Introduction to Machine Learning

Maximum Margin Methods

Varun Chandola

Computer Science & Engineering
State University of New York at Buffalo
Buffalo, NY, USA
chandola@buffalo.edu

University at Buffalo
Department of Computer Science and Engineering
School of Engeering and Applied Sciences

Outline

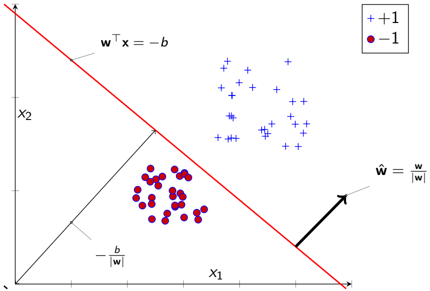
Training vs. Generalization Error

- Difference between training error and generalization error
- We can train a model to minimize the training error
- What we really want is a model that can minimize the generalization error
- But we do not have the *unseen* data to compute the generalization error
- What do we do?
  1. Focus on the training error and hope that generalization error is automatically minimized
  2. Incorporate some way to hedge (insure) against possible unseen issues

Maximum Margin Classifiers

$$y = \mathbf{w}^\top \mathbf{x} + b$$

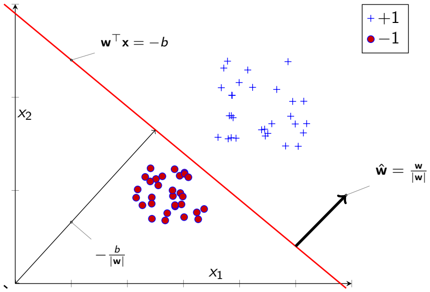
- Remember the Perceptron!
- If data is linearly separable
  - Perceptron training guarantees learning the decision boundary
- There can be other boundaries
  - Depends on initial value for $\mathbf{w}$



Maximum Margin Classifiers

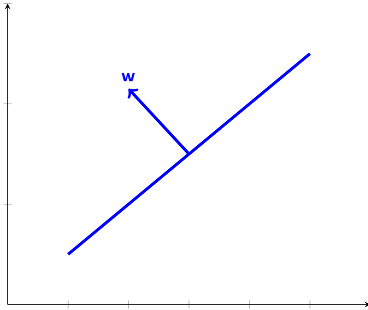
$$y = \mathbf{w}^\top \mathbf{x} + b$$

- Remember the Perceptron!
- If data is linearly separable
  - Perceptron training guarantees learning the decision boundary
- There can be other boundaries
  - Depends on initial value for $\mathbf{w}$
- **But what is the best boundary?**



Linear Hyperplane

- Separates a D-dimensional space into two half-spaces
- Defined by $\mathbf{w} \in \Re^D$
  - *Orthogonal* to the hyperplane
  - This $\mathbf{w}$ goes through the origin
  - How do you check if a point lies "above" or "below" $\mathbf{w}$?
  - What happens for points **on** $\mathbf{w}$?



Make hyperplane not go through origin

- Add a bias b
  - $b > 0$ - move along $\mathbf{w}$
  - $b < 0$ - move opposite to $\mathbf{w}$
- How to check if point lies above or below $\mathbf{w}$?
  - If $\mathbf{w}^\top\mathbf{x} + b > 0$ then $\mathbf{x}$ is *above*
  - Else, *below*

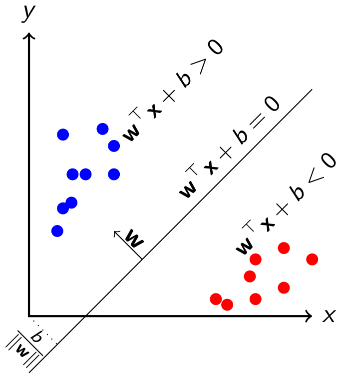
Line as a Decision Surface

- Decision boundary represented by the hyperplane $\mathbf{w}$
- For binary classification, $\mathbf{w}$ points **towards** the positive class

Decision Rule

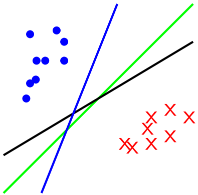
$$y = sign(\mathbf{w}^\top \mathbf{x} + b)$$

- $\mathbf{w}^\top \mathbf{x} + b > 0 \Rightarrow y = +1$
- $\mathbf{w}^\top \mathbf{x} + b < 0 \Rightarrow y = -1$



What is Best Hyperplane Separator

- **Perceptron** can find a hyperplane that separates the data
  - . . . if the data is linearly separable
- But there can be many choices!
- Find the one with best separability (largest margin)
- Gives better generalization performance



1. Intuitive reason
2. Theoretical foundations

What is a Margin?

