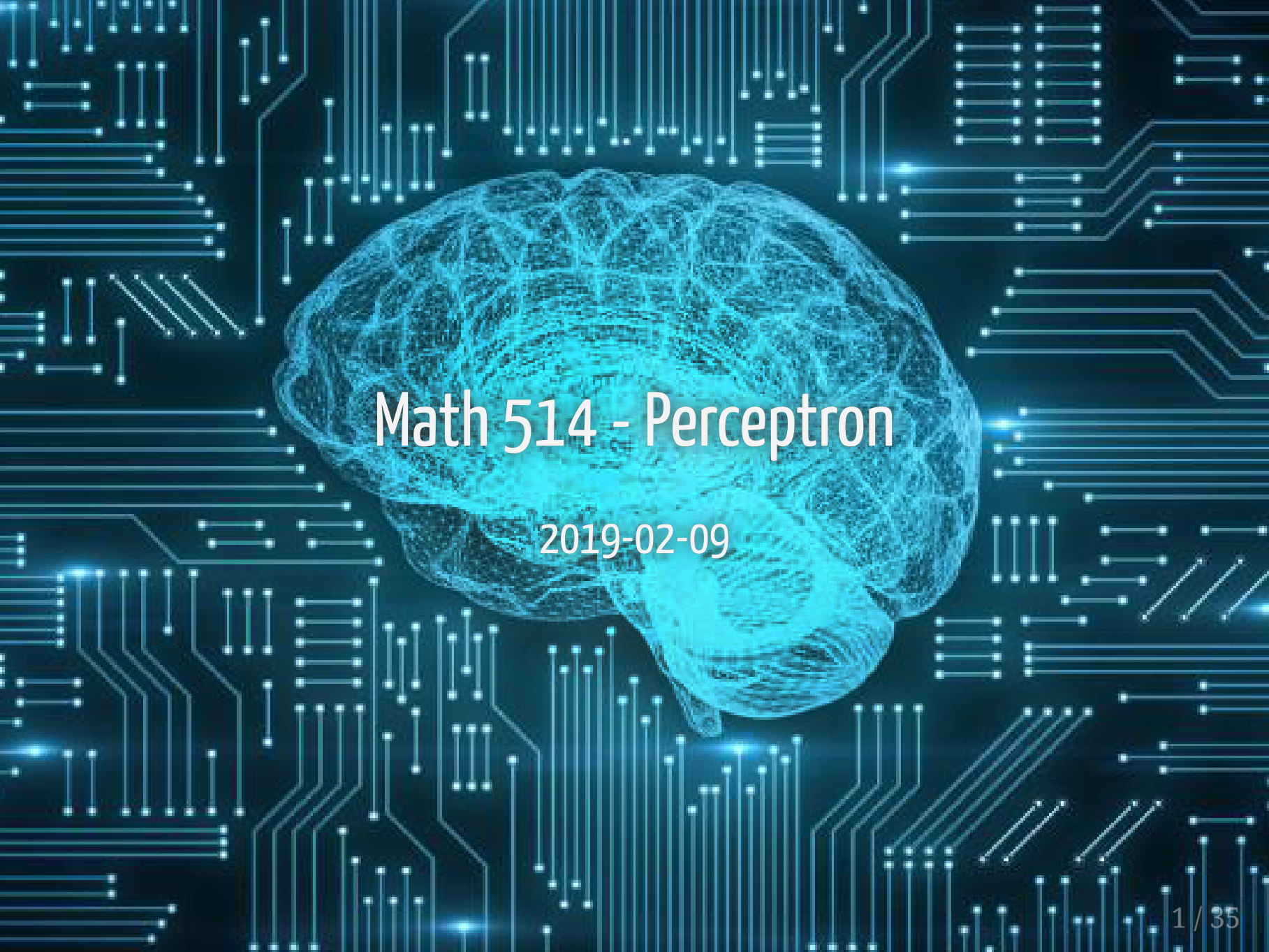


Math 514 - Perceptron

2019-02-09

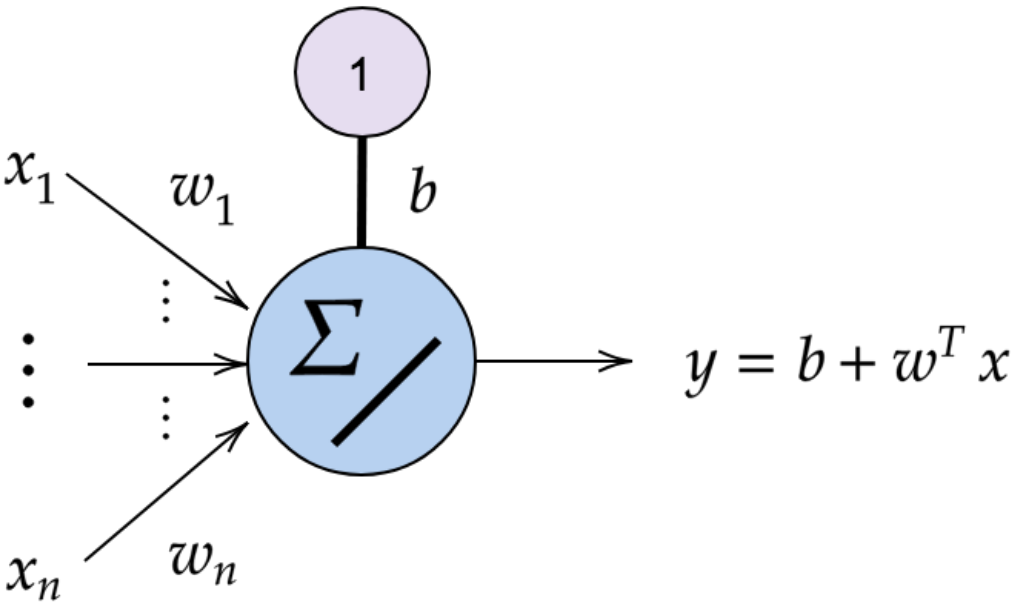
Perceptron

Second linear classifier:

- Fisher discriminant

First network:

- Perceptron



Linear Classifiers

Assume data $\mathbf{x}^{(i)} \in \mathbb{R}^d$ have been categorized into two classes.

