Chapter 2

Learning Linear Separators, SVMs and Kernels

Contents

2 Learning Linear Separators, SVMs and Kernels **1**

2.1 Linear Separator 3
2.2 Perceptron Algorithm 3
2.3 Geometric Margin 3
2.4 Support Vector Machine 4
2.5 Optimal Linear Separator 4
2.6 Classification Margin 4
2.7 Maximizing the Margin 4
2.8 Linear SVM 4
2.9 Lagrangian Duality 5
2.10 SVM Solution 5
2.11 Soft Margin Classification 6
2.12 Kernel Method 6
2.13 Kernel Trick 7

Contents

2.1 Linear Separator

Assuming that red and blue datasets represents points X_1 and X_2, then the two sets X_1 and X_2 are linearly separable if there exists $(n+1)$ real numbers $w_1, w_2, \ldots, w_n, k$

- such that every point in X_1 satisfies $\sum_{i=1}^{n} w_i x_i < k$
- such that every point in X_2 satisfies $\sum_{i=1}^{n} w_i x_i > k$

Binary classification $y_i \in \{-1, 1\}$ can be viewed as the task of separating classes in feature space.

- Hypothesis class of linear decision surfaces is $f(x_i) = \text{sign}(\mathbf{w}^T \mathbf{x_i} + b)$.
- Without loss of generality, we assume that $b = 0$. Thus, we get the simplified $f(x_i) = \text{sign}(\mathbf{w}^T \mathbf{x_i})$.
- $(y_i)(\mathbf{w}^T \mathbf{x}_i) > 0 \Leftrightarrow$ data point x_i is correctly classified.
  - Remember, y_i is counting as 1 or -1.

2.2 Perceptron Algorithm

- Set time $t = 1$, start with vector $\mathbf{w}_1 = \vec{0}$.
- Given example $\mathbf{x}$, predict positive iff (if and only if) $\mathbf{w_1} \cdot \mathbf{x} \geq \mathbf{0}$.
- On a mistake, update as follows:
  - Mistake on positive, then update $\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t + \mathbf{x}$.
  - Mistake on negative, then update $\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t - \mathbf{x}$.

2.3 Geometric Margin

The margin of example ...

The margin γ of a set of examples S w.r.t (with respect to) a linear separator $\mathbf{w}$ is the largest margin over points $\mathbf{x} \in S$. Theorem: If the data has a margin γ and all points lie inside a ball of radius R, then the Perceptron algorithm makes $\leq \frac{R}{\gamma^2}$ mistakes.

2.4 Support Vector Machine

Support vector machines (SVMs) are supervised max-margin models with associated learning algorithms.

- Good generalization in theory.
- Good generalization in practice.
- Work well with few training instances.
- Find globally best model.
- Efficient algorithms.
- Amenable to the kernel trick.

2.5 Optimal Linear Separator

Which of the linear separators is optimal?

2.6 Classification Margin

Examples closest to the hyperplane are support vectors. Margin ρ of the separator is the distance between support vectors.

2.7 Maximizing the Margin

- Better Generalization – A larger margin allows the SVM to better generalize to new, unseen data, leading to higher predictive accuracy.
- Improved Robustness – A larger margin can lead to improved robustness against noise and outliers in the training data, as it allows for greater tolerance of misclassified examples.
- Reducing Overfitting – A larger…

2.8 Linear SVM

Let training set $\left\{(\mathbf{x}_i, y_i)_{i=1\dots n}, \mathbf{x}_i \in \mathbb{R}^d, y_i \in \{-1, 1\}\right\}$ be separated by a hyperplane with margin ρ. Then for each training example $(\mathbf{x}_i, y_i)$

$$\begin{array}{ll} \mathbf{w}^T\mathbf{x}_i + b \geq 1 & \text{if } y_i = 1 \\ \mathbf{w}^T\mathbf{x}_i + b \leq -1 & \text{if } y_i = -1 \end{array} \Leftrightarrow y_i\left(\mathbf{w}^T\mathbf{x}_i + b\right) \geq 1$$