- **Margin** is the distance between an example and the decision line
- Denoted by γ
- For a positive point:

$$\gamma = \frac{\mathbf{w}^\top \mathbf{x} + b}{\|\mathbf{w}\|}$$

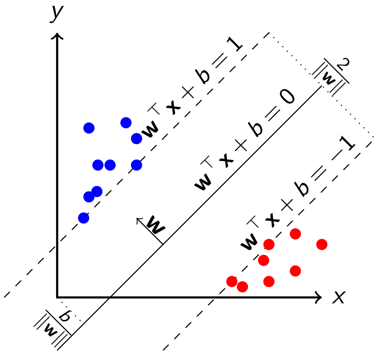
- For a negative point:

$$\gamma = -\frac{\mathbf{w}^\top \mathbf{x} + b}{\|\mathbf{w}\|}$$

Functional Interpretation

- Margin **positive** if prediction is **correct**; **negative** if prediction is **incorrect**

Maximum Margin Principle



Support Vector Machines

- A hyperplane based classifier defined by **w** and *b*
- Like perceptron
- Find hyperplane with *maximum separation margin* on the training data
- Assume that data is linearly separable (will relax this later)
  - Zero training error (loss)

SVM Prediction Rule

$$y = sign(\mathbf{w}^\top \mathbf{x} + b)$$

SVM Learning

- **Input**: Training data $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_N, y_N)\}$
- **Objective**: Learn **w** and *b* that maximizes the margin

SVM Learning

- SVM learning task as an optimization problem
- Find $\mathbf{w}$ and b that gives zero training error
- Maximizes the margin $(= \frac{2}{\|\mathbf{w}\|})$
- Same as minimizing $\|\mathbf{w}\|$

Optimization Formulation

$$\begin{aligned} \underset{\mathbf{w},b}{\text{minimize}} \quad & \frac{\|\mathbf{w}\|^2}{2} \\ \text{subject to} \quad & y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1, \; i = 1, \ldots, N. \end{aligned}$$

- **Optimization** with N linear inequality constraints

Solving the Optimization Problem

Optimization Formulation

$$\begin{aligned} \underset{\mathbf{w},b}{\text{minimize}} \quad & \frac{\|\mathbf{w}\|^2}{2} \\ \text{subject to} \quad & y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1, \; i = 1, \ldots, N. \end{aligned}$$

or

$$\begin{aligned} \underset{\mathbf{w},b}{\text{minimize}} \quad & \frac{\|\mathbf{w}\|^2}{2} \\ \text{subject to} \quad & 1 - [y_i(\mathbf{w}^\top \mathbf{x}_i + b)] \leq 0, \; i = 1, \ldots, N. \end{aligned}$$

- There is an quadratic objective function to minimize with N inequality constraints
- "Off-the-shelf" packages - quadprog (MATLAB), CVXOPT
- Is that the best way?

Basic Optimization

$$\boxed{\underset{x,y}{\text{minimize}} \quad f(x,y) = \quad x^2 + 2y^2 - 2}$$

Basic Optimization

$$\boxed{\underset{x,y}{\text{minimize}} \quad f(x,y) = \quad x^2 + 2y^2 - 2}$$

$$\boxed{\begin{aligned} \underset{x,y}{\text{minimize}} \quad & f(x,y) = \quad x^2 + 2y^2 - 2 \\ \text{subject to} \quad & h(x,y) = \quad x + y - 1 = 0. \end{aligned}}$$

Lagrange Multipliers - A Primer

- Tool for solving constrained optimization problems of differentiable functions

$$\boxed{\begin{array}{lll} \underset{x,y}{\text{minimize}} & f(x,y) = & x^2 + 2y^2 - 2 \\ \text{subject to} & h(x,y): & x + y - 1 = 0. \end{array}}$$

- A Lagrangian multiplier (β) lets you combine the two equations into one

Lagrange Multipliers - A Primer

- Tool for solving constrained optimization problems of differentiable functions

$$\begin{aligned} \underset{x,y}{\text{minimize}} \quad & f(x,y) = & x^2 + 2y^2 - 2 \\ \text{subject to} \quad & h(x,y): & x + y - 1 = 0. \end{aligned}$$