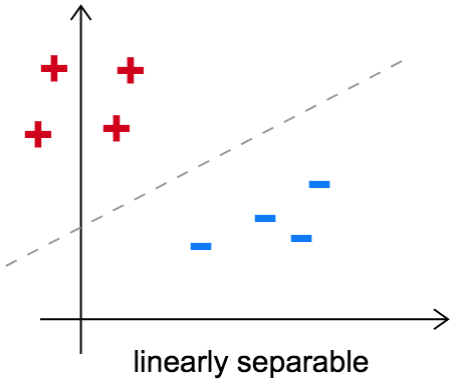
- Last time used probabilistic methods (Bayes' Rule) to separate classes.

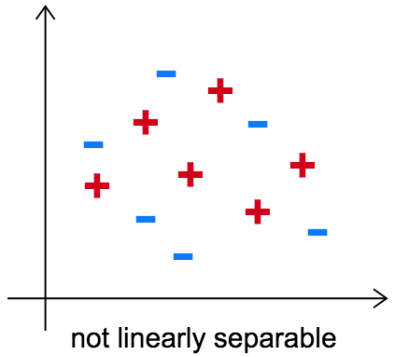
Method 1 - Fisher Discriminant

- Classical statistical technique

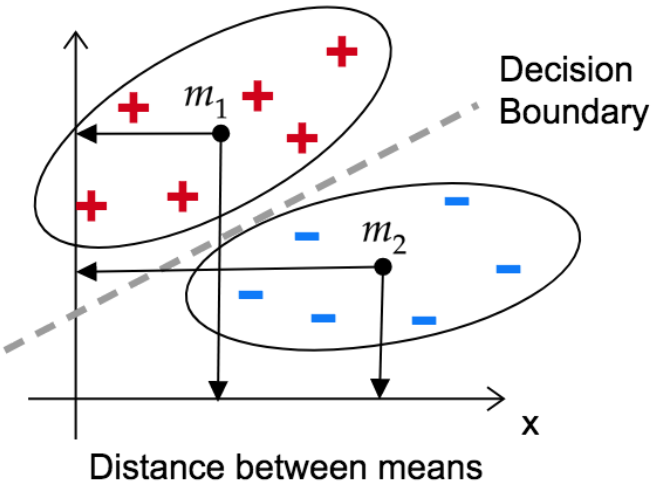
Method 2 - Perceptron

- Algorithm with many variants
- One of the few learning algorithms with a proof of convergence





Linear Classifiers



- Data sets are linearly separable but not when projected onto the x or y axis
- Is there a better line? What about line that maximizes projected distance between data sets?

Fisher Discriminant

Given points $\mathbf{x} \in \mathbb{R}^d$, compute class means $\mathbf{m}_1$ and $\mathbf{m}_2$

$$\mathbf{m}_1 = \frac{1}{n_1} \sum_{C_1} \mathbf{x}_i \qquad \mathbf{m}_2 \qquad = \frac{1}{n_2} \sum_{C_2} \mathbf{x}_i$$

Find projection vector $\mathbf{w}$ that maximizes the projected distance between $\mathbf{m}_1$ and $\mathbf{m}_2$

$$\max_{\mathbf{w}} \|\mathbf{w}^T(\mathbf{m}_2 - \mathbf{m}_1)\|$$

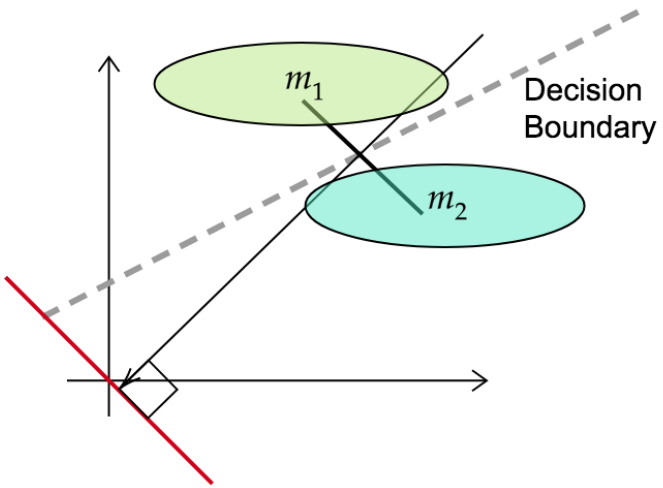
We know for unit vectors $\mathbf{w}$, that this is maximized when $\mathbf{w}$ is in the same direction as $\mathbf{m}_2 - \mathbf{m}_1$

- Projecting onto direction defined by the line connecting the data set means maximizes distance between means of projections.

