Lecture 15: 7 March, 2024

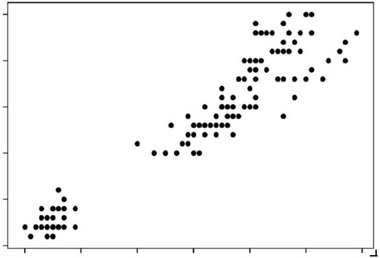
Madhavan Mukund

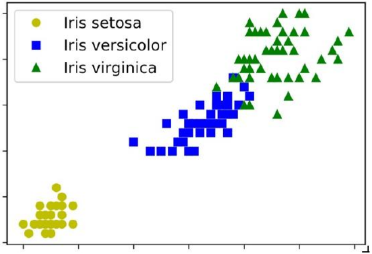
https://www.cmi.ac.in/~madhavan

Data Mining and Machine Learning
January–April 2024

Unsupervised learning

- Supervised learning requires labelled data
- Vast majority of data is unlabelled
- What insights can you get into unlabelled data?





"If intelligence was a cake, unsupervised learning would be the cake, supervised learning would be the icing on the cake ..."

- Yann LeCun
ACM Turing Award 2018

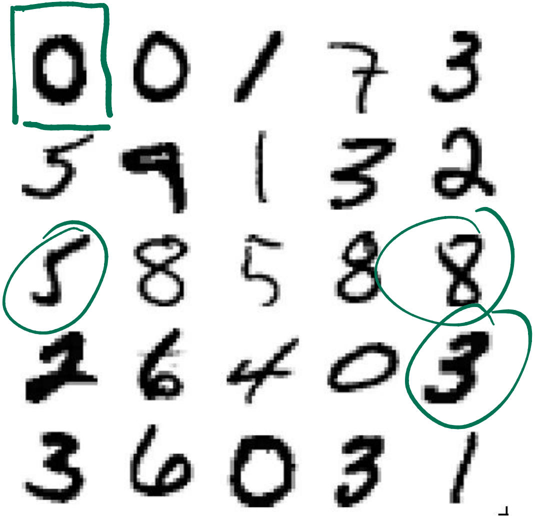
Applications

- Customer segmentation
  - Marketing campaigns
- Anomaly detection
  - Outliers
- Semi-supervised learning
  - Propagate limited labels
- Image segmentation
  - Object detection

Semi-supervised learning

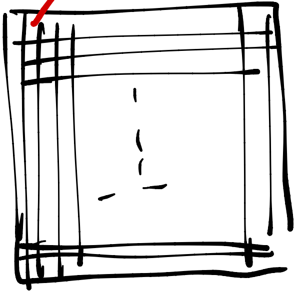
28x28 pixel, grayscale 0-1 → 28^2 column matrix

- Labelling training data is a bottleneck of supervised learning
- Handwritten digits 0,1,...,9
    - 1797 images
- Standard logistic regression model has 96.9% accuracy
- Suppose we take 50 random samples as training set
- Logistic regression gives 83.3%



Image

1 pixel — "gray" value in [0..1]