- A Lagrangian multiplier (β) lets you combine the two equations into one

$$\underset{x,y,\beta}{\text{minimize}} \quad L(x,y,\beta) = \quad f(x,y) + \beta h(x,y)$$

Multiple Constraints

$$
\begin{aligned}
\underset{x,y,z}{\text{minimize}} \quad & f(x,y,z) = && x^2 + 4y^2 + 2z^2 + 6y + z \\
\text{subject to} \quad & h_1(x,y,z): && x + z^2 - 1 = 0 \\
& h_2(x,y,z): && x^2 + y^2 - 1 = 0.
\end{aligned}
$$

Multiple Constraints

$$\begin{array}{lll}\underset{x,y,z}{\text{minimize}} & f(x,y,z) = & x^2 + 4y^2 + 2z^2 + 6y + z \\ \text{subject to} & h_1(x,y,z): & x + z^2 - 1 = 0 \\ & h_2(x,y,z): & x^2 + y^2 - 1 = 0.\end{array}$$

$$L(x,y,z,\boldsymbol{\beta}) = f(x,y,z) + \sum_i \beta_i h_i(x,y,z)$$

Handling Inequality Constraints

$$\boxed{\begin{aligned} \underset{x,y}{\text{minimize}} \quad & f(x,y) = & x^3 + y^2 \\ \text{subject to} \quad & g(x): & x^2 - 1 \leq 0. \end{aligned}}$$

Handling Inequality Constraints

$$\begin{aligned} \underset{x,y}{\text{minimize}} \quad & f(x,y) = & x^3 + y^2 \\ \text{subject to} \quad & g(x): & x^2 - 1 \leq 0. \end{aligned}$$

- Inequality constraints are **transferred** as constraints on the generalized Lagrangian, using the multiplier, α
- Technically, α is a `Kahrun-Kuhn-Tucker` (KKT) multiplier
- Lagrangian formulation is a special case of KKT formulation with no inequality constraints
- We will use the term *generalized Lagrangian* instead

Handling Both Types of Constraints

$$\begin{array}{lll} \underset{\mathbf{w}}{\text{minimize}} & f(\mathbf{w}) & \\ \text{subject to} & g_i(\mathbf{w}) \leq 0 & i = 1, \ldots, k \\ \text{and} & h_i(\mathbf{w}) = 0 & i = 1, \ldots, l. \end{array}$$

Generalized Lagrangian

$$L(\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = f(\mathbf{w}) + \sum_{i=1}^{k} \alpha_i g_i(\mathbf{w}) + \sum_{i=1}^{l} \beta_i h_i(\mathbf{w})$$

subject to, $\alpha_i \geq 0, \forall i$

Lagrange Multipliers for SVM

Optimization Formulation

$$
\begin{aligned}
& \underset{\mathbf{w},b}{\text{minimize}} && \frac{\|\mathbf{w}\|^2}{2} \\
& \text{subject to} && 1 - [y_i(\mathbf{w}^\top \mathbf{x}_i + b)] \leq 0, \; i = 1, \ldots, N.
\end{aligned}
$$

Lagrange Multipliers for SVM

Optimization Formulation

$$\begin{aligned} \underset{\mathbf{w},b}{\text{minimize}} \quad & \frac{\|\mathbf{w}\|^2}{2} \\ \text{subject to} \quad & 1 - [y_i(\mathbf{w}^\top \mathbf{x}_i + b)] \leq 0, \;\; i = 1, \ldots, N. \end{aligned}$$

A Toy Example

- $\mathbf{x} \in \Re^2$
- Two training points:

$$\mathbf{x}_1, y_1 = (1, 1), -1$$

$$\mathbf{x}_2, y_2 = (2, 2), +1$$

- Find the best hyperplane $\mathbf{w} = (w_1, w_2)$

Optimization problem for a toy example

$$
\begin{aligned}
\underset{\mathbf{w}}{\text{minimize}} \quad & f(\mathbf{w}) = && \frac{1}{2}\|\mathbf{w}\|^2 \\
\text{subject to} \quad & g_1(\mathbf{w}, b) = && 1 - y_1(\mathbf{w}^\top \mathbf{x}_1 + b) \leq 0 \\
& g_2(\mathbf{w}, b) = && 1 - y_2(\mathbf{w}^\top \mathbf{x}_2 + b) \leq 0.
\end{aligned}
$$

