Reminder
HWS due tonight at 11:59 pm!
- Remember to do both the questions on gradient descent and the midterm re-do (even if you got most of the questions right on the midterm).

Today
Gaussian Mixture Models (cont'd)
Support Vector Machines

Recap

GMMs can easily overfit:
- It can collapse a Gaussian component onto a single data example, making the likelihood of that data example arbitrarily high, and therefore likelihood of the entire dataset arbitrarily high.

How to avoid this:
- Force \( \Sigma_j \) for all \( j \in \{1, \ldots, K\} \), and not learn different \( \Sigma_j \)'s at all.
- This turns the likelihood \( P(x_i | z_i = j) \) to:

\[
\frac{1}{\sqrt{(2\pi)^d \det(\Sigma)}} \exp\left(-\frac{1}{2} \| x_i - \mu_j \|_2^2 \right)
\]
If we further constrain $p_1 = \cdots = p_k$ (so that the prior, $Pr(z=j)$ becomes a uniform distribution), then the E-step becomes:

$$T_i^* = \Pr(z_i = j \mid x_i) = \frac{\exp(-\frac{1}{2} \|x_i - \mu_j\|_2^2)}{\sum_{j=1}^K \exp(-\frac{1}{2} \|x_i - \mu_j\|_2^2)}$$

Probability of assigning $x_i$ to cluster $j$.

Compare to the cluster assignment in k-means:

$$z_i^* = \arg\min_{k \in \{1, \ldots, K\}} \|x_i - \bar{c}_k\|_2^2 = \arg\max_{k \in \{1, \ldots, K\}} -\frac{1}{2} \|x_i - \bar{c}_k\|_2^2$$

Hard max.

Then it becomes clear that GMM is a "softer" version of k-means.

Effect of forcing $\sum_j \sigma_j^2 I$:

Cannot model differences in the spread of data in different clusters.
This might be too drastic. One alternative is to force
\[ \Sigma_j = \sigma_j^2 I \]
\( \sigma_j^2 \) different for each cluster

What if we had the following dataset:

A more flexible alternative is to force \( \Sigma_j = \begin{pmatrix} \sigma_j^2 & \cdots \\ \vdots & \ddots \end{pmatrix} \)
Support Vector Machines
Let's return to the problem of classification. Given the following dataset, what are the possible decision boundaries of a linear classifier trained on the data?

Some decision boundaries are better than others, however. Consider the following decision boundary:
If the training examples were randomly drawn from some underlying distribution, when presented with a test example drawn from the same distribution, the predicted label could be incorrect:

![Diagram showing a decision boundary and a test example classified incorrectly.]

Intuitively, we want a decision boundary that is as far away from all the training examples as possible, so that the classifier generalizes well to unseen test examples.

![Diagram illustrating the concept of the "margin".]

The best decision boundary is the one that maximizes the margin.
Let's formulate an optimization problem to find a maximum-margin decision boundary. Unlike previously, we will use the convention $y_i \in \{-1, 1\}$ rather than $y_i \in \{0, 1\}$, which will make our formulation simpler.

First, we need to come up with a mathematical expression for the margin.

Consider a binary thresholding linear classifier, i.e.:

$$y = \begin{cases} 1 & \mathbf{w}^T \mathbf{x} > b \\ -1 & \mathbf{w}^T \mathbf{x} < b \end{cases} = \begin{cases} 1 & \mathbf{w}^T \mathbf{x} - b > 0 \\ -1 & \mathbf{w}^T \mathbf{x} - b < 0 \end{cases}$$

or equivalently,

The decision boundary is the set $\{\mathbf{x} : \mathbf{w}^T \mathbf{x} = b\}$, $\{\mathbf{x} : \mathbf{w}^T \mathbf{x} - b = 0\}$ which is a level set a linear function $\mathbf{x} \mapsto \mathbf{w}^T \mathbf{x}$.

Hence, it is a hyperplane. The vector $\mathbf{w}$ is perpendicular to the hyperplane. How do we see this?

\[\text{(Diagram: Vector and Hyperplane)}\]

For any point $A$ on this line, the vector from $O$ to $A$, denoted as $\overrightarrow{OA}$ is perpendicular to $\mathbf{w}$. Hence $\mathbf{w}^T(\overrightarrow{OA}) = 0$. So, $A$ must lie on the hyperplane. Special case when $b = 0$

Mathematically, if we pick any two points $\mathbf{x}_1$ and $\mathbf{x}_2$ on the hyperplane, and consider the direction from $\mathbf{x}_1$ to $\mathbf{x}_2$, i.e. $(\mathbf{x}_2 - \mathbf{x}_1)$, then

$$\mathbf{w}^T(\mathbf{x}_2 - \mathbf{x}_1) = \mathbf{w}^T \mathbf{x}_2 - \mathbf{w}^T \mathbf{x}_1 = b - b = 0.$$
This shows that any direction on the hyperplane is perpendicular to \( \overrightarrow{w} \), and so the hyperplane itself is perpendicular to \( \overrightarrow{w} \).

Now, we find an expression for the distance from any point \( \overrightarrow{x} \) to the hyperplane.

Let \( \overrightarrow{z} \) be an arbitrary point on the hyperplane; the distance to the hyperplane is then simply the length of the projection of \( \overrightarrow{x} - \overrightarrow{z} \) onto \( \overrightarrow{w} \).

So, the distance to the hyperplane is: \( \left| \langle \overrightarrow{x} - \overrightarrow{z}, \frac{\overrightarrow{w}}{\| \overrightarrow{w} \|_2} \rangle \right| \), which is \( \left| (\overrightarrow{x} - \overrightarrow{z})^T \frac{\overrightarrow{w}}{\| \overrightarrow{w} \|_2} \right| = \frac{1}{\| \overrightarrow{w} \|_2} \left| (\overrightarrow{x}^T \overrightarrow{w} - \overrightarrow{z}^T \overrightarrow{w}) \right| \). Since \( \overrightarrow{z} \) is on the hyperplane, \( \overrightarrow{z}^T \overrightarrow{w} = b \) (by definition of the hyperplane).

The distance to the hyperplane is therefore \( \frac{1}{\| \overrightarrow{w} \|_2} \left| \overrightarrow{x}^T \overrightarrow{w} - b \right| \).

Since the margin is defined as the minimum distance from the hyperplane from any training example, the distances from all training examples to the hyperplane must be at least the margin.
For the hyperplane to be a decision boundary of a classifier that classifies all training examples perfectly, it must perfectly separate all positive examples from all negative examples. In other words, all positive examples must lie on one side of the hyperplane, and all negative examples must lie on the other side. The set of inputs that result in a positive predicted label is \( \exists \vec{x}: \vec{w}^T \vec{x} > b \), and the set that results in a negative predicted label is \( \exists \vec{x}: \vec{w}^T \vec{x} < b \). To make the decision boundary perfect, we have to make sure that \( \forall i \) s.t. \( y_i = 1 \), \( \vec{w}^T \vec{x}_i > b \) and \( \forall i \) s.t. \( y_i = -1 \), \( \vec{w}^T \vec{x}_i < b \). We can simplify this condition to: \( y_i (\vec{w}^T \vec{x}_i - b) > 0 \), \( \forall i \).

Let's denote the margin as \( m \). Then we know that \( \forall i \),\( \frac{1}{\|\vec{w}\|_2} |\vec{x}_i^T \vec{w} - b| \geq m \) ensures that \( \vec{w}^T \vec{x}_i - b > 0 \). So, in this case, \( \vec{x}_i^T \vec{w} - b \geq m \|\vec{w}\|_2 \).

We rewrite the first condition as:

\[ |\vec{x}_i^T \vec{w} - b| \geq m \|\vec{w}\|_2 \]

So, either \( \vec{x}_i^T \vec{w} - b \geq m \|\vec{w}\|_2 \) or \( \vec{x}_i^T \vec{w} - b \leq -m \|\vec{w}\|_2 \).

When \( y_i = 1 \), the condition \( y_i (\vec{w}^T \vec{x}_i - b) > 0 \) ensures that \( \vec{w}^T \vec{x}_i - b > 0 \). So, in this case, \( \vec{x}_i^T \vec{w} - b \geq m \|\vec{w}\|_2 \).

