What is a good Decision Boundary?

- Consider a binary classification task with \( y = \pm 1 \) labels (not 0/1 as before).
- When the training examples are linearly separable, we can set the parameters of a linear classifier so that all the training examples are classified correctly.
- Many decision boundaries!
  - Generative classifiers
  - Logistic regressions ...
- Are all decision boundaries equally good?
What is a good Decision Boundary?

Not All Decision Boundaries Are Equal!

- Why we may have such boundaries?
  - Irregular distribution
  - Imbalanced training sizes
  - Outliners
Classification and Margin

- Parameterizing decision boundary
  - Let $\mathbf{w}$ denote a vector orthogonal to the decision boundary, and $b$ denote a scalar "offset" term, then we can write the decision boundary as:

$$w^T x + b = 0$$

Class 1

Class 2

Margin

$$\frac{(\mathbf{w}^T \mathbf{x}_i + b)}{||\mathbf{w}||} > \frac{c}{||\mathbf{w}||}$$

for all $\mathbf{x}_i$ in class 2

$$\frac{(\mathbf{w}^T \mathbf{x}_i + b)}{||\mathbf{w}||} < \frac{-c}{||\mathbf{w}||}$$

for all $\mathbf{x}_i$ in class 1

Or more compactly:

$$\frac{(\mathbf{w}^T \mathbf{x}_i + b)}{||\mathbf{w}||} > \frac{c}{||\mathbf{w}||}$$

The margin between two points

$$m = d^+ + d^- = \frac{y_i (\mathbf{x}_i, -\mathbf{x}_j)}{||\mathbf{w}||}$$
Maximum Margin Classification

- The margin is:
  \[ m = \frac{w^T (x_i - x_j)}{||w||} = \frac{2c}{||w||} \]

- Here is our Maximum Margin Classification problem:
  \[
  \max_w \quad \frac{2c}{||w||} \\
  \text{s.t.} \quad y_i (w^T x_i + b) \geq \frac{c}{||w||}, \quad \forall i
  \]

Maximum Margin Classification, con'd.

- The optimization problem:
  \[
  \max_{w,b} \quad \left\{ \frac{c}{||w||} \right\} \\
  \text{s.t.} \quad y_i (w^T x_i + b) \geq \frac{c}{||w||}, \quad \forall i
  \]

- But note that the magnitude of \( c \) merely scales \( w \) and \( b \), and does not change the classification boundary at all! (why?)

- So we instead work on this cleaner problem:
  \[
  \max_{w,b} \quad \frac{1}{||w||} \\
  \text{s.t.} \quad y_i (w^T x_i + b) \geq 1, \quad \forall i
  \]

- The solution to this leads to the famous Support Vector Machines -- believed by many to be the best "off-the-shelf" supervised learning algorithm
Support vector machine

- A convex quadratic programming problem with linear constrains:

\[
\max_{w,b} \frac{1}{2} \|w\|^2 \\
\text{s.t.} \quad y_i(w^T x_i + b) \geq 1, \quad \forall i
\]

- The attained margin is now given by

\[
\frac{1}{\|w\|}
\]

- Only a few of the classification constraints are relevant \(\Rightarrow\) support vectors

Constrained optimization

- We can directly solve this using commercial quadratic programming (QP) code
- But we want to take a more careful investigation of Lagrange duality, and the solution of the above in its dual form.
  \(\Rightarrow\) deeper insight: support vectors, kernels ...
  \(\Rightarrow\) more efficient algorithm

Digression to Lagrangian Duality

- The Primal Problem

\[
\min_w f(w) \leftarrow \\
\text{s.t.} \quad g_i(w) \leq 0, \quad i = 1, \ldots, k \\
\quad h_i(w) = 0, \quad i = 1, \ldots, l
\]

The generalized Lagrangian:

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

the \(\alpha\)'s \((\alpha \geq 0)\) and \(\beta\)s are called the Lagrangian multipliers

Lemma:

\[
\max_{w,\beta \geq 0} \mathcal{L}(w, \alpha, \beta) = \begin{cases} 
  f(w) & \text{if } w \text{ satisfies primal constraints} \\
  \infty & \text{o/w}
\end{cases}
\]

A re-written Primal:

\[
\min_w \max_{\alpha, \beta \geq 0} \mathcal{L}(w, \alpha, \beta)
\]
Lagrangian Duality, cont.

