Linear Models for Classification
Discriminant Functions

Sargur N. Srihari
University at Buffalo, State University of New York
USA

Topics

- Linear Discriminant Functions
  - Definition (2-class), Geometry
  - Generalization to $K > 2$ classes
- Methods to learn parameters
  1. Least Squares Classification
  2. Fisher's Linear Discriminant
  3. Perceptrons

Discriminant Functions

- A discriminant function assigns input vector $\mathbf{x}$ to one of K classes denoted by C_k
- We restrict attention to linear discriminants
  - i.e., Decision surfaces are hyperplanes
- First consider $K = 2$, and then extend to $K > 2$

Geometry of Linear Discriminant Functions:

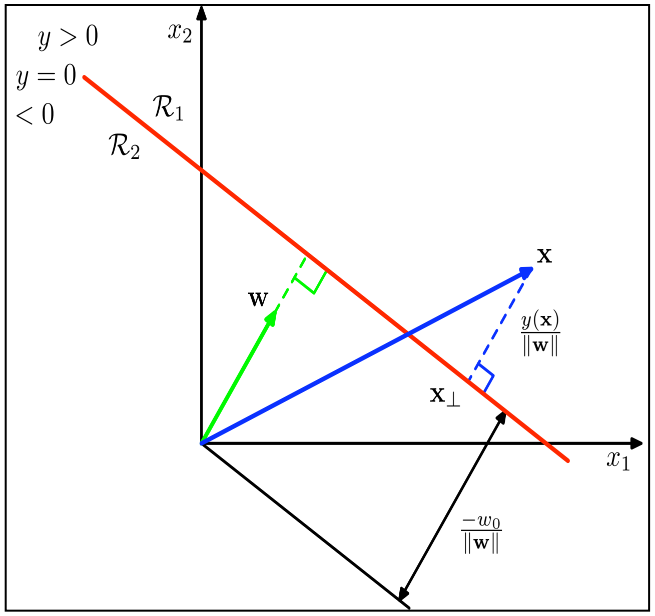
- Two-class linear discriminant function:

  $$y(\mathrm{x}) = \mathrm{w}^{\mathrm{T}}\,\mathrm{x} + w_0$$

  A linear function of input vector

  w is weight vector and w_0 is *bias*

  Negative of bias sometimes called *threshold*



- Assign x to C_1 if $y(\mathrm{x}) \geq 0$ else C_2
  - Defines decision boundary $y(\mathrm{x}) = 0$
    - It corresponds to a $(D\text{-}1)$- dimensional hyperplane in a D-dimensional input space

Distance of Origin to Surface is w_0

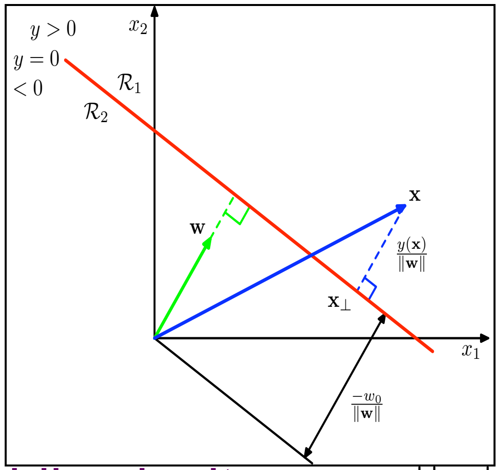
Let x_A and x_B be points on surface $y(\mathrm{x})=\mathrm{w}^T\mathrm{x}+w_0=0$

- Because $y(\mathrm{x}_A)=y(\mathrm{x}_B)=0$, we have $\mathrm{w}^T(\mathrm{x}_A-\mathrm{x}_B)=0$,
- Thus w is orthogonal to every vector lying on decision surface
- So w determines orientation of the decision surface

– If x is a point on surface then $y(\mathrm{x})=0$ or $\mathrm{w}^T\mathrm{x} = -w_0$

- Normalized distance from origin to surface:

$$\boxed{\begin{array}{l}\dfrac{\mathrm{w}^T\mathrm{x}}{||\,\mathrm{w}\,||}=-\dfrac{w_0}{||\,\mathrm{w}\,||}\\ \text{where } ||\mathrm{w}|| \text{ is the norm defined as}\\ ||\,\mathrm{w}\,||^2=\mathrm{w}^T\mathrm{w}=w_1^2+..+w_{M-1}^2\end{array}}$$



