Support Vector Machines

Algorithm: Perceptron

- Initialize w and η
- On each round
  - Receive example x
  - Predict $\hat{y} = sign(w \cdot x)$
  - Receive correct label $y \in \{-1, +1\}$
  - Suffer loss $\ell_{0/1}(\hat{y}, y)$
  - Update w: $w^{t+1} = w^t + \eta y_i x_i$

Perceptron

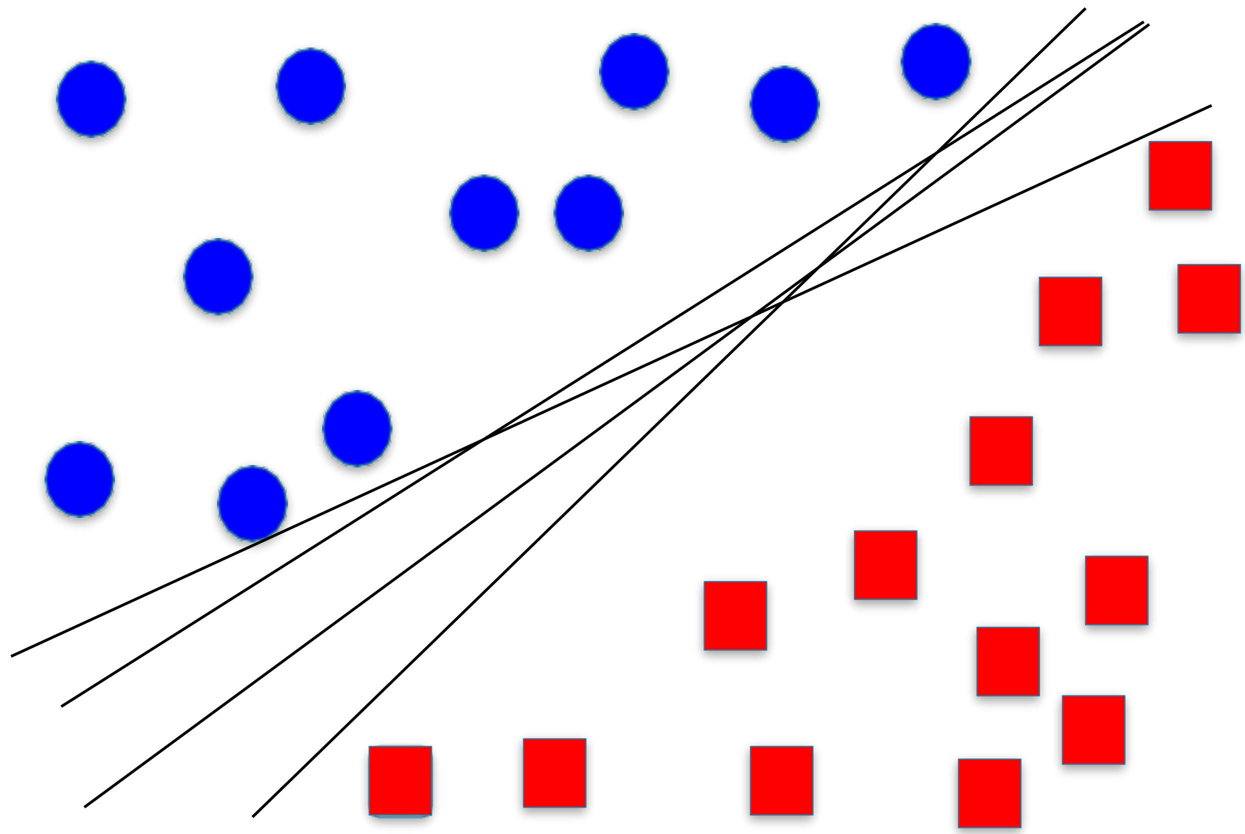
- **Fitting a function to data**

- Fitting: Stochastic gradient decent
- Function: 0/1 loss with linear function
- Data: Update using a single example at a time

