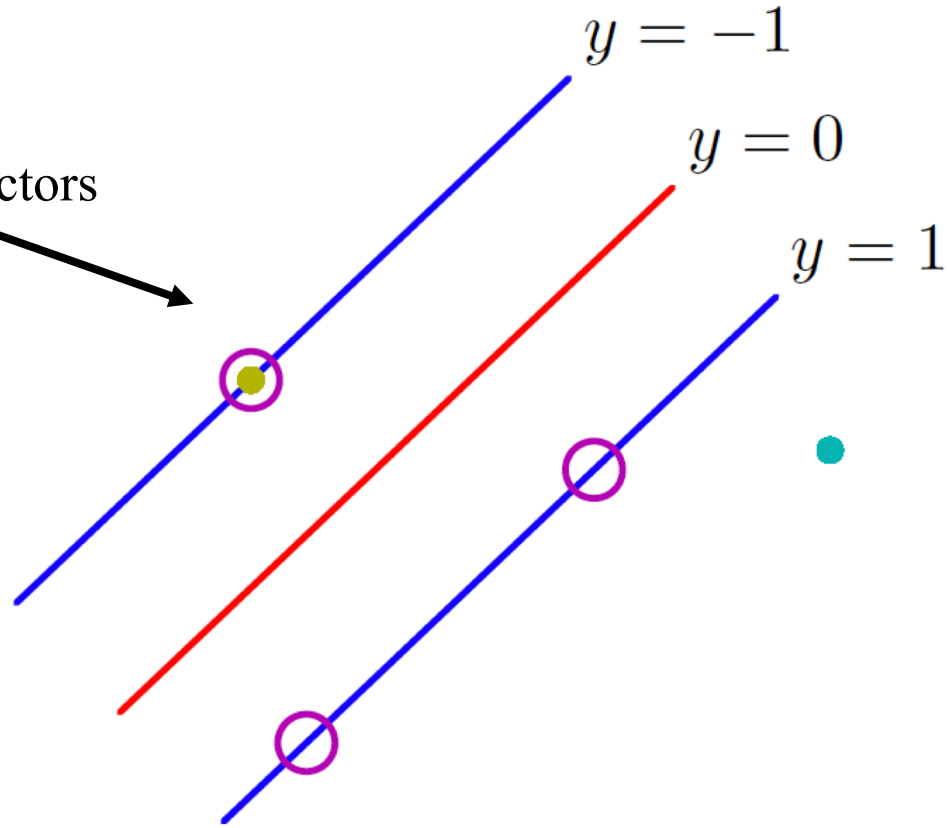
SVM

Hyperplane is $y(x) = 0$

Right classification $t_n y(x_n) > 0$

$$\frac{t_n y(\boldsymbol{x_n})}{||\boldsymbol{w}||} = \frac{t_n[\boldsymbol{w^T}\phi(\boldsymbol{x_n}) + b]}{||\boldsymbol{w}||}$$

Support Vectors



Optimize the parameters w and b to maximize the distance.

Maximum margin solution is

$$\underset{\boldsymbol{w},b}{\operatorname{argmax}} \left\{ \frac{1}{||\boldsymbol{w}||} \min_n \left[t_n \left(\boldsymbol{w^T}\phi(\boldsymbol{x_n}) + b \right) \right] \right\}$$

w does not depend on n. canonical representation of the decision hyperplane.

SVM

Direction solution is quite complex, we convert it into an equivalent problem (rescale)

$$t_n(w^T\phi(x)+b)=1$$

for the point that is closest to the surface. All data points will satisfy the constrains

$$t_n(w^T\phi(x)+b)\geq 1$$

once the margin has been maximized there will be at least two active constraints.

So we maximize $||w||^{-1}$ and this is same as minimizing $||w||^2$

$$\underset{w,b}{\operatorname{argmax}}\frac{1}{2}||w||^2$$

Lagrange multipliers

We introduce Lagrange Multipliers $a_n \geq 0$

$$L(\boldsymbol{w}, b, \boldsymbol{a}) = \frac{1}{2}\left|\left|w\right|\right|^2 - \sum_{n=1}^{N} a_n \left\{t_n\left(\boldsymbol{w}^T \phi(\boldsymbol{x_n}) + b\right) - 1\right\} \quad (1)$$

- a strategy for finding the local maxima and minima of a function subject to equality constraints

Derivative:

$$w = \sum_{n=1}^{N} a_n \, t_n \phi(x_n)$$

$$0 = \sum_{n=1}^{N} a_n t_n$$

Dual Representation

Dual representation of the maximum margin problem

$$L(a) = \sum_{n=1}^{N} a_n - \frac{1}{2}\sum_{m=1}^{N}\sum_{n=1}^{N} a_n\, a_m t_n t_m k(x_n, x_m) \qquad (2)$$

Use the kernel function.

