Statistical Machine Learning

Christian Walder

Machine Learning Research Group
CSIRO Data61

and

College of Engineering and Computer Science
The Australian National University

Canberra
Semester One, 2020.

(Many figures from C. M. Bishop, "Pattern Recognition and Machine Learning")

Outlines

Part V

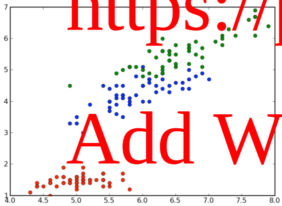
Linear Classification 1

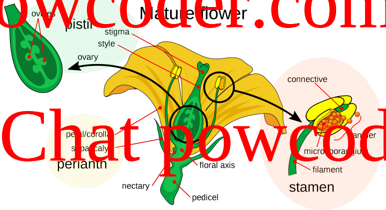
Strategy in this course

- Estimate best predictor = training = learning
  Given data $(x_1, y_1), \ldots, (x_n, y_n)$, find a predictor $f_{\mathbf{w}}(\cdot)$.
  1. Identify the type of input x and output y data
  2. Propose a (linear) mathematical model for $f_{\mathbf{w}}$
  3. Design an objective function or likelihood
  4. Calculate the optimal parameter $(\mathbf{w})$
  5. Model uncertainty using the Bayesian approach
  6. Implement and compute (the algorithm in python)
  7. Interpret and diagnose results

Classification

- Goal : Given input data $\mathbf{x}$, assign it to one of K discrete classes $\mathcal{C}_k$ where $k = 1, \dots, K$.
- Divide the input space into different regions.
- Equivalently: map each point to a categorical label.





Length of petal [in cm] vs sepal [cm] for three types of flowers (Iris Setosa, Iris Versicolor, Iris Virginica).

How to represent binary class labels?

- Class labels are no longer real values as in regression, but a discrete set.
- Two classes : $t \in \{0, 1\}$
  ($t = 1$ represents class $\mathcal{C}_1$ and $t = 0$ represents class $\mathcal{C}_2$)
- Can interpret the value of t as the probability of class $\mathcal{C}_1$, with only two values possible for the probability, 0 or 1.
- Note: Other conventions to map classes into integers possible, check the setup.

How to represent multi-class labels?

- If there are more than two classes ($K > 2$), we call it a multi-class setup.
- Often used: 1-of-K coding scheme in which $\mathbf{t}$ is a vector of length K which has all values 0 except for $t_j = 1$, where j comes from the membership in class $\mathcal{C}_j$ to encode.

- Example: Given 5 classes, $\{\mathcal{C}_1, \dots, \mathcal{C}_5\}$. Membership in class $\mathcal{C}_2$ will be encoded as the target vector

$$\mathbf{t} = (0, 1, 0, 0, 0)^T$$

- Note: Other conventions to map multi-classes into integers possible, check the setup.

Linear Model

- Idea: Use again a Linear Model as in regression: $y(\mathbf{x}, \mathbf{w})$ is a linear function of the parameters $\mathbf{w}$

$$y(\mathbf{x}_n, \mathbf{w}) = \mathbf{w}^\top \phi(\mathbf{x}_n)$$

- But generally $y(\mathbf{x}_n, \mathbf{w}) \in \mathbb{R}$.
  Example: Which class is $y(\mathbf{x}, \mathbf{w}) = 0.71623$?

Generalised Linear Model

- Apply a mapping $f : \mathbb{R} \to \mathbb{Z}$ to the linear model to get the discrete class labels.
- Generalised Linear Model

$$y(\mathbf{x}_n, \mathbf{w}) = f(\mathbf{w}^\top \phi(\mathbf{x}_n))$$

- Activation function: $f(\cdot)$
- Link function : $f^{-1}(\cdot)$

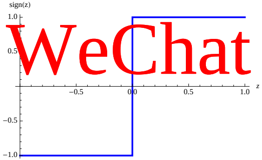


Figure: Example of an activation function $f(z) = \text{sign}(z)$.

