6.867: Exercises (Week 5)

Sept 29, 2016

Contents

Solution: Don't look at the solutions until you have tried your absolute hardest to solve the problems.

1 Colonel Vector

Consider the kernel

$$K(\underline{x}, \underline{z}) = \underline{x} \cdot \underline{z} + 4\,(\underline{x} \cdot \underline{z})^2$$

where the vectors $\underline{x}$ and $\underline{z}$ are 2-dimensional. This kernel is equal to an inner product $\phi(\underline{x}) \cdot \phi(\underline{z})$ for some definition of ϕ. What is the function ϕ?

Solution: $(\underline{x} \cdot \underline{z})^2 = (x_1 z_1)^2 + 2(x_1 x_2)(z_1 z_2) + (x_2 z_2)^2$, so that

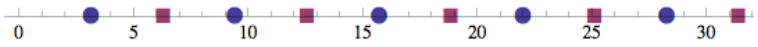
$$\begin{aligned} K(\underline{x}, \underline{z}) &= x_1 z_1 + x_2 z_2 + 4(x_1 z_1)^2 + 8(x_1 x_2)(z_1 z_2) + 4(x_2 z_2)^2 \\ &= [x_1, x_2, 2x_1^2, 2\sqrt{2}x_1 x_2, 2x_2^2] \cdot [z_1, z_2, 2z_1^2, 2\sqrt{2}z_1 z_2, 2z_2^2] \end{aligned}$$

Thus $\phi(\underline{x}) = [x_1, x_2, 2x_1^2, 2\sqrt{2}x_1 x_2, 2x_2^2]$.

2 Kernels of truth

Consider the data set, where the input is a one-dimensional real:

$$\{(\pi, -1), (2\pi, +1), (3\pi, -1), (4\pi, +1), (5\pi, -1), (6\pi, +1), (7\pi, -1), (8\pi, +1), (9\pi, -1), (10\pi, +1)\} \ .$$



For each of the kernels, indicate whether it can separate the data exactly.

(a) $K(x, y) = (xy)^2$

Solution: No

(b) $K(x, y) = (xy)^{20}$

Solution: No

(c) $K(x, y) = (xy + 1)^2$

Solution: No

(d) $K(x, y) = (xy + 100)^{20}$

Solution: Yes

(e) $K(x, y) = e^{-10(x-y)^2}$

Solution: Yes

(f) $K(x, y) = e^{-0.1(x-y)^2}$

Solution: Yes

(g) $K(x, y) = \cos(x)\cos(y)$

Solution: Yes

(h) $K(x, y) = \sin(x)\sin(y)$

Solution: No

3 SVM with 3 points

Consider a simple classification problem (of the kind that you could only encounter in an exam). The training data consist of only three labeled points

$$(x_1 = -1, y_1 = +1), (x_2 = 0, y_2 = -1), (x_3 = +1, y_3 = +1)$$

which we will try to separate with a linear classifier through origin in the feature space. In other words, our discriminant function is of the form $\underline{\theta} \cdot \underline{\phi}(x)$. The corresponding primal and dual estimation problems are given by

$$\textbf{Primal:}\text{Minimize} \quad \frac{1}{2}\|\underline{\theta}\|^2 + C\sum_{i=1}^{3} \xi_i \tag{3.1}$$

$$\text{subject to} \quad y_i(\underline{\theta} \cdot \underline{\phi}(x_i)) \geqslant 1 - \xi_i, \quad \xi_i \geqslant 0, \quad i = 1, 2, 3 \tag{3.2}$$

$$\tag{3.3}$$

$$\textbf{Dual:}\text{Maximize} \quad \sum_{i=1}^{3} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{3} \alpha_i \alpha_j y_i y_j K(x_i, x_j) \tag{3.4}$$

$$\text{subject to} \quad 0 \leqslant \alpha_i \leqslant C, \quad i = 1, 2, 3 \tag{3.5}$$

(a) We decided to solve the problem in the dual using kernel $K(x, x') = 1 + |xx'|$ where $|\cdot|$ is the absolute value. What is the feature mapping $\underline{\phi}(x)$ corresponding to this kernel?

Solution: $\underline{\phi}(\underline{x}) = (1, |x|)^T$

(b) Using this kernel (feature mapping), are the three training examples linearly separable through origin in the feature space?

Solution: Yes.

(c) If we decrease the slack penalty C, the solution might not satisfy the margin constraint for $(x_2 = 0, y_2 = -1)$ without a positive slack $\xi_2 > 0$. What does this mean in terms of α_2?

Solution: $\alpha_2 = C$

(d) Assume $K(x, x') = 1 + |xx'|$ and the three point training set. Express the value of the discriminant function in the dual form for $x_2 = 0$. If we set $C < 1$, do we necessarily get a positive slack ($\xi_2 > 0$) for this example $(x_2 = 0, y_2 = -1)$? Briefly justify your answer.

Solution: ξ_t will be non-zero. The margin constraint for $(x_2 = 0, y_2 = -1)$ without slack requires that

$$\begin{aligned} y_2(\alpha_1 y_1 K(0,-1) + \alpha_2 y_2 K(0,0) + \alpha_3 y_3 K(0,1)) &= -1(\alpha_1 - \alpha_2 + \alpha_3) \\ &= \alpha_2 - \alpha_1 - \alpha_3 \\ &\geqslant 1 \end{aligned}$$

However, this constraint cannot be satisfied with $0 \leqslant \alpha_2 \leqslant C < 1$.

4 Slacking off

Consider training an SVM with slack variables, but with no bias variable. The kernel used is $K(\underline{x}, \underline{z})$; it has the property that for any two points $\underline{x}_i$ and $\underline{x}_j$ in the training set, $-1 < K(\underline{x}_i, \underline{x}_j) < 1$. $K(\underline{x}_i, \underline{x}_i) < 1$ as well. There are n points in the training set. Show that if the slack-variable constant C is chosen such that $C < \frac{1}{n-1}$, then all dual variables α_i are non-zero (i.e., all points in the training set become support vectors).