The projection of the mean vectors equals the mean of the projections

$$\mathbf{w}^T\mathbf{m}_i = \mathbf{w}^T \left(\frac{1}{n_i} \sum_{C_i} \mathbf{x}_i \right) = \frac{1}{n_i} \sum_{C_i} \mathbf{w}^T\mathbf{x}_i$$

Linear Classifiers



- The data sets are linearly separable.
- Projecting onto a line parallel to line between means maximizes mean distance of the projected points.
- Again, data is separable but not when projected onto line connecting means.
- Is there a better line?
- Variance of the projected data must be part of the solution.This projection smears out the classes too much

Fisher Discriminant

Two preliminary results:

Empirical Covariance

- Will show that if X is a centered (zero mean) data matrix with samples arranged in rows, then the empirical covariance matrix is proportional to $\text{Cov} \sim X^T X$. If the data samples are arranged by column, then $\text{Cov}(X) \sim X X^T$

Re-expressing a projection using covariance

- The scalar $(\mathbf{w}^T \mathbf{x})^2$ can be computed from a vectors covariance matrix.

$$\begin{aligned}(\mathbf{w}^T \mathbf{x})^2 &= (\mathbf{w}^T \mathbf{x})(\mathbf{w}^T \mathbf{x})^T \\ &= (\mathbf{w}^T \mathbf{x})(\mathbf{x}^T \mathbf{w}) \\ &= \mathbf{w}^T (\mathbf{x}\mathbf{x}^T) \mathbf{w}\end{aligned}$$

Empirical Covariance - Take 1

$\text{Cov}[X, Y] = E[(X - E[X])(Y - E[Y])] = E[XY]$ when $E[X] = E[Y] = 0$

Given data vectors $\mathbf{x}^{(i)}, i = 1, \cdots, m$ want to compute the empirical covariance of the $\mathbf{x}$ components

Let $\tilde{\mathbf{x}}^{(i)}$ be the centered (zero mean) version of the data $\mathbf{x}^{(i)}$

$$\tilde{\mathbf{x}}^{(i)} = \mathbf{x}^{(i)} - \frac{1}{m} \sum \mathbf{x}^{(i)}$$

Using that $\text{Cov}(X, Y) = E[XY]$ when $E[X] = E[Y] = 0$, the (j, k) element of the (symmetric) covariance matrix is the mean of $\tilde{x}_j$ times $\tilde{x}_k$

$$C_{j\,k} = \frac{1}{m} \sum_{i=1}^{m} \tilde{x}_j \tilde{x}_k \approx E[X_j X_k]$$

Note: This empirical equation is biased, should divide by $m - 1$ to get the unbiased estimate. The difference is seldom important.

Empirical Covariance - Take 1

The matrix outer product gives products of vector components

$$\begin{bmatrix} x_1^{(i)} \\ \vdots \\ x_d^{(i)} \end{bmatrix} [x_1^{(i)}, \cdots, x_d^{(i)}] = \begin{bmatrix} x_1^{(i)} x_1^{(i)} & \cdots & x_1^{(i)} x_d^{(i)} \\ \vdots & & \vdots \\ x_d^{(i)} x_1^{(i)} & \cdots & x_d^{(i)} x_d^{(i)} \end{bmatrix}$$

Showed earlier that one way to interpret matrix multiplication AB is

$AB = \sum_i col_i(A) \bigotimes row_i(B)$

If the vectors $\mathbf{x}^{(i)}$ are arranged as the columns of X then the covariance matrix is:

$$\begin{aligned} C &= \frac{1}{m} \sum_i col_i(X)\, row_i(X^T) \\ &= \frac{1}{m} X X^T \end{aligned}$$

note 1: It is the transpose of this if data is arranged by rows.

note 2: The dimension of the covariance matrix should be $d \times d$ if the data vectors have d elements

Empirical Covariance - Take 2

Given m data points $\mathbf{x}^{(i)} \in \mathbb{R}^d$ write the data matrix X as

$$X = \begin{bmatrix} \mathbf{x}_1^{(1)} & \cdots & \mathbf{x}_d^{(1)} \\ \vdots & & \vdots \\ \mathbf{x}_1^{(m)} & \cdots & \mathbf{x}_d^{(m)} \end{bmatrix}$$

In X, the samples are arranged in rows and features/coordinates in columns

The sample mean vector (column means) is

$$\mu = \frac{1}{m} \sum_{i=1}^{m} \mathbf{x}^{(i)}$$

$\mu \in \mathbb{R}^d$. The covariance matrix for the data has generic (j,k) entries (definition)

$$\mathrm{Cov}(X)_{jk} = E[(\mathbf{x}_j - \mu_j)(\mathbf{x}_j - \mu_j)] \approx \frac{1}{m-1} \sum_{i=1}^{m} (\mathbf{x}_j^{(i)} - \mu_j)(\mathbf{x}_k^{(i)} - \mu_k)$$

Empirical Covariance - Take 2

$\text{Cov}(X) \in d \times d$

If we take just one row of the data, say $\mathbf{x}^{(p)}$ and think of $\mathbf{x}^{(p)} - \mu$ as a row vector, compute

$$S^{(p)} = \underbrace{(\mathbf{x}^{(p)} - \mu)^T}_{\text{column}} \underbrace{(\mathbf{x}^{(p)} - \mu)}_{\text{row}}$$

The generic (i, j) entry for $S^{(p)}$ is

$$S_{ij}^{(p)} = (\mathbf{x}_i^{(p)} - \mu_i)^T (\mathbf{x}_j^{(p)} - \mu_j)$$

This is the $\mathbf{x}^{(p)}$ contribution to the covariance matrix. Using this, the $\text{Cov}(X)$ can be written as

$$\begin{aligned} \text{Cov}(X) &= \frac{1}{m-1} \sum_i^m (\mathbf{x}^{(i)} - \mu)^T (\mathbf{x}^{(i)} - \mu) \\ &= \frac{1}{m-1} \sum_i^m S^{(i)} \end{aligned}$$

This is a sum of outer products.