Geometrically, the distance between the 2 hyperplanes can be expressed as:

$$\rho = \frac{2}{||w||} \tag{2.1}$$

Then we can formulate the quadratic optimization problem:

$$\text{Find } \mathbf{w} \text{ and } b \text{ such that}$$

$$\rho = \frac{2}{||\mathbf{w}||}$$

$$\text{is maximized and for all } (\mathbf{x}_i, y_i), i = 1 \dots n : y_i \left(\mathbf{w}^T \mathbf{x}_i + b\right) \geq 1$$

Which can be reformulated as:

$$\mathbf{x}_i, y_i, \text{ find } \mathbf{w} \text{ and } b \text{ such that}$$

$$\text{Minimize } Q(w) = \frac{1}{2}||\mathbf{w}||^2 = \frac{1}{2}\mathbf{w}^T\mathbf{w}$$

$$\text{subject to } \quad \dots$$

2.9 Lagrangian Duality

- Need to optimize a quadratic function subject to linear constraints.
- Quadratic optimization problems are a well-known class of mathematical programming problems for which several (non-trivial) algorithms exist.
- Solution involves constructing dual problem where Lagrange multipliers a_i is associated with all inequality constraint in primal (original) problem:

$$\forall i, \text{ find } a_1, \dots, a_n \text{ such that ... subject to } a_i \geq 0$$

2.10 SVM Solution

- Given a solution $a_1 \dots a_n$ to the dual problem, solution to the primal is:

$$\mathbf{w} = \sum a_i y_i \mathbf{x} \quad b = y_k - \sum a_i y_i \mathbf{x}_i^T \mathbf{x}_k \text{ for any } a_k > 0$$

- Each non-zero a_i indicates that corresponding $\mathbf{x}_i$ is a support vector.
- Then the classifying function is:

$$f(x) = \sum a_i y_i \mathbf{x}_i^T \mathbf{x} + b$$

- Notice that it relies on the inner product between the test point $\mathbf{x}$ and the support vectors $\mathbf{x}_i$.
- Solving the optimization problem involves computing the inner products $\mathbf{x}_i^T \mathbf{x}_j$ between all training points.

2.11 Soft Margin Classification

- What if the training set is not linearly separable?
- Slack variables can be added to allow misclassification of difficult or noisy examples; thus, the result margin is called soft.

2.12 Kernel Method

- If we map the input vectors into a very high-dimensional feature space, the task of finding the maximum-margin separator can become computationally intractable.
- All of the computations that we need to do to find the maximum-margin separator (SVM optimization problem) can be expressed in terms of inner products between pairs of data points (in the high-dimensional feature space).
- These inner products are the only part of the computation that depends on the dimensionality of the high-dimensional space. So, if we had a fast way to do the dot products, we would not have ot pay a price
- The kernel trick is just a way of doing inner products a whole lot faster than is usually possible. It relies on choosing a way of mapping to the high-dimensional feature space that allows fast scalar products.
- By using a nonlinear vector function $\phi(x) = \langle \phi(x_1), \ldots, \phi(x_n) \rangle$, the n-dimensional input vector $\mathbf{x}$ can be mapped into high-dimensional feature space. The decision function in the feature space is expressed as:

$$f(x) = \mathbf{w}^T \phi(\mathbf{x}) + b$$

- In terms of solving the quadratic optimization problem of SVM, each training data point is in the form of dot products. A kernel function K simplifies the calculation of dot product terms

$$K(\mathbf{x_1},)$$

2.13 Kernel Trick

Example: Take this 2-dimensional vector: $\mathbf{x} = [x_1, x_2]$

$$\begin{aligned} K(\mathbf{x}_i, \mathbf{x}_j) &= (1 + \mathbf{x}_i^T \mathbf{x}_j)^2 \\ &= 1 + x_{i1}^2 x_{j1}^2 + 2x_{i1}x_{j1}x_{i2}x_{j2} + x_{i2}^2 x_{j2}^2 + 2x_{i1}x_{j1} + 2x_{i2}x_{j2} \\ &= \left[1 \; x_{i1}^2 \sqrt{2} x_{i1}\right] \end{aligned}$$