Solution: In the absence of bias,
$\underline{\theta} \cdot \underline{\phi}(\underline{x}_i) = \sum_{j=1}^{n} \alpha_j y_j \underline{\phi}(\underline{x}_j) \cdot \underline{\phi}(\underline{x}_i) = \sum_{j=1}^{n} \alpha_j y_j K(\underline{x}_j, \underline{x}_i)$
A point has to be a support vector unless

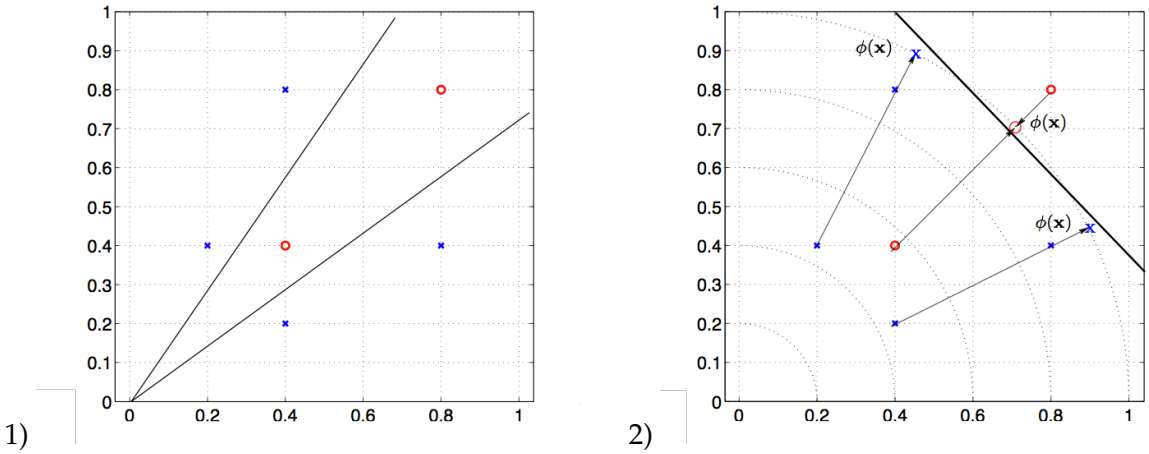
$$y_i \left(\sum_{j=1}^{n} \alpha_j y_j K(\underline{x}_j, \underline{x}_i) \right) \geqslant 1$$

holds with the help of the other examples, i.e., with $\alpha_i = 0$. We will show that this cannot happen. Assuming $\alpha_i = 0$ (not a support vector), then

$$y_i \left(\sum_{j \neq 1}^{n} \alpha_j y_j K(x_j, x_i) \right) \leqslant y_i \left(\sum_{j \neq 1}^{n} \alpha_j |K(x_j, x_i)| \right) \leqslant \sum_{j \neq i}^{n} \alpha_j < (n-1)C$$

So, if $C < 1/(n-1)$, we cannot satisfy the constraint without $\alpha_i > 0$.

5 Almost a linear kernel



1) Points that should be separable with a normalized linear kernel. 2) feature space with the original points overlaid with their original coordinate values.

A student in a machine learning course claimed that the points in part 1) above can be separated with "almost a linear kernel". Hard to believe, we responded, since the points are clearly not linearly separable. But the student insisted. The "almost a linear kernel" they had in mind was the following normalized kernel:

$$K_{norm}(\mathbf{x}, \mathbf{x}') = \frac{\mathbf{x}^T \mathbf{x}'}{\|\mathbf{x}\| \|\mathbf{x}'\|}$$

(a) What are the feature vectors corresponding to this kernel?

Solution: The feature vectors are just $\phi(\mathbf{x}) = \mathbf{x}/\|\mathbf{x}\|$. These are two dimensional vectors (although they only vary along the unit circle).

(b) Using figure 2 (right), graphically map the points to their new feature representation using the figure as the feature space.

Solution: The points are mapped radially to the unit circle (the largest dotted circle in the figure)

(c) Draw the resulting maximum margin decision boundary in the feature space. Use the same figure 2 (right). The student was right, the points are separable!

Solution: See the figure.

(d) Does the value of the discriminant function corresponding to your solution change if we scale any point, i.e., evaluate it at $s\,\mathbf{x}$ instead of $\mathbf{x}$ for some $s > 0$? **(Y/N)**

Solution: No.

(e) Draw the decision boundary in the original input space resulting from the normalized linear kernel. Use Figure 1 (left).

Solution: The maximum margin boundary in the feature space crosses the unit circle in two places. These are the feature vectors right on the boundary. Since points that are already normalized map onto themselves in the feature space, these are also points right on the boundary in the original space. We know that scaling doesnâĂŹt affect the discriminant function and thus you can simply draw lines from these points on the unit circle to/from the origin to get the decision boundary in the original space.

6 Kernel decision boundaries

The figure belowplots SVM decision boundaries resulting from using different kernels and/or different slack penalties. The methods used to generate the plots are listed below but (the absent minded) professor forgot to label them. Please assign the plots to the right method. Oh, we also forgot to list one of the methods.

Primal Method with slack penalties:

$$\min \frac{1}{2}\|\underline{\theta}\|^2 + C\sum_{t=1}^{n}\xi_t \;\text{ s.t. } \xi_t \geqslant 0,\;\; y_t(\underline{\theta}^T\underline{x}_t + \theta_0) - 1 + \xi_t \geqslant 0,\;\; t = 1,\ldots,n$$

Dual Method:

$$\max \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j K(\underline{x}_i,\underline{x}_j)\alpha_i \geqslant 0,\; i = 1,\ldots,n,\;\; \sum_{i=1}^{n}\alpha_i y_i = 0$$

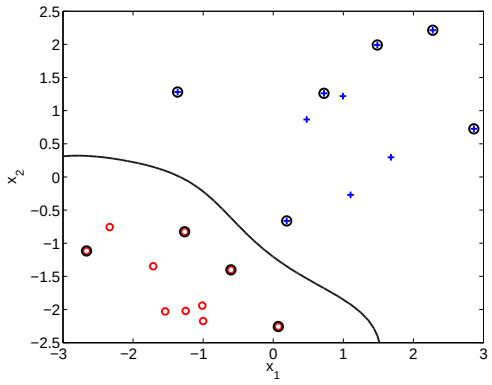
(a) Primal method where $C = 0.1$.

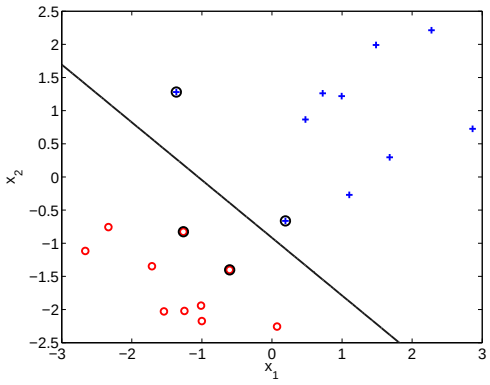
(b) Primal method where $C = 1$.

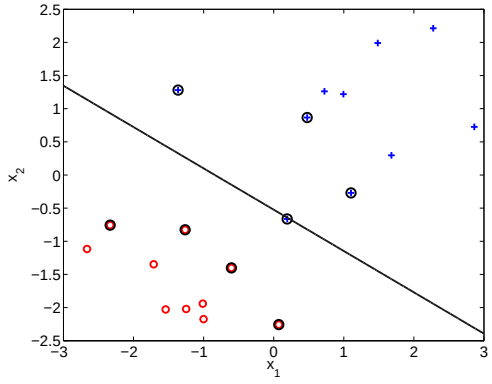
(c) Dual method where $K(\underline{x}, \underline{x}') = \underline{x}^T\underline{x}' + (\underline{x}^T\underline{x}')^2$.

