Introduction to Kernel Methods

Fabio A. González Ph.D.

Depto. de Ing. de Sistemas e Industrial
Universidad Nacional de Colombia, Bogotá

September 11, 2015

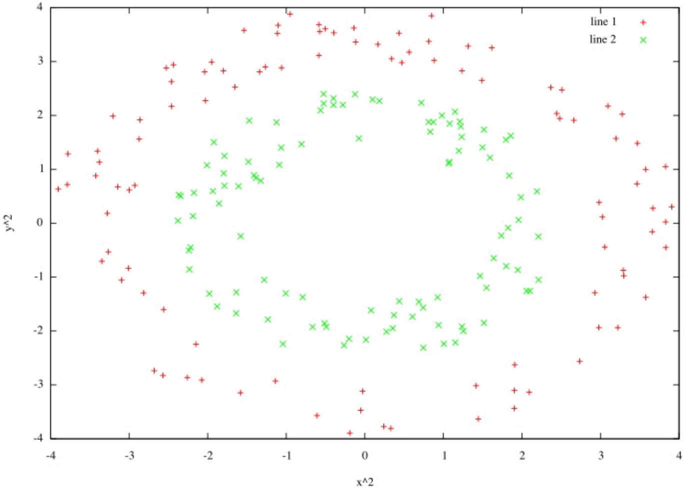
Outline

Outline

Outline

Problem 1

How to separate these two classes using a linear function?



Problem 2

How to do symbolic regression?

$$\Sigma = \{A, C, G, T\}$$

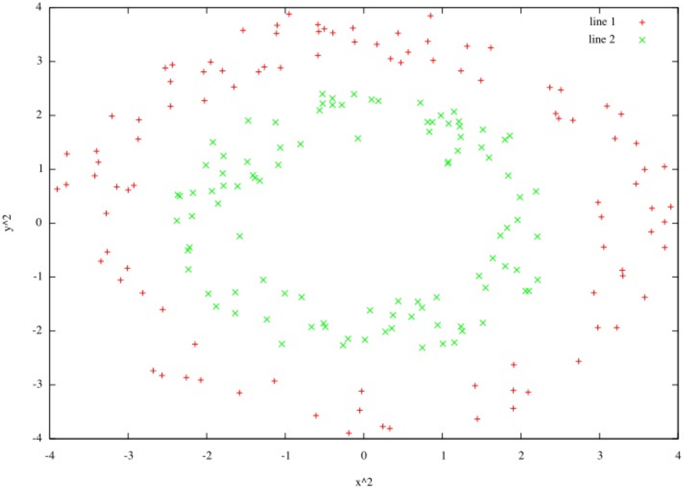
$$\begin{array}{rccc} f: & \Sigma^d & \rightarrow & \mathbb{R} \\ & ACGTA & \mapsto & 10.0 \\ & GTCCA & \mapsto & 11.3 \\ & GGTAC & \mapsto & 1.0 \\ & CCTGA & \mapsto & 4.5 \\ & \vdots & \vdots & \vdots \end{array}$$

Outline

Outline

Problem 1

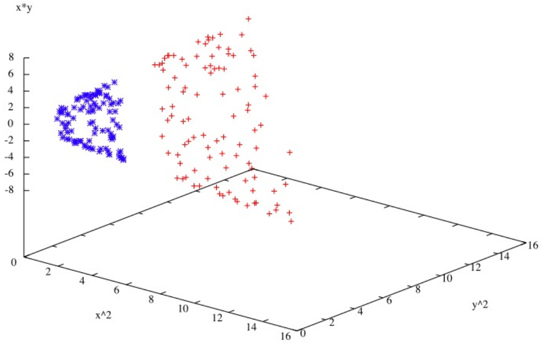
- How to separate these two classes using a linear function?



Solution

- Map to $\mathbb{R}^3$:

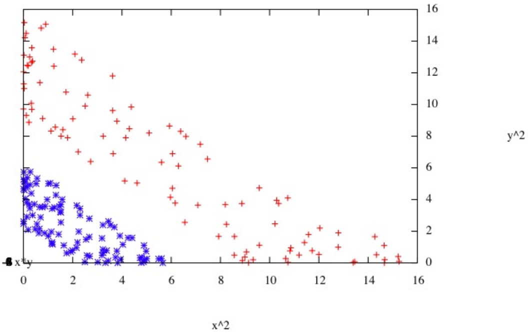
$$\begin{aligned} \phi : \mathbb{R}^2 &\rightarrow \mathbb{R}^3 \\ (x, y) &\mapsto (x^2, y^2, xy) \end{aligned}$$



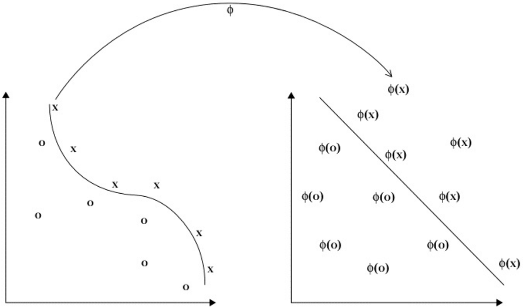
Solution

- Map to $\mathbb{R}^3$:

$$
\begin{aligned}
\phi : \mathbb{R}^2 &\rightarrow \mathbb{R}^3 \\
(x, y) &\mapsto (x^2, y^2, xy)
\end{aligned}
$$



Input space vs. feature space



Outline

Dot product in the feature space

- $$\begin{aligned} \phi : \mathbb{R}^2 &\to \mathbb{R}^3 \\ (x_1, x_2) &\mapsto (x_1^2, x_2^2, \sqrt{2}x_1x_2) \end{aligned}$$
- $$\begin{aligned} \langle \phi(x), \phi(z) \rangle &= \left\langle (x_1^2, x_2^2, \sqrt{2}x_1x_2), (z_1^2, z_2^2, \sqrt{2}z_1z_2) \right\rangle \\ &= x_1^2 z_1^2 + x_2^2 z_2^2 + 2x_1x_2z_1z_2 \\ &= (x_1z_1 + x_2z_2)^2 \\ &= \langle x, z \rangle^2 \end{aligned}$$
- A function $k : X \times X \to \mathbb{R}$ such that $k(x, z) = \langle \phi(x), \phi(z) \rangle$ is called a kernel
- Morale: **you don't need to apply ϕ explicitly to calculate the dot product in the feature space!**

