Intro ML: Tutorial on Kernel Methods

Gideon Dresdner,[1] Vincent Fortuin [1]

[1]ETH Zurich

Motivation

1. Need to fit non-linear functions.
2. Map data using a non-linear function. Then, perform a linear regression on the resulting “features”.
3. We have a problem — number of features explodes.

Solution — the “kernel trick”

Solution — the “kernel trick”

1. “Only compute what you need”

Solution — the “kernel trick”

1. “Only compute what you need”
2. Let $\phi(x_i)$ be the feature expansion. Don’t compute $\phi(x_i)$ for every data point. Instead, find a shortcut to compute $\phi(x_i)^\top \phi(x_j)$ for every pair (x_i, x_j).
3. $(x, y) \mapsto \phi(x)^\top \phi(y)$ is called a kernel

Solution — the “kernel trick”

1. “Only compute what you need”
2. Let $\phi(x_i)$ be the feature expansion. Don’t compute $\phi(x_i)$ for every data point. Instead, find a shortcut to compute $\phi(x_i)^\top \phi(x_j)$ for every pair (x_i, x_j).
3. $(x, y) \mapsto \phi(x)^\top \phi(y)$ is called a kernel
4. The matrix $K_{ij} = \phi(x_i)^\top \phi(x_j)$ generated from all the data is called a kernel matrix, also referred to as a “Gram” matrix.

Example — kernelizing absolute value loss (1)

Example — kernelizing absolute value loss (1)

- $\arg\min_{w} \frac{1}{n} \sum_{i=1}^{n} |w^\top x_i - y_i| + \lambda \|w\|_2^2$

Example — kernelizing absolute value loss (1)

- $\arg\min_{w} \frac{1}{n} \sum_{i=1}^{n} |w^\top x_i - y_i| + \lambda \|w\|_2^2$

- Let $w = \sum_{i=1}^{n} \alpha_i x_i$

Example — kernelizing absolute value loss (1)

- $\arg\min_w \frac{1}{n}\sum_{i=1}^{n} |w^\top x_i - y_i| + \lambda\|w\|_2^2$
- Let $w = \sum_{i=1}^{n} \alpha_i x_i$
- Rewrite in terms of $\vec{\alpha}$:

$$w^\top x_i = \left(\sum_{j=1}^{n} \alpha_j x_j\right)^\top x_i = \sum_{j=1}^{n} \alpha_j x_j^\top x_i \qquad (1)$$

Example — kernelizing absolute value loss (1)

- $\arg\min_w \frac{1}{n}\sum_{i=1}^{n} |w^\top x_i - y_i| + \lambda \|w\|_2^2$
- Let $w = \sum_{i=1}^{n} \alpha_i x_i$
- Rewrite in terms of $\vec{\alpha}$:

$$w^\top x_i = \left(\sum_{j=1}^{n} \alpha_j x_j\right)^\top x_i = \sum_{j=1}^{n} \alpha_j x_j^\top x_i \quad (1)$$

And,

$$\|w\|_2^2 = w^\top w = \sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j x_i^\top x_j \quad (2)$$

Example — kernelizing absolute value loss (2)

- Putting this together,

$$\arg\min_w \frac{1}{n}\sum_{i=1}^{n} |w^\top x_i - y_i| + \lambda\|w\|_2^2 \tag{3}$$

$$= \frac{1}{n}\sum_{i=1}^{n} \left|\sum_{j=1}^{n} \alpha_j x_j^\top x_i - y_i\right| + \lambda \sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j x_i^\top x_j \tag{4}$$

- x_i's only appear within inner products! Let $K_{ij} := k(x_i, x_j)$ for some kernel function k.

Example — kernelizing absolute value loss (3)

- $\sum_{j=1}^{n} \alpha_j k(x_j, x_i) - y_i = \vec{\alpha}^\top K_i - y_i$ where $[K_i]_j := k(x_j, x_i)$
- $\sum_{i=1}^{n} |\vec{\alpha}^\top K_i - y_i| = \|\vec{\alpha}^\top K - \vec{y}\|_1$
- Thus,

$$\arg\min_{w} \frac{1}{n} \sum_{i=1}^{n} |w^\top x_i - y_i| + \lambda \|w\|_2^2 \tag{5}$$

$$= \arg\min_{\vec{\alpha}} \frac{1}{n} \|\vec{\alpha}^\top K - \vec{y}\|_1 + \lambda \vec{\alpha}^\top K \vec{\alpha} \tag{6}$$

How to derive kernel formulation

1. Assume a linear kernel

How to derive kernel formulation

1. Assume a linear kernel
2. Only consider solutions that are in the linear span of the data: $w^* = \sum_{i=1}^{n} \alpha_i x_i$.