Three Models for Decision Problems

- Find a discriminant function $f(\mathbf{x})$ which maps each input directly onto a class label
- Discriminative Models
  1. Solve the inference problem of determining the posterior class probabilities $p(\mathcal{C}_k \mid \mathbf{x})$.
  2. Use decision theory to assign each new $\mathbf{x}$ to one of the classes.

- Generative Models
  1. Solve the inference problem of determining the class-conditional probabilities $p(\mathbf{x} \mid \mathcal{C}_k)$.
  2. Also, infer the prior class probabilities $p(\mathcal{C}_k)$.
  3. Use Bayes' theorem to find the posterior $p(\mathcal{C}_k \mid \mathbf{x})$.
  4. Alternatively, model the joint distribution $p(\mathbf{x}, \mathcal{C}_k)$ directly.
  5. Use decision theory to assign each new $\mathbf{x}$ to one of the classes.

Two Classes

Definition

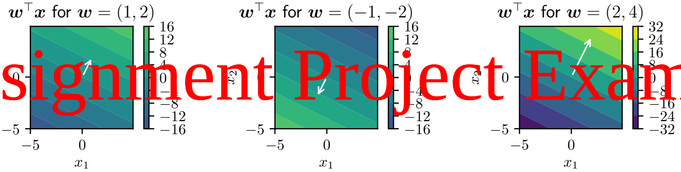
A discriminant is a function that maps from an input vector $\mathbf{x}$ to one of K classes, denoted by $\mathcal{C}_k$.

- Consider first two classes ($K = 2$).
- Construct a linear function of the inputs $\mathbf{x}$

$$y(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + w_0$$

  such that $\mathbf{x}$ being assigned to class $\mathcal{C}_1$ if $y(\mathbf{x}) \geq 0$, and to class $\mathcal{C}_2$ otherwise.
- weight vector $\mathbf{w}$
- bias w_0 (sometimes $-w_0$ called threshold)

Linear Functions



- Gradient = direction of steepest ascent = $\nabla_{\mathbf{x}} \mathbf{w}^\top \mathbf{x} = \mathbf{w}^\top$.
- The set $\mathbf{w}^\top \mathbf{x} + w_0 = 0$ is a hyper-plane.
- Projecting $\mathbf{x}$ on that hyper-plane means finding $\arg\min_{\mathbf{x}_\perp} \|\mathbf{x} - \mathbf{x}_\perp\|$ subject to the constraint $\mathbf{w}^\top \mathbf{x}_\perp + w_0 = 0$. Geometrically: move in the direction $\frac{\mathbf{w}}{\|\mathbf{w}\|}$.
- Rate of change of function value in that direction is $\frac{\mathrm{d}}{\mathrm{d}a}\left(a \frac{\mathbf{w}}{\|\mathbf{w}\|}\right)^\top \mathbf{w} = a\|\mathbf{w}\|$.
- The length $\left\| a \frac{\mathbf{w}}{\|\mathbf{w}\|} \right\| = \frac{a}{\|\mathbf{w}\|} \|\mathbf{w}\| = a$.
- For a fixed change in $\mathbf{w}^\top \left(a \frac{\mathbf{w}}{\|\mathbf{w}\|}\right)$, $a \propto \frac{1}{\|\mathbf{w}\|}$.

Two Classes

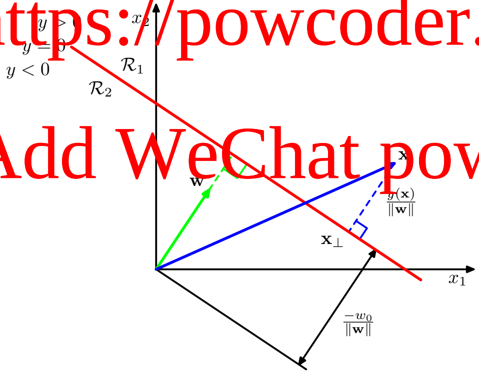
- Decision boundary $y(\mathbf{x}) = 0$ is a $(D-1)$-dimensional hyperplane in a D-dimensional input space (decision surface).
- $\mathbf{w}$ is orthogonal to any vector lying in the decision surface.
- Proof: Assume $\mathbf{x}_A$ and $\mathbf{x}_B$ are two points lying in the decision surface. Then,

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

Two Classes

- $y(\mathbf{x})$ gives a **signed** measure of the perpendicular distance r **from** the decision surface **to** $\mathbf{x}$, that is $r = y(\mathbf{x})/\|w\|$.