– Where elements of w are normalized by dividing by its norm $||\mathrm{w}||$

» By definition of a normalized vector which has length *1*

– w_0 sets distance of origin to surface

Distance of arbitrary point x to surface

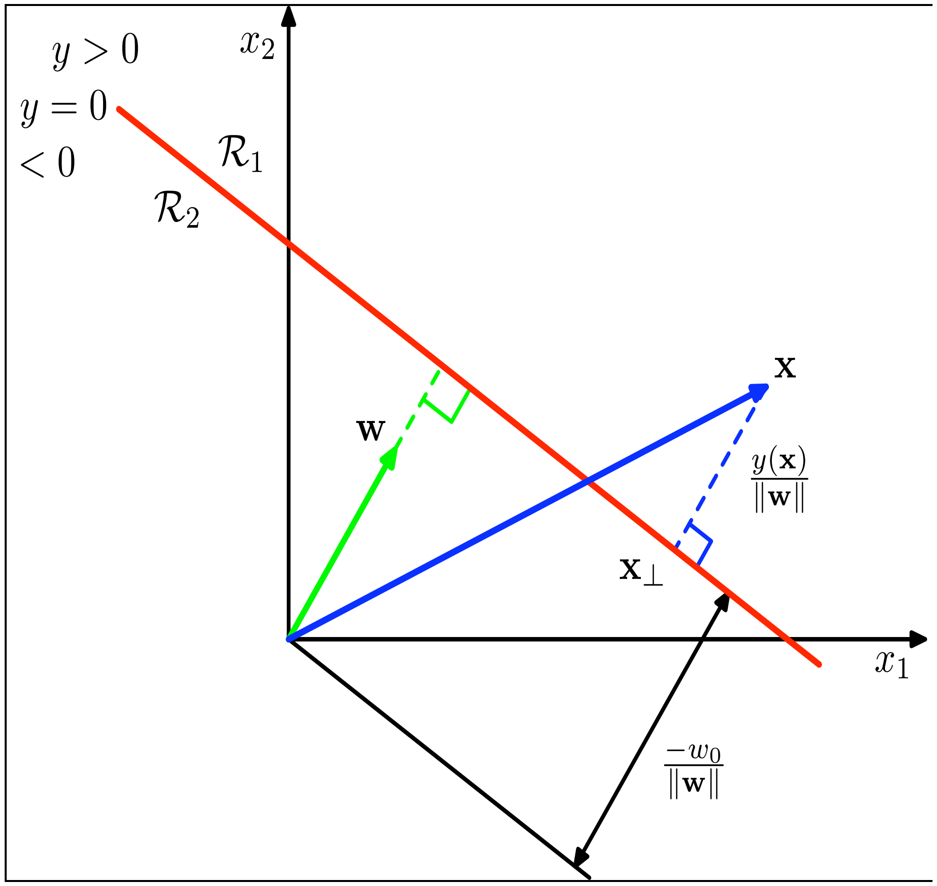
Let x be an arbitrary point

- We can show that $y(\mathrm{x})$ gives signed measure of perpendicular distance r from x to surface as follows:
  - If x_p is orthogonal projection of x to surface then

$$x = x_p + r\frac{\mathrm{w}}{||\,\mathrm{w}\,||} \text{ by vector addition}$$

Second term is a vector normal to surface.
This vector is parallel to w
which is normalized by length ||w|| .
Since a normalized vector has length 1
we need to scale by r.
From which we can get

$$r = \frac{y(\mathrm{x})}{||\,\mathrm{w}\,||}$$



Augmented vector

- With dummy input $x_0=1$
- and w $=(w_0,\mathrm{w})$ then $y(\mathrm{x}) = \mathbf{w}^{\mathrm{T}}\mathrm{x}$
  - passes through origin in *augmented* $D+1$ dimensional space

Extension to Multiple Classes

- Two approaches:
  - Using several two-class classifiers
    - But leads to serious difficulties
  - Use K linear functions

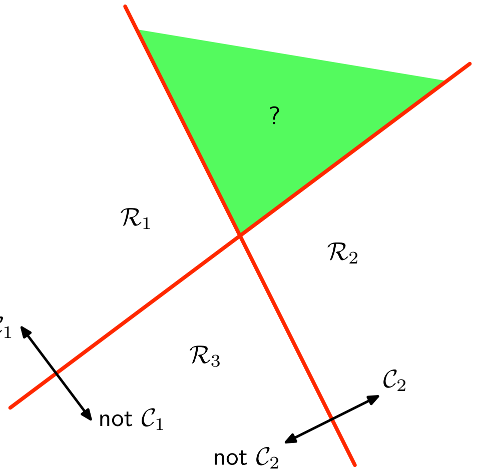
Multiple Classes with 2-class classifiers