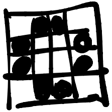


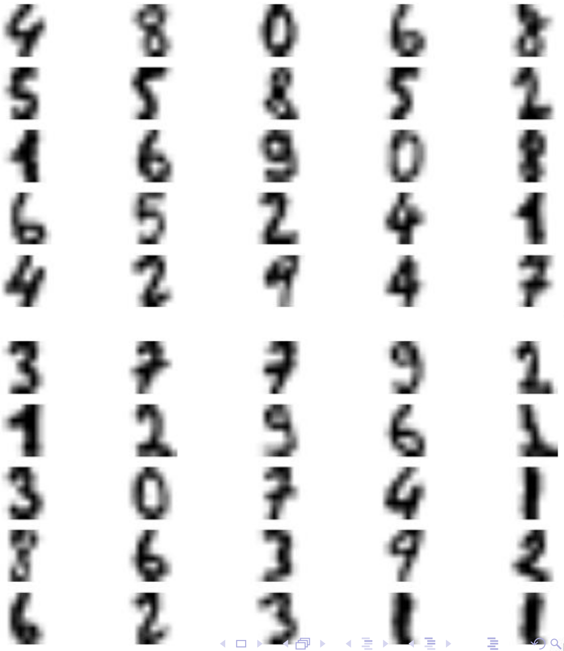
Image data

(0,0) | (0,1) | .. | (0,27) | (1,0) | -- (1,27) | -- | (27,27

Single Image

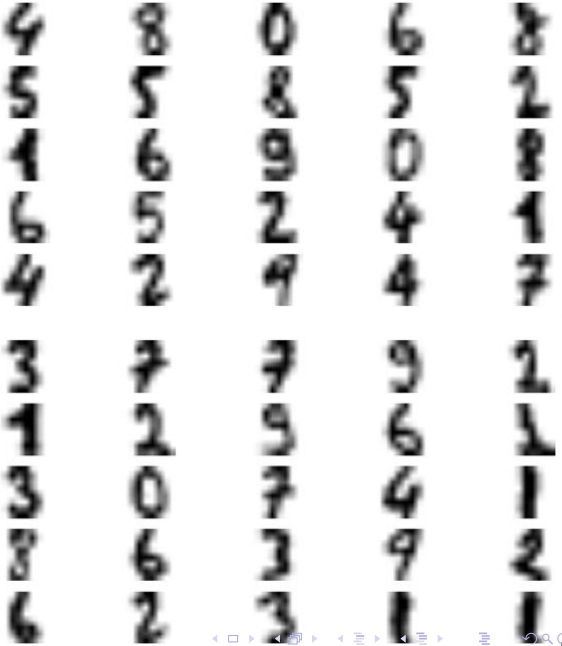
Semi-supervised learning

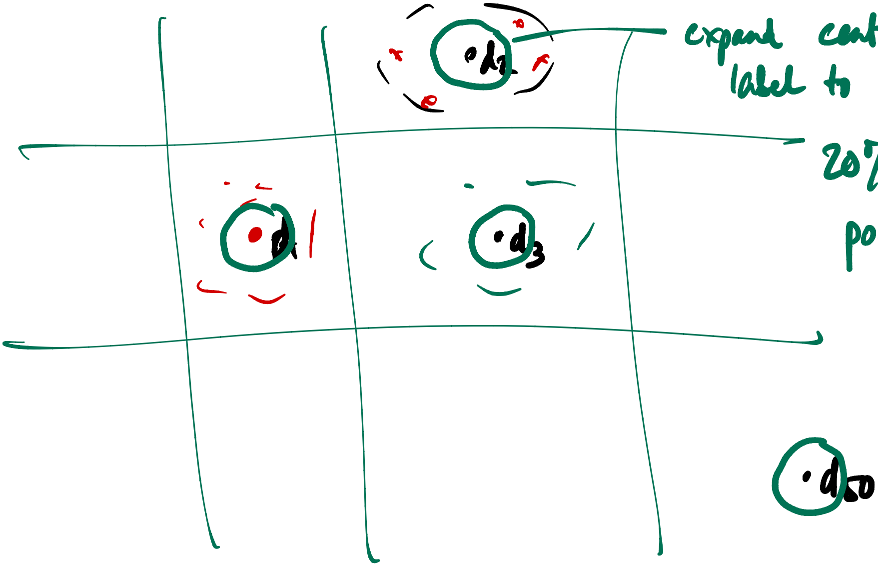
- Instead of 50 random samples, 50 clusters using K means
- Use image nearest to each centroid as training set
    - *50 representative images*
- Logistic regression accuracy jumps to 92.2%



Semi-supervised learning

- Propagate representative image label to entire cluster
- Logistic regression improves to 93.3%
- Propagage representive image label to only 20% items closest to centroid
- Logistic regression improves to 94%
- Only 50 actual labels used, about 5 per class!





expand centroid label to nearest

20% ↑ points

Image segmentation

- An image is a matrix of pixels
- Each pixel has (R,G,B) values
- K means clustering on these values merges colours

Image segmentation

- An image is a matrix of pixels
- Each pixel has (R,G,B) values
- K means clustering on these values merges colours
- With 10 clusters, not much change

10 colors

Image segmentation

- An image is a matrix of pixels
- Each pixel has (R,G,B) values
- K means clustering on these values merges colours
- With 10 clusters, not much change
- Same with 8

Image segmentation

- An image is a matrix of pixels
- Each pixel has (R,G,B) values
- K means clustering on these values merges colours
- With 10 clusters, not much change
- Same with 8
- At 6 colours, ladybug red goes