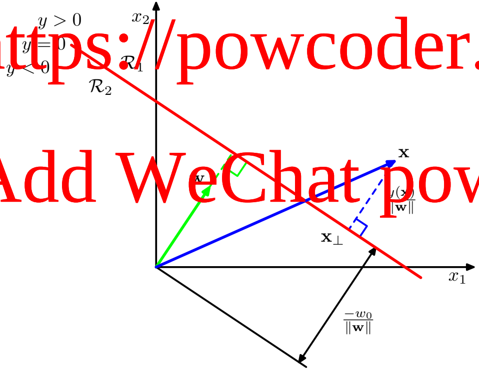
$$y(\mathbf{x}) = \mathbf{w}^T \overbrace{\left(\mathbf{x}_\perp + r\frac{\mathbf{w}}{\|w\|}\right)}^{\mathbf{x}} + w_0 = r\frac{\mathbf{w}^\top \mathbf{w}}{\|w\|} + \overbrace{\mathbf{w}^\top \mathbf{x}_\perp + w_0}^{0} = r\|w\|$$



Two Classes

- The normal distance from the origin to the decision surface is therefore

$$-\frac{y(\mathbf{0})}{\|w\|} = -\frac{w_0}{\|w\|}$$



Two Classes

- More compact notation : Add an extra dimension to the input space and set the value to $x_0 = 1$.
- Also define $\widetilde{\mathbf{w}} = (w_0, \mathbf{w})$ and $\widetilde{\mathbf{x}} = (1, \mathbf{x})$

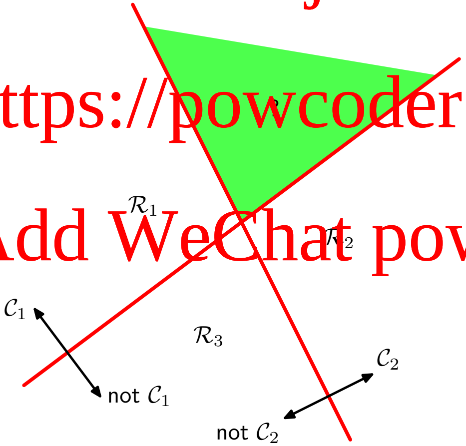
$$y(\mathbf{x}) = \widetilde{\mathbf{w}}^\top \widetilde{\mathbf{x}}$$

(if it helps, you may think of $\widetilde{\mathbf{w}}^\top$ as a function).

- Decision surface is now a D-dimensional hyperplane in a $D+1$-dimensional expanded input space.

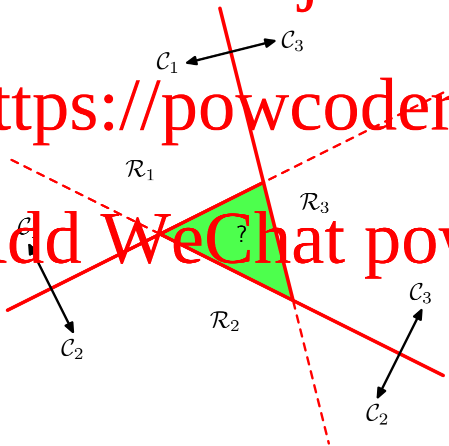
Multi-Class

- Number of classes $K > 2$
- Can we combine a number of two-class discriminant functions using $K - 1$ one-versus-the-rest classifiers?



Multi-Class

- Number of classes $K > 2$
- Can we combine a number of two-class discriminant functions using $K(K-1)/2$ one-versus-one classifiers?

