Binary Classification / Perceptron

Nicholas Ruozzi
University of Texas at Dallas

Slides adapted from David Sontag and Vibhav Gogate

Supervised Learning

- Input: $(x_1, y_1), \ldots, (x_n, y_n)$
  - x_i is the i^{th} data item and y_i is the i^{th} **label**
- Goal: find a function f such that $f(x_i)$ is a "good approximation" to y_i
  - Can use it to predict y values for previously unseen x values

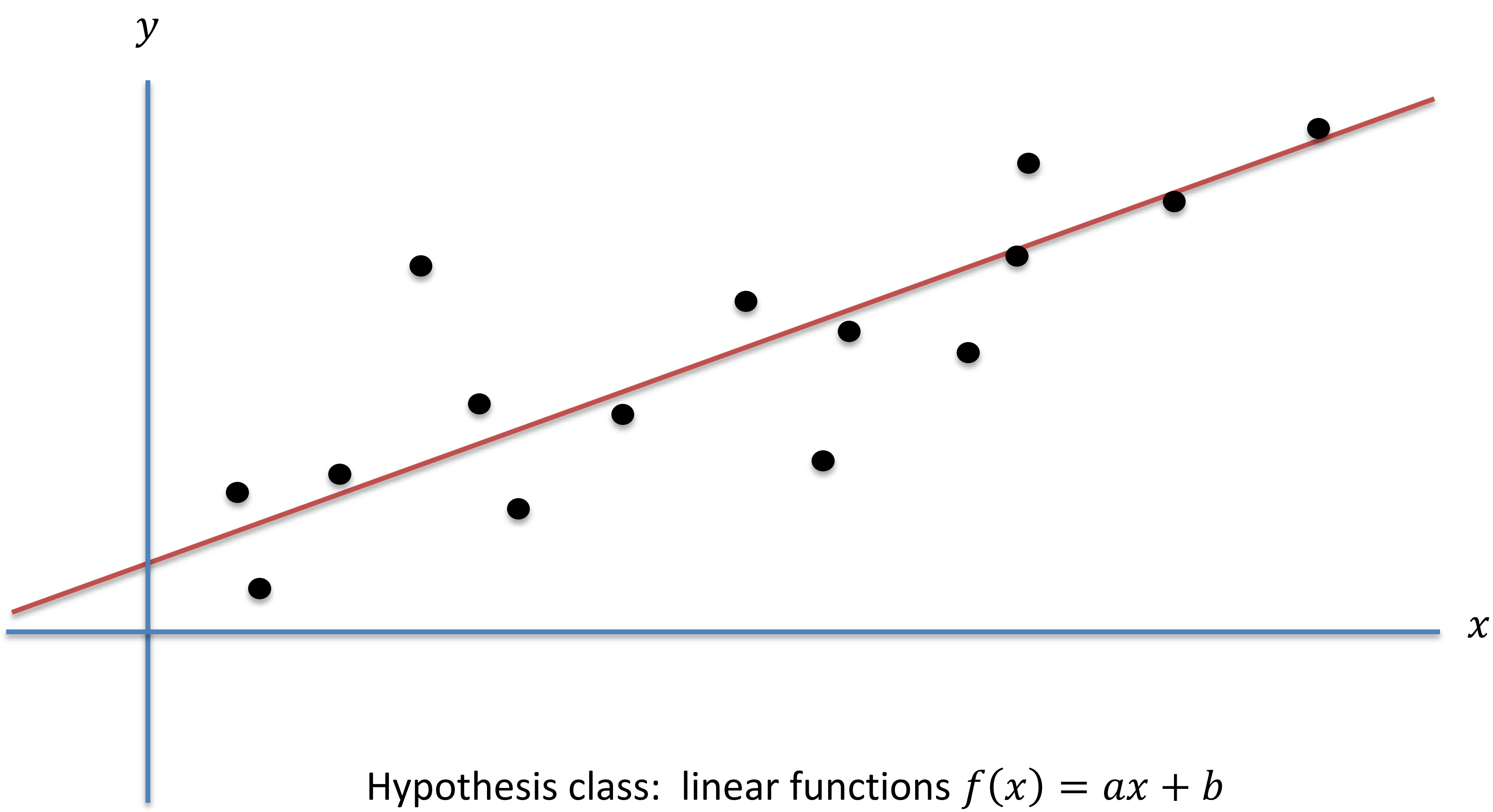
Examples of Supervised Learning

- Spam email detection
- Handwritten digit recognition
- Stock market prediction
- More?

Supervised Learning

- Hypothesis space: set of allowable functions $f : X \to Y$
- Goal: find the "best" element of the hypothesis space
  - How do we measure the quality of f?