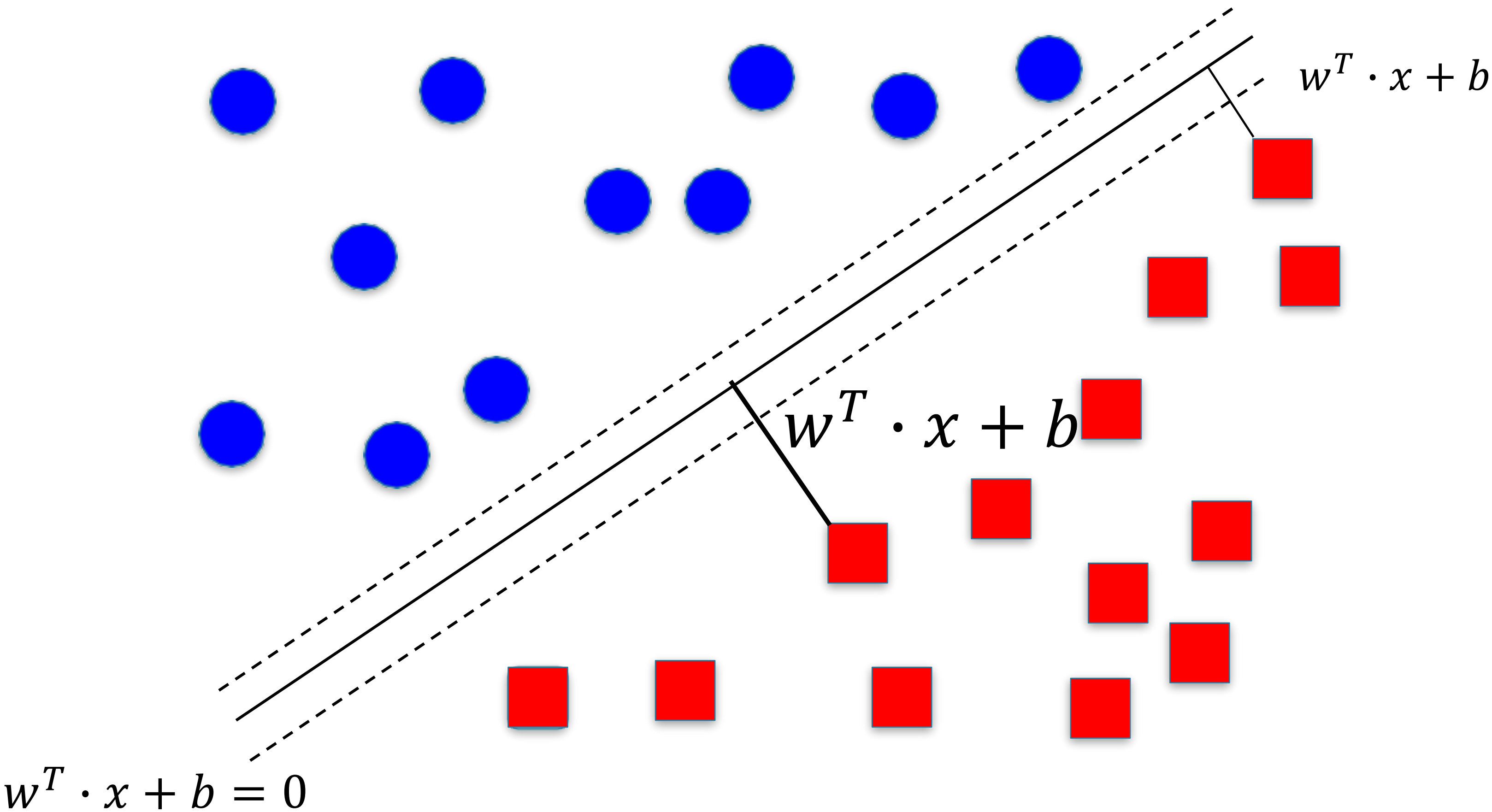
Questions

- Perceptron picks one separating hyperplane (of many)
  - What would we do if we saw all of the data (batch)?
  - We'd pick the best separating hyperplane!
- Which separating hyperplane is the best?
  - Let's look at the geometric model
- Better solutions for non-linear data?