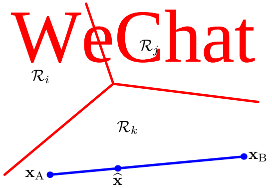


Multi-Class

- Number of classes $K > 2$
- Solution: Use K linear functions

$$y_k(\mathbf{x}) = \mathbf{w}_k^\top \mathbf{x} + w_{k0}$$

- Assign input $\mathbf{x}$ to class $\mathcal{C}_k$ if $y_k(\mathbf{x}) > y_j(\mathbf{x})$ for all $j \neq k$.
- Decision boundary between class $\mathcal{C}_k$ and $\mathcal{C}_j$ given by $y_k(\mathbf{x}) = y_j(\mathbf{x})$



Least Squares for Classification

- Regression with a linear function of the model parameters and minimisation of sum-of-squares error function resulted in a closed-from solution for the parameter values.
- Is this also possible for classification?
- Given input data $\mathbf{x}$ belonging to one of K classes $\mathcal{C}_k$.
- Use 1-of-K binary coding scheme.
- Each class is described by its own linear model

$$y_k(\mathbf{x}) = \mathbf{w}_k^\top \mathbf{x} + w_{k0} \qquad k = 1, \dots, K$$

- With the conventions

$$\widetilde{\mathbf{w}}_k = \begin{bmatrix} w_{k0} \\ \mathbf{w}_k \end{bmatrix} \in \mathbb{R}^{D+1}$$

$$\widetilde{\mathbf{x}} = \begin{bmatrix} 1 \\ \mathbf{x} \end{bmatrix} \in \mathbb{R}^{D+1}$$

$$\widetilde{\mathbf{W}} = \begin{bmatrix} \widetilde{\mathbf{w}}_1 & \dots & \widetilde{\mathbf{w}}_K \end{bmatrix} \in \mathbb{R}^{(D+1)\times K}$$

- we get for the (vector valued) discriminant function

$$\mathbf{y}(\mathbf{x}) = \widetilde{\mathbf{W}}^\top \widetilde{\mathbf{x}} \in \mathbb{R}^K$$

(if it helps, you may think of $\widetilde{\mathbf{W}}^\top$ as a **vector-valued** function).

- For a new input $\mathbf{x}$, the class is then defined by the index of the largest value in the row vector $\mathbf{y}(\mathbf{x})$

Determine $\widetilde{\mathbf{W}}$

- Given a training set $\{\mathbf{x}_n, \mathbf{t}_n\}$ where $n = 1, \dots, N$, and $\mathbf{t}_n$ is the class in the 1-of-K coding scheme.
- Define a matrix $\mathbf{T}$ where row n corresponds to $\mathbf{t}_n^\top$.
- The sum-of-squares error can now be written as (check that $\operatorname{tr}\left\{A^\top A\right\} = \|A\|_F^2$)

$$E_D(\widetilde{\mathbf{W}}) = \frac{1}{2} \operatorname{tr}\left\{(\widetilde{\mathbf{X}}\widetilde{\mathbf{W}} - \mathbf{T})^T (\widetilde{\mathbf{X}}\widetilde{\mathbf{W}} - \mathbf{T})\right\}$$

- The minimum of $E_D(\widetilde{\mathbf{W}})$ will be reached for

$$\widetilde{\mathbf{W}} = (\widetilde{\mathbf{X}}^\top \widetilde{\mathbf{X}})^{-1} \widetilde{\mathbf{X}}^\top \mathbf{T} = \widetilde{\mathbf{X}}^\dagger \mathbf{T}$$

where $\widetilde{\mathbf{X}}^\dagger$ is the pseudo-inverse of $\widetilde{\mathbf{X}}$.

- The discriminant function $\mathbf{y}(\mathbf{x})$ is therefore

$$\mathbf{y}(\mathbf{x}) = \widetilde{\mathbf{W}}^\top \widetilde{\mathbf{x}} = \mathbf{T}^\top (\widetilde{\mathbf{X}}^\dagger)^\top \widetilde{\mathbf{x}},$$

where $\widetilde{\mathbf{X}}$ is given by the training data, and $\widetilde{\mathbf{x}}$ is the new input.