Dot product in the feature space

- $$\begin{aligned} \phi : \mathbb{R}^2 &\to \mathbb{R}^3 \\ (x_1, x_2) &\mapsto (x_1^2, x_2^2, \sqrt{2}x_1 x_2) \end{aligned}$$

- $$\begin{aligned} \langle \phi(x), \phi(z) \rangle &= \left\langle (x_1^2, x_2^2, \sqrt{2}x_1 x_2), (z_1^2, z_2^2, \sqrt{2}z_1 z_2) \right\rangle \\ &= x_1^2 z_1^2 + x_2^2 z_2^2 + 2x_1 x_2 z_1 z_2 \\ &= (x_1 z_1 + x_2 z_2)^2 \\ &= \langle x, z \rangle^2 \end{aligned}$$

- A function $k : X \times X \to \mathbb{R}$ such that $k(x, z) = \langle \phi(x), \phi(z) \rangle$ is called a kernel
- Morale: **you don't need to apply ϕ explicitly to calculate the dot product in the feature space!**

Dot product in the feature space

-

$$\begin{aligned} \phi : \mathbb{R}^2 &\rightarrow \mathbb{R}^3 \\ (x_1, x_2) &\mapsto (x_1^2, x_2^2, \sqrt{2}x_1 x_2) \end{aligned}$$

-

$$\begin{aligned} \langle \phi(x), \phi(z) \rangle &= \left\langle (x_1^2, x_2^2, \sqrt{2}x_1 x_2), (z_1^2, z_2^2, \sqrt{2}z_1 z_2) \right\rangle \\ &= x_1^2 z_1^2 + x_2^2 z_2^2 + 2x_1 x_2 z_1 z_2 \\ &= (x_1 z_1 + x_2 z_2)^2 \\ &= \langle x, z \rangle^2 \end{aligned}$$

- A function $k : X \times X \rightarrow \mathbb{R}$ such that $k(x, z) = \langle \phi(x), \phi(z) \rangle$ is called a kernel
- Morale: **you don't need to apply ϕ explicitly to calculate the dot product in the feature space!**

Dot product in the feature space

- $$\begin{aligned} \phi : \mathbb{R}^2 &\to \mathbb{R}^3 \\ (x_1, x_2) &\mapsto (x_1^2, x_2^2, \sqrt{2}x_1x_2) \end{aligned}$$

- $$\begin{aligned} \langle \phi(x), \phi(z) \rangle &= \left\langle (x_1^2, x_2^2, \sqrt{2}x_1x_2), (z_1^2, z_2^2, \sqrt{2}z_1z_2) \right\rangle \\ &= x_1^2z_1^2 + x_2^2z_2^2 + 2x_1x_2z_1z_2 \\ &= (x_1z_1 + x_2z_2)^2 \\ &= \langle x, z \rangle^2 \end{aligned}$$

- A function $k : X \times X \to \mathbb{R}$ such that $k(x, z) = \langle \phi(x), \phi(z) \rangle$ is called a kernel
- Morale: **you don't need to apply ϕ explicitly to calculate the dot product in the feature space!**

Kernel induced feature space

- The feature space induced by the kernel is not unique: The kernel

$$k(x, z) = \langle x, z \rangle^2$$

also calculates the dot product in the four dimensional feature space:

$$\begin{aligned} \phi : \mathbb{R}^2 &\to \mathbb{R}^4 \\ (x_1, x_2) &\mapsto (x_1^2, x_2^2, x_1 x_2, x_2 x_1) \end{aligned}$$

- The example can be generalised to $\mathbb{R}^n$

Kernel induced feature space

- The feature space induced by the kernel is not unique: The kernel

$$k(x, z) = \langle x, z \rangle^2$$

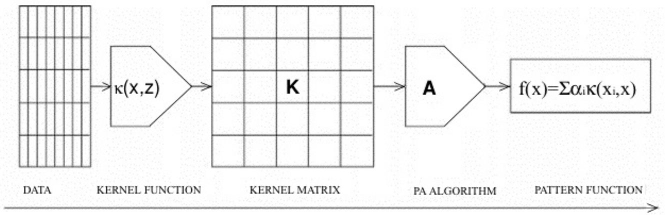
also calculates the dot product in the four dimensional feature space:

$$\begin{aligned} \phi : \mathbb{R}^2 &\rightarrow \mathbb{R}^4 \\ (x_1, x_2) &\mapsto (x_1^2, x_2^2, x_1 x_2, x_2 x_1) \end{aligned}$$

- The example can be generalised to $\mathbb{R}^n$

Outline

The Process


κ(x,z)
K
A
f(x)=Σαᵢκ(xᵢ,x)
DATA
KERNEL FUNCTION
KERNEL MATRIX
PA ALGORITHM
PATTERN FUNCTION

The Approach

- Data items are embedded into a vector space called the feature space
- Linear relations are sought among the images of the data items in the feature space
- The pattern analysis algorithm are based only on the pairwise dot products, they do not need the actual coordinates of the embedded points
- The pairwise dot products in the feature space could be efficiently calculated using a kernel function

The Approach

- Data items are embedded into a vector space called the feature space
- Linear relations are sought among the images of the data items in the feature space
- The pattern analysis algorithm are based only on the pairwise dot products, they do not need the actual coordinates of the embedded points
- The pairwise dot products in the feature space could be efficiently calculated using a kernel function

The Approach

- Data items are embedded into a vector space called the feature space
- Linear relations are sought among the images of the data items in the feature space
- The pattern analysis algorithm are based only on the pairwise dot products, they do not need the actual coordinates of the embedded points
- The pairwise dot products in the feature space could be efficiently calculated using a kernel function

The Approach

- Data items are embedded into a vector space called the feature space
- Linear relations are sought among the images of the data items in the feature space
- The pattern analysis algorithm are based only on the pairwise dot products, they do not need the actual coordinates of the embedded points
- The pairwise dot products in the feature space could be efficiently calculated using a kernel function

Outline

Outline

Problem definition

- Given a training set $S = \{(\mathrm{x}_1, y_1), \ldots, (\mathrm{x}_l, y_l)\}$ of points $\mathrm{x}_i \in \mathbb{R}^n$ with corresponding labels $y_i \in \mathbb{R}$ the problem is to find a real-valued linear function that best interpolates the training set:

$$g(\mathrm{x}) = \langle \mathrm{w}, \mathrm{x} \rangle = \mathrm{w}'\mathrm{x} = \sum_{i=1}^{n} w_i x_i$$