Regression



Hypothesis class: linear functions $f(x) = ax + b$

Squared loss function used to measure the error of the approximation

Linear Regression

- In typical regression applications, measure the fit using a squared loss function

$$L(f, y_i) = (f(x_i) - y_i)^2$$

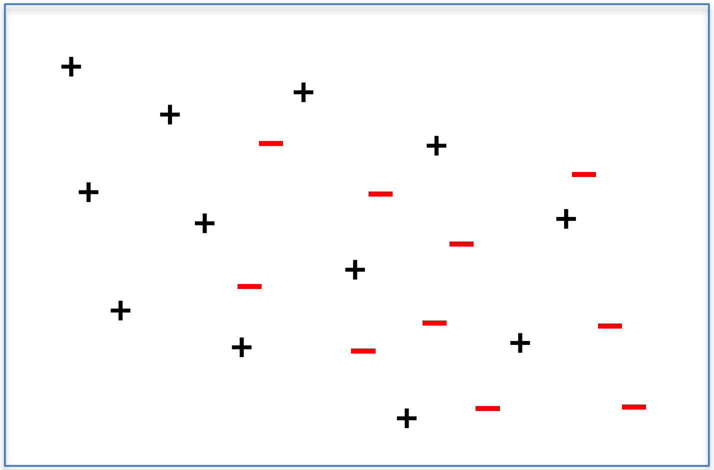
- Want to minimize the average loss on the training data
- For linear regression, the learning problem is then

$$\min_{a,b} \frac{1}{n} \sum_i (ax_i + b - y_i)^2$$

- For an unseen data point, x, the learning algorithm predicts $f(x)$

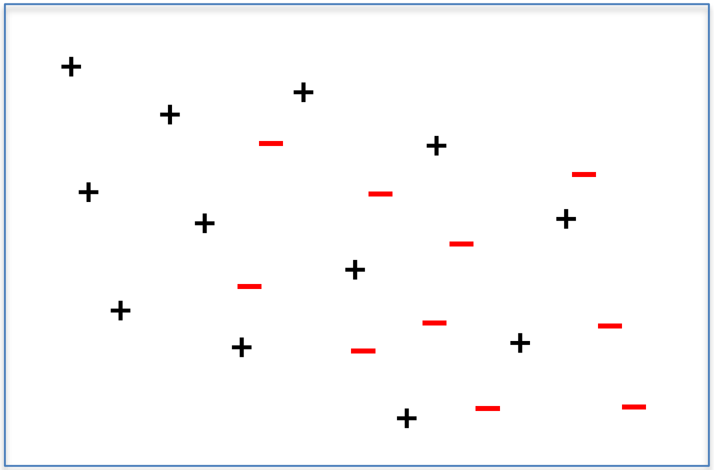
Binary Classification

- Input $(x^{(1)}, y_1), \ldots, (x^{(n)}, y_n)$ with $x_i \in \mathbb{R}^m$ and $y_i \in \{-1, +1\}$
- We can think of the observations as points in $\mathbb{R}^m$ with an associated sign (either +/- corresponding to 0/1)
- An example with $m = 2$



Binary Classification

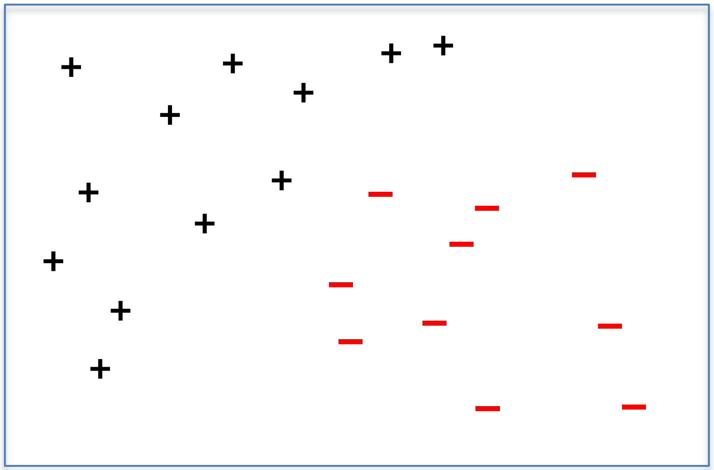
- Input $(x^{(1)}, y_1), \ldots, (x^{(n)}, y_n)$ with $x_i \in \mathbb{R}^m$ and $y_i \in \{-1, +1\}$
- We can think of the observations as points in $\mathbb{R}^m$ with an associated sign (either +/- corresponding to 0/1)
- An example with $m = 2$



What is a good hypothesis space for this problem?

Binary Classification

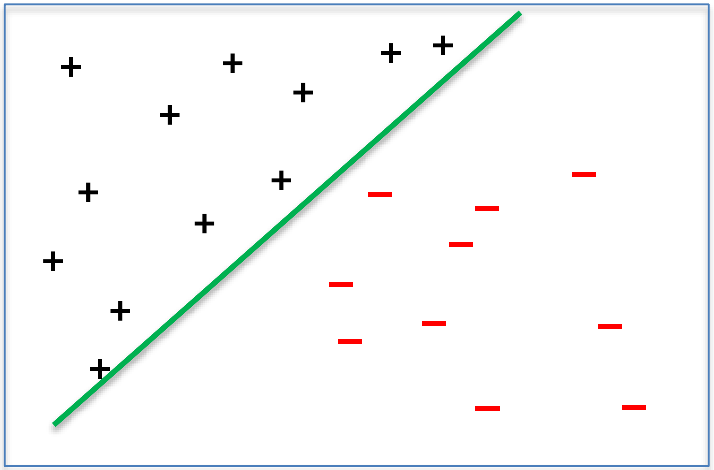
- Input $(x^{(1)}, y_1), \ldots, (x^{(n)}, y_n)$ with $x_i \in \mathbb{R}^m$ and $y_i \in \{-1, +1\}$
- We can think of the observations as points in $\mathbb{R}^m$ with an associated sign (either +/- corresponding to 0/1)
- An example with $m = 2$



What is a good hypothesis space for this problem?