- Recall the Primal Problem:
  \[
  \min_w \max_{\alpha, \beta, \alpha \geq 0} \mathcal{L}(w, \alpha, \beta)
  \]

- The Dual Problem:
  \[
  \max_{\alpha, \beta, \alpha \geq 0} \min_w \mathcal{L}(w, \alpha, \beta)
  \]

- Theorem (weak duality):
  \[
  d^* = \max_{\alpha, \beta, \alpha \geq 0} \min_w \mathcal{L}(w, \alpha, \beta) \leq \min_w \max_{\alpha, \beta, \alpha \geq 0} \mathcal{L}(w, \alpha, \beta) = p^*
  \]

- Theorem (strong duality):
  If there exist a saddle point of \( \mathcal{L}(w, \alpha, \beta) \), we have
  \[
  d^* = p^*
  \]

A sketch of strong and weak duality

- Now, ignoring \( h(x) \) for simplicity, let's look at what's happening graphically in the duality theorems.
  \[
  d^* = \max_{\alpha \geq 0} \min_w f(w) + \alpha^T g(w) \leq \min_w \max_{\alpha \geq 0} f(w) + \alpha^T g(w) = p^*
  \]
A sketch of strong and weak duality

- Now, ignoring $h(x)$ for simplicity, let's look at what's happening graphically in the duality theorems.

\[
\begin{align*}
    d^* &= \max_{\alpha, \geq 0} \min_w f(w) + \alpha^T g(w) \\
    &= \min_w \max_{\alpha, \geq 0} f(w) + \alpha^T g(w) = p^*
\end{align*}
\]

The KKT conditions

- If there exists some saddle point of $\mathcal{L}$, then the saddle point satisfies the following "Karush-Kuhn-Tucker" (KKT) conditions:

\[
\begin{align*}
    \frac{\partial}{\partial w_i} \mathcal{L}(w, \alpha, \beta) &= 0, \quad i = 1, \ldots, k \\
    \frac{\partial}{\partial \beta_i} \mathcal{L}(w, \alpha, \beta) &= 0, \quad i = 1, \ldots, l \\
    a_i g_j(w) &= 0, \quad i = 1, \ldots, m \\
    g_j(w) &\leq 0, \quad i = 1, \ldots, m \\
    \alpha_i &\geq 0, \quad i = 1, \ldots, m
\end{align*}
\]

- **Theorem:** If $w^*, \alpha^*$ and $\beta^*$ satisfy the KKT condition, then it is also a solution to the primal and the dual problems.
Solving optimal margin classifier

- Recall our opt problem:
  \[
  \begin{align*}
  & \max_{w,b} \frac{1}{2} \|w\|^2 \\
  & \text{s.t.} \quad y_i (w^T x_i + b) \geq 1, \quad \forall i
  \end{align*}
  \]

- This is equivalent to
  \[
  \begin{align*}
  & \min_{w,b} \frac{1}{2} w^T w \\
  & \text{s.t.} \quad 1 - y_i (w^T x_i + b) \leq 0, \quad \forall i
  \end{align*}
  \]

- Write the Lagrangian:
  \[
  \mathcal{L}(w, b, \alpha) = \frac{1}{2} w^T w - \sum_{i=1}^m \alpha_i \left[ y_i (w^T x_i + b) - 1 \right]
  \]

  Recall that (*) can be reformulated as \( \min_{w,b} \max_{\alpha \geq 0} \mathcal{L}(w, b, \alpha) \)

  Now we solve its dual problem: \( \max_{\alpha \geq 0} \min_{w,b} \mathcal{L}(w, b, \alpha) \)

The Dual Problem

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

- We minimize \( \mathcal{L} \) with respect to \( w \) and \( b \) first:
  \[
  \begin{align*}
  \nabla_w \mathcal{L}(w, b, \alpha) &= w - \sum_{i=1}^m \alpha_i y_i x_i = 0, \quad (*) \\
  \nabla_b \mathcal{L}(w, b, \alpha) &= \sum_{i=1}^m \alpha_i y_i = 0, \quad (**) \\
  \end{align*}
  \]

  Note that (*) implies:
  \[
  w = \sum_{i=1}^m \alpha_i y_i x_i \quad (***)
  \]

- Plug (***) back to \( \mathcal{L} \), and using (**), we have:
  \[
  \mathcal{L}(w, b, \alpha) = \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i,j=1}^m \alpha_i \alpha_j y_i y_j (x_i^T x_j)
  \]
The Dual Problem, cont.

- Now we have the following dual opt problem:

\[
\max_{\alpha} J(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j (x_i^T x_j)
\]

s.t.  \( \alpha_i \geq 0, \quad i = 1, \ldots, k \)

\[\sum_{i=1}^{m} \alpha_i y_i = 0.\]

- This is, (again,) a quadratic programming problem.
  - A global maximum of \( \alpha_i \) can always be found.
  - But what’s the big deal??
  - Note two things:
    1. \( w \) can be recovered by \( w = \sum_{i=2}^{m} \alpha_i y_i x_i \). See next ...
    2. The "kernel" \( x_i^T x_j \). More later ...