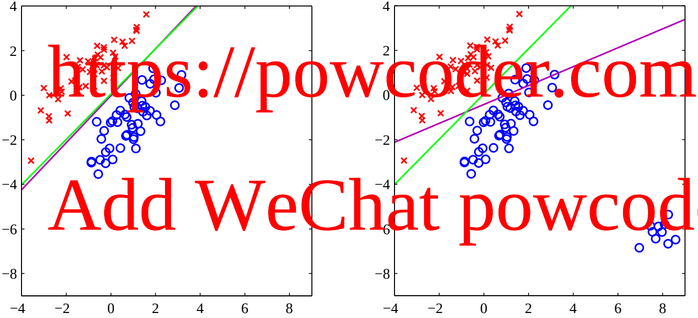
- Interesting property: If for every $\mathbf{t}_n$ the same linear constraint $\mathbf{a}^\top \mathbf{t}_n + b = 0$ holds, then the prediction $\mathbf{y}(\mathbf{x})$ will also obey the same constraint

$$\mathbf{a}^\top \mathbf{y}(\mathbf{x}) + b = 0.$$

- For the 1-of-K coding scheme, the sum of all components in $\mathbf{t}_n$ is one, and therefore all components of $\mathbf{y}(\mathbf{x})$ will sum to one. BUT: the components are not probabilities, as they are not constraint to the interval $(0, 1)$.

Deficiencies of the Least Squares Approach

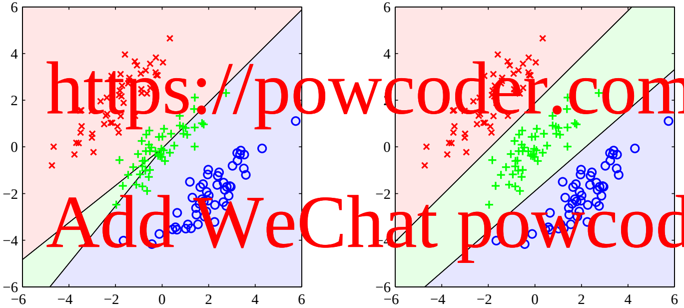
Magenta curve :
Decision Boundary for the least squares approach
Green curve :
Decision boundary for the logistic regression (described later)



(Imagine heat-maps of the quadratic penalty function, similarly to those of the linear functions earlier in the slides.)

Left plot : Decision Boundary for least squares

Right plot : Boundaries for logistic regression (described later)



- View linear classification as dimensionality reduction.

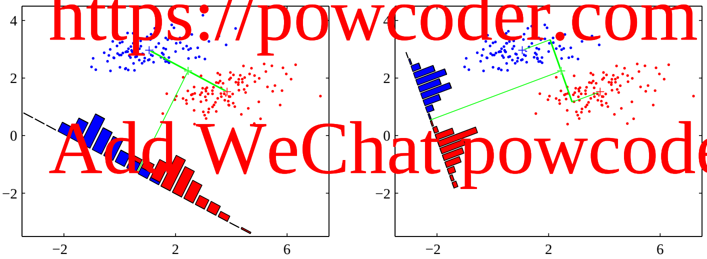
$$y(\mathbf{x}) = \mathbf{w}^\top \mathbf{x}$$

If $y \geq -w_0$ then class $\mathcal{C}_1$, otherwise $\mathcal{C}_2$.

- But there are many projections from a D-dimensional input space onto one dimension.
- Projection always means loss of information.
- For classification we want to preserve the class separation in one dimension.
- Can we find a projection which maximally preserves the class separation ?

Fisher's Linear Discriminant

Samples from two classes in a two-dimensional input space and their histogram when projected to two different one-dimensional spaces.



