Linear Regression and Classification Revisited

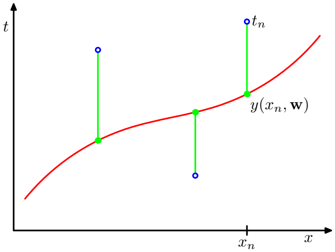
Dr. Víctor Uc Cetina

Facultad de Matemáticas
Universidad Autónoma de Yucatán

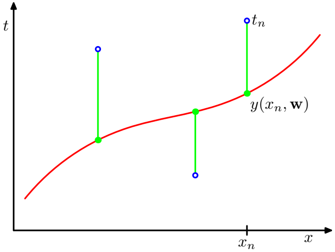
cetina@informatik.uni-hamburg.de
https://sites.google.com/view/victoruccetina

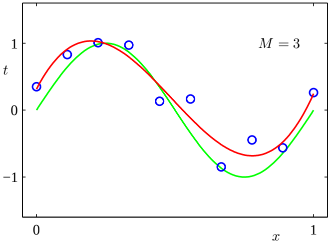
Content

Linear Regression

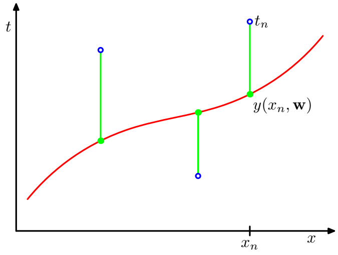


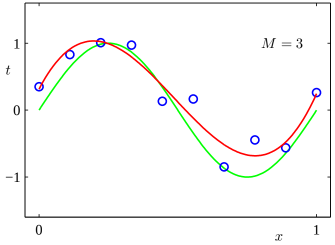
Linear Regression





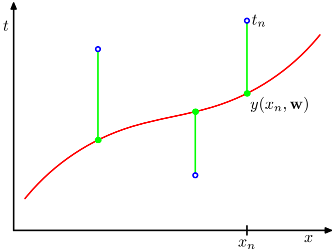
Linear Regression

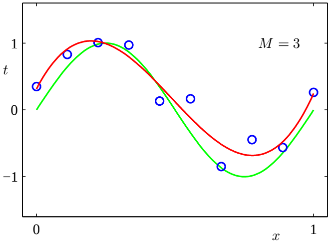




- x is the input variable, t is the output variable, $\mathbf{w}$ is the parameters vector of our model and the data points were generated from $\sin(2\pi x) + \varepsilon$.

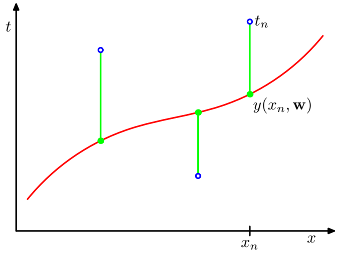
Linear Regression

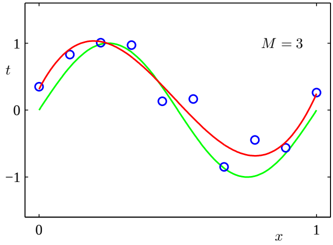




- x is the input variable, t is the output variable, $\mathbf{w}$ is the parameters vector of our model and the data points were generated from $\sin(2\pi x) + \varepsilon$.
- For our model $y(x, \mathbf{w}) = w_0 + w_1 x + \ldots + w_M x^M$, we need to search for the best M and we need to learn the parameters $\mathbf{w}$.

Linear Regression





- x is the input variable, t is the output variable, $\mathbf{w}$ is the parameters vector of our model and the data points were generated from $\sin(2\pi x) + \varepsilon$.
- For our model $y(x, \mathbf{w}) = w_0 + w_1 x + \ldots + w_M x^M$, we need to search for the best M and we need to learn the parameters $\mathbf{w}$.
- Such parameter vector $\mathbf{w}$ can be learned iteratively or directly.

Estimating the Parameters **w**

Stochastic Gradient Descent

Loop {

for $i = 1$ to m {

$$w_j := w_j + \alpha \left[t^{(i)} - y(x^{(i)}, \mathbf{w}) \right] x_j^{(i)} \qquad \text{(for every } j\text{)}.$$

}

}

Estimating the Parameters w

Stochastic Gradient Descent

Loop {

for $i = 1$ to m {

$$w_j := w_j + \alpha\left[t^{(i)} - y(x^{(i)}, \mathbf{w})\right]x_j^{(i)} \qquad \text{(for every } j\text{)}.$$

}

}

Normal Equations

$$\mathbf{w} = (\mathbf{X}^\top \mathbf{X})^{-1}\mathbf{X}^\top \mathbf{y}.$$

Locally Weighted Linear Regression

The algorithm works as follows:

1. Fit $\mathbf{w}$ to minimize $\sum_i \sigma^{(i)}(t^{(i)} - \mathbf{w}^\top x^{(i)})^2$.
2. Output $\mathbf{w}^\top x$.

Locally Weighted Linear Regression

The algorithm works as follows:

1. Fit $\mathbf{w}$ to minimize $\sum_i \sigma^{(i)}(t^{(i)} - \mathbf{w}^\top x^{(i)})^2$.
2. Output $\mathbf{w}^\top x$.

Where $\sigma^{(i)}$'s are non-negative valued weights.

A good choice for the weights is:

$$\sigma^{(i)} = \exp\left(-\frac{(x^{(i)}-x)^2}{2\tau^2}\right)$$

Locally Weighted Linear Regression

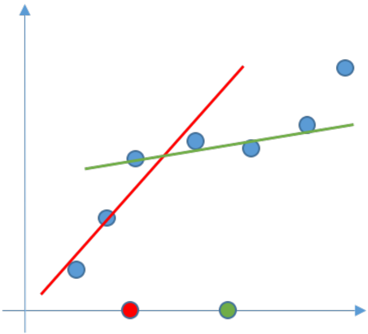
The algorithm works as follows:

1. Fit $\mathbf{w}$ to minimize $\sum_i \sigma^{(i)}(t^{(i)} - \mathbf{w}^\top x^{(i)})^2$.
2. Output $\mathbf{w}^\top x$.

Where $\sigma^{(i)}$'s are non-negative valued weights.

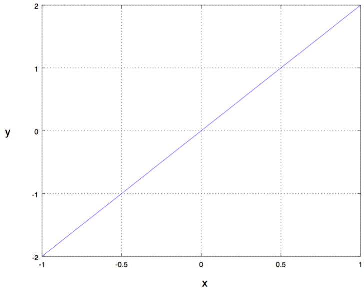
A good choice for the weights is:

$$\sigma^{(i)} = \exp\Big(-\frac{(x^{(i)}-x)^2}{2\tau^2}\Big)$$



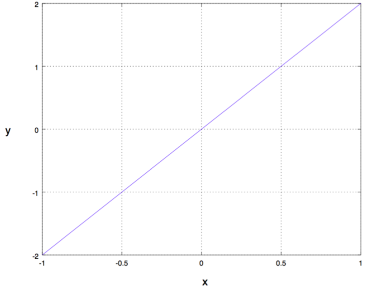
Polynomial Functions

$y = 2x$

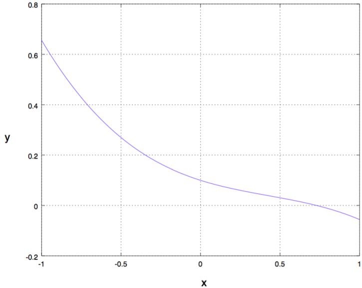


Polynomial Functions

$y = 2x$

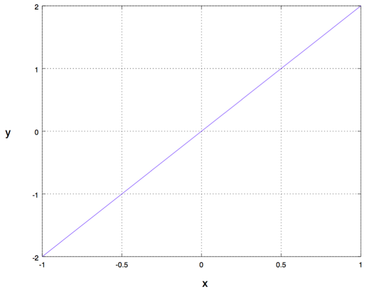


$y = 0.1 - 0.2x + 0.2x^2 - 0.156x^3$

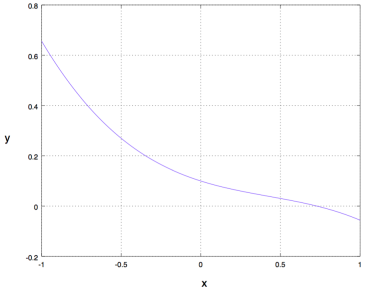


Polynomial Functions

$y = 2x$

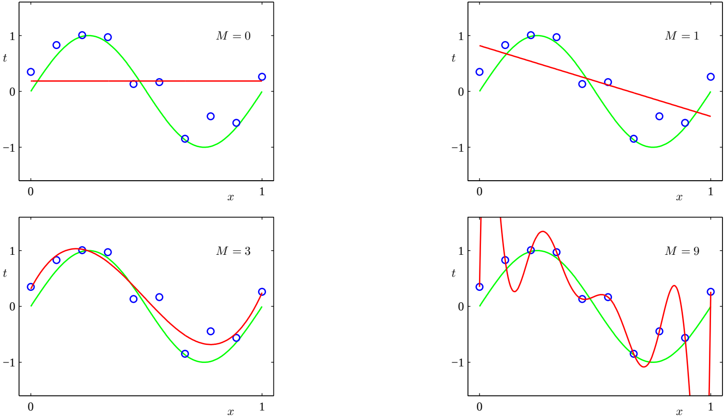


$y = 0.1 - 0.2x + 0.2x^2 - 0.156x^3$



- For polynomial functions, we need to try systematically different M's and evaluate the performance of our current model.

Polynomial Functions



- Polynomial functions with different orders M.

Evaluation of Performance

- For each choice of M we can evaluate the performance of the model using the root-mean-square error E_{RMS}.

$$E_{\mathrm{RMS}} = \sqrt{2E(\mathbf{w})/N}$$

where

$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n, \mathbf{w}) - t_n\}^2$$

Evaluation of Performance

- For each choice of M we can evaluate the performance of the model using the root-mean-square error E_{RMS}.

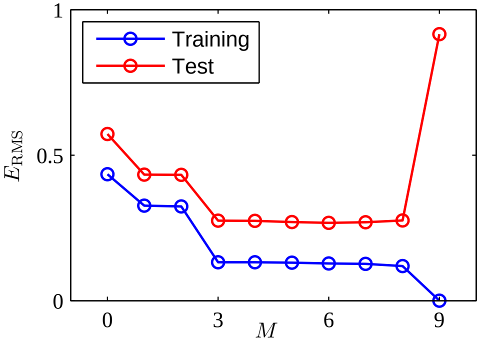
$$E_{\mathrm{RMS}} = \sqrt{2E(\mathbf{w})/N}$$

where

$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n, \mathbf{w}) - t_n\}^2$$

- This error can also be used to evaluate if our model's performance is improving after each iteration of the learning algorithm.

Evaluation of Performance



Generalized Linear Regression

- The goal of regression is to predict the value of one or more continuous target variables t given the value of a D-dimensional vector $\mathbf{x}$ of input variables.

Generalized Linear Regression

- The goal of regression is to predict the value of one or more continuous target variables t given the value of a D-dimensional vector $\mathbf{x}$ of input variables.
- The simplest linear model for regression is one that involves a linear combination of the input variables

$$y(\mathbf{x}, \mathbf{w}) = w_0 + w_1 x_1 + \ldots + w_D x_D$$

where

$$\mathbf{x} = (x_1, \ldots, x_D)^\top$$

Generalized Linear Regression

- The goal of regression is to predict the value of one or more continuous target variables t given the value of a D-dimensional vector $\mathbf{x}$ of input variables.
- The simplest linear model for regression is one that involves a linear combination of the input variables

$$y(\mathbf{x}, \mathbf{w}) = w_0 + w_1 x_1 + \ldots + w_D x_D$$

where

$$\mathbf{x} = (x_1, \ldots, x_D)^\top$$

- The key property of this model is that it is a linear function of the parameters $w_0, \ldots, w_D$. It is also, however, a linear function of the input variables x_i, and this imposes significant limitations on the model.

Generalized Linear Regression

- However, we can obtain a much more useful class of functions by taking linear combinations of a fixed set of nonlinear functions of the input variables, of the form

$$y(\mathbf{x}, \mathbf{w}) = w_0 + \sum_{j=1}^{M-1} w_j \phi_j(\mathbf{x})$$

where $\phi_j(\mathbf{x})$ are known as basis functions.

Generalized Linear Regression

- However, we can obtain a much more useful class of functions by taking linear combinations of a fixed set of nonlinear functions of the input variables, of the form

$$y(\mathbf{x}, \mathbf{w}) = w_0 + \sum_{j=1}^{M-1} w_j \phi_j(\mathbf{x})$$

  where $\phi_j(\mathbf{x})$ are known as basis functions.
- Such models are linear functions of the parameters, which gives them simple analytical properties, and yet can be nonlinear with respect to the input variables.

Generalized Linear Regression

- It is often convenient to define an additional dummy basis function $\phi_0(x) = 1$ so that

$$y(\mathbf{x}, \mathbf{w}) = \sum_{j=0}^{M-1} w_j \phi_j(\mathbf{x}) = \mathbf{w}^\top \phi(\mathbf{x})$$

where

$$w = (w_0, w_1, \dots, w_{M-1})^\top$$

and

$$\phi = (\phi_0, \phi_1, \dots, \phi_{M-1})^\top$$

Radial Basis Functions

- Polynomial regression is a particular example of radial basis functions models in which there is a single input variable x, and the basis functions take the form of powers of x so that $\phi_j(x) = x^j$.

Radial Basis Functions

- Polynomial regression is a particular example of radial basis functions models in which there is a single input variable x, and the basis functions take the form of powers of x so that $\phi_j(x) = x^j$.
- One limitation of polynomial basis functions is that they are global functions of the input variable, so that changes in one region of input space affect all other regions.

Radial Basis Functions

- Polynomial regression is a particular example of radial basis functions models in which there is a single input variable x, and the basis functions take the form of powers of x so that $\phi_j(x) = x^j$.
- One limitation of polynomial basis functions is that they are global functions of the input variable, so that changes in one region of input space affect all other regions.
- This can be resolved by dividing the input space into regions and fit a different polynomial in each region, leading to spline functions.

Radial Basis Functions

- Other possible choices for the basis functions are Gaussian basis functions and sigmoidal basis functions

Radial Basis Functions

- Other possible choices for the basis functions are Gaussian basis functions and sigmoidal basis functions
- Gaussian basis functions:

$$\phi_j(x) = \exp\left\{ -\frac{(x-\mu_j)^2}{2s^2} \right\}$$

where the μ_j govern the locations of the basis functions in input space, and the parameter s governs their spatial scale.

Radial Basis Functions

- Other possible choices for the basis functions are Gaussian basis functions and sigmoidal basis functions
- Gaussian basis functions:

$$\phi_j(x) = \exp\left\{ -\frac{(x-\mu_j)^2}{2s^2} \right\}$$

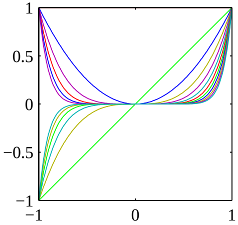
  where the μ_j govern the locations of the basis functions in input space, and the parameter s governs their spatial scale.
- Sigmoidal basis functions:

$$\phi_j(x) = \sigma\left(\frac{x-\mu_j}{s}\right)$$

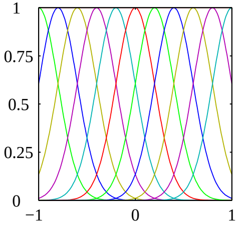
  where

$$\sigma(a) = \frac{1}{1+\exp(-a)}$$

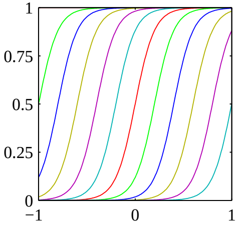
Radial Basis Functions



Polynomial

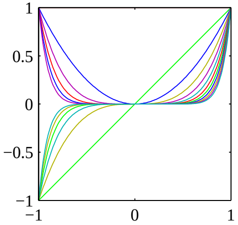


Gaussian

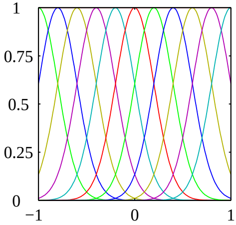


Sigmoidal

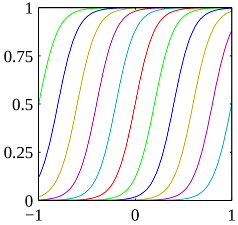
Radial Basis Functions



Polynomial



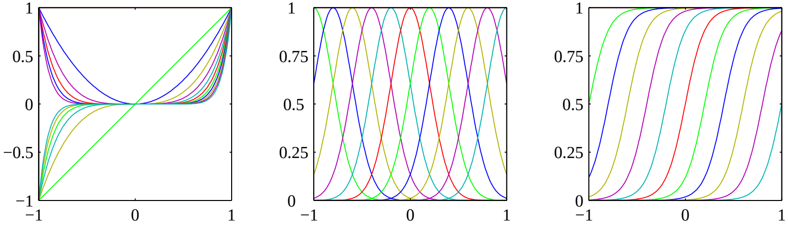
Gaussian



Sigmoidal

- Linear models have significant limitations as practical techniques for machine learning, particularly for problems involving input spaces of high dimensionality.

