Recitation 5

Kernels

DS-GA 1003 Machine Learning

Spring 2021

March 2, 2021

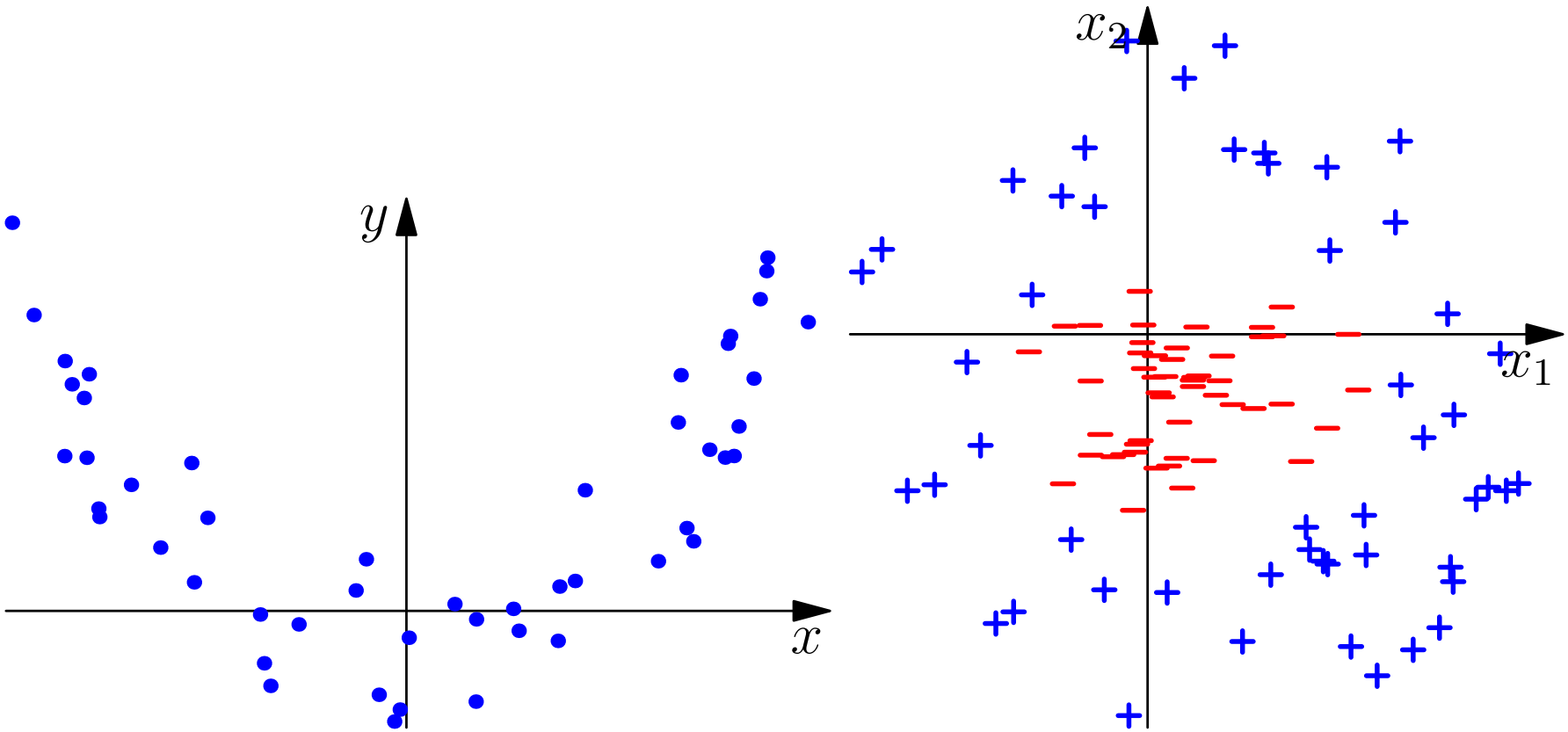
Agenda

- Intro question
- Cost of feature map
- Kernel
- Incorporating kernels into ridge regression and SVM
- RBF Kernel
- Coding exercise: revisit MNIST with RBF kernel

Intro Question

Question

Consider applying linear regression to the data set on the left, and an SVM to the data set on the right. What is the issue? Can it be improved?