Optimization problem for a toy example

$$
\begin{array}{lrr}
\underset{\mathbf{w}}{\text{minimize}} & f(\mathbf{w}) = & \frac{1}{2}\|\mathbf{w}\|^2 \\
\text{subject to} & g_1(\mathbf{w}, b) = & 1 - y_1(\mathbf{w}^\top \mathbf{x}_1 + b) \leq 0 \\
 & g_2(\mathbf{w}, b) = & 1 - y_2(\mathbf{w}^\top \mathbf{x}_2 + b) \leq 0.
\end{array}
$$

- Substituting actual values for $\mathbf{x}_1, y_1$ and $\mathbf{x}_2, y_2$.

$$
\begin{array}{lrr}
\underset{\mathbf{w}}{\text{minimize}} & f(\mathbf{w}) = & \frac{1}{2}\|\mathbf{w}\|^2 \\
\text{subject to} & g_1(\mathbf{w}, b) = & 1 + (\mathbf{w}^\top \mathbf{x}_1 + b) \leq 0 \\
 & g_2(\mathbf{w}, b) = & 1 - (\mathbf{w}^\top \mathbf{x}_2 + b) \leq 0.
\end{array}
$$

Primal and Dual Formulations

Generalized Lagrangian

$$L(\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = f(\mathbf{w}) + \sum_{i=1}^{k} \alpha_i g_i(\mathbf{w}) + \sum_{i=1}^{l} \beta_i h_i(\mathbf{w})$$

subject to, $\alpha_i \geq 0, \forall i$

Primal Optimization

- Let θ_P be defined as:

$$\theta_P(\mathbf{w}) = \max_{\boldsymbol{\alpha},\boldsymbol{\beta}:\alpha_i \geq 0} L(\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta})$$

- One can prove that the optimal value for the original constrained problem is same as:

$$p^* = \min_{\mathbf{w}} \theta_P(\mathbf{w}) = \min_{\mathbf{w}} \max_{\boldsymbol{\alpha},\boldsymbol{\beta}:\alpha_i \geq 0} L(\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta})$$

Primal and Dual Formulations (II)

Dual Optimization

- Consider θ_D, defined as:

$$\theta_D(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \min_{\mathbf{w}} L(\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta})$$

- The **dual** optimization problem can be posed as:

$$d^* = \max_{\boldsymbol{\alpha}, \boldsymbol{\beta} : \alpha_i \geq 0} \theta_D(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \max_{\boldsymbol{\alpha}, \boldsymbol{\beta} : \alpha_i \geq 0} \min_{\mathbf{w}} L(\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta})$$

$d^* == p^*$?

- Note that $d^* \leq p^*$
- "Max min" of a function is always less than or equal to "Min max"
- When will they be equal?
  - $f(\mathbf{w})$ is convex
  - Constraints are affine
  - $\exists \mathbf{w}, s.t., g_i(\mathbf{w}) < 0, \forall i$
- For SVM optimization the equality holds

Kahrun-Kuhn-Tucker (KKT) Conditions

- First derivative tests to check if a solution for a non-linear optimization problem is *optimal*
- For $d^* = p^* = L(\mathbf{w}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$:

$$
\begin{aligned}
\frac{\partial}{\partial \mathbf{w}} L(\mathbf{w}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*) & = 0 \\
\frac{\partial}{\partial \beta_i} L(\mathbf{w}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*) & = 0, \quad i = 1, \ldots, l \\
\alpha_i^* g_i(\mathbf{w}^*) & = 0, \quad i = 1, \ldots, k \\
g_i(\mathbf{w}^*) & \leq 0, \quad i = 1, \ldots, k \\
\alpha_i^* & \geq 0, \quad i = 1, \ldots, k
\end{aligned}
$$

Back to SVM Optimization

Optimization Formulation

$$\begin{aligned} \underset{\mathbf{w},b}{\text{minimize}} \quad & \frac{\|\mathbf{w}\|^2}{2} \\ \text{subject to} \quad & y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1,\ i = 1,\ldots,N. \end{aligned}$$

- Introducing Lagrange Multipliers,$\alpha_i,\ i = 1,\ldots,N$

Rewriting as a (primal) Lagrangian

$$\begin{aligned} \underset{\mathbf{w},b,\alpha}{\text{minimize}} \quad & L_P(\mathbf{w},b,\alpha) = \frac{\|\mathbf{w}\|^2}{2} + \sum_{i=1}^{N} \alpha_i\{1 - y_i(\mathbf{w}^\top \mathbf{x}_i + b)\} \\ \text{subject to} \quad & \alpha_i \geq 0\ i = 1,\ldots,N. \end{aligned}$$

Solving the Lagrangian

- Set gradient of L_P to 0

$$\frac{\partial L_P}{\partial \mathbf{w}} = 0 \Rightarrow \boxed{\mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i}$$

$$\frac{\partial L_P}{\partial b} = 0 \Rightarrow \boxed{\sum_{i=1}^{N} \alpha_i y_i = 0}$$

- Substituting in L_P to get the dual L_D

Solving the Lagrangian

- Set gradient of L_P to 0

$$\frac{\partial L_P}{\partial \mathbf{w}} = 0 \Rightarrow \boxed{\mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i}$$

$$\frac{\partial L_P}{\partial b} = 0 \Rightarrow \boxed{\sum_{i=1}^{N} \alpha_i y_i = 0}$$

- Substituting in L_P to get the dual L_D

Dual Lagrangian Formulation

$$\underset{b,\alpha}{\text{maximize}} \quad L_D(\mathbf{w}, b, \alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{m,n=1}^{N} \alpha_m \alpha_i y_m y_i (\mathbf{x}_m^\top \mathbf{x}_i)$$

$$\text{subject to} \quad \sum_{i=1}^{N} \alpha_i y_i = 0, \alpha_i \geq 0 \; i = 1, \ldots, N.$$

Solving the Dual

- Dual Lagrangian is a *quadratic programming problem* in α_i's
  - Use "off-the-shelf" solvers
- Having found α_i's

$$\mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i$$

- What will be the bias term b?

Investigating Kahrun Kuhn Tucker Conditions

- For the primal and dual formulations
- We can optimize the dual formulation (as shown earlier)
- Solution should satisfy the **Karush-Kuhn-Tucker** (KKT) Conditions

The Kahrun-Kuhn-Tucker Conditions

$$
\begin{aligned}
\frac{\partial}{\partial \mathbf{w}} L_P(\mathbf{w}, b, \alpha) &= \mathbf{w} - \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i = 0 && (1)\\
\frac{\partial}{\partial b} L_P(\mathbf{w}, b, \alpha) &= -\sum_{i=1}^{N} \alpha_i y_i = 0 && (2)\\
1 - y_i\{\mathbf{w}^\top \mathbf{x}_i + b\} &\leq 0 && (3)\\
\alpha_i &\geq 0 && (4)\\
\alpha_i(1 - y_i\{\mathbf{w}^\top \mathbf{x}_i + b\}) &= 0 && (5)
\end{aligned}
$$

Estimating Bias b

- Use KKT condition #5
- For $\alpha_i > 0$

$$(y_i\{\mathbf{w}^\top \mathbf{x}_i + b\} - 1) = 0$$

- Which means that:

$$b = -\frac{\underset{n:y_i=-1}{max}\,\mathbf{w}^\top \mathbf{x}_i + \underset{n:y_i=1}{min}\,\mathbf{w}^\top \mathbf{x}_i}{2}$$

Key Observation from Dual Formulation

Most α_i's are 0

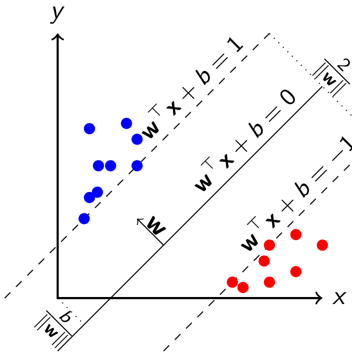
- KKT condition #5:

$$\alpha_i(1 - y_i\{\mathbf{w}^\top \mathbf{x}_i + b\}) = 0$$

- If $\mathbf{x}_i$ **not** on margin

$$y_i\{\mathbf{w}^\top \mathbf{x}_i + b\} > 1$$
$$\Rightarrow \quad \alpha_i = 0$$

- $\alpha_i \neq 0$ only for $\mathbf{x}_i$ on margin
- These are the **support vectors**
- Only need these for prediction



What if data is not linearly separable?

- Cannot go for zero training error
- Still learn a maximum margin hyperplane

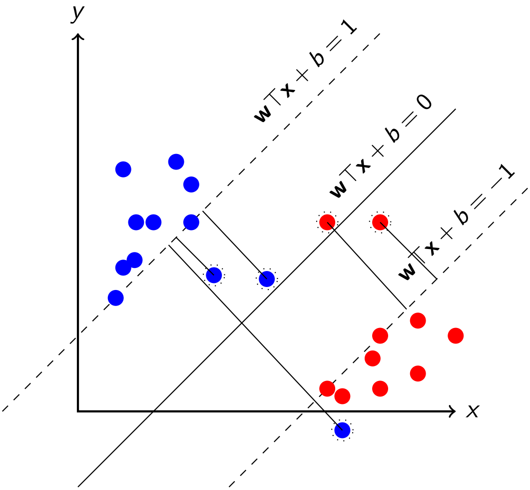
What if data is not linearly separable?

- Cannot go for zero training error
- Still learn a maximum margin hyperplane
  1. Allow some examples to be misclassified
  2. Allow some examples to fall **inside** the margin

What if data is not linearly separable?

- Cannot go for zero training error
- Still learn a maximum margin hyperplane
  1. Allow some examples to be misclassified
  2. Allow some examples to fall **inside** the margin
- How do you set up the optimization for SVM training

Cutting Some Slack



Introducing Slack Variables

- **Separable Case**: To ensure zero training loss, constraint was

$$y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 \quad \forall i = 1 \ldots N$$

Introducing Slack Variables

- **Separable Case**: To ensure zero training loss, constraint was

$$y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 \quad \forall i = 1 \ldots N$$

- **Non-separable Case**: Relax the constraint

$$y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 - \xi_i \quad \forall i = 1 \ldots N$$

- ξ_i is called **slack variable** ($\xi_i \geq 0$)
- For misclassification, $\xi_i > 1$

Relaxing the Constraint

- It is OK to have some misclassified training examples
  - Some ξ_i's will be non-zero

Relaxing the Constraint

- It is OK to have some misclassified training examples
  - Some ξ_i's will be non-zero
- Minimize the number of such examples
  - Minimize $\boxed{\sum_{i=1}^{N} \xi_i}$
- Optimization Problem for Non-Separable Case

$$\boxed{\begin{aligned} &\underset{\mathbf{w},b}{\text{minimize}} && f(\mathbf{w}, b) = \|\mathbf{w}\|^2 + C\sum_{i=1}^{N} \xi_i \\ &\text{subject to} && y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 - \xi_i, \xi_i \geq 0 \; i = 1, \ldots, N. \end{aligned}}$$

Estimating Weights

- Similar optimization procedure as for the separable case (QP for the dual)
- Weights have the same expression

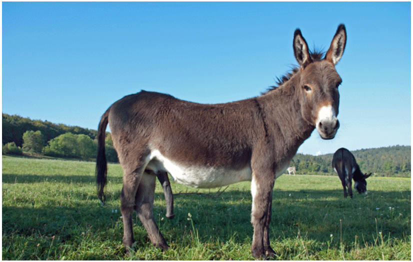
$$\mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i$$

- Support vectors are slightly different
  1. Points on the margin ($\xi_i = 0$)
  2. Inside the margin but on the correct side ($0 < \xi_i < 1$)
  3. On the wrong side of the hyperplane ($\xi_i \geq 1$)

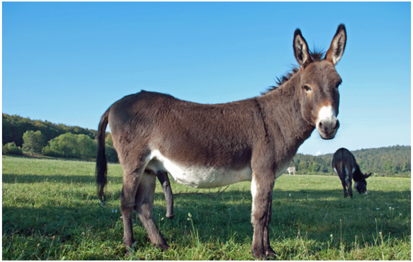
What is the role of C?

- C dictates if we focus more on maximizing the margin or reducing the training error.
- Controls the *bias-variance* tradeoff

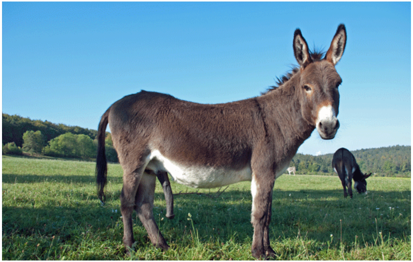
The Bias-Variance Tradeoff



The Bias-Variance Tradeoff



The Bias-Variance Tradeoff



- C allows the model to be a mule or a sheep or something in between
- Question: What do you want the model to be?

Concluding Remarks on SVM

- Training time for SVM training is $O(N^3)$
- Many *faster* but approximate approaches exist
  - Approximate QP solvers
  - Online training
- SVMs can be extended in different ways
  1. Non-linear boundaries (**kernel trick**)
  2. Multi-class classification
  3. Probabilistic output
  4. Regression (Support Vector Regression)

References