- By using several 2-class classifiers

<u>One-versus-the-rest</u>

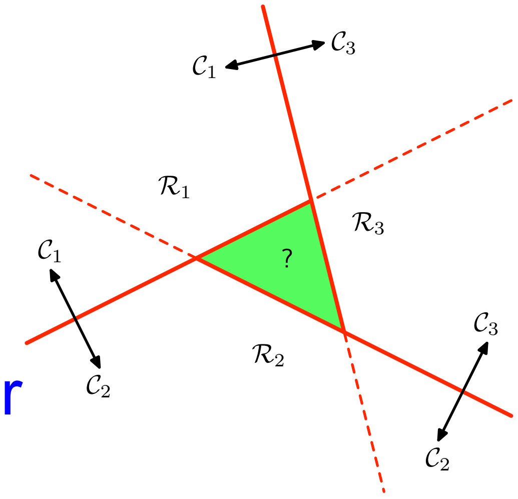
Build a K class discriminant

Use $K-1$ classifiers, each solve a two-class problem



<u>One-versus-one</u>

Alternative is $K(K-1)/2$ binary discriminant functions, one for every pair



Both result in ambiguous regions of input space

Multiple Classes with K discriminants

- Consider a single K class discriminant of the form

  $$y_k(\mathrm{x}) = \mathrm{w}^{\mathrm{T}}{}_k\, \mathrm{x} + w_{k0}$$
- Assign a point x to class C_k if $y_k(\mathrm{x}) > y_j\,(\mathrm{x})$ for all $j \neq k$

- Decision boundary between class C_k and C_{j} is given by $y_k(\mathrm{x}) = y_j\,(\mathrm{x})$
  - This corresponds to $D - 1$ dimensional hyperplane defined by
  - $(\mathrm{w}_{\mathrm{k}} - \mathrm{w}_{\mathrm{j}}\,)^{\mathrm{T}}\,\mathrm{x} + (w_{k0} - w_{\mathrm{j}0}\,) = 0$
  - Same form as the decision boundary for 2-class case $\mathrm{w}^{\mathrm{T}}\mathrm{x} + w_0 = 0$
- Decision regions of such a discriminant are always singly connected and convex
  - Proof follows

Convexity of Decision Regions (Proof)

Consider two points x_A and x_B both in decision region $\mathcal{R}_k$

Any point $\hat{x}$ on line connecting x_A and x_B can be expressed as

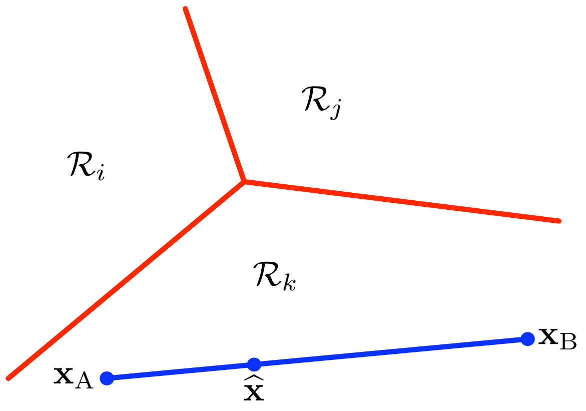
$$\hat{x} = \lambda x_A + (1-\lambda) x_B$$ where $0 \le l \le 1$.

From linearity of discriminant functions

$y_k(x) = w^T_k\, x + w_{k0}$

Combining the two, we have

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



Because 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 $\hat{x}$ also lies inside R_k

Thus R_k is singly-connected and convex
(single straight line connects any two points in region)

Learning the Parameters of Linear Discriminant Functions

- Three Methods
  - Least Squares
  - Fisher's Linear Discriminant
  - Perceptrons
- Each is simple but several disadvantages