- If the data points were generated by a function like $g(\mathrm{x})$, it is possible to find the parameters w by solving

$$\mathrm{X}\mathrm{w} = \mathrm{y}$$

where

$$\mathrm{X} = \begin{bmatrix} \mathrm{x}'_1 \\ \vdots \\ \mathrm{x}'_l \end{bmatrix}$$

Problem definition

- Given a training set $S = \{(\mathrm{x}_1, y_1), \ldots, (\mathrm{x}_l, y_l)\}$ of points $\mathrm{x}_i \in \mathbb{R}^n$ with corresponding labels $y_i \in \mathbb{R}$ the problem is to find a real-valued linear function that best interpolates the training set:

$$g(\mathrm{x}) = \langle \mathrm{w}, \mathrm{x} \rangle = \mathrm{w}'\mathrm{x} = \sum_{i=1}^{n} w_i x_i$$

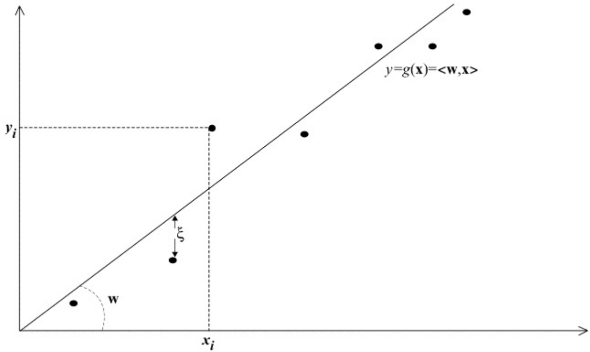
- If the data points were generated by a function like $g(\mathrm{x})$, it is possible to find the parameters w by solving

$$\mathrm{X}\mathrm{w} = \mathrm{y}$$

where

$$\mathrm{X} = \begin{bmatrix} \mathrm{x}'_1 \\ \vdots \\ \mathrm{x}'_l \end{bmatrix}$$

Graphical representation



Loss function

- Minimize

$$\begin{aligned}\mathcal{L}(g,S) &= \mathcal{L}(\mathrm{w},S) = \sum_{i=1}^{l}(y_i - g(x_i))^2 = \sum_{i=1}^{l}\xi_i^2 \\ &= \sum_{i=1}^{l}\mathcal{L}(g,(\mathrm{x}_i,y_i))\end{aligned}$$

- This could be written as

$$\mathcal{L}(\mathrm{w},S) = \|\xi\|^2 = (\mathrm{y} - \mathrm{X}\mathrm{w})'(\mathrm{y} - \mathrm{X}\mathrm{w})$$

- Optimization problem:

$$\min_{\mathrm{w}} \mathcal{L}(\mathrm{w},S) = \min_{\mathrm{w}} (\mathrm{y} - \mathrm{X}\mathrm{w})'(\mathrm{y} - \mathrm{X}\mathrm{w})$$

Loss function

- Minimize

$$\begin{aligned}\mathcal{L}(g,S) &= \mathcal{L}(\mathrm{w},S) = \sum_{i=1}^{l}(y_i - g(x_i))^2 = \sum_{i=1}^{l}\xi_i^2 \\ &= \sum_{i=1}^{l}\mathcal{L}(g,(\mathrm{x}_i,y_i))\end{aligned}$$

- This could be written as

$$\mathcal{L}(\mathrm{w},S) = \|\xi\|^2 = (\mathrm{y} - \mathrm{X}\mathrm{w})'(\mathrm{y} - \mathrm{X}\mathrm{w})$$

- Optimization problem:

$$\min_{\mathrm{w}} \mathcal{L}(\mathrm{w},S) = \min_{\mathrm{w}} (\mathrm{y} - \mathrm{X}\mathrm{w})'(\mathrm{y} - \mathrm{X}\mathrm{w})$$

Loss function

- Minimize

$$\begin{aligned}\mathcal{L}(g,S) &= \mathcal{L}(\mathrm{w},S) = \sum_{i=1}^{l}(y_i - g(x_i))^2 = \sum_{i=1}^{l}\xi_i^2 \\ &= \sum_{i=1}^{l}\mathcal{L}(g,(\mathrm{x}_i,y_i))\end{aligned}$$

- This could be written as

$$\mathcal{L}(\mathrm{w},S) = \|\xi\|^2 = (\mathrm{y} - \mathrm{Xw})'(\mathrm{y} - \mathrm{Xw})$$

- Optimization problem:

$$\min_{\mathrm{w}} \mathcal{L}(\mathrm{w},S) = \min_{\mathrm{w}}(\mathrm{y} - \mathrm{Xw})'(\mathrm{y} - \mathrm{Xw})$$

Solution

$$\frac{\partial \mathcal{L}(\mathrm{w}, S)}{\partial \mathrm{w}} = -2\mathrm{X}'\mathrm{y} + 2\mathrm{X}'\mathrm{X}\mathrm{w} = 0,$$

therefore

$$\mathrm{X}'\mathrm{X}\mathrm{w} = \mathrm{X}'\mathrm{y},$$

and

$$\mathrm{w} = (\mathrm{X}'\mathrm{X})^{-1}\mathrm{X}'\mathrm{y}$$

Outline

Dual representation of the problem

- $\mathrm{w} = (\mathrm{X}'\mathrm{X})^{-1}\mathrm{X}'\mathrm{y} = \mathrm{X}'\mathrm{X}(\mathrm{X}'\mathrm{X})^{-2}\mathrm{X}'\mathrm{y} = \mathrm{X}'\alpha$
- So, w is a linear combination of the training samples, $\mathrm{w} = \sum_{i=1}^{l} \alpha_i \mathrm{x}_i$.

Dual representation of the problem

- $\mathrm{w} = (\mathrm{X}'\mathrm{X})^{-1}\mathrm{X}'\mathrm{y} = \mathrm{X}'\mathrm{X}(\mathrm{X}'\mathrm{X})^{-2}\mathrm{X}'\mathrm{y} = \mathrm{X}'\alpha$
- So, w is a linear combination of the training samples, $\mathrm{w} = \sum_{i=1}^{l} \alpha_i \mathrm{x}_i$.