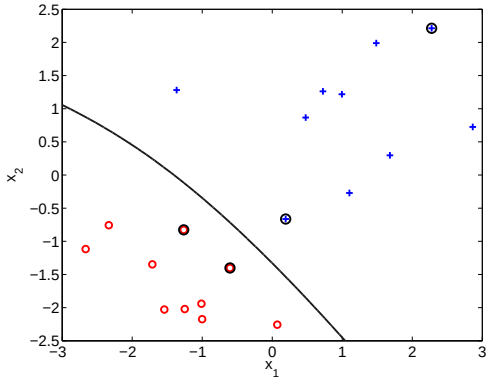
(d) Dual method where $K(\underline{x}, \underline{x}') = \exp(-1/2\|\underline{x} - \underline{x}'\|^2)$.

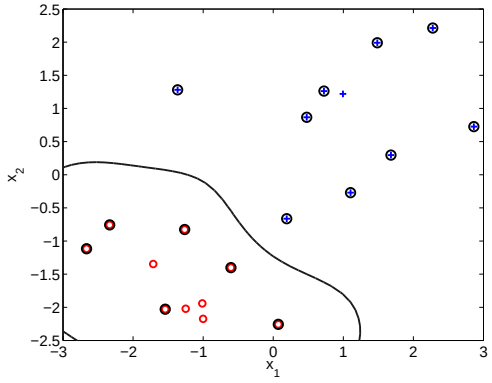
(e) Dual method where $K(\underline{x}, \underline{x}') = \exp(-\|\underline{x} - \underline{x}'\|^2)$.

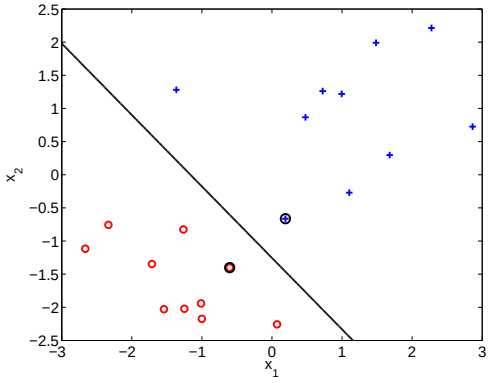
1) 

2) 

3) 

4) 

5) 

6) 

(f) Consider the linear SVM with slack penalties (primal method with slack penalties above):

Indicate which of the following statements hold as we *increase* the parameter C from any starting value. Use 'Y' for statements that *will necessarily hold*, 'N' if the statement is *never true*, and 'D' if the validity of the statement depends on the situation when C increases.

( ) θ_0 will not increase

( ) $\|\underline{\hat{\theta}}\|$ increases

( ) $\|\underline{\hat{\theta}}\|$ will not decrease

( ) more points will be misclassified

( ) the geometric margin for the problem will not increase

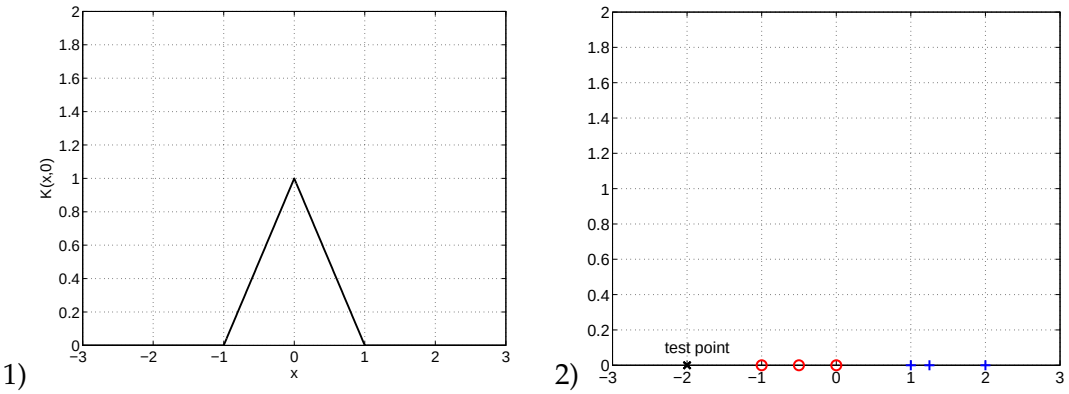
Solution:

6.0.0.1 a-e 3, 2, 4, 1, 5

f

(D) θ_0 will not increase
(D) $\|\underline{\hat{\theta}}\|$ increases
(Y) $\|\underline{\hat{\theta}}\|$ will not decrease
(N) more points will be misclassified
(Y) the geometric margin will not increase

7 1D Classification



1) 2)

Figure 3. 1) Kernel $K(x, 0)$ for problem 3. 2) data for problems 3.b and 3.c.

Consider solving a 1-dimensional classification problem with SVMs and the kernel

$$K(x, x') = (1 - |x - x'|)^+ = \max\{0,\ 1 - |x - x'|\}$$

Figure 3.1) illustrates this kernel $K(x, 0)$ as a function of x. The feature "vectors" corresponding to this kernel are actually functions $\phi(x)(i)$ such that

$$K(x, x') = \int_{-\infty}^{\infty} \phi(x)(i)\phi(x')(i)di$$

(a) What is the form of $\phi(x)(i)$?

Solution: $\phi(x)(i)$ is a box of height 1 and width 1 centered at of $i = x$, so it extends from $x - 1/2$ to $x + 1/2$. You can see that the integral above yields the triangular kernel.

(b) What is the dual objective function for training SVMs (no slack) when we do not include the offset term θ_0 in the classifier? We maximize

$$\sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j K(x_i, x_j)$$

subject to ?

Solution: $\alpha_i \geqslant 0$. Note that the additional constraint $\sum_{i=1}^{n} \alpha_i y_i = 0$ is missing as it came about because of the offset term.

(c) What is the value of the discriminant function $\sum_{i \in n} \hat{\alpha}_i y_i K(x, x_i)$ on the test point in Figure 3.2)? Assume that $\hat{\alpha}_i$ are estimated on the basis of the training data in the figure without an offset parameter.

Solution: 0. The kernel $K(x, x_i)$ is zero for any points that are further than 1 apart. None of the training examples are that close to the test point.

(d) Would the test point in Figure 3.2) become a support vector if it were included in the training set?

Solution: Yes. Otherwise, we could not ensure that the margin for that point is $\geqslant 1$.

(e) We can improve the kernel function a bit by introducing a width parameter σ such that

$$K(x, x') = (1 - |x - x'|/\sigma)^+$$

What would be a reasonable method for choosing σ?

Solution: Leave-one-out cross-validation would ensure, for example, that the discriminant function, trained on the basis of $n - 1$ points, would have a non-zero value for the held-out point. In other words, none of the training points would be as problematic as the above test point.

