

Introduction to Machine Learning

Maximum Margin Methods

Varun Chandola

March 2, 2020

Outline

Contents

1	Training vs. Generalization Error	1
2	Maximum Margin Classifiers	2
2.1	Linear Classification via Hyperplanes	3
2.2	Concept of Margin	5
3	Support Vector Machines	6
3.1	SVM Learning	7
3.2	Solving SVM Optimization Problem	7
4	Constrained Optimization and Lagrange Multipliers	8
4.1	Toy SVM Example	11
4.2	Karun-Kuhn-Tucker Conditions	15
4.3	Support Vectors	16
4.4	Optimization Constraints	17
5	The Bias-Variance Tradeoff	19

1 Training vs. Generalization Error

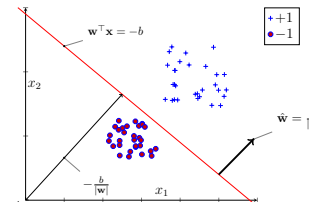
Training vs. Generalization Error

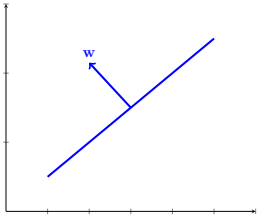
- Difference between training error and generalization error
- We can train a model to minimize the training error
- What we really want is a model that can minimize the generalization error
- But we do not have the *unseen* data to compute the generalization error
- What do we do?
 1. Focus on the training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

2 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?**



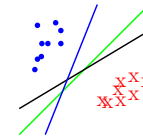
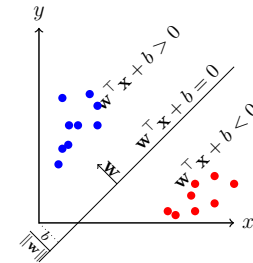


2.1 Linear Classification via Hyperplanes

- Separates a D -dimensional space into two half-spaces
- Defined by $\mathbf{w} \in \mathbb{R}^D$
 - *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} ?

For a hyperplane that passes through the origin, a point \mathbf{x} will lie above the hyperplane if $\mathbf{w}^\top \mathbf{x} > 0$ and will lie below the plane if $\mathbf{w}^\top \mathbf{x} < 0$, otherwise. This can be further understood by understanding that $b\mathbf{w}^\top \mathbf{x}$ is essentially equal to $|\mathbf{w}||\mathbf{x}|\cos\theta$, where θ is the angle between \mathbf{w} and \mathbf{x} .

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



Decision Rule

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

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

2.2 Concept of 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:

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

To understand the margin from a geometric perspective, consider the projection of the vector connecting the origin to a point \mathbf{x} on the decision line. Let the point be denoted as \mathbf{x}' . Obviously the vector \mathbf{r} connecting \mathbf{x}' and \mathbf{x} is given by:

$$\mathbf{r} = \gamma \hat{\mathbf{w}} = \gamma \frac{\mathbf{w}}{\|\mathbf{w}\|}$$

if \mathbf{x} lies on the positive side of \mathbf{w} . But the same vector can be computed as:

$$\mathbf{r} = \mathbf{x} - \mathbf{x}'$$

Equating above two gives us \mathbf{x}' as:

$$\mathbf{x}' = \mathbf{x} - \gamma \frac{\mathbf{w}}{\|\mathbf{w}\|}$$

Noting that, since \mathbf{x}' lies on the hyperplane and hence:

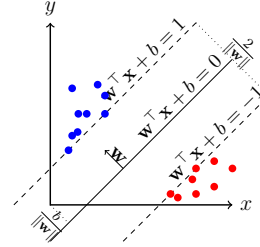
$$\mathbf{w}^\top \mathbf{x}' + b = 0$$

Substituting \mathbf{x}' from above:

$$\mathbf{w}^\top \mathbf{x} - \gamma \frac{\mathbf{w}^\top \mathbf{w}}{\|\mathbf{w}\|} + b = 0$$

Noting that $\frac{\mathbf{w}^\top \mathbf{w}}{\|\mathbf{w}\|} = \|\mathbf{w}\|$, we get γ as:

$$\gamma = \frac{\mathbf{w}^\top \mathbf{x} + b}{\|\mathbf{w}\|} \quad (1)$$



Similar analysis can be done for points on the negative side of \mathbf{x} . In general, one can write the expression for the margin as:

$$\gamma = y \frac{\mathbf{w}^\top \mathbf{x} + b}{\|\mathbf{w}\|} \quad (2)$$

where $y \in \{-1, +1\}$.

Functional Interpretation

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

From the figure one can note that the size of the margin is $\frac{2}{\|\mathbf{w}\|}$. We can show this as follows. Since the data is separable, we can get two parallel lines represented by $\mathbf{w}^\top \mathbf{x} + b = +1$ and $\mathbf{w}^\top \mathbf{x} + b = -1$. Using result from (1) and (2), the distance between the two lines is given by $2\gamma = \frac{2}{\|\mathbf{w}\|}$.

3 Support Vector Machines

- A hyperplane based classifier defined by \mathbf{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

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

SVM Learning

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

3.1 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

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

- **Optimization** with N linear inequality constraints

3.2 Solving SVM Optimization Problem

Optimization Formulation

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

or

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

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

4 Constrained Optimization and Lagrange Multipliers

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

- Tool for solving constrained optimization problems of differentiable functions

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

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

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

Solution 1. Writing the objective as Lagrangian.

$$L(x, y, \beta) = x^2 + 2y^2 - 2 + \beta(x + y - 1)$$

Setting the gradient to 0 with respect to x, y and β will give us the optimal values.

$$\begin{aligned} \frac{\partial L}{\partial x} &= 2x + \beta = 0 \\ \frac{\partial L}{\partial y} &= 4y + \beta = 0 \\ \frac{\partial L}{\partial \beta} &= x + y - 1 = 0 \end{aligned}$$

Multiple Constraints

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

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

Handling Inequality Constraints

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

- Inequality constraints are **transferred** as constraints on the generalized Lagrangian, using the multiplier, α
- Technically, α is a **Karun-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

The Lagrangian in the above example becomes:

$$\begin{aligned} L(x,y,\alpha) &= f(x,y) + \alpha g(x,y) \\ &= x^3 + y^2 + \alpha(x^2 - 1) \end{aligned}$$

Solving for the gradient of the Lagrangian gives us:

$$\begin{aligned} \frac{\partial}{\partial x} L(x,y,\alpha) &= 3x^2 + 2\alpha x = 0 \\ \frac{\partial}{\partial y} L(x,y,\alpha) &= 2y = 0 \\ \frac{\partial}{\partial \alpha_1} L(x,y,\alpha) &= x^2 - 1 = 0 \end{aligned}$$

Furthermore we require that:

$$\alpha \geq 0$$

From above equations we get $y = 0$, $x = \pm 1$ and $\alpha = \pm \frac{3}{2}$. But since $\alpha \geq 0$, hence $\alpha = \frac{3}{2}$. This gives $x = 1$, $y = 0$, and $f = 1$.

Handling Both Types of Constraints

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

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

Optimization Formulation

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

A Toy Example

- $\mathbf{x} \in \mathbb{R}^2$

- Two training points:

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

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

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

4.1 Toy SVM Example

Optimization problem for a toy example

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

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

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

The above problem can be also written as:

$$\begin{aligned} \underset{w_1, w_2, b}{\text{minimize}} \quad & f(w_1, w_2) = \frac{1}{2} (w_1^2 + w_2^2) \\ \text{subject to} \quad & g_1(w_1, w_2, b) = 1 + (w_1 + w_2 + b) \leq 0 \\ & g_2(w_1, w_2, b) = 1 - (2w_1 + 2w_2 + b) \leq 0. \end{aligned}$$

To solve the toy optimization problem, we rewrite it in the Lagrangian form:

$$L(w_1, w_2, b, \alpha) = \frac{1}{2} (w_1^2 + w_2^2) + \alpha_1 (1 + w_1 + w_2 + b) + \alpha_2 (1 - (2w_1 + 2w_2 + b))$$

Setting $\nabla L = 0$, we get:

$$\begin{aligned} \frac{\partial}{\partial w_1} L(w_1, w_2, b, \alpha) &= w_1 + \alpha_1 - 2\alpha_2 = 0 \\ \frac{\partial}{\partial w_2} L(w_1, w_2, b, \alpha) &= w_2 + \alpha_1 - 2\alpha_2 = 0 \\ \frac{\partial}{\partial b} L(w_1, w_2, b, \alpha) &= \alpha_1 - \alpha_2 = 0 \\ \frac{\partial}{\partial \alpha_1} L(w_1, w_2, b, \alpha) &= w_1 + w_2 + b + 1 = 0 \\ \frac{\partial}{\partial \alpha_2} L(w_1, w_2, b, \alpha) &= 2w_1 + 2w_2 + b - 1 = 0 \end{aligned}$$

Solving the above equations, we get, $w_1 = w_2 = 1$ and $b = -3$.

Primal and Dual Formulations

Generalized Lagrangian

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

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

Primal Optimization

- Let θ_P be defined as:

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

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

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

Consider

$$\begin{aligned} \theta_P(\mathbf{w}) &= \max_{\alpha, \beta: \alpha_i \geq 0} L(\mathbf{w}, \alpha, \beta) \\ &= \max_{\alpha, \beta: \alpha_i \geq 0} f(\mathbf{w}) + \sum_{i=1}^k \alpha_i g_i(\mathbf{w}) + \sum_{i=1}^l \beta_i h_i(\mathbf{w}) \end{aligned}$$

It is easy to show that if any constraints are not satisfied, i.e., if either $g_i(\mathbf{w}) > 0$ or $h_i(\mathbf{w}) \neq 0$, then $\theta_P(\mathbf{w}) = \infty$. Which means that:

$$\theta_P(\mathbf{w}) = \begin{cases} f(\mathbf{w}) & \text{if primal constraints are satisfied} \\ \infty & \text{otherwise,} \end{cases}$$

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

Karun-Kuhn-Tucker (KKT) Conditions

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

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

Back to SVM Optimization

Optimization Formulation

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

- Introducing [Lagrange Multipliers](#), $\alpha_i, i = 1, \dots, N$

Rewriting as a (primal) Lagrangian

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

Solving the Lagrangian

- Set gradient of L_P to 0

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

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

- Substituting in L_P to get the dual L_D

Dual Lagrangian Formulation

$$\begin{aligned} \underset{b, \alpha}{\text{maximize}} \quad & L_D(\mathbf{w}, b, \alpha) = \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{m, n=1}^N \alpha_m \alpha_n y_m y_n (\mathbf{x}_m^\top \mathbf{x}_n) \\ \text{subject to} \quad & \sum_{i=1}^N \alpha_i y_i = 0, \alpha_i \geq 0 \quad i = 1, \dots, N. \end{aligned}$$

- 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 Kahrn 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

4.2 Kahrn-Kuhn-Tucker Conditions

$$\frac{\partial}{\partial \mathbf{w}} L_P(\mathbf{w}, b, \alpha) = \mathbf{w} - \sum_{i=1}^N \alpha_i y_i \mathbf{x}_i = 0 \quad (1)$$

$$\frac{\partial}{\partial b} L_P(\mathbf{w}, b, \alpha) = - \sum_{i=1}^N \alpha_i y_i = 0 \quad (2)$$

$$1 - y_i \{\mathbf{w}^\top \mathbf{x}_i + b\} \leq 0 \quad (3)$$

$$\alpha_i \geq 0 \quad (4)$$

$$\alpha_i (1 - y_i \{\mathbf{w}^\top \mathbf{x}_i + b\}) = 0 \quad (5)$$

- Use KKT condition #5

- For $\alpha_i > 0$

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

- Which means that:

$$b = - \frac{\max_{n:y_i=-1} \mathbf{w}^\top \mathbf{x}_i + \min_{n:y_i=1} \mathbf{w}^\top \mathbf{x}_i}{2}$$

4.3 Support Vectors

Most α_i 's are 0

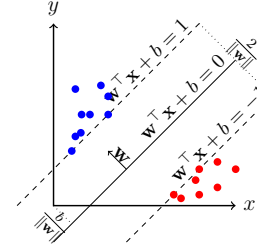
- KKT condition #5:

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

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

$$\begin{aligned} y_i \{\mathbf{w}^\top \mathbf{x}_i + b\} &> 1 \\ \Rightarrow \alpha_i &= 0 \end{aligned}$$

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



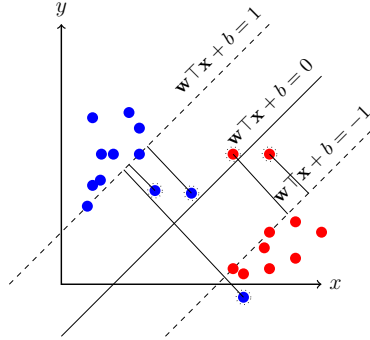
One can see from the prediction equation

that:

$$y^* = \text{sign} \left(\sum_{i=1}^N \alpha_i y_i (\mathbf{x}_i^\top \mathbf{x}^*) \right)$$

In the summation, the entries for \mathbf{x}_i that do not lie on the margin will have no contribution to the sum because α_i for those \mathbf{x}_i 's will be 0. Hence we only need to the non-zero input examples to get the prediction.

- 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

Introducing Slack Variables

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

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

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

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

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

4.4 Optimization Constraints

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

- Minimize the number of such examples

$$\text{– Minimize } \sum_{i=1}^N \xi_i$$

- Optimization Problem for Non-Separable Case

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

- 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 ($\xi_i \geq 1$)

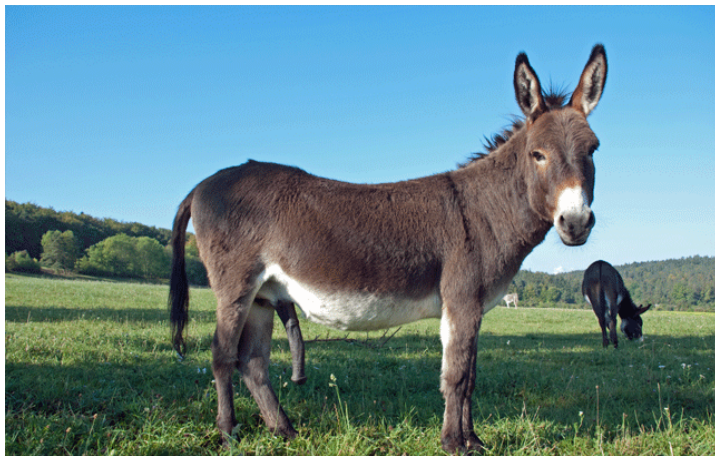
It is straightforward to see why the support vectors also includes points that are on the wrong side of the margin. The KKT condition #5, in this case will be:

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

For any point that is not on the margin, but is on the correct side of the margin, ξ_i will be 0, and $y_i\{\mathbf{w}^\top \mathbf{x}_i + b\} > 1$. Thus to satisfy the above condition, α_i will be 0. However, for the points on the margin, both ξ_i and $y_i\{\mathbf{w}^\top \mathbf{x}_i + b\}$ will be 0, thus, α_i will be non-zero. Finally, for the points that are on the wrong side of the margin, ξ_i will be equal to $1 - \{y_i\{\mathbf{w}^\top \mathbf{x}_i + b\}\}$, because that is how the slack is defined. Thus, α_i will have to be non-zero to satisfy the condition.

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

5 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?
- 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