Lecture 13 | EE 559 | Feb 26, 2024

Machine Learning I: Supervised Methods

B. Keith Jenkins

Announcements

- Slido event code : 1345352
- Homework 5 is due Friday
- Sample exam problems
  - Discussion 7
  - Discussion 8
  - Additional problems will be posted on D2L

Today's lecture

- Lagrangian optimization (2)
  - Inequality constraints
- Kernels for nonlinear tranformations
  - Examples of kernels
  - Kernel substitution
  - Valid or Mercer kernels

Lagrange Optimization with 1 Inequality Constraint

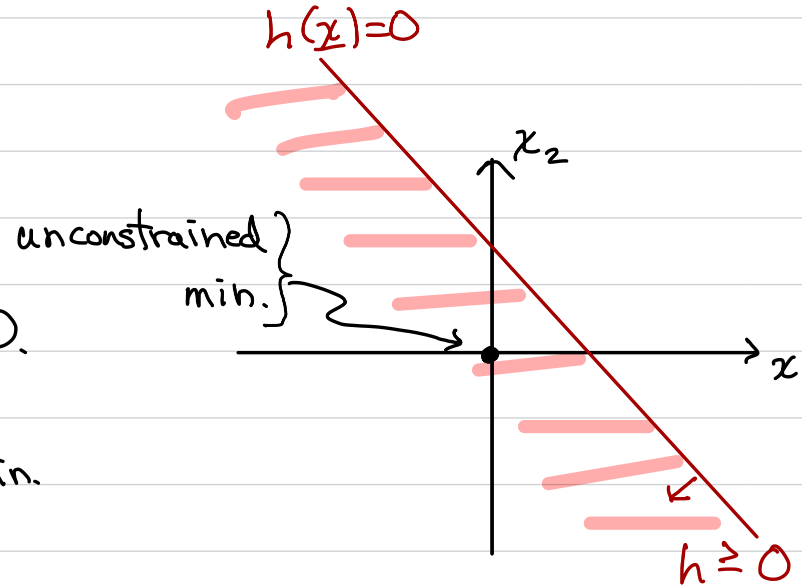
Extension of the equality constraint case.

Find min. of $f(\underline{x})$ s.t. $h(\underline{x}) \geq 0$

Let:

$$L(\underline{x}, \mu) = f(\underline{x}) - \mu h(\underline{x})$$

Find extremum of L over $\underline{x}$ and μ.



2 cases:

(a) unconstrained min. is in region $h \geq 0$.

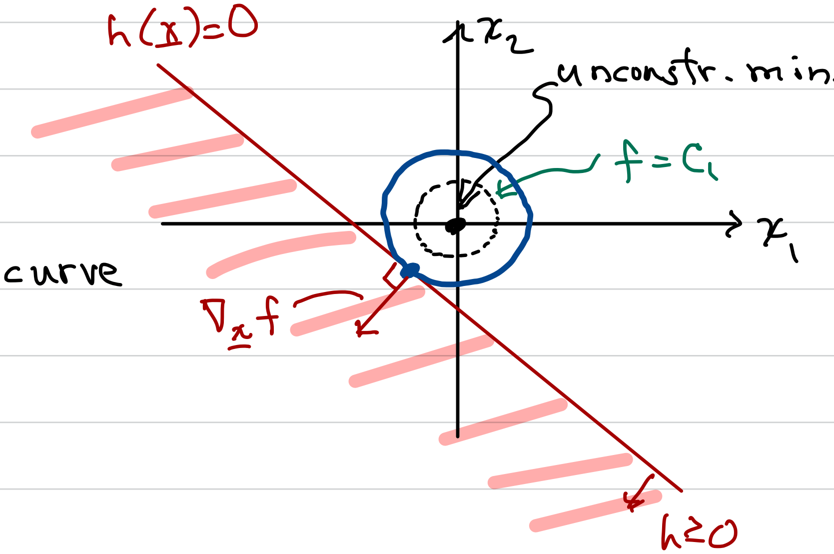
$\Rightarrow$ constrained min. = unconstrained min.

