CS229 lecture 3 notes

James Chuang

February 6, 2017

Contents

My notes on Andrew Ng's CS229 lecture 3 notes.

Support Vector Machines

1. Margins: Intuition

Consider logistic regression:

- $p(y = 1 \mid x; \theta)$ is modeled by $h_\theta(x) = g\left(\theta^T x\right)$
  - $h_\theta(x) \geq 0.5$, i.e. $\theta^T x \geq 0 \rightarrow$ predict "1"
  - larger $\theta^T x$, larger $h_\theta(x) = p(y = 1 \mid x; w, b)$, higher "confidence" in prediction of label 1
    - informally, a prediction $y = 1$ is a very confident one if $\theta^T x \gg 0$
    - similarly, a prediction $y = 0$ is a very confident one if $\theta^T x \ll 0$
      - therefore, a good fit to the data would be to find θ such that $\theta^T x^{(i)} \gg 0$ whenever $y^{(i)} = 1$, and $\theta^T x^{(i)} \ll 0$ whenever $y^{(i)} = 0$.
  - geometrically, points far away from the ***separating hyperplane*** can be predicted with higher confidence than points close to the separating hyperplane

2. Notation

Consider a linear classifier for a binary classification problem with labels y and features x:

- $y \in \{-1, 1\}$
- parameters w, b (treat the bias/intercept b separately from the weights w)
- write classifier as:

$$h_{w,b}(x) = g(w^T x + b)$$

$$g(z) = \begin{cases} 1 & \text{if } z \geq 0 \\ -1 & \text{otherwise} \end{cases}$$

Note that from the definition of g, this classifier directly predicts either 1 or -1 without first going through the intermediate step of estimating the probability of y being 1, as in logistic regression.

3. Functional and geometric margins

Functional margins

Given a training example $\left(x^{(i)}, y^{(i)}\right)$, define the **functional margin** $\hat{\gamma}^{(i)}$ of (w, b) w.r.t. the training example:

$$\hat{\gamma}^{(i)} = y^{(i)} \left(w^T x + b\right)$$

Note:

$$\begin{cases} \text{if } y^{(i)} = 1, \text{ then } \hat{\gamma}^{(i)} \gg 0 \quad \text{if} \quad w^T x + b \gg 0 \\ \text{if } y^{(i)} = -1, \text{ then } \hat{\gamma}^{(i)} \gg 0 \quad \text{if} \quad w^T x + b \ll 0 \end{cases}$$

A prediction is correct if $y^{(i)} \left(w^T x + b\right) > 0$. Large functional margin = a confident and correct prediction.

One property of this classifier needs to be addressed: g, and hence $h_{w,b}(x)$ depends on the sign, but *not* on the magnitude of $w^T x + b$. (E.g., $g(w^T x + b) = g(2w^T x + 2b)$). Therefore, without an additional normalization condition, the functional margin can be made arbitrarily large by scaling w and b. We will come back to the normalization condition later.

Given a training set $S = \left\{\left(x^{(i)}, y^{(i)}\right); i = 1, \ldots, m\right\}$, define the ***functional margin*** $\hat{\gamma}$ of (w, b) w.r.t. S to be the smallest of the functional margins of the individual training examples:

$$\hat{\gamma} = \min_{i=1,\ldots,m} \hat{\gamma}^{(i)}$$

The functional margin simply tells you whether a particular point is properly classified or not. In order to be able to maximize the margin, there needs to be a notion of magnitude. Therefore, we introduce the ***geometric margin***, a scaled version of the functional margin that tells you not only if a point if properly classified or not, but also the magnitude of the distance in units of $\|w\|$.

Geometric margins

The vector w is orthogonal to the separating hyperplane defined by $w^T x + b = 0$. To see this, consider two points x_1 and x_2 on the hyperplane (see ESL Ch. 4.5):

$$\begin{aligned} w^T x_1 + b &= w^T x_2 + b = 0 && \text{by def. of the hyperplane} \\ w^T x_1 &= w^T x_2 \\ w^T (x_1 - x_2) &= 0 \\ \therefore \quad w &\perp \left\{x : w^T x + b = 0\right\} \end{aligned}$$

The **geometric margin** $\gamma^{(i)}$ is the distance from a training example $\left(x^{(i)}, y^{(i)}\right)$to the separating hyperplane. The projection of $x^{(i)}$ onto the separating hyperplane is the point $x^{(i)} - \gamma^{(i)} \frac{w}{\|w\|}$ (remember that $\gamma^{(i)} \in \mathbb{R}$ is just a scalar). Since this point is on the decision boundary, it satisfies $w^T x + b = 0$:

$$\begin{aligned} w^T \left(x^{(i)} - \gamma^{(i)} \frac{w}{\|w\|}\right) + b &= 0 \\ w^T x^{(i)} - \gamma^{(i)} \frac{1}{\|w\|} w^T w + b &= 0 && w^T w = \|w\|^2 \\ \gamma^{(i)} \|w\| &= w^T x^{(i)} + b \\ \gamma^{(i)} &= \frac{w^T x^{(i)} + b}{\|w\|} \\ \gamma^{(i)} &= \left(\frac{w}{\|w\|}\right)^T x^{(i)} + \frac{b}{\|w\|} \end{aligned}$$

To account for training examples on the other side of the decision boundary, we define the **geometric margin** of (w, b) w.r.t. a training example $\left(x^{(i)}, y^{(i)}\right)$ to be:

$$\gamma^{(i)} = y^{(i)} \left(\left(\frac{w}{\|w\|} \right)^T x^{(i)} + \frac{b}{\|w\|} \right)$$

The geometric margin with $\|w\| = 1$ is equal to the functional margin.

The geometric margin is invariant to rescaling of the parameters w and b. This means that we can impose an arbitrary scaling constraint on w, e.g. $\|w\| = 1, |w_1| = 5$, or $|w_1 + b| + |w_2| = 2$, and any of these can be satisfied simply by rescaling w and b.

Finally, given a training set $S = \left\{ \left(x^{(i)}, y^{(i)}\right); i = 1, \ldots, m \right\}$, we also define the geometric margin of (w, b) w.r.t. S to be the smallest of the geometric margins on the individual training examples:

$$\gamma = \min_{i=1,\ldots,m} \gamma^{(i)}$$

4. The optimal margin classifier

Given a training set, a natural criterion is to set a decision boundary that maximizes the (geometric) margin, since this reflects a confident set of predictions on the training set.

Assume a training set that is linearly separable, i.e. a separating hyperplane can separate the two classes. Finding the hyperplane that maximizes the geometric margin is an optimization problem:

$$\begin{aligned} \max_{\gamma,w,b} \quad & \gamma \\ \text{s.t.} \quad & y^{(i)} \left(w^T x^{(i)} + b \right) \geq \gamma, \quad i = 1, \ldots, m \\ & \|w\| = 1 \end{aligned}$$

The above optimization problem is non-convex and therefore difficult to solve (due to the $\|w\| = 1$ constraint). We can transform the problem, remembering that γ is the geometric margin and $\hat{\gamma}$ is the functional margin:

$$\begin{aligned} \max_{\hat{\gamma},w,b} \quad & \frac{\hat{\gamma}}{\|w\|} \\ \text{s.t.} \quad & y^{(i)} \left(w^T x^{(i)} + b \right) \geq \hat{\gamma}, \quad i = 1, \ldots, m \end{aligned}$$

This problem is equivalent since the geometric and functional margins are related by $\gamma = \frac{\hat{\gamma}}{\|w\|}$. This is better, since the $\|w\| = 1$ constraint is gone, but the problem is still non-convex. To simplify it further, remember that w and b can be arbitrarily scaled without changing anything, and set the scaling by constraining the geometric margin of w, b w.r.t. the training set to 1:

$$\hat{\gamma} = 1$$

$$\begin{aligned} & \max_{\gamma,w,b} \quad \frac{1}{\|w\|} \\ = & \min_{\gamma,w,b} \quad \frac{1}{2}\|w\|^2 \\ & \text{s.t.} \quad y^{(i)} \left(w^T x^{(i)} + b \right) \geq 1, \quad i = 1, \ldots, m \end{aligned}$$

This optimization problem has a convex quadratic objective and linear constraints. Solving it (using quadratic programming) gives the ***optimal margin classifier***. In order to solve this, we will use the method of Lagrange multipliers generalized to include inequality constraints in addition to equality constraints.

5. Digression: Lagrange duality

Consider the following problem:

$$\begin{aligned} \min_{w} \quad & f(w) \\ \text{s.t.} \quad & h_i(w) = 0, \quad i = 1, \ldots, l \end{aligned}$$

This can be solved with Lagrange multipliers. Define the ***Lagrangian***:

$$\mathcal{L} = f(w) + \sum_{i=1}^{\ell} \beta_i h_i(w)$$

The β_i are the ***Lagrange multipliers***.