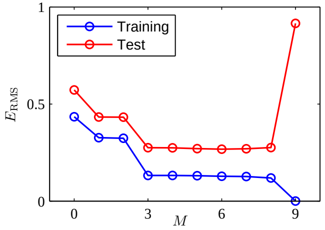
Radial Basis Functions



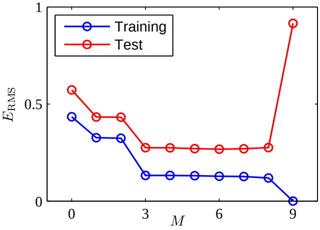
Polynomial   Gaussian   Sigmoidal

- Linear models have significant limitations as practical techniques for machine learning, particularly for problems involving input spaces of high dimensionality.
- However, they form the foundation of more sophisticated models such as neural networks and support vector machines.

Parameters Going Wild

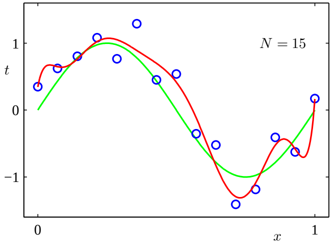


Parameters Going Wild

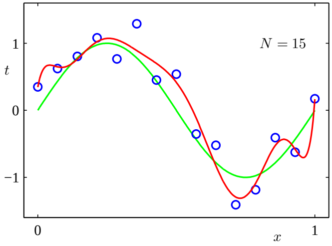


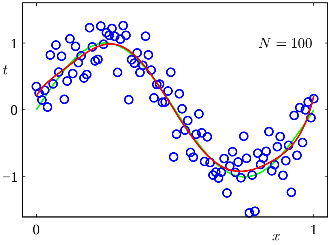
	$M=0$	$M=1$	$M=6$	$M=9$
w_0	0.19	0.82	0.31	0.35
w_1		-1.27	7.99	232.37
w_2			-25.43	-5321.83
w_3			17.37	48568.31
w_4				-231639.30
w_5				640042.26
w_6				-1061800.52
w_7				1042400.18
w_8				-557682.99
w_9				125201.43

Importance of Dataset Size

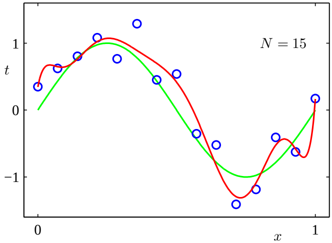


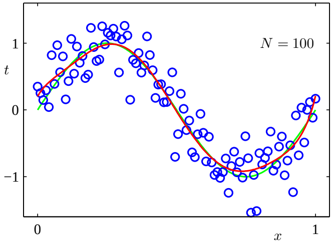
Importance of Dataset Size





Importance of Dataset Size





- Two solutions with $M = 9$. In the left using $N = 15$ training examples. In the right using $N = 100$ training examples.

Regularization Term

- We can add a regularization term to the error function in order to control over-fitting, so that the total error function to be minimized takes the form

$$E(\mathbf{w}) = E_D(\mathbf{w}) + \lambda E_W(\mathbf{w})$$

Regularization Term

- We can add a regularization term to the error function in order to control over-fitting, so that the total error function to be minimized takes the form

$$E(\mathbf{w}) = E_D(\mathbf{w}) + \lambda E_W(\mathbf{w})$$

where λ is the regularization coefficient that controls the relative importance of the data-dependent error $E_D(\mathbf{w})$ and the regularization term $E_W(\mathbf{w})$.

$$E_D(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{t_n - \mathbf{w}^\top \phi(\mathbf{x}_n)\}^2$$

Regularization Term

- One of the simplest forms of regularizer is given by the sum-of-squares of the weight vector elements

$$E_W(\mathbf{w}) = \frac{1}{2}\mathbf{w}^\top \mathbf{w}$$

Regularization Term

- One of the simplest forms of regularizer is given by the sum-of-squares of the weight vector elements

$$E_W(\mathbf{w}) = \frac{1}{2}\mathbf{w}^\top \mathbf{w}$$

- Then, instead of minimizing

$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{t_n - \mathbf{w}^\top \phi(\mathbf{x}_n)\}^2.$$

Regularization Term

- One of the simplest forms of regularizer is given by the sum-of-squares of the weight vector elements

$$E_W(\mathbf{w}) = \frac{1}{2}\mathbf{w}^\top \mathbf{w}$$

- Then, instead of minimizing

$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{t_n - \mathbf{w}^\top \phi(\mathbf{x}_n)\}^2.$$

- We minimize

$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{t_n - \mathbf{w}^\top \phi(\mathbf{x}_n)\}^2 + \frac{\lambda}{2}\mathbf{w}^\top \mathbf{w}.$$

Estimating the Parameters $\mathbf{w}$ with Regularization

Stochastic Gradient Descent

Loop {

for $i = 1$ to m {

$$w_j := w_j + \alpha\left[t^{(i)} - y(x^{(i)}, \mathbf{w})\right]x_j^{(i)} + \frac{\lambda}{m}w_j \qquad \text{(for every } j\text{)}.$$

}

}

Estimating the Parameters $\mathbf{w}$ with Regularization

Stochastic Gradient Descent

Loop {

for $i = 1$ to m {

$$w_j := w_j + \alpha\left[t^{(i)} - y(x^{(i)}, \mathbf{w})\right]x_j^{(i)} + \frac{\lambda}{m}w_j \qquad \text{(for every } j\text{).}$$

}

}

Normal Equations

$$\mathbf{w} = (\lambda \mathrm{I} + \mathbf{X}^\top \mathbf{X})^{-1}\mathbf{X}^\top \mathbf{y}.$$

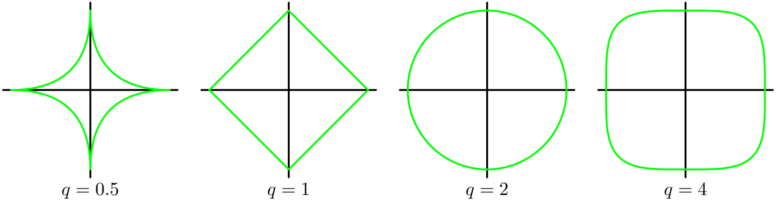
Different Types of Regularizers

- Sometimes a more general regularizer is used, for which de regularized error takes the form

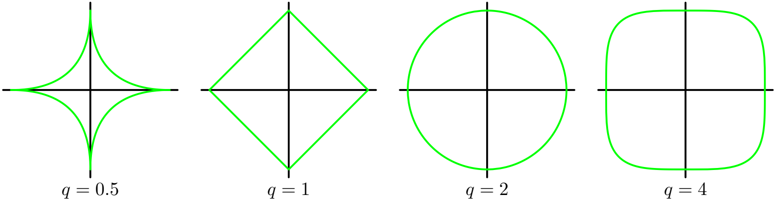
$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{t_n - \mathbf{w}^\top\phi(\mathbf{x}_n)\}^2 \; + \; \frac{\lambda}{2}\sum_{j=1}^{M}|w_j|^q.$$

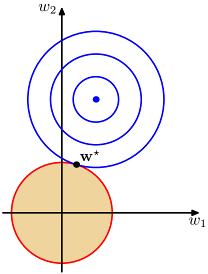
where $q = 2$ corresponds to the quadratic regularizer.