Fisher's Linear Discriminant - First Try

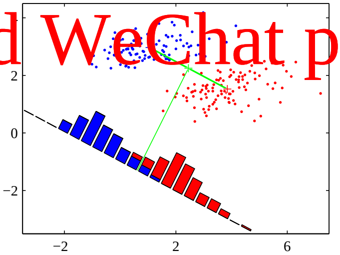
- Given N_1 input data of class $\mathcal{C}_1$, and N_2 input data of class $\mathcal{C}_2$, calculate the centres of the two classes

$$\mathbf{m}_1 = \frac{1}{N_1} \sum_{n \in \mathcal{C}_1} \mathbf{x}_n, \qquad \mathbf{m}_2 = \frac{1}{N_2} \sum_{n \in \mathcal{C}_2} \mathbf{x}_n$$

- Choose $\mathbf{w}$ so as to maximise the separation of the projected class means

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

- Problem with non-uniform covariance



Fisher's Linear Discriminant

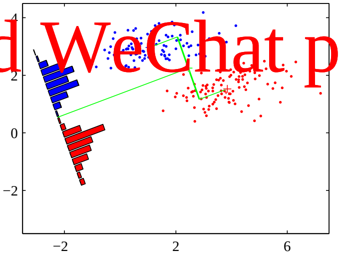
- Measure also the within-class variance for each class

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

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

- Maximise the Fisher criterion

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



Primer: Bilinear form with a Covariance Matrix

Let

$$\begin{aligned}\boldsymbol{\mu} &= \mathbb{E}[\mathbf{x}] \\ \Sigma &= \operatorname{cov}[\mathbf{x}] \\ &= \mathbb{E}[(\mathbf{x}-\boldsymbol{\mu})(\mathbf{x}-\boldsymbol{\mu})^\top].\end{aligned}$$

Then

$$\begin{aligned}\operatorname{var}\left[\mathbf{w}^\top\mathbf{x}\right] &= \mathbb{E}\left[\left(\mathbf{w}^\top\mathbf{x}-\mathbb{E}\left[\mathbf{w}^\top\mathbf{x}\right]\right)^2\right] \\ &= \mathbb{E}\left[\left(\mathbf{w}^\top\mathbf{x}-\mathbf{w}^\top\mathbb{E}\left[\mathbf{x}\right]\right)^2\right] \\ &= \mathbb{E}\left[\left(\mathbf{w}^\top\mathbf{x}-\mathbf{w}^\top\boldsymbol{\mu}\right)^2\right] \\ &= \mathbb{E}\left[\left(\mathbf{w}^\top\mathbf{x}-\mathbf{w}^\top\boldsymbol{\mu}\right)\left(\mathbf{w}^\top\mathbf{x}-\mathbf{w}^\top\boldsymbol{\mu}\right)\right] \\ &= \mathbb{E}\left[\left(\mathbf{w}^\top\mathbf{x}-\mathbf{w}^\top\boldsymbol{\mu}\right)\left(\mathbf{x}^\top\mathbf{w}-\boldsymbol{\mu}^\top\mathbf{w}\right)\right] \\ &= \mathbb{E}\left[\mathbf{w}^\top(\mathbf{x}-\boldsymbol{\mu})(\mathbf{x}-\boldsymbol{\mu})^\top\mathbf{w}\right] \\ &= \mathbf{w}^\top\mathbb{E}\left[(\mathbf{x}-\boldsymbol{\mu})(\mathbf{x}-\boldsymbol{\mu})^\top\right]\mathbf{w} \\ &= \mathbf{w}^\top\Sigma\mathbf{w}.\end{aligned}$$

Fisher's Linear Discriminant

- The Fisher criterion can be rewritten as

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

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

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

  so by the previous slide, the numerator of $J(\mathbf{w})$ is:
  the variance of the projection of the means