Intro Solution

Regression Solution

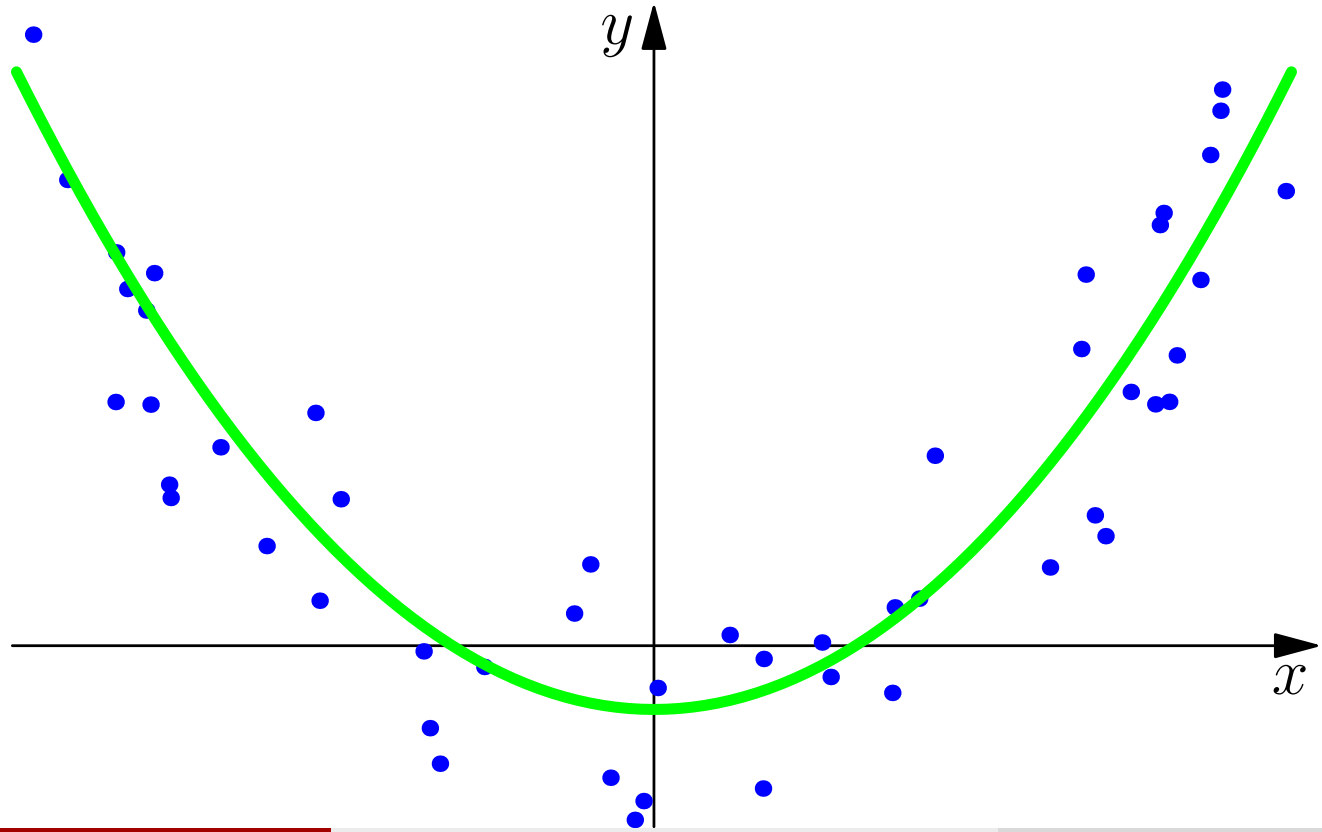
We want to allow for non-linear regression functions, but we would like to reuse the same fitting procedures we have already developed. To do this we will expand our feature set by adding non-linear functions of old features. We change our features from $(1, x)$ to $(1, x, x^2)$. That is

$$X = \begin{pmatrix} 1 & -1 \\ 1 & -.7 \\ \vdots & \vdots \\ 1 & 1 \end{pmatrix} \implies \Phi = \begin{pmatrix} 1 & -1 & (-1)^2 \\ 1 & -.7 & (-.7)^2 \\ \vdots & \vdots & \vdots \\ 1 & 1 & 1^2 \end{pmatrix}.$$

Intro Solution

Regression Solution

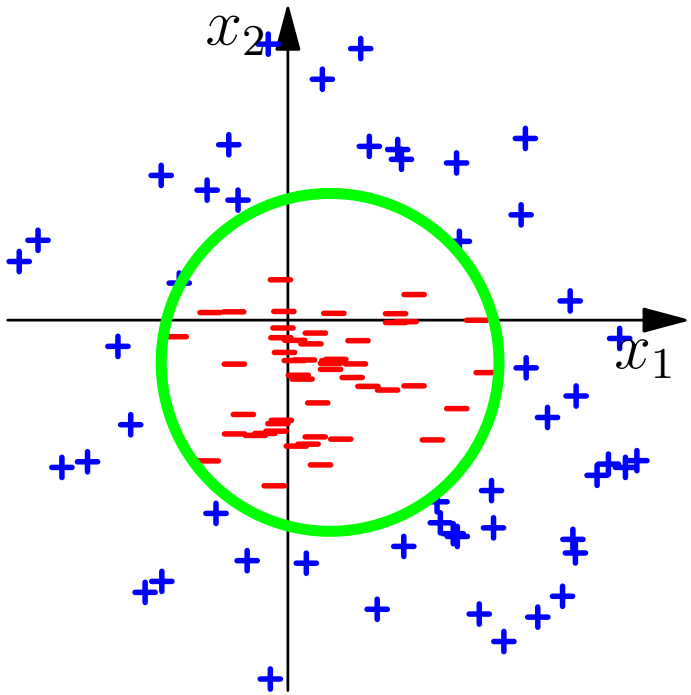
Using features $(1, x, x^2)$ and $w = (-.1, 0, 1)$ gives us $f_w(x) = -.1 + 0x + 1x^2 = x^2 - .1$. Our prediction function is quadratic but we obtained it through standard linear methods.