Types of Regularizers and their Effects

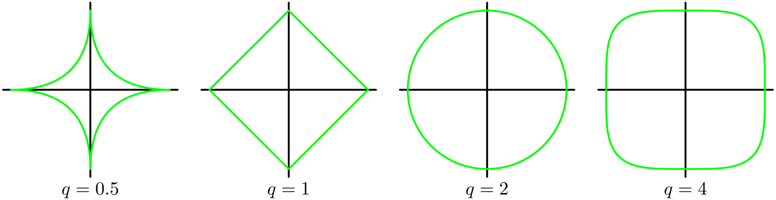


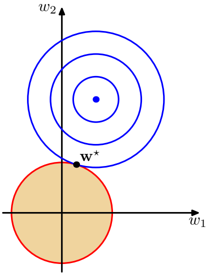
Types of Regularizers and their Effects

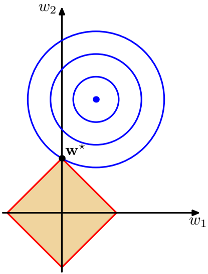




Types of Regularizers and their Effects







Benefits of Regularization

- Regularization allows complex models to be trained on data sets of limited size without severe over-fitting, essentially by limiting the effective model complexity.

Benefits of Regularization

- Regularization allows complex models to be trained on data sets of limited size without severe over-fitting, essentially by limiting the effective model complexity.
- However, the problem of determining the optimal model complexity is then shifted from one of finding the appropriate number of basis functions to one of determining a suitable value of the regularization coefficient λ.

Linear Classification

- The goal in classification is to take an input vector $\mathbf{x}$ and to assign it to one of K discrete classes C_k where $k = 1, \ldots, K$.

Linear Classification

- The goal in classification is to take an input vector $\mathbf{x}$ and to assign it to one of K discrete classes C_k where $k = 1, \ldots, K$.
- In the most common scenario, the classes are taken to be disjoint, so that each input is assigned to one and only one class.

Linear Classification

- The goal in classification is to take an input vector $\mathbf{x}$ and to assign it to one of K discrete classes C_k where $k = 1, \ldots, K$.
- In the most common scenario, the classes are taken to be disjoint, so that each input is assigned to one and only one class.
- The input space is thereby divided into decision regions whose boundaries are called decision boundaries or decision surfaces.

Linear Classification

- The goal in classification is to take an input vector $\mathbf{x}$ and to assign it to one of K discrete classes C_k where $k = 1, \ldots, K$.
- In the most common scenario, the classes are taken to be disjoint, so that each input is assigned to one and only one class.
- The input space is thereby divided into decision regions whose boundaries are called decision boundaries or decision surfaces.
- In linear models for classification, the decision surfaces are linear functions of the input vector $\mathbf{x}$ and hence are defined by (D-1)-dimensional hyperplanes within the D-dimensional input space.

Linear Classification

- The goal in classification is to take an input vector $\mathbf{x}$ and to assign it to one of K discrete classes C_k where $k = 1, \ldots, K$.
- In the most common scenario, the classes are taken to be disjoint, so that each input is assigned to one and only one class.
- The input space is thereby divided into decision regions whose boundaries are called decision boundaries or decision surfaces.
- In linear models for classification, the decision surfaces are linear functions of the input vector $\mathbf{x}$ and hence are defined by (D-1)-dimensional hyperplanes within the D-dimensional input space.
- Data sets whose classes can be separated exactly by linear decision surfaces are said to be linearly separable.

Linear Classification

- For classification problems, however, we wish to predict discrete class labels, or more generally posterior probabilities that lie in the range $(0, 1)$.

Linear Classification

- For classification problems, however, we wish to predict discrete class labels, or more generally posterior probabilities that lie in the range $(0, 1)$.
- To achieve this, we consider a generalization of this model in which we transform the linear function of **w** using a nonlinear function $f(\cdot)$ so that

$$y(\mathbf{x}) = f(\mathbf{w}^\top \mathbf{x} + w_o).$$

where $f(\cdot)$ is known as the activation function.

Linear Classification

- For classification problems, however, we wish to predict discrete class labels, or more generally posterior probabilities that lie in the range $(0, 1)$.
- To achieve this, we consider a generalization of this model in which we transform the linear function of $\mathbf{w}$ using a nonlinear function $f(\cdot)$ so that

$$y(\mathbf{x}) = f(\mathbf{w}^\top \mathbf{x} + w_o).$$

  where $f(\cdot)$ is known as the activation function.
- An input vector $\mathbf{x}$ is assigned to class C_1 if $y(\mathbf{x}) \geq 0$ and to class C_2 otherwise.

Discriminant Functions

- A discriminant is a function that takes an input vector $\mathbf{x}$ and assigns it to one of K classes, denoted C_k.

Discriminant Functions

- A discriminant is a function that takes an input vector $\mathbf{x}$ and assigns it to one of K classes, denoted C_k.
- The simplest representation of a linear discriminant function is obtained by taking a linear function of the input vector so that

$$y(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + w_o.$$

where $\mathbf{w}$ is called a weight vector, and w_0 is a bias.

Discriminant Functions

- The corresponding decision boundary is therefore defined by the relation $y(x) = 0$, which corresponds to a (D-1)-dimensional hyperplane within the D-dimensional input space.

Discriminant Functions

- The corresponding decision boundary is therefore defined by the relation $y(x) = 0$, which corresponds to a (D-1)-dimensional hyperplane within the D-dimensional input space.
- Consider two points x_A and x_B both of which lie on the decision surface.

Discriminant Functions

- The corresponding decision boundary is therefore defined by the relation $y(x) = 0$, which corresponds to a (D-1)-dimensional hyperplane within the D-dimensional input space.
- Consider two points x_A and x_B both of which lie on the decision surface.
- Because $y(\mathbf{x}_A) = y(\mathbf{x}_B) = 0$, we have $\mathbf{w}^\top(\mathbf{x}_A - \mathbf{x}_B) = 0$ and hence the vector $\mathbf{w}$ is orthogonal to every vector lying within the decision surface.

Discriminant Functions

- The corresponding decision boundary is therefore defined by the relation $y(x) = 0$, which corresponds to a (D-1)-dimensional hyperplane within the D-dimensional input space.
- Consider two points x_A and x_B both of which lie on the decision surface.
- Because $y(\mathbf{x}_A) = y(\mathbf{x}_B) = 0$, we have $\mathbf{w}^\top(\mathbf{x}_A - \mathbf{x}_B) = 0$ and hence the vector $\mathbf{w}$ is orthogonal to every vector lying within the decision surface.
- So $\mathbf{w}$ determines the orientation of the decision surface.

Discriminant Functions

Explaining $\mathbf{w}^\top(\mathbf{x}_A - \mathbf{x}_B) = 0$.

Discriminant Functions

Explaining $\mathbf{w}^\top(\mathbf{x}_A - \mathbf{x}_B) = 0$.

$$y(\mathbf{x}_A) = y(\mathbf{x}_B) = 0$$

Discriminant Functions

Explaining $\mathbf{w}^\top(\mathbf{x}_A - \mathbf{x}_B) = 0$.

$$y(\mathbf{x}_A) = y(\mathbf{x}_B) = 0$$

$$y(\mathbf{x}_A) = y(\mathbf{x}_B)$$

Discriminant Functions

Explaining $\mathbf{w}^\top(\mathbf{x}_A - \mathbf{x}_B) = 0$.

$$y(\mathbf{x}_A) = y(\mathbf{x}_B) = 0$$

$$y(\mathbf{x}_A) = y(\mathbf{x}_B)$$

$$\mathbf{w}^\top \mathbf{x}_A + w_o = \mathbf{w}^\top \mathbf{x}_B + w_o$$

Discriminant Functions

Explaining $\mathbf{w}^\top(\mathbf{x}_A - \mathbf{x}_B) = 0$.

$$y(\mathbf{x}_A) = y(\mathbf{x}_B) = 0$$

$$y(\mathbf{x}_A) = y(\mathbf{x}_B)$$

$$\mathbf{w}^\top \mathbf{x}_A + w_o = \mathbf{w}^\top \mathbf{x}_B + w_o$$

$$\mathbf{w}^\top \mathbf{x}_A = \mathbf{w}^\top \mathbf{x}_B$$

Discriminant Functions

Explaining $\mathbf{w}^\top(\mathbf{x}_A - \mathbf{x}_B) = 0$.

$$y(\mathbf{x}_A) = y(\mathbf{x}_B) = 0$$

$$y(\mathbf{x}_A) = y(\mathbf{x}_B)$$

$$\mathbf{w}^\top \mathbf{x}_A + w_o = \mathbf{w}^\top \mathbf{x}_B + w_o$$

$$\mathbf{w}^\top \mathbf{x}_A = \mathbf{w}^\top \mathbf{x}_B$$

$$\mathbf{w}^\top \mathbf{x}_A - \mathbf{w}^\top \mathbf{x}_B = 0$$

Discriminant Functions

Explaining $\mathbf{w}^\top(\mathbf{x}_A - \mathbf{x}_B) = 0$.