To solve the problem, find the partial derivatives of $\mathcal{L}$, set to zero, and solve for w and β:

$$\frac{\partial \mathcal{L}}{\partial w_i} = 0; \frac{\partial \mathcal{L}}{\partial \beta_i} = 0.$$

This can be generalized to constrained optimization problems with inequality as well as equality constraints. Consider the following, called the ***primal*** optimization problem:

$$\begin{aligned} \min_{w} \quad & f(w) \\ \text{s.t.} \quad & g_i(w) \leq 0, \quad i = 1, \ldots, k \\ & h_i(w) = 0, \quad i = 1, \ldots, k \end{aligned}$$

To solve it, start by defining the ***generalized Lagrangian***

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

Here, the α_i's and β_i's are the Lagrange multipliers. Consider the quantity

$$\theta_{\mathcal{P}}(w) = \max_{\alpha, \beta : \alpha_i \geq 0} \mathcal{L}(w, \alpha, \beta),$$

where the "$\mathcal{P}$" subscript stands for "primal". Let some w be given. If w violates any of the primal constraints (i.e., $g_i(w) > 0$ or $h_i(w) \neq 0$ for some i), then:

$$\begin{aligned} \theta_{\mathcal{P}}(w) &= \max_{\alpha, \beta : \alpha_i \geq 0} \left(f(w) + \sum_{i=1}^{k} \alpha_i g_i(w) + \sum_{i=1}^{\ell} \beta_i h_i(w) \right) \\ &= \infty \end{aligned}$$

Conversely, if the constraints are satisfied for a particular w, then $\theta_{\mathcal{P}}(w) = f(w)$. Hence,

$$\theta_{\mathcal{P}}(w) = \begin{cases} f(w) & \text{if } w \text{ satisfies primal constraints} \\ \infty & \text{otherwise} \end{cases}$$

I.e., $\theta_{\mathcal{P}}$ takes the same value as the objective function for all values of w that satisfy the primal constraints, and is positive infinity if the constraints are violated. Hence, the minimization problem

$$\min_w \theta_{\mathcal{P}}(w) = \min_w \max_{\alpha,\beta:\alpha_i \geq 0} \mathcal{L}(w, \alpha, \beta)$$

is the same problem (has the same solutions as) the original, primal problem. We also define the optimal value of the objective to be $p^* = \min_w \theta_{\mathcal{P}}(w)$; we call this the ***value*** of the primal problem.

Now, consider a slightly different problem. Define

$$\theta_{\mathcal{D}}(\alpha, \beta) = \min_w \mathcal{L}(w, \alpha, \beta).$$

Here, the "$\mathcal{D}$" subscript stands for "dual". Note that whereas in the definition of $\theta_{\mathcal{P}}$ we optimized w.r.t. α, β, here we are minimizing w.r.t. w.

Now we define the ***dual*** optimization problem:

$$\max_{\alpha,\beta:\alpha_i \geq 0} \theta_{\mathcal{D}}(\alpha, \beta) = \max_{\alpha,\beta:\alpha_i \geq 0} \min_w \mathcal{L}(w, \alpha, \beta).$$

This is the same as the primal problem, except that the order of the max and min are exchanged. We also define the optimal value of the dual problem's objective to be $d^* = \max_{\alpha,\beta:\alpha \geq 0} \theta_{\mathcal{D}}(w)$.

The primal and dual problems are related in the following way:

$$\begin{aligned} d^* &\leq p^* \\ \max_{\alpha,\beta:\alpha_i \geq 0} \min_w \mathcal{L}(w, \alpha, \beta) &\leq \min_w \max_{\alpha,\beta:\alpha_i \geq 0} \mathcal{L}(w, \alpha, \beta) \qquad \text{because } \max \min f \leq \min \max f \end{aligned}$$

Under certain conditions, $d^* = p^*$, so that the solution can be found by solving the dual problem instead of the primal problem. What are the conditions?

Suppose f and the g_i's are convex (for this purpose, this is when the Hessian is PSD), and the h_i's are affine (i.e. linear, with an intercept term). Suppose further that the constraints g_i are (strictly) feasible, i.e. there exists some w s.t. $g_i(w) < 0 \; \forall \; i$.

Under these assumptions, there must exist w^*, α^*, β^* such that:

- w^* is the solution to the primal problem
- α^*, β^* are the solution to the dual problem
- $p^* = d^* = \mathcal{L}(w^*, \alpha^*, \beta^*)$

If some w^*, α^*, β^* satisfy the **Karush-Kuhn-Tucker (KKT) conditions**, then it is also a solution to the primal and dual problems:

$$\begin{aligned} \frac{\partial}{\partial w_i} \mathcal{L}(w^*, \alpha^*, \beta^*) &= 0, \quad i = 1, \ldots, n \\ \frac{\partial}{\partial \beta_i} \mathcal{L}(w^*, \alpha^*, \beta^*) &= 0, \quad i = 1, \ldots, l \\ \alpha_i^* g_i(w^*) &= 0, \quad i = 1, \ldots, k \qquad \text{the } \textbf{dual complementarity} \text{ condition} \\ g_i(w^*) &\leq 0, \quad i = 1, \ldots, k \\ \alpha_i^* &\geq 0, \quad i = 1, \ldots, k \end{aligned}$$