Otherwise, \( \vec{w}^T \vec{x}_i - b < 0 \) and so \( \vec{x}_i^T \vec{w} - b \leq -m \|\vec{w}\|_2 \).
This condition can be more compactly written as:

$$\gamma_i (\vec{x}_i \cdot \vec{w} - b) \geq m \| \vec{w} \|_2.$$ 

Recall that our goal is to find a hyperplane where the margin is the greatest. This hyperplane can be characterized by the solution to the following optimization problem:

$$\begin{align*}
\max & \quad m \\
\text{subject to} & \quad \gamma_i (\vec{x}_i \cdot \vec{w} - b) \geq m \| \vec{w} \|_2 \quad \forall i \in \{1, \ldots, n\} \\
& \quad m \geq 0
\end{align*}$$

Unlike the optimization problems we've seen so far, this is a **constrained optimization** problem. In a constrained problem, the goal is to find the values of the variables that maximize the objective, **among** the set of values, i.e., those that cause all the constraints to be satisfied.

When we maximize the margin, there is at least one positive example and at least one negative example whose distance to the hyperplane is exactly the margin (otherwise we'd be able to increase the margin further). These examples are known as the support vectors. This is why the classifier that we obtain by solving the above optimization problem is called a **support vector machine**.
The circled examples are the support vectors. They are so named because they hold the hyperplanes that are on either side of the decision boundary in place. Removing any training example that is not a support vector will not change the solution (either \(m, \bar{w}\) or \(b\)). Removing a support vector could change the solution, but will not necessarily.

It turns out that the constraints in our optimization problem can be simplified to linear constraints, thereby allowing us to use a broader array of algorithms to solve it.

Observe that if \((\bar{w}_0, b_0, m)\) satisfies the constraints, then \(\forall \alpha > 0\) \((\alpha \bar{w}_0, \alpha b_0, m)\) would also satisfy the constraints, since

\[
Y_i (\vec{x}_i^T (\alpha \bar{w}_0) - \alpha b_0) = \alpha Y_i (\vec{x}_i^T \bar{w}_0 - b_0)
\]

Since \((\bar{w}_0, b_0, m)\) satisfies \(m \| \bar{w} \|_2 \geq \alpha m \| \bar{w} \|_2\), the constraints

To get rid of this, we can force \(\| \bar{w} \|_2\) to be a constant. Let's force it to be \(\frac{1}{m}\), or equivalently,

\[
m = \frac{1}{\| \bar{w} \|_2}.
\]
Then the optimization problem becomes:

\[
\begin{align*}
\max_{m, \tilde{w}, b} & \quad m \\
\text{s.t.} & \quad \gamma_i (\tilde{x}_i^T \tilde{w} - b) \geq m \|\tilde{w}\|_2 \quad \forall i \in \mathbb{I}, \ldots, n_3 \\
& \quad m \geq 0 \\
& \quad m = \frac{1}{\|\tilde{w}\|_2}
\end{align*}
\]

We then substitute the last constraint into the other constraints and the objective:

\[
\max_{\tilde{w}, b} \quad \frac{1}{\|\tilde{w}\|_2}
\]

\[
\text{s.t.} \quad \gamma_i (\tilde{x}_i^T \tilde{w} - b) \geq 1 \quad \forall i \in \mathbb{I}, \ldots, n_3
\]

Let's apply a strictly decreasing transformation \( t \mapsto \frac{2}{\sqrt{t}} \) to the objective:

\[
\min_{\tilde{w}, b} \quad \frac{1}{2} \|\tilde{w}\|_2^2
\]

\[
\text{s.t.} \quad \gamma_i (\tilde{x}_i^T \tilde{w} - b) \geq 1 \quad \forall i \in \mathbb{I}, \ldots, n_3
\]

This is now a convex quadratic objective with linear constraints and is therefore a quadratic programming (QP) problem. It can be solved using extensions of the simplex algorithm or the ellipsoid method.

As in kernel ridge regression, there is an alternative form that allows for kernelization. We derive this form by considering the dual of this optimization problem.
Lagrangian Duality

Consider a generic constrained optimization problem:

$$\min_{\overline{w}} \quad f(\overline{w})$$  \hspace{1cm} \text{objective function}

subject to:

$$g_i(\overline{w}) \leq 0 \quad \forall i \in \{1, \ldots, k\}$$  \hspace{1cm} \text{inequality constraints}

$$h_i(\overline{w}) = 0 \quad \forall i \in \{1, \ldots, l\}$$  \hspace{1cm} \text{equality constraints}

To differentiate this from the alternative optimization problem we'll derive, this is also known as the "primal optimization problem" or just "primal" for short.

