SVM and Kernels

Kairit Sirts

09.05.2014

Keywords

- Lagrangian and Lagrange multipliers
- Primal and dual problems
- Kernel trick

Lagrangian theory

When we have an objective function $f(\mathbf{w})$ and equality constraints $h_i(\mathbf{w}) = 0, i = 1, \ldots, m$, then the Lagrangian function is defined as:

$$L(\mathbf{w}, \boldsymbol{\beta}) = f(\mathbf{w}) + \sum_{i=1}^{m} \beta_i h_i(\mathbf{w}),$$

where the coefficients β_i are called Lagrange multipliers.

Minimality conditions

Theorem (Fermat)

A necessary condition for $\mathbf{w}^*$ *to be a minimum of* $f(\mathbf{w})$ *is* $\frac{\partial f(\mathbf{w}^*)}{\partial \mathbf{w}} = \mathbf{0}$. *This condition, together with convexity of* f, *is also a sufficient condition.*

Theorem (Lagrange)

A necessary condition for a point $\mathbf{w}^*$ *to be a minimum of* $f(\mathbf{w})$ *subject to* $h_i(\mathbf{w}) = 0, i = 1, \ldots, m$ *is:*

$$\frac{\partial L(\mathbf{w}^*, \boldsymbol{\beta}^*)}{\partial \mathbf{w}} = 0$$
$$\frac{\partial L(\mathbf{w}^*, \boldsymbol{\beta}^*)}{\partial \boldsymbol{\beta}} = 0,$$

The above conditions are also sufficient provided that $L(\mathbf{w}, \boldsymbol{\beta}^*)$ *is a convex function of* $\mathbf{w}$.

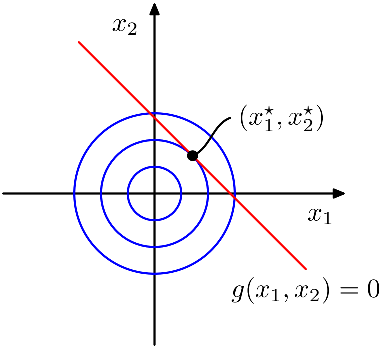
Lagrange multipliers: example

Maximize:

$$f(x_1, x_2) = 1 - x_1^2 - x_2^2$$

Subject to:

$$g(x_1, x_2) = x_1 + x_2 - 1 = 0$$



Lagrange multipliers example: solution

The corresponding Lagrangian function is:

$$L(\mathbf{x}, \lambda) = 1 - x_1^2 - x_2^2 + \lambda(x_1 + x_2 - 1)$$

The partial derivatives are:

$$\frac{\partial L(\mathbf{x}, \lambda)}{\partial x_1} = -2x_1 + \lambda = 0$$
$$\frac{\partial L(\mathbf{x}, \lambda)}{\partial x_2} = -2x_2 + \lambda = 0$$
$$\frac{\partial L(\mathbf{x}, \lambda)}{\partial \lambda} = x_1 + x_2 - 1 = 0$$

Solving the system of equations gives: $(x_1^*, x_2^*) = (0.5, 0.5)$ and the value for the Lagrange multiplier is: $\lambda = 1$.

Generalized Lagrangian: Primal problem

Given an optimization problem:

$$\begin{aligned} \text{minimize} \quad & f(\mathbf{w}) \\ \text{subject to} \quad & g_i(\mathbf{w}) \leq 0, i = 1, \dots, k \\ & h_i(\mathbf{w}) = 0, i = 1, \dots, m, \end{aligned}$$

the generalized Lagrangian is defined as:

$$\begin{aligned} L(\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta}) &= f(\mathbf{w}) + \sum_{i=1}^{k} \alpha_i g_i(\mathbf{w}) + \sum_{i=1}^{m} \beta_i h_i(\mathbf{w}) \\ &= f(\mathbf{w}) + \boldsymbol{\alpha}^T \mathbf{g}(\mathbf{w}) + \boldsymbol{\beta}^T \mathbf{h}(\mathbf{w}) \end{aligned}$$

This is called the **primal optimization problem**.