Geometric Representation



The Margin



Functional Margin

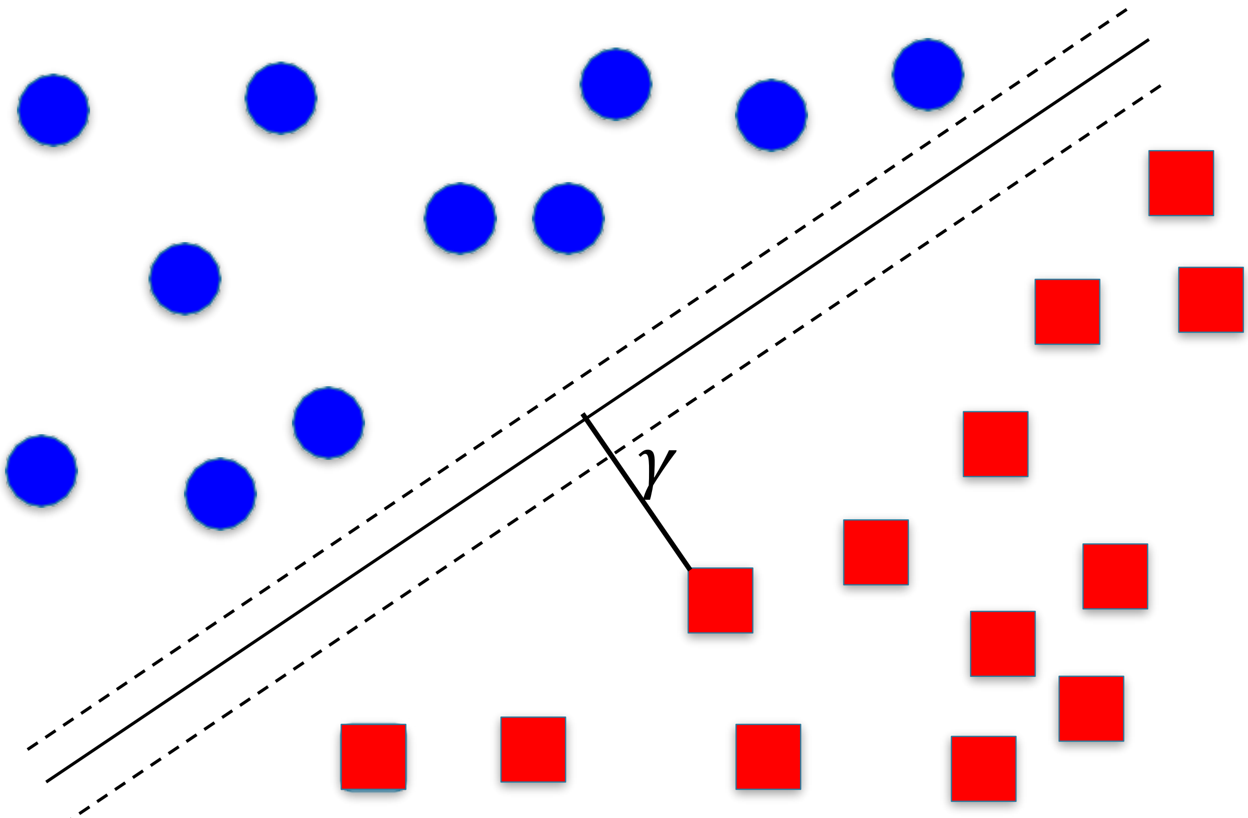
- Prediction and y should agree to get large margin
$$\widehat{\gamma_i} = y_i(w^T \cdot x_i + b)$$
- What if we double w?
$$\widehat{\gamma_i} = y_i(2w^T \cdot x_i + 2b)$$
- Doubles margin, but no practical change
  - We will address this in a moment

Functional Margin of Data

- Given a training set of size N:
  - Smallest margin

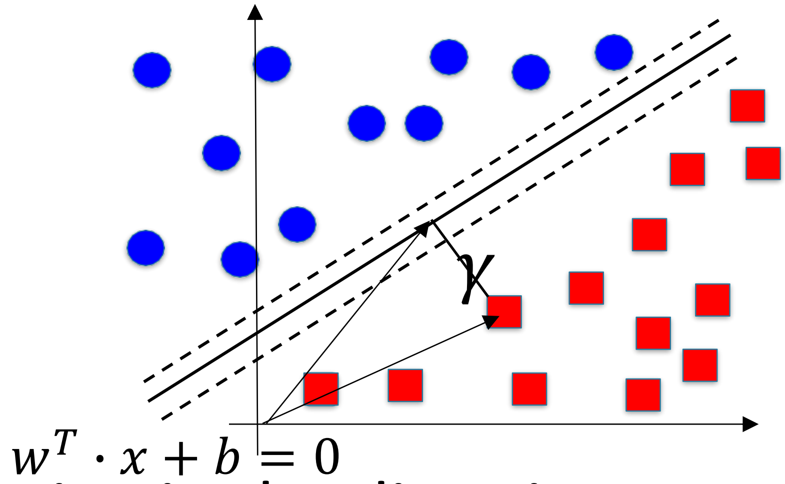
$$\hat{\gamma} = \min_{i=1,\ldots,N} \widehat{\gamma_i}$$