For convenience, let's turn this into an equivalent unconstrained problem:

Define the following objective, which is known as a "generalized Lagrangian": known as the "dual variables"

$$L(\overline{w}, \overline{\lambda}, \overline{\nu}) = f(\overline{w}) + \sum_{i=1}^{k} \lambda_i g_i(\overline{w}) + \sum_{i=1}^{l} \nu_i h_i(\overline{w})$$

Consider the function $\Phi$ defined as follows:

$$\Phi(\overline{w}) = \max_{\overline{\lambda}, \overline{\nu}, \overline{x}} L(\overline{w}, \overline{\lambda}, \overline{\nu})$$

where $\lambda_i \geq 0 \forall i$

Observe that:

$$\Phi(\overline{w}) = \begin{cases} f(\overline{w}) & \text{if } g_i(\overline{w}) \leq 0 \quad \forall i \in \{1, \ldots, k\} \\
\infty & \text{otherwise} \end{cases}$$

and $h_i(\overline{w}) = 0 \quad \forall i \in \{1, \ldots, l\}$
Hence,
\[ \min \limits_{\bar{w}} \psi(\bar{w}) = \min \limits_{\bar{w}} \max \limits_{\lambda_i : \bar{\lambda}} L(\bar{w}, \bar{\lambda}, \bar{\nu}) \] is equivalent to our original constrained optimization problem (i.e. both have the same solutions).

Let's define \( p^* \equiv \min \limits_{\bar{w}} \max \limits_{\lambda_i : \bar{\lambda}} L(\bar{w}, \bar{\lambda}, \bar{\nu}) \), which is known as the "value" of the primal problem.

Now consider a different optimization problem:
\[ \max \limits_{\bar{\lambda}, \bar{\nu}} \min \limits_{\bar{w}} L(\bar{w}, \bar{\lambda}, \bar{\nu}) \]

This differs from the primal problem above, because we swapped the order of "max" and "min".

This is known as the "dual optimization problem".

Define \( d^* \equiv \max \limits_{\bar{\lambda}, \bar{\nu}} \min \limits_{\bar{w}} L(\bar{w}, \bar{\lambda}, \bar{\nu}) \), which is known as the "value" of the dual problem.

What's the relationship between the primal and the dual problems?

It turns out that \( p^* \geq d^* \) always. This condition is known as "weak duality".
Why?

We know that $\forall \wv, \quad L(\wv, \xv, \dv) \geq \min_{\wv} L(\wv', \xv, \dv)$

at every point in the space of $\xv$ and $\dv$. (1)

It follows that $\max_{\xv, \dv: \lambda; \vd} L(\wv, \xv, \dv) \geq \max_{\xv, \dv: \lambda; \vd} \min_{\wv} L(\wv', \xv', \dv'), \lambda; \vd$.

which we'll call $\bar{x}_0, \bar{d}_0$ (2)

since if we consider a particular point in the space of $\xv$ and $\dv$ that maximizes $\min_{\wv} L(\wv', \xv', \dv')$, then

$L(\wv, \bar{x}_0, \bar{d}_0) \geq \min_{\wv} L(\wv', \bar{x}_0, \bar{d}_0) = \max_{\xv, \dv: \lambda; \vd} \min_{\wv} L(\wv', \xv', \dv') \lambda; \vd$

by fact (1). Because $\max_{\xv, \dv: \lambda; \vd} L(\wv, \xv, \dv) \geq L(\wv, \bar{x}_0, \bar{d}_0), \forall \wv$, fact (2) follows. Since fact (2) is true for all $\wv$, it is true for a particular $\wv$ that minimizes

$\max_{\xv, \dv: \lambda; \vd} L(\wv, \xv, \dv)$. Hence, $\min_{\wv} \max_{\xv, \dv: \lambda; \vd} L(\wv, \xv, \dv) \geq \max_{\xv, \dv: \lambda; \vd} \min_{\wv} L(\wv', \xv', \dv') \lambda; \vd$

The quantity $p^* - d^*$ is known as the "duality gap." Under some conditions, $p^* = d^*$. In this case, we say "strong duality" holds, or the duality gap is 0. One sufficient condition for strong duality to hold is Slater's condition.
Slater's Condition:

If \( f(\overline{w}) \) and \( g_i(\overline{w}) \) are convex in \( \overline{w} \) \( \forall i \in \mathbb{I}, \ldots, k_3 \) and \( h_i(\overline{w}) \) is linear in \( \overline{w} \) \( \forall i \in \mathbb{I}, \ldots, l_3 \), and there exists a point \( \overline{w}_0 \) such that \( g_i(\overline{w}_0) < 0 \) \( \forall i \) and \( h_i(\overline{w}_0) = 0 \) \( \forall i \), then \( p^* = d^* \). (strict inequality!)