6 colors

Image segmentation

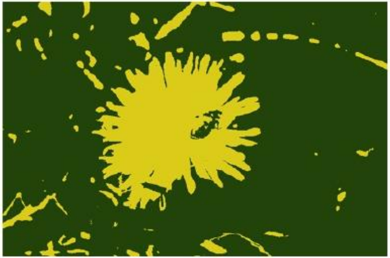
- An image is a matrix of pixels
- Each pixel has (R,G,B) values
- K means clustering on these values merges colours
- With 10 clusters, not much change
- Same with 8
- At 6 colours, ladybug red goes
- 4 colours

4 colors

Image segmentation

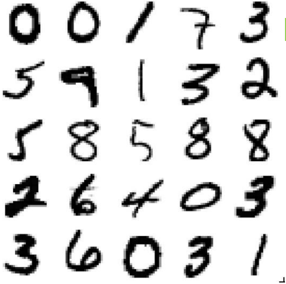
- An image is a matrix of pixels
- Each pixel has (R,G,B) values
- K means clustering on these values merges colours
- With 10 clusters, not much change
- Same with 8
- At 6 colours, ladybug red goes
- 4 colours
- Finally 2 colours, flower and rest

2 colors



Summary

- Unsupervised learning is useful as a preprocessing step
- Semi supervised learning
    - Identify a small subset of items to label manually
    - Propagate labels via cluster
- Image segmentation
    - Highlight objects by colour

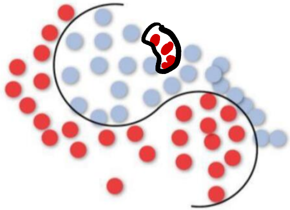


A geometric view of supervised learning

- Think of data as points in space
- Find a separating curve (surface)

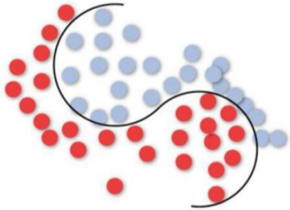
A geometric view of supervised learning

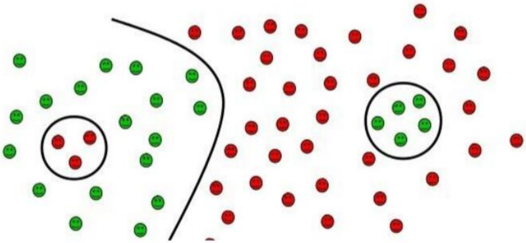
- Think of data as points in space
- Find a separating curve (surface)
- Separable case
  - Each class is a connected region
  - A single curve can separate them



A geometric view of supervised learning

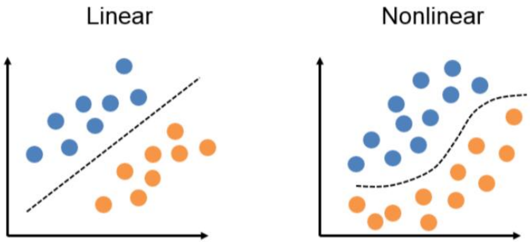
- Think of data as points in space
- Find a separating curve (surface)
- Separable case
  - Each class is a connected region
  - A single curve can separate them
- More complex scenario
  - Classes form multiple connected regions
  - Need multiple separators





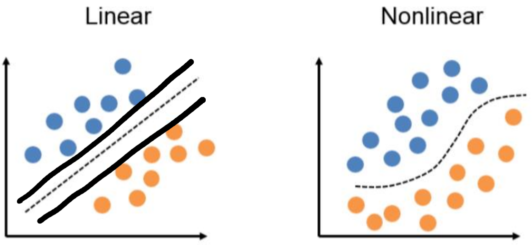
Linear separators

- Simplest case — linearly separable data



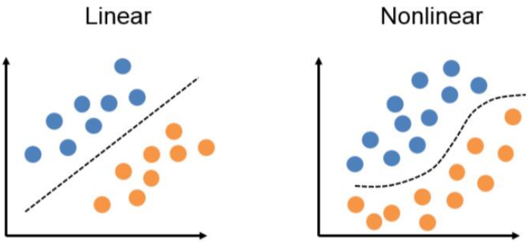
Linear separators

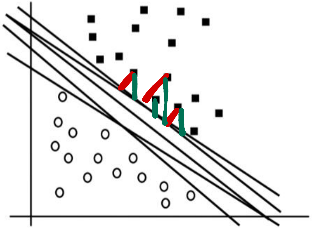
- Simplest case — linearly separable data
- Dual of linear regression
  - Find a line that passes close to a set of points
  - Find a line that separates the two sets of points