Geometric Margin



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

Geometric Margin



- Size of γ?
- $\frac{w}{\|w\|}$ is a unit length vector pointing in the direction of w
- γ intersects with the decision boundary at

$$x_i - \gamma_i \frac{w}{\|w\|}$$

and points on the boundary must give a prediction 0

$$\gamma_i = y_i(\left(\frac{w}{\|w\|}\right)^T x_i + \frac{b}{\|w\|})$$

if ||w|| = 1 then functional = geometric margin

Max-Margin Principle

- Assuming the observed data is linearly separable
- Select the hyperplane that separates the data with the maximal margin
- Why?
  - New examples are likely to be close to old examples
  - Gives the best generalization error on new data

Maximum Geometric Margin

$$\max_{\gamma, w, b} \gamma$$

$$s.t. \quad y_i(w^T x_i + b) \geq \gamma, i = 1, \ldots, N$$
$$\|w\| = 1$$

- Every training instance has margin at least γ
- ||w|| constraint means geometric = functional margin
- Problem: ||w|| constraint is non-convex!

Maximum Geometric Margin

- Functional and geometric related by $\gamma = \frac{\hat{\gamma}}{\|w\|}$

$$\max_{\hat{\gamma},w,b} \frac{\hat{\gamma}}{\|w\|}$$

$$s.t. \quad y_i(w^T x_i + b) \geq \hat{\gamma}, i = 1, \ldots, N$$

Maximum Geometric Margin

- Recall: we can arbitrarily scale w!
  - Arbitrarily set $\gamma = 1$

$$\min_{w,b} \frac{1}{2} \|w\|^2$$
$$s.t. \quad y_i(w^T x_i + b) \geq 1, i = 1, \dots, N$$

  - min $||w||^2$ same as max $1/||w||$
- Quadratic program (QP): quadratic objective with linear constraints

Support Vector Machines

- **Fitting a function to data**

- Fitting: Batch optimization method: QP solver
- Function: hyperplane with functional margin >= 1
  - New loss function?
- Data: Train in batch mode

SVM vs. Logistic Regression

- Both minimize the empirical loss with some regularization
- SVM:

$$\frac{1}{N}\sum_{i=1}^{N}(1-y_i[w\cdot x_i])^{+}+\lambda\frac{1}{2}\|w\|^2$$

- LR:

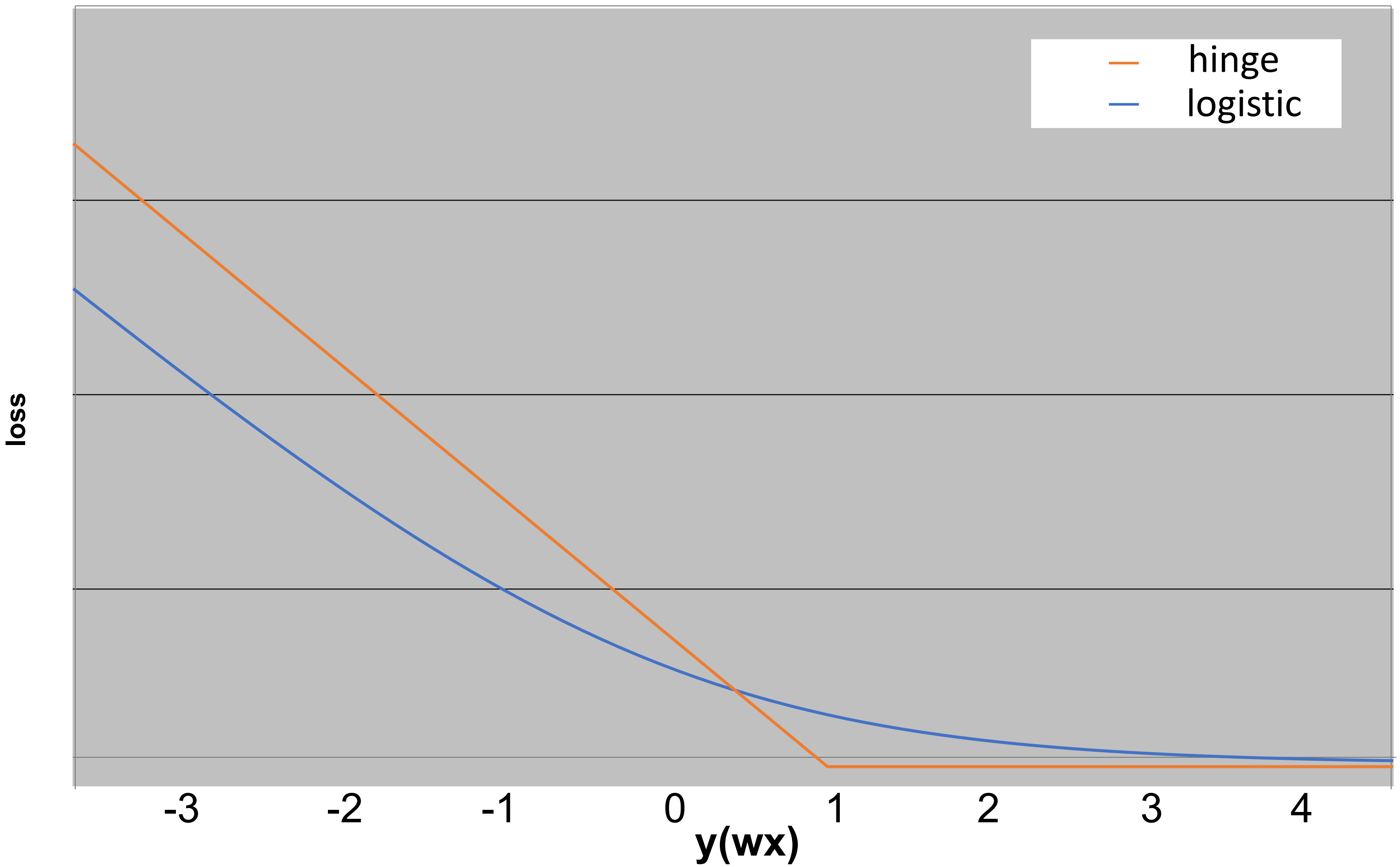
$$\frac{1}{N}\sum_{i=1}^{N}\underbrace{-\log(g(y_i[w\cdot x_i]))}_{-P(y_i|x_i,w)}+\lambda\frac{1}{2}\|w\|^2$$

  - (z)+ indicates only positive values
  - g(z) = (1+exp(-z))$^{-1}$ is the logistic function