Fisher Discriminant

Fisher condition

Find

$$\max_{\mathbf{w}} J(\mathbf{w})$$

where

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

- m_i are class means projected onto $\mathbf{w}$, so $m_2 - m_1$ measures between class distance
- s_i^2 are within-class scatter, so $s_1^2 + s_2^2$ measures total within-class variation
- J is maximized by finding a $\mathbf{w}$ that balances increasing the distance between the projected class means while not smearing the class scatter too much

Fisher Discriminant

Let $\mathbf{w}$ define the unknown 1^d projection subspace. Assume $\|\mathbf{w}\| = 1$

Data sets $x_j^{(i)} \in \mathbb{R}^d$, $i = 1, 2 \ldots n_j$, $j \in 1, 2$

$$
\begin{aligned}
y^{(i)} &= \mathbf{w}^T \mathbf{x}^{(i)} \\
\mathbf{m}_j &= \frac{1}{n_j} \sum_{C_j} \mathbf{x}^{(i)} \\
\tilde{m}_j &= \frac{1}{n_j} \sum_{C_j} y^{(i)} \\
&= \frac{1}{n_j} \sum_{C_j} \mathbf{w}^T \mathbf{x}^{(i)} \\
&= \mathbf{w}^T \mathbf{m}_j
\end{aligned}
$$

The distance between projected means is

$$|\tilde{m}_1 - \tilde{m}_2| = |\mathbf{w}^T (\mathbf{m}_1 - \mathbf{m}_2)|$$

Already showed that the distance between projected means is maximized by the line connecting the data set means.

Fisher Discriminant

Define the scatter for the projected data as

$$s_j^2 = \sum_{y \in C_j} (y - \tilde{m})^2$$

so $s_1^2 + s_2^2$ is an estimate of the total variance of the projected data.

The **Fisher Linear Discriminant** is the line that maximizes:

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

It takes some algebra to express J as a function $\mathbf{w}$

Consider $s_1^2 + s_2^2$

$$s_1^2 + s_2^2 = \sum_{C_1} (y - \tilde{m_1})^2 + \sum_{C_2} (y - \tilde{m_2})^2$$

Fisher Discriminant

$$
\begin{aligned}
s_i^2 &= \sum_{x \in C_i} (\mathbf{w}^T x - \mathbf{w}^T \mathbf{m}_i)^2 \\
&= \sum_{x \in C_i} (\mathbf{w}^T (\mathbf{x} - \mathbf{m}_i))^2
\end{aligned}
$$

Now use result derived earlier $(\mathbf{w}^T x)^2 = \mathbf{w}^T (X X^T) \mathbf{w}$ and rewrite the within class scatter as

$$
\begin{aligned}
s_i &= \sum_{x \in C_i} \mathbf{w}^T (\mathbf{x} - \mathbf{m}_i)(\mathbf{x} - \mathbf{m}_i)^T \mathbf{w} \\
&= \mathbf{w}^T \Big(\sum_{x \in C_i} (\mathbf{x} - \mathbf{m}_i)(\mathbf{x} - \mathbf{m}_i)^T \Big) \mathbf{w}
\end{aligned}
$$

Note that summation is proportional to the data covariance matrix for class i. It measures variation within the class.

Defining

$$
\begin{aligned}
S_i &= \sum_{x \in C_i} (\mathbf{x} - \mathbf{m}_i)(\mathbf{x} - \mathbf{m}_i)^T \\
S_w &= S_1 + S_2
\end{aligned}
$$

Gives

$$
\begin{aligned}
s_1^2 + s_2^2 &= \mathbf{w}^T S_1 \mathbf{w} + \mathbf{w}^T S_2 \mathbf{w} \\
&= \mathbf{w}^T S_w \mathbf{w}
\end{aligned}
$$

Fisher Discriminant

The numerator of the Fisher cost function J is

$$\begin{aligned}(\tilde{m}_1 - \tilde{m}_2)^2 &= (\mathbf{w}^T\mathbf{m}_1 - \mathbf{w}^T\mathbf{m}_2)^2 \\ &= (\mathbf{w}^T(\mathbf{m}_1 - \mathbf{m}_2))^2 \\ &= \mathbf{w}^T\Big((\mathbf{m}_1 - \mathbf{m}_2)(\mathbf{m}_1 - \mathbf{m}_2)^T\Big)\mathbf{w}\end{aligned}$$

Define $S_B = (\mathbf{m}_1 - \mathbf{m}_2)(\mathbf{m}_1 - \mathbf{m}_2)^T$. S_B measures variation between classes.
Combining results:

$$J(\mathbf{w}) = \frac{\mathbf{w}^T S_B \mathbf{w}}{\mathbf{w}^T S_w \mathbf{w}}$$

Fisher Discriminant

Want to find the $\mathbf{w}$ that maximizes J. Note that S_B has rank 1 and

$$S_B\mathbf{x} = (\mathbf{m}_2 - \mathbf{m}_1)(\mathbf{m}_2 - \mathbf{m}_1)^T\mathbf{x}$$

The objective function is a ratio of scalars. Use vector/matrix derivatives for before to compute the derivative of J wrt $\mathbf{w}$

$$\frac{\partial J}{\partial \mathbf{w}} = 0$$

gives:

$$(\mathbf{w}^T S_B \mathbf{w}) S_w \mathbf{w} = (\mathbf{w}^T S_w \mathbf{w}) S_B \mathbf{w}$$

S_B is the outer product of $\mathbf{m}_2 - \mathbf{m}_1$ so $S_B\mathbf{w}$ is in the direction of $\mathbf{m}_2 - \mathbf{m}_1$.

We are only interested in the direction of $\mathbf{w}$ so constant multipliers can be ignored

Fisher Discriminant

$$c_B\, S_w \mathbf{w} = c_w((\mathbf{m}_2 - \mathbf{m}_1)^T \mathbf{w})(\mathbf{m}_2 - \mathbf{m}_1)$$

With $c_m = (\mathbf{m}_2 - \mathbf{m}_1)^T \mathbf{w}$

$$\mathbf{w} = \frac{c_w}{c_B} c_m S_w^{-1}(\mathbf{m}_2 - \mathbf{m}_1)$$

$$\mathbf{w} \sim S_w^{-1}(\mathbf{m}_2 - \mathbf{m}_1)$$

where

$$\mathbf{m}_j = \frac{1}{n_j} \sum_{C_j} \mathbf{x}^{(i)}$$
$$S_i = \sum_{x \in C_i} (\mathbf{x} - \mathbf{m}_i)(\mathbf{x} - \mathbf{m}_i)^T$$
$$S_w = S_1 + S_2$$

If S_w is diagonal, then W is in direction of $\mathbf{m}_2 - \mathbf{m}_1$

Fisher Discriminant

So far have a direction which helps separate two classes when they are projected. Now need threshold which discriminates the projected data (scalars)

- Could assume that the projected data are Gaussian and use Bayes' rule from Week 1
- Bishop shows the following:

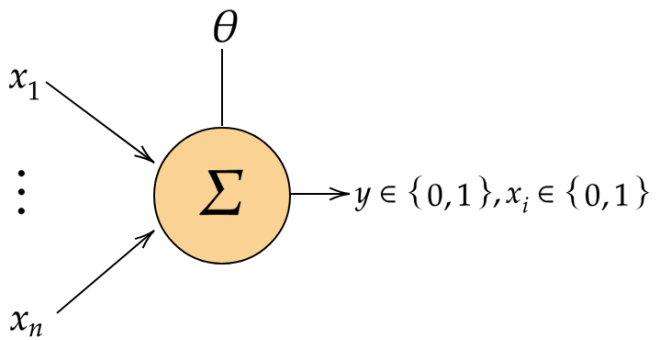
$$\mathbf{m} = \frac{1}{n_1 + n_2}(n_1\mathbf{m}_1 + n_2\mathbf{m}_2)$$
$$g(\mathbf{x}) = \mathbf{w}^T(\mathbf{x} - \mathbf{m})$$

- The discriminant is:

$$g(\mathbf{x}) = \begin{cases} > 0 & \mathcal{C}_1 \\ o.w. & \mathcal{C}_2 \end{cases}$$

Perceptron

The perceptron evolved from the McCulloch-Pitts neuron (1943) where $y \in \{0,1\}$, $\mathbf{x}_i \in \{0,1\}$. There are no dynamic weights



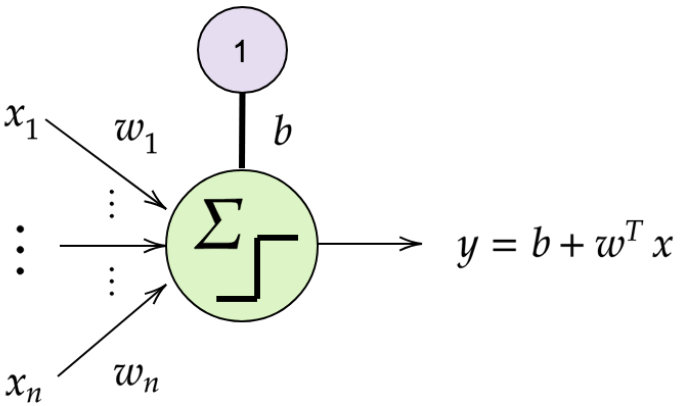
$$y = f(g(\mathbf{x}))$$

$$g(\mathbf{x}) = \sum_{i}^{n} \mathbf{x}_i$$

$$f = \begin{cases} 1 \text{ if } g(x) \geq 0 \\ 0 \qquad \text{o.w.} \end{cases}$$

- Threshold θ fixed
- Inhibitory inputs have veto power
- Excitory inputs have integer values (weights)

Perceptron



No cost function

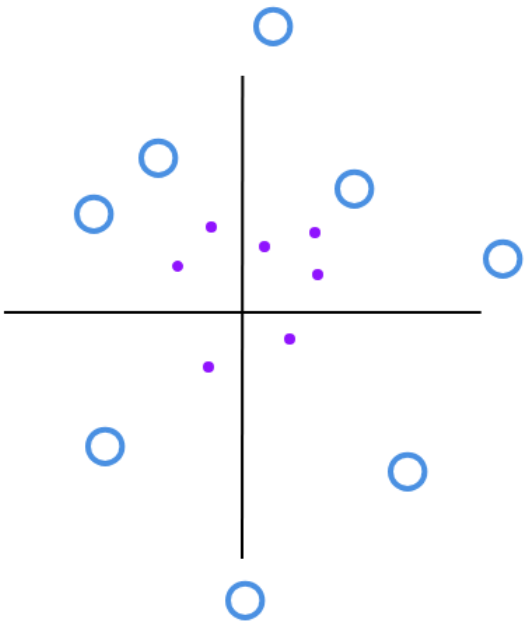
The perceptron modifies the M-P neuron to be more flexible

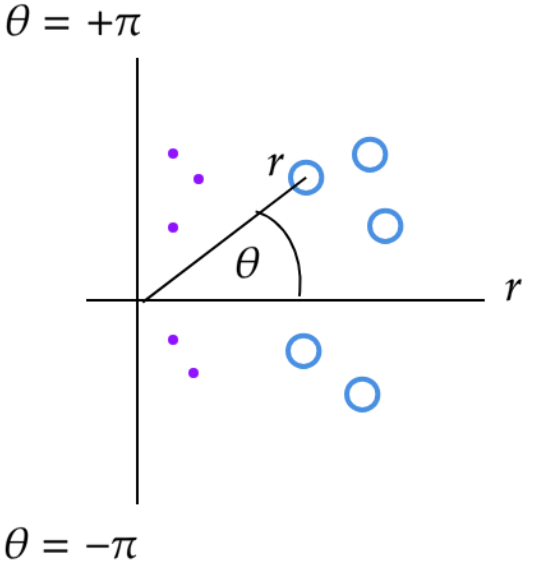
- Real valued inputs (positive or negative)
- Threshold becomes an adjustable parameter
- Modified output to $\{-1, 1\}$
- Drops inhibitory inputs
- **Has Learning rule (justly famous)**
- **Convergence proof (for linearly seperable data)**

Perceptron

The original Perceptron algorithm is only useful for linearly separable data.