(f) Would your method solve the problem identified in 3.c? Briefly explain why or why not.

Solution: No, because the training points are close enough together that the appropriate leave- one-out value for σ would not be large enough for the kernel $K(x, x_i; \sigma)$ to extend over the test point.

(g) It is sometimes useful to incorporate test inputs (if available) in some manner in training the classifier. How could you include the test points in selecting the kernel width parameter σ?

Solution: You could, for example, set σ such that the value of the discriminant function is sufficiently far away from zero for all the test points. In other words, it would be clear how to classify the test points.

8 Radial basis kernel

This is a difficult question but it gets at an interesting and important point.

We can write the radial basis kernel in the following form:

$$K(x, x') = \exp\{-\frac{1}{2\sigma^2}\|x - x'\|^2\},$$

where σ is a width parameter specifying how quickly the kernel vanishes as the points move further away from each other. This kernel has some remarkable properties. Indeed, we can perfectly separate *any* finite set of *distinct* training points. Moreover, this result holds for any positive finite value of σ. While the kernel width does not affect whether we'll be able to perfectly separate the training points, it does affect generalization performance. We will try to understand both of these issues a bit better.

Let's proceed in stages. To make things easier we are going to prove a bit stronger result than we need to. In particular, we'll show that

$$\text{minimize } \frac{1}{2}\|\theta\|^2 \text{ subject to } y^i\theta \cdot \phi(x^i) = 1,\ i = 1, \ldots, n$$

has a solution regardless of how we set the ± 1 training labels y^i. You should convince yourself first that this is consistent with our goal. Here $\phi(x^i)$ is the feature vector (function actually) corresponding to the radial basis kernel. Our formulation here is a bit non-standard for two reasons. We try to find a solution where *all* the points are support vectors. This is not possible for all valid kernels but makes it easier to prove the result. We also omit the bias term since it is not needed for the result.

1. Introduce Lagrange multipliers for the constraints similarly to finding the SVM solution (see also the tutorial on Lagrange multipliers that has been posted) and show the form that the solution $\hat{\theta}$ has to take. You can assume that θ and $\phi(x^i)$ are finite vectors for the purposes of these calculations. Note that the Lagrange multipliers here are no longer constrained to be positive. Since you are trying to satisfy equality constraints, the Lagrange multipliers can take any real value.

We are after $\hat{\theta}$ as a function of the Lagrange multipliers. (this should not involve lengthy calculations).

Solution: The Lagrangian for this optimization problem is:

$$\begin{aligned} L(\boldsymbol{\theta}, \boldsymbol{\alpha}) &= \frac{1}{2}\|\boldsymbol{\theta}\|^2 - \sum_{i=1}^{n} \alpha_i \left(y^i \boldsymbol{\theta} \cdot \boldsymbol{\phi}(\boldsymbol{x}^i) - 1 \right) \\ &= \frac{1}{2}\|\boldsymbol{\theta}\|^2 - \boldsymbol{\theta} \cdot \left(\sum_{i=1}^{n} \alpha_i y^i \boldsymbol{\phi}(\boldsymbol{x}^i) \right) + \sum_{i=1}^{n} \alpha_i \end{aligned} \tag{8.1}$$

Here each α_i is *unconstrained*, because we have equality constraints rather than inequality constraints.

As usual, the dual optimization problem is $\max_{\boldsymbol{\alpha}} g(\boldsymbol{\alpha}) = \max_{\boldsymbol{\alpha}} \min_{\boldsymbol{\theta}} L(\boldsymbol{\theta}, \boldsymbol{\alpha})$. For a fixed $\boldsymbol{\alpha}$ (namely some optimal $\boldsymbol{\alpha}^*$), $L(\boldsymbol{\theta}, \boldsymbol{\alpha})$ is positively quadratic in $\boldsymbol{\theta}$. We can obtain the optimal $\boldsymbol{\theta}^*$ from the first-order condition $\frac{\partial L(\boldsymbol{\theta}, \boldsymbol{\alpha})}{\partial \boldsymbol{\theta}} = 0$:

$$\boldsymbol{\theta}^* = \sum_{j=1}^{n} \alpha_j^* y^j \phi(\boldsymbol{x}^j) \tag{8.2}$$

For convenience, we will use the short-hand $\boldsymbol{\theta}^* = \Phi(\boldsymbol{y} \bullet \boldsymbol{\alpha}^*)$. Here $\bullet$ represents an element-wise product and Φ is a $d \times n$ matrix, where the i^{th} column is $\boldsymbol{\phi}(\boldsymbol{x}^i)$. (Of course, $d = \infty$ for the RBF kernel.)

2. Put the resulting solution back into the classification (margin) constraints and express the result in terms of a linear combination of the radial basis kernels.

Solution: Substituting the form above into the constraints gives:

$$y^i \boldsymbol{\theta} \cdot \boldsymbol{\phi}(\boldsymbol{x}^i) = 1 \quad \Longrightarrow \quad y^i \left(\sum_{j=1}^{n} \alpha_j y^j \boldsymbol{\phi}(\boldsymbol{x}^j) \right) \cdot \boldsymbol{\phi}(\boldsymbol{x}^i) = 1 \quad \Longrightarrow \quad y^i \sum_{j=1}^{n} \alpha_j y^j K(\boldsymbol{x}^j, \boldsymbol{x}^i) = 1 \tag{8.3}$$

which is a linear combination of the kernels. It will actually be more convenient to work in matrix form, so we will rewrite this result. Since $y^i = \pm 1, y^i = \frac{1}{y^i}$, our constraints are equivalent to:

$$\boldsymbol{\phi}(\boldsymbol{x}^i)^T \boldsymbol{\theta} = y^i, \; i = 1, \ldots, n \tag{8.4}$$

Using matrix short-hand notation and substituting $\boldsymbol{\theta}^* = \Phi(\boldsymbol{y} \bullet \boldsymbol{\alpha}^*)$, we obtain:

$$\Phi^T \boldsymbol{\theta} = \boldsymbol{y} \quad \Longrightarrow \quad \Phi^T \Phi(\boldsymbol{y} \bullet \boldsymbol{\alpha}^*) = \boldsymbol{y} \quad \Longrightarrow \quad K(\boldsymbol{y} \bullet \boldsymbol{\alpha}^*) = \boldsymbol{y} \tag{8.5}$$

where $K = \Phi^T \Phi$ denotes the kernel matrix.

3. Indicate briefly how we can use the following Michelli theorem to show that any n by n RBF kernel matrix $K_{ij} = K(x^i, x^j)$ for $i, j = 1, \ldots, n$ is invertible.