Intro Solution

SVM Solution

For the SVM we expand our feature vector from $(1, x_1, x_2)$ to $(1, x_1, x_2, x_1x_2, x_1^2, x_2^2)$. Using $w = (-1.875, 2.5, -2.5, 0, 1, 1)$ gives $-1.875 + 2.5x_1 - 2.5x_2 + x_1^2 + x_2^2 = (x_1 + 1.25)^2 + (x_2 - 1.25)^2 - 5 = 0$ as our decision boundary.



Feature Mapping

- In both cases, we find out that we are not able to construct accurate linear models on the original input space $\mathcal{X}$.
- We heuristically form **feature map** $\varphi(x) : \mathcal{X} \mapsto \mathcal{Z}$ that maps an input $x \in \mathcal{X}$ from the input space $\mathcal{X}$ to a feature space $\mathcal{Z}$.
  - For ridge regression, $\varphi(1, x) = [1, x, x^2]$.
  - For SVM, $\varphi(1, x_1, x_2) = [1, x_1, x_2, x_1 x_2, x_1^2, x_2^2]$.
- We then apply ridge regression / SVM on the feature space $\mathcal{Z}$.

Question

While we obtain stronger representation power using the feature map $\varphi(x)$, it comes with a cost. How to quantify this cost?

Cost of Adding Features

Question

Suppose we begin with d-dimensional inputs $x = (x_1, \ldots, x_d)$. We add all monomial features up to degree M. More precisely, all terms of the form $x_1^{p_1} \cdots x_d^{p_d}$ where $p_i \geq 0$ and $p_1 + \cdots + p_d \leq M$. How many features will we have in total?

- In our SVM example, we begin with $d = 2$ dimensions $\mathrm{x} = (x_1, x_2)$.
- Our feature is $\mathrm{z} = (1, x_1, x_2, x_1x_2, x_1^2, x_2^2)$.
- Each **monomial feature** z_i can be expressed using $x_1^{p_1} x_2^{p_2}$.
- For instance, $1 = x_1^0 x_2^0$ so $p_1 = p_2 = 0$ and $x_1^2 = x_1^2 x_2^0$ so $p_1 = 2, p_2 = 0$.
- Important observation: both d and M determines the total number of features. d controls how many dimensions we start with and M controls how "complex" our resulting features are.

Cost of Adding Features

Question

Suppose we begin with d-dimensional inputs $x = (x_1, \ldots, x_d)$. We add all monomial features up to degree M. More precisely, all terms of the form $x_1^{p_1} \cdots x_d^{p_d}$ where $p_i \geq 0$ and $p_1 + \cdots + p_d \leq M$. How many features will we have in total?

Solution

There will be $\binom{M+d}{M}$ terms total. If M is fixed and we let d grow, this behaves like $\frac{d^M}{M!}$. For example, if $d = 40$ and $M = 8$ we get $\binom{40+8}{8} = 377348994$. If we are training or predicting with a linear model $w^T x$, this product now takes $O(d^M)$ operations to evaluate.

Cost of Adding Features

- If we stick with polynomial features up to order M, it's takes exponential time $O(d^M)$ to compute all features.
- Can we avoid this computational cost while still building ridge regression / SVM models that takes all polynomial features into account?

The Kernel Function

Definition (Kernel)

Given a feautre map $\varphi(x) : \mathcal{X} \mapsto \mathcal{Z}$, the **kernel function** corresponding to $\varphi(x)$ is

$$k(x, x^{'}) = \langle \varphi(x), \varphi(x^{'}) \rangle$$

where $\langle \cdot, \cdot \rangle$ is an inner product operator.

- So a kernel function computes the inner product of applying the feature map $\varphi(x)$ for two inputs $x, x' \in \mathcal{X}$.
- Why kernel can help us avoid computational cost?
- How can we integrate kernel into ridge regression / SVM?

Efficiency of Kernel

Consider the polynomial kernel $k(x, y) = \langle \varphi(x), \varphi(y) \rangle = (1 + x^T y)^M$ where $x, y \in \mathbb{R}^d$. For example, if $M = 2$ we have

$$\begin{aligned} (1 + x^T y)^2 &= 1 + 2x^T y + x^T y x^T y \\ &= 1 + 2\sum_{i=1}^{d} x_i y_i + \sum_{i,j=1}^{d} x_i y_i x_j y_j. \end{aligned}$$

Option 1: First explicitly evaluate $\varphi(x)$ and $\varphi(y)$, and then compute $\langle \varphi(x), \varphi(y) \rangle$.

- $\varphi(x) = (1, \sqrt{2}x_1, \ldots, \sqrt{2}x_d, x_1^2, \ldots, x_d^2, \sqrt{2}x_1x_2, \sqrt{2}x_1x_3, \ldots, \sqrt{2}x_{d-1}x_d)$
- Takes $O(d^M)$ times to evaluate $\varphi(x)$ and $\varphi(y)$.
- Takes another $O(d^M)$ times to compute the inner product.
- Time complexity is $O(d^M)$.

Efficiency of Kernel

Consider the polynomial kernel $k(x, y) = \langle \varphi(x), \varphi(y) \rangle = (1 + x^T y)^M$ where $x, y \in \mathbb{R}^d$. This computes the inner product of all monomials up to degree M in time $O(d)$. For example, if $M = 2$ we have

$$\begin{aligned} (1 + x^T y)^2 &= 1 + 2x^T y + x^T y x^T y \\ &= 1 + 2\sum_{i=1}^{d} x_i y_i + \sum_{i,j=1}^{d} x_i y_i x_j y_j. \end{aligned}$$

Option 2: First calculate $1 + x^T y$, then calculate $(1 + x^T y)^M$.

- Takes $O(d)$ time to evaluate $1 + x^T y$.
- Takes $O(1)$ time to calculate $(1 + x^T y)^M$
- Time complexity is $O(d)$

Kernel for SVM

- Directly calculating the kernel is much more computationally efficient than explicitly expressing $\varphi(x)$ and evaluating the inner product.
- But how can we make use of the kernel to support ridge regression / SVM?
- To answer this question, we need first to understand the **Representer Theorem**.

"In sovjet rashiya, machine vector supports you."

Representer Theorem (Baby Version)

Theorem ((Baby) Representer Theorem)

Suppose you have a loss function of the form

$$J(w) = L(w^T\varphi(x_1), \ldots, w^T\varphi(x_n)) + R(\|w\|_2)$$

where

- $x_i \in \mathbb{R}^d, w \in \mathbb{R}^D, \varphi(x) : \mathbb{R}^d \mapsto \mathbb{R}^D$.
- $L : \mathbb{R}^n \to \mathbb{R}$ *is an arbitrary function (loss term).*
- $R : \mathbb{R}_{\geq 0} \to \mathbb{R}$ *is increasing (regularization term).*

Assume J has at least one minimizer. Then J has a minimizer w^* of the form $w^* = \sum_{i=1}^n \alpha_i \varphi(x_i)$ for some $\alpha \in \mathbb{R}^n$. If R is strictly increasing, then all minimizers have this form.

Representer Theorem: Proof

Proof.

- Let $w^* \in \mathbb{R}^D$ and let $S = \text{Span}(\varphi(x_1), \ldots, \varphi(x_n))$.
- $w^* = \sum_{i=1}^n \alpha_i \varphi(x_i)$ indicates that w^* lies in S.
- Let's first suppose w^* **does not** lie in S. Then we can write $w^* = u + v$ where $u \in S$ and $v \in S^\perp$. Here u is the orthogonal projection of w^* onto S, and $S^\perp$ is the subspace of all vectors orthogonal to S.
- Then $(w^*)^T \varphi(x_i) = (u+v)^T \varphi(x_i) = u^T \varphi(x_i) + v^T \varphi(x_i) = u^T \varphi(x_i)$. So the prediction only depends on $u^T \varphi(x_i)$.
- But $\|w^*\|_2^2 = \|u+v\|_2^2 = \|u\|_2^2 + \|v\|_2^2 + 2u^T v = \|u\|_2^2 + \|v\|_2^2 \geq \|u\|_2^2$.
- Thus $R(\|w^*\|_2) \geq R(\|u\|_2)$ showing $J(w^*) \geq J(u)$.
- Above we showed that $\|u+v\|_2^2 = \|u\|_2^2 + \|v\|_2^2$ when $u^T v = 0$. This is called the Pythagorean theorem.

Representer Theorem: Meaning

- If your loss function only depends on w via its inner products with the inputs, and the regularization is an increasing function of the ℓ_2 norm, then we can write w^* as a linear combination of the training data.
- This applies to ridge regression and SVM.

Question

Suppose you have $n = 100$ samples, $d = 40$ features, and $M = 8$ degree monomial terms giving 377348994 features. This implies $w \in \mathbb{R}^{377348994}$ for ridge regression. What does the representer theorem say?

Representer Theorem: Meaning

- If your loss function only depends on w via its inner products with the inputs, and the regularization is an increasing function of the ℓ_2 norm, then we can write w^* as a linear combination of the training data.
- This applies to ridge regression and SVM.

Question

Suppose you have $n = 100$ samples, $d = 40$ features, and $M = 8$ degree monomial terms giving 377348994 features. This implies $w \in \mathbb{R}^{377348994}$ for ridge regression. What does the representer theorem say?

Solution

As $y \in \mathbb{R}^n$ varies, the solution w must lie in a 100-dimensional subspace of $\mathbb{R}^{377348994}$.

Recap

- We want ridge regression / SVM to have the capacity of incorporating non-linear features.
- We can achieve this by designing feature map $\varphi(x)$, but explicit evaluating $\varphi(x)$ is sometimes computationally infeasible.
- We found that while explicit evaluating $\varphi(x)$ is expansive, evaluating its inner product $k(x, x') = \langle \varphi(x), \varphi(x') \rangle$ on two data points are fairly cheap.
- Representer Theorem tells us that with some reasonable assumptions, the optimal set of parameters w^* for ridge regression / SVM's loss function can be expressed as $w^* = \sum_{i=1}^{n} \alpha_i \varphi(x_i)$.
- Now let's plug in the Representer Theorem into the formulation of ridge regression / SVM.