This takes the form of a quadratic programming problem

$$a_n \geq 0$$

$$0 = \sum_{n=1}^{N} a_n t_n$$

Prediction

$$y(\boldsymbol{x}) = \sum_{n=1}^{N} a_n t_n k(\boldsymbol{x}, \boldsymbol{x_n}) + b \tag{3}$$

Satisfies the following conditions:

$$a_n \geq 0$$
$$t_n y(x_n) - 1 \geq 0$$
$$a_n\{t_n y(x_n) - 1\} = 0$$

for point with $a_n = 0$ will not appear in the sum and plays no role making predictions for the new point.

The remaining data points are called support vectors because they satisfy $t_n y(x_n) = 1$ and correspond to points that lie on the maximum margin hyperplanes in feature space.

Prediction

Using Eq (3)

$$t_n\left(\sum_{m\in S} a_m t_m k(x_n, x_m) + b\right) = 1 \tag{4}$$

S denotes the set of indices of the support vectors.

Making a use of $t_n^2 = 1$ averaging over all SV and solve for b

$$b = \frac{1}{N_s}\left(\sum_{n\in S}\left(t_n - \sum_{m\in S} a_m t_m k(x_n, x_m)\right)\right) \tag{5}$$

N_s is the total number of SV.

The maximum margin classifier in terms of the minimization of an error function

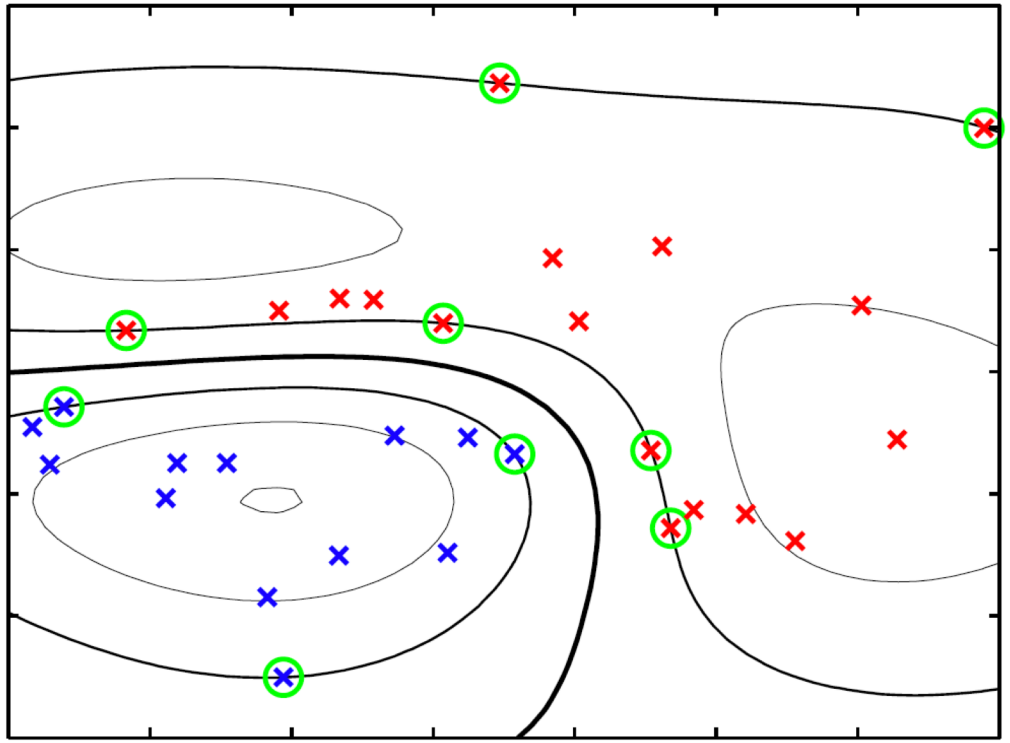
$$\sum_{n=1}^{N} E_\infty(y(x_n)t_n - 1) + \lambda\left|\left|w\right|\right|^2 \tag{6}$$