   Theorem: *If $\rho(t)$ is monotonic function in $t \in [0, \infty)$, then the matrix $\rho_{ij} = \rho(\|x^i - x^j\|)$ is invertible for any distinct set of points x^i, $i = 1, \ldots, n$.*

   Solution: Note that $\rho(t) = \exp\{-\frac{1}{2\sigma^2}t^2\}$ is a monotonic function in $t \in [0, \infty)$. Using the Michelli theorem, for any distinct set of points $\boldsymbol{x}^i$, the matrix K with entries $K_{ij} = \exp\{-\frac{1}{2\sigma^2}\|\boldsymbol{x}^i - \boldsymbol{x}^j\|^2\}$ is invertible.

4. Based on the above results put together the argument to show that we can indeed find a solution where all the points are support vectors.

   Solution: As we have a distinct set of points, K is invertible. Then the linear system $K(\boldsymbol{y} \bullet \boldsymbol{\alpha}^*) = \boldsymbol{y}$ is feasible and can be rewritten as $\boldsymbol{y} \bullet \boldsymbol{\alpha}^* = K^{-1}\boldsymbol{y}$. Therefore, $\boldsymbol{\theta}^* = \Phi(\boldsymbol{y} \bullet \boldsymbol{\alpha}^*) = \Phi K^{-1}\boldsymbol{y}$. Recalling again that each y^i is equal to its reciprocal, $\boldsymbol{\alpha}$ has a unique solution given by $\boldsymbol{\alpha}^* = \boldsymbol{y} \bullet (K^{-1}\boldsymbol{y})$. This means that the constraints in the original problem can all be satisfied, which by construction implies that for this solution, all points are support vectors.

5. Of course, the fact that we can in principle separate any set of training examples does not mean that our classifier does well (on the contrary). So, why do we use the radial basis kernel? The reason has to do with margin that we can attain by varying σ. Note that the effect of varying σ on the margin is not simple rescaling of the feature vectors. Indeed, for the radial basis kernel we have

   $$\phi(x) \cdot \phi(x) = K(x, x) = 1$$

   Let's begin by setting σ to a very small positive value. What is the margin that we attain in response to any n distinct training points?

   Solution: As $\sigma \to 0$, the points become very far apart with respect to σ, and our kernel matrix $K \to I$, the identity matrix. Because our constraints dictate that $K(\boldsymbol{y} \bullet \boldsymbol{\alpha}^*) = \boldsymbol{y}$, then $\boldsymbol{\alpha}^* \to \mathbf{1}$, the all-ones vector. Therefore $\boldsymbol{\theta}^* = \Phi(\boldsymbol{y} \bullet \boldsymbol{\alpha}^*) \to \Phi\boldsymbol{y}$, and

   $$\|\boldsymbol{\theta}^*\|^2 = \boldsymbol{\theta}^{*T}\boldsymbol{\theta}^* \to (\Phi\boldsymbol{y})^T(\Phi\boldsymbol{y}) = \boldsymbol{y}^T(\Phi^T\Phi)\boldsymbol{y} = \boldsymbol{y}^T K\boldsymbol{y} = \boldsymbol{y}^T I\boldsymbol{y} = \sum_{i=1}^{n}(y^i)^2 = n \qquad (8.6)$$

   The margin is $\frac{1}{\|\boldsymbol{\theta}\|}$, hence we obtain a margin of $\frac{1}{\sqrt{n}}$ in the limit.

   As $\sigma \to 0$, we can intuitively think of the kernel centered on $\boldsymbol{x}^i$, $K(\cdot, \boldsymbol{x}^i)$, as becoming a delta function $\delta(\cdot, \boldsymbol{x}^i)$. So consider the n-dimensional feature mapping $\boldsymbol{\phi}'(\cdot)$, where the i^{th} component feature is $\delta(\cdot, \boldsymbol{x}^i)$. In effect, the i^{th} data point then becomes the i^{th} standard basis vector $\boldsymbol{e_i}$ in the limit. The geometric margin for the set of standard basis vectors $\{\boldsymbol{e_i}\}$ is $\frac{1}{\sqrt{n}}$.

6. Provide a 1-dimensional example to show how the margin can be larger than the answer to part 5. You are free to set σ and the points so as to highlight how they might "contribute to each other's margin".

Solution: The simplest example to create is a set of 2 distinct points $\boldsymbol{x}$ and $\boldsymbol{x}'$, both labeled $+1$. Denote $k = K(\boldsymbol{x}, \boldsymbol{x}') = \exp\{-\frac{1}{2\sigma^2}\|\boldsymbol{x} - \boldsymbol{x}'\|^2\}$. The kernel matrix can be written as:

$$K = \begin{bmatrix} 1 & k \\ k & 1 \end{bmatrix}$$

Solving the system $K(\mathbf{1} \bullet \boldsymbol{\alpha}) = \mathbf{1}$ yields $\boldsymbol{\alpha} = [\frac{1}{k+1} \ \frac{1}{k+1}]^T$ and $\|\boldsymbol{\theta}^*\|^2 = \boldsymbol{\alpha} K \boldsymbol{\alpha} = \frac{2}{k+1}$. Therefore, the margin is $\sqrt{\frac{k+1}{2}}$. For distinct $\boldsymbol{x}$ and $\boldsymbol{x}'$, we have that $k > 0$, so the margin is always greater than $\frac{1}{\sqrt{2}}$.

9 Using only positive training examples

One evening we thought we had come up with a great machine learning approach to predicting movie ratings. The idea was to base the predictions solely on positive training examples, movies we already know we like ($y = +1$), and simply ignore (as far as the training is concerned) all the negative examples ($y = -1$). Assume movies are represented by vectors $\underline{x}_1, \ldots, \underline{x}_m$, where $\underline{x}_j \in \mathcal{R}^d$. We created these vectors from movie descriptions (automatically, of course).

Our primal SVM optimization problem, written only for positive examples without offset, is given by

$$\min \ \frac{1}{2}\|\underline{\theta}\|^2 \quad \text{subject to } \underline{\theta} \cdot \underline{x}_j \geqslant 1, \ \ j \in J_+ \tag{9.1}$$

where $J_+ \subset \{1, \ldots, m\}$ indexes our positive training examples (movies we already know we like).

1. What would the solution $\hat{\theta}$ be if we included an offset parameter θ_0, i.e., changed the constraints to be $\underline{\theta} \cdot \underline{x}_j + \theta_0 \geqslant 1$?

Solution: The solution would be $\hat{\theta} = \underline{0}$ since the constraints could always be satisfied by setting the offset parameter appropriately ($\theta_0 = 1$ would suffice).

2. Assume we can find the solution $\hat{\underline{\theta}}$ to the problem described in Eq.(9.1). What is the value of $\min_{j \in J_+}(\hat{\underline{\theta}} \cdot \underline{x}_j)$?