Representer Theorem: Ridge Regression

- By adding features to ridge regression we had

$$\begin{aligned} J(\tilde{w}) &= \frac{1}{n}\sum_{i=1}^{n}(\tilde{w}^T\varphi(x_i) - y_i)^2 + \lambda\|\tilde{w}\|_2^2 \\ &= \frac{1}{n}\|\Phi\tilde{w} - y\|_2^2 + \lambda\tilde{w}^T\tilde{w}, \end{aligned}$$

  where $\Phi \in \mathbb{R}^{n\times D}$ is the matrix with $\varphi(x_i)^T$ as its ith row.
- Representer Theorem applies giving $\tilde{w} = \sum_{j=1}^{n}\alpha_j\varphi(x_j) = \Phi^T\alpha$.
- Plugging in gives

$$J(\alpha) = \frac{1}{n}\left\|\Phi\Phi^T\alpha - y\right\|_2^2 + \lambda\alpha^T\Phi\Phi^T\alpha.$$

Representer Theorem: Ridge Regression

- Let $K \in \mathbb{R}^{n \times n}$ be given by $K = \Phi\Phi^T$. This is called the **Gram Matrix** and satisfies $K_{ij} = k(x_i, x_j) = \varphi(x_i)^T\varphi(x_j)$:

$$K = \begin{pmatrix} \varphi(x_1)^T\varphi(x_1) & \cdots & \varphi(x_1)^T\varphi(x_n) \\ \vdots & \ddots & \vdots \\ \varphi(x_n)^T\varphi(x_1) & \cdots & \varphi(x_n)^T\varphi(x_n) \end{pmatrix}.$$

- We can write ridge regression in the kernelized form by turning

$$J(\alpha) = \frac{1}{n}\left\|\Phi\Phi^T\alpha - y\right\|_2^2 + \lambda\alpha^T\Phi\Phi^T\alpha.$$

into

$$J(\alpha) = \frac{1}{n}\|K\alpha - y\|_2^2 + \lambda\alpha^T K\alpha.$$

- Can derive the solution algebraically (see Homework 4).
- Prediction function is $f_\alpha(x) = (w^*)^T\varphi(x) = \sum_{i=1}^n \alpha_i k(x_i, x)$.

Representer Theorem: Ridge Regression

Remarks

- With Representer Theorem, we can re-parameterize our prediction function from $f_w(x) = w^T \varphi(x)$ to $f_\alpha(x) = \sum_{i=1}^n \alpha_i k(x_i, x)$.
- The feature representation $\varphi(x)$ only appears in inner product form in both the loss function and the prediction function.
- Therefore, we just need to evaluate the kernel function $k(x, y)$ and never need to explicitly evaluate $\varphi(x)$.
- We know that it's much easier to compute the kernel $k(x, y)$.
- The kernel $k(x, y)$, to some extent, represents a similarity score between two data points.

Representer Theorem: Primal SVM

- For a general linear model, the same derivation above shows

$$J(w) = L(\Phi w) + R(\|w\|_2)$$

becomes

$$J(\alpha) = L(K\alpha) + R(\sqrt{\alpha^T K \alpha}).$$

Here $\varphi(x_i)^T w$ became $(K\alpha)_i$.

- The primal SVM (bias in features) has loss function

$$J(w) = \frac{C}{n} \sum_{i=1}^{n} (1 - y_i(\varphi(x_i)^T w))_+ + \|w\|_2^2.$$

- This is kernelized to

$$J(\alpha) = \frac{C}{n} \sum_{i=1}^{n} (1 - y_i(K\alpha)_i)_+ + \alpha^T K \alpha.$$

- Positive decision made if $(w^*)^T \varphi(x) = \sum_{i=1}^{n} \alpha_i k(x_i, x) > 0$.

Dual SVM

- The dual SVM problem (with features) is given by

$$\begin{aligned} \text{maximize}_{\alpha} \quad & \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \varphi(x_i)^T \varphi(x_j) \\ \text{subject to} \quad & \sum_{i=1}^{n} \alpha_i y_i = 0 \\ & \alpha_i \in \left[0, \frac{c}{n}\right] \quad \text{for } i = 1, \ldots, n. \end{aligned}$$

- We can immediately kernelize (no representer theorem needed) by replacing $\varphi(x_i)^T \varphi(x_j) = k(x_i, x_j)$.
- Recall that we were able to derive the conclusion of the representer theorem using strong duality for SVMs.

Mercer's Theorem

- Not all function $f(x, y)$ are valid kernels. Why?
- $k(x, y) = \varphi(x)\varphi(y)$
- How can we know if $k(x, y)$ is a valid kernel or not?

Theorem (Mercer's Theorem)

Fix a kernel $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. *There is a Hilbert space* H *and a feature map* $\varphi : \mathcal{X} \to H$ *such that* $k(x, y) = \langle \varphi(x), \varphi(y) \rangle_H$ *if and only if for any* $x_1, \ldots, x_n \in \mathcal{X}$ *the associated matrix* K *is positive semi-definite:*

$$K = \begin{pmatrix} k(x_1, x_1) & \cdots & k(x_1, x_n) \\ \vdots & \ddots & \vdots \\ k(x_n, x_1) & \cdots & k(x_n, x_n) \end{pmatrix}.$$

Such a kernel k *is called* positive semi-definite.

Positive Semi-Definite

Definition (Positive Semi-Definite)

A matrix $A \in \mathbb{R}^{n \times n}$ is **positive semi-definite** if it is symmetric and

$$x^T A x \geq 0$$

for all $x \in \mathbb{R}^n$.