Prediction

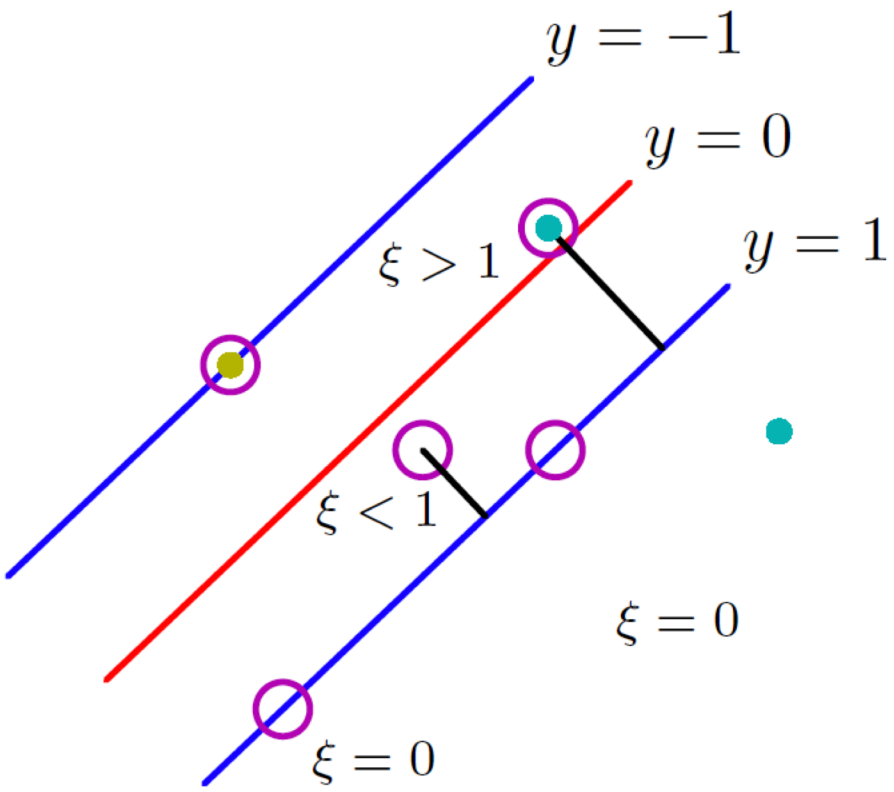
The maximum margin classifier in terms of the minimization of an error function

$$\sum_{n=1}^{N} E_{\infty}(y(x_n)t_n - 1) + \lambda\left|\left|w\right|\right|^2 \qquad (6)$$

where E is a function that is zero if $z \geq 0$ and ∞ otherwise to ensure that the constraints satisfies.



Overlapping Class Distributions



The class-conditional distributions may overlap, in which case exact separation of the training data can lead to poor generalization.

Need to modify SVM

Modify so that data points are allowed to be on the wrong side of the margin boundary, but with penalty that increases with the distance from that boundary.

Overlapping Class Distributions – soft SVM

Make the penalty a linear function of the distance.

slack variable $\xi_n \geq 0$

$\xi = 0$ for data points that are on or inside the correct margin boundary and $\xi = |t_n - y(x_n)|$ for other points. on the decision boundary, $y(x_n) = 0$ and have $\xi = 1$.w/ $\xi > 1$ will be misclassified

$$\boxed{t_n y(x_n) \geq 1 - \xi_n}$$

Goal is to maximize the margin while **softly penalizing points** that lie on the wrong side of the margin boundary

Overlapping Class Distributions – soft SVM

$$C\sum_{n=1}^{N}\xi_n + \frac{1}{2}\left||w|\right|^2 \tag{7}$$

where parameter C>0 controls the trade-off between the slack variable penalty and the margin.

$$L(w,b,\xi,a,\mu) = \frac{1}{2}\left||w|\right|^2 + C\sum_{n=1}^{N}\xi_n + \sum_{n=1}^{N}a_n\{t_n y_n(x_n) - 1 + \xi_n\} - \sum_{n=1}^{N}\mu_n\xi_n \tag{8}$$

Lagrange Multipliers - $\{a_n \geq 0\}$ and $\{\mu_n \geq 0\}$:

