CISC 371 Class 34

SVM – The Kernel Trick

Texts: [1] pp. 423–432

Main Concepts:

- *Embedding vectors in a high-dimensional space*
- *Kernel functions compute embedded dot products*
- *Gram matrix for a design matrix*
- *Kernel trick: use the Gram matrix in the dual formulation*

Sample Problem, Machine Inference: What is the optimal decision surface for linearly inseparable data?

In machine learning, binary classification problems do not always have a linearly separable structure. Data vectors that are not linearly separable can sometimes be mapped to linearly separable vectors by means of a non-linear transformation. Nonlinear transformations are common in the SVM literature, in part because the dual representation of the SVM conveniently represents and solves certain kinds of these transformations.

34.1 Linear Separation Using an Embedding

The principal method for managing non-linearly separable data, which is commonly used in machine learning and data analytics, is to map each data vector to a higher-dimensional vector space. In many applications, data vectors that seem to be not linearly separable can be mapped to another vector space in which they *are* linearly separable.

For example, we might be provided with labeled data vectors that are in a 2D vector space. The next higher dimension is 3D. Some forms of mappings that are effective in practice are to copy the entries of $\vec{x}_j$ into a new size 3 vector, then to put some function of each data vector $\vec{x}_j$ into the third entry. These forms would map a 2D vector $\vec{x}$ to a 3D vector $\vec{y}$.

These are examples of *embeddings*, which are smooth maps of one dimension into a space of higher dimension. Although it is not exact, we can think of an embedding as being an "inverse projection": a projection maps a higher-dimensional vector space to a lower-dimensional vectors space and an embedding reverse this process.

Suppose we are given a 2D data set that has 24 data vectors with label +1 and 16 data vectors with label -1 in the picture-like data set that is illustrated in Figure 34.1. By inspection, we can determine that these sets are not linearly separable – but they do seem to be *non-linearly* separable.

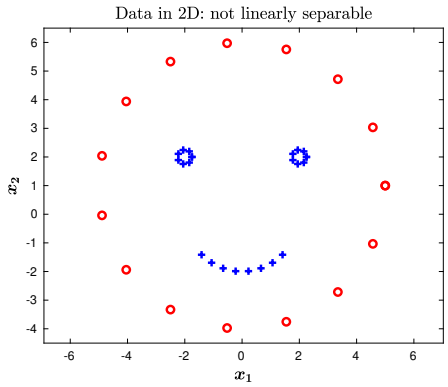

Figure 34.1: Pictorial data in 2D. The vectors with label +1 are plotted as a plus sign; vectors with label -1 are plotted as open circles.

Suppose that we try an embedding map that, for each data vector $\vec{u}_i$, "copies" the data vector into the first two entries of a 3D data vector $\hat{u}_i$ and then computes the squared length of $\vec{u}_i$ as the third entry of $\hat{u}_i$. The idea is that data vectors that are further from the original are embedded as 3D vectors that have a larger third entry. This would be the map

$$\vec{u} \hookrightarrow \hat{u} \quad \text{which is} \quad \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \hookrightarrow \begin{bmatrix} u_1 \\ u_2 \\ u_1^2 + u_2^2 \end{bmatrix} = \begin{bmatrix} u_1 \\ u_2 \\ \|\vec{u}\|^2 \end{bmatrix} \tag{34.1}$$

As we can see in Figure 34.2, these 3D data are linearly separable.

34.2 Kernel Functions and the Gram Matrix

In machine learning, the embedding function is typically written as $\vec{\phi}(\vec{u})$. If the embedding function has certain properties, it is possible to perform the necessary computations for the SVM without actually embedding the data vector $\vec{x}_i$ in a higher-dimensional space.