- $\mathbf{S}_W$ is the within-class covariance

$$\mathbf{S}_W = \sum_{n \in \mathcal{C}_1} (\mathbf{x}_n - \mathbf{m}_1)(\mathbf{x}_n - \mathbf{m}_1)^T + \sum_{n \in \mathcal{C}_2} (\mathbf{x}_n - \mathbf{m}_2)(\mathbf{x}_n - \mathbf{m}_2)^T$$

  so so by the previous slide and
  $\mathbf{w}^\top (A + B)\mathbf{w} = \mathbf{w}^\top A \mathbf{w} + \mathbf{w}^\top B \mathbf{w}$, the denominator of $J(\mathbf{w})$ is:
  (the variance of the projection of the points in class $\mathcal{C}_1$) +
  (the variance of the projection of the points in class $\mathcal{C}_2$)

- The Fisher criterion

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

has a maximum for Fisher's linear discriminant

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

- Fisher's linear discriminant is NOT a discriminant, but can be used to construct one by choosing a threshold y_0 in the projection space.

Fisher's Discriminant For Multi-Class

- Assume that the dimensionality of the input space D is greater than the number of classes K.
- Use $D' > 1$ linear 'features' $y_k = \mathbf{w}_k^\top \mathbf{x}$ and write everything in vector form (with no bias term)

$$\mathbf{y} = \mathbf{W}^\top \mathbf{x}.$$

- The within-class covariance is then the sum of the covariances for all K classes

$$\mathbf{S}_W = \sum_{k=1}^{K} \mathbf{S}_k$$

where

$$\mathbf{S}_k = \sum_{n \in \mathcal{C}_k} (\mathbf{x}_n - \mathbf{m}_k)(\mathbf{x}_n - \mathbf{m}_k)^\top$$

$$\mathbf{m}_k = \frac{1}{N_k} \sum_{n \in \mathcal{C}_k} \mathbf{x}_n$$

Fisher's Discriminant For Multi-Class

- Between-class covariance

$$\mathbf{S}_B = \sum_{k=1}^{K} N_k (\mathbf{m_k} - \mathbf{m})(\mathbf{m_k} - \mathbf{m})^T.$$

where $\mathbf{m}$ is the total mean of the input data

$$\mathbf{m} = \frac{1}{N} \sum_{n=1}^{N} \mathbf{x}_n.$$

- One possible way to define a function of $\mathbf{W}$ which is large when the between-class covariance is large and the within-class covariance is small is given by

$$J(\mathbf{W}) = \operatorname{tr}\left\{ (\mathbf{W}^\top \mathbf{S}_W \mathbf{W})^{-1} (\mathbf{W}^\top \mathbf{S}_B \mathbf{W}) \right\}$$

- The maximum of $J(\mathbf{W})$ is determined by the D' eigenvectors of $\mathbf{S}_W^{-1}\mathbf{S}_B$ with the largest eigenvalues.

The Perceptron Algorithm

- Frank Rosenblatt (1928 - 1969)
- "Principles of neurodynamics: Perceptrons and the theory of brain mechanisms" (Spartan Books, 1962)

- Perceptron ("MARK 1") was the first computer which could learn new skills by trial and error

The Perceptron Algorithm

- Two class model
- Create feature vector $\phi(\mathbf{x})$ by a fixed nonlinear transformation of the input $\mathbf{x}$.
- Generalised linear model

$$y(\mathbf{x}) = f(\mathbf{w}^\top \phi(\mathbf{x}))$$

with $\phi(\mathbf{x})$ containing some bias element $\phi_0(\mathbf{x}) = 1$.

- nonlinear activation function

$$f(a) = \begin{cases} +1, & a \geq 0 \\ -1, & a < 0 \end{cases}$$

- Target coding for perceptron

$$t = \begin{cases} +1, & \text{if } \mathcal{C}_1 \\ -1, & \text{if } \mathcal{C}_2 \end{cases}$$

The Perceptron Algorithm - Error Function

- Idea : Minimise total number of misclassified patterns.
- Problem : As a function of $\mathbf{w}$, this is piecewise constant and therefore the gradient is zero almost everywhere.
- Better idea: Using the $(-1, +1)$ target coding scheme, we want all patterns to satisfy $\mathbf{w}^T \phi(\mathbf{x}_n) t_n > 0$.