The **dual complementarity** condition above implies:

- if $\alpha_i^* > 0$, then $g_i(w^*) = 0$
  - i.e, the $g_i(w) \leq 0$ constraint is **active** (holds with equality $g_i(w) = 0$ rather than w/inequality)
  - later, this is important for showing that the SVM has only a small number of support vectors

6. Optimal margin classifiers

The primal optimization problem for finding the optimal margin classifier, derived in section 4:

$$\begin{aligned} \min_{\gamma,w,b} \quad & \frac{1}{2}\|w\|^2 \\ \text{s.t.} \quad & y^{(i)}\left(w^T x^{(i)} + b\right) \geq 1, \quad i = 1, \ldots, m \end{aligned}$$

The constraints can be written as:

$$g_i(w) = -y^{(i)}\left(w^T x^{(i)} + b\right) + 1 \leq 0$$

We have one constraint for each training example. From the KKT dual complementarity condition, $\alpha_i > 0$ only for the training examples that have geometric margin exactly equal to one (i.e., the ones corresponding to constraints that hold with equality $g_i(w) = 0$). These training points are exactly the ones closest to the decision boundary, and are called the ***support vectors*** of the optimal margin classifier. The fact that the number of support vectors can be much smaller than the size of the training set will be useful later.

We recast the optimization problem as the dual form of the problem, because the solution will be in terms of inner products $\left\langle x^{(i)}, x^{(j)} \right\rangle$ between points in the input feature space. This will allow application of the kernel trick later.

The Lagrangian for the optimal margin classifier problem:

$$\mathcal{L}(w, b, \alpha) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{m} \alpha_i \left[y^{(i)}\left(w^T x^{(i)} + b\right) - 1\right]$$

Find the dual form $\theta_{\mathcal{D}} = \min_{w,b} \mathcal{L}(w, b, \alpha)$. Start by taking derivatives w.r.t. w and b and setting to zero:

$$\begin{aligned} \mathcal{L}(w, b, \alpha) &= \frac{1}{2}\|w\|^2 - \sum_{i=1}^{m} \alpha_i \left[y^{(i)}\left(w^T x^{(i)} + b\right) - 1\right] \\ \nabla_w \mathcal{L}(w, b, \alpha) &= w - \sum_{i=1}^{m} \alpha_i y^{(i)} x^{(i)} = 0 \\ w &= \sum_{i=1}^{m} \alpha_i y^{(i)} x^{(i)} \end{aligned}$$

$$\begin{aligned} \mathcal{L}(w, b, \alpha) &= \frac{1}{2}\|w\|^2 - \sum_{i=1}^{m} \alpha_i \left[y^{(i)}\left(w^T x^{(i)} + b\right) - 1\right] \\ \frac{\partial}{\partial b} \mathcal{L}(w, b, \alpha) &= -\sum_{i=1}^{m} \alpha_i y^{(i)} = 0 \\ \sum_{i=1}^{m} \alpha_i y^{(i)} &= 0 \end{aligned}$$

$$\begin{aligned} \mathcal{L}(w, b, \alpha) &= \frac{1}{2}\|w\|^2 - \sum_{i=1}^{m} \alpha_i \left[y^{(i)}\left(w^T x^{(i)} + b\right) - 1\right] && \text{plug in } w \\ &= \frac{1}{2} \sum_{i,j=1}^{m} y^{(i)} y^{(j)} \alpha_i \alpha_j \left(x^{(i)}\right)^T x^{(j)} - \sum_{i,j=1}^{m} y^{(i)} y^{(j)} \alpha_i \alpha_j \left(x^{(i)}\right)^T x^{(j)} - b \sum_{i=1}^{m} \alpha_i y^{(i)} + \sum_{i=1}^{m} \alpha_i \quad \sum_{i=1}^{m} \alpha_i y^{(i)} = 0 \\ &= \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} y^{(i)} y^{(j)} \alpha_i \alpha_j \left(x^{(i)}\right)^T x^{(j)} \end{aligned}$$

The dual optimization problem:

$$\begin{aligned}\max_{\alpha} W(\alpha) &= \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} y^{(i)} y^{(j)} \alpha_i \alpha_j \left(x^{(i)}\right)^T x^{(j)} \\ \text{s.t. } \alpha_i &\geq 0, \quad i = 1, \ldots, m \\ \sum_{i=1}^{m} \alpha_i y^{(i)} &= 0\end{aligned}$$

The conditions for $p^* = d^*$ and the KKT conditions to hold are satisfied in the above problem. If we solve this maximization problem w.r.t. the α_i's for the optimal α_i's, we can solve for the w by the above equation ($w = \sum_{i=1}^{m} \alpha_i y^{(i)} x^{(i)}$).

