Linear classification

Perceptron

$f(\mathbf{x}) = \boldsymbol{\theta} \cdot \mathbf{x} + \theta_0 = \sum_{i=1}^{d} \theta_i x_i + \theta_0$, $\hat{y} = \begin{cases} 1, & \text{if } f(x) \geq 0 \\ -1, & \text{if } f(x) < 0 \end{cases}$

Decision boundary, a **hyperplane** in $\mathbb{R}^d$:
$H = \{\mathbf{x} \in \mathbb{R}^d : f(\mathbf{x}) = 0\} = \{\mathbf{x} \in \mathbb{R}^d : \boldsymbol{\theta} \cdot \mathbf{x} + \theta_0 = 0\}$

$\boldsymbol{\theta}$ is the **normal** of the hyperplane,
θ_0 is the **offset** of the hyperplane from origin,
$-\frac{\theta_0}{\|\boldsymbol{\theta}\|}$ is the **signed distance** from the origin to hyperplane.

Perceptron algorithm,
Input: $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n) \in \mathbb{R}^d \times \{\pm 1\}$
while some $y_i \neq \text{sign}(\boldsymbol{\theta} \cdot \mathbf{x}_i)$
    pick some misclassified $(\mathbf{x}_i, y_i)$
    $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + y_i \mathbf{x}_i$

Given a **linearly separable data**, perceptron algorithm will take no more than $\frac{R^2}{\gamma^2}$ updates to **converge**, where $R = \max_i \|\mathbf{x}_i\|$ is the radius of the data, $\gamma = \min_i \frac{y_i(\boldsymbol{\theta} \cdot \mathbf{x}_i)}{\|\boldsymbol{\theta}\|}$ is the margin.
Also, $\frac{\boldsymbol{\theta} \cdot \mathbf{x}}{\|\boldsymbol{\theta}\|}$ is the signed distance from H to $\mathbf{x}$ in the direction $\boldsymbol{\theta}$.

$\boldsymbol{\theta} = \sum_i \alpha_i y_i \mathbf{x}_i$, thus any inner product space will work, this is a **kernel**.

Gradient descent view of perceptron, minimize margin cost function $J(\boldsymbol{\theta}) = \sum_i (-y_i(\boldsymbol{\theta} \cdot \mathbf{x}_i))_+$ with $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta \nabla J(\boldsymbol{\theta})$

Support Vector Machine

Hard margin SVM,
$\min_{\boldsymbol{\theta}} \|\boldsymbol{\theta}\|^2$, such that $y_i \boldsymbol{\theta} \cdot \mathbf{x}_i \geq 1 (i = 1, \ldots, n)$
Soft margin SVM,
$\min_{\boldsymbol{\theta}} \|\boldsymbol{\theta}\|^2 + C \sum_{i=1}^{n} (1 - y_i \boldsymbol{\theta} \cdot \mathbf{x}_i)_+$

Regularization and SVMs: Simulated data with many features $\phi(\mathbf{x})$; C controls trade-off between margin $1/\|\boldsymbol{\theta}\|$ and fit to data; Large C: focus on fit to data (small margin is ok). More overfitting. Small C: focus on large margin, less tendency to overfit. Overfitting increases with: less data, more features.

$\boldsymbol{\theta} = \sum_j \alpha_j y_j \mathbf{x}_j$, $\alpha_j \neq 0$ only for support vectors.

$K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j)$, K is called a kernel.
Solve α_j to determine $\sum_j \alpha_j y_j \phi(\mathbf{x}_j)$.
Compute the classifier for a test point $\mathbf{x}$ via
$\boldsymbol{\theta} \cdot \phi(\mathbf{x}) = \sum_j \alpha_j y_j K(\mathbf{x}_j, \mathbf{x})$

---

degree-m polynomial kernel: $K_m(\mathbf{x}, \tilde{\mathbf{x}}) = (1 + \mathbf{x} \cdot \tilde{\mathbf{x}})^m$
radial basis function kernel: $K_{rbf}(\mathbf{x}, \tilde{\mathbf{x}}) = \exp(-\gamma \|\mathbf{x} - \tilde{\mathbf{x}}\|^2)$