Solution

- From the solution of the primal problem:

$$X'Xw = X'y,$$

- then

$$XX'Xw = XX'y,$$

- using the dual representation

$$XX'XX'\alpha = XX'y,$$

- then

$$\alpha = (XX')^{-1}y,$$

- and

$$g(x) = w'x = \alpha'Xx.$$

- Note: XX' may be close to singular, or singular according to machine precision.

Solution

- From the solution of the primal problem:

$$X'Xw = X'y,$$

- then

$$XX'Xw = XX'y,$$

- using the dual representation

$$XX'XX'\alpha = XX'y,$$

- then

$$\alpha = (XX')^{-1}y,$$

- and

$$g(x) = w'x = \alpha'Xx.$$

- Note: XX' may be close to singular, or singular according to machine precision.

Solution

- From the solution of the primal problem:

$$X'Xw = X'y,$$

- then

$$XX'Xw = XX'y,$$

- using the dual representation

$$XX'XX'\alpha = XX'y,$$

- then

$$\alpha = (XX')^{-1}y,$$

- and

$$g(x) = w'x = \alpha'Xx.$$

- Note: XX' may be close to singular, or singular according to machine precision.

Solution

- From the solution of the primal problem:

$$X'Xw = X'y,$$

- then

$$XX'Xw = XX'y,$$

- using the dual representation

$$XX'XX'\alpha = XX'y,$$

- then

$$\alpha = (XX')^{-1}y,$$

- and

$$g(x) = w'x = \alpha'Xx.$$

- Note: XX' may be close to singular, or singular according to machine precision.

Solution

- From the solution of the primal problem:

$$\mathrm{X}'\mathrm{X}\mathrm{w} = \mathrm{X}'\mathrm{y},$$

- then

$$\mathrm{X}\mathrm{X}'\mathrm{X}\mathrm{w} = \mathrm{X}\mathrm{X}'\mathrm{y},$$

- using the dual representation

$$\mathrm{X}\mathrm{X}'\mathrm{X}\mathrm{X}'\alpha = \mathrm{X}\mathrm{X}'\mathrm{y},$$

- then

$$\alpha = (\mathrm{X}\mathrm{X}')^{-1}\mathrm{y},$$

- and

$$g(\mathrm{x}) = \mathrm{w}'\mathrm{x} = \alpha'\mathrm{X}\mathrm{x}.$$

- Note: $\mathrm{X}\mathrm{X}'$ may be close to singular, or singular according to machine precision.

Solution

- From the solution of the primal problem:

$$X'Xw = X'y,$$

- then

$$XX'Xw = XX'y,$$

- using the dual representation

$$XX'XX'\alpha = XX'y,$$

- then

$$\alpha = (XX')^{-1}y,$$

- and

$$g(x) = w'x = \alpha'Xx.$$

- Note: XX' may be close to singular, or singular according to machine precision.

Ridge regression

- If XX' is singular, the pseudo-inverse could be used: to find the w that satisfies $\mathrm{X}'\mathrm{Xw} = \mathrm{X}'\mathrm{y}$ with minimal norm.
- Optimisation problem:

$$\min_{\mathrm{w}} \mathcal{L}_\lambda(\mathrm{w}, S) = \min_{\mathrm{w}} \lambda \|\mathrm{w}\|^2 + \sum_{i=1}^{l} (y_i - g(x_i))^2,$$

where λ defines the trade-off between norm and loss. This controls the complexity of the model (the process is called *regularization*).

Ridge regression

- If $\mathrm{X}\mathrm{X}'$ is singular, the pseudo-inverse could be used: to find the w that satisfies $\mathrm{X}'\mathrm{X}\mathrm{w} = \mathrm{X}'\mathrm{y}$ with minimal norm.
- Optimisation problem:

$$\min_{\mathrm{w}} \mathcal{L}_{\lambda}(\mathrm{w}, S) = \min_{\mathrm{w}} \lambda \|\mathrm{w}\|^2 + \sum_{i=1}^{l} (y_i - g(x_i))^2,$$

where λ defines the trade-off between norm and loss. This controls the complexity of the model (the process is called *regularization*).

Solution

- Taking the derivative and making it equal to zero:

$$X'Xw + \lambda w = (X'X + \lambda I_n)w = X'y,$$

where I_n is an identity matrix of $n \times n$ dimension,

- then,

$$w = (X'X + \lambda I_n)^{-1} X'y.$$

- In terms of α:

$$w = \lambda^{-1} X'(y - Xw) = X'\alpha,$$

- then

$$\alpha = \lambda^{-1}(y - Xw) = (XX' + \lambda I_l)^{-1} y.$$

Solution

- Taking the derivative and making it equal to zero:

$$\mathrm{X'Xw} + \lambda \mathrm{w} = (\mathrm{X'X} + \lambda \mathrm{I_n})\mathrm{w} = \mathrm{X'y},$$

  where I_n is an identity matrix of $n \times n$ dimension,

- then,

$$\mathrm{w} = (\mathrm{X'X} + \lambda \mathrm{I_n})^{-1}\mathrm{X'y}.$$

- In terms of α:

$$\mathrm{w} = \lambda^{-1}\mathrm{X'}(\mathrm{y} - \mathrm{Xw}) = \mathrm{X'}\alpha,$$

- then

$$\alpha = \lambda^{-1}(\mathrm{y} - \mathrm{Xw}) = (\mathrm{XX'} + \lambda \mathrm{I}_l)^{-1}\mathrm{y}.$$

Solution

- Taking the derivative and making it equal to zero:

$$\mathrm{X}'\mathrm{X}\mathrm{w} + \lambda\mathrm{w} = (\mathrm{X}'\mathrm{X} + \lambda\mathrm{I}_\mathrm{n})\mathrm{w} = \mathrm{X}'\mathrm{y},$$

  where I_n is an identity matrix of $n \times n$ dimension,
- then,

$$\mathrm{w} = (\mathrm{X}'\mathrm{X} + \lambda\mathrm{I}_\mathrm{n})^{-1}\mathrm{X}'\mathrm{y}.$$

- In terms of α:

$$\mathrm{w} = \lambda^{-1}\mathrm{X}'(\mathrm{y} - \mathrm{X}\mathrm{w}) = \mathrm{X}'\alpha,$$

- then

$$\alpha = \lambda^{-1}(\mathrm{y} - \mathrm{X}\mathrm{w}) = (\mathrm{X}\mathrm{X}' + \lambda\mathrm{I}_l)^{-1}\mathrm{y}.$$