Linear separators

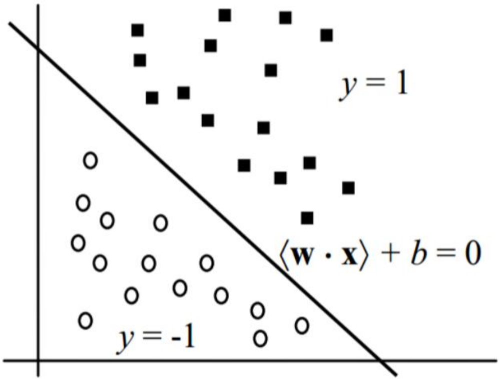
- Simplest case — linearly separable data
- Dual of linear regression
  - Find a line that passes close to a set of points
  - Find a line that separates the two sets of points
- Many lines are possible
  - How do we find the best one?
  - What is a good notion of “cost” to optimize?





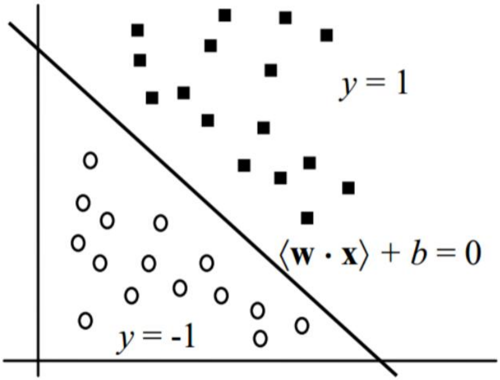
Linear separators

- Each input x has n attributes $\langle x_1, x_2, \ldots, x_n \rangle$



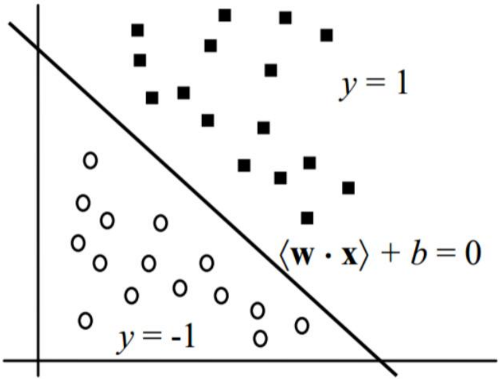
Linear separators

- Each input x has n attributes $\langle x_1, x_2, \ldots, x_n \rangle$
- Linear separator has the form $w_1x_1 + w_2x_2 + \cdots w_nx_n + b$



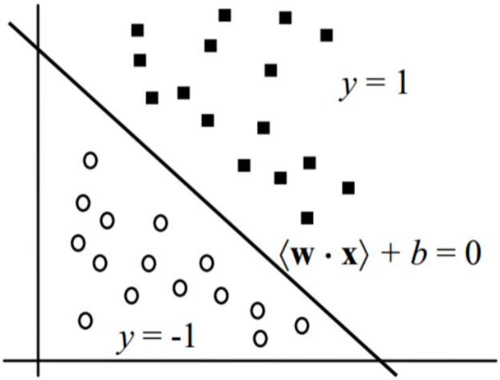
Linear separators

- Each input x has n attributes $\langle x_1, x_2, \ldots, x_n \rangle$
- Linear separator has the form $w_1x_1 + w_2x_2 + \cdots w_nx_n + b$
- Classification criterion
  - $w_1x_1 + w_2x_2 + \cdots w_nx_n + b > 0$, classify yes, $+1$
  - $w_1x_1 + w_2x_2 + \cdots w_nx_n + b < 0$, classify no, -1