$$y(\mathbf{x}_A) = y(\mathbf{x}_B) = 0$$

$$y(\mathbf{x}_A) = y(\mathbf{x}_B)$$

$$\mathbf{w}^\top \mathbf{x}_A + w_o = \mathbf{w}^\top \mathbf{x}_B + w_o$$

$$\mathbf{w}^\top \mathbf{x}_A = \mathbf{w}^\top \mathbf{x}_B$$

$$\mathbf{w}^\top \mathbf{x}_A - \mathbf{w}^\top \mathbf{x}_B = 0$$

$$\mathbf{w}^\top(\mathbf{x}_A - \mathbf{x}_B) = 0$$

Discriminant Functions

Explaining $\mathbf{w}^\top(\mathbf{x}_A - \mathbf{x}_B) = 0$.

$$y(\mathbf{x}_A) = y(\mathbf{x}_B) = 0$$

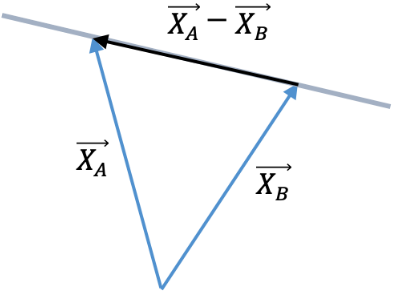
$$y(\mathbf{x}_A) = y(\mathbf{x}_B)$$

$$\mathbf{w}^\top\mathbf{x}_A + w_o = \mathbf{w}^\top\mathbf{x}_B + w_o$$

$$\mathbf{w}^\top\mathbf{x}_A = \mathbf{w}^\top\mathbf{x}_B$$

$$\mathbf{w}^\top\mathbf{x}_A - \mathbf{w}^\top\mathbf{x}_B = 0$$

$$\mathbf{w}^\top(\mathbf{x}_A - \mathbf{x}_B) = 0$$



Discriminant Functions

- Similarly, if x is a point on the decision surface, then $y(x) = 0$, and so the normal distance from the origin to the decision surface is given by

$$\frac{\mathbf{w}^\top \mathbf{x}}{||\mathbf{w}||} = -\frac{w_o}{||\mathbf{w}||}$$

Discriminant Functions

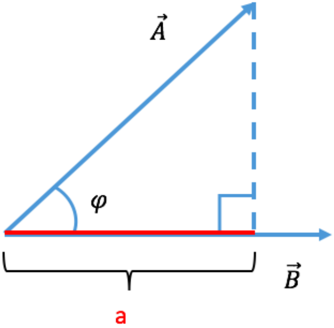
- Similarly, if x is a point on the decision surface, then $y(x) = 0$, and so the normal distance from the origin to the decision surface is given by

$$\frac{\mathbf{w}^\top \mathbf{x}}{||\mathbf{w}||} = -\frac{w_o}{||\mathbf{w}||}$$

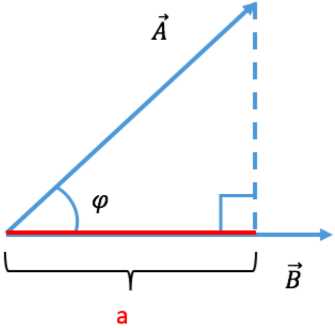
- We therefore see that the bias parameter w_0 determines the location of the decision surface.

Discriminant Functions

$$\cos\varphi = \frac{a}{||\vec{A}||}$$

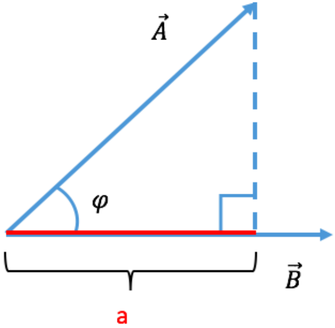


Discriminant Functions



$$\begin{aligned} \cos\varphi &= \frac{a}{||\vec{A}||} \\ a &= ||\vec{A}||\cos\varphi \end{aligned} \tag{1}$$

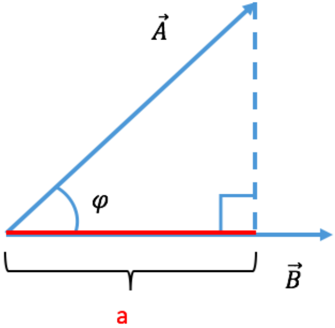
Discriminant Functions



$$
\begin{aligned}
\cos\varphi &= \frac{a}{||\vec{A}||} \\
a &= ||\vec{A}||\cos\varphi
\end{aligned}
\tag{1}
$$

$$\vec{A}\cdot\vec{B} = ||\vec{A}||\;||\vec{B}||\cos\varphi$$

Discriminant Functions



$$\cos\varphi = \frac{a}{||\vec{A}||}$$

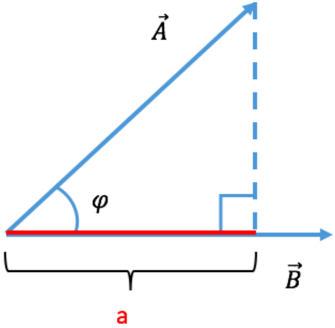
$$a = ||\vec{A}||\cos\varphi \qquad (1)$$

$$\vec{A}\cdot\vec{B} = ||\vec{A}||\;||\vec{B}||\cos\varphi$$

$$\cos\varphi = \frac{\vec{A}\cdot\vec{B}}{||\vec{A}||\;||\vec{B}||} \qquad (2)$$

Subst. (2) in (1)

Discriminant Functions



$$\cos\varphi = \frac{a}{||\vec{A}||}$$

$$a = ||\vec{A}||\cos\varphi \qquad (1)$$

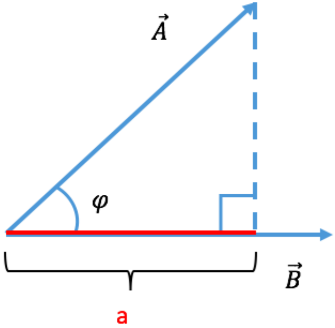
$$\vec{A}\cdot\vec{B} = ||\vec{A}||\;||\vec{B}||\cos\varphi$$

$$\cos\varphi = \frac{\vec{A}\cdot\vec{B}}{||\vec{A}||\;||\vec{B}||} \qquad (2)$$

Subst. (2) in (1)

$$a = ||\vec{A}||\left(\frac{\vec{A}\cdot\vec{B}}{||\vec{A}||\;||\vec{B}||}\right)$$

Discriminant Functions



$$\cos\varphi = \frac{a}{||\vec{A}||}$$

$$a = ||\vec{A}||\cos\varphi \qquad (1)$$

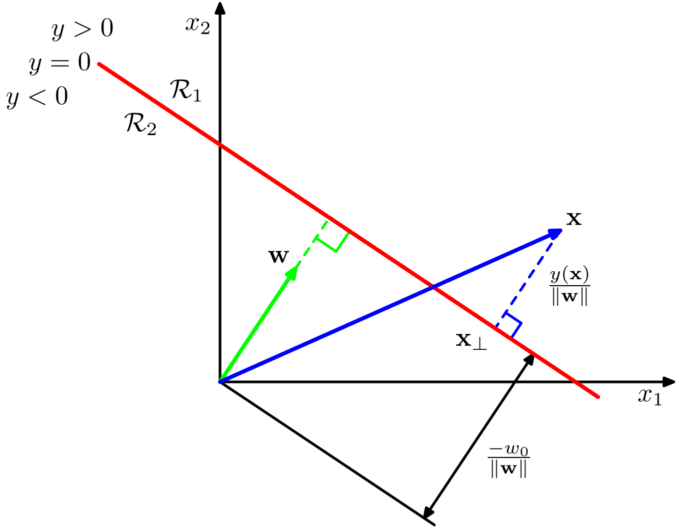
$$\vec{A}\cdot\vec{B} = ||\vec{A}||\;||\vec{B}||\cos\varphi$$

$$\cos\varphi = \frac{\vec{A}\cdot\vec{B}}{||\vec{A}||\;||\vec{B}||} \qquad (2)$$

Subst. (2) in (1)

$$a = ||\vec{A}||\left(\frac{\vec{A}\cdot\vec{B}}{||\vec{A}||\;||\vec{B}||}\right) = \frac{\vec{A}\cdot\vec{B}}{||\vec{B}||}$$

Discriminant Functions



Multiple Classes

Consider the extension of linear discriminants to $K > 2$ classes.
There are two approaches:

Multiple Classes

Consider the extension of linear discriminants to $K > 2$ classes. There are two approaches:

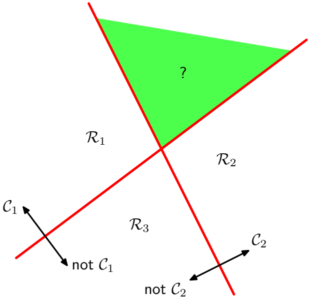
- One-versus-the-rest classifier: build a K-class discriminant by combining a number of two-class discriminant functions. However, this leads to some serious ambiguity difficulties.