Use $\nabla_{\underline{x}} f(\underline{x}) = \underline{0}$

$\Rightarrow$ can solve using:

$L(\underline{x}, \mu)$ with $\mu = 0$.

(b) unconstrained min. is outside $h \geq 0$ region.



Ⓢ Constrained min. is <u>on</u> $h(\underline{x})=0$ curve

$\Rightarrow$ same as equality constraint case

$L(\underline{x}, \mu)$ with $\mu \neq 0$

$\nabla_{\underline{x}, \mu} L = \underline{0}$

$$\Rightarrow \nabla_{\underline{x}} f = \mu \nabla_{\underline{x}} h, \qquad \mu > 0$$

$\therefore$ $L(\underline{x}, \mu) = f(\underline{x}) - \mu h(\underline{x})$, $\quad h \geq 0$ (constraint)

In case (a): $\mu = 0$, $h(\underline{x}^*) > 0$.

In case (b): $\mu > 0$, $h(\underline{x}^*) = 0$.

$\Rightarrow$ in both cases, $\mu h(\underline{x}^*) = 0$.

Summary of 1 ineq. constr.

Min. $f(\underline{x})$ s.t. $h(\underline{x}) \geq 0$

$$L(\underline{x}, \mu) = f(\underline{x}) - \mu h(\underline{x})$$

$$\nabla_{\underline{x}, \mu} L(\underline{x}, \mu) = \underline{0}$$

$$\left.\begin{aligned} h(\underline{x}^*) &\geq 0 \\ \mu &\geq 0 \\ \mu h(\underline{x}^*) &= 0 \end{aligned}\right\}$$ Karush-Kuhn-Tucker (KKT) conditions

General case: multiple equality and multiple inequality constraints

Min. $f(\underline{x})$ s.t. $g_i(\underline{x}) = 0 \ \forall i, \quad h_j(\underline{x}) \geq 0 \ \forall j$

$$L(\underline{x}, \underline{\lambda}, \underline{\mu}) = f(\underline{x}) + \sum_{i=1}^{R} \lambda_i g_i(\underline{x}) - \sum_{j=1}^{R'} \mu_j h_j(\underline{x})$$

$$\nabla_{\underline{x}, \underline{\lambda}, \underline{\mu}} L(\underline{x}, \underline{\lambda}, \underline{\mu}) = \underline{0}.$$

Require: $\mu_j \geq 0 \ \forall j$
$\mu_j h_j(\underline{x}^*) = 0 \ \forall j$
Also given: $g_i(\underline{x}^*) = 0 \ \forall i$
$h_j(\underline{x}^*) \geq 0 \ \forall j$

KKT conditions

Kernels

→ An alternate way of including nonlinear mappings $\underline{u} = \underline{\phi}(\underline{x})$.

μ vs. $\underline{u}$

Ex: Nearest-means classifier.
Stores class means μ_k, $k=1,2,\cdots,C$.
Predicts $\hat{y}(\underline{x})$ by taking $\underset{k}{\operatorname{argmin}} \|\underline{x} - \mu_k\|_2^2$.

Let $\tilde{g}_k(\underline{x}) = -\|\underline{x} - \mu_k\|_2^2$ ← function that measures similarity between $\underline{x}$ and μ_k.

Simplify ⇒

$$\tilde{g}_k(\underline{x}) = -(\underbrace{\underline{x}^T\underline{x}}_{\text{Const. of } k} - 2\mu_k^T\underline{x} + \mu_k^T\mu_k)$$

$$\Rightarrow \text{Let } g_k(\underline{x}) = 2\mu_k^T\underline{x} - \mu_k^T\mu_k$$

What if we want to implement nearest-means classifier in $\underline{u}$-space (after a nonlinear mapping $\underline{u} = \underline{\phi}(\underline{x})$) ?

Tips: μ_k is a constant for a given dataset
$\underline{x}_i$, $\underline{x}_\ell$ are datapoints (given with the dataset)
$\underline{x}$ is a (vector) variable in the original feature space
$\underline{u}$, $\underline{\phi}$ are (vector) variables in the expanded feature space

2 approaches to include a nonlinear mapping $\underline{\phi}(\underline{x})$:

1. Specify $\underline{\phi}(\underline{x})$ explicitly. Nearest-means example: $(D=2)$

(1) Quadratic polynomial mapping $\underline{\phi}(\underline{x}) = [1, x_1, x_2, x_1^2, x_1 x_2, x_2^2]^T$.