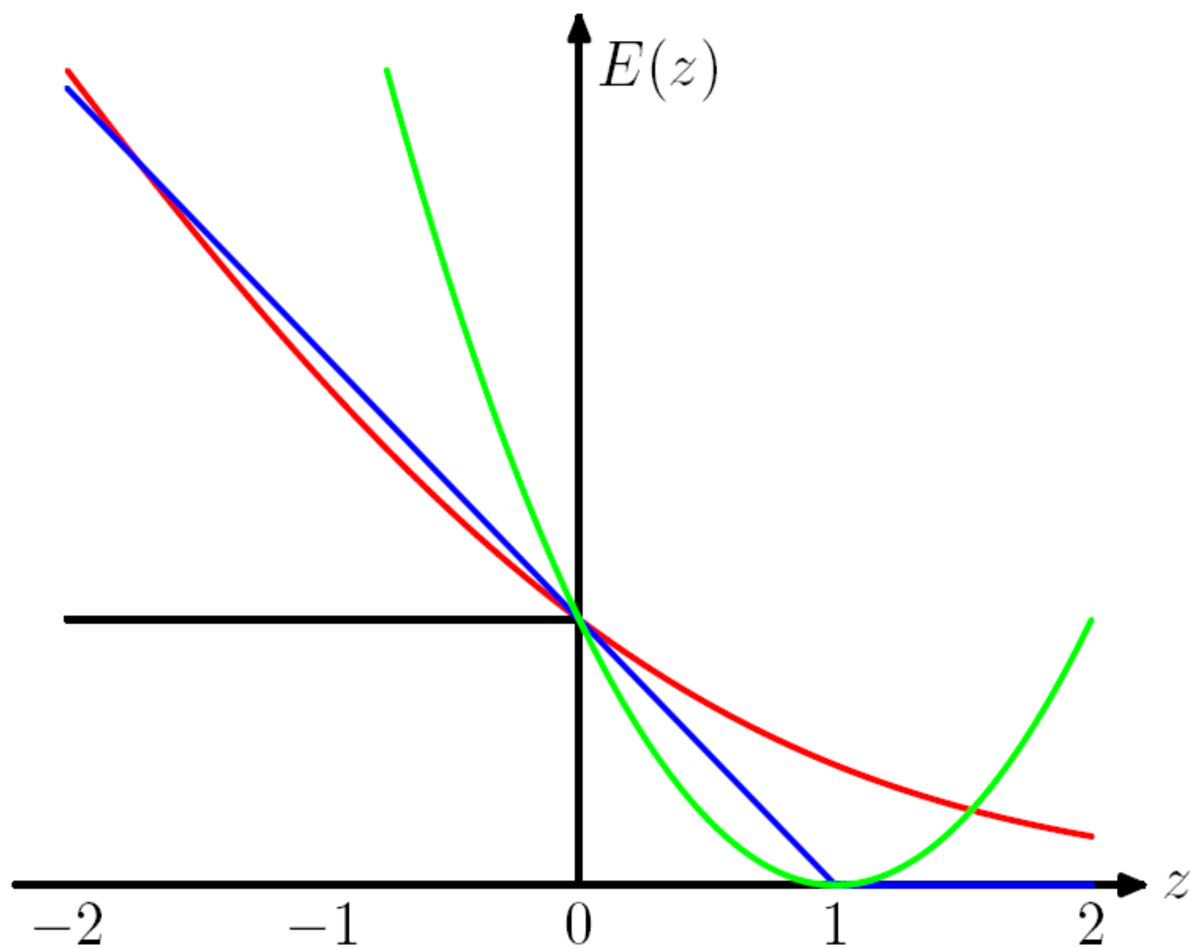
Loss Function

- Both minimize
$$\frac{1}{N}\sum_{i=1}^{N} \ell(y_i[w \cdot x_i]) + \lambda \frac{1}{2}\|w\|^2$$
- Different loss functions
- SVM: Hinge Loss
$$\ell(y_i[w \cdot x_i]) = \max(0,1 - y_i[w \cdot x_i])$$
- Logistic regression: Logistic loss
$$\ell(y_i[w \cdot x_i]) = \log(1 + \exp(-y_i[w \cdot x_i]))$$

Loss Function



Rethinking Loss Functions



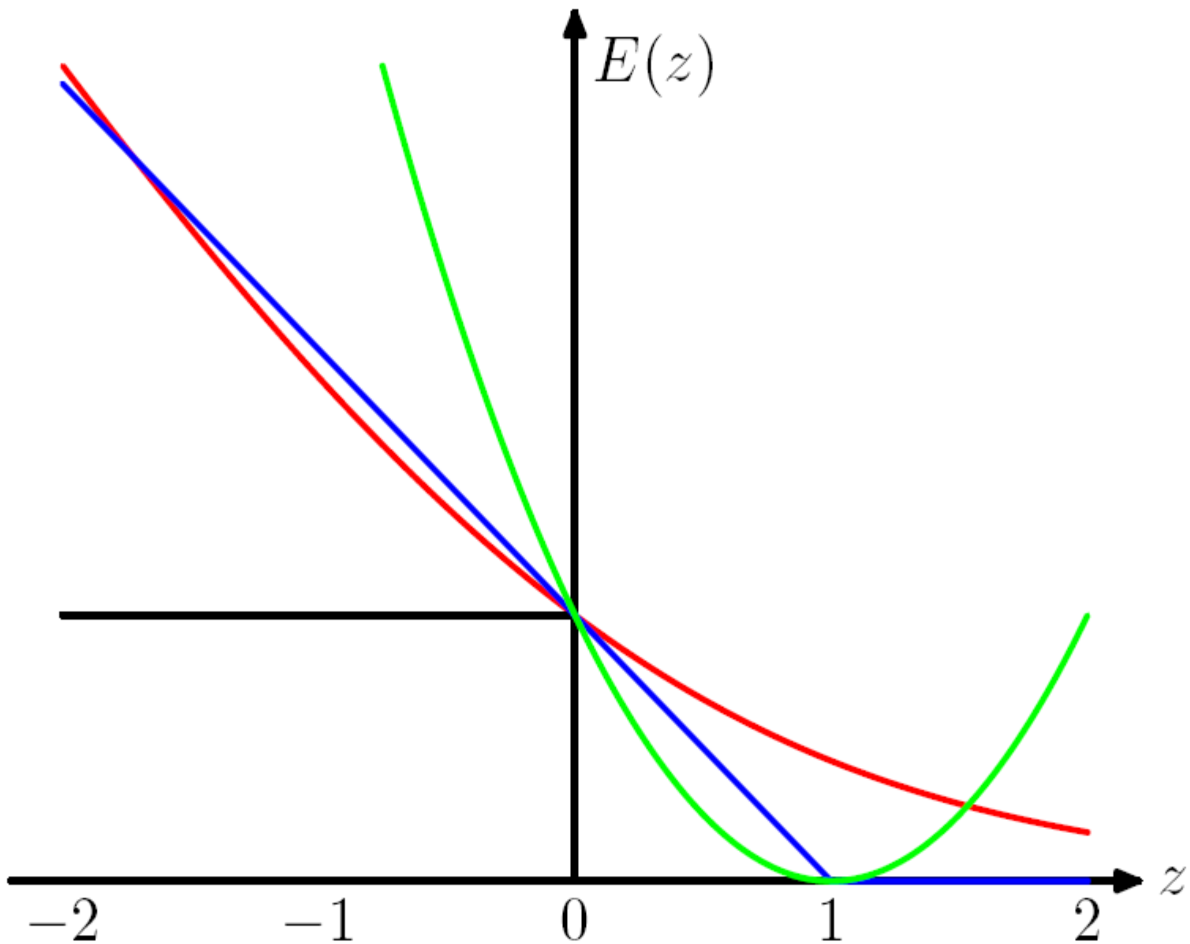
Perceptron: How to Update?

