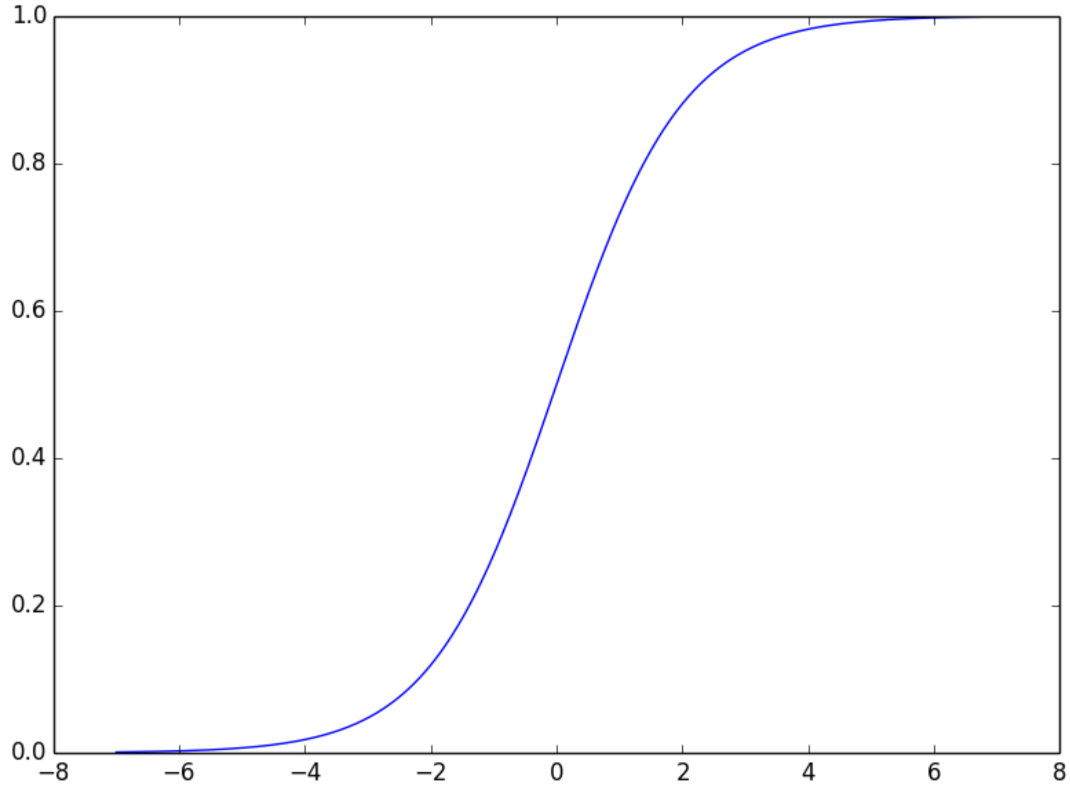
Support Vector Machines vs Logistic Regression

Kevin Swersky

University of Toronto CSC2515 Tutorial

Part of this tutorial is borrowed from Mark Schmidt's excellent note on structural SVMs: http://www.di.ens.fr/~mschmidt/Documents/ssvm.pdf

Logistic regression



Logistic regression

- Assign probability to each outcome

$$P(y = 1|x) = \sigma(w^T x + b)$$

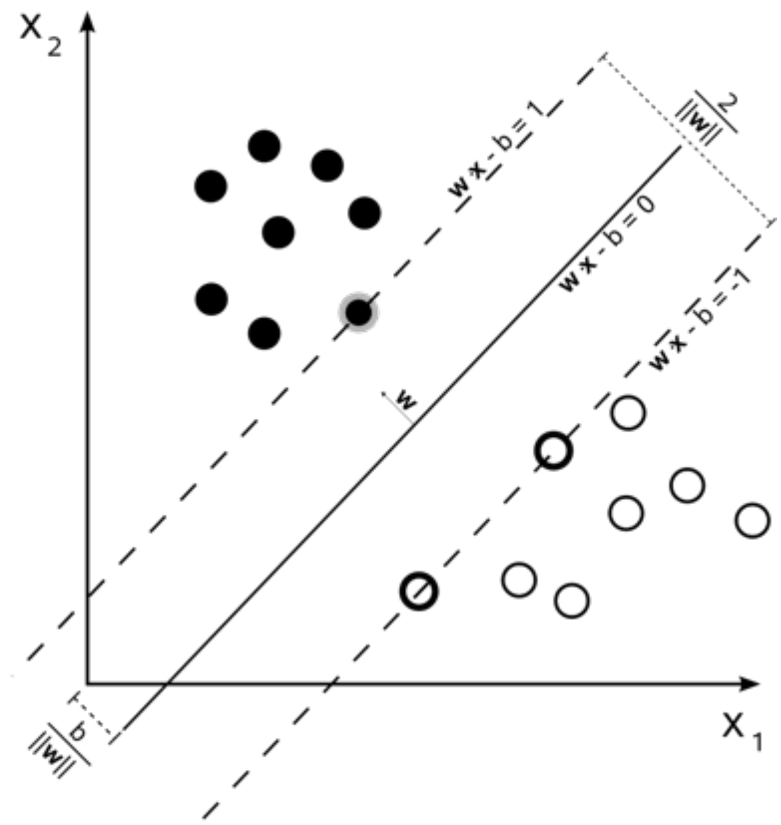
- Train to maximize likelihood

$$l(w) = -\sum_{n=1}^{N} \sigma(w^T x_n + b)^{y_n}(1 - \sigma(w^T x_n + b))^{(1-y_n)}$$

- Linear decision boundary (with y being 0 or 1)

$$\hat{y} = \mathrm{I}[w^T x + b \geq 0]$$

Support vector machines



Source: Wikipedia

Support vector machines

- Enforce a margin of separation (here, $y \in \{0, 1\}$)

$$(2y_n - 1)w^T x_n \geq 1, \ \forall n = 1 \ldots N$$

- Train to find the maximum margin

$$\begin{aligned} \min \quad & \frac{1}{2}||w||^2 \\ \text{s.t.} \quad & (2y_n - 1)(w^T x_n + b) \geq 1, \ \forall n = 1 \ldots N \end{aligned}$$

- Linear decision boundary

$$\hat{y} = \mathrm{I}[w^T x + b \geq 0]$$

Recap

- Logistic regression focuses on maximizing the probability of the data. The farther the data lies from the separating hyperplane (on the correct side), the happier LR is.
- An SVM tries to find the separating hyperplane that maximizes the distance of the closest points to the margin (the support vectors). If a point is not a support vector, it doesn't really matter.

A different take

- Remember, in this example $y \in \{0, 1\}$
- Another take on the LR decision function uses the probabilities instead:

$$\hat{y} = \begin{cases} 1 & \text{if } P(y=1|x) \geq P(y=0|x) \\ 0 & \text{otherwise} \end{cases}$$

$$P(y=1|x) \propto \exp(w^T x + b)$$
$$P(y=0|x) \propto 1$$

A different take

- What if we don't care about getting the right probability, we just want to make the right decision?
- We can express this as a constraint on the likelihood ratio,

$$\frac{P(y=1|x)}{P(y=0|x)} \geq c$$

- For some arbitrary constant c>1.

A different take

- Taking the log of both sides,

$$\log\left(P(y=1|x)\right) - \log\left(P(y=0|x)\right) \geq \log(c)$$

- and plugging in the definition of P,

$$w^T x + b - 0 \geq \log(c)$$
$$\implies (w^T x + b) \geq \log(c)$$

- c is arbitrary, so we pick it to satisfy $\log(c) = 1$

$$w^T x + b \geq 1$$

A different take

- This gives a feasibility problem (specifically the perceptron problem) which may not have a unique solution.
- Instead, put a quadratic penalty on the weights to make the solution unique:

$$\begin{aligned} \min \quad & \frac{1}{2}||w||^2 \\ \text{s.t.} \quad & (2y_n - 1)(w^T x_n + b) \geq 1,\ \forall n = 1 \dots N \end{aligned}$$

- This gives us an SVM!
- We derived an SVM by asking LR to make the right *decisions*.

The likelihood ratio

- The key to this derivation is the likelihood ratio,

$$
\begin{aligned}
r &= \frac{P(y=1|x)}{P(y=0|x)} \\
&= \frac{\exp(w^T x + b)}{1} \\
&= \exp(w^T x + b)
\end{aligned}
$$

- We can think of a classifier as assigning some cost to r.
- Different costs = different classifiers.

LR cost

- Pick $\text{cost}(r) = \log(1 + \frac{1}{r})$
  $= \log(1 + \exp(-(w^T x + b)))$
- This is the LR objective (for a positive example)!

SVM with slack variables

- If the data is not linearly separable, we can change the program to:

$$\begin{aligned} \min \quad & \frac{1}{2}||w||^2 + \sum_{n=1}^{N} \xi_n \\ \text{s.t.} \quad & (2y_n - 1)(w^T x_n + b) \geq 1 - \xi_n, \ \forall n = 1 \ldots N \\ & \xi_n \geq 0, \ \forall n = 1 \ldots N \end{aligned}$$

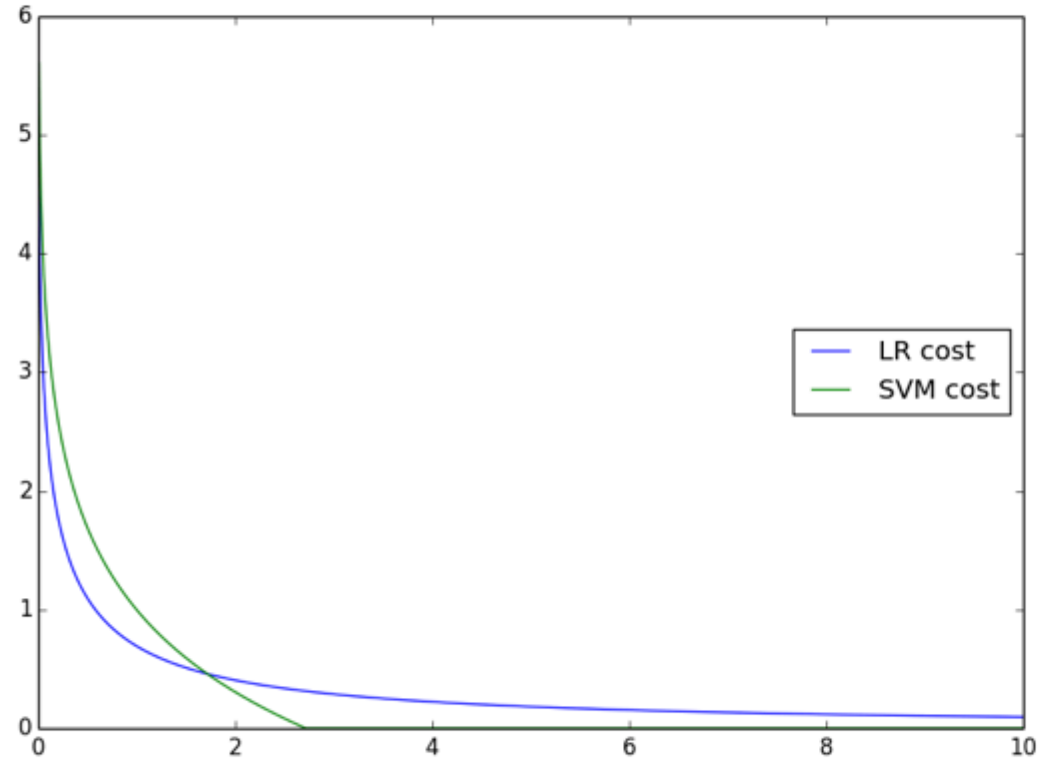
- Now if a point n is misclassified, we incur a cost of ξ_n, it's distance to the margin.

SVM with slack variables cost

- Pick $\begin{aligned} \text{cost}(r) =& \max(0, 1 - \log(r)) \\ =& \max(0, 1 - (w^T x + b)) \end{aligned}$

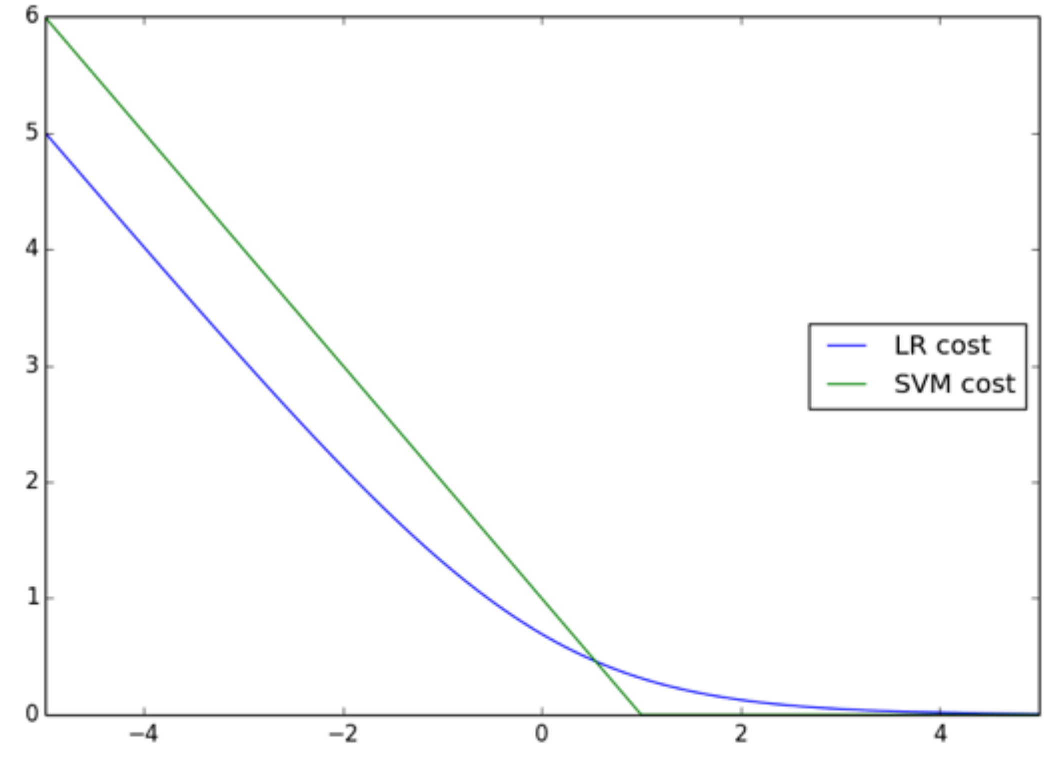
LR cost vs SVM cost

- Plotted in terms of r,



LR cost vs SVM cost

- Plotted in terms of $w^T x + b$,



Exploiting this connection

- We can now use this connection to derive extensions to each method.
- These might seem obvious (maybe not) and that's usually a good thing.
- The important point though is that they are *principled*, rather than just hacks. We can trust that they aren't doing anything crazy.

Kernel trick for LR

- Recall that in it's dual form, we can represent an SVM decision boundary as:

$$w^T\phi(x) + b = \sum_{n=1}^{N} \alpha_n K(x, x_n) = 0$$

- where $\phi(x)$ is an ∞-dimensional basis expansion of x .

- Plugging this into the LR cost:

$$\log(1 + \exp(-\sum_{n=1}^{N} \alpha_n K(x, x_n)))$$

Multi-class SVMs

- Recall for multi-class LR we have:

$$P(y = i|x) = \frac{\exp(w_i^T x + b_i)}{\sum_k \exp(w_k^T x + b_k)}$$

Multi-class SVMs

- Suppose instead we just want the decision rule to satisfy:

$$\frac{P(y=i|x)}{P(y=k|x)} \geq c \;\forall\; k \neq i$$

- Taking logs as before, this gives:

$$w_i^T x - w_k^T x \geq 1 \;\forall\; k \neq i$$

Multi-class SVMs

- This produces the following quadratic program:

$$\begin{aligned} \min \quad & \frac{1}{2}||w||^2 \\ \text{s.t.} \quad & (w_{y_n}^T x_n + b_{y_n}) - (w_k^T x_n + b_k) \geq 1, \; \forall n = 1 \dots N, \; \forall k \neq y_n \end{aligned}$$

Take-home message

- Logistic regression and support vector machines are closely linked.
- Both can be viewed as taking a probabilistic model and minimizing some cost associated with misclassification based on the likelihood ratio.
- This lets us analyze these classifiers in a decision theoretic framework.
- It also allows us to extend them in principled ways.

Which one to use?

- As always, depends on your problem.
- LR gives calibrated probabilities that can be interpreted as confidence in a decision.
- LR gives us an unconstrained, smooth objective.
- LR can be (straightforwardly) used within Bayesian models.
- SVMs don’t penalize examples for which the correct decision is made with sufficient confidence. This may be good for generalization.
- SVMs have a nice dual form, giving sparse solutions when using the kernel trick (better scalability).