- Preprocessing, like transformation to polar coordinates, can change unseparable to to separable



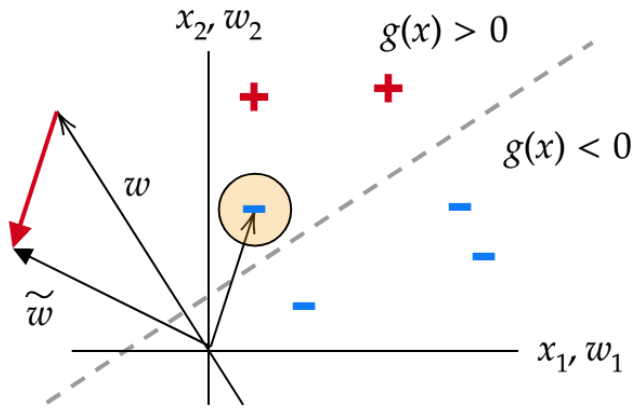


Perceptron Algorithm

Data $\mathcal{D} = (x^{(i)}, t^{(i)}), i = 1, \cdots, m$
$x^{(i)} \in \mathbb{R}^d, t^{(i)} \in \{-1, 1\}$
$b, w \leftarrow 0$
{Scale Data}
For $i = 1, \cdots, \max iterations$ do
  for each pair $(x^{(i)}, t^{(i)}) \in \mathcal{D}$ do:
    $z = b + w^T x^{(i)}$
    if $t^{(i)} z \leq 0$) then
      $w \leftarrow w + t^{(i)} x^{(i)}$
      $b \leftarrow b + t^{(i)}$
    end
  end
end

Perceptron Algorithm

Heuristic View



Highlighted point is misclassified

say $b + \mathbf{w}^T\mathbf{x}^{(i)}) > 0$ but $t^{(i)} = -1$

Update rule:

$$\tilde{\mathbf{w}} = \mathbf{w} + t^{(i)}\mathbf{x} = \mathbf{w} - \mathbf{x}^{(i)}$$

$$\tilde{b} = b + t^{(i)} \cdot 1 = b - 1$$

Update b,w and apply to $x^{(i)}$ again:

$$\begin{aligned} b + \mathbf{w}^T\mathbf{x}^{(i)} &= z^{(i)} > 0 \\ \tilde{b} + \tilde{\mathbf{w}}^T\mathbf{x}^{(i)} &= b - 1 + (\mathbf{w} - \mathbf{x}^{(i)})^T\mathbf{x}^{(i)} \\ &= z^{(i)} - 1 - (\mathbf{x}^{(i)})^T\mathbf{x}_i \\ &= z^{(i)} - 1 - \|\mathbf{x}^{(i)}\|^2 \end{aligned}$$

So $z^{(i)}$ is reduced

- Helps this point, does it hurt other points?

Perceptron Convergence

Two ways to write equations

- with explicit bias
- implicit bias and augmented data

Explicit bias

$$
\begin{aligned}
y^{(i)} &= b + \mathbf{w}^T \mathbf{x}^{(i)} \\
\mathbf{w} &= (w_1, \cdots, w_d) \in \mathbb{R}^d \\
\mathbf{x}^{(i)} &= (x_1^{(i)}), \cdots, x_d^{(i)}) \in \mathbb{R}^d
\end{aligned}
$$

Implicit bias (homogeneous form)

$$
\begin{aligned}
y^{(i)} &= \tilde{\mathbf{w}}^T \tilde{\mathbf{x}}^{(i)} \\
\tilde{\mathbf{w}} &= (w_0, w_1, \cdots, w_d) \in \mathbb{R}^{d+1} \\
\tilde{\mathbf{x}}^{(i)} &= (1, x_1^{(i)}, \cdots, x_d^{(i)}) \in \mathbb{R}^{d+1}
\end{aligned}
$$

identify w_0 with b

- typically, the augmented vectors $\tilde{\mathbf{w}}$ and $\tilde{\mathbf{x}}^{(i)}$ are written without tildes and are made clear by context
- Implicit bias almost always used for analysis
- Will use explicit bias in homeworks to match later neural network equations

Perceptron Convergence

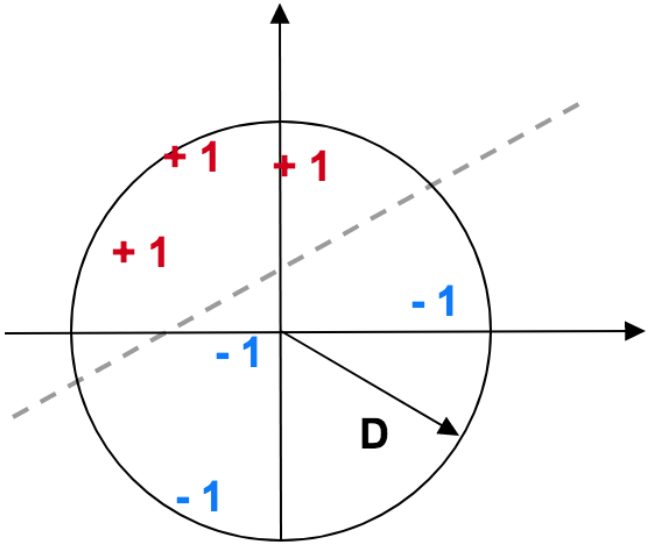
Will use homogeneous form where $x_1^{(i)} = 1$