$a_n = 0, t_n y_n(x_n) - 1 + \xi_n \geq 0, a_n\{t_n y_n(x_n) - 1 + \xi_n\} = 0$

$\mu_n \geq 0, \xi_n \geq 0, \mu_n\xi_n = 0$

Overlapping Class Distributions – soft SVM

Optimize out w, b, and ξ

$$\frac{\partial L}{\partial w} = 0 \Rightarrow w = \sum_{n=1}^{N} a_n t_n \phi(x_n)$$

$$\frac{\partial L}{\partial b} = 0 \Rightarrow \sum_{n=1}^{N} a_n t_n = 0$$

$$\frac{\partial L}{\partial \xi_N} = 0 \Rightarrow a_n = C - \mu_n$$

$0 \leq a_n \leq C$ Box Constraints

$$L(a) = \sum_{n=1}^{N} a_n - \frac{1}{2} \sum_{m=1}^{N} \sum_{n=1}^{N} a_n \, a_m t_n t_m k(x_n, x_m) \Big) \qquad (9)$$

Overlapping Class Distributions – soft SVM

Solution

$$t_n\left(\sum_{m\in S} a_m t_m k(x_n, x_m) + b\right) = 1 \qquad (10\text{-}1)$$

$$b = \frac{1}{N_{\mathcal{M}}}\left(\sum_{n\in\mathcal{M}}\left(t_n - \sum_{m\in S} a_m t_m k(x_n, x_m)\right)\right) \qquad (10\text{-}2)$$

where M is the set of indices of data points having $0 < a_n < C$.

Overlapping Class Distributions – soft SVM

IF we wish to use the SVM as a module in a larger probabilistic system

Platt (2000) has proposed fitting a logistic sigmoid to the outputs of a trained SVM.

$$p(t = 1|x) = \sigma(Ay(x) + B)$$

A & B are found by minimizing the cross-entropy error function defined by a training set consisting of pairs of values y and t.

- The data used to fit the sigmoid needs to be independent to avoid sever over-fitting.
- Give a poor approximation to the posterior prob.

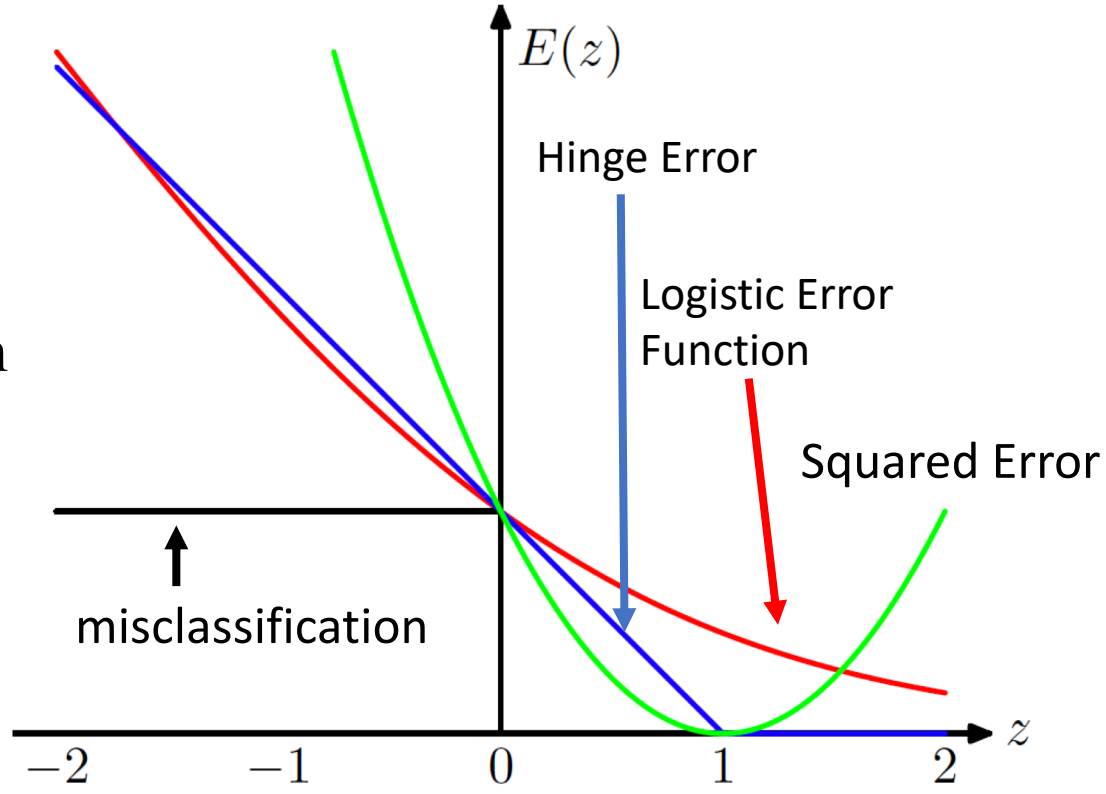
Relation to logistic regression – Hinge Error

