CSE 546: Machine Learning **Lecture 7**

Feature construction: Boosting, Kernels, and Random Features...

Instructor: Sham Kakade

1 Boosted Decision Trees

The input is a training set:

$$\{(x_1, y_1), \dots (x_n, y_n)\}$$

1. Initialize: set the residual $r_i = y_i$ for all i and initialize our model so that $f(x) = 0$ (where x is the input vector).
2. (*"weak learning"*) find some feature $\psi_*(x)$, where this feature mapping takes x to a real number (e.g. a decision tree) which improves our residual error. Here the residual error of a feature ψ is defined as:

$$\hat{L}(\psi) = \frac{1}{n} \sum_i (r_i - \psi(x_i))^2$$

3. update the model and the residual error

$$\begin{aligned} f(x) &\leftarrow f(x) + \psi(x) \\ r_i &\leftarrow r_i - \psi(x_i) \end{aligned}$$

   and return to step 2 (or terminate based on some stopping criterion).

2 Kernel methods: The basic idea

The basic idea is to make new predictions based on a similarity measure to points in our dataset. Suppose we have a training set:

$$\{(x_1, y_1), \dots (x_n, y_n)\}$$

Let $K(x, x')$ be a a similarity measure between two points x and x'.

Given a new point x, we seek to make a prediction of y in the following form:

$$y = \sum_{i=1}^{n} \alpha_i K(x_i, x)$$

where x_i are points in our training set and $\alpha = (\alpha_1 \dots \alpha_n)$ is our weight vector. Note that the dimension of our weight vector is now n. Also, here, x need not be a vector (it could be some arbitrary object).

3 Kernels

Let us instead make a feature vector out of a point $x \in \mathcal{X}$ (where $\mathcal{X}$ is our input space) through a function ϕ which maps x to some high dimensional space, where $\phi : \mathcal{X} \leftarrow \mathbb{R}^{d'}$ (often d may be much greater than the sample size n).

A *kernel* is an inner product mapping where:

$$K(x, x') := \phi(x)^\top \phi(x')$$

In other words, the kernel just specifies the inner product under some feature mapping ϕ.

Sometimes we specify the kernel without explicitly defining a function ϕ. In particular *Mercer's theorem*, states conditions under which a function $K(x, x')$ is a valid Kernel. In particular $K(x, x')$ is a valid kernel (*i.e.* there exists a corresponding ϕ so that $K(x, x') := \phi(x)^\top \phi(x')$) if and only if K is positive semidefinite in the following sense: for all points $x_1, \ldots x_l$, the matrix D whose i, j-th coordinate in $K(x_i, x_j)$ is a positive semidefinite matrix.

3.1 Examples

Suppose x and x' are d-dimensional vectors. Let us consider the following Kernel:

$$K(x, x') = (x^\top x')^2$$

Here, we have that:

$$K(x, x') = (\sum_{j=1}^{d} x_j x'_j)^2 = \sum_j x_j^2 (x'_j)^2 + 2 \sum_{j<k} x_j x_k x'_j x'_k$$

Hence, we see that the feature map is just:

$$\phi(x) = (x_1^2, x_2^2, \ldots x_d^2, \sqrt{2} x_1 x_2, \sqrt{2} x_1 x_3, \ldots \sqrt{2} x_1 x_d, \sqrt{2} x_2 x_3, \sqrt{2} x_2 x_4, \ldots)$$

This proves that K is indeed a kernel. Also, we see that K is a kernel corresponding to exactly a degree two polynomial.

A similar argument show that:

$$K(x, x') = (x^\top x')^k$$

is a kernel corresponding do exactly a degree k polynomial. Also, one can see that:

$$K(x, x') = (x^\top x' + c)^k$$

(where c is a constant) is a kernel corresponding a polynomial containing terms of degree k or less.

A kernel which often works well in practice is the *radial basis kernel*, which is defined as follows:

$$K(x, x') = \exp(-\frac{\|x - x'\|^2}{2\sigma^2})$$

One can explicitly prove that this is a valid kernel (though the dimension of the corresponding feature map ϕ is note finite).

4 Kernel Regression

In the case of Kernel regression, let us suppose we want to fit the line:

$$w^\top \phi(x)$$

to our data. Here, ϕ is the feature mapping corresponding to the kernel K.

In particular, we could consider fitting the weights with ridge regression:

$$\hat{w} = \arg\min_w \frac{1}{n}\sum_i (y_i - w^\top \phi(x_i))^2 + \lambda \|w\|^2$$

One can show that this best fit line is:

$$\hat{w}^\top \phi(x) = \sum_i \hat{\alpha}_i K(x_i, x)$$

where:

$$\hat{\alpha} = D(D + \lambda \mathrm{I}_n)^{-1} Y$$

where D is the $n \times n$ matrix in which:

$$D_{j,k} = K(x_j, x_k)$$

4.1 An Equivalent "dual" viewpoint

Equivalently, the following formulation will result in the same mapping. Note that $(\sum_j \alpha_j K(x_j, x_i))$ is our prediction of the point y_i. We can find an estimate of α as follows:

$$\hat{\alpha} = \arg\min_\alpha \frac{1}{n}\sum_i \left(y_i - (\sum_j \hat{\alpha}_j K(x_j, x_i)) \right)^2 + \lambda \alpha^\top D \alpha$$

where $\alpha^\top D \alpha$ is our regularizer. This choice of a regularizer is natural (as it will give rise to the same solution had we worked with ϕ).

If we solve for the above (and rearrange the expression for the solution), we obtain that:

$$\hat{\alpha} = D(D + \lambda \mathrm{I}_n)^{-1} Y$$

which is precisely what we obtained in the "primal" problem.

5 Random Features

For the radial basis function, a natural way to approximate this function is as follows. First sample vectors: $v_1, v_2, \ldots v_l$ where each $v_i \in \mathbb{R}^d$ and sampled from a $N(0, \frac{1}{\sigma_2}\mathrm{I}_d)$. Then construct the feature vector:

$$\phi(x) = (\cos(v_1^\top x), \sin(v_1^\top x), \cos(v_2^\top x), \sin(v_2^\top x), \ldots \cos(v_l^\top x), \sin(v_l^\top x))$$

For large enough l, one can show that this well approximates the radial basis function.

5.1 A little intuition for the construction

The intuition for this mapping is as follows: Let's look at the two vectors $(\cos v_1^\top x, \sin v_1^\top x)$ and $(\cos v_1^\top x', \sin v_1^\top x')$ (which is part of our random features). Using that v_1 is sampled from a normal distribution, we have that:

$$E[(\cos v_1^\top x, \mathrm{i} \sin v_1^\top x)^\top (\cos v_1^\top x', \mathrm{i} \sin v_1^\top x')] = K(x, x')$$

where the K above is the radial basis function, the expectation with respect to v_1, and i is an imaginary number. So we see that, in expectation, the above feature mapping is correct. Furthermore, with an appropriate choice of l, it will be the case that our feature vectors approximates the correct inner products on all relevant points (through a law of large numbers argument).