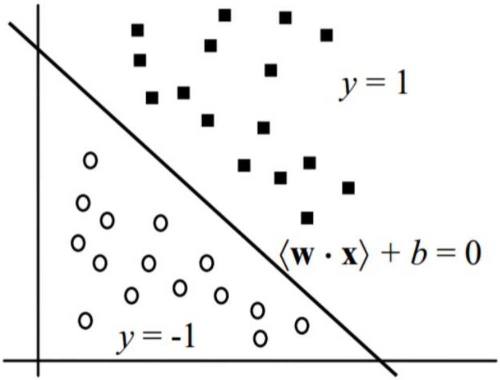
Linear separators

- Dot product $w \cdot x$
  $\langle w_1, w_2, \ldots, w_n \rangle \cdot \langle x_1, x_2, \ldots, x_n \rangle =$
  $w_1x_1 + w_2x_2 + \cdots + w_nx_n$



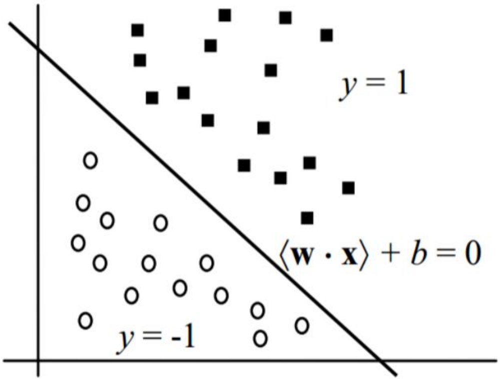
Linear separators

- Dot product $w \cdot x$
  $\langle w_1, w_2, \ldots, w_n \rangle \cdot \langle x_1, x_2, \ldots, x_n \rangle =$
  $w_1 x_1 + w_2 x_2 + \cdots + w_n x_n$
- Collapsed form
  $w \cdot x + b > 0$, $w \cdot x + b < 0$



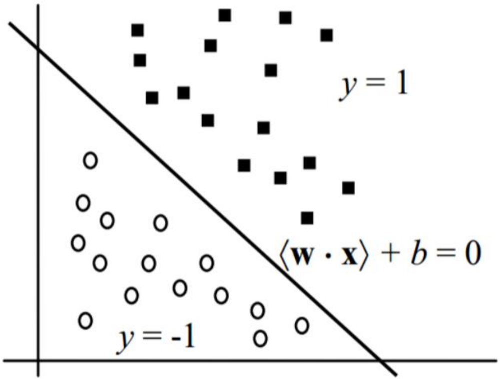
Linear separators

- Dot product $w \cdot x$
  $\langle w_1, w_2, \ldots, w_n \rangle \cdot \langle x_1, x_2, \ldots, x_n \rangle =$
  $w_1 x_1 + w_2 x_2 + \cdots + w_n x_n$
- Collapsed form
  $w \cdot x + b > 0$, $w \cdot x + b < 0$
- Rename bias b as w_0, create fictitious
  $x_0 = 1$



Linear separators

- Dot product $w \cdot x$
  $\langle w_1, w_2, \ldots, w_n \rangle \cdot \langle x_1, x_2, \ldots, x_n \rangle = w_1 x_1 + w_2 x_2 + \cdots + w_n x_n$
- Collapsed form
  $w \cdot x + b > 0$, $w \cdot x + b < 0$
- Rename bias b as w_0, create fictitious $x_0 = 1$
- Classification criteria become
  $w \cdot x > 0$, $w \cdot x < 0$



Perceptron algorithm

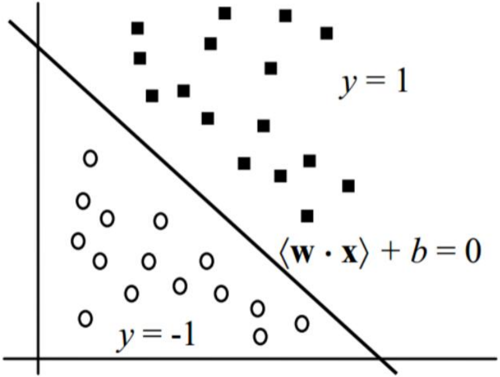
(Frank Rosenblatt, 1958)

- Each training input is (x_i, y_i), where $x_i = \langle x_{i_1}, x_{i_2}, \ldots, x_{i_n} \rangle$ and $y_i = +1$ or -1
- Need to find $w = \langle w_0, w_1, \ldots, w_n \rangle$
  - Recall $x_{i_0} = 1$, always

W·x < 0 if y = −1

W·x > 0 if y = +1

(W·x)y > 0 always



(Frank Rosenblatt, 1958)