(2) Then: $g_k(\underline{x}) = -\underline{\mu}_k^T \underline{\mu}_k + 2\underline{\mu}_k^T \underline{x}$

(3) $$= \frac{-1}{N_k}\left(\sum_{i=1}^{N_k} \underline{x}_i^{(k)T}\right)\frac{1}{N_k}\left(\sum_{\ell=1}^{N_k} \underline{x}_\ell^{(k)}\right) + 2\frac{1}{N_k}\left(\sum_{i=1}^{N_k} \underline{x}_i^{(k)T}\right)\underline{x}$$

$\underline{\phi}(\underline{x})$ $\quad \underline{x} \rightarrow \underline{\phi}(\underline{x}),\ \ \underline{x}_i^{(k)} \rightarrow \underline{\phi}(\underline{x}_i^{(k)})$, etc.

(4) $$g_k(\underline{x}) = -\frac{1}{N_k^2}\sum_{i=1}^{N_k}\sum_{\ell=1}^{N_k} \underline{\phi}^T(\underline{x}_i^{(k)})\,\underline{\phi}(\underline{x}_\ell^{(k)}) + \frac{2}{N_k}\sum_{i=1}^{N_k} \underline{\phi}^T(\underline{x}_i^{(k)})\,\underline{\phi}(\underline{x})$$

Can then plug in for $\underline{\phi}$ everywhere and simplify, or compute numerically directly from (4) and (1).

Ⓢ

For approach 2, first write $g_k(\underline{x})$ (or $J(\underline{w})$) as a function of inner products of $\underline{x}$ only ($\underline{x}^T\underline{x}_i$, $\underline{x}_i^T\underline{x}_\ell$, etc.), if possible:

(5) From (3), $g_k(\underline{x}) = -\frac{1}{N_k^2}\sum_{i=1}^{N_k}\sum_{\ell=1}^{N_k} \underline{x}_i^{(k)T}\underline{x}_\ell^{(k)} + \frac{2}{N_k}\sum_{i=1}^{N_k}\underline{x}_i^{(k)T}\underline{x}$

(In Lagrange optimization problems, we will get this form by deriving a dual representation.)

Then use kernels.

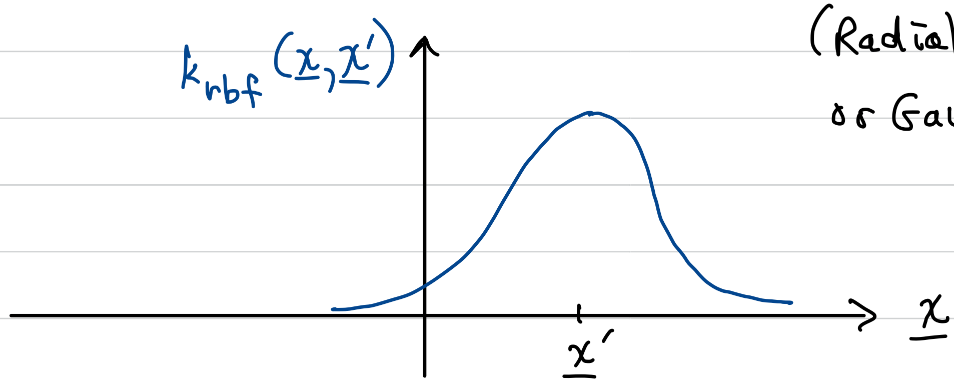
Kernels are functions that help to measure similarity between 2 input vectors, call them $\underline{x}$ and $\underline{x}'$; and/or, add a nonlinear mapping to the system.

Ex:

(i) $k_{linear}(\underline{x}, \underline{x}') = \underline{x}^T\underline{x}'$ (linear kernel).