- Equivalent to saying the matrix is symmetric with non-negative eigenvalues.

Valid Kernels

In plain English, a function $k(x, y)$ is a valid kernel iff:

- It's symmetric, i.e. $f(x, y) = f(y, x)$.
- The Gram Matrix K is positive semi-definitive.

Finding Your Own Kernels

Let $k_1, k_2 : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be positive semi-definite kernels. Then so are the following:

- $k_3(w, x) = k_1(w, x) + k_2(w, x)$
- $k_4(w, x) = \alpha k_1(w, x)$ for $\alpha \geq 0$
- $k_5(w, x) = f(w)f(x)$ for any function $f : \mathcal{X} \to \mathbb{R}$
- $k_6(w, x) = k_1(w, x)k_2(w, x)$

RBF Kernel

- As we saw last time, the most frequently used kernel is the Radial Basis Function (RBF) kernel

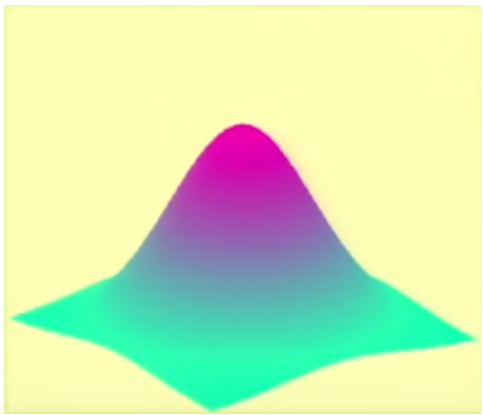
$$k(w, x) = \exp\left(-\frac{\|w - x\|_2^2}{2\sigma^2}\right).$$

- Is there a corresponding feature map $\varphi : \mathbb{R}^d \to \mathbb{R}^D$ so that $k(w, x) = \varphi(w)^T \varphi(x)$?
- Unfortunately there is no finite D that will work.
- Why? Think about Taylor Series for the exponential function. See this link for details.

RBF Kernel

$$k(w, x) = \exp\left(-\frac{\|w - x\|_2^2}{2\sigma^2}\right).$$

- 2d RBF kernel looks like the following.
- Let's say we fix w. The $k(w, x)$ is high when x is very close to w. The value decays as x is moving away from w.
- σ controls the spread of the kernel. The higher σ is the wider / flatter the landscape is for $k(w, x)$.