Active and inactive constraints

- Generalized Lagrangian:

$$L(\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = f(\mathbf{w}) + \sum_{i=1}^{k} \alpha_i g_i(\mathbf{w}) + \sum_{i=1}^{m} \beta_i h_i(\mathbf{w})$$

- Recall that the g constraints were inequality constraints: $g_i(\mathbf{w}) \leq 0$
- Those constraints for which $g_i(\mathbf{w}) = 0$ are called **active**
- Constraints with $g_i(\mathbf{w}) < 0$ are called **inactive**

Generalized Lagrangian: dual problem

The **Lagrangian dual problem** is defined as:

$$\begin{aligned} \text{maximize} \quad & \hat{L}(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \inf_{\mathbf{w}} L(\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta}) \\ \text{subject to} \quad & \boldsymbol{\alpha} \geq \mathbf{0} \end{aligned}$$

- $\inf$ stands for **infimum** that is the **greatest lower bound** of a set or a function.
- The value of the dual problem is upper bounded by the value of the primal.
- If the values of primal and dual are equal and $\mathbf{w}^*$ and $(\boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$ solve the primal and dual problems respectively, then $\alpha_i^* g_i(\mathbf{w}^*) = 0$, for $i = 1, \ldots, k$.
- The difference between the values of the primal and dual problems is called the **duality gap**.

Strong duality theorem

Theorem

Given a convex optimization problem:

$$\begin{aligned} \textit{minimize} \quad & f(\mathbf{w}) \\ \textit{subject to} \quad & g_i(\mathbf{w}) \leq 0, i = 1, \ldots, k \\ & h_i(\mathbf{w}) = 0, i = 1, \ldots, m, \end{aligned}$$

where the g_i and h_i are affine functions, then the duality gap is zero.

- This means that instead of the primal problem we can solve the dual problem.

Karush-Kuhn-Tucker (KKT) conditions

Given an optimization problem:

$$\begin{aligned} \text{minimize} \quad & f(\mathbf{w}) \\ \text{subject to} \quad & g_i(\mathbf{w}) \leq 0, i = 1, \dots, k \\ & h_i(\mathbf{w}) = 0, i = 1, \dots, m, \end{aligned}$$

where f is convex and g_i, h_i are affine, the necessary and sufficient conditions for a point $\mathbf{w}^*$ to be an optimum are the existence of $\boldsymbol{\alpha}^*$, $\boldsymbol{\beta}^*$ such that:

$$\begin{aligned} \frac{\partial L(\mathbf{w}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)}{\partial \mathbf{w}} &= \mathbf{0}, \\ \frac{\partial L(\mathbf{w}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)}{\partial \boldsymbol{\beta}} &= \mathbf{0}, \\ \alpha_i^* g_i(\mathbf{w}^*) = 0, i = 1, \dots, k, \\ g_i(\mathbf{w}^*) \leq 0, i = 1, \dots, k, \\ \alpha_i^* \geq 0, i = 1, \dots, k \end{aligned}$$

Remarks

- If some of the conditions are violated then the value of the primal problem is infinity, because the dual problem attempts to maximize the Lagrangian with respect to $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ and the problem is maximized by choosing arbitrarily large parameters.
- If the constraints are satisfied then, regardless of the values of dual variables, the value of the primal problem is $f(\mathbf{w})$
- The relations $\alpha_i^* g_i(\mathbf{w}^*) = 0$ are known as **KKT complementary conditions**. They imply that for active constraints $\alpha^* \geq 0$, whereas for inactive constraints $\alpha^* = 0$

Objective function for both hard and soft margin

- For hard margin:

$$\min_{\mathbf{w},b} \frac{1}{2}||\mathbf{w}||^2$$
$$\text{subject to } y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1, \text{ for all } i$$

- For soft margin:

$$\min_{\mathbf{w},b,\xi} \frac{1}{2}||\mathbf{w}||^2 + C\sum_i \xi_i$$
$$y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i, \quad \text{for all } i$$
$$\xi_i \geq 0, \quad \text{for all } i$$

Support vectors

- For the hard margin SVM, the constraints can be written as:

$$g_i(\mathbf{w}) = -y_i(\mathbf{w}^T \mathbf{x}_i + b) + 1 \leq 0$$

- There is one such constraint for each training item.
- According to KKT complementary conditions, $\alpha_i > 0$ only for those data points that have functional margin exactly 1, because for those $g_i(\mathbf{w}) = 0$.
- These data points are called the **support vectors**, because they lie exactly on the decision boundary and thus "support" it.