- Each training input is (x_i, y_i), where $x_i = \langle x_{i_1}, x_{i_2}, \ldots, x_{i_n} \rangle$ and $y_i = +1$ or -1
- Need to find $w = \langle w_0, w_1, \ldots, w_n \rangle$
  - Recall $x_{i_0} = 1$, always

Initialize $w = \langle 0, 0, \ldots, 0 \rangle$

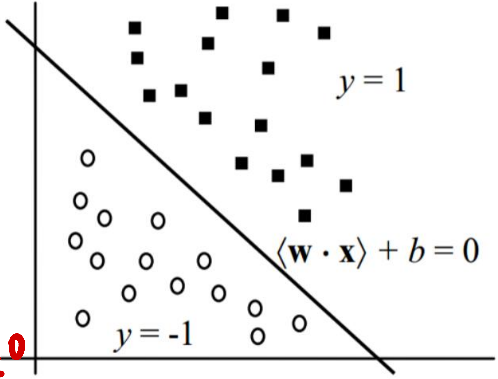
While there exists x_i, y_i *such that*

$y_i = +1$ *and* $w \cdot x_i < 0$, *or*

$y_i = -1$ *and* $w \cdot x_i > 0$

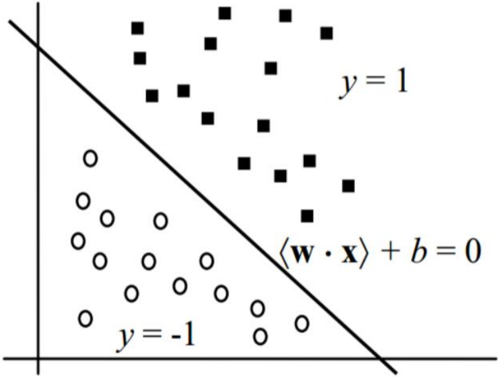
Update w *to* $w + x_i y_i$

If $w \cdot x_i y_i \leq 0$ for any input (x_i, y_i)



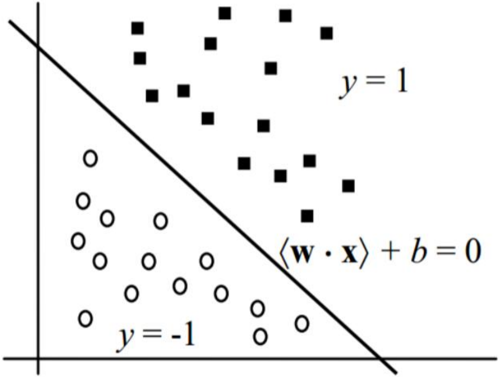
Perceptron algorithm ...

- Keep updating w as long as some training data item is misclassified
- Update is an offset by misclassified input



Perceptron algorithm . . .

- Keep updating w as long as some training data item is misclassified
- Update is an offset by misclassified input
- Need not stabilize, potentially an infinite loop

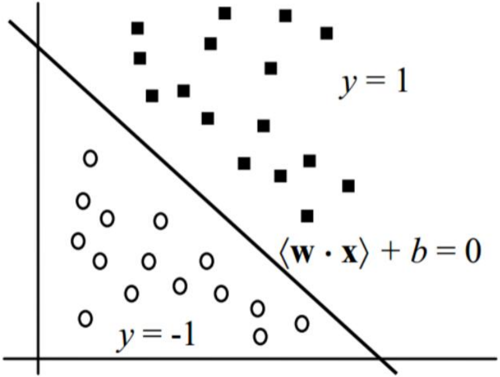


Perceptron algorithm . . .

- Keep updating w as long as some training data item is misclassified
- Update is an offset by misclassified input
- Need not stabilize, potentially an infinite loop

Theorem

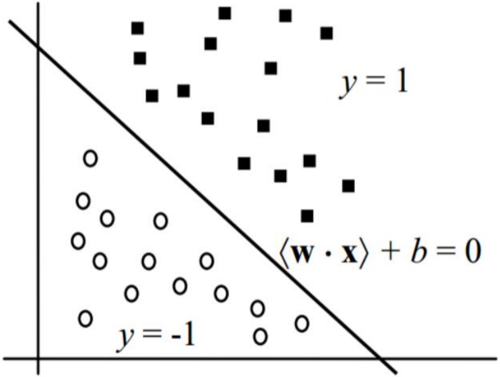
If the points are linearly separable, the Perceptron algorithm~~s~~ always terminates with a valid separator



Perceptron algorithm ...