Solution

- Taking the derivative and making it equal to zero:

$$\mathrm{X}'\mathrm{X}\mathrm{w} + \lambda\mathrm{w} = (\mathrm{X}'\mathrm{X} + \lambda\mathrm{I_n})\mathrm{w} = \mathrm{X}'\mathrm{y},$$

  where I_n is an identity matrix of $n \times n$ dimension,

- then,

$$\mathrm{w} = (\mathrm{X}'\mathrm{X} + \lambda\mathrm{I_n})^{-1}\mathrm{X}'\mathrm{y}.$$

- In terms of α:

$$\mathrm{w} = \lambda^{-1}\mathrm{X}'(\mathrm{y} - \mathrm{X}\mathrm{w}) = \mathrm{X}'\alpha,$$

- then

$$\alpha = \lambda^{-1}(\mathrm{y} - \mathrm{X}\mathrm{w}) = (\mathrm{X}\mathrm{X}' + \lambda\mathrm{I}_l)^{-1}\mathrm{y}.$$

Prediction function

$$g(\mathrm{x}) = \langle \mathrm{w}, \mathrm{x} \rangle \quad = \quad \left\langle \sum_{i=1}^{l} \alpha_i \mathrm{x_i}, \mathrm{x} \right\rangle = \sum_{i=1}^{l} \alpha_i \langle \mathrm{x_i}, \mathrm{x} \rangle$$

Ridge regression as a kernel method

- The Gram matrix $G = XX'$ is the matrix of dot products

$$G = XX' = \begin{bmatrix} x'_1 \\ \vdots \\ x'_l \end{bmatrix} [x_1 \cdots x_l] = \begin{bmatrix} \langle x_1, x_1 \rangle & \langle x_1, x_l \rangle \\ \langle x_l, x_1 \rangle & \langle x_l, x_l \rangle \end{bmatrix}$$

- G may be replaced by a general kernel matrix, K, with $k_{ij} = k(x_i, x_j) = < \phi(x_i), \phi(x_j) >$
- The α's are calculated as:

$$\alpha = (K + \lambda I_l)^{-1} y$$

- The predicted function is approximated as:

$$g(x) = \sum_{i=1}^{l} \alpha_i k(x, x_i) = y'(K + \lambda I_l)^{-1} \begin{bmatrix} k(x, x_1) \\ \vdots \\ k(x, x_l) \end{bmatrix}$$

Ridge regression as a kernel method

- The Gram matrix $\mathrm{G} = \mathrm{X}\mathrm{X}'$ is the matrix of dot products

$$\mathrm{G} = \mathrm{X}\mathrm{X}' = \begin{bmatrix} \mathrm{x}'_1 \\ \vdots \\ \mathrm{x}'_l \end{bmatrix} [\mathrm{x}_1 \cdots \mathrm{x}_l] = \begin{bmatrix} \langle \mathrm{x}_1, \mathrm{x}_1 \rangle & \langle \mathrm{x}_1, \mathrm{x}_l \rangle \\ \langle \mathrm{x}_l, \mathrm{x}_1 \rangle & \langle \mathrm{x}_l, \mathrm{x}_l \rangle \end{bmatrix}$$

- G may be replaced by a general kernel matrix, K, with $\mathrm{k_{ij}} = k(\mathrm{x_i}, \mathrm{x_j}) = < \phi(\mathrm{x_i}), \phi(\mathrm{x_j}) >$
- The α's are calculated as:

$$\alpha = (\mathrm{K} + \lambda \mathrm{I}_l)^{-1} \mathrm{y}$$

- The predicted function is approximated as:

$$g(\mathrm{x}) = \sum_{i=1}^{l} \alpha_i k(\mathrm{x}, \mathrm{x_i}) = y'(\mathrm{K} + \lambda \mathrm{I}_l)^{-1} \begin{bmatrix} k(\mathrm{x}, \mathrm{x}_1) \\ \vdots \\ k(\mathrm{x}, \mathrm{x_l}) \end{bmatrix}$$

Ridge regression as a kernel method

- The Gram matrix $\mathrm{G} = \mathrm{XX}'$ is the matrix of dot products

$$\mathrm{G} = \mathrm{XX}' = \begin{bmatrix} \mathrm{x}'_1 \\ \vdots \\ \mathrm{x}'_l \end{bmatrix} [\mathrm{x}_1 \cdots \mathrm{x}_l] = \begin{bmatrix} \langle \mathrm{x}_1, \mathrm{x}_1 \rangle & \langle \mathrm{x}_1, \mathrm{x}_l \rangle \\ \langle \mathrm{x}_l, \mathrm{x}_1 \rangle & \langle \mathrm{x}_l, \mathrm{x}_l \rangle \end{bmatrix}$$

- G may be replaced by a general kernel matrix, K, with $\mathrm{k_{ij}} = k(\mathrm{x_i}, \mathrm{x_j}) = < \phi(\mathrm{x_i}), \phi(\mathrm{x_j}) >$
- The α's are calculated as:

$$\alpha = (\mathrm{K} + \lambda \mathrm{I}_l)^{-1} \mathrm{y}$$

- The predicted function is approximated as:

$$g(\mathrm{x}) = \sum_{i=1}^{l} \alpha_i k(\mathrm{x}, \mathrm{x_i}) = y'(\mathrm{K} + \lambda \mathrm{I}_l)^{-1} \begin{bmatrix} k(\mathrm{x}, \mathrm{x}_1) \\ \vdots \\ k(\mathrm{x}, \mathrm{x_l}) \end{bmatrix}$$

Ridge regression as a kernel method

- The Gram matrix $\mathrm{G} = \mathrm{XX}'$ is the matrix of dot products

$$\mathrm{G} = \mathrm{XX}' = \begin{bmatrix} \mathrm{x}'_1 \\ \vdots \\ \mathrm{x}'_l \end{bmatrix} [\mathrm{x}_1 \cdots \mathrm{x}_l] = \begin{bmatrix} \langle \mathrm{x}_1, \mathrm{x}_1 \rangle & \langle \mathrm{x}_1, \mathrm{x}_l \rangle \\ \langle \mathrm{x}_l, \mathrm{x}_1 \rangle & \langle \mathrm{x}_l, \mathrm{x}_l \rangle \end{bmatrix}$$