Lagrangian for SVM

- The Lagrangian for the hard margin SVM is:

$$L(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}||\mathbf{w}||^2 - \sum_{i=1}^{n} \alpha_i \left(y_i(\mathbf{w}^T \mathbf{x}_i + b) - 1 \right)$$

- Note that there are no β variables as there are only inequality constraints.
- Similarly, the Lagrangian for the soft margin SVM is:

$$L(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \frac{1}{2}||\mathbf{w}||^2 + C \sum_{i=1}^{n} \xi_i - \sum_{i=1}^{n} r_i \xi_i - \sum_{i=1}^{n} \alpha_i [y_i(\mathbf{w}^T \mathbf{x}_i + b) - 1 + \xi_i]$$

Dual for the SVM

- For finding the dual we first have to minimize the Lagrangian with respect to primal variables keeping dual variables fixed. We do that by taking partial derivatives and imposing stationarity.
- For the hard margin case we get:

$$\frac{\partial L(\mathbf{w}, b, \boldsymbol{\alpha})}{\partial \mathbf{w}} = \mathbf{w} - \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i = 0 \Longrightarrow \mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i$$

$$\frac{\partial L(\mathbf{w}, b, \boldsymbol{\alpha})}{\partial b} = -\sum_{i=1}^{n} \alpha_i y_i = 0$$

- Note that $\mathbf{w}$ is expressed as a **linear combination** of the input points.

Dual for the SVM

- Substituting $\mathbf{w}$ back to the Lagrangian we get:

$$L(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}||\mathbf{w}||^2 - \sum_{i=1}^{n} \alpha_i \left(y_i(\mathbf{w}^T \mathbf{x}_i + b) - 1\right)$$

$$= \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i \cdot \mathbf{x}_j \rangle - \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i \cdot \mathbf{x}_j \rangle$$

$$- b \sum_{i=1}^{n} \alpha_i y_i + \sum_{i=1}^{n} \alpha_i$$

- Considering that $\sum_{i=1}^{n} \alpha_i y_i = 0$ this can be simplified:

$$L(\mathbf{w}, b, \boldsymbol{\alpha}) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i \cdot \mathbf{x}_j \rangle$$

subject to $\quad \alpha_i \geq 0, i = 1, \ldots, n$

Dual for the SVM

- Similarly, the dual can be found for soft margin SVM, giving the result:

$$L(\mathbf{w}, b, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i \cdot \mathbf{x}_j \rangle$$

$$\text{subject to} \qquad C \geq \alpha \geq 0, i = 1, \ldots, n$$

- For the optimal value we have to maximize the dual, which is equivalent to minimizing the negative dual.
- Note that the training data points in dual problem never occur alone, but only in dot products. This leads us to the **kernels**.

Feature spaces

- Linear models can only learn linear decision boundaries.
- We can make a linear model to learn non-linear decision boundary by adding combinations of features as new features. For example for a data point (x_1, x_2) we can add features $x_1^2, x_1 x_2, x_2^2$.
- This is the same as to say that we are mapping the linearly non-separable data into the space of higher dimension and thus make it linearly separable.
- We define a **feature map** $\Phi(\cdot)$ that is the function that maps the input into the feature space and then use the resulting feature vectors as inputs in SVM.

Dot products and kernels

- Recall that the data points in SVM dual problem only occur in dot-products.
- This means that if our feature map produces high dimensional feature spaces then optimizing SVM is computationally prohibitive.
- However, we can use **kernel functions** K to induce the high-dimensional feature vectors implicitly and compute the dot product by using the original low-dimensional input vectors.
- This is called the **kernel trick** and it enables to use infinite-dimensional feature vectors without ever explicitly computing them.