I. Support vectors

- Note the KKT condition --- only a few \( \alpha_i \)'s can be nonzero!!

\[\alpha_i g_i(w) = 0, \quad i = 1, \ldots, m\]

Call the training data points whose \( \alpha_i \)'s are nonzero the support vectors (SV)
Support vector machines

- Once we have the Lagrange multipliers \( \{\alpha_i\} \), we can reconstruct the parameter vector \( \mathbf{w} \) as a weighted combination of the training examples:

\[
\mathbf{w} = \sum_{i \in SV} \alpha_i y_i \mathbf{x}_i
\]

- For testing with a new data \( \mathbf{z} \):
  - Compute

\[
\mathbf{w}^T \mathbf{z} + b = \sum_{i \in SV} \alpha_i y_i (\mathbf{x}_i^T \mathbf{z}) + b \geq 0
\]

and classify \( \mathbf{z} \) as class 1 if the sum is positive, and class 2 otherwise
- Note: \( \mathbf{w} \) need not be formed explicitly

Interpretation of support vector machines

- The optimal \( \mathbf{w} \) is a linear combination of a small number of data points. This “sparse” representation can be viewed as data compression as in the construction of kNN classifier

- To compute the weights \( \{\alpha_i\} \), and to use support vector machines we need to specify only the inner products (or kernel) between the examples \( \mathbf{x}_i^T \mathbf{x}_j \)

- We make decisions by comparing each new example \( \mathbf{z} \) with only the support vectors:

\[
y^* = \text{sign} \left( \sum_{i \in SV} \alpha_i y_i (\mathbf{x}_i^T \mathbf{z}) + b \right)
\]
Non-linearly Separable Problems

- We allow “error” $\xi_i$ in classification; it is based on the output of the discriminant function $w^T x + b$.
- $\xi_i$ approximates the number of misclassified samples.

Soft Margin Hyperplane

- Now we have a slightly different opt problem:

$$\min_{w,b} \frac{1}{2} w^T w + C \sum_{i=1}^{m} \xi_i \quad \text{slack first} $$

$$\text{s.t.} \quad y_i (w^T x_i + b) \geq 1 - \xi_i, \quad \forall i$$

$$\xi_i \geq 0, \quad \forall i$$

- $\xi_i$ are “slack variables” in optimization.
- Note that $\xi_i = 0$ if there is no error for $x_i$.
- $\xi_i$ is an upper bound of the number of errors.
- $C$ : tradeoff parameter between error and margin.
Hinge Loss

- Remember Ridge regression
  - \( \min \left( \sum \frac{1}{1+e^{w^T x_i}} + \lambda ||w|| \right) \)

- How about SVM?
  \[
  \arg\min_{w, b} w^T w + \lambda \sum_{i=1}^{m} \max(1 - y_i(w^T x_i + b), 0)
  \]
  regularization  
  Loss: hinge loss

\[
\min_{w, b} ||w||
\]
\[\text{s.t. } y_i(w^T x_i + b) \geq 1, \ \forall i\]
II. The Kernel Trick

- Is this data linearly-separable?

- How about a quadratic mapping $\phi(x_i)$?

Recall the SVM optimization problem

\[ \max_{\alpha} \quad J(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j (x_i^T x_j) \]

subject to

\[ 0 \leq \alpha_i \leq C, \quad i = 1, \ldots, m \]

\[ \sum_{i=1}^{m} \alpha_i y_i = 0. \]

The data points only appear as inner product.

As long as we can calculate the inner product in the feature space, we do not need the mapping explicitly.

Many common geometric operations (angles, distances) can be expressed by inner products.

Define the kernel function $K$ by

\[ K(x_i, x_j) = \phi(x_i)^T \phi(x_j) \]
II. The Kernel Trick

- Computation depends on feature space
  - Bad if its dimension is much larger than input space

\[
\max_{\alpha} \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j K(x_i, x_j)
\]

s.t. \( \alpha_i \geq 0, \quad i = 1, \ldots, k \)

\[
\sum_{i=1}^{m} \alpha_i y_i = 0.
\]

Where \( K(x_i, x_j) = \phi(x_i)^T \phi(x_j) \)

\[
y^*(z) = \text{sign}\left( \sum_{i \in S^V} \alpha_i y_i K(x_i, z) + b \right)
\]