When strong duality holds, there exists \( (\overline{w}^*, \overline{\lambda}^*, \overline{\nu}^*) \) such that \( \overline{w}^* \) is a solution to the primal problem, \( \overline{\lambda}^* \) and \( \overline{\nu}^* \) are a solution to the dual problem and

\[
\begin{align*}
\min_{\overline{w}} & \quad f(\overline{w}) + \sum_{i=1}^k \lambda_i \nabla \overline{w} g_i(\overline{w}^*) + \sum_{i=1}^l \nu_i \nabla \overline{w} h_i(\overline{w}^*) \\
\text{s.t.} & \quad \gamma_i (\overline{x}_i^T \overline{w} - b) \geq 1 \quad \forall i \in \mathbb{I}, \ldots, n_3
\end{align*}
\]

Back to SVMs:

Recall that the original (primal) problem for SVMs is:

\[
\min_{\overline{w}, b} \frac{1}{2} ||\overline{w}||_2^2 \\
\text{s.t.} \quad \gamma_i (\overline{x}_i^T \overline{w} - b) \geq 1 \quad \forall i \in \mathbb{I}, \ldots, n_3
\]
What's the generalized Lagrangian?

\[ L((\vec{w} \ b)^T, \lambda) = \frac{1}{2} ||\vec{w}||_2^2 + \sum_{i=1}^{n} \lambda_i (1 - y_i (\vec{x}_i^T \vec{w} - b)) \]

The dual problem is:

\[ \max_{\lambda: \lambda_i \geq 0 \ \forall i} \ \min_{\vec{w}, b} L((\vec{w} \ b)^T, \lambda) \]

Let's find a closed-form expression for \( \min_{\vec{w}, b} L((\vec{w} \ b)^T, \lambda) \).

\[ \nabla_{\vec{w}} L((\vec{w} b)^T, \lambda) = \frac{1}{2} (2 \vec{w}) + \sum_{i=1}^{n} \lambda_i (-y_i \vec{x}_i) \]

\[ = \vec{w} - \sum_{i=1}^{n} \lambda_i y_i \vec{x}_i \]

At optimality, \( \vec{w} = \sum_{i=1}^{n} \lambda_i y_i \vec{x}_i = 0 \Rightarrow \vec{w} = \sum_{i=1}^{n} \lambda_i y_i \vec{x}_i \). (2)

\[ \nabla_{\vec{b}} L((\vec{w} b)^T, \lambda) = \sum_{i=1}^{n} \lambda_i (-y_i)(-1) = \sum_{i=1}^{n} \lambda_i y_i \]

At optimality, \( \sum_{i=1}^{n} \lambda_i y_i = 0 \). (3)

Now, we plug in the expression for \( \vec{w} \) (2) into the Lagrangian (1):

\[ L((\vec{w} \ b)^T, \lambda) = \frac{1}{2} \left( \sum_{i=1}^{n} \lambda_i y_i \vec{x}_i \right)^T \left( \sum_{i=1}^{n} \lambda_i y_i \vec{x}_i \right) \]

\[ + \sum_{i=1}^{n} \lambda_i (1 - y_i (\vec{x}_i^T (\sum_{j=1}^{n} \lambda_j y_j \vec{x}_j))) + \sum_{i=1}^{n} \lambda_i y_i b \]

\[ = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \lambda_i \lambda_j y_i y_j \vec{x}_i \vec{x}_j + \sum_{i=1}^{n} \lambda_i - \sum_{j=1}^{n} \lambda_j y_j \vec{x}_j^T \vec{x}_j \]

\[ + b \sum_{i=1}^{n} \lambda_i y_i \]

\[ = 0 \text{ by (3)} \]
After simplifying, we have:

\[ \mathcal{L}((\overline{\mathbf{w}} b)^T, \overline{\mathbf{x}}) = \sum_{i=1}^{n} \lambda_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \lambda_i \lambda_j \gamma_i \gamma_j \overline{x}_i^T \overline{x}_j \]

So, the dual optimization problem is:

\[
\begin{align*}
\max_{\lambda} & \quad \sum_{i=1}^{n} \lambda_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \lambda_i \lambda_j \gamma_i \gamma_j \overline{x}_i^T \overline{x}_j \\
\text{s.t.} & \quad \sum_{i=1}^{n} \lambda_i \gamma_i = 0 \quad \text{(from (3))} \\
& \quad \lambda_i \geq 0 \quad \forall i \in \{1, \ldots, n\} 
\end{align*}
\]

Let's check if Slater's condition is satisfied:

The objective function \( \frac{1}{2} \|\overline{\mathbf{w}}\|_2^2 \) is convex.

The inequality constraints \( \gamma_i (\overline{x}_i^T \overline{\mathbf{w}} - b) + 1 \leq 0 \) are linear, and therefore convex.

As long as the dataset is linearly separable, we can make the margin small enough (or equivalently \( \|\overline{\mathbf{w}}\|_2 \) large enough so that all inequality constraints are strictly satisfied).

Hence, strong duality holds. We can therefore use (2) to find the primal solution \( \overline{\mathbf{w}}^* \) from the dual solution \( \overline{\lambda}^* \):

\[ \overline{\mathbf{w}}^* = \sum_{i=1}^{n} \lambda_i^* \gamma_i \overline{x}_i \]

The predicted label of the model is:

\[ \hat{y}_{\text{test}} = \begin{cases} 
1 & \text{if } z > 0 \\
-1 & \text{if } z < 0
\end{cases} \]

where \( z = \overline{\mathbf{w}}^* \overline{x}_{\text{test}} - b^* = \sum_{i=1}^{n} \lambda_i y_i \overline{x}_i^T \overline{x}_{\text{test}} - b^* \) a dot product.
Since the predicted label only depends on $X_i$'s through dot products between them and the test example, we can kernelize the model! This can make the decision boundary non-linear.

Observe that the model's prediction (before thresholding), $\hat{z}$, is a linear combination of dot products between training examples and the test example, just as in kernel ridge regression.

This is not a coincidence! It turns out that there is a broad class of models where the optimal prediction rule is always of this form. This class of models is characterized precisely by the representer theorem. These models are known as "kernel machines" or "kernel methods".

Implications of Complementary Slackness

The optimal solution to the dual form of the SVM, $\lambda^*$, tells us which of the inequality constraints in the primal form are binding.

Since $\lambda_i^* g_i(\overrightarrow{w}^*) = 0 \ \forall i$, if $\lambda_i^* \neq 0$, $g_i(\overrightarrow{w}^*) = 0$.

This means that if $\lambda_i^* \neq 0$, $\gamma_i(\overrightarrow{X_i}^T \overrightarrow{w} - b) = 1$, which means that $\overrightarrow{X_i}$ is on the margin and is therefore a support vector.
On the other hand, if \( g_i(w^* w) \geq 0 \), \( \lambda_i^* = 0 \). This means that if \( y_i (\overline{x_i}^T \overline{w} - b) \neq 1 \), \( \lambda_i^* = 0 \); so if \( \overline{x_i} \) is not on the margin/is not a support vector, \( \lambda_i^* = 0 \).

Note the direction of implications: if \( \lambda_i^* = 0 \), \( \overline{x_i} \) may or may not be on the margin/may or may not be a support vector. Similarly, if \( g_i(w^*) = 0 \), i.e. if \( y_i (\overline{x_i}^T \overline{w} - b) = 1 \), i.e.: \( \overline{x_i} \) is on the margin/is a support vector, \( \lambda_i^* \) may or may not be 0.

**Linear Inseparability**

What if the data is not linearly separable?

1. If the data is almost linearly separable:
   - Extend the formulation to have slack variables, which permit (but penalize) violations of constraints.
   - This is known as a "soft-margin SVM".
   - More details in the course notes and HW.

2. If the data is separable by a non-linear decision boundary:
   - Kernelize and choose the right kernel.
   - Can be viewed as lifting data to higher-dimensional feature space, in which data is linearly separable.