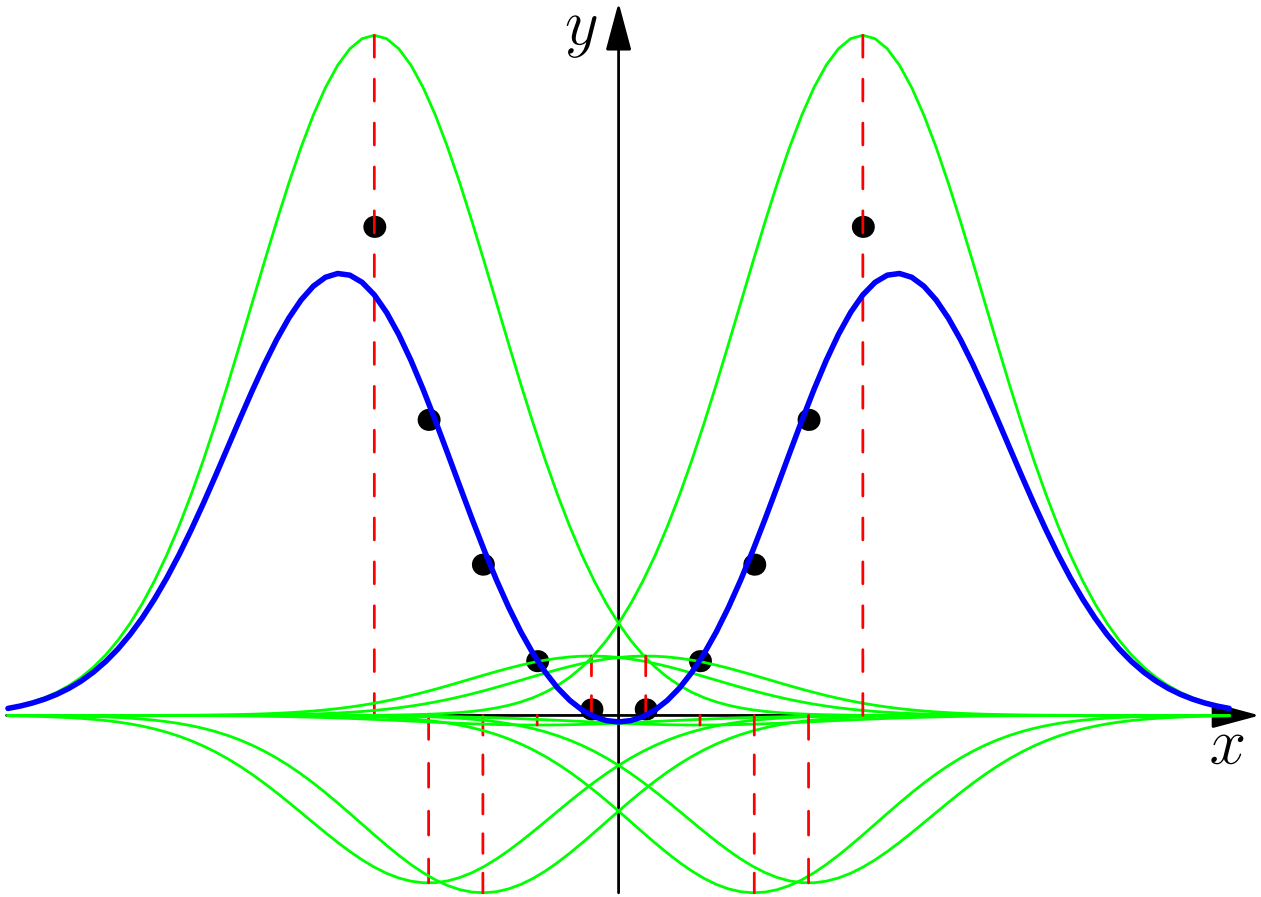
Representer Theorem for RBF Kernels

- As we saw earlier for ridge regression and SVM classification, the decision function has the form $f_\alpha(x) = \sum_{i=1}^n \alpha_i k(x_i, x)$.
- For ridge regression, this means that using the RBF kernel amounts to approximating our data by a linear combination of Gaussian bumps.
- For SVM classification, each $k(x_i, x) = \exp\left(-\|x_i - x\|_2^2/(2\sigma^2)\right)$ represents a exponentially decaying distance between x_i and x. Thus our decisions depend on our proximities to data points.

RBF Regression

- Below we use 10 uniformly spaced x-values between -2 and 2, with $y_i = x_i^2$. We fit kernelized ridge regression with the RBF kernel using $\sigma = 1$ and $\lambda = .1$.
- Each green curve is $g(x) = \alpha_i k(x_i, x)$. The predicted function is drawn in blue.
- As you might expect, extrapolating outside of $[-2, 2]$ can have poor results.
- People will often normalize the RBF kernel (see Hastie, Tibshirani, Friedman p. 213).