Solution: 1

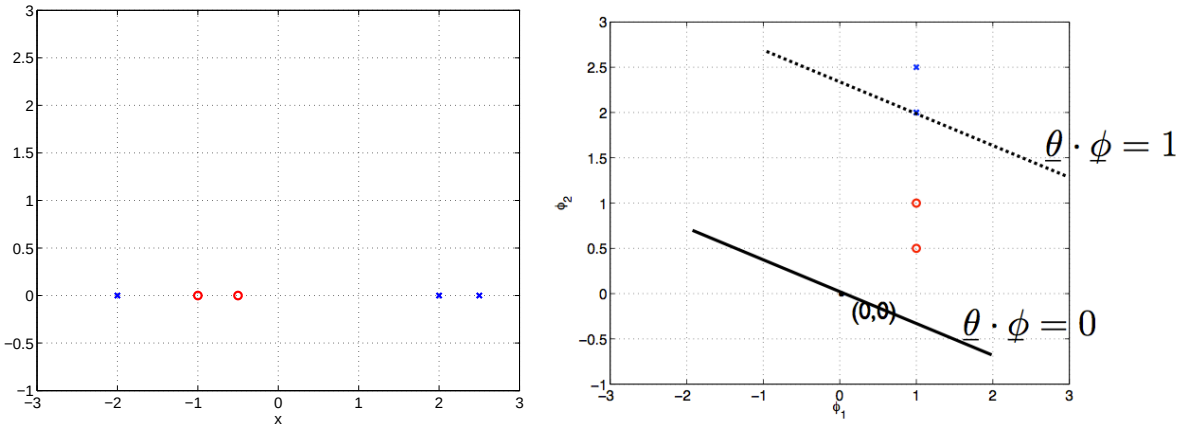


Figure 1: Movie data. Original space (left). Feature space (right).

3. Suppose again that the solution $\hat{\underline{\theta}}$ to Eq.(9.1) exists. Based on this $\hat{\underline{\theta}}$, we predict labels for movies $\underline{x}$ (new and training examples) according to

$$\hat{y} = \begin{cases} 1, & \text{if } (\hat{\underline{\theta}} \cdot \underline{x}) \geqslant \min_{j \in J_+} (\hat{\underline{\theta}} \cdot \underline{x}_j) - \epsilon \\ -1, & \text{otherwise} \end{cases}$$

for some small $\epsilon > 0$. Would this decision rule ensure that all the training movies, positive and negative, are classified correctly? Briefly justify your answer.

Solution: The estimation method only pays attention to the positive examples. We do not know where the negative points lie and therefore could not guarantee which side of the boundary they would fall.

4. The problem might sometimes get a little challenging. Figure 3 (left) shows the movie data, positive (small blue 'x') and negative (red 'o') examples, when movies are represented by real numbers x_j. Briefly describe why we cannot solve Eq.(9.1) in this case.

Solution: We cannot satisfy the constraints for the positive examples. For example, consider two positive examples in the figure, $x = -2$ and $x = +2$. We cannot satisfy $\theta(-2) \geqslant 1$ and $\theta(+2) \geqslant 1$ at the same time.

5. We will apply the algorithm described in Eq.(9.1) with a feature mapping, i.e., we replace one dimensional x with $\underline{\phi}(x) = [1, |x|]^T$. In Figure 3 (right), we have plotted the movie data, positive ('x') and negative ('o'), in the feature coordinates ϕ_1 and ϕ_2. Sketch the solution $\hat{\underline{\theta}}$ in the feature space by drawing $\hat{\underline{\theta}} \cdot \underline{\phi} = 0$ and $\hat{\underline{\theta}} \cdot \underline{\phi} - 1 = 0$ in the figure on the right.

Solution: As shown in the Figure above.

6. Is $\hat{\underline{\theta}} \cdot \underline{\phi}(x) > 0$ at $x = -1$?

Solution: Yes.

10 String theory

We are interested in doing regression, in which the input to our regressor will be strings of arbitrary length, and the output will be a real number. We plan to apply the extension of ordinary least-squares regression to use a kernel function.

Pat claims the following function is a kernel:

$$K(x, z) = \sum_{\beta \in \text{alphabet}} (\text{occurrences of } \beta \text{ in } x)(\text{occurrences of } \beta \text{ in } z)$$

where the alphabet is the set of Roman characters 'a' through 'z'. We will perform a kernelized regression, finding parameters α_i, so that the predictions are of the form:

$$y(x) = \sum_{i=1}^{N} \alpha_i K(x^{(i)}, x).$$

Answer the following questions assuming the training data is:

x	y
"abalone"	10
"xyzygy"	1
"zigzag"	3

(a) What is the feature vector associated with Pat's kernel? Be sure to specify the dimension, d, of the vector.

Solution: It's a vector of integer counts of how frequently each letter occurs in the word.

(b) Determine an expression for $y(\text{"ziggy"})$ in terms of the α_i parameters.

Solution:

$$K = 6\alpha_2 + 7\alpha_3$$

(c) The vector α can be determined by solving a system of equations of the form $A\alpha = b$. Give the numerical values for matrix A and vector b.

Solution:

```
((9, 0, 2),
 (0, 12, 4),
 (2, 4, 10))
```

$$\alpha = K^{-1}(10, 1, 3)^T$$

11 Silly friends

(a) Pat sees your neural network implementation with sigmoidal activation and says there's a much simpler way! Just leave out sigmoids, and let $g(a) = a$. The derivatives are a lot less hassle and it runs faster.

What's wrong with Pat's approach?

Solution: It will degenerate to a linear model.

(b) Chris comes in and says that your network is too complicated, but for a different reason. The sigmoids are confusing and basically the same answer would be computed if we used step functions instead.

What's wrong with Chris's approach?

Solution: The derivative of a step function is zero at any point in its domain except at the non-differentiable point at origin. Accordingly, by chain rule, the derivatives with respect to any parameters inside a step function are zero (at all the differentiable points). Therefore, backpropagation does not work.

(c) Jody sees that you are handling a multi-class problem with 4 classes by using 4 output values, where each target $y^{(i)}$ is actually a length-4 vector with three 0 values and a single 1 value, indicating which class the associated $x^{(i)}$ belongs to.

Jody suggests that you just encode the target $y^{(i)}$ values using integers 1, 2, 3, and 4.

What's wrong with Jody's approach?

Solution: Labels are not equally treated. E.g., it becomes easier to distinguish between label 1 and 4 than between label 1 and 2.

12 Yes vs No

In two-class classification with a standard sigmoid logistic function, the negative log-likelihood error function is the cross entropy:

$$E(w) = -\sum_{n=1}^{N} (y^{(n)} \log h(x^{(n)}, w) + (1 - y^{(n)}) \log(1 - h(x^{(n)}, w)) \quad .$$

(a) Assuming input x and weight w are scalar, and that $N = 1$ (there is a single training example), what is $\partial E(w)/\partial w$?

Solution:

$$\frac{\partial E(w)}{\partial w} = -\frac{\partial}{\partial w} (y \log \sigma(xw) + (1 - y) \log(1 - \sigma(xw))) \tag{12.1}$$

$$= -\frac{\partial}{\partial (xw)} (y \log \sigma(xw) + (1 - y) \log(1 - \sigma(xw))) \frac{\partial (xw)}{\partial w} \tag{12.2}$$

$$= -(y(1 - \sigma(xw)) - (1 - y)\sigma(xw)) x \tag{12.3}$$

$$= (\sigma(wx) - y)x \tag{12.4}$$

(b) What would the neural network weight-update rule be?