- Perceptron Criterion : Add the errors for all patterns belonging to the set of misclassified patterns $\mathcal{M}$

$$E_P(\mathbf{w}) = - \sum_{n \in \mathcal{M}} \mathbf{w}^T \phi(\mathbf{x}_n) t_n$$

Perceptron - Stochastic Gradient Descent

- Perceptron Criterion (with notation $\phi_n = \phi(\mathbf{x}_n)$)

$$E_P(\mathbf{w}) = -\sum_{n \in \mathcal{M}} \underbrace{\mathbf{w}^\top \boldsymbol{\phi}_n t_n}_{\equiv E_P^{(n)}(\mathbf{w})}$$

- One iteration at step τ
  1. Choose a training data point index n (uniformly at random or by cycling though the data)
  2. Update the weight vector $\mathbf{w}$ by

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} - \eta \nabla E_P^{(n)}(\mathbf{w})$$

where

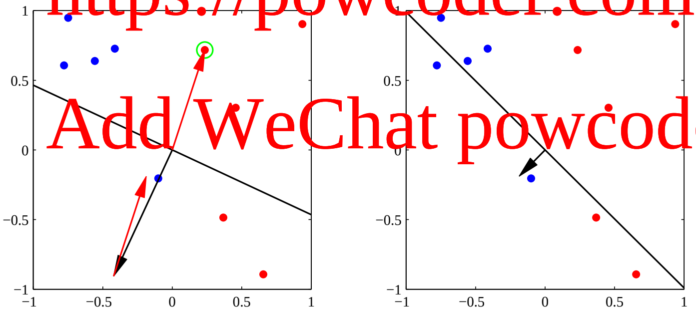
$$\nabla E_P^{(n)}(\mathbf{w}) = \begin{cases} -\boldsymbol{\phi}_n t_n & \text{if } \left(\mathbf{w}^{(\tau)\top} \boldsymbol{\phi}(\mathbf{x}_n) \cdot t_n\right) \leq 0 \\ 0 & \text{otherwise.} \end{cases}$$

- As $y(\mathbf{x}, \mathbf{w})$ is invariant to the norm of $\mathbf{w}$, we may set $\eta = 1$.

The Perceptron Algorithm - Update 1

Update of the perceptron weights from a misclassified pattern (green)

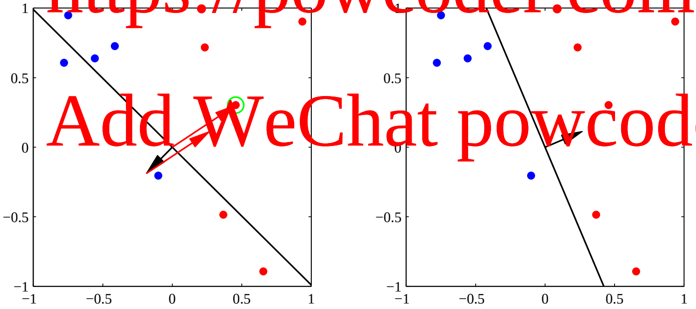
$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} + \phi_n t_n$$



The Perceptron Algorithm - Update 2

Update of the perceptron weights from a misclassified pattern (green)

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} + \phi_n t_n$$



The Perceptron Algorithm - Convergence

- Does the algorithm converge ?
- For a single update step, letting $\eta = 1$, and considering the error from a single point,

$$-\mathbf{w}^{(\tau+1)T}\phi_n t_n = -\mathbf{w}^{(\tau)T}\phi_n t_n - (\phi_n t_n)^\top \phi_n t_n < -\mathbf{w}^{(\tau)T}\phi_n t_n$$

  because $(\phi_n t_n)^\top \phi_n t_n = \|\phi_n t_n\| > 0$. In other words, gradient descent on a linear function decreases that function.
- BUT: contributions to the error from the other misclassified patterns might have increased.
- AND: some correctly classified patterns might now be misclassified.
- Perceptron Convergence Theorem : If the training set is linearly separable, the perceptron algorithm is guaranteed to find a solution in a finite number of steps.