- G may be replaced by a general kernel matrix, K, with $\mathrm{k_{ij}} = k(\mathrm{x_i}, \mathrm{x_j}) = < \phi(\mathrm{x_i}), \phi(\mathrm{x_j}) >$
- The α's are calculated as:

$$\alpha = (\mathrm{K} + \lambda \mathrm{I}_l)^{-1} \mathrm{y}$$

- The predicted function is approximated as:

$$g(\mathrm{x}) = \sum_{i=1}^{l} \alpha_i k(\mathrm{x}, \mathrm{x_i}) = y'(\mathrm{K} + \lambda \mathrm{I}_l)^{-1} \begin{bmatrix} k(\mathrm{x}, \mathrm{x}_1) \\ \vdots \\ k(\mathrm{x}, \mathrm{x_l}) \end{bmatrix}$$

Outline

Outline

Characterisation

Theorem

(Mercer's Theorem)
A function

$$k : X \times X \to \mathbb{R},$$

which is either continuous or has a countable domain, can be decomposed

$$k(\mathrm{x}, \mathrm{z}) = \langle \phi(\mathrm{x}), \phi(\mathrm{z}) \rangle$$

into a feature map ϕ *into a Hilbert space* F *applied to both its arguments followed by the evaluation of the inner product in* F *if and only if it satisfies the finitely positive semi-definite property.*

Some kernel functions

Assume k_1 and k_2 kernels:

- $k(\mathrm{x}, \mathrm{z}) = p(k_1(\mathrm{x}, \mathrm{z}))$. p a polynomial with positive coefficients.
- $k(\mathrm{x}, \mathrm{z}) = \exp(k_1(\mathrm{x}, \mathrm{z}))$.
- $k(\mathrm{x}, \mathrm{z}) = \exp(-\left\|\mathrm{x} - \mathrm{z}\right\|^2 / (2\sigma^2))$. Gaussian kernel.
- $k(\mathrm{x}, \mathrm{z}) = k_1(\mathrm{x}, \mathrm{z}) k_2(\mathrm{x}, \mathrm{z})$

Some kernel functions

Assume k_1 and k_2 kernels:

- $k(\mathrm{x}, \mathrm{z}) = p(k_1(\mathrm{x}, \mathrm{z}))$. p a polynomial with positive coefficients.
- $k(\mathrm{x}, \mathrm{z}) = \exp(k_1(\mathrm{x}, \mathrm{z}))$.
- $k(\mathrm{x}, \mathrm{z}) = \exp(- \|\mathrm{x} - \mathrm{z}\|^2 / (2\sigma^2))$. Gaussian kernel.
- $k(\mathrm{x}, \mathrm{z}) = k_1(\mathrm{x}, \mathrm{z}) k_2(\mathrm{x}, \mathrm{z})$

Some kernel functions

Assume k_1 and k_2 kernels:

- $k(\mathrm{x}, \mathrm{z}) = p(k_1(\mathrm{x}, \mathrm{z}))$. p a polynomial with positive coefficients.
- $k(\mathrm{x}, \mathrm{z}) = \exp(k_1(\mathrm{x}, \mathrm{z}))$.
- $k(\mathrm{x}, \mathrm{z}) = \exp(-\|\mathrm{x} - \mathrm{z}\|^2 / (2\sigma^2))$. Gaussian kernel.
- $k(\mathrm{x}, \mathrm{z}) = k_1(\mathrm{x}, \mathrm{z}) k_2(\mathrm{x}, \mathrm{z})$

Some kernel functions

Assume k_1 and k_2 kernels:

- $k(\mathrm{x}, \mathrm{z}) = p(k_1(\mathrm{x}, \mathrm{z}))$. p a polynomial with positive coefficients.
- $k(\mathrm{x}, \mathrm{z}) = \exp(k_1(\mathrm{x}, \mathrm{z}))$.
- $k(\mathrm{x}, \mathrm{z}) = \exp(-\|\mathrm{x} - \mathrm{z}\|^2 / (2\sigma^2))$. Gaussian kernel.
- $k(\mathrm{x}, \mathrm{z}) = k_1(\mathrm{x}, \mathrm{z}) k_2(\mathrm{x}, \mathrm{z})$

Embeddings corresponding to kernels

- It is possible to calculate the feature space induced by a kernel (Mercer's Theorem)
- This can be done in a constructive way
- The feature space can even be of infinite dimension.

Embeddings corresponding to kernels

- It is possible to calculate the feature space induced by a kernel (Mercer's Theorem)
- This can be done in a constructive way
- The feature space can even be of infinite dimension.

Embeddings corresponding to kernels

- It is possible to calculate the feature space induced by a kernel (Mercer's Theorem)
- This can be done in a constructive way
- The feature space can even be of infinite dimension.

Outline

How to visualize?

- Choose a point in input space p_0
- Calculate the distance from another point x to p_0 in the feature space:

$$\begin{aligned}\|\phi(p_0)-\phi(x)\|_F^2 &= \langle\phi(p_0)-\phi(x),\phi(p_0)-\phi(x)\rangle_F \\ &= \langle\phi(p_0),\phi(p_0)\rangle_F+\langle\phi(x),\phi(x)\rangle_F \\ &\quad -2\langle\phi(p_0),\phi(x)\rangle_F \\ &= k(p_0,p_0)+k(x,x)-2k(p_0,x)\end{aligned}$$

- Plot $f(x)=\|\phi(p_0)-\phi(x)\|_F^2$

How to visualize?

- Choose a point in input space p_0
- Calculate the distance from another point x to p_0 in the feature space:

$$
\begin{aligned}
\|\phi(p_0) - \phi(x)\|_F^2 &= \langle \phi(p_0) - \phi(x), \phi(p_0) - \phi(x) \rangle_F \\
&= \langle \phi(p_0), \phi(p_0) \rangle_F + \langle \phi(x), \phi(x) \rangle_F \\
&\quad -2 \langle \phi(p_0), \phi(x) \rangle_F \\
&= k(p_0, p_0) + k(x, x) - 2k(p_0, x)
\end{aligned}
$$

- Plot $f(x) = \|\phi(p_0) - \phi(x)\|_F^2$

How to visualize?

