STA 314: Statistical Methods for Machine Learning I

Lecture 8 - Support Vector Machine

Xin Bing

Department of Statistical Sciences
University of Toronto

Linear decision boundaries

In binary classification problems, we have seen examples of classifiers that use linear decision boundaries.

- LDA:

$$\delta_k(x) = x^\top \Sigma^{-1} \mu_k - \frac{1}{2} \mu_k^\top \Sigma^{-1} \mu_k + \log \pi_k, \quad \forall k \in \{0, 1\}.$$

  Hence, $\delta_1(x) \geq \delta_0(x)$ is if and only if

$$\left(x - \frac{u_0 + u_1}{2} \right)^\top \Sigma^{-1} (u_1 - u_0) + \log \frac{\pi_1}{\pi_0} \geq 0.$$

- Logistic regression:

$$\log \left(\frac{\mathbb{P}(Y = 1 \mid X = x)}{\mathbb{P}(Y = 0 \mid X = x)} \right) = \beta_0 + \boldsymbol{\beta}^\top x.$$

  Hence, $\mathbb{P}(Y = 1 \mid X = x) \geq \mathbb{P}(Y = 0 \mid X = x)$ if and only if

$$\beta_0 + \boldsymbol{\beta}^\top x \geq 0.$$

A general formulation of classifiers based on a linear decision boundary

Binary classification: predicting a target with two values, $y \in \{-1, +1\}$, (small change from the past).

- Consider the linear decision boundary

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

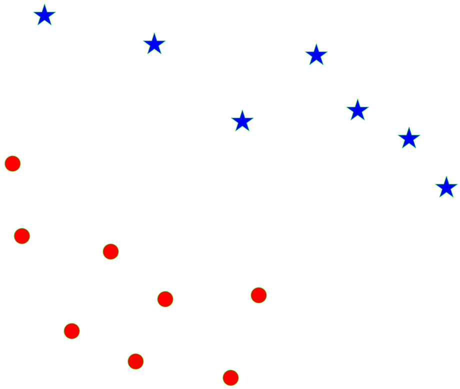
  for some weights $\mathbf{w} \in \mathbb{R}^p$ and $b \in \mathbb{R}$.

- A good decision boundary should satisfy: for a given point $(\mathbf{x}, y)$,

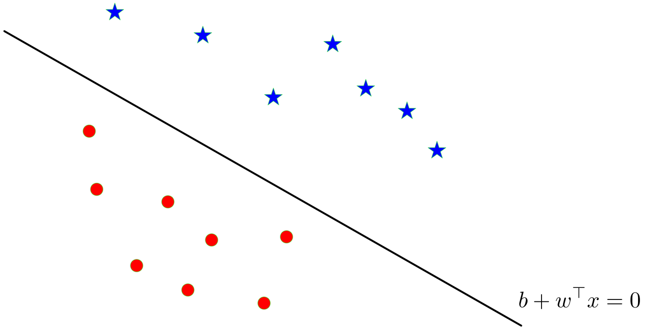
$$\begin{aligned} \mathbf{w}^\top \mathbf{x} + b > 0 \quad & \text{if } y = 1 \\ \mathbf{w}^\top \mathbf{x} + b < 0 \quad & \text{if } y = -1. \end{aligned}$$

Separating Hyperplanes

Suppose we are given these data points from two different classes and want to find a linear classifier that separates them.



Separating Hyperplanes



- The decision boundary looks like a line because $\mathbf{x} \in \mathbb{R}^2$
- $\{\mathbf{x} \in \mathbb{R}^p : \mathbf{w}^\top \mathbf{x} + b = 0\}$ is a $(p-1)$ dimensional space , a.k.a. hyperplane.

Discussion on this simple formulation

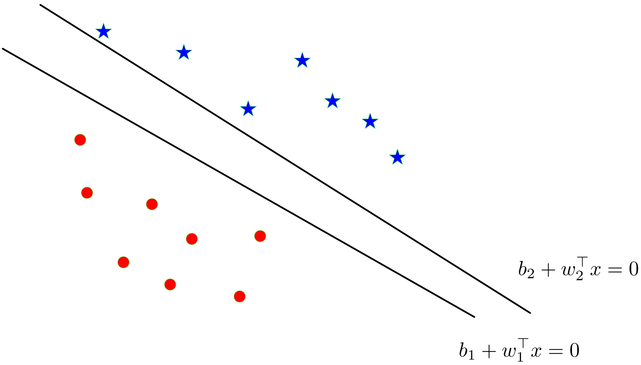
The above intuition leads to the following way of estimating $\mathbf{w}$ and b

$$\min_{\mathbf{w},b} -\sum_{i\in M} y_i(\mathbf{x}_i^\top \mathbf{w} + b)$$

where M indexes the set of misclassified points among the training data $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)$.

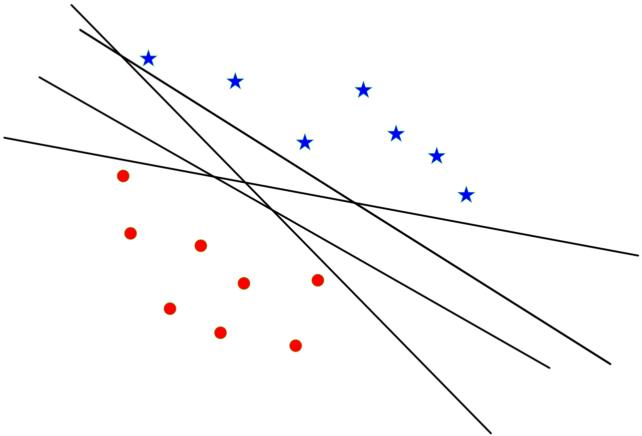
- We could use (sub-)gradient descent to solve.
- However:
  - When the data is separable, there exists multiple solutions of $\mathbf{w}$ and b such that the above loss is zero. Which one should we choose?
  - When the data is not separable, it is oftentimes hard to achieve convergence by using gradient descent.

Separating Hyperplanes



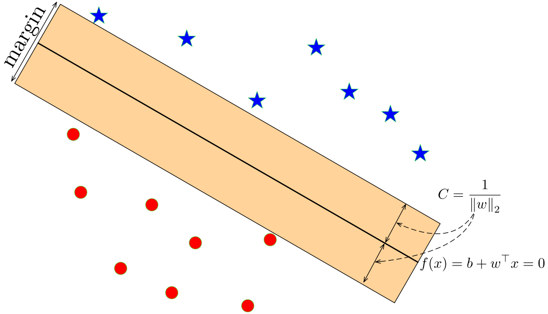
- There are multiple separating hyperplanes, determined by different parameters $(\mathbf{w}, b)$.

Separating Hyperplanes



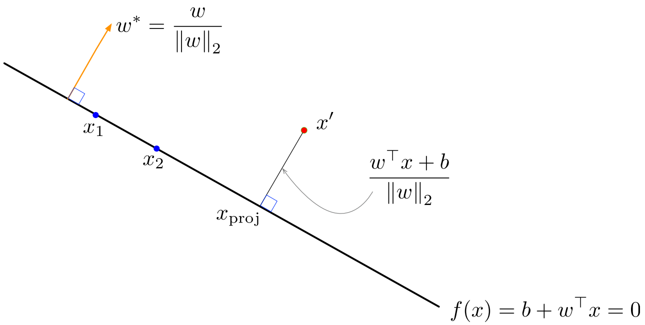
Optimal Separating Hyperplane

Optimal Separating Hyperplane: A hyperplane that separates two classes and maximizes the distance to the closest point from either class, i.e., maximize the margin of the classifier.



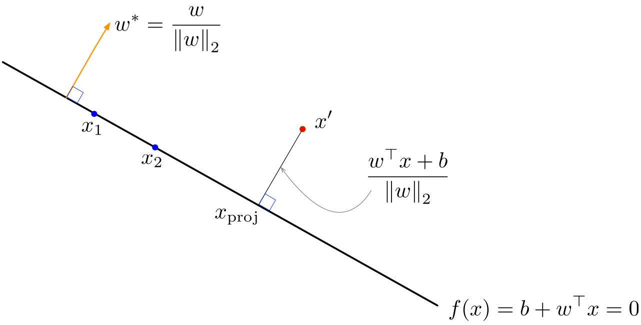
Intuitively, ensuring that a classifier is not too close to any data points leads to better generalization on the test data.

Geometry of Points and Planes

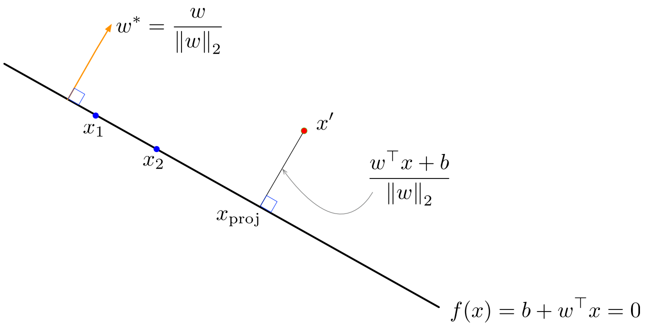


- Recall that the decision hyperplane is orthogonal (perpendicular) to $\mathbf{w}$. I.e., for any two points $\mathbf{x}_1$ and $\mathbf{x}_2$ on the decision hyperplane we have that $\mathbf{w}^\top(\mathbf{x}_1 - \mathbf{x}_2) = 0$.

Geometry of Points and Planes



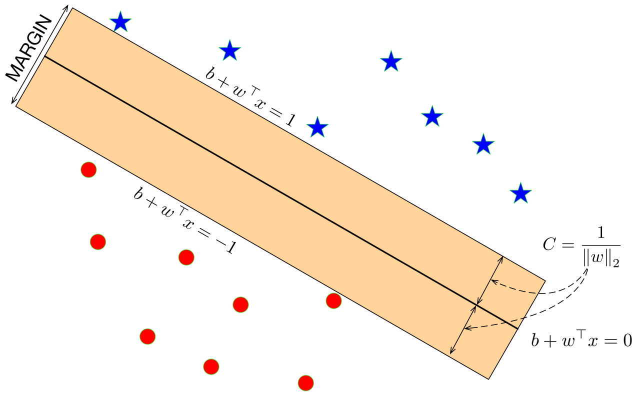
- The vector $\mathbf{w}^* = \frac{\mathbf{w}}{\|\mathbf{w}\|_2}$ is a unit vector pointing in the same direction as $\mathbf{w}$.
- The same hyperplane could equivalently be defined in terms of $\mathbf{w}^*$.



- To get the distance from a point $\mathbf{x}'$ to the hyperplane, take the closest point $\mathbf{x}_{\text{proj}}$ on the hyperplane and project $\mathbf{x}' - \mathbf{x}_{\text{proj}}$ onto $\mathbf{w}/\|\mathbf{w}\|_2$:

$$\|\mathbf{x}' - \mathbf{x}_{\text{proj}}\|_2 = \left|(\mathbf{x}' - \mathbf{x}_{\text{proj}})^\top \frac{\mathbf{w}}{\|\mathbf{w}\|_2}\right| = \frac{\left|\mathbf{x}^\top \mathbf{w} - \mathbf{x}_{\text{proj}}^\top \mathbf{w}\right|}{\|\mathbf{w}\|_2} = \frac{\left|\mathbf{x}^\top \mathbf{w} + b\right|}{\|\mathbf{w}\|_2}$$

Maximizing Margin as an Optimization Problem



- Now consider the two parallel hyperplanes

$$\mathbf{w}^\top \mathbf{x} + b = 1 \qquad \mathbf{w}^\top \mathbf{x} + b = -1$$

- Using the distance formula, can see that the margin is $2/\ \|\mathbf{w}\|_2$.

Maximizing Margin as an Optimization Problem

- Recall: to correctly classify all points we require that

$$\mathrm{sign}(\mathbf{w}^\top \mathbf{x}_i + b) = y_i \qquad \text{for all } i$$

- Let's impose a stronger requirement: correctly classify all points and prevent them from falling in the margin.

$$\begin{aligned} \mathbf{w}^\top \mathbf{x}_i + b &\geq M && \text{if } y_i = 1 \\ \mathbf{w}^\top \mathbf{x}_i + b &\leq -M && \text{if } y_i = -1 \end{aligned}$$

  for some $M > 0$.

- This is equivalent to

$$y_i \left(\mathbf{w}^\top \mathbf{x}_i + b \right) \geq M \qquad \text{for all } i$$

  which we call the margin constraints.

Maximizing Margin as an Optimization Problem

- Now, we want to pick $\mathbf{w}, b$ that maximize the size of the margin (the region where we do not allow points to fall), while ensuring all points are correctly classified.
  - Margin has width
$$\frac{\left|\mathbf{x}^\top \mathbf{w} + b\right|}{||\mathbf{w}||_2} = \frac{M}{||\mathbf{w}||_2},$$
    so maximizing this is equivalent to minimizing $||\mathbf{w}||_2^2 / M$.

- This leads to the max-margin objective:

$$\min_{\mathbf{w},b} \frac{||\mathbf{w}||_2^2}{M}$$
$$\text{s.t. } y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq M \qquad i = 1, \ldots, n$$

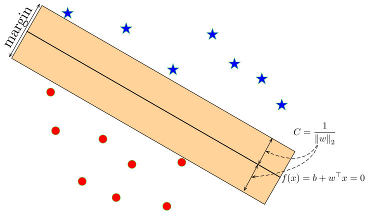
W.l.o.g. we can set $M = 1$. (Why?)

Maximizing Margin as an Optimization Problem

Max-margin objective:

$$\min_{\mathbf{w},b} \|\mathbf{w}\|_2^2$$

$$\text{s.t. } y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 \quad i = 1, \dots, n$$



- Intuitively, if the margin constraint is not tight for $\mathbf{x}_i$, we could remove $\mathbf{x}_i$ from the training set and the optimal $\mathbf{w}$ would be the same. (This can be rigorously shown via the K.K.T. conditions.)
- The important training points are the ones with equality constraints, and are called support vectors.
- Hence, this algorithm is called the (hard-margin) Support Vector Machine (SVM).
- SVM-like algorithms are often called max-margin or large-margin.

Computation of the hard-margin SVM

Primal-formulation:

$$\min_{\mathbf{w},b} ||\mathbf{w}||_2^2$$
$$\text{s.t. } y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 \quad i = 1, \ldots, n$$

- Convex, in fact, a quadratic program. (Stochastic) Gradient descent can be directly used.
- It is more common to solve the optimization problem based on its dual formulation.

Dual-formulation of the hard-margin SVM

For $\alpha_i \geq 0$ for all $i = 1, \ldots, n$, write the Lagrangian function

$$L(\mathbf{w}, b, \boldsymbol{\alpha}) = ||\mathbf{w}||_2^2 + \sum_{i=1}^n \alpha_i \left[1 - y_i(\mathbf{w}^\top \mathbf{x}_i + b)\right],$$

Taking the derivative w.r.t. $\mathbf{w}$ and b yields

$$\mathbf{w} = \frac{1}{2} \sum_{i=1}^n \alpha_i y_i \mathbf{x}_i, \qquad \sum_{i=1}^n \alpha_i y_i = 0.$$

Plugging into $L(\mathbf{w}, b, \boldsymbol{\alpha})$ yields

$$\frac{1}{4} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j \mathbf{x}_i^\top \mathbf{x}_j + \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j \mathbf{x}_i^\top \mathbf{x}_j - b \sum_{i=1}^n \alpha_i y_i$$

$$= \sum_{i=1}^n \alpha_i - \frac{1}{4} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j \mathbf{x}_i^\top \mathbf{x}_j.$$

Dual-formulation of the hard-margin SVM

The dual problem is

$$\max_{\boldsymbol{\alpha}} \sum_{i=1}^{n} \alpha_i - \frac{1}{4} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^\top \mathbf{x}_j$$

$$\text{s.t.} \sum_{i=1}^{n} \alpha_i y_i = 0, \ \alpha_i \geq 0, \quad i = 1, \ldots, n.$$

The K.K.T. conditions ensure the following relationships between the primal and dual formulations.

- Their optimal objective values are equal.
- The optimal solutions $\hat{\mathbf{w}}$ and $\hat{\boldsymbol{\alpha}}$ satisfy

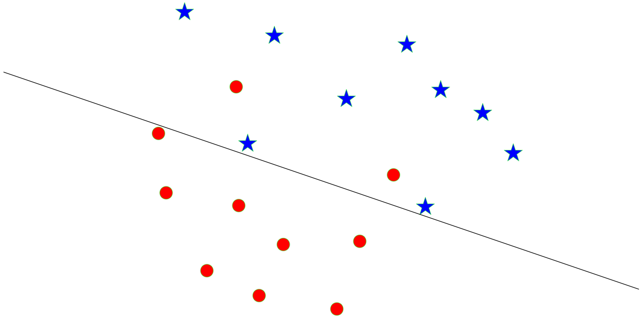
$$\hat{\mathbf{w}} = \frac{1}{2} \sum_{i=1}^{n} \hat{\alpha}_{\mathbf{i}} \, y_i \mathbf{x}_i, \qquad \begin{array}{ll} \hat{\alpha}_i > 0, & \text{if } y_i(\hat{\mathbf{w}}^\top \mathbf{x}_i + \hat{b}) = 1 \\ \hat{\alpha}_i = 0, & \text{if } y_i(\hat{\mathbf{w}}^\top \mathbf{x}_i + \hat{b}) > 1 \end{array}.$$

- The predicted label for any $\mathbf{x}$ is

$$\text{sign}(\hat{\mathbf{w}}^\top \mathbf{x} + \hat{b}).$$

Extension to Non-Separable Data Points

How can we apply the max-margin principle if the data are **not** linearly separable?



Soft-margin SVM

We introduce slack variables $\zeta = (\zeta_1, \ldots, \zeta_n)$ and consider

$$\begin{aligned} \min_{\mathbf{w}, b, \zeta} \quad & \|\mathbf{w}\|_2^2 \\ \text{s.t.} \quad & y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 - \zeta_i, \ \zeta_i \geq 0, \ i = 1, \ldots, n \\ & \sum_{i=1}^{n} \zeta_i \leq K. \end{aligned}$$

- Misclassification occurs if $\zeta_i > 1$.
- $\sum_{i=1}^{n} \zeta_i \leq K$ restricts the total number of misclassified points less than K.
- $K = 0$ reduces to the hard-margin SVM.
- $K \geq 0$ is a tuning parameter.

Another interpretation of the soft-margin SVM

- Soft-margin SVM is equivalent to

$$\min_{\mathbf{w},b,\zeta} \quad \|\mathbf{w}\|_2^2 + C\sum_{i=1}^{n}\zeta_i$$
$$\text{s.t. } \; y_i(\mathbf{w}^\top\mathbf{x}_i + b) \geq 1 - \zeta_i, \;\; \zeta_i \geq 0, \;\; i = 1, \ldots, n$$

  for some $C = C(K)$.

- This is further equivalent to

$$\min_{\mathbf{w},b,\zeta} \quad \left\{ \frac{1}{n}\sum_{i=1}^{n} \underbrace{\max\left\{0, 1 - y_i\left(\mathbf{w}^\top\mathbf{x}_i + b\right)\right\}}_{\text{hinge loss}} \right\} + \lambda\,\|\mathbf{w}\|_2^2$$

  with $\lambda = 1/(nC)$.

- Hence, the soft-margin SVM can be seen as a linear classifier with the hinge loss and the ℓ_2 regularization (ridge penalty).

Dual-formulation of the soft-margin SVM

It can be shown[1] that the dual-formulation of the soft-margin SVM is

$$\max_{\boldsymbol{\alpha}} \ \sum_{i=1}^{n} \alpha_i - \frac{1}{4} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \ \mathbf{x}_i^\top \mathbf{x}_j$$

$$\text{s.t.} \ \sum_{i=1}^{n} \alpha_i y_i = 0, \ 0 \leq \alpha_i \leq C, \quad i = 1, \ldots, n.$$

Here $C > 0$ is the tuning parameter.

[1] Chapter 12.2.1 in ESL.

Kernel SVM: extension to non-linear boundary

Recall

$$\max_{\boldsymbol{\alpha}} \sum_{i=1}^{n} \alpha_i - \frac{1}{4} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \, \mathbf{x_i}^\top \mathbf{x_j}$$

$$\text{s.t. } \sum_{i=1}^{n} \alpha_i y_i = 0, \; 0 \le \alpha_i \le C, \quad i = 1, \ldots, n.$$

Represent $\mathbf{x}_i$ in different bases, $h(\mathbf{x}_i)$, to have non-linear boundary (in $\mathbf{x}_i$).

All we need to change is

$$\max_{\boldsymbol{\alpha}} \sum_{i=1}^{n} \alpha_i - \frac{1}{4} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \, \mathbf{h}(\mathbf{x_i})^\top \mathbf{h}(\mathbf{x_j}).$$

Kernel trick

- We can represent the inner-product $h(\mathbf{x}_i)^\top h(\mathbf{x}_j) = \langle h(\mathbf{x}_i), h(\mathbf{x}_j) \rangle$ by using

$$K(\mathbf{x}_i, \mathbf{x}_j) = h(\mathbf{x}_i)^\top h(\mathbf{x}_j), \qquad \forall i \neq j \in \{1, \ldots, n\}.$$

  The function K is called **kernel** that quantifies the similarity of two feature vectors.

- Regardless how large the space of $h(\mathbf{x}_i)$ is, all we need to compute is the pairwise kernel

$$K(\mathbf{x}_i, \mathbf{x}_j), \qquad \forall i \neq j \in \{1, \ldots, n\}.$$

  This is known as the kernel trick.

Examples of kernel SVM

- Linear:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^\top \mathbf{x}_j$$

  with the corresponding $h(\mathbf{x}_i) = \mathbf{x}_i$.

- dth-Degree polynomial:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \left(1 + \mathbf{x}_i^\top \mathbf{x}_j\right)^d.$$

  The corresponding h would be polynomials. For example, consider $d = 2$, $\mathbf{x}_i = x_i$ and $h(\mathbf{x}_i) = [1, \sqrt{2}x_i, x_i^2]$, then

$$K(\mathbf{x}_i, \mathbf{x}_j) = h(\mathbf{x}_i)^\top h(\mathbf{x}_j) = 1 + 2x_i x_j + x_i^2 x_j^2 = \left(1 + \mathbf{x}_i^\top \mathbf{x}_j\right)^2.$$

- Radial basis: for some $\gamma > 0$,

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|_2^2\right).$$

  The corresponding $h(\mathbf{x}_i)$ has infinite dimensions!

Limitations of SVM

- The classifier based on SVM is
$$\text{sign}(\hat{\mathbf{w}}^\top \mathbf{x} + \hat{b}).$$
Hence, SVM does not estimate the posterior probability.

- For multi-class classification problems,
  - It is non-trivial to generalize the notion of a margin to multiclass setting.
  - Many different proposals for multi-class SVMs. We discuss two commonly used ad-hoc approaches.

SVMs with More than Two Classes

One-Versus-One: Let $C = \{1, 2, \ldots, K\}$.

- Construct $\binom{K}{2}$ SVMs for each pair of classes.
  - For classes {1,2}, consider data $(\mathbf{x}_i, y_i)$ with $y_i \in \{1, 2\}$. Let

    $$z_i = -1\{y_i = 1\} + 1\{y_i = 2\}.$$

    Fit SVM by using $(\mathbf{x}_i, z_i)$ with $y_i \in \{1, 2\}$.
  - For classes {1,3}, consider data $(\mathbf{x}_i, y_i)$ with $y_i \in \{1, 3\}$. Let

    $$z_i = -1\{y_i = 1\} + 1\{y_i = 3\}.$$

    Fit SVM by using $(\mathbf{x}_i, z_i)$ with $y_i \in \{1, 3\}$.
  - Repeat for all pairs.
- For each test point $\mathbf{x}_0$, assign it to the majority class predicted by $\binom{K}{2}$ SVMs.

SVMs with More than Two Classes

One-Versus-All

- Construct K SVMs by choosing each class one at a time.
  - For class $\{1\}$, consider ALL data $(\mathbf{x}_i, y_i)$, $i = 1, \ldots, n$. Let

    $$z_i = 2 \cdot 1\{y_i = 1\} - 1.$$

    Fit SVM and let its parameter be $(\hat{b}^{(1)}, \hat{\mathbf{w}}^{(1)})$.
  - For class $\{2\}$, consider ALL data $(\mathbf{x}_i, y_i)$, $i = 1, \ldots, n$. Let

    $$z_i = 2 \cdot 1\{y_i = 2\} - 1.$$

    Fit SVM and let its parameter be $(\hat{b}^{(2)}, \hat{\mathbf{w}}^{(2)})$.
  - Repeat for all classes.
- For each test point $\mathbf{x}_0$, assign it to the class

$$\arg\max_{k \in C} \left(\hat{b}^{(k)} + \mathbf{x}_0^\top \hat{\mathbf{w}}^{(k)} \right).$$

LDA vs SVM vs Logistic Regression (LR)

- In essence, SVM is more similar as LR than LDA. (LDA makes additional Gaussianity assumptions.)
- SVM does not estimate the probabilities $\mathbb{P}(Y = 1 \mid X)$ but LDA and LR do.
- When classes are (nearly) separable, SVM and LDA perform better than LR.
- When classes are non-separable, LR (with ridge penalty) and SVM are very similar.
- When Gaussianity can be justified, LDA has the best performance.
- SVM and LR are less used for multi-class classification problems.