Then b^* can be solved for:

$$b^* = -\frac{\max_{i:y^{(i)}=-1} w^{*T} x^{(i)} + \min_{i:y^{(i)}=1} w^{*T} x^{(i)}}{2}$$

Suppose the model has been fit to a training set. To make a prediction at a new input point x, calculate $w^T x + b$, and predict $y = 1$ iff this quantity is bigger than zero.

$$\begin{aligned}w^T x + b &= \left(\sum_{i=1}^{m} \alpha_i y^{(i)} x^{(i)}\right)^T x + b \\ &= \sum_{i=1}^{m} \alpha_i y^{(i)} \langle x^{(i)}, x \rangle + b\end{aligned}$$

Hence, if we've found the α_i's, in order to make a prediction, we need to calculate a quantity that only depends on the inner product between x and the points in the training set. Also, the α_i's will all be zero except for the support vectors. Thus, many of the terms in the sum will be zero, meaning that we only need to find the inner products between x and the support vectors in order to make a prediction.

By using different kernels, the optimal margin classifier generalizes to the ***support vector machine***, which can learn efficiently in very high dimensional spaces.

7. Kernels

- def: original input values of problem = ***attributes*** (e.g. x, the living area of a house)
- def: the input values that are passed to the learning algorithm = ***features*** (e.g. $\begin{bmatrix} x & x^2 & x^3 \end{bmatrix}^T$)
- def: the ***feature mapping*** ϕ, a map from the attributes to the features, e.g.:

$$\phi(x) = \begin{bmatrix} x \\ x^2 \\ x^3 \end{bmatrix}$$

Rather than applying an SVM using the original input attributes x, we may instead want to learn using some features $\phi(x)$. Since the SVM algorithm can be written entirely in terms of the inner products $\langle x, z \rangle$, this entails replacing all those inner products with $\langle \phi(x), \phi(z) \rangle$. Specifically, given a feature mapping ϕ, we define the corresponding ***kernel*** to be

$$K(x, z) = \phi(x)^T \phi(z)$$

Then, everywhere where the algorithm had $\langle x, z \rangle$, we replace it with $K(x, z)$ and the algorithm now learns with the features ϕ.

Often, $K(x, z)$ is inexpensive to calculate, even though $\phi(x)$ itself may be very expensive (or impossible) to calculate (perhaps because it is an extremely high dimensional vector). In such settings, by using an efficient way to calculate $K(x, z)$ in the algorithm, an SVM can learn in the high dimensional feature space given by ϕ, but without ever having to explicitly find or represent vectors $\phi(x)$.

An example: Suppose $x, z \in \mathbb{R}^n$, and consider

$$K(x,z) = (x^T z)^2$$

This can also be written as

$$\begin{aligned} K(x,z) &= \left(\sum_{i=1}^{n} x_i z_i\right)\left(\sum_{i=1}^{n} x_i z_i\right) \\ &= \sum_{i=1}^{n}\sum_{j=1}^{n} x_i x_j z_i z_j \\ &= \sum_{i,j=1}^{n} (x_i x_j)(z_i z_j) \\ &= \phi(x)^T \phi(z), \qquad \text{where} \end{aligned}$$

$$\phi(x) = \begin{bmatrix} x_1 x_1 \\ x_1 x_2 \\ x_1 x_3 \\ x_2 x_1 \\ x_2 x_2 \\ x_2 x_3 \\ x_3 x_1 \\ x_3 x_2 \\ x_3 x_3 \end{bmatrix} \text{ (for } n = 3\text{)}$$

Calculating the high-dimensional $\phi(x)$ requires $O(n^2)$ time, while calculating $K(x,z)$ requires only $O(n)$ time – linear in the dimension of the input attributes.

Consider a related kernel,

$$\begin{aligned} K(x,z) &= \left(x^T z + c\right)^2 \\ &= \left(x^T z\right)^2 + 2c\left(x^T z\right) + c^2 \\ &= \sum_{i,j=1}^{n} (x_i x_j)(z_i z_j) + \sum_{i=1}^{n} \left(\sqrt{2c} x_i\right)\left(\sqrt{2c} z_i\right) + c^2 \end{aligned}$$

$$\phi(x) = \begin{bmatrix} x_1 x_1 \\ x_1 x_2 \\ x_1 x_3 \\ x_2 x_1 \\ x_2 x_2 \\ x_2 x_3 \\ x_3 x_1 \\ x_3 x_2 \\ x_3 x_3 \\ \sqrt{2c} x_1 \\ \sqrt{2c} x_2 \\ \sqrt{2c} x_3 \\ c \end{bmatrix}$$

, where the parameter c controls the relative weighting between the x_i (first order) and the $x_i x_j$ (second order) terms.