(ii) $k_{poly}(\underline{x}, \underline{x}') = (1 + \underline{x}^T\underline{x}')^d$ (polynomial kernel)

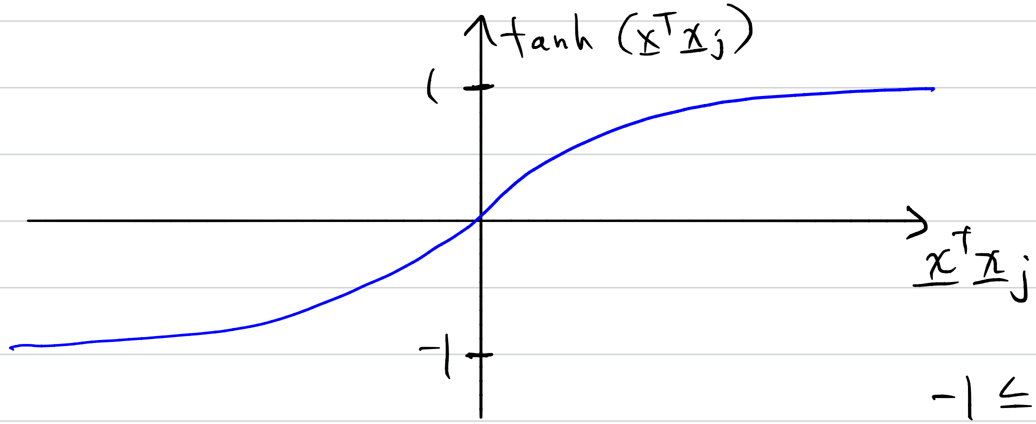
(iii) $k_{RBF}(\underline{x}, \underline{x}') = \exp\{-\gamma \|\underline{x} - \underline{x}'\|_2^2\}$, $\gamma > 0$



(Radial Basis Function (RBF) or Gaussian kernel)

(iv) $k_{sigmoid}(\underline{x}, \underline{x}_j) = \tanh\left[(\underline{x}^T\underline{x}_j)\alpha + c\right]$ (Sigmoid kernel)

Interpret: Let $\alpha = 1,\ c = 0 \implies k_{sigm} = \tanh(\underline{x}^T\underline{x}_j)$



$-1 \le k_{Sigmoid} \le +1$ always.

$k_{sigmoid}$ is also a similarity measure.

Often kernels are chosen so that $k(\underline{x},\underline{x}') = \underline{\phi}^T(\underline{x})\,\underline{\phi}(\underline{x}')$
(k is an inner product in $\underline{u}$-space).

2. Specify $k(\underline{x}, \underline{x}')$ explicitly, which defines the nonlinear mapping implicitly.

Ex: RBF kernel: $k_{RBF}(\underline{x}, \underline{x}') = \exp\{-\gamma \|\underline{x} - \underline{x}'\|_2^2\}$.

Once $g_k(\underline{x})$ or $J(\underline{w})$ is written as a function of $\underline{x}^T \underline{x}'$ (e.g., $\underline{x}^T \underline{x}_n$),

substitute: $\boxed{\underline{x}^T \underline{x}' \rightarrow k(\underline{x}, \underline{x}')}$

to implement the nonlinear mapping.

Nearest-means example: We had

(5) $$g_k(\underline{x}) = -\frac{1}{N_k^2} \sum_{i=1}^{N_k} \sum_{l=1}^{N_k} \underline{x}_i^{(k)T} \underline{x}_l^{(k)} + \frac{2}{N_k} \sum_{i=1}^{N_k} \underline{x}_i^{(k)T} \underline{x}$$