Binary Classification

- Input $(x^{(1)}, y_1), \ldots, (x^{(n)}, y_n)$ with $x_i \in \mathbb{R}^m$ and $y_i \in \{-1, +1\}$
- We can think of the observations as points in $\mathbb{R}^m$ with an associated sign (either +/- corresponding to 0/1)
- An example with $m = 2$



What is a good hypothesis space for this problem?

In this case, we say that the observations are linearly separable

Linear Separators

- In n dimensions, a hyperplane is a solution to the equation

$$w^T x + b = 0$$

with $w \in \mathbb{R}^n, b \in \mathbb{R}$

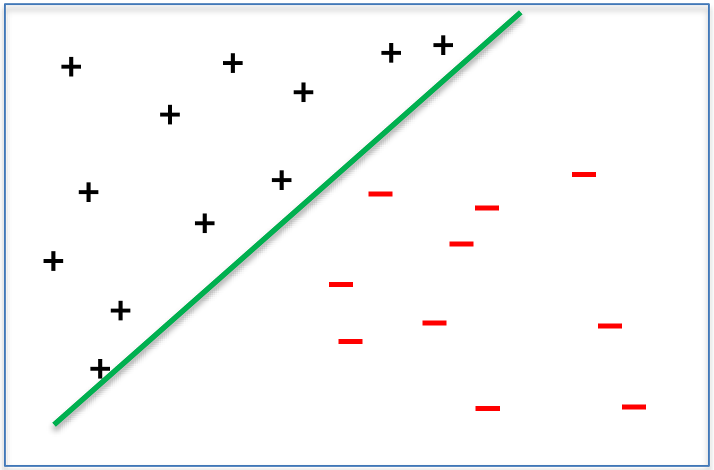
- Hyperplanes divide $\mathbb{R}^n$ into two distinct sets of points (called halfspaces)

$$w^T x + b > 0$$

$$w^T x + b < 0$$

Binary Classification

- Input $(x^{(1)}, y_1), \ldots, (x^{(n)}, y_n)$ with $x_i \in \mathbb{R}^m$ and $y_i \in \{-1, +1\}$
- We can think of the observations as points in $\mathbb{R}^m$ with an associated sign (either +/- corresponding to 0/1)
- An example with $m = 2$



What is a good hypothesis space for this problem?

In this case, we say that the observations are linearly separable

The Linearly Separable Case

- Input $(x^{(1)}, y_1), \dots, (x^{(n)}, y_n)$ with $x_i \in \mathbb{R}^m$ and $y_i \in \{-1,1\}$
- Hypothesis space: separating hyperplanes

$$f(x) = w^T x + b$$

- How should we choose the loss function?

The Linearly Separable Case

- Input $(x^{(1)}, y_1), \ldots, (x^{(n)}, y_n)$ with $x_i \in \mathbb{R}^m$ and $y_i \in \{-1,1\}$
- Hypothesis space: separating hyperplanes

$$f_{w,b}(x) = w^T x + b$$

- How should we choose the loss function?
  - Count the number of misclassifications

$$loss = \sum_i |y_i - sign(f_{w,b}(x^{(i)}))|$$

    - Tough to optimize, gradient contains no information

The Linearly Separable Case

- Input $(x^{(1)}, y_1), \ldots, (x^{(n)}, y_n)$ with $x_i \in \mathbb{R}^m$ and $y_i \in \{-1,1\}$
- Hypothesis space: separating hyperplanes

$$f_{w,b}(x) = w^T x + b$$

- How should we choose the loss function?
  - Penalize each misclassification by the size of the violation

$$perceptron\ loss = \sum_i \max\{0, -y_i f_{w,b}(x^{(i)})\}$$

    - Modified hinge loss (this loss is convex, but not differentiable)

The Perceptron Algorithm

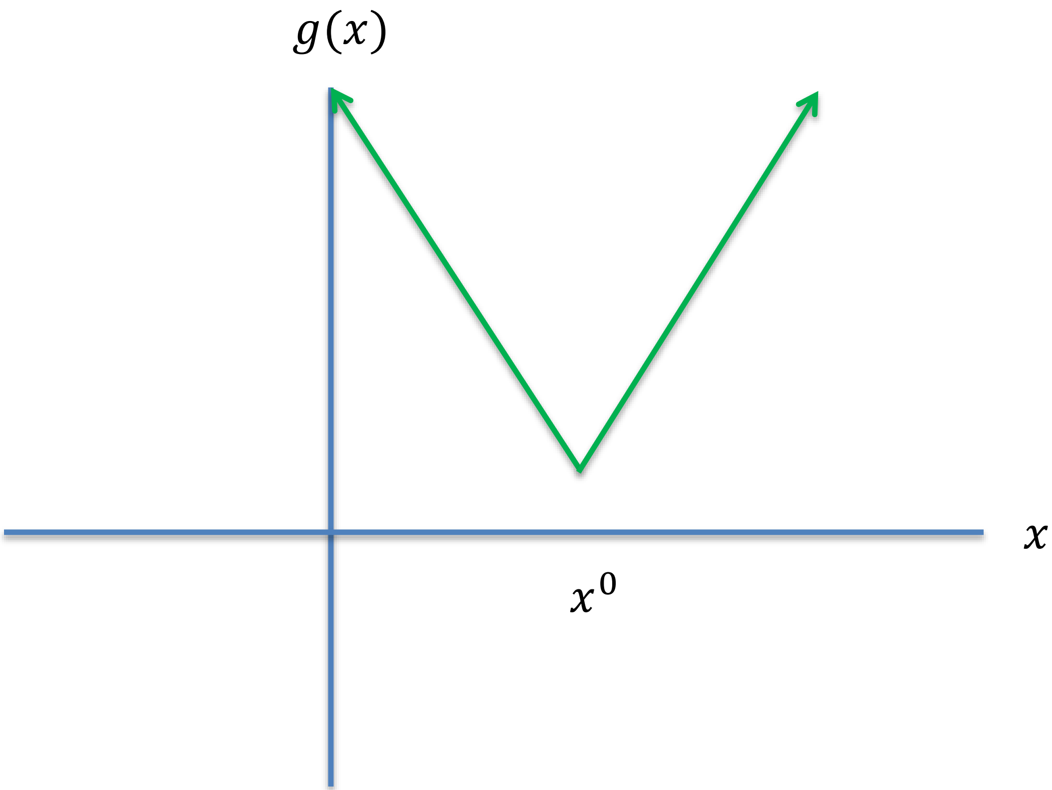
- Try to minimize the perceptron loss using (sub)gradient descent

$$\nabla_w(perceptron\ loss) = \sum_{i:-y_i f_{w,b}(x^{(i)}) \geq 0} -y_i x^{(i)}$$

$$\nabla_b(perceptron\ loss) = \sum_{i:-y_i f_{w,b}(x^{(i)}) \geq 0} -y_i$$

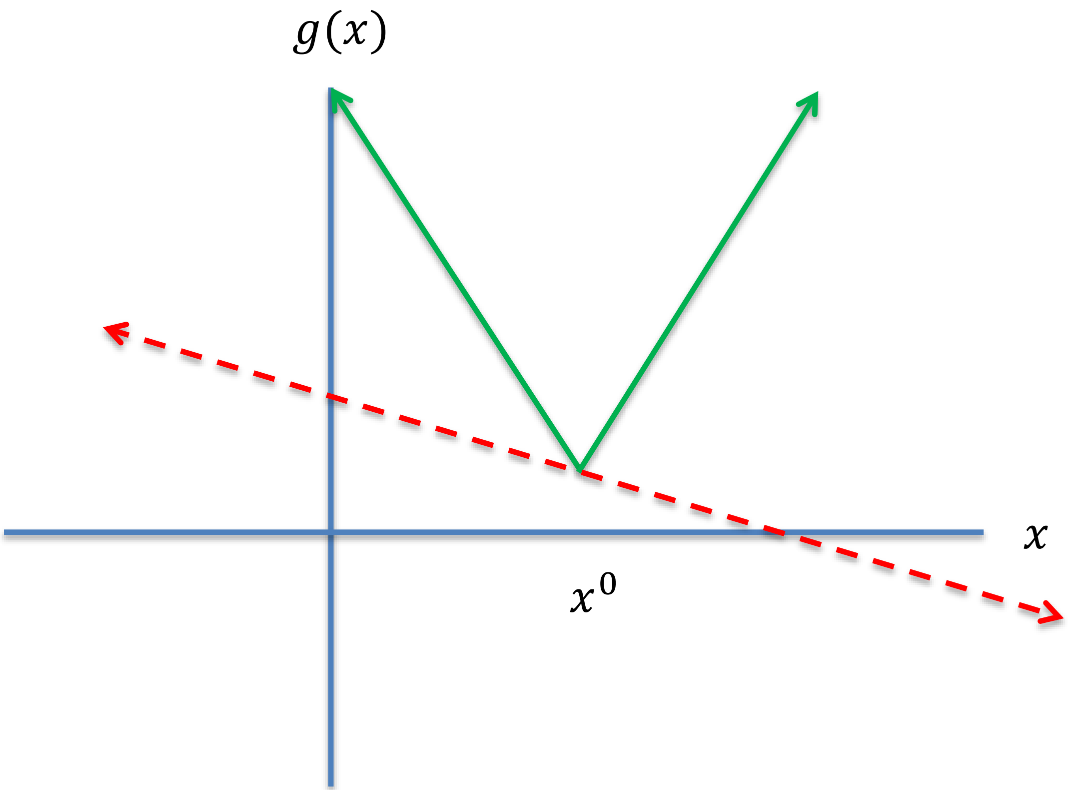
Subgradients

- For a convex function $g(x)$, a subgradient at a point x^0 is any tangent line/plane through the point x^0 that underestimates the function everywhere



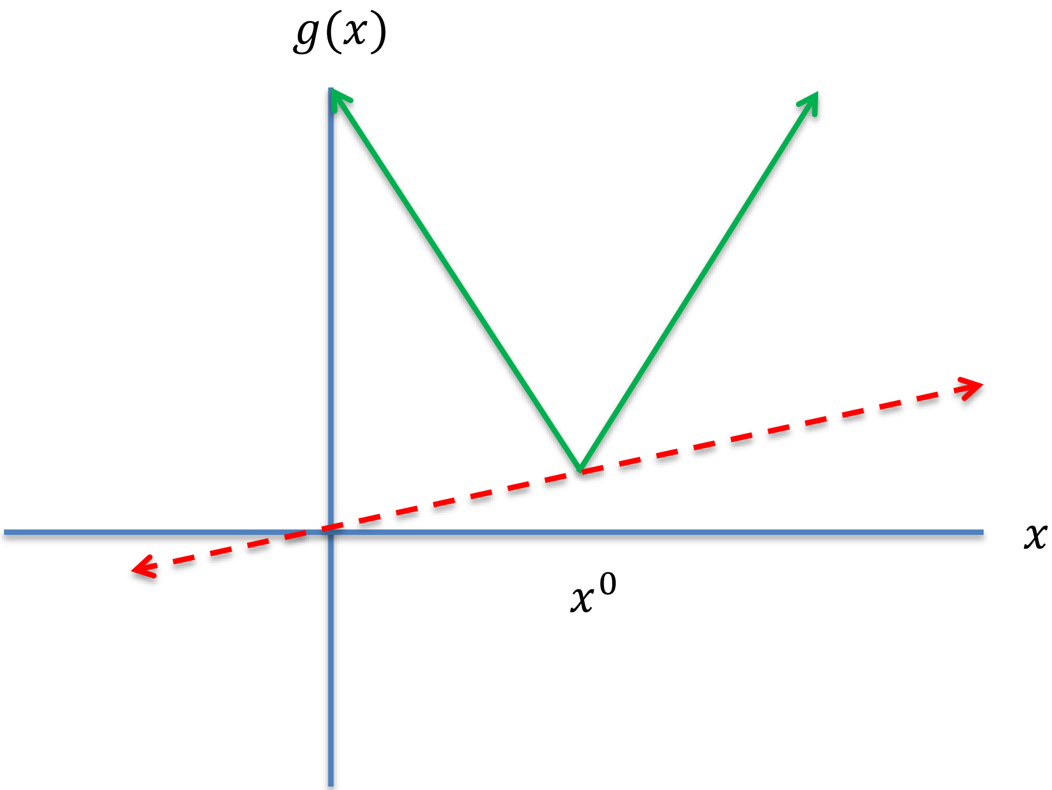
Subgradients

- For a convex function $g(x)$, a subgradient at a point x^0 is any tangent line/plane through the point x^0 that underestimates the function everywhere



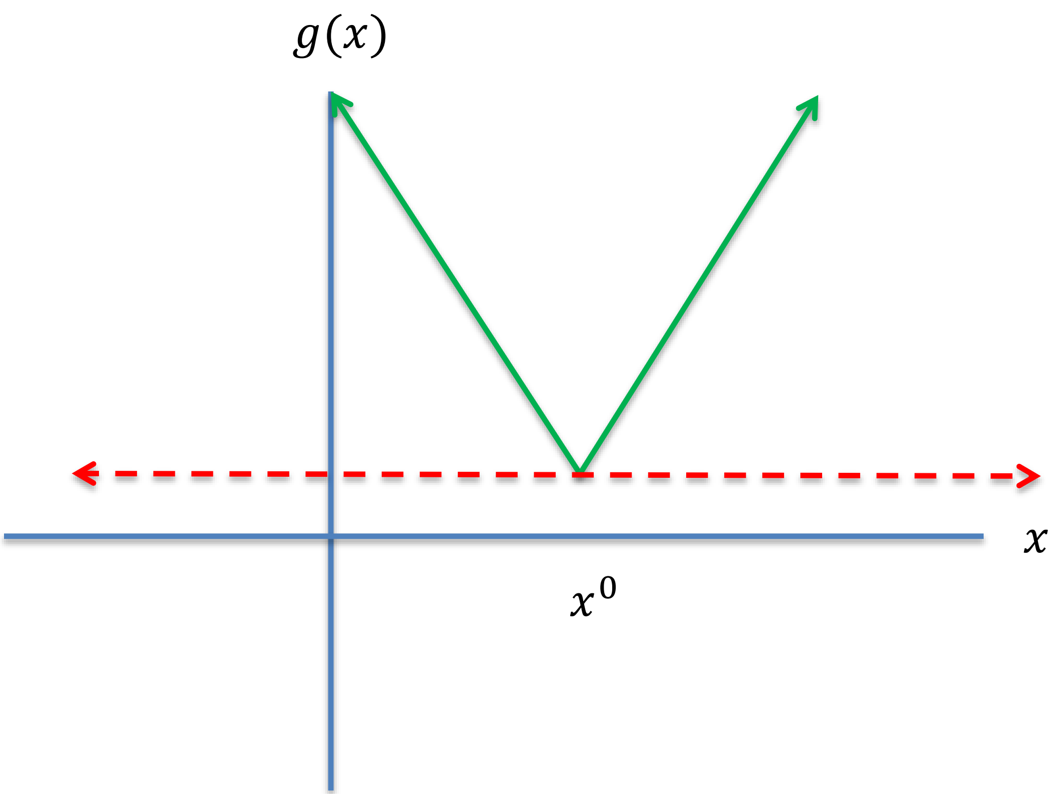
Subgradients

- For a convex function $g(x)$, a subgradient at a point x^0 is any tangent line/plane through the point x^0 that underestimates the function everywhere



Subgradients

- For a convex function $g(x)$, a subgradient at a point x^0 is any tangent line/plane through the point x^0 that underestimates the function everywhere



If $\vec{0}$ is subgradient at x^0, then x^0 is a global minimum

The Perceptron Algorithm

- Try to minimize the perceptron loss using (sub)gradient descent

$$w^{(t+1)} = w^{(t)} + \gamma_t \cdot \sum_{i:-y_i f_{w^{(t)},b^{(t)}}(x^{(i)}) \geq 0} y_i x^{(i)}$$

$$b^{(t+1)} = b^{(t)} + \gamma_t \cdot \sum_{i:-y_i f_{w^{(t)},b^{(t)}}\left(x^{(i)}\right) \geq 0} y_i$$

- With step size γ_t (sometimes called the learning rate)

Stochastic Gradient Descent

- To make the training more practical, stochastic gradient descent is used instead of standard gradient descent
- Approximate the gradient of a sum by sampling a few indices (as few as one) uniformly at random and averaging

$$\nabla_x \left[\sum_{i=1}^{n} g_i(x) \right] \approx \frac{1}{K} \sum_{k=1}^{K} \nabla_x g_{i_k}(x)$$

here, each i^k is sampled uniformly at random from $\{1, \dots, n\}$

- Stochastic gradient descent converges under certain assumptions on the step size

Stochastic Gradient Descent

- Setting $K = 1$, we can simply pick a random observation i and perform the following update if the i^{th} data point is misclassified

$$w^{(t+1)} = w^{(t)} + \gamma_t y_i x^{(i)}$$

$$b^{(t+1)} = b^{(t)} + \gamma_t y_i$$

and

$$w^{(t+1)} = w^{(t)}$$
$$b^{(t+1)} = b^{(t)}$$

otherwise

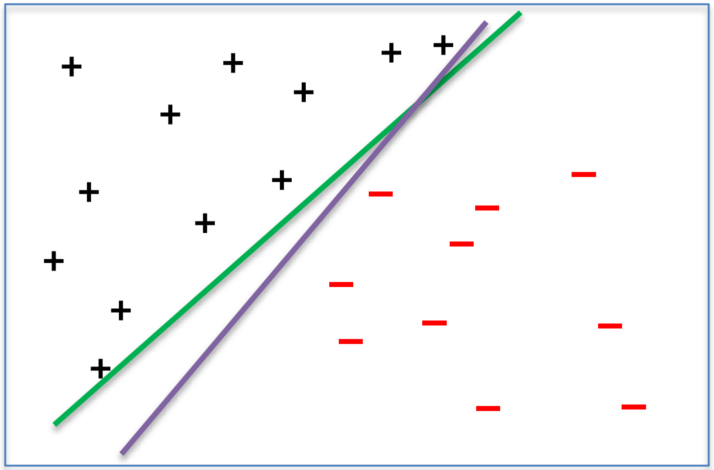
- Sometimes, you will see the perceptron algorithm specified with $\gamma_t = 1$ for all t

Application of Perceptron

- Spam email classification
  - Represent emails as vectors of counts of certain words (e.g., sir, madam, Nigerian, prince, money, etc.)
  - Apply the perceptron algorithm to the resulting vectors
  - To predict the label of an unseen email:
    - Construct its vector representation, x'
    - Check whether or not $w^T x' + b$ is positive or negative

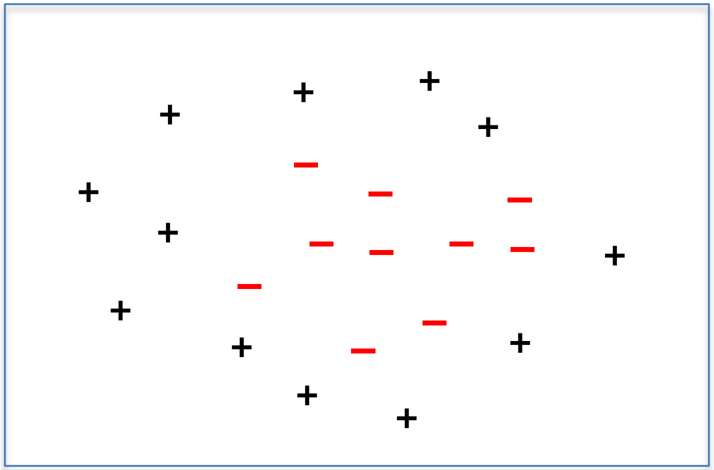
Perceptron Learning

- Drawbacks:
  - No convergence guarantees if the observations are not linearly separable
  - Can overfit
    - There are a number of perfect classifiers, but the perceptron algorithm doesn't have any mechanism for choosing between them



What If the Data Isn‘t Separable?

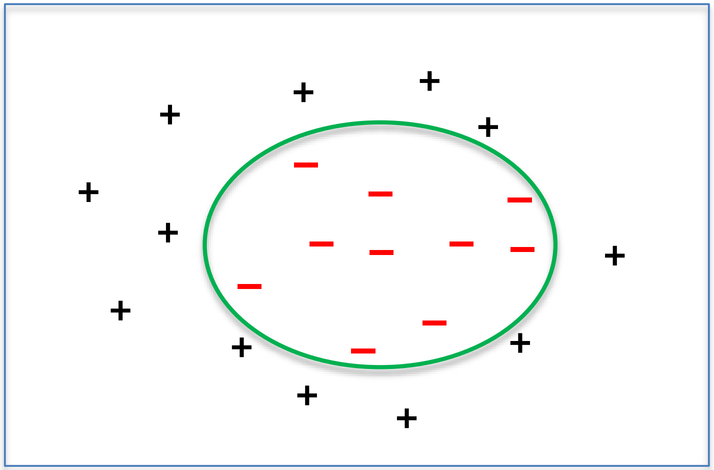
- Input $(x^{(1)}, y_1), \dots, (x^{(n)}, y_n)$ with $x_i \in \mathbb{R}^m$ and $y_i \in \{-1, +1\}$
- We can think of the observations as points in $\mathbb{R}^m$ with an associated sign (either +/- corresponding to 0/1)
- An example with $m = 2$



What is a good hypothesis space for this problem?

What If the Data Isn't Separable?

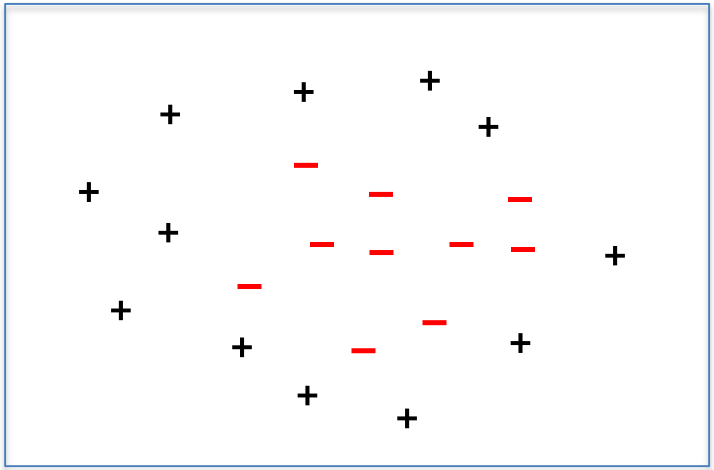
- Input $(x^{(1)}, y_1), \ldots, (x^{(n)}, y_n)$ with $x_i \in \mathbb{R}^m$ and $y_i \in \{-1, +1\}$
- We can think of the observations as points in $\mathbb{R}^m$ with an associated sign (either +/- corresponding to 0/1)
- An example with $m = 2$



What is a good hypothesis space for this problem?

Adding Features

- Perceptron algorithm only works for linearly separable data



Can add features to make the data linearly separable over a larger space!

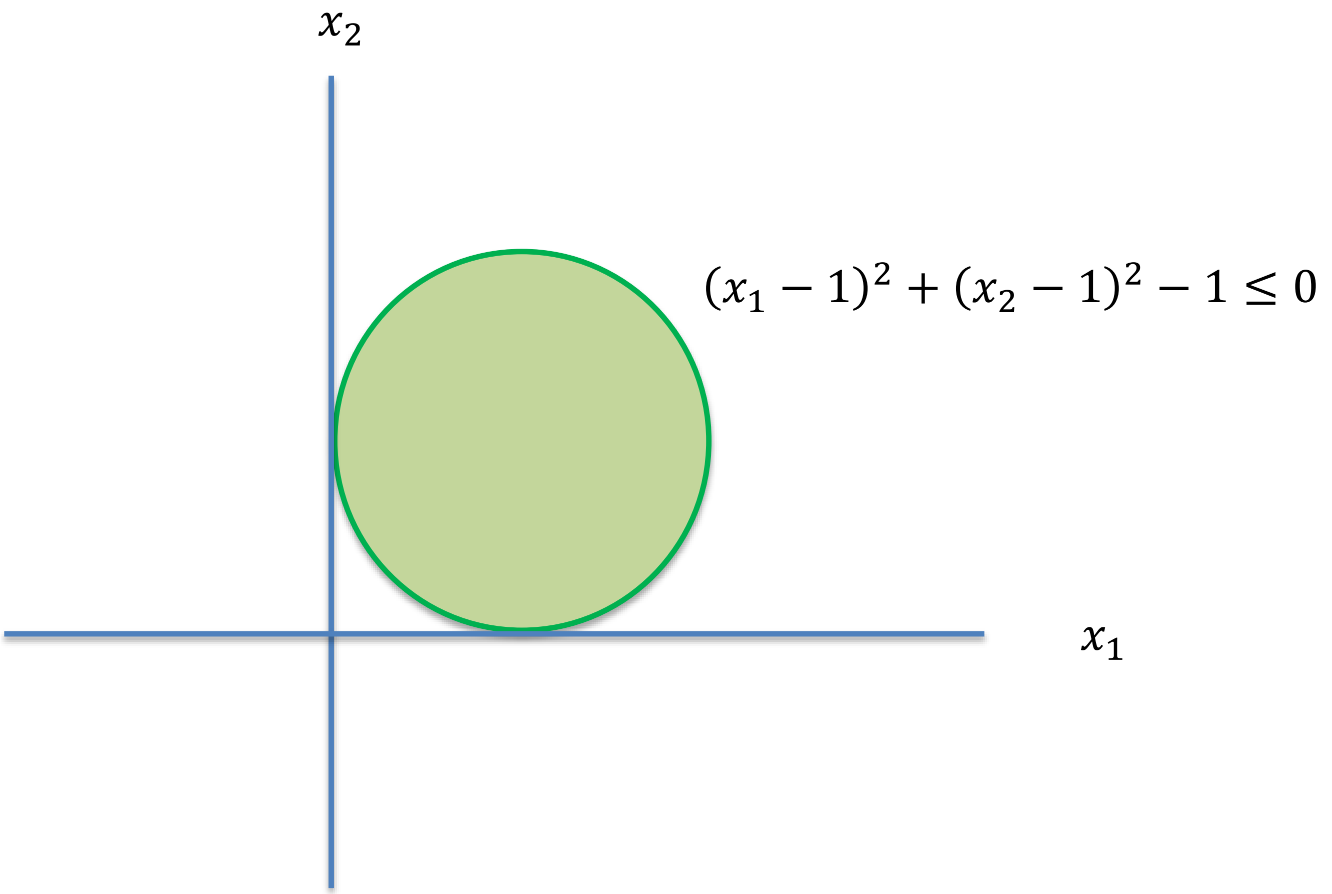
Adding Features

- The idea:
  - Given the observations $x^{(1)}, \dots, x^{(n)}$, construct a feature vector $\phi(x)$
  - Use $\phi\left(x^{(1)}\right), \dots, \phi\left(x^{(n)}\right)$ instead of $x^{(1)}, \dots, x^{(n)}$ in the learning algorithm
  - Goal is to choose ϕ so that $\phi\left(x^{(1)}\right), \dots, \phi\left(x^{(n)}\right)$ are linearly separable
  - Learn linear separators of the form $w^T\phi(x)$ (instead of $w^T x$)
- Warning: more expressive features can lead to overfitting

Adding Features

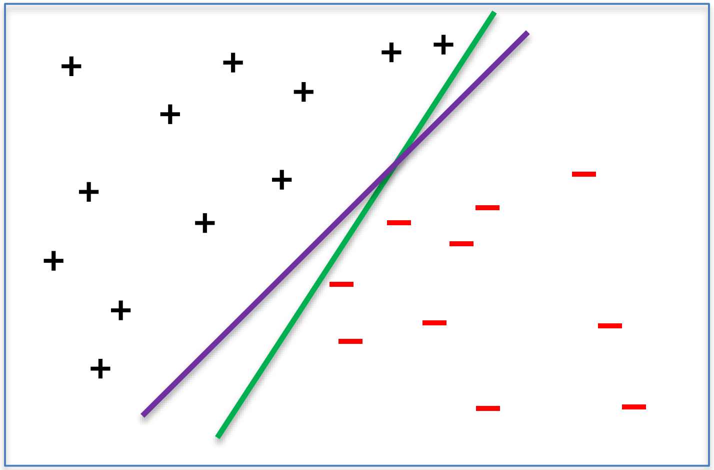
- Examples
  - $\phi(x_1, x_2) = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$
    - This is just the input data, without modification
  - $\phi(x_1, x_2) = \begin{bmatrix} 1 \\ x_1 \\ x_2 \\ x_1^2 \\ x_2^2 \end{bmatrix}$
    - This corresponds to a second degree polynomial separator, or equivalently, elliptical separators in the original space

Adding Features



Support Vector Machines

- How can we decide between two perfect classifiers?



- What is the practical difference between these two solutions?