More broadly, the kernel $K(x,z) = \left(x^T z + c\right)^d$ corresponds to a feature mapping to an $\binom{n+d}{d}$ feature space, corresponding of all monomials of the form $x_{i_1} x_{i_2} \ldots x_{i_k}$ that are up to order d. Despite working in $O(n^d)$-dimensional space, computing $K(x,z)$ still takes only $O(n)$ time, and we never need to explicitly represent feature vectors in the very high dimensional feature space.

A slightly different view of kernels. Like the dot product, the kernel $K(x,z) = \phi(x)^T\phi(z)$ can be thought of as a similarity measure, i.e. if $\phi(x)$ and $\phi(z)$ are close together, we expect $K(x,z)$ to be large, and if $\phi(x)$ and $\phi(z)$ are far apart, we expect $K(x,z)$ to be small. Therefore, given a learning problem a good kernel to use would be a function represents the similarity of examples. For example, the function

$$K(x,z) = \exp\left(-\frac{\|x-z\|^2}{2\sigma^2}\right)$$

might be a reasonable measure of the similarity of x and z, as it is near 1 when x and z are close, and near 0 when x and z are far apart. However, is this a valid function to be used as a kernel? (In this case, yes, as this is the ***Gaussian kernel*** corresponding to an infinite-dimensional feature mapping ϕ). But more broadly, given a function K, how can we tell if it's a valid kernel; i.e. is there a feature mapping ϕ such that $K(x,z) = \phi(x)^T\phi(z)\,\forall x, z$?

- Suppose that K is a valid kernel corresponding to some feature mapping ϕ
  - consider some finite set of m points (not necessarily the training set) $\left\{x^{(1)}, \ldots, x^{(m)}\right\}$
  - define the ***kernel matrix***: a square, m-by-m matrix K s.t.

$$K_{ij} = K\left(x^{(i)}, x^{(j)}\right)$$

    - note that the notation K is overloaded to denote both the kernel function $K(x,z)$ and the kernel matrix K
  - if K is a valid kernel, then

$$\begin{aligned}
&K_{ij}\\
&= K\left(x^{(i)}, x^{(j)}\right)\\
&= \phi\left(x^{(i)}\right)^T \phi\left(x^{(j)}\right)\\
&= \phi\left(x^{(j)}\right)^T \phi\left(x^{(i)}\right)\\
&= K\left(x^{(j)}, x^{(i)}\right)\\
&= K_{ji}
\end{aligned}$$

  - therefore, K for a valid kernel is symmetric
  - in addition, let $\phi_k(x)$ denote the k-th coordinate of the vector $\phi(x)$. Then, for any vector z:

$$\begin{aligned}
z^T K z &= \sum_i \sum_j z_i K_{ij} z_j\\
&= \sum_i \sum_j z_i \phi\left(x^{(i)}\right)^T \phi\left(x^{(j)}\right) z_j \qquad \text{def. } K\\
&= \sum_i \sum_j z_i \sum_k \phi_k\left(x^{(i)}\right) \phi_k\left(x^{(j)}\right) z_j\\
&= \sum_k \sum_i \sum_j z_i \phi_k\left(x^{(i)}\right) \phi_k\left(x^{(j)}\right) z_j\\
&= \sum_k \left(\sum_i z_i \phi_k\left(x^{(i)}\right)\right)^2\\
&\geq 0
\end{aligned}$$

  - therefore, if K is a valid kernel (i.e., it corresponds to some feature mapping ϕ), then the corresponding Kernel matrix $K \in \mathbb{R}^{m\times m}$ is symmetric positive semidefinite
    - this is a necessary and sufficient condition for K to be a valid kernel (aka a Mercer kernel)
    - Theorem (Mercer). Let $K : \mathbb{R}^n \times \mathbb{R}^n \mapsto \mathbb{R}$ be given. Then for K to be a valid (Mercer) kernel, it is necessary and sufficient that for any $\left\{x^{(1)}, \ldots, x^{(m)}\right\}, (m < \infty)$, the corresponding kernel matrix is symmetric and positive semi-definite

- examples of kernels
  - on the MNIST digit recognition dataset, SVMs with a simple polynomial kernel or the Gaussian kernel performed extremely well
    - this was surprising, since the input attributes were a 256-dimensional vector of pixel intensity without knowledge of which pixels are adjacent to which other ones
  - if objects to be classified are strings (e.g. sequence of amino acids), it is hard to construct a reasonable, "small" set of features, especially if different strings have different lengths
    - consider letting $\phi(x)$ be a feature vector that counts the number of occurrences of each length-k substring in x
      - if considering English letters, there are 26^k such strings $\rightarrow$ $\phi(x)$ is 26^k dimensional vector (to big to work with efficiently)
      - however, using (dynamic programming-ish) string matching algorithms, $K(x,z) = \phi(x)^T\phi(z)$ can be efficiently computed
      - therefore, can implicitly work in 26^k-dimensional feature space, without ever explicitly computing feature vectors in this space
- the kernel trick is applicable beyond SVMs
  - can be applied to any learning algorithm that can be written in terms of only inner products $\langle x, z\rangle$ between input attribute vectors by replacing the inner product with a kernel $K(x,z)$

8. Regularization and the non-separable case

- the derivation of the SVM so far assumes the data is linearly separable
  - mapping data to a high dimensional feature space via ϕ increases the likelihood that the data is separable, but does not guarantee it
  - also, in some cases, finding a separating hyperplane might not be ideal, since it might be susceptible to outliers
    - therefore, reformulate the problem using ℓ_1 (lasso) regularization:

$$\begin{aligned}\min_{\gamma,w,b} \quad & \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{m}\xi_i \\ \text{s.t.} \quad & y^{(i)}\left(w^T x^{(i)} + b\right) \geq 1 - \xi_i, \quad i = 1,\ldots,m \\ & \xi_i \geq 0, \quad i = 1,\ldots,m\end{aligned}$$

    - examples are now permitted to have (functional) margin less than 1:
      - if an example has functional margin $1-\xi_i \quad (\xi > 0)$, we pay a cost of the objective function being increased by $C\xi_i$
        - C controls the weighting between the goals of minimizing $\|w\|^2$ (which makes the margin large) and of ensuring that most examples have functional margin at least 1
    - the Lagrangian:

$$\mathcal{L}(w,b,\xi,\alpha,r) = \frac{1}{2}w^T w + C\sum_{i=1}^{m}\xi_i - \sum_{i=1}^{m}\alpha_i\left[y^{(i)}\left(x^T w + b\right) - 1 + \xi_i\right] - \sum_{i=1}^{m} r_i\xi_i$$

    - to get the dual form:
      - set derivatives w.r.t w and b to zero
      - substitute back in and simplify

$$\begin{aligned}\max_{\alpha} \quad & W(\alpha) = \sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{m} y^{(i)}y^{(j)}\alpha_i\alpha_j\left\langle x^{(i)}x^{(j)}\right\rangle \\ \text{s.t.} \quad & 0 \leq \alpha_i \leq C, \quad i = 1,\ldots,m \\ & \sum_{i=1}^{m}\alpha_i y^{(i)} = 0\end{aligned}$$

    - w in terms of α_i's is the same as the non-regularized case:

$$w = \sum_{i=1}^{m} \alpha_i y^{(i)} x^{(i)},$$

so

$$w^T x + b = \sum_{i=1}^{m} \alpha_i y^{(i)} \left\langle x^{(i)}, x \right\rangle + b$$

can still be used to make predictions once the dual problem is solved. - note that adding ℓ_1 regularization only has the effect of changing $0 \leq \alpha_i$ to $0 \leq \alpha_i \leq C$ - the calculation for b^* also has to be modified

- the KKT dual-complementarity conditions:

$$\begin{aligned}
\alpha_i = 0 &\implies y^{(i)} \left(w^T x^{(i)} + b \right) \geq 1 \\
\alpha_i = C &\implies y^{(i)} \left(w^T x^{(i)} + b \right) \leq 1 \\
0 < \alpha_i < C &\implies y^{(i)} \left(w^T x^{(i)} + b \right) = 1
\end{aligned}$$

To solve the dual problem, we use the ***sequential minimal optimization (SMO) algorithm***.