Multiple Classes

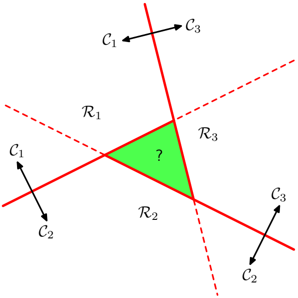
Consider the extension of linear discriminants to $K > 2$ classes. There are two approaches:

- One-versus-the-rest classifier: build a K-class discriminant by combining a number of two-class discriminant functions. However, this leads to some serious ambiguity difficulties.
- One-versus-one classifier: Introduce $K(K-1)/2$ binary discriminant functions, one for every possible pair of classes. Each point is then classified according to a majority vote amongst the discriminant functions. However, this too runs into the problem of ambiguous regions.

Multiple Classes

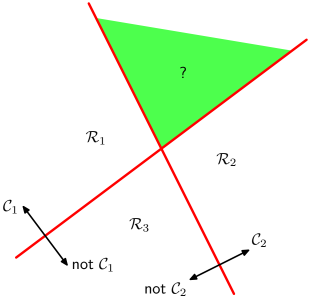


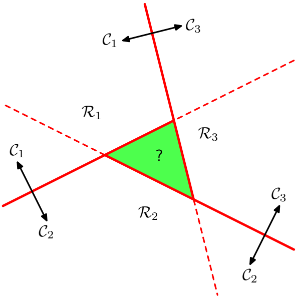
One-versus-the-rest



One-versus-one

Multiple Classes





One-versus-the-rest One-versus-one

- Both result in ambiguous regions of input space.

Multiple Classes

- Consider a single K class discriminant of the form

$$y_k(x) = w_k^\top x + w_{k0}.$$

Multiple Classes

- Consider a single K class discriminant of the form

$$y_k(x) = w_k^\top x + w_{k0}.$$

- Then we can assign a point x to class C_k if

$$y_k(x) > y_j(x) \text{ for all } j \neq k.$$

Multiple Classes

- Consider a single K class discriminant of the form

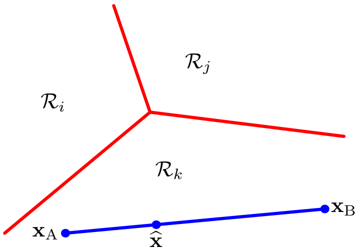
$$y_k(x) = w_k^\top x + w_{k0}.$$

- Then we can assign a point x to class C_k if

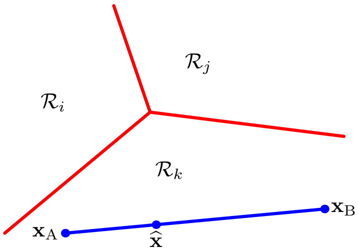
$$y_k(x) > y_j(x) \;\text{ for all }\; j \neq k.$$

- Decision regions of such a discriminant are always singly connected and convex.

Multiple Classes

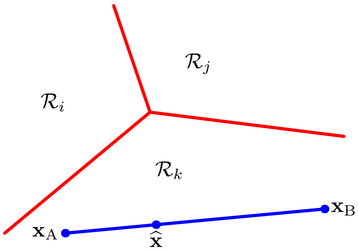


Multiple Classes



- Consider two points x_A and x_B both in decision region R_k.

Multiple Classes

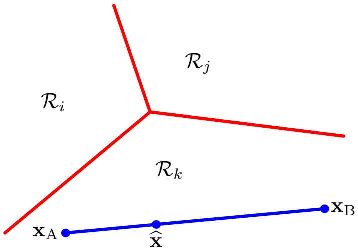


- Consider two points x_A and x_B both in decision region R_k.
- Any point $\hat{x}$ on line connecting x_A and x_B can be expressed as

$$\hat{x} = \lambda x_A + (1 - \lambda) x_B$$

where $0 \leq \lambda \leq 1$

Multiple Classes



- Consider two points x_A and x_B both in decision region R_k.
- Any point $\hat{x}$ on line connecting x_A and x_B can be expressed as

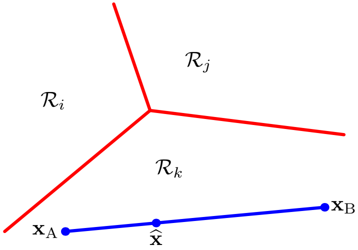
$$\hat{x} = \lambda x_A + (1 - \lambda) x_B$$

where $0 \leq \lambda \leq 1$

- From linearity of discriminant functions, it follows that

$$y_k(\hat{x}) = \lambda y_k(x_A) + (1 - \lambda) y_k(x_B).$$

Multiple Classes



- Because both x_A and x_B lie inside R_k, it follows that

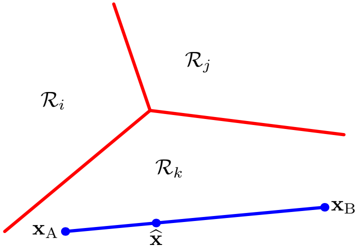
$$y_k(x_A) > y_j(x_A),$$

and

$$y_k(x_B) > y_j(x_B),$$

for all $j \neq k$.

Multiple Classes



- Because both x_A and x_B lie inside R_k, it follows that

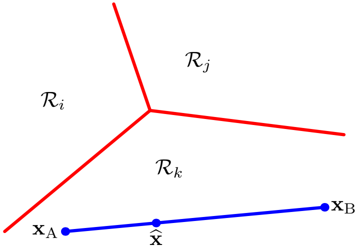
$$y_k(x_A) > y_j(x_A),$$

and

$$y_k(x_B) > y_j(x_B),$$

for all $j \neq k$.
- Hence $y_k(\hat{x}) > y_j(\hat{x})$, and so $\hat{x}$ also lies inside R_k.

Multiple Classes



- Because both x_A and x_B lie inside R_k, it follows that

$$y_k(x_A) > y_j(x_A),$$

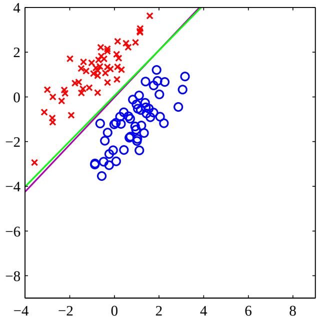
and

$$y_k(x_B) > y_j(x_B),$$

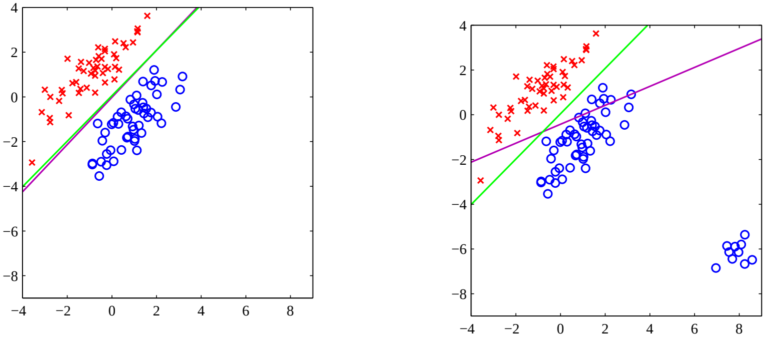
for all $j \neq k$.

- Hence $y_k(\hat{x}) > y_j(\hat{x})$, and so $\hat{x}$ also lies inside R_k.
- Thus R_k is singly connected and convex.