- Given data $(\mathbf{x}^{(i)}, t^{(i)})$ $i = 1, \cdots, m$

$$\mathbf{x}^{(i)} \in \mathbb{R}^{d+1}$$

$$t^{(i)} \in \{-1, 1\}$$

- Assume $\|\mathbf{x}^{(i)}\| \leq D \quad \forall \quad i$
- Assume classes are linearly separable.



Perceptron Convergence

Theorem:

$$\exists \text{ vector } \mathbf{u},\ |\mathbf{u}| = 1 \text{ such that } t^{(i)}\mathbf{u}^T\mathbf{x}^{(i)} > 0\ \forall\ i$$

This implies there is a margin $\gamma > 0 \ni\ t^{(i)}\mathbf{u}^T\mathbf{x}^{(i)} > \gamma$ and the total number of mistakes is bounded above by $(\frac{D}{\gamma})^2$ where $\|\mathbf{x}\| \leq D$

Let the adjustable parameters be $\mathbf{w} \in \mathbb{R}^{d+1}$ and let $w = 0$ initially

- Theorem says that a solution (separator) exists
- Puts an upper bound on how much work is needed to find $\mathbf{u}$
- Will show that $\mathbf{w}$ will iteratively get closer to $\mathbf{u}$ $(\mathbf{w} \cdot \mathbf{u}$ grows $)$
- The growth in $(\mathbf{w} \cdot \mathbf{u})$ is not dominated by the growth in $\mathbf{w}$

Perceptron Convergence

Let $\mathbf{w}^k$ be parameter values when k-th mistake is made

- $\mathbf{w} = 0$ initially, so $\mathbf{w}^{(1)} = 0$

It follows that

$$\text{sign}(\mathbf{x^{(j)}} \cdot \mathbf{w}^{(k)}) \neq t^{(j)} \quad \text{some } j$$
$$t^{(j)} \,\text{sign}(\mathbf{x^{(j)}} \cdot \mathbf{w}^{(k)}) < 0$$
$$\mathbf{w}^{k+1} = \mathbf{w}^{(k)} + t^{(j)}\mathbf{x}^{(j)}$$

$$\begin{aligned}\mathbf{u}^T\mathbf{w}^{k+1} &= \mathbf{u}^T\mathbf{w}^{(k)} + t^{(j)}\mathbf{u}^T\mathbf{x}^{(j)} \\ &\geq \mathbf{u}^T\mathbf{w}^{(k)} + \gamma\end{aligned}$$

Telescope, using $\mathbf{w}^{(1)} = 0$

$$\begin{aligned}\mathbf{u}^T\mathbf{w}^{(2)} &\geq \gamma \\ \mathbf{u}^T\mathbf{w}^{(3)} &\geq \mathbf{u}^T\mathbf{w}^{(2)} + \gamma \\ &\geq 2\gamma \\ &\vdots\end{aligned}$$

$$\mathbf{u}^T\mathbf{w}^{k+1} \geq k\gamma$$

shows projection of $\mathbf{w}$ on $\mathbf{u}$ grows

Perceptron Convergence

Now need to show that the growth is not all due to growth in $\mathbf{w}^{(k)}$

$$\begin{aligned}
\|\mathbf{w}^{k+1}\|^2 &= \|\mathbf{w}^{(k)} + t^{(j)}x^{(j)}\|^2 \\
&= \|\mathbf{w}^{k}\|^2 + \|x^{(j)}\|^2 + \underbrace{2t^{(j)}(\mathbf{x^{(j)}})^T\mathbf{w^{(k)}}}_{must\ be\ <0} \\
&\leq \|\mathbf{w}^{k}\|^2 + \|x^{(j)}\|^2 \\
&\leq \|\mathbf{w}^{k}\|^2 + D^2
\end{aligned}$$

By induction

$$\begin{aligned}
\|\mathbf{w}^{k+1}\|^2 &\leq kD^2 \\
\sqrt{k}D &\geq \|\mathbf{w}^{k+1}\| \\
&\geq (\mathbf{w}^{(k+1)})^T\mathbf{u} \\
&\geq k\gamma \quad \text{by previous slide}
\end{aligned}$$

$$k \leq (D/\gamma)^2$$

Perceptron Convergence

$$\mathbf{u}^T \mathbf{w}^{k+1} \geq k\gamma$$

$$\|\mathbf{w}^{k+1}\|^2 \leq kD^2$$

$$k \leq (D/\gamma)^2$$

$$\mathbf{u}^T \mathbf{w}^{k+1} = \|\mathbf{w}^{k+1}\| \cos\theta$$

$$\cos\theta = \frac{\mathbf{u}^T \mathbf{w}^{k+1}}{\|\mathbf{w}^{k+1}\|} \geq \frac{k\gamma}{\sqrt{k}D} = \sqrt{k}\frac{\gamma}{D}$$

It follows that θ, the angle between $\mathbf{u}$ and $\mathbf{w}$ decreases as each mistake is corrected

Learning Rules

Error correcting rules
(no cost function):

- Perceptron (vanilla)
- Pocket
- Voted
- Averaged
- Perceptron with margin

Cost reduction rules
(gradient descent, MSE)

- Widrow-Hoff (LMS)
- Delta rule
- Ho - Kashyap

Performance

- The error correcting algorithm will perfectly separate separable data.