Solution:

$$w := w - \eta(\sigma(wx) - y) x$$

13 Grady Ent

Grady Ent decides to train a single sigmoid unit using the following error function:

$$E(w) = 1/2 \sum_i (y(x^i, w) - y^{i*})^2 + 1/2\beta \sum_j w_j^2$$

where $y(x^i, w) = s(x^i \cdot w)$ with $s(z) = 1/(1 + e^{-z})$ being our usual sigmoid function.

1. Write an expression for $\partial E/\partial w_j$. Your answer should be in terms of the training data.

Solution:

$$\begin{aligned} \frac{\partial E}{\partial w_j} &= \frac{\partial}{\partial w_j} \left(\sum_i (y(x^i, w) - y^{i*})^2 \right) + \frac{\partial}{\partial w_j} \left(\frac{1}{2} \beta \sum_j w_j^2 \right) \\ &= \frac{\partial E}{\partial y} \frac{\partial y}{\partial z} \frac{\partial z}{\partial w_j} + \beta w_j \\ &= \sum_i (y - y^{i*})(y)(1 - y)(x_j^i) + \beta w_j \end{aligned}$$

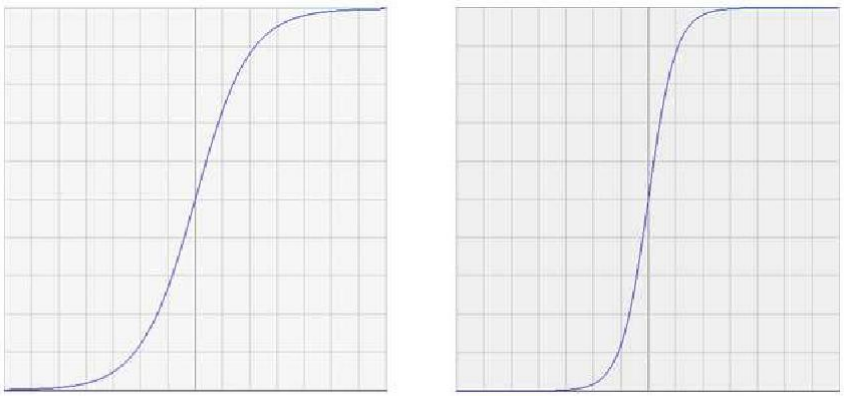
We're using y in the last line here as shorthand for $y(x^i, w)$, that is, the output of the network on input x^i.

2. What update should be made to weight w_j given a single training example $x, y^*)$. Your answer should be in terms of the training data.

Solution:

$$w_j := w_j - \alpha((y - y^*)(y)(1-y)(x_j) + \beta w_j)$$

3. Here are two graphs of the output of the sigmoid unit as a function of a single feature x. The unit has a weight for x and an offet. The two graphs are made using different values of the magnitude of the weight vector ($\|w\|^2 = \sum_j w_j^2$).



A B

Which of the graphs is produced by the larger $\|w\|^2$? Explain.

Solution:

Graph B, because for a given input x, the larger the magnitude of the weight vector, the greater the change in the output of the sigmoid relative to that x.

4. Why might penalizing large $\|w\|^2$, as we could do above by choosing a positive β, be desirable?

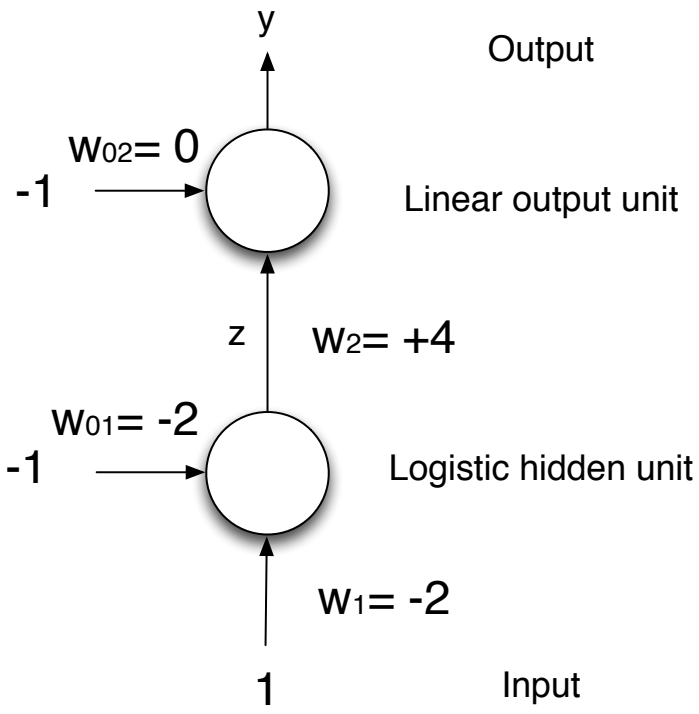
Solution:

Penalizing large weight vectors would be desirable so that we avoid saturation. If the magnitude of the weight vector is too large, gradient descent will not work because the derivative of the sigmoid for large values is nearly 0.

Also, large weights can cause overfitting, since they let you draw boundaries closer to the points.

14 Backpropagation

Here you see a very small neural network: it has one input unit, one hidden unit (logistic), and one output unit (linear).



Let's consider one training case. For that training case, the input value is 1 (as shown in the diagram), and the target output value $t = 1$. We're using the following loss function:

$$E = \frac{1}{2}(t - y)^2$$

Please supply numeric answers; the numbers in this question have been constructed in such a way that you don't need a calculator. Show your work in case of mis-calculation in earlier steps.

(a) What is the output of the hidden unit for this input?

Solution: $1/2$

(b) What is the output of the output unit for this input?

Solution: 2

(c) What is the loss, for this training case?

Solution: $1/2$

(d) What is the derivative of the loss with respect to w_2, for this training case?

Solution: Let z be the output of the hidden unit

$$\begin{aligned}\frac{\partial E}{\partial w_2} &= (1-y)\frac{\partial(-y)}{\partial w_2} \\ &= (1-2)\cdot -z \\ &= (1-2)\cdot -(1/2) \\ &= (1/2)\end{aligned}$$

(e) What is the derivative of the loss with respect to w_1, for this training case?