Transforming the Data

- Computation in the feature space can be costly because it is high dimensional
  - The feature space is typically infinite-dimensional!
- The kernel trick comes to rescue
An Example for feature mapping and kernels

- Consider an input $x = [x_1, x_2]$
- Suppose $\phi(.)$ is given as follows
  \[ \phi\left(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}\right) = 1, \sqrt{2}x_1, \sqrt{2}x_2, x_1^2, x_2^2, \sqrt{2}x_1x_2 \]
- An inner product in the feature space is
  \[ \left\langle \phi\left(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}\right), \phi\left(\begin{bmatrix} x'_1 \\ x'_2 \end{bmatrix}\right) \right\rangle = \left\langle \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \begin{bmatrix} x'_1 \\ x'_2 \end{bmatrix} \right\rangle \]
- So, if we define the kernel function as follows, there is no need to carry out $\phi(.)$ explicitly
  \[ K(x, x') = \left(1 + x^T x'\right)^2 \]

More examples of kernel functions

- Linear kernel (we've seen it)
  \[ K(x, x') = x^T x' \]
- Polynomial kernel (we just saw an example)
  \[ K(x, x') = \left(1 + x^T x'\right)^p \]
  where $p = 2, 3, \ldots$. To get the feature vectors we concatenate all $p$th order polynomial terms of the components of $x$ (weighted appropriately)
- Radial basis kernel
  \[ K(x, x') = \exp\left(-\frac{1}{2}||x - x'||^2\right) \]
  In this case the feature space consists of functions and results in a non-parametric classifier.
The Optimization Problem

- The dual of this new constrained optimization problem is
  \[
  \max_{\alpha} \quad \mathcal{J}(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j (x_i^T x_j)
  \]

  s.t. \quad 0 \leq \alpha_i \leq C, \quad i = 1, \ldots, m

  \sum_{i=1}^{m} \alpha_i y_i = 0.

- This is very similar to the optimization problem in the linear separable case, except that there is an upper bound $C$ on $\alpha_i$ now.
- Once again, a QP solver can be used to find $\alpha_i$.

The SMO algorithm

- Consider solving the **unconstrained** opt problem:
  \[
  \max_{\alpha} W(\alpha_1, \alpha_2, \ldots, \alpha_m)
  \]

- We’ve already seen several opt algorithms!
  - ?
  - ?
  - ?

- Coordinate ascend:
Coordinate ascend

Sequential minimal optimization

- Constrained optimization:

\[
\max_{\alpha} \quad \tilde{J}(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j (x_i^T x_j)
\]

s.t. \( 0 \leq \alpha_i \leq C, \quad i = 1, \ldots, m \)

\[
\sum_{i=1}^{m} \alpha_i y_i = 0.
\]

- Question: can we do coordinate along one direction at a time (i.e., hold all \( \alpha_{[\cdot]} \) fixed, and update \( \alpha_i \)?)
The SMO algorithm

Repeat till convergence

1. Select some pair $\alpha_i$ and $\alpha_j$ to update next (using a heuristic that tries to pick the two that will allow us to make the biggest progress towards the global maximum).

2. Re-optimize $J(\alpha)$ with respect to $\alpha_i$ and $\alpha_j$, while holding all the other $\alpha_k$'s ($k \neq i, j$) fixed.

Will this procedure converge?

Convergence of SMO

$$\max_{\alpha} J(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j (x_i^T x_j)$$

KKT:

$$\begin{align*}
\text{s.t.} & \quad 0 \leq \alpha_i \leq C, \quad i = 1, \ldots, k \\
& \quad \sum_{i=1}^{m} \alpha_i y_i = 0.
\end{align*}$$

- Let's hold $\alpha_3, \ldots, \alpha_m$ fixed and reopt J w.r.t. $\alpha_1$ and $\alpha_2$
Convergence of SMO

- The constraints:
  \[ \alpha_1 y_1 + \alpha_2 y_2 = \xi \]
  \[ 0 < \alpha_1 < C \]
  \[ 0 \leq \alpha_2 \leq C \]

- The objective:
  \[ J(\alpha_1, \alpha_2, \ldots, \alpha_m) = J((\xi - \alpha_2 y_2)y_1, \alpha_2, \ldots, \alpha_m) \]

- Constrained opt:

Cross-validation error of SVM

- The leave-one-out cross-validation error does not depend on the dimensionality of the feature space but only on the # of support vectors!

Leave-one-out CV error = \[ \frac{\# \text{ support vectors}}{\# \text{ of training examples}} \]
Summary

- Max-margin decision boundary

- Constrained convex optimization
  - Duality
  - The KTT conditions and the support vectors
  - Non-separable case and slack variables
  - The SMO algorithm