Least Squares for Classification

- Analogous to regression: simple closed- form solution exists for parameters
- Each C_k , $k=1,..K$ is described by its own linear model

$$y_k(\mathrm{x}) = \mathrm{w}^{\mathrm{T}}{}_{k}\, \mathrm{x} + w_{k0}$$

- Create augmented vector $\mathbf{x}=(1,\mathrm{x}^{\mathrm{T}})$ and $\mathbf{w}_{\mathrm{k}}=(w_{k0},\mathrm{w}_{\mathrm{k}}{}^{\mathrm{T}})$
- Grouping into vector notation $\mathrm{y}(\mathrm{x}) = \mathbf{W}^{\mathrm{T}}\, \mathbf{x}$

  $\mathbf{W}$ is the *parameter matrix* whose k^{th} column is a $D+1$ dimensional vector (including bias)

- New input vector x is assigned to class for which output $y_k = \mathbf{w}^{\mathrm{T}}{}_{k}\, \mathbf{x}$ is largest
- Determine $\mathbf{W}$ by minimizing squared error

Parameters using Least Squares

- Training data $\{\mathrm{x}_n, \mathrm{t}_n\}, n = 1, \ldots, N$

  t_n is a column vector of K dimensions using $1\text{-}\mathrm{of}-K$ form

- Define matrices

  $\mathrm{T} \equiv n^{\mathrm{th}}$ row is the vector $\mathrm{t}^{\mathrm{T}}_n$

  $\mathbf{X} \equiv n^{\mathrm{th}}$ row of which is $\mathbf{x}_n{}^{\mathrm{T}}$

- Sum of squares error function

$$E_{\mathrm{D}}(\mathbf{W}) = ½\ \mathrm{Tr}\ \{(\mathbf{XW} - \mathrm{T})^{\mathrm{T}}\ (\mathbf{XW} - \mathrm{T})\}$$

Notes:

$(\mathrm{XW}\text{-}\mathrm{T})$ is error vector, whose square is a diagonal matrix

Trace is the sum of diagonal elements

Minimizing Sum of Squares

- Sum of squares error function

$$E_D(\mathbf{W}) = ½ \,\mathrm{Tr}\, \{(\mathbf{XW} - \mathrm{T})^{\mathrm{T}} (\mathbf{XW} - \mathrm{T})\}$$

- Set derivative w.r.t. **W** to zero, gives solution

$$\mathbf{W} = (\mathbf{X}^{\mathrm{T}} \mathbf{X})^{-1} \mathbf{X}^{\mathrm{T}}\mathrm{T} \boxed{= \mathbf{X}}^{\dagger}\mathrm{T}$$

where $\mathbf{X}^{\dagger}$ is pseudo-inverse of matrix $\mathbf{X}$

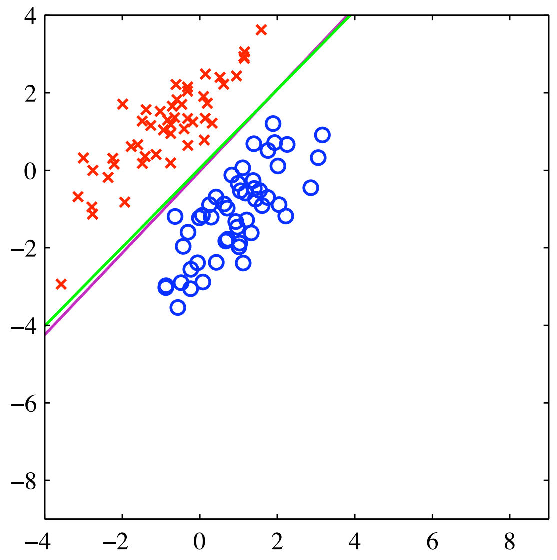
- Discriminant function, after rearranging, is

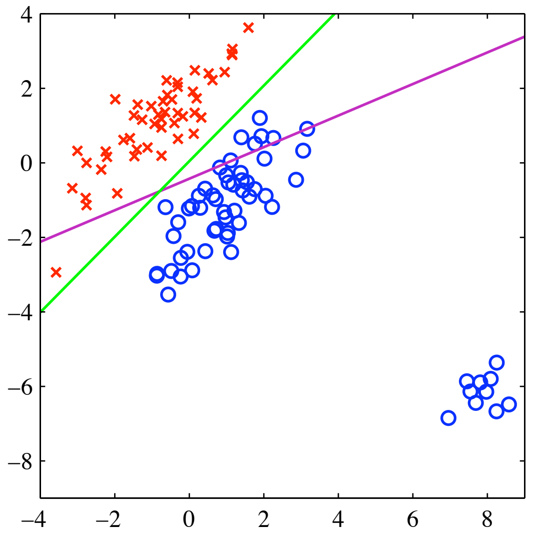
$$\boxed{\mathrm{y}(\mathrm{x}) = \mathbf{W}^{\mathrm{T}} \mathbf{x} = \mathrm{T}^{\mathrm{T}}(\mathbf{X}^{\dagger})^{\mathrm{T}}}\mathbf{x}$$