9. The SMO algorithm

- SMO, by John Platt, is an efficient way of solving the dual problem in the derivation of the SVM
- first, a digression about coordinate ascent:

9.1 Coordinate ascent

- consider the unconstrained optimization problem

$$\max_{\alpha} W(\alpha_1, \alpha_2, \ldots, \alpha_m)$$

- gradient ascent and Newton's method could both solve this
- now we consider ***coordinate ascent***:
  - Loop until convergence:
    - For $i = 1, \ldots, m$,
      - $\alpha_i \leftarrow \arg\max_{\hat{\alpha}_i} W\left(\alpha_1, \ldots, \alpha_{i=1}, \hat{\alpha}_i, \alpha_{i+1}, \ldots, \alpha_m\right)$
  - in the innermost loop, W is reoptimized w.r.t *just* the parameter α_i, holding all other variables fixed
  - when W is of a form s.t. the $\arg\max$ can be found efficiently, coordinate ascent can be fairly efficient

9.2 SMO

- the dual optimization problem for SVM with ℓ_1 regularization:

$$\begin{aligned}
\max_{\alpha} \quad & W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} y^{(i)} y^{(j)} \alpha_i \alpha_j \left\langle x^{(i)} x^{(j)} \right\rangle \\
\text{s.t.} \quad & 0 \leq \alpha_i \leq C, \quad i = 1, \ldots, m \\
& \sum_{i=1}^{m} \alpha_i y^{(i)} = 0
\end{aligned}$$

- suppose we have a set of α_i's that satisfy the above constraints
  - then suppose we want to take a coordinate ascent step optimizing w.r.t α_1 holding $\alpha_2, \ldots, \alpha_m$ fixed. Progress is not possible, because:

$$\alpha_1 y^{(1)} = -\sum_{i=2}^{m} \alpha_i y^{(i)}$$

$$\alpha_1 = -y^{(1)} \sum_{i=2}^{m} \alpha_i y^{(i)} \quad \text{remember } y^{(1)} \in \{-1, 1\}$$

- α_1 is uniquely determined by the other α_i's- thus, to update the α_i's, we must update at least two of them simultaneously to keep satisfying the constraints, motivating the SMO algorithm:
  - repeat until convergence:
    1. Select some pair α_i and α_j to update next (using some heuristic that tries to pick the two that will allow the biggest progress towards the global max)
    2. Reoptimize $W(\alpha)$ w.r.t. α_i and α_j, holding the other α_k's $(k \neq i, j)$ fixed
- suppose we currently have α_i's that satisfy the constraints
  - suppose we decide to hold $\alpha_3, \ldots, \alpha_m$ fixed, and reoptimize $W(\alpha_1, \alpha_2, \ldots, \alpha_m)$ w.r.t. α_1 and α_2

$$\alpha_1 y^{(1)} + \alpha_2 y^{(2)} = -\sum_{i=3}^{m} \alpha_i y^{(i)} \quad \text{RHS is constant, set to } \zeta$$

$$\alpha_1 y^{(1)} + \alpha_2 y^{(2)} = \zeta$$

  - See the figure in the notes:
    - α_1 and α_2 must lie within the box $[0, C] \times [0, C]$
      - in order to lie within the box and satisfy $\alpha_1 y^{(1)} + \alpha_2 y^{(2)} = \zeta, \quad L \leq \alpha_2 \leq H$, where L is a lower bound
  - get α_1 as a function of α_2:

$$\begin{aligned} \alpha_1 y^{(1)} + \alpha_2 y^{(2)} &= \zeta \\ \alpha_1 y^{(1)} &= \zeta - \alpha_2 y^{(2)} \qquad\qquad y^{(1)} \in \{-1, 1\} \\ \alpha_1 &= \left(\zeta - \alpha_2 y^{(2)}\right) y^{(1)} \end{aligned}$$

  - substitute back into $W(\alpha)$:

$$W(\alpha_1, \alpha_2, \ldots, \alpha_m) = W\left(\left(\zeta - \alpha_2 y^{(2)}\right) y^{(1)}, \alpha_2, \ldots, \alpha_m\right)$$

  - this is a quadratic function in α_2, i.e. it can be expressed $a\alpha_2^2 + b\alpha_2 + c$
    - ignoring the "box" constraints, this function can easily be solving by setting the derivative to zero and solving
    - let $\alpha_2^{\text{new, unclipped}}$ denote the resulting value of α_2
      - we can find the optimal α_2 by "clipping" it to the $[L, H]$ interval specified by the box constraint:

$$\alpha_2^{\text{new}} = \begin{cases} H & \text{if } \alpha_2^{\text{new, unclipped}} > H \\ \alpha_2^{\text{new, unclipped}} & \text{if } L \leq \alpha_2^{\text{new, unclipped}} \leq H \\ L & \text{if } \alpha_2^{\text{new, unclipped}} < L \end{cases}$$

    - having solved for α_2^{new}, solve for α_1^{new}
      - $\alpha_1 = \left(\zeta - \alpha_2 y^{(2)}\right) y^{(1)}$
  - details left unresolved here:
    - heuristic for choosing next α_i, α_j to update
    - how to update b