Theorem

If the points are linearly separable, the Perceptron algorithms always terminates with a valid separator

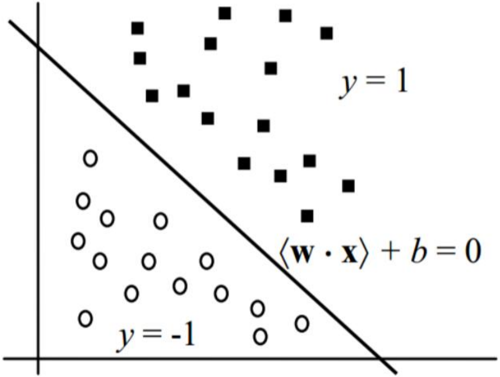


Perceptron algorithm . . .

Theorem

If the points are linearly separable, the Perceptron algorithms always terminates with a valid separator

- Termination time depends on two factors

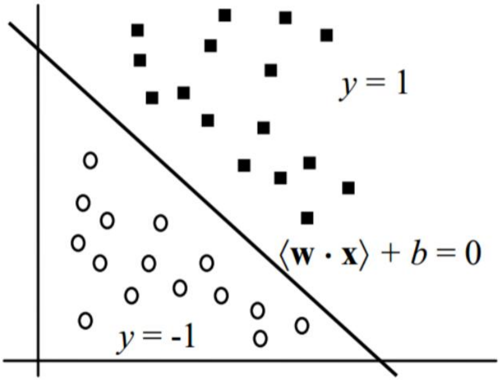


Perceptron algorithm . . .

Theorem

If the points are linearly separable, the Perceptron algorithms always terminates with a valid separator

- Termination time depends on two factors
  - Width of the band separating the positive and negative points
    - Narrow band takes longer to converge

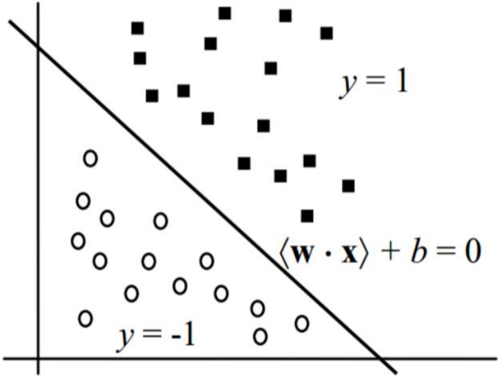


Perceptron algorithm . . .

Theorem

If the points are linearly separable, the Perceptron algorithms always terminates with a valid separator

- Termination time depends on two factors
  - Width of the band separating the positive and negative points
    - Narrow band takes longer to converge
  - Magnitude of the x values
    - Larger spread of points takes longer to converge



Perceptron Algorithm — Proof

$(w \cdot x) \cdot y \geq \cancel{0}\ 1$ always

Theorem

If there is w^* satisfying $(w^* \cdot x_i)y_i \geq 1$ for all i, then the Perceptron Algorithm finds a solution w with $(w \cdot x_i)y_i > 0$ for all i in at most $r^2|w^*|^2$ updates, where $r = \max_i |x_i|$.

Coordinate

inversely proportional to margin

Perceptron Algorithm — Proof

Theorem

If there is w^* satisfying $(w^* \cdot x_i)y_i \geq 1$ for all i, then the Perceptron Algorithm finds a solution w with $(w \cdot x_i)y_i > 0$ for all i in at most $r^2|w^*|^2$ updates, where $r = \max_i |x_i|$.

- Assume w^* exists. Keep track of two quantities: $w^\top w^*$, $|w|^2$.

ideal

current estimate

Perceptron Algorithm — Proof

Theorem

If there is w^* satisfying $(w^* \cdot x_i)y_i \geq 1$ for all i, then the Perceptron Algorithm finds a solution w with $(w \cdot x_i)y_i > 0$ for all i in at most $r^2|w^*|^2$ updates, where $r = \max_i |x_i|$.