RBF Regression

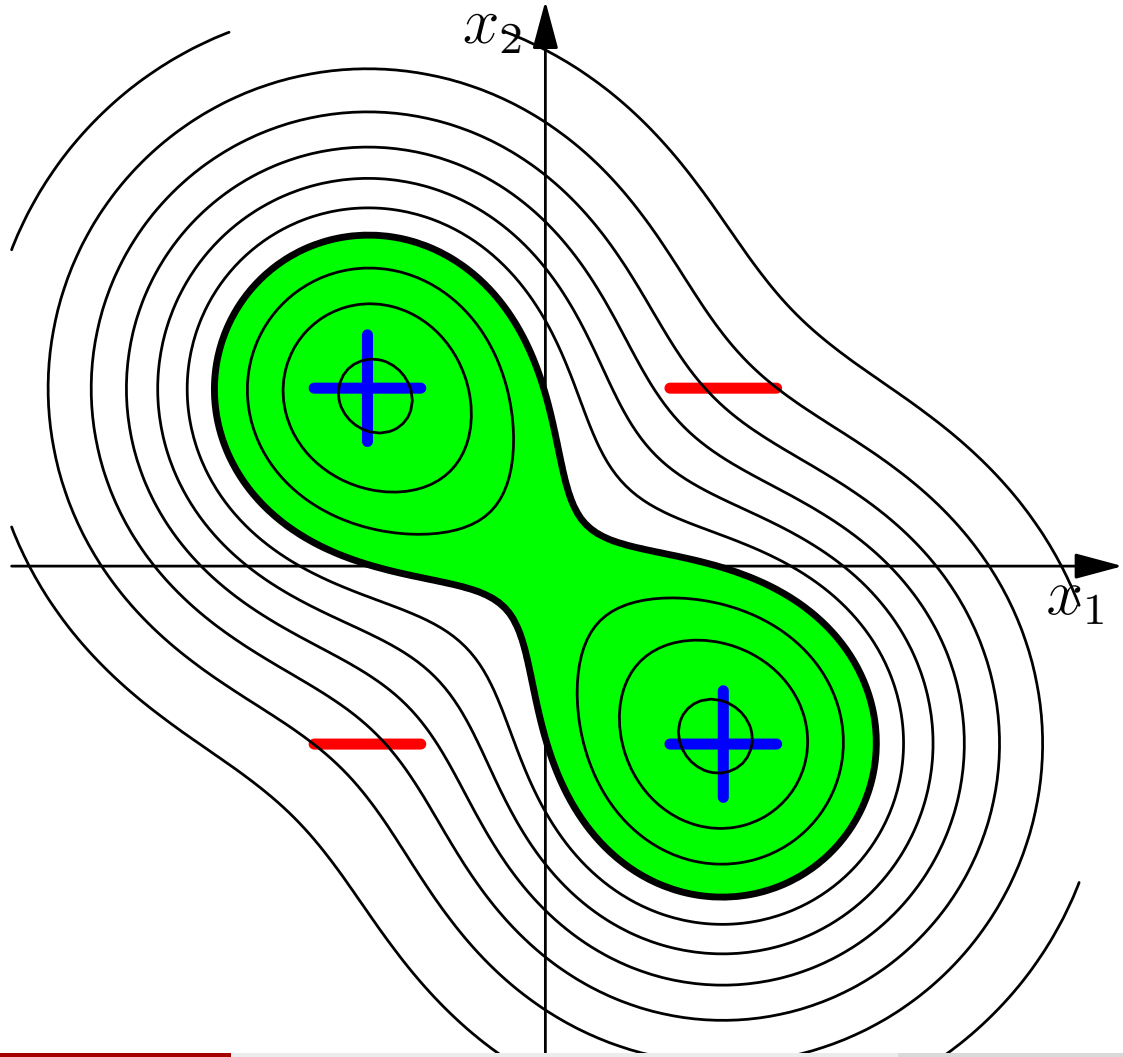


RBF Classification

- Next we show 4 points placed on the corners of a square with positive and negative points on each diagonal.

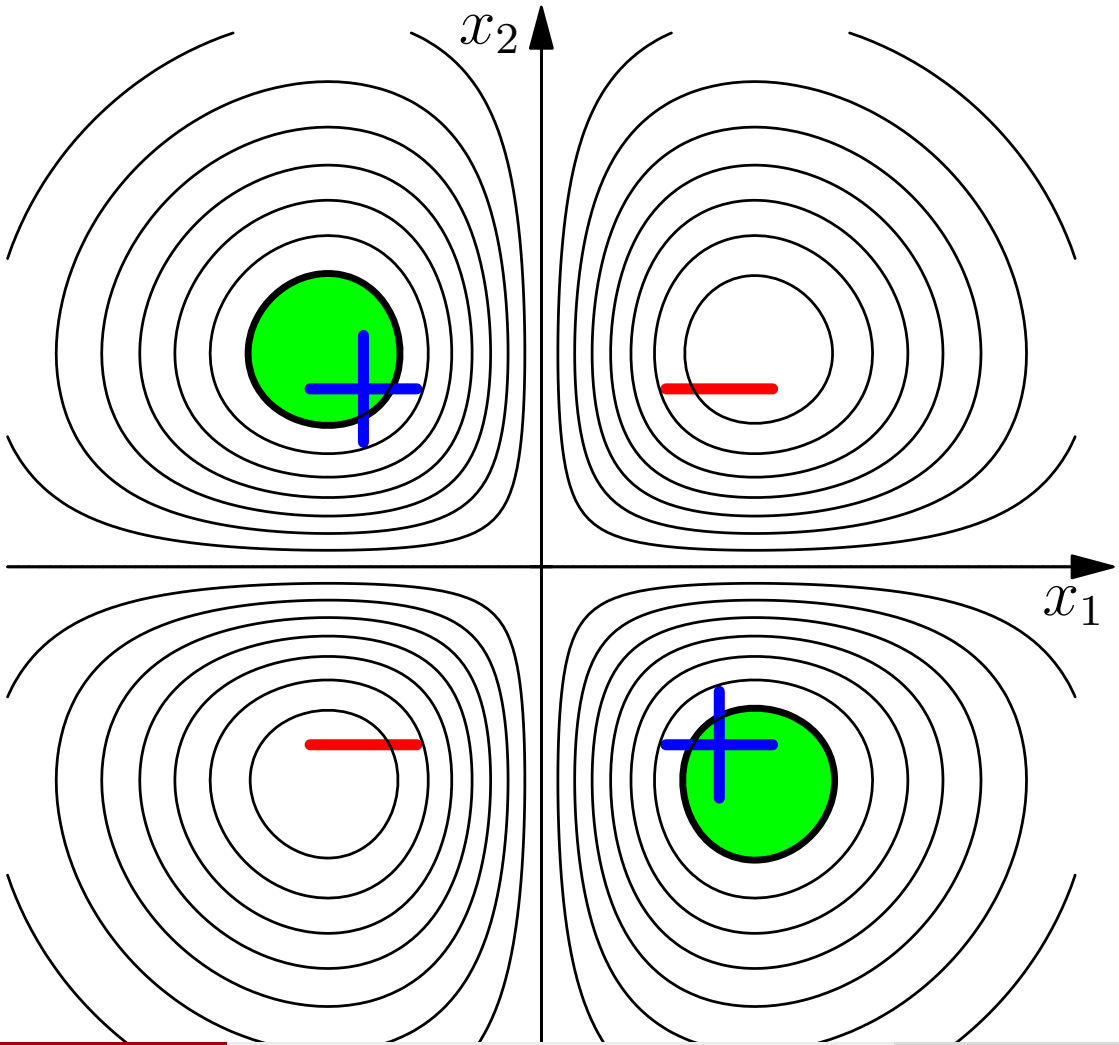
RBF Classification

Contours of $f(x) = k(x_1, x) + k(x_2, x)$ where x_1, x_2 are positive examples, and $\sigma = 1$.



RBF Classification

Contours of $f(x) = k(x_1, x) + k(x_2, x) - k(x_3, x) - k(x_4, x)$ where x_1, x_2 are positive examples, and $\sigma = 1$.



References

- DS-GA 1003 Machine Learning Spring 2019
- DS-GA 1003 Machine Learning Spring 2020