Update Eq (7)

$$\sum_{n=1}^{N} E_{SV}(y_n t_n) + \lambda \left|\left|w\right|\right|^2 \qquad (11)$$

where $\lambda = (2C)^{-1}$ and E_{SV} is the **hinge error** function

$$E_{SV}(y_n t_n) = [1 - y_n t_n]_+ \qquad (12)$$

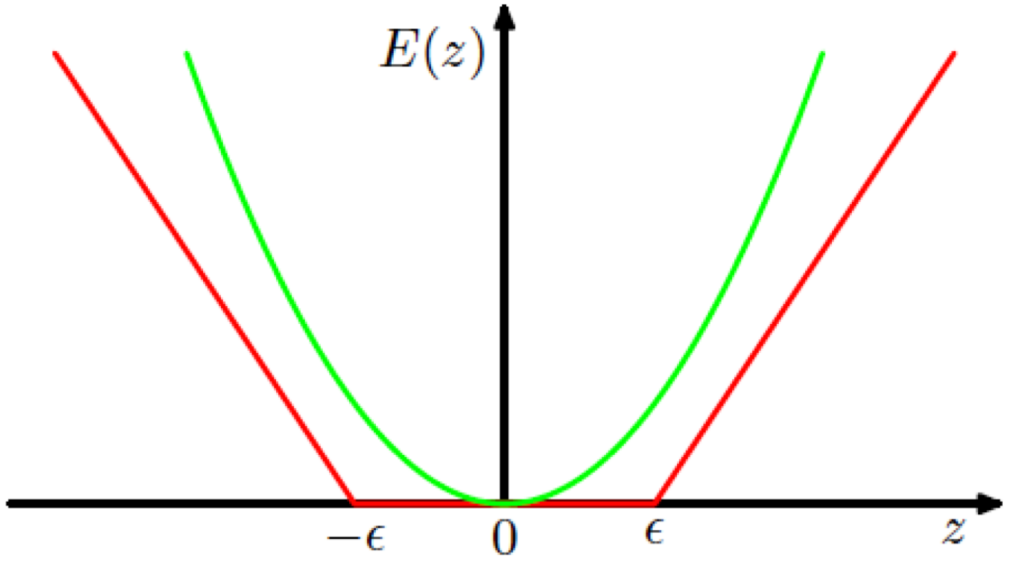


Relation to logistic regression

	SVMs	Logistic Regression
Loss Function	Hinge Loss	Log-loss
High dimensional features	Yes	No
Solution Sparse	Yes	Almost no
Output	Margin	Real Probabilities!

SVM for regression

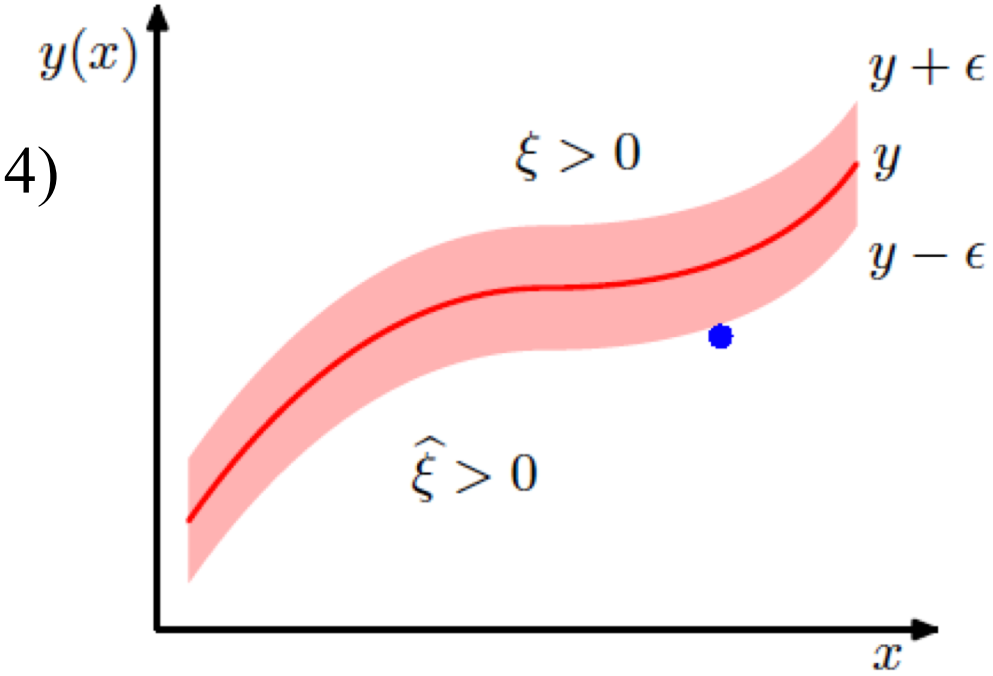
$$\frac{1}{2}\sum_{n=1}^{N}\{y_n - t_n\}^2 + \frac{\lambda}{2}||w||^2 \qquad (13)$$