Solution:

$$\begin{aligned}\frac{\partial E}{\partial w_1} &= \frac{\partial E}{\partial z}\frac{\partial z}{\partial w_1} \\ &= (t-y)\frac{\partial(-y)}{\partial z}\cdot z\cdot(1-z)\cdot x \\ &= (t-y)\cdot -w_2\cdot z\cdot(1-z)\cdot x \\ &= (1-2)\cdot -4\cdot(1/2)\cdot(1/2)\cdot 1 \\ &= 1\end{aligned}$$

(f) With sigmoidal activation, the derivative with respect to w_1 and w_2 are

$$\frac{\partial E}{\partial w_2} = -(t-y)z, \text{ and } \frac{\partial E}{\partial w_1} = -(t-y)\cdot w_2\cdot z\cdot(1-z)\cdot x.$$

Assume that we now use the rectified linear unit (ReLU) as our activation (or a *ramp* function). This means that $z = max(0, w_1x + w_{01})$. What is the derivative of the loss with respect to w_1 and w_2 *at differentiable points* with ReLU? Don't use numerical value for this question.

Solution: It is the same for w_2 as sigmoidal activation case:

$$\frac{\partial E}{\partial w_2} = -(t-y)z$$

For w_1,

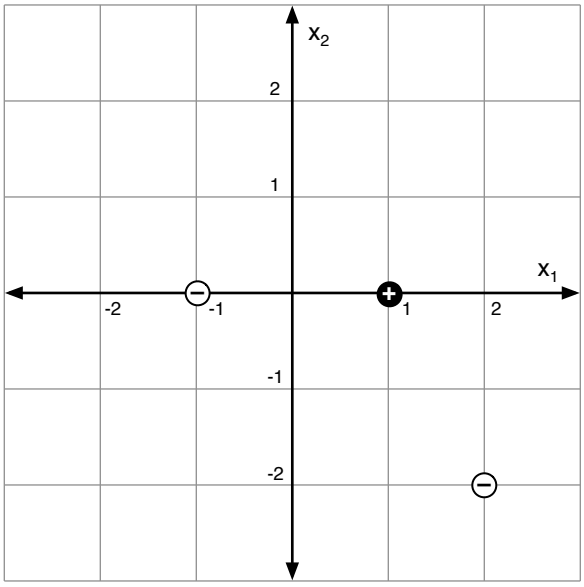
$$\begin{aligned}\frac{\partial E}{\partial w_1} &= \frac{\partial E}{\partial z}\frac{\partial z}{\partial w_1} \\ &= (t-y)\frac{\partial(-y)}{\partial z}\cdot z'\cdot x \\ &= -(t-y)\cdot w_2\cdot z'\cdot x,\end{aligned}$$

where,

$$z' = \begin{cases} 1 & \text{if } z > 0 \\ 0 & \text{if } z < 0. \end{cases}$$

At $z = 0$, it is not differentiable. For such points, we need to consider subdifferential (or subgradient), but it is not required in this question.

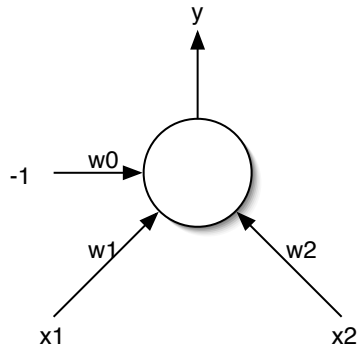
15 Neural Net



Data points are: Negative: (-1, 0) (2, -2) Positive: (1, 0)

Recall that for neural nets with sigmoidal output units, the negative class is represented by a desired output of 0 and the positive class by a desired output of 1. Hint: Some useful values of the sigmoid $s(z)$ are $s(-1) = 0.27$ and $s(1) = 0.73$.

Assume we have a single sigmoid unit:



Assume that the weights are $w_0 = 0, w_1 = 1, w_2 = 1$. What is the computed y value for each of the points on the diagram above?

(a) $x = (-1, 0)$

Solution: $y = \mathbf{s}(\mathbf{0} \cdot -\mathbf{1} + \mathbf{1} \cdot -\mathbf{1} + \mathbf{1} \cdot \mathbf{0}) = \mathbf{s}(-\mathbf{1}) = \mathbf{0.27}$

(b) $x = (2, -2)$

Solution: $y = \mathbf{s}(\mathbf{0} \cdot -\mathbf{1} + \mathbf{1} \cdot -\mathbf{2} + \mathbf{1} \cdot \mathbf{2}) = \mathbf{s}(\mathbf{0}) = \mathbf{0.5}$

(c) $x = (1, 0)$

Solution: $y = \mathbf{s}(\mathbf{0} \cdot -\mathbf{1} + \mathbf{1} \cdot \mathbf{1} + \mathbf{1} \cdot \mathbf{0}) = \mathbf{s}(\mathbf{1}) = \mathbf{0.73}$

(d) What would be the change in w_2 as determined by backpropagation using a step size (η) of 1.0? Assume the squared loss function. Assume that the input is $x = (2, -2)$ and the initial weights are as specified above. Show the formula you are using as well as the numerical result.

1. $\Delta w_2 =$

Solution: Solution:

$$\begin{aligned}
\Delta w_2 &= -\eta \frac{\partial E}{\partial w_2} \\
&= -\eta \frac{\partial E}{\partial y} \frac{\partial y}{\partial z} \frac{\partial z}{\partial w_2} \\
&= -\eta (y - y^i) y (1 - y) x_2 \\
&= (-1)(0.5 + 0)(0.5)(0.5)(-2) \\
&= 0.25
\end{aligned}$$

Derivations:

$$\begin{aligned} E &= \frac{1}{2}(y - y^i)^2 \\ y &= s(z) \\ z &= \sum_{i=0}^{2} w_i x_i \\ \frac{\partial E}{\partial y} &= y - y^i \\ \frac{\partial y}{\partial z} &= y(1-y) \\ \frac{\partial z}{\partial w_2} &= x_2 \end{aligned}$$

16 Probable cause

You have a binary classification problem, but your training examples are only labeled with probabilities, so your data set consists of pairs $(x^{(i)}, p^{(i)})$, where $p^{(i)}$ is the probability that $x^{(i)}$ belongs to class 1.

You want to train a neural network **with a single unit** to predict these probabilities.

(a) What is a good choice for the activation function of your final output unit?

Solution: Sigmoid.

(b) You can think of the training label $p^{(i)}$ as specifying a true probability and the current output of your neural network as specifying an approximate probability $q^{(i)}$. You think a reasonable objective would be to minimize the KL divergence $KL(p \parallel q)$ between the distributions implicitly represented by the predicted and target outputs. So, the empirical risk would be

$$E = \sum_i -(p^{(i)} \log q^{(i)} + (1 - p^{(i)}) \log(1 - q^{(i)}))$$

If $q^{(i)} = f(w \cdot x^{(i)})$ where f is your activation function, what is the SGD (stochastic gradient descent) weight update rule when using the KL objective function above? For simplicity, assume that $x^{(i)}$ and w are both scalars and write f' for the derivative of f.

Solution: Define

$$e = x \frac{(f(wx) - p) f'(wx)}{(1 - f(wx)) f(wx)}$$

Update rule:

$$w := w - \alpha e$$

where α is the step size (or learning rate)