- How do we update w to improve our loss?
- Define an error function based on 0/1 loss

$$L_w(y) = \sum_{i}^{N} \max\left(0, -y_i w \cdot x_i\right)$$

- What is the difference?

Rethinking Loss Functions



The Perceptron Connection

- SVM minimizes the Perceptron but goes further
- Perceptron gives local updates, SVM gives global updates
- SVM is more aggressive: max-margin principle
- Could we apply max-margin to online learning?
  - Yes! Perceptron with margin
  - Other methods as well

Support Vector Machines

- **Fitting a function to data**

- Fitting: Batch optimization method: QP solver
- Function: select hyperplane that ensures a fixed margin, L2 regularization
  - Loss: hinge loss
- Data: Train in batch mode

Another Formulation

Dual Formulation

- The primal and dual formulations are complimentary
  - Solving one will give the solution for the other

- Primal problem: objective function is a combination of the m variables
  - Minimize the objective function
  - Solution is a vector of m values that minimize function

- Dual problem: objective function is a combination of n variables
  - Maximize the objective function
  - Solution is a vector of n values called the dual variables

SVM Solution

- Select αs that maximize

$$\sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} y_i y_j \alpha_i \alpha_j \left(x_i^T x_j \right)$$

- such that $\alpha_i > 0$ and $\sum_{i=1}^{N} (\alpha_i y_i) = 0$
- Predictions for new examples

$$x^T w = x^T \sum_{i=1}^{N} [\alpha_i y_i x_i] = \sum_{i=1}^{N} \alpha_i y_i (x^T x_i)$$

New Approach

- **Fitting a function to data**

- Fitting: Maximize objective in the dual using a QP solver
- Function: max margin linear classifier

$$\hat{y} = sign(x^T \cdot w) = sign\left(\sum_{i=1}^{N} \alpha_i y_i \left(x_i^T x_j\right)\right)$$

- Data: Train in batch mode

Dual vs. Primal Formulation

- In the primal we have M variables to solve
  - Solve for the vector w (length of features)
- In the dual we have N variables to solve
  - Solve for the vector α (length of examples)
- When to use the primal?
  - Lots of examples without many features
- When to use the dual?
  - Lots of features without many examples
  - Some other reasons (we'll talk about later)

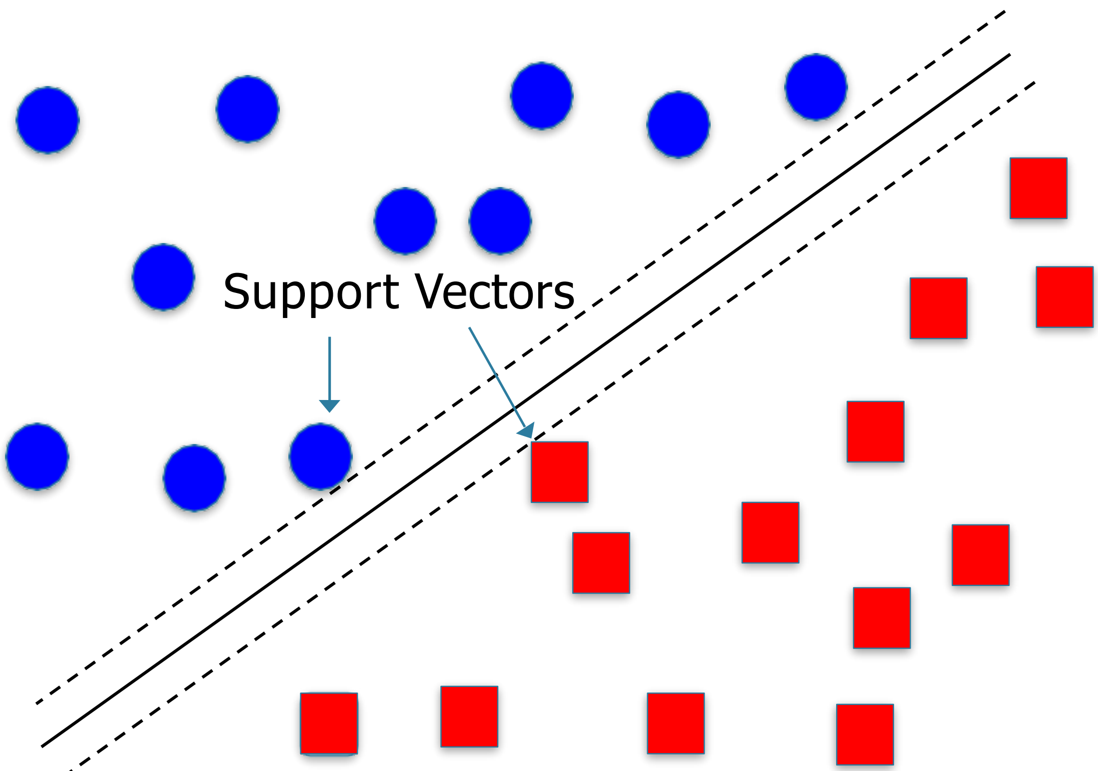
Support Vectors

- Why is it called support vector machine?
- Only some of the αs will be non-zero
  - All misclassified examples will be support vectors

$$\sum_{i=1}^{N} \alpha_i y_i \left(x_i^T x_i\right)$$

  - Only these vector support the hyperplane
  - These are the vectors closest to the hyperplane
- These are called “support vectors”

Support Vectors



By the Way

- We represented w in terms of the input X
- w is a linear combination of the inputs
    - Before: prediction was linear combination of w and x

$$w = \sum_{i=1}^{N} [\alpha_i y_i x_i]$$

- The same is true of Perceptron
- If we store the support examples

Dual Perceptron

Non-Separable Data

- But not all data is linearly separable
  - Previous solution: add a unique feature to every example to make it separable
- What will SVMs do?
  - The regularization forces the weights to be small
  - But it must still find a max margin solution
  - Result: even with significant regularization, still leads to over-fitting

Slack Variables

$$\min_{w} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N} \xi_i$$

$$\text{such that } (wx_i)y_i + \xi_i \geq 1, \forall i$$

$$\xi_i \geq 0, \forall i$$

- We can always satisfy the margin using ξ
  - We want these ξs to be small
  - Trade off parameter C (similar to λ before)
- ξs are called slack variables
  - The cut the margin some “slack”

Non-Separable Solution

- Similar form to the separable solution
- Extra term added to objective

Bias vs. Variance

- Smaller C means more slack (larger ξ)
  - More training examples are wrong
  - More bias (less variance) in the output
- Larger C means less slack (smaller ξ)
  - Better fit to the data
  - Less bias (more variance) in the output
- For non-separable data we can't learn a perfect separator so we don't want to try too hard
  - Finding the right balance is a tradeoff

Lingering Questions

- What would we do if we saw all of the data (batch)?
  - We'd pick the best separating hyperplane!
- Which separating hyperplane is the best?
  - The maximum margin separator
  - Use a quadratic regularizer on the weights
- What can we do for non-linear data?
  - It's not separable, use slack variables
  - Can we do better?