ϵ-insensitive error function (Vapnik, 1995)

$$E_\epsilon(y(x) - t) = \begin{cases} 0 & if\ y(x) - t < \epsilon \\ |y(x) - t| & \text{Otherwise} \end{cases}$$

$$C\sum_{n=1}^{N} E_\epsilon(y_n(x_n) - t_n) + \frac{\lambda}{2}||w||^2 \qquad (14)$$



$\xi_n > 0$ corresponds to a point $t_n > y(x_n) + \epsilon$ and
$\hat{\xi}_n \geq 0$ for $t_n < y(x_n) - \epsilon$

$$C\sum_{n=1}^{N}\left(\xi_n + \hat{\xi}_n\right) + \frac{1}{2}||w||^2 \qquad (15)$$

SVM for regression - Optimize by slack variables

$$L = C\sum_{n=1}^{N}\left(\xi_n + \hat{\xi}_n\right) + \frac{1}{2}\left||w|\right|^2 - \sum_{n=1}^{N}\left(\mu_n\xi_n + \hat{\mu}_n\hat{\xi}_n\right) - \sum_{n=1}^{N}a_n(\epsilon + \xi_n + y_n - t_n) - \sum_{n=1}^{N}\hat{a}_n\left(\epsilon + \hat{\xi}_n - y_n + t_n\right) \quad (16)$$

The derivative w.r.t. w, b, ξ_n, and $\hat{\xi}_n$

$$\frac{\partial L}{\partial w} = 0 \Rightarrow w = \sum_{n=1}^{N}(a_n - \hat{a}_n)\phi(x_n), \frac{\partial L}{\partial b} = 0 \Rightarrow \sum_{n=1}^{N}(a_n - \hat{a}_n) = 0$$

$$\frac{\partial L}{\partial \xi_n} = 0 \Rightarrow a_n = C - \mu_n, \frac{\partial L}{\partial \hat{\xi}_n} = 0 \Rightarrow \hat{a}_n = C - \hat{\mu}_n$$

SVM for regression - Optimize by slack variables

$$L(a,\hat{a}) = -\frac{1}{2}\sum_{m=1}^{N}\sum_{n=1}^{N}(a_n-\hat{a}_n)(a_m-\hat{a}_m)k(x_n,x_m) - \epsilon\sum_{n=1}^{N}(a_n+\hat{a}_n) + \sum_{n=1}^{N}(a_n-\hat{a}_n)\,t_n \quad (17)$$

$$y(x) = \sum_{n=1}^{N}(a_n-\hat{a}_n)k(x,x_n) + b$$

$$a_n(\epsilon+\xi_n+y_n-t_n) = 0$$
$$\hat{a}_n(\epsilon+\hat{\xi}_n-y_n+t_n) = 0$$
$$(C-a_n)\xi_n = 0$$
$$(C-\hat{a}_n)\hat{\xi}_n = 0$$

$$b = t_n - \epsilon - w^T\phi(x_n) = t_n - \epsilon - \sum_{m=1}^{N}(a_m-\hat{a}_m)k(x_n,x_m) \quad (18)$$

$$L(a,\hat{a}) = -\frac{1}{2}\sum_{m=1}^{N}\sum_{n=1}^{N}(a_n-\hat{a}_n)(a_m-\hat{a}_m)k(x_n,x_m) + \sum_{n=1}^{N}(a_n-\hat{a}_n)\,t_n \quad (19)$$

Outline

Gaussian Distribution

- Saw the beauty of Gaussians – Everything returns to Gaussians!

Gaussian Process

- We can get various possible functions directly with observed data

Supportive Vector Machine

- Overcomes the limit of linear classification/regression problems from spatial perspective.
- Use Lagrange Multipliers

ML Demonstration