- An exact closed form solution for **W** using which we can classify x to class k for which y_k is maximum

but has severe limitations

Least Squares is Sensitive to Outliers





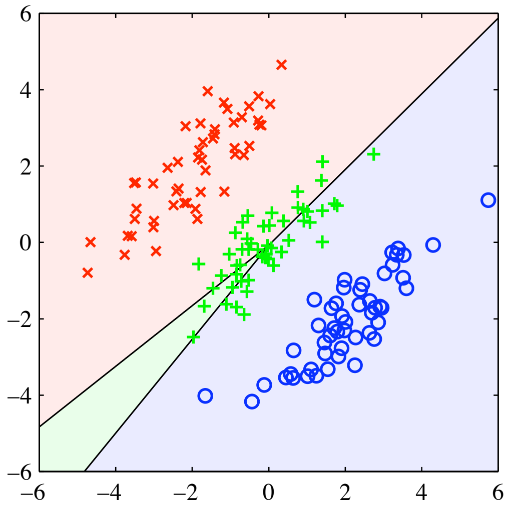
Magenta: Least Squares
Green: Logistic Regression (more robust)

Sum of squared errors penalizes predictions that are “too correct”
Or long way from decision boundary

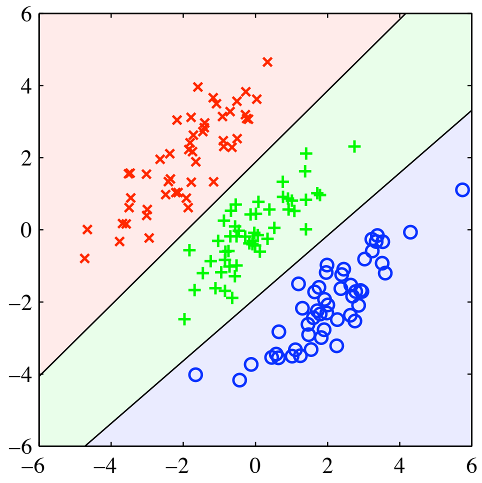
SVMs have an alternate error function (hinge function) that does not have this limitation

Disadvantages of Least Squares

Least Squares



Logistic Regression



Three classes
2-D space

Region assigned to green class is too small, mostly misclassified
Yet linear decision boundaries of logistic regression can give perfect results

- Lack robustness to outliers
- Certain datasets unsuitable for least squares classification
- Decision boundary corresponds to ML solution
  under Gaussian conditional distribution
- But binary target values have a distribution far from Gaussian

4. Fisher Linear Discriminant

- View classification in terms of dimensionality reduction
  - Project D-dimensional input vector x into one dimension using $y = \mathrm{w}^{\mathrm{T}}\,\mathrm{x}$
- Place threshold on y to classify

  $y \geq -w_0$ as C_1 and otherwise C_2

  we get standard linear classifier
- Classes well-separated in D-space may strongly overlap in 1-dimension
  - Adjust component of the weight vector w
  - Select projection to maximize class-separation

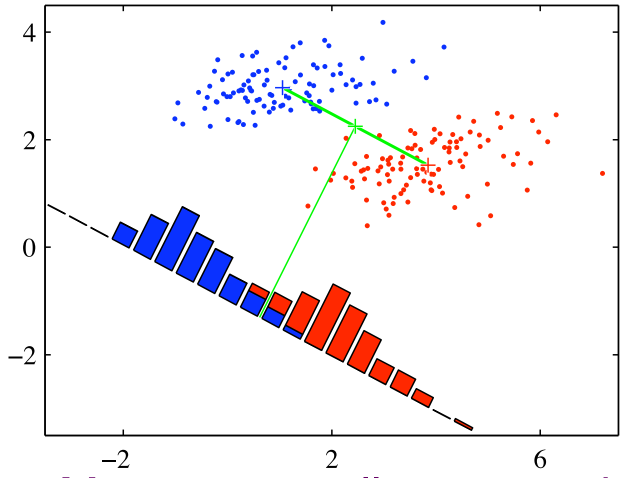
Fisher: Maximizing Mean Separation

- Two class problem:
  - N_1 points of class C_1 and N_2 points of class C_2
- Mean Vectors $\mathrm{m}_1 = \frac{1}{N_1}\sum_{n \varepsilon C_1} \mathrm{x}_n$ $\mathrm{m}_2 = \frac{1}{N_1}\sum_{n \varepsilon C_2} \mathrm{x}_n$
- Choose w to best separate class means
- Maximize $m_2 - m_1 = \mathrm{w}^{\mathrm{T}} (\mathrm{m}_2 - \mathrm{m}_1)$,