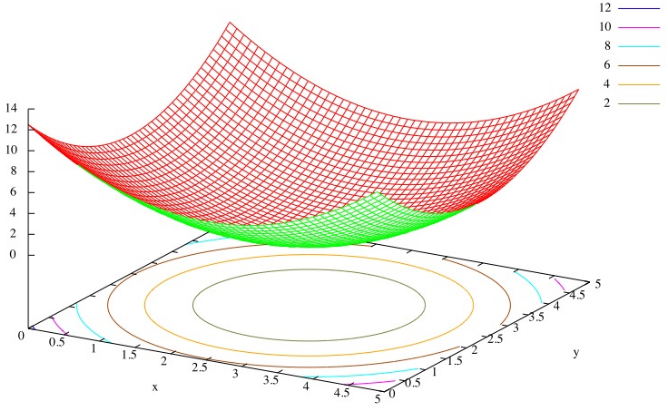
- Choose a point in input space p_0
- Calculate the distance from another point x to p_0 in the feature space:

$$
\begin{aligned}
\|\phi(p_0) - \phi(x)\|_F^2 &= \langle \phi(p_0) - \phi(x), \phi(p_0) - \phi(x) \rangle_F \\
&= \langle \phi(p_0), \phi(p_0) \rangle_F + \langle \phi(x), \phi(x) \rangle_F \\
&\quad -2 \langle \phi(p_0), \phi(x) \rangle_F \\
&= k(p_0, p_0) + k(x, x) - 2k(p_0, x)
\end{aligned}
$$

- Plot $f(x) = \|\phi(p_0) - \phi(x)\|_F^2$

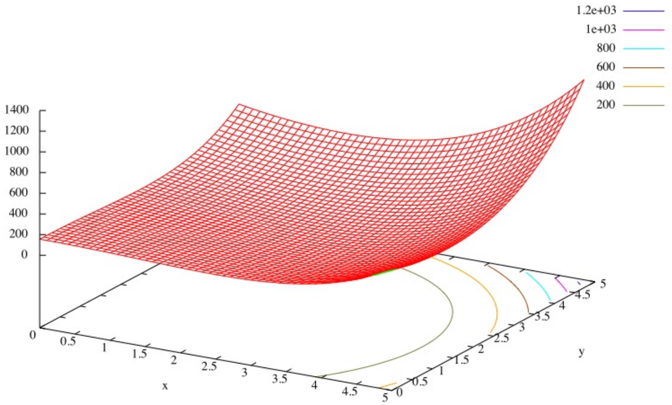
Identity kernel

$$k(x, z) = \langle x, z \rangle$$



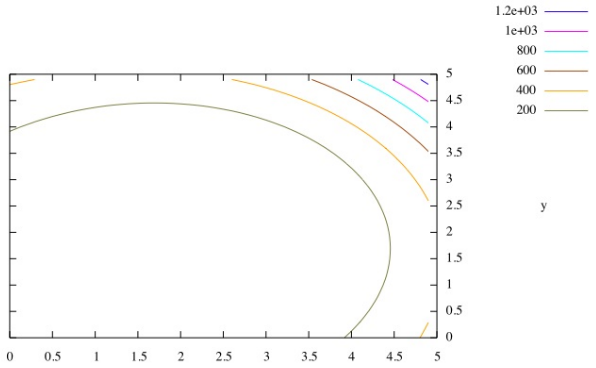
Quadratic kernel (1)

$$k(x, z) = \langle x, z \rangle^2$$



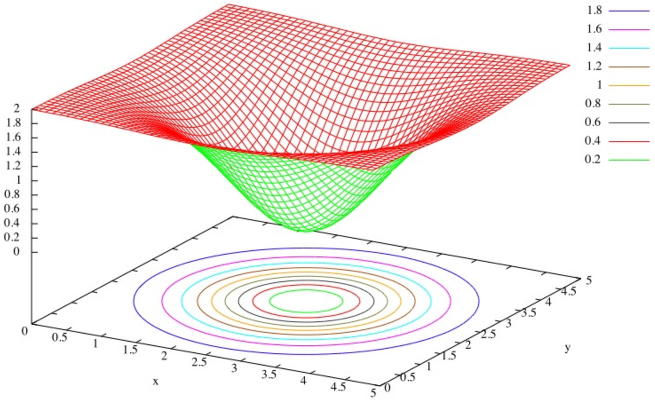
Identity kernel (2)

$$k(x, z) = \langle x, z \rangle^2$$



Gaussian kernel

$$k(\mathrm{x}, \mathrm{z}) = e^{-\frac{\|\mathrm{x}-\mathrm{z}\|^2}{2\sigma^2}}$$



Outline

Basic computations in feature space

- Means
- Distances
- Projections
- Covariance

Basic computations in feature space

- Means
- Distances
- Projections
- Covariance

Basic computations in feature space

- Means
- Distances
- Projections
- Covariance

Basic computations in feature space

- Means
- Distances
- Projections
- Covariance

Classification and regression

- Support Vector Machines
- Support Vector Regression
- Kernel Fisher Discriminant
- Kernel Perceptron

Classification and regression

- Support Vector Machines
- Support Vector Regression
- Kernel Fisher Discriminant
- Kernel Perceptron

Classification and regression

- Support Vector Machines
- Support Vector Regression
- Kernel Fisher Discriminant
- Kernel Perceptron

Classification and regression

- Support Vector Machines
- Support Vector Regression
- Kernel Fisher Discriminant
- Kernel Perceptron

Dimensionality reduction and clustering

- Kernel PCA
- Kernel CCA
- Kernel k-means
- Kernel SOM

Dimensionality reduction and clustering

- Kernel PCA
- Kernel CCA
- Kernel k-means
- Kernel SOM

Dimensionality reduction and clustering

- Kernel PCA
- Kernel CCA
- Kernel k-means
- Kernel SOM

Dimensionality reduction and clustering

- Kernel PCA
- Kernel CCA
- Kernel k-means
- Kernel SOM

Outline

Kernels in complex structured data

- Since kernel methods do not require an attribute-based representation of objects, it is possible to perform learning over complex structured data (or unstructured data)
- We only need to define a dot product operation (similarity, dissimilarity measure)
- Examples:
  - Strings
  - Texts
  - Trees
  - Graphs

Kernels in complex structured data

- Since kernel methods do not require an attribute-based representation of objects, it is possible to perform learning over complex structured data (or unstructured data)
- We only need to define a dot product operation (similarity, dissimilarity measure)
- Examples:
    - Strings
    - Texts
    - Trees
    - Graphs

Kernels in complex structured data

- Since kernel methods do not require an attribute-based representation of objects, it is possible to perform learning over complex structured data (or unstructured data)
- We only need to define a dot product operation (similarity, dissimilarity measure)
- Examples:
  - Strings
  - Texts
  - Trees
  - Graphs

Problem 2

How to do symbolic regression?

$$\Sigma = \{A, C, G, T\}$$

$$\begin{array}{rccc} f: & \Sigma^d & \rightarrow & \mathbb{R} \\ & ACGTA & \mapsto & 10.0 \\ & GTCCA & \mapsto & 11.3 \\ & GGTAC & \mapsto & 1.0 \\ & CCTGA & \mapsto & 4.5 \\ & \vdots & \vdots & \vdots \end{array}$$

Solution

- Define a kernel on strings

$$k : \Sigma^d \times \Sigma^d \to \mathbb{R}$$

- Use the kernel along with a kernel learning regression algorithm to find the regression function
- What is a good candidate for k?
  - a function that measures string similarity
  - higher value for similar strings, smaller value for different strings
  - $k(s_1 \dots s_d, t_1 \dots t_d) = \sum_{i=1}^{n} equal(s_i, t_i)$

$$equal(s_i, t_i) = \begin{cases} 1 & \text{if } s_i = t_i \\ 0 & \text{otherwise} \end{cases}$$

- $k(ACTAG, CCTCG) = ?$
- Is it a kernel?

Solution

- Define a kernel on strings

$$k : \Sigma^d \times \Sigma^d \to \mathbb{R}$$

- Use the kernel along with a kernel learning regression algorithm to find the regression function
- What is a good candidate for k?
  - a function that measures string similarity
  - higher value for similar strings, smaller value for different strings
  - $k(s_1 \ldots s_d, t_1 \ldots t_d) = \sum_{i=1}^{n} equal(s_i, t_i)$

$$equal(s_i, t_i) = \begin{cases} 1 & \text{if } s_i = t_i \\ 0 & \text{otherwise} \end{cases}$$

- $k(ACTAG, CCTCG) =?$
- Is it a kernel?

Solution

- Define a kernel on strings

$$k : \Sigma^d \times \Sigma^d \to \mathbb{R}$$

- Use the kernel along with a kernel learning regression algorithm to find the regression function
- What is a good candidate for k?
  - a function that measures string similarity
  - higher value for similar strings, smaller value for different strings
  - $k(s_1 \dots s_d, t_1 \dots t_d) = \sum_{i=1}^{n} equal(s_i, t_i)$

$$equal(s_i, t_i) = \begin{cases} 1 & \text{if } s_i = t_i \\ 0 & \text{otherwise} \end{cases}$$

- $k(ACTAG, CCTCG) = ?$
- Is it a kernel?

Solution

- Define a kernel on strings

$$k : \Sigma^d \times \Sigma^d \to \mathbb{R}$$

- Use the kernel along with a kernel learning regression algorithm to find the regression function
- What is a good candidate for k?
  - a function that measures string similarity
  - higher value for similar strings, smaller value for different strings
  - $k(s_1 \ldots s_d, t_1 \ldots t_d) = \sum_{i=1}^{n} equal(s_i, t_i)$

    $$equal(s_i, t_i) = \begin{cases} 1 & \text{if } s_i = t_i \\ 0 & \text{otherwise} \end{cases}$$
- $k(ACTAG, CCTCG) = ?$
- Is it a kernel?

Solution

- Define a kernel on strings

$$k : \Sigma^d \times \Sigma^d \to \mathbb{R}$$

- Use the kernel along with a kernel learning regression algorithm to find the regression function
- What is a good candidate for k?
  - a function that measures string similarity
  - higher value for similar strings, smaller value for different strings
  - $k(s_1 \dots s_d, t_1 \dots t_d) = \sum_{i=1}^{n} equal(s_i, t_i)$

    $$equal(s_i, t_i) = \begin{cases} 1 & \text{if } s_i = t_i \\ 0 & \text{otherwise} \end{cases}$$
- $k(ACTAG, CCTCG) =?$
- Is it a kernel?

Solution

- Define a kernel on strings

$$k : \Sigma^d \times \Sigma^d \to \mathbb{R}$$

- Use the kernel along with a kernel learning regression algorithm to find the regression function
- What is a good candidate for k?
  - a function that measures string similarity
  - higher value for similar strings, smaller value for different strings
  - $k(s_1 \ldots s_d, t_1 \ldots t_d) = \sum_{i=1}^{n} equal(s_i, t_i)$

    $$equal(s_i, t_i) = \begin{cases} 1 & \text{if } s_i = t_i \\ 0 & \text{otherwise} \end{cases}$$
- $k(ACTAG, CCTCG) = ?$
- Is it a kernel?

Induced Feature Space

- What is the feature space induced by k?
-

$$
\begin{aligned}
\phi : \Sigma^d &\rightarrow \mathbb{R}^{4d} \\
s_1 \dots s_d &\mapsto (x_1^1, \dots, x_4^1, x_1^2, \dots, x_4^2, \dots, x_1^d, \dots, x_4^d)
\end{aligned}
$$

$$
(x_1^j, \dots, x_4^j) = \begin{cases} (1,0,0,0) & \text{if } s_j = '\mathrm{A}' \\ (0,1,0,0) & \text{if } s_j = '\mathrm{C}' \\ (0,0,1,0) & \text{if } s_j = '\mathrm{G}' \\ (0,0,0,1) & \text{if } s_j = '\mathrm{T}' \end{cases}
$$

Induced Feature Space

- What is the feature space induced by k?
-

$$\begin{array}{rcl} \phi : \Sigma^d & \rightarrow & \mathbb{R}^{4d} \\ s_1 \ldots s_d & \mapsto & (x_1^1, \ldots, x_4^1, x_1^2, \ldots, x_4^2, \ldots, x_1^d, \ldots, x_4^d) \end{array}$$

$$(x_1^j, \ldots, x_4^j) = \begin{cases} (1,0,0,0) & \text{if } s_j = '\mathrm{A}' \\ (0,1,0,0) & \text{if } s_j = '\mathrm{C}' \\ (0,0,1,0) & \text{if } s_j = '\mathrm{G}' \\ (0,0,0,1) & \text{if } s_j = '\mathrm{T}' \end{cases}$$

References

Shawe-Taylor, J. and Cristianini, N. 2004 Kernel Methods for Pattern Analysis. Cambridge University Press.