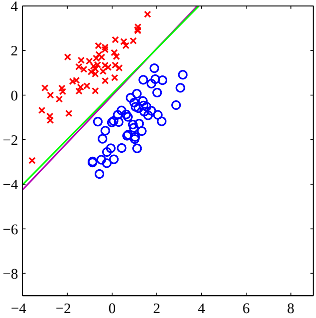
Least Squares Vs Logistic Regression

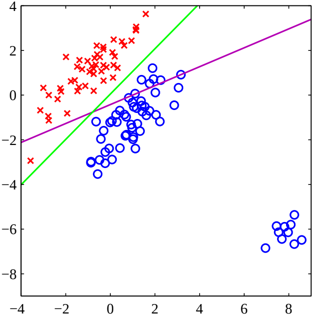


Least Squares Vs Logistic Regression



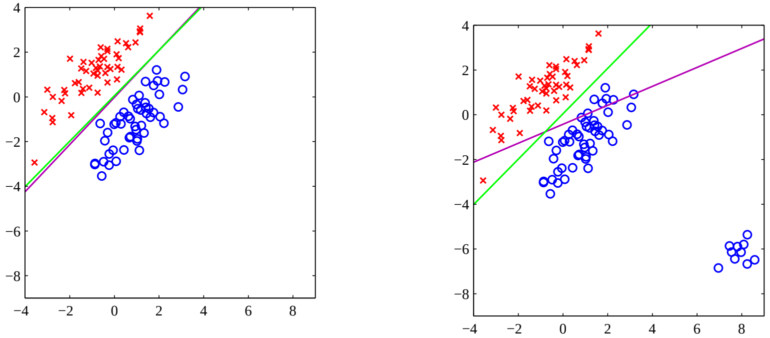
Least Squares Vs Logistic Regression





- Decision boundaries found by least squares (magenta curve) and also by the logistic regression model (green curve).

Least Squares Vs Logistic Regression



- Decision boundaries found by least squares (magenta curve) and also by the logistic regression model (green curve).
- The right-hand plot shows that least squares (Maximum Likelihood with Gaussian assumption) is highly sensitive to outliers, unlike logistic regression.

Fisher's Linear Discriminant

- One way to view a linear classification model is in terms of dimensionality reduction.

Fisher's Linear Discriminant

- One way to view a linear classification model is in terms of dimensionality reduction.
- Consider the case of two classes, and suppose we take D-dimensional input vector $\mathbf{x}$ and project it down to one dimension using
$$y = \mathbf{w}^\top \mathbf{x}.$$

Fisher's Linear Discriminant

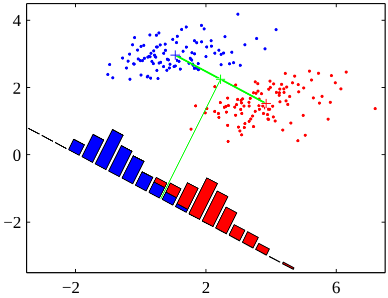
- One way to view a linear classification model is in terms of dimensionality reduction.
- Consider the case of two classes, and suppose we take D-dimensional input vector $\mathbf{x}$ and project it down to one dimension using
$$y = \mathbf{w}^\top \mathbf{x}.$$
- If we place a threshold on y and classify $y \geq -w_o$ as class C_1, and otherwise class C_2, then we obtain a standard linear classifier.

Fisher's Linear Discriminant

- In general, the projection onto one dimension leads to a considerable loss of information, and classes that are well separated in the original D-dimensional space may become strongly overlapping in one dimension.

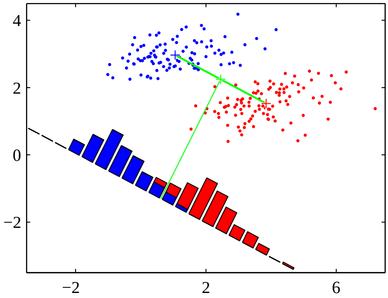
Fisher's Linear Discriminant

- In general, the projection onto one dimension leads to a considerable loss of information, and classes that are well separated in the original D-dimensional space may become strongly overlapping in one dimension.



Fisher's Linear Discriminant

- In general, the projection onto one dimension leads to a considerable loss of information, and classes that are well separated in the original D-dimensional space may become strongly overlapping in one dimension.



- However, by adjusting the components of the weight vector $\mathbf{w}$, we can select a projection that maximizes the class separation.

Fisher's Linear Discriminant

- Consider a two-class problem in which there are N_1 points of class C_1 and N_2 points of class C_2, so that the mean vectors of the two classes are given by

$$\mathbf{m}_1 = \frac{1}{N_1} \sum_{n \in C_1} \mathbf{x}_n,$$

$$\mathbf{m}_2 = \frac{1}{N_2} \sum_{n \in C_2} \mathbf{x}_n.$$

Fisher's Linear Discriminant

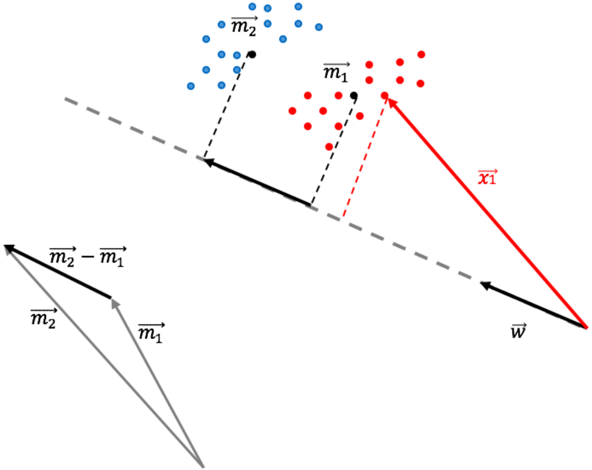
- Consider a two-class problem in which there are N_1 points of class C_1 and N_2 points of class C_2, so that the mean vectors of the two classes are given by

$$\mathbf{m}_1 = \frac{1}{N_1} \sum_{n \in C_1} \mathbf{x}_n,$$

$$\mathbf{m}_2 = \frac{1}{N_2} \sum_{n \in C_2} \mathbf{x}_n.$$

- The simplest measure of the separation of the classes, when projected onto $\mathbf{w}$, is the separation of the projected class means.

Fisher's Linear Discriminant



Fisher's Linear Discriminant

- This suggests that we might choose $\mathbf{w}$ so as to maximize

$$m_2 - m_1 = \mathbf{w}^\top(\mathbf{m}_2 - \mathbf{m}_1)$$

where

$$m_k = \mathbf{w}^\top \mathbf{m}_k$$

is the mean of the projected data from class C_k.

Fisher's Linear Discriminant

- This suggests that we might choose $\mathbf{w}$ so as to maximize

  $$m_2 - m_1 = \mathbf{w}^\top(\mathbf{m}_2 - \mathbf{m}_1)$$

  where

  $$m_k = \mathbf{w}^\top \mathbf{m}_k$$

  is the mean of the projected data from class C_k.
- However, this expression can be made arbitrarily large simply by increasing the magnitude of $\mathbf{w}$.

Fisher's Linear Discriminant

- This suggests that we might choose **w** so as to maximize

$$m_2 - m_1 = \mathbf{w}^\top(\mathbf{m}_2 - \mathbf{m}_1)$$

  where

$$m_k = \mathbf{w}^\top \mathbf{m}_k$$

  is the mean of the projected data from class C_k.
- However, this expression can be made arbitrarily large simply by increasing the magnitude of **w**.
- To solve this problem we could constrain **w** to have unit length, so that $\sum_i w_i^2 = 1$.

Fisher's Linear Discriminant

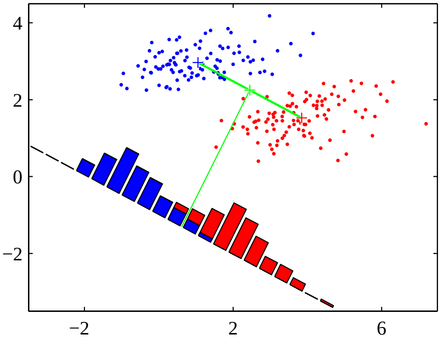
- Problem with this approach: two classes that are well separated in the original two-dimensional space may have considerable overlap when projected onto the line joining their means.

Fisher's Linear Discriminant

- Problem with this approach: two classes that are well separated in the original two-dimensional space may have considerable overlap when projected onto the line joining their means.
- This difficulty arises from strongly nondiagonal covariances of the class distributions.

Fisher's Linear Discriminant

- Problem with this approach: two classes that are well separated in the original two-dimensional space may have considerable overlap when projected onto the line joining their means.
- This difficulty arises from strongly nondiagonal covariances of the class distributions.