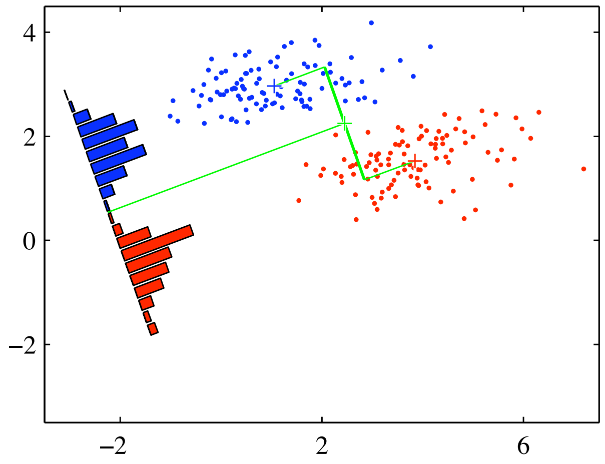
  where $m_{\mathrm{k}} = \mathrm{w}^{\mathrm{T}}\mathrm{m}_{\mathrm{k}}$

  is the mean of projected data of class C_k
- Can be made arbitrarily large by increasing w
  - Introduce Lagrange multiplier to enforce (w to have unit length) $\Sigma_{\mathrm{i}}\, w_i^2 = 1$

Fisher: Minimizing Variance



Means are well-separated but classes overlap



- Maximizing mean separation is insufficient for classes with non-diagonal covariance
- Fisher formulation
  1. Maximize function to separate projected class means
  2. Also give small variance within each class, thereby minimizing the class overlap

Fisher: Derivation

- Within class variance

$$s_k^2 = \Sigma_{n \,\varepsilon\, C_k} (y_n - m_k)^2, \text{ where } y_n = w^T x_n$$

- Total within-class variance is given by $s_1^2 + s_2^2$
- Fisher criterion = $J(w) = (m_2 - m_1) / s_1^2 + s_2^2$

Rewriting $\boxed{J(w) = w^T S_B w \;/\; w^T S_W w}$

where $S_B = (m_2 - m_1)(m_2 - m_1)^T$ and

$$S_W = \Sigma_{n \in C1} (x_n - m_1)(x_n - m_1)^T + \Sigma_{n \in C2} (x_n - m_2)(x_n - m_2)^T$$

- Differentiating wrt w, $J(w)$ is maximized when

$$(w^T S_B w) S_W w = (w^T S_W w) S_B w$$

Dropping scalar factors (in parentheses) & noting S_B is in same direction as $(m_2 - m_1)$ & multiplying by S_W^{-1}

$$\boxed{w \;\alpha\; S_W^{-1}(m_2 - m_1)}$$

Relation to Least Squares

- Least Squares: goal of making Model predictions as close as possible to target values
- Fisher: require maximum class separation
- For two-class problem Fisher is special case of least squares
  - Proof starts with sum-of-square errors and shows that weight vector found coincides with Fisher criterion

Fisher’ s Discriminant for Multiple Classes

- Can be generalized for multiple classes
- Derivation is fairly involved [Fukunaga 1990]

5. Perceptron Algorithm

- Two-class model
  - Input vector x transformed by a fixed nonlinear transformation to give feature vector $\phi(\mathrm{x})$

$$y(\mathrm{x}) = f\left(\mathrm{w}^{\mathrm{T}}\phi(\mathrm{x})\right)$$

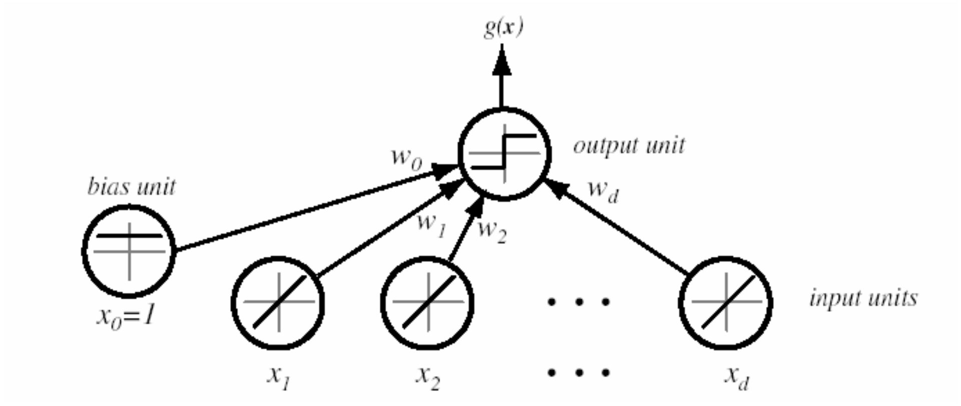
where non-linear activation $f(.)$ is a step function

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

- Use a target coding scheme
  - $t=+1$, for class C_1 and $t=-1$ for C_2 matching the activation function

Perceptron: As a single neuron

- $g(\mathrm{x}) = f\left(\mathrm{w}^{\mathrm{T}}\mathrm{x}\right)$



Perceptron Error Function

- Error function: number of misclassifications
- This error function is a piecewise constant function of $\mathbf{w}$ with discontinuities (unlike regression)
- Hence no closed form solution (no derivatives exist for non smooth functions)

Perceptron Criterion

- Seek w such that $\mathrm{x_n} \in C_1$ will have $\mathrm{w}^{\mathrm{T}}(\mathrm{x_n}) \geq 0$ whereas patterns $\mathrm{x_n} \in C_2$ will have $\mathrm{w}^{\mathrm{T}}(\mathrm{x_n}) < 0$
- Using $t \in \{+1,-1\}$, all patterns need to satisfy

$$\mathrm{w}^{\mathrm{T}} \phi(\mathrm{x}_n) t_n > 0$$

- For each misclassified sample, Perceptron Criterion tries to minimize $-\mathrm{w}^{\mathrm{T}}\phi(\mathrm{x}_n)t_n$ or

$$E_P(\mathrm{w}) = -\sum_{n \in M} \mathrm{w}^{\mathrm{T}} \phi_n t_n$$

M denotes set of all misclassified patterns and

$\phi_n = \phi(\mathrm{x}_n)$

Perceptron Algorithm

- Error function $E_P(\mathrm{w}) = -\Sigma_{n\in M}\mathrm{w}^{\mathrm{T}}\phi_n t_n$
- Stochastic Gradient Descent
  - Change in weight is given by

$$\mathrm{w}^{\mathrm{t}+1} = \mathrm{w}^{\mathrm{t}} - \eta\nabla E_P(\mathrm{w}) = \mathrm{w}^{\mathrm{t}} + \eta\,\phi_n t_n$$

η is learning rate, t indexes the steps

- The algorithm

  Cycle through the training patterns in turn

  If incorrectly classified for class C_1 add to weight vector

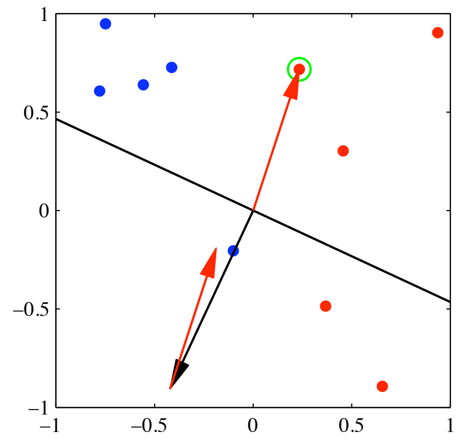
  If incorrectly classified for class C_2 subtract from weight vector

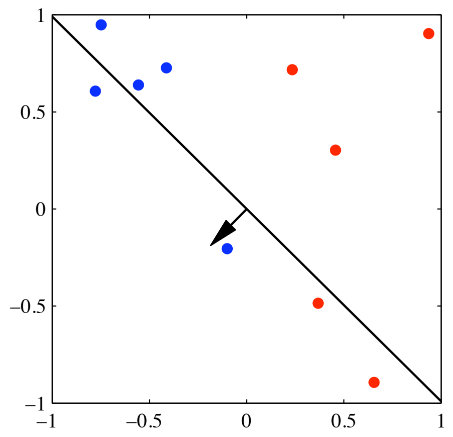
Perceptron Learning Illustration

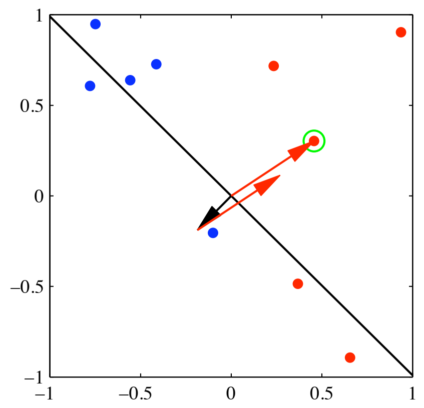
Two-dimensional Feature space (ϕ_1, ϕ_2) and Two-classes

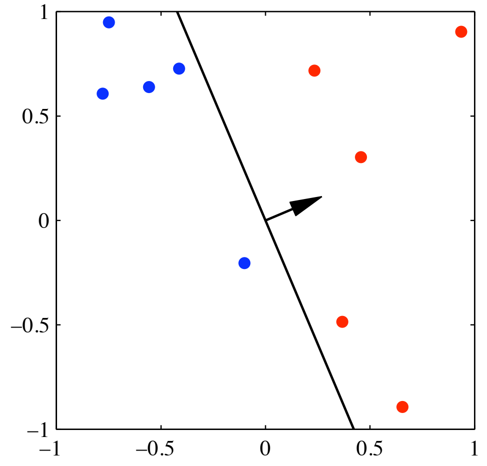
Weight vector in black

Green point is misclassified, which is added to weight vector



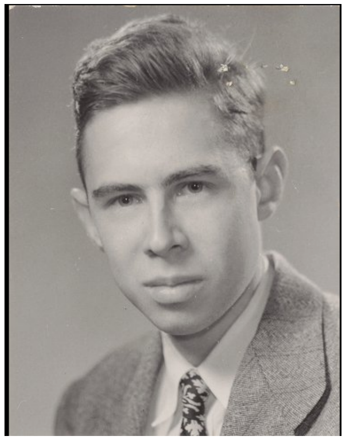




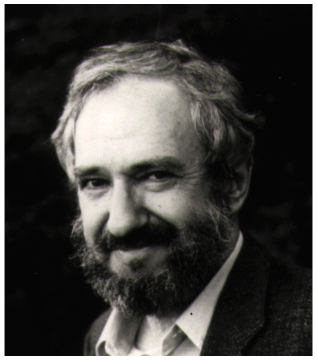


Data points
Correctly classified

History of Perceptrons

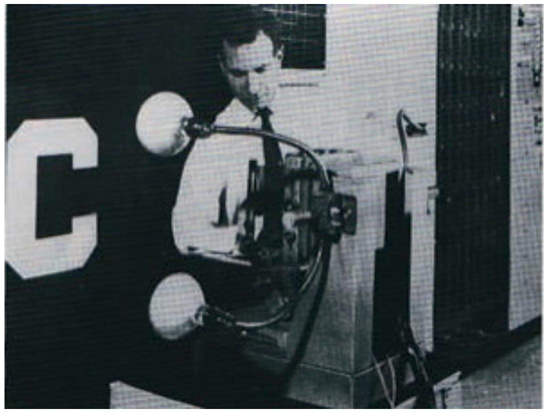


Perceptron
Invented at Calspan Buffalo, NY
Rosenblatt, Frank,
The Perceptron--a perceiving and recognizing automaton.
Report 85-460-1, 1957
Cornell Aeronautical Laboratory



Minsky and Papert
dedicated book to him
Minsky M. L. and Papert S. A. 1969
Perceptrons, MIT Press

Analog Perceptron Hardware



Learning to discriminate shapes of characters

20x20 cell
Image of character

Patch-board to allow different configurations of input features ϕ

Racks of Adaptive Weights

Implemented as potentiometers

Disadvantages of Perceptrons

- Does not converge if classes not linearly separable
- Does not provide probabilistic output
- Not readily generalized to $K > 2$ classes

Summary

- Linear Discrimin. Funcs have simple geometry
- Extensible to multiple classes
- Parameters can be learnt using
  - Least squares
    - not robust to outliers, model close to target values
  - Fisher's linear discriminant
    - Two class is special case of least squares
    - Not easily generalized to more classes
  - Perceptrons
    - Does not converge if classes not linearly separable
    - Does not provide probabilistic output
    - Not readily generalized to $K>2$ classes