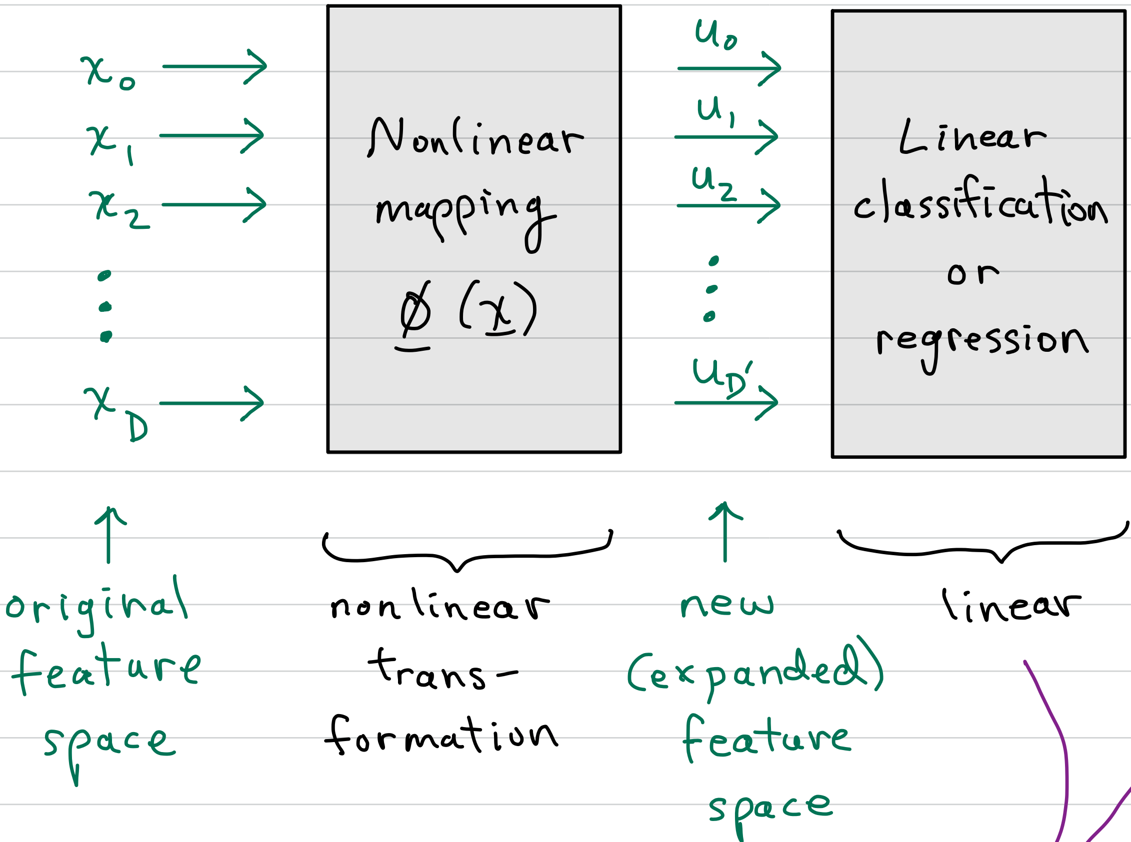
$\rightarrow$ Nonlinear mapping $\underline{u} = \underline{\phi}(\underline{x}) \longrightarrow$

$$g_k(\underline{u}) = -\frac{1}{N_k^2} \sum_{i=1}^{N_k} \sum_{l=1}^{N_k} \underbrace{k(\underline{x}_i^{(k)}, \underline{x}_l^{(k)})} + \frac{2}{N_k} \sum_{i=1}^{N_k} \underbrace{k(\underline{x}_i^{(k)}, \underline{x})}$$

For RBF kernel use $k_{RBF}(\underline{x}, \underline{x}') = \exp\{-\gamma \|\underline{x} - \underline{x}'\|_2^2\}$

(6) $$g_k(\underline{u}) = -\frac{1}{N_k^2} \sum_{i=1}^{N_k} \sum_{l=1}^{N_k} \exp\{-\gamma \|\underline{x}_i^{(k)} - \underline{x}_l^{(k)}\|_2^2\} + \frac{2}{N_k} \sum_{i=1}^{N_k} \exp\{-\gamma \|\underline{x} - \underline{x}_i^{(k)}\|_2^2\}$$

<u>But</u> our nonlinear-transformation approach was [Lecture 11, p. 3]:



Classification:
$g(\underline{u}) = \underline{w}_k'^T \underline{u}$

→ Linear in $\underline{u}$

→ Linear in $\underline{w}_k'$

The linear classifier or regressor should be the same as original classifier or regressor, except with $\underline{x} \to \underline{u}$ or $\underline{x} \to \underline{\phi}(\underline{x})$.

Is that true here? Compare (6) with (2), (4) for nearest-means example.

Because we substituted $\underline{x}^T\underline{x}' \to k(\underline{x},\underline{x}')$ to map $\underline{x} \to \underline{u} = \underline{\phi}(\underline{x})$, we want $k(\underline{x},\underline{x}') = \underline{\phi}^T(\underline{x})\,\underline{\phi}(\underline{x}')$ for some N.L. mapping $\underline{\phi}$.

How do we know if our choice of $k(\underline{x},\underline{x}')$ can be expressed this way?

→ If $k(\underline{x},\underline{x}')$ is a <u>valid kernel</u>, then yes, it can.

Def: Gram matrix $\underline{\underline{K}}$:

$$\underline{\underline{K}} = \begin{bmatrix} k(\underline{x}_1,\underline{x}_1) & \cdots & k(\underline{x}_1,\underline{x}_N) \\ \vdots & \ddots & \vdots \\ k(\underline{x}_N,\underline{x}_1) & \cdots & k(\underline{x}_N,\underline{x}_N) \end{bmatrix}$$

If the Gram matrix $\underline{\underline{K}}$ is positive definite for all sets of input vectors $\{\underline{x}_i\}_{i=1}^N$, then $k(\underline{x},\underline{x}')$ is a <u>valid kernel</u> (also called <u>Mercer kernel</u> or <u>positive-definite kernel</u>).

How to prove $k(\underline{x},\underline{x}')$ is a valid kernel, for a given $k(\underline{x},\underline{x}')$?
More commonly, for a given (possible) kernel $k(\underline{x},\underline{x}')$, rather than proving $\underline{\underline{K}}$ is positive definite, one can use known valid kernels $k'(\underline{x},\underline{x}')$, and properties to build up a new valid kernel from known valid kernels $k'(\underline{x},\underline{x}')$ [Bishop 6.2, "Constructing Kernels"].

Ex:
The RBF (Gaussian) kernel $k(\underline{x},\underline{x}') = \exp\{-\gamma \|\underline{x}-\underline{x}'\|_2^2\}$ can be shown to be a valid kernel, by building it out of $k'(\underline{x},\underline{x}') = \underline{x}^T\underline{x}'$ using known properties [Bishop 6.2].

So, in our earlier ($\underline{u}$-space) equation:

(6) $$g_k(\underline{u}) = \frac{-1}{N_k^2}\sum_{i=1}^{N_k}\sum_{\ell=1}^{N_k} \underbrace{\exp\{-\gamma\|\underline{x}_i^{(k)}-\underline{x}_\ell^{(k)}\|_2^2\}}_{\substack{k(\underline{x}_i^{(k)},\underline{x}_\ell^{(k)}) \\ = \underline{\phi}^T(\underline{x}_i^{(k)})\,\underline{\phi}(\underline{x}_\ell^{(k)})}} + \frac{2}{N_k}\sum_{i=1}^{N_k} \underbrace{\exp\{-\gamma\|\underline{x}-\underline{x}_i^{(k)}\|_2^2\}}_{\substack{\underline{k}(\underline{x},\underline{x}_i^{(k)}) \\ = \underline{\phi}^T(\underline{x})\,\underline{\phi}(\underline{x}_i^{(k)})}}$$

for some mapping $\underline{\phi}(\underline{x})$ that exists but hasn't been stated.

Comments:

1. Implementing a N.L. mapping $\underline{\phi}(\underline{x})$ by substituting:

$$\underline{x}^T\underline{x}' \longrightarrow k(\underline{x},\underline{x}')$$

is known as the <u>kernel trick</u> or <u>kernel substitution</u>.

2. Why bother?

(i) Sometimes we know a good $k(\underline{x},\underline{x}')$, but don't know the $\underline{\phi}(\underline{x})$.

<u>Ex</u>: $k_{rbf}(\underline{x},\underline{x}') = \exp\{-\gamma\|\underline{x}-\underline{x}'\|_2^2\}$ can be a good similarity fcn.

(ii) Sometimes $\underline{\phi}(\underline{x})$ is difficult to work with.

<u>Ex</u>: $\underline{\phi}(\underline{x})$ corresponding to $k_{rbf}(\underline{x},\underline{x}')$ is infinite dimensional!