Fisher's Linear Discriminant

- The idea proposed by Fisher is to maximize a function that will give a large separation between the projected class means while also giving a small variance within each class, thereby minimizing the class overlap.

Fisher's Linear Discriminant

- The idea proposed by Fisher is to maximize a function that will give a large separation between the projected class means while also giving a small variance within each class, thereby minimizing the class overlap.
- The projection then transforms the set of labelled data points in $\mathbf{x}$ into a labelled set in the one-dimensional space y.

Fisher's Linear Discriminant

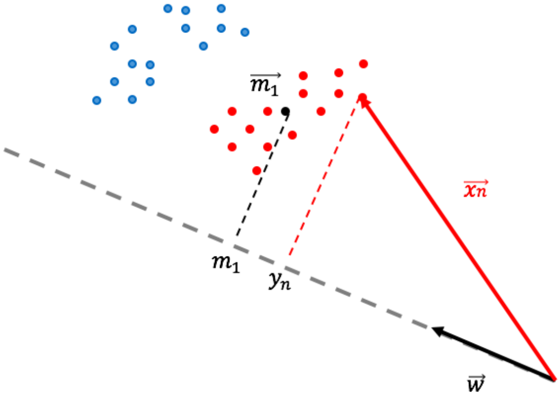
- The idea proposed by Fisher is to maximize a function that will give a large separation between the projected class means while also giving a small variance within each class, thereby minimizing the class overlap.
- The projection then transforms the set of labelled data points in $\mathbf{x}$ into a labelled set in the one-dimensional space y.
- The within-class variance of the transformed data from class C_k is therefore given by

$$s_k^2 = \sum_{n \in C_k} (y_n - m_k)^2$$

where

$$y_n = \mathbf{w}^\top \mathbf{x}_n.$$

Fisher's Linear Discriminant



Fisher's Linear Discriminant

- We can define the total within-class variance for the whole data set to be simply $s_1^2 + s_2^2$.

Fisher's Linear Discriminant

- We can define the total within-class variance for the whole data set to be simply $s_1^2 + s_2^2$.
- The Fisher criterion is defined to be the ratio of the between-class variance to the within-class variance and is given by

$$J(\mathbf{w}) = \frac{(m_2 - m_1)^2}{s_1^2 + s_2^2}.$$

Fisher's Linear Discriminant

- We can define the total within-class variance for the whole data set to be simply $s_1^2 + s_2^2$.
- The Fisher criterion is defined to be the ratio of the between-class variance to the within-class variance and is given by

$$J(\mathbf{w}) = \frac{(m_2 - m_1)^2}{s_1^2 + s_2^2}.$$

- We can make the dependence on **w** explicit and rewrite the Fisher criterion in the form

$$J(\mathbf{w}) = \frac{\mathbf{w}^\top \mathbf{S}_B \mathbf{w}}{\mathbf{w}^\top \mathbf{S}_W \mathbf{w}}$$

Fisher's Linear Discriminant

- In

$$J(\mathbf{w}) = \frac{\mathbf{w}^\top \mathbf{S}_B \mathbf{w}}{\mathbf{w}^\top \mathbf{S}_W \mathbf{w}}$$

Fisher's Linear Discriminant

- In

$$J(\mathbf{w}) = \frac{\mathbf{w}^\top \mathbf{S}_B \mathbf{w}}{\mathbf{w}^\top \mathbf{S}_W \mathbf{w}}$$

- $\mathbf{S}_B$ is the between-class covariance matrix given by

$$\mathbf{S}_B = (\mathbf{m}_2 - \mathbf{m}_1)(\mathbf{m}_2 - \mathbf{m}_1)^\top$$

Fisher's Linear Discriminant

- In
$$J(\mathbf{w}) = \frac{\mathbf{w}^\top \mathbf{S}_B \mathbf{w}}{\mathbf{w}^\top \mathbf{S}_W \mathbf{w}}$$
- $\mathbf{S}_B$ is the between-class covariance matrix given by
$$\mathbf{S}_B = (\mathbf{m}_2 - \mathbf{m}_1)(\mathbf{m}_2 - \mathbf{m}_1)^\top$$
- and $\mathbf{S}_W$ is the within-class covariance matrix given by
$$\mathbf{S}_W = \sum_{n \in C_1} (\mathbf{x}_n - \mathbf{m}_1)(\mathbf{x}_n - \mathbf{m}_1)^\top + \sum_{n \in C_2} (\mathbf{x}_n - \mathbf{m}_2)(\mathbf{x}_n - \mathbf{m}_2)^\top.$$

Fisher's Linear Discriminant

- In

$$J(\mathbf{w}) = \frac{\mathbf{w}^\top \mathbf{S}_B \mathbf{w}}{\mathbf{w}^\top \mathbf{S}_W \mathbf{w}}$$

- $\mathbf{S}_B$ is the between-class covariance matrix given by

$$\mathbf{S}_B = (\mathbf{m}_2 - \mathbf{m}_1)(\mathbf{m}_2 - \mathbf{m}_1)^\top$$

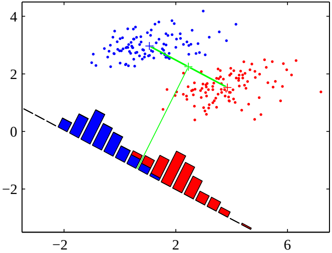
- and $\mathbf{S}_W$ is the within-class covariance matrix given by

$$\mathbf{S}_W = \sum_{n \in C_1} (\mathbf{x}_n - \mathbf{m}_1)(\mathbf{x}_n - \mathbf{m}_1)^\top + \sum_{n \in C_2} (\mathbf{x}_n - \mathbf{m}_2)(\mathbf{x}_n - \mathbf{m}_2)^\top.$$

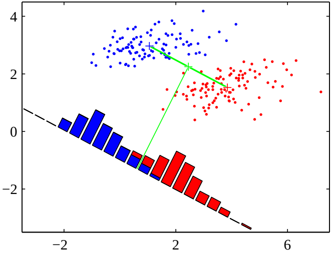
- Finally, by maximizing $J(\mathbf{w})$ we find that

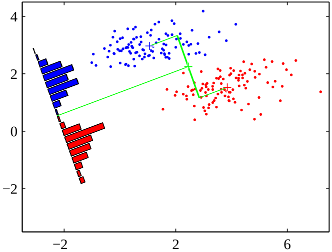
$$\mathbf{w} \propto \mathbf{S}_W^{-1}(\mathbf{m}_2 - \mathbf{m}_1).$$

Fisher's Linear Discriminant

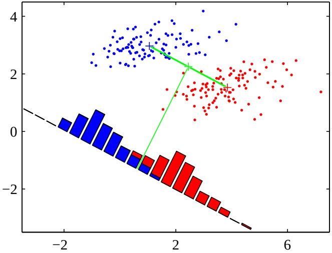


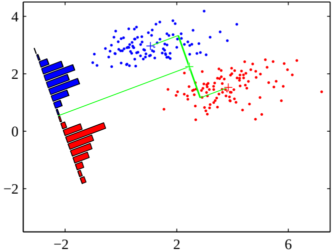
Fisher's Linear Discriminant





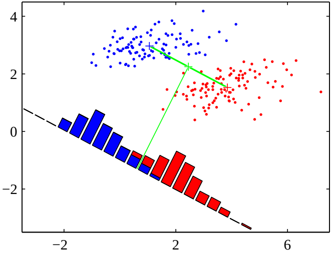
Fisher's Linear Discriminant

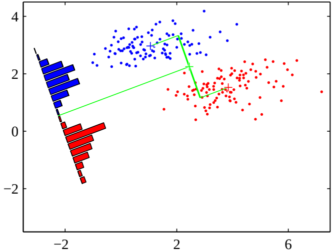




- The result is known as Fisher's linear discriminant, although strictly it is not a discriminant but rather a specific choice of direction for projection of the data down to one dimension.

Fisher's Linear Discriminant





- The result is known as Fisher's linear discriminant, although strictly it is not a discriminant but rather a specific choice of direction for projection of the data down to one dimension.
- However, the projected data can subsequently be used to construct a discriminant, by choosing a threshold y_0 so that we classify a new point as belonging to C_1 if $y(\mathbf{x}) \geq y_0$ and classify it as belonging to C_2 otherwise.

Reference

- Andrew Ng. **Machine Learning Course Notes**. 2003.
- Christopher Bishop. **Pattern Recognition and Machine Learning**. Springer. 2006.

Thank you!