Let us consider the effect that embedding has on the primal Lagrange formula, which is Equation 32.7. The objective term, which is $1/2\vec{w}^T\vec{w}$, would have the vector $\vec{w}$ replaced by the higher-

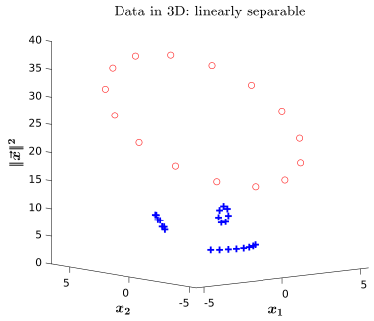

Figure 34.2: Pictorial data in 3D. The vectors with label +1 are plotted as a plus sign; vectors with label -1 are plotted as open circles.

dimensional vector $\hat{w}$. The embedded objective would be

$$\frac{1}{2}\hat{w}^T\hat{w} \tag{34.2}$$

The inequality constraints, which are managed by using the KKT conditions to introduce the Lagrange multipliers $\vec{\alpha}$, would have:

- The weight vector $\vec{w}$ replaced by the higher-dimensional vector $\hat{w}$
- Each data vector $\vec{x}_i$ replaced by the higher-dimensional vector $\hat{x}_i$, which produce a higher-dimensional design matrix $\hat{X}$

The revised constraint term would then be

$$\vec{\alpha}^T[\vec{1} - Y\hat{X}^T\hat{w} - b\vec{y}] \tag{34.3}$$

From Equation 34.2 and Equation 34.3, the previous methods can be applied to derive the dual formulation. The dual Lagrange function of Equation 33.17 would be revised by replacing the matrix product XX^T with $\hat{X}\hat{X}^T$. The objective of the dual formulation would be

$$-\frac{1}{2}\vec{\alpha}^T Y\hat{X}^T\hat{X}Y\vec{\alpha} \tag{34.4}$$

The insight for the *kernel trick* in training a SVM with a design matrix X is: *the embedding does not need to be computed*. In optimizing the objective function in Equation 34.4, dot products of the embedded vectors are needed. It is possible to compute the dot products without computing the embedding, for certain useful embeddings.

We can see that the matrix XX^T is symmetric and, if the data vectors are linearly independent, then it is also positive semidefinite. The entry (i, j) of the symmetric matrix is

$$\vec{x}_i^T \vec{x}_j \tag{34.5}$$

If we embed the data vector $\vec{x}_i$ in a higher-dimensional space as $\hat{x}_i$, and embed $\vec{x}_j$ as $\hat{x}_j$, the dot product of Equation 34.5 would be

$$\hat{x}_i^T \hat{x}_j = [\vec{\phi}(\vec{x}_i)]^T \vec{\phi}(\vec{x}_j) \tag{34.6}$$

As defined in the extra notes for this class, a *kernel function* is a function that is symmetric and positive semidefinite. For an appropriate kernel function $\kappa(\cdot, \cdot)$ that is defined for the vector space $\mathbb{R}^n$, the mapping in Equation 34.6 can be avoided and the dot product can be computed directly.

For example, the embedding $\vec{\phi}(\cdot)$ in Equation 34.1 has a kernel function

$$\kappa(\vec{u}, \vec{v}) = \vec{u}^T \vec{v} + (\vec{u}^T \vec{v})^2 \tag{34.7}$$

For any design matrix X that is full rank, and any kernel function $\kappa(\cdot, \cdot)$ defined on the vectors that are the columns of X, the *Gram matrix* is the matrix K that has the entries

$$K_{ij} \stackrel{\text{def}}{=} \kappa(\vec{x}_i, \vec{x}_j) \tag{34.8}$$

As described in the extra notes, and proved in many textbooks on matrix analysis, the Gram matrix of a full-rank design matrix K is symmetric and positive semidefinite. This implies that the objective term in Equation 34.4 is a quadratic convex function.

34.3 The Kernel Trick in the Dual Formulation of the SVM

With these mathematical preliminaries, the kernel trick is simple to describe. For an appropriately defined kernel function κ, the objective term in Equation 34.4 is replaced with the Gram matrix of X, so the objective is

$$\frac{1}{2} \vec{\alpha}^T Y K Y \vec{\alpha} \tag{34.9}$$

Equation 34.9 is easily incorporated into the SMO algorithm, or other quadratic programming methods such as sequential quadratic programming or an interior-point method.

34.4 Some Kernel Functions for Vector Spaces

The kernel function of Equation 34.7 is specialized for education in this course, and is not widely used. Here are some examples of commonly used kernel functions and their interpretations. In all cases, the kernel function will be described for a pair of vectors $\kappa(\vec{u}, \vec{v})$.