How to derive kernel formulation

1. Assume a linear kernel
2. Only consider solutions that are in the linear span of the data: $w^* = \sum_{i=1}^{n} \alpha_i x_i$.
3. Reformulate so that X only appears as $X^\top X$

How to derive kernel formulation

1. Assume a linear kernel
2. Only consider solutions that are in the linear span of the data: $w^* = \sum_{i=1}^{n} \alpha_i x_i$.
3. Reformulate so that X only appears as $X^\top X$
4. Replace $X^\top X$ with the kernel matrix $K(X, X)_{ij} = k(x_i, x_j)$, also called the “Gram Matrix.”

Kernel Methods in practice

1. Determine the type of problem you have (regression, binary classification, multi-class classification, etc.)
2. Select an appropriate model (perceptron, SVM, etc.)
3. **Construct a kernel**
4. Repeat

Kernel methods in practice — “with great power comes great responsibility”

Pros

1. Explicit control (and understanding) of the model.
2. Incorporate prior knowledge via kernel engineering.

Cons

1. Avoided large d (e.g. $d = \infty$), but now our solution is in $\mathbb{R}^n$ — lots of research on this scaling problem
2. Kernels can be hard to design without thorough understanding of the problem/dataset — lots of research on this kernel design problem

Choose the right kernel

So you have decided to use some kernel-based method, say SVM. **How do you design a kernel?**

- Sit and think very hard

- Combine known kernels (that other people have laborously constructed) to suit your particular problem.

Spectrum kernel for biological sequences

Let x, y be two bio sequences (e.g. GATAACA). Define,

$$k(x, y) = \sum_{s_k} \#(x, s_k)\#(y, s_k) \tag{7}$$

where the sum is over *all* subsequences, s_k of length k.

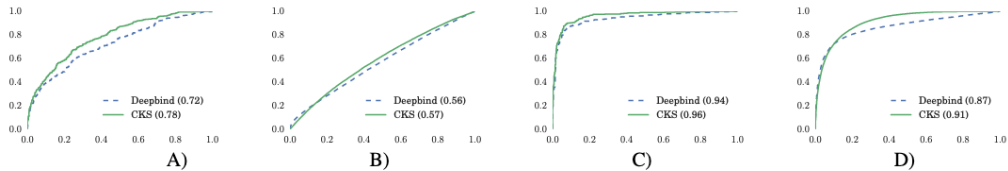
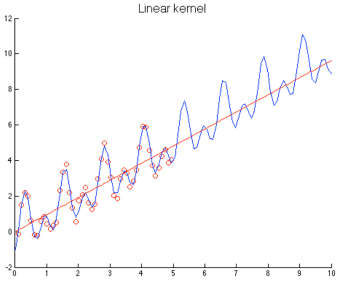
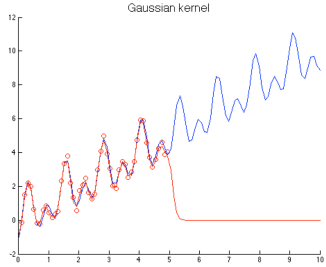
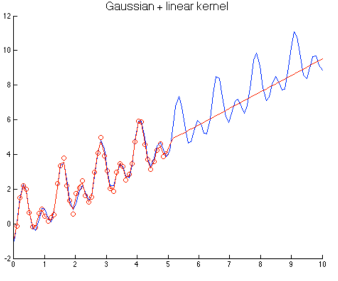


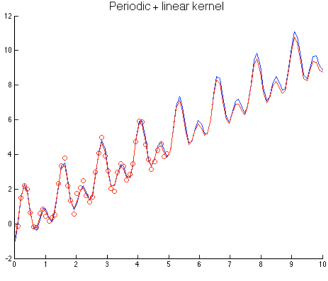
Figure 1: Comparison of ROC between DeepBind on CKS on EGR1 and ATF2 for GM12878. A) ROC for ATF2 on the DeepBind's test set. B) ROC for ATF2 on the ENCODE set. C) ROC for EGR1 on the DeepBind's test set. D) ROC for EGR1 on the the ENCODE set.

Example fits









Properties of Kernels

Symmetry and positive semi-definiteness

A function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a valid kernel if and only if it is symmetric and positive semi-definite, that is,

1. $k(\mathbf{x}, \mathbf{x}') = k(\mathbf{x}', \mathbf{x})$
2. Any of the following equivalent statements holds
   - 2.1 The kernel matrix K computed on data $X \subset \mathcal{X}$ is positive definite for all X, that is, $\mathbf{v}^\top K \mathbf{v} \geq 0 \quad \forall \mathbf{v} \in \mathbb{R}^n$
   - 2.2 $\sum_i \sum_j c_i c_j k(\mathbf{x}_i, \mathbf{x}_j) \geq 0 \quad \forall X \subset \mathcal{X}, c_i, c_j \in \mathbb{R}$

Properties of Kