Week 12

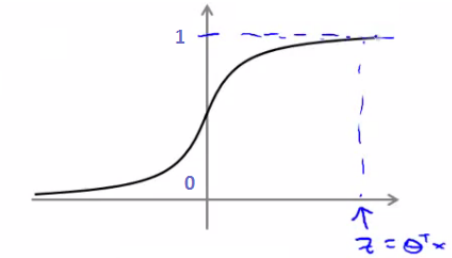
July 31, 2021

An alternative view of logistic regression

As previously stated, the logistic regression hypothesis is as follows:

$$h_\theta(x) = \frac{1}{1 + e^{-\theta^T x}}$$

We have an example in which $y = 1$. We expect that $h_\theta(x)$ is close to 1.

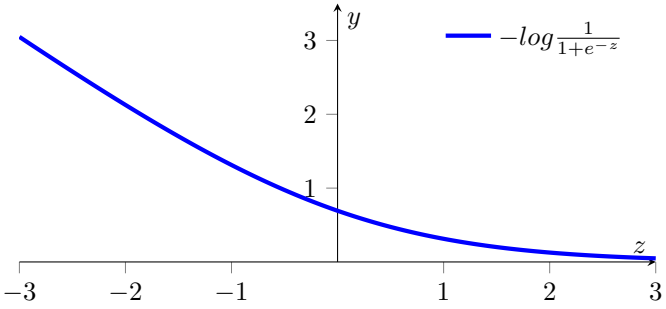


When you look at the cost function, you'll see that each example contributes a term like the one below to the total cost function.

$$-(ylogh_\theta(x) + (1 - y)log(1 - h_\theta(x)))$$

After plugging in the hypothesis function $h_\theta(x)$, you obtain an enlarged cost function equation:

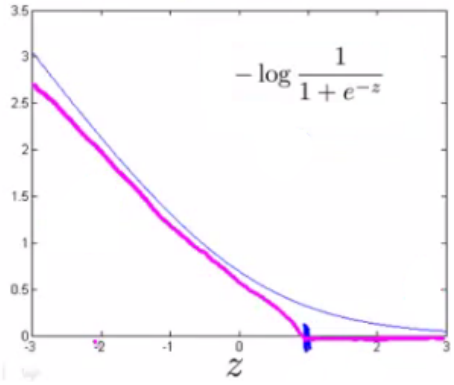
$$-ylog\frac{1}{1 + e^{-\theta^T x}} - (1 - y)log(1 - \frac{1}{1 + e^{-\theta^T x}})$$



- As a result, if z is large, the cost is small.
- If z is 0 or negative, however, the cost contribution is large..
- This is why, when logistic regression encounters a positive case, it attempts to make $\theta^T x$ a very big term.

SVM cost functions from logistic regression cost functions

- Instead of a curved line, draw two straight lines (magenta) to approximate the logistic regression y = 1 function.
- Flat when cost is 0.
- Straight growing line after 1.
- So this is the new y=1 cost function, which provides the SVM with a computational advantage and makes optimization easier.



Logistic regression cost function:

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}[y^{(i)}logh_\theta(x^{(i)}) + (1-y^{(i)})log(1-h_\theta(x^{(i)}))] + \frac{\lambda}{2m}\sum_{j=1}^{m}\theta_j^2$$

For the SVM we take our two logistic regression $y = 1$ and $y = 0$ terms described previously and replace with $cost_1(\theta^T x)$ and $cost_0(\theta^T x)$.

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}[y^{(i)}cost_1(\theta^T x^{(i)}) + (1-y^{(i)})cost_0(\theta^T x^{(i)})] + \frac{\lambda}{2m}\sum_{j=1}^{m}\theta_j^2$$

Which can be rewritten as:

$$J(\theta) = C\sum_{i=1}^{m}[y^{(i)}cost_1(\theta^T x^{(i)}) + (1-y^{(i)})cost_0(\theta^T x^{(i)})] + \frac{1}{2}\sum_{j=1}^{m}\theta_j^2$$

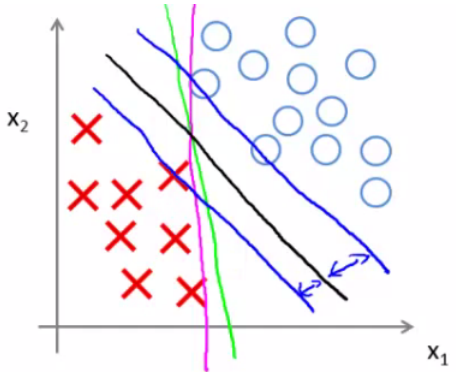
- Large C gives a hypothesis of low bias high variance $->$ overfitting
- Small C gives a hypothesis of high bias low variance $->$ underfitting

Large margin intuition

- So, given that we're aiming to minimize CA + B.
- Consider the following scenario: we set C to be really large.
- If C is large, we will choose an A value such that A equals zero.
- If y = 1, then we must find a value of θ so that $\theta^T x$ is larger than or equal to 1 in order to make our "A" term 0.
- If y = 0, then we must find a value of θ so that $\theta^T x$ is equal to or less than -1 in order to make our "A" term 0.
- So we're minimizing B, under the constraints shown below:

$$min\ \frac{1}{2}\sum_{j=1}^{m}\theta_j^2$$

$$\theta^T x^{(i)} \geq 1 \quad if\ y^{(i)} = 1$$
$$\theta^T x^{(i)} \leq 1 \quad if\ y^{(i)} = 0$$



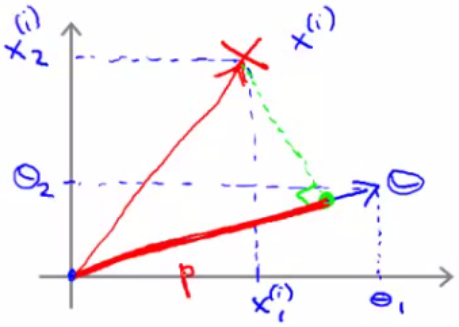
- The green and magenta lines represent functional decision limits that might be selected using logistic regression. However, they are unlikely to generalize effectively.
- The black line, on the other hand, is the one picked by the SVM as a result of the optimization graph's safety net. Stronger separator.
- That black line has a greater minimum distance (margin) than any of the training samples.

SVM decision boundary

Assume we only have two features and $\theta_0 = 0$. Then we can rewrite th expression for minimizing B as follows:

$$\frac{1}{2}(\theta_1^2 + \theta_2^2) = \frac{1}{2}(\sqrt{\theta_1^2 + \theta_2^2})^2 = \frac{1}{2}||\theta||^2$$

- Given this, what are $\theta^T x$ parameters doing?
- Assume we have just one positive training example (red cross below).
- Assume we have our parameter vector and plot it on the same axis.
- The following question asks what the inner product of these two vectors is.



p, is in fact p^i, because it's the length of p for example i.

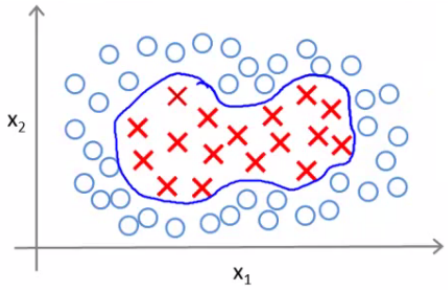
$$\theta^T x^{(i)} = p^i \cdot ||\theta||$$

$$min\ \frac{1}{2}\sum_{j=1}^{m}\theta_j^2 = \frac{1}{2}||\theta||^2$$

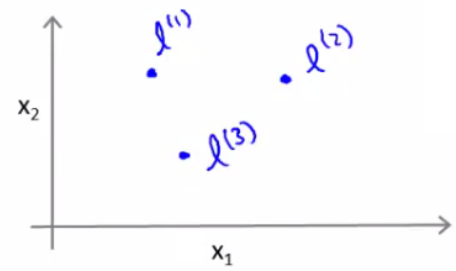
$$p^{(i)} \cdot ||\theta|| \geq 1 \quad if\ y^{(i)} = 1$$
$$p^{(i)} \cdot ||\theta|| \leq 1 \quad if\ y^{(i)} = 0$$

Adapting SVM to non-linear classifiers

- We have a training set.
- We want to find a non-linear boundary.



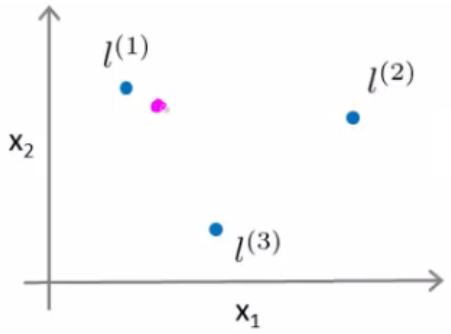
- Define three features in this example (ignore x_0).
- Have a graph of x_1 vs. x_2 (don't plot the values, just define the space).
- Pick three points.



- These points l^1, l^2, and l^3, were chosen manually and are called landmarks.
- Kernel is the name given to the similarity function between (x, l^i).

$$f_1 = k(X, l^1) = exp(-\frac{||x - l^{(1)}||^2}{2\sigma^2})$$

- Large σ^2 - f features vary more smoothly - higher bias, lower variance.
- Small σ^2 - f features vary abruptly - low bias, high variance.
- With training examples x we predict "1" when: $\theta_0 + \theta_1 f_1 + \theta_2 f_2 + \theta_3 f_3 \geq 0$
- Let's say that: $\theta_0 = -0.5,\ \theta_1 = 1,\ \theta_2 = 1,\ \theta_3 = 0$
- Given our placement of three examples, what happens if we evaluate an example at the magenta dot below?



- We can see from our formula that f1 will be close to 1, whereas f2 and f3 will be close to 0.
- We have: $-0.5 + 1 \cdot 1 + 0 \cdot 1 + 0 \cdot 0 \geq 0$.
- The inequality holds. We predict 1.
- If we had another point far away from all three. The inequality wouldn't hold. As a result, we would predict 0.

Choosing the landmarks

- Take the training data. Vectors X and Y, both with m elements.
- As a result, you'll wind up having m landmarks. Each training example has one landmark per location.
- So we just cycle over each landmark, determining how close x^i is to that landmark. Here we are using the kernel function.
- Take these m features $(f_1, f_2 ... f_m)$ group them into an $[m + 1 \times 1]$ dimensional vector called f.

Kernels

- Linear kernel: no kernel, no f vector. Predict $y = 1$ if $(\theta^T x) \geq 0$.
- Not all similarity functions you develop are valid kernels. Must satisfy Merecer's Theorem.
- Polynomial kernel.
- String kernel.
- Chi-squared kernel.
- Histogram intersection kernel.

Logistic regression vs. SVM

- Use logistic regression or SVM with a linear kernel if n (features) is much greater than m (training set).
- If n is small and m is intermediate, the Gaussian kernel is suitable.
- With a Gaussian kernel, SVM will be sluggish if n is small and m is large. Use logistic regression or SVM with a linear kernel.
- A lot of SVM's power is using diferent kernels to learn complex non-linear functions.
- Because SVM is a convex optimization problem, it gives a global minimum.