Linear Kernel: The dot product of vectors is the basic kernel function

$$\kappa(\vec{u}, \vec{v}) = \vec{u}^T \vec{v} \tag{34.10}$$

Polynomial Kernel: The dot product of vectors can be added to a constant c and raised to a power p, which is kernel function

$$\kappa(\vec{u}, \vec{v}) = (\vec{u}^T \vec{v} + c)^l \tag{34.11}$$

Equation 34.11 is usually presented for $c = 1$. An alternative that is sometimes presented is $c = 0$, which is an *exact power* kernel function.

Gaussian Kernel: The square of the norm of the distance between vectors can be used in the unscaled normal distribution function $e^{-x^2/2}$. In the SVM literature, this is usually presented with a variance σ^2 as the kernel function

$$\kappa(\vec{u}, \vec{v}) = \exp\left(\frac{-\|\vec{u} - \vec{v}\|^2}{2\sigma^2}\right) \tag{34.12}$$

The variance σ^2 is a crucial hyper-parameter in the SVM. A small variance causes the Gram matrix to approach the identity matrix. A large variance causes the entries in the Gram matrix to approach a constant value. The variance can be supplied by the user and can sometimes be estimated from the design matrix X.

In `MATLAB`, the Gaussian kernel is

$$\kappa(\vec{u}, \vec{v}) = \exp\left(-\|\vec{u} - \vec{v}\|^2\right) \tag{34.13}$$

Laplacian kernel: The norm of the distance between vectors, rather than the square of the norm, can be used in an exponential function.

$$\kappa(\vec{u}, \vec{v}) = \exp\left(\frac{-\|\vec{u} - \vec{v}\|}{\sigma}\right) \tag{34.14}$$

Sigmoid kernel: The inner product of vectors can be used as an argument in the hyperbolic tangent function that is common in neural networks. This must be done with care because not all of the sigmoid hyper-parameters produce a positive semidefinite Gram matrix from a full-rank design matrix X. The sigmoid kernel is

$$\kappa(\vec{u}, \vec{v}) = \tanh(\gamma \vec{u}^T \vec{v} + \beta) \tag{34.15}$$

A sufficient condition for Equation 34.15 to produce a positive semidefinite Gram matrix is that γ is positive and β is negative, which is $\gamma > 0$ and $\beta < 0$. The conditions are so sensitive that `MATLAB` does not currently offer this kernel as an option.

Extra Notes

34.5 Extra Notes for Gram Matrix and Kernel Function

Definition: Gram matrix

For any non-empty set $\mathbb{X}$, any $m \in \mathbb{N}_+$, any $i \in \mathbb{N}_+ : i \leq m$, any $j \in \mathbb{N}_+ : j \leq m$, any $\vec{x}_i \in \mathbb{X}$, any $\vec{x}_j \in \mathbb{X}$, and any symmetric function $\kappa : \mathbb{X} \times \mathbb{X} \to \mathbb{R}$,

the *Gram matrix* of the members $\{\vec{x}_i : 0 < i \leq m\}$ is defined as the matrix K such that

$$K_{ij} \stackrel{\text{def}}{=} \kappa(\vec{x}_i, \vec{x}_j) \tag{34.16}$$

Observation: Because the function $\kappa(\cdot, \cdot)$ is symmetric, the Gram matrix of a set of members of $\mathbb{X}$ is symmetric.

In linear algebra, the set $\mathbb{X}$ is typically a vector space $\mathbb{R}^n$ and the symmetric function κ is typically an inner product $\langle \cdot, \cdot \rangle$. See Horn and Johnson, 1990, for definitions and examples.

In machine learning, the set $\mathbb{X}$ can be quite diverse, including strings of symbols in text analysis. The symmetric function κ may be a measure, especially a metric, on the set $\mathbb{X}$.

Definition: Kernel function

For any non-empty set $\mathbb{X}$, any $n \in \mathbb{N}_+$, a symmetric function $\kappa : \mathbb{X} \times \mathbb{X} \to \mathbb{R}$ is a *positive semidefinite kernel function* is defined as:

a function for which the Gram matrix is positive semidefinite, or

$$K \succeq 0 \tag{34.17}$$

Observation: In linear algebra, an inner product on $\mathbb{R}^n$ is a kernel function. For a vector space and a kernel that is an inner product, if the vectors $\vec{x}_i$ are linearly independent then the Gram matrix is positive semidefinite.

End of Extra Notes

References

[1] Hastie T, Tibshirani R, Friedman J: The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer, New York, 2009