- Assume w^* exists. Keep track of two quantities: $w^\top w^*$, $|w|^2$.
- Each update increases $w^\top w^*$ by at least 1.

$$\underbrace{(w + x_i y_i)}_{\text{updated value}}{}^\top w^* = w^\top w^* + \underbrace{x_i^\top y_i w^*}_{\text{always} \geq 1} \geq w^\top w^* + 1$$

Perceptron Algorithm — Proof

Theorem

If there is w^* satisfying $(w^* \cdot x_i)y_i \geq 1$ for all i, then the Perceptron Algorithm finds a solution w with $(w \cdot x_i)y_i > 0$ for all i in at most $r^2|w^*|^2$ updates, where $r = \max_i |x_i|$.

$x_i^\top y_i w < 0$

- Assume w^* exists. Keep track of two quantities: $w^\top w^*$, $|w|^2$.
- Each update increases $w^\top w^*$ by at least 1.

$$(w + x_i y_i)^\top w^* = w^\top w^* + x_i^\top y_i w^* \geq w^\top w^* + 1$$

- Each update increases $|w|^2$ by at most r^2

$$\underline{(w + x_i y_i)}^\top \underline{(w + x_i y_i)} = \underline{|w|^2} + \cancel{2x_i^\top y_i w} + \underline{|x_i y_i|^2} \leq |w|^2 + |x_i|^2 \leq |w|^2 + r^2$$

  - Note that we update only when $x_i^\top y_i w < 0$

Perceptron Algorithm — Proof (cont'd)

- Assume Perceptron Algorithm makes m updates

Perceptron Algorithm — Proof (cont'd)

- Assume Perceptron Algorithm makes m updates
- Then, $w^\top w^* \geq m$, $|w|^2 \leq mr^2$

Perceptron Algorithm — Proof (cont'd)

- Assume Perceptron Algorithm makes m updates
- Then, $w^\top w^* \geq m$, $|w|^2 \leq mr^2$
- $m \leq |w||w^*|$, because $a \cdot b = |a||b| \cos\theta$

Perceptron Algorithm — Proof (cont'd)

- Assume Perceptron Algorithm makes m updates
- Then, $w^\top w^* \geq m$, $\underline{|w|^2} \leq \underline{mr^2}$ $\quad |w| \leq \sqrt{m} \cdot r$
- $$\begin{aligned} m &\leq |w||w^*| \\ m/|w^*| &\leq |w| \end{aligned}$$

$\leq$

Perceptron Algorithm — Proof (cont'd)

- Assume Perceptron Algorithm makes m updates
- Then, $w^\top w^* \geq m$, $|w|^2 \leq mr^2$
- $$\begin{aligned} m &\leq |w||w^*| \\ m/|w^*| &\leq |w| \\ m/|w^*| &\leq r\sqrt{m}, \text{ because } |w|^2 \leq mr^2 \end{aligned}$$

Perceptron Algorithm — Proof (cont'd)

- Assume Perceptron Algorithm makes m updates
- Then, $w^\top w^* \geq m$, $|w|^2 \leq mr^2$
- $$\begin{aligned} m &\leq |w||w^*| \\ m/|w^*| &\leq |w| \\ m/|w^*| &\leq r\sqrt{m} \\ \sqrt{m} &\leq r|w^*| \end{aligned}$$

Perceptron Algorithm — Proof (cont'd)

- Assume Perceptron Algorithm makes m updates
- Then, $w^\top w^* \geq m$, $|w|^2 \leq mr^2$
- $$\begin{aligned} m &\leq |w||w^*| \\ m/|w^*| &\leq |w| \\ m/|w^*| &\leq r\sqrt{m} \\ \sqrt{m} &\leq r|w^*| \\ m &\leq r^2|w^*|^2 \end{aligned}$$

Perceptron Algorithm — Proof (cont'd)

- Assume Perceptron Algorithm makes m updates
- Then, $w^\top w^* \geq m$, $|w|^2 \leq mr^2$
- $$\begin{aligned} m &\leq |w||w^*| \\ m/|w^*| &\leq |w| \\ m/|w^*| &\leq r\sqrt{m} \\ \sqrt{m} &\leq r|w^*| \\ m &\leq r^2|w^*|^2 \end{aligned}$$

- Note (for later) that final w is of the form $\sum_i n_i x_i$

$0 \pm x_{i_1} \pm x_{i_2} - -$

Linear separators

- Simplest case — linearly separable data
- Perceptron algorithm is a simple procedure to find a linear separator, if one exists
- Many lines are possible
  - Does the Perceptron algorithm find the best one?
  - What is a good notion of "cost" to optimize?