Example: Polynomial kernel

- Suppose we have a data point $\mathbf{x} = (x_1, x_2, \ldots, x_d)$.
- And suppose we have a feature map that does a quadratic feature expansion, resulting in a feature vector:

$$\begin{aligned}\phi(\mathbf{x}) = (1, \sqrt{2}x_1, \sqrt{2}x_2, \ldots, \sqrt{2}x_d, \\ x_1^2, x_1x_2, \ldots, x_1x_d, \\ x_2x_1, x_2^2, \ldots, x_2x_d, \\ \ldots, \\ x_dx_1, x_dx_2, \ldots, x_d^2)\end{aligned}$$

- These feature vectors can be used to train a classifier.
- However, there are two problems:
  - computational: the number of necessary computations is now squared
  - statistical: we need (quadratically) more training data to avoid overfitting.

Example: polynomial kernel

- Consider that in the SVM dual problem we have to compute $\langle \phi(\mathbf{x}) \cdot \phi(\mathbf{z}) \rangle$ for some input data points $\mathbf{x}$ and $\mathbf{z}$.
- Let's do this!

$$
\begin{aligned}
\langle \phi(\mathbf{x}) \cdot \phi(\mathbf{z}) \rangle &= 1 + 2x_1z_1 + 2x_2z_2 + \ldots + 2x_dz_d \\
&+ x_1^2z_1^2 + \ldots + x_1x_dz_1z_d + \ldots \\
&+ x_dx_1z_dz_1 + x_dx_2z_dz_2 + \ldots + x_d^2z_d^2 \\
&= 1 + 2\sum_{i=1}^{d} x_iz_i + \sum_{i,j=1}^{d} x_ix_jz_iz_j \\
&= 1 + 2\langle \mathbf{x} \cdot \mathbf{z} \rangle + \langle \mathbf{x} \cdot \mathbf{z} \rangle^2 \\
&= (1 + \langle \mathbf{x} \cdot \mathbf{z} \rangle)^2
\end{aligned}
$$

Polynomial kernel

- It turns out that we can compute the dot product between the feature vectors implicitly by using the original input vectors only!
- In a similar fashion we can induce even more complex feature vectors by using the kernel function $K(\mathbf{x}, \mathbf{z}) = (1 + \langle \mathbf{x} \cdot \mathbf{z} \rangle)^3$ or $K(\mathbf{x}, \mathbf{z}) = (1 + \langle \mathbf{x} \cdot \mathbf{z} \rangle)^4$.
- In general, it is possible to use any polynomial of degree p, so that the kernel function has the form $K(\mathbf{x}, \mathbf{z}) = (r + \gamma \langle \mathbf{x} \cdot \mathbf{z} \rangle)^p$. This class of kernels are called **polynomial kernels**.

Designing kernels

- In case of the polynomial kernel we saw that it indeed implemented a dot product between the feature vectors.
- Do we always have to construct the feature vector and work out their dot products to define a kernel function?
- Or can we use any function as a kernel?
- A kernel function can be defined by using either of the following definitions:
  - $K(\cdot, \cdot)$ is a valid kernel, if it corresponds to the inner product between two vectors.
  - $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a kernel, if K is **positive semi-definite**. This condition is called the **Mercer's condition** and the kernels satisfying it are called **Mercer's kernels**.

Mercer's kernels

- More complicated kernels can be constructed from simple kernels
- It can be shown that if K_1 and K_2 are Mercer's kernels then so are these (not an exhaustive list):

$$K_1(\mathbf{x}, \mathbf{z}) + K_2(\mathbf{x}, \mathbf{z})$$
$$K_1(\mathbf{x}, \mathbf{z}), a \in \mathbb{R}$$
$$K_1(\mathbf{x}, \mathbf{z}) K_2(\mathbf{x}, \mathbf{z})$$
$$\exp K_1(\mathbf{x}, \mathbf{z})$$