- The delta rule minimizes squared error but may not be perfect on separable data.

Ensembles

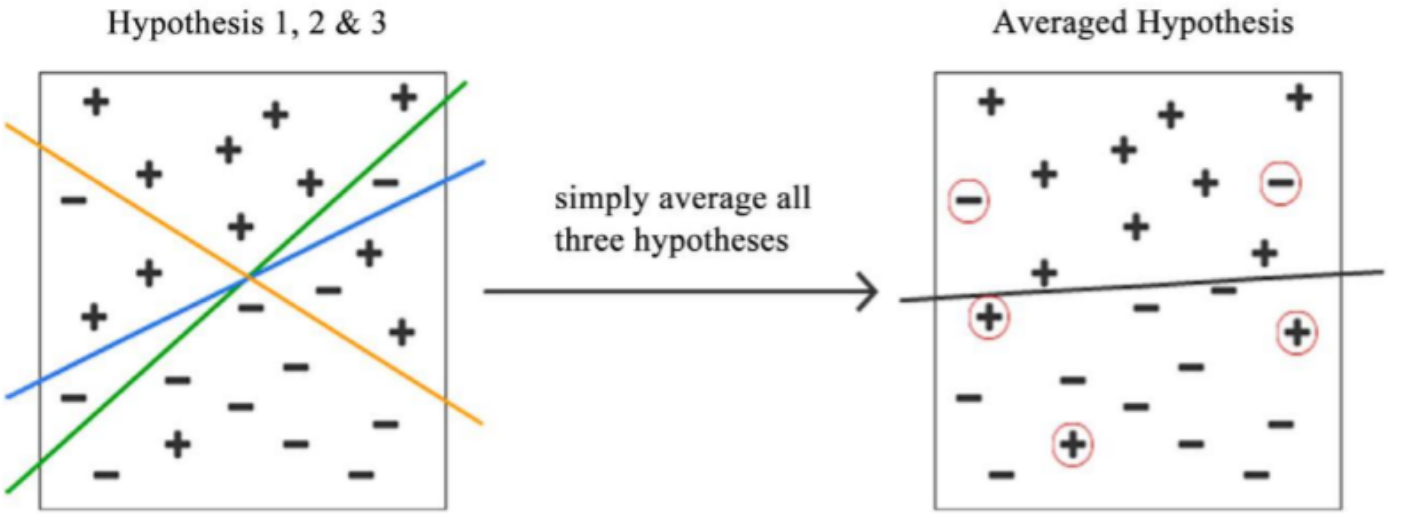


Figure 4. Ensemble model by Averaging.

Averaged Perceptron Algorithm

All hyperplane solutions $b^{(k)}, \mathbf{w}^{(k)}$ and all survival times $c^{(k)}$ are stored. The prediction rule uses the survival time weighting of the saved solutions:

$$\begin{aligned}\hat{y}(x) &= \text{sign}(\sum_{k=1}^{K} c^{(k)}(b^{(k)} + \mathbf{w}^{(k)} \cdot x)) \\ &= \text{sign}\left(\sum_{k=1}^{K} \left(c^{(k)}\mathbf{w}^{(k)}\right) \cdot x + \sum_{k=1}^{K} c^{(k)} b^{(k)}\right)\end{aligned}$$

See Daume for details

Voted Perceptron Algorithm

All hyperplane solutions are remembered and weighted by how long they survive.

Run the baseline perceptron algorithm and while training store

- bias and weights at each update
- the number of samples the bias/weights survived
- The last update will survive for the entire data set

The voted perceptron prediction rule is

$$\hat{y}(x) = \text{sign} \sum_{1}^{K} c^{(k)} \, \text{sign}(b^{(k)} + \mathbf{w}^{(k)} \cdot \mathbf{x})$$

See freund paper and Daume for details

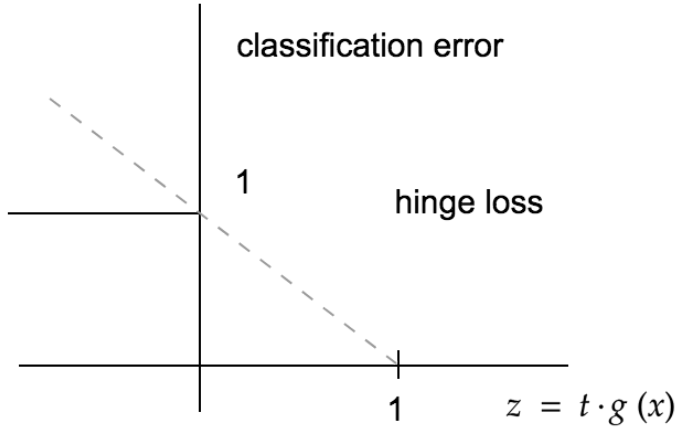
Perceptron Cost Function

Cost Function

$t^{(i)} \in \{-1, +1\}$

$$y^{(i)} = \begin{cases} +1 \text{ if } g(x^{(i)}) \geq 0 \\ -1 \qquad\qquad o.w. \end{cases}$$

$t^{(i)}$	$y^{(i)}$	$t^{(i)} \cdot g(x)$	
- 1	- 1	> 0	correct
- 1	+ 1	≥ 0	incorrect
+ 1	- 1	< 0	incorrect
+ 1	- 1	> 0	correct



$$Cost = \max(0, 1 - z) = \max(0, 1 - t \cdot g(x))$$

$$\frac{\partial C}{\partial W_j} = -t \cdot x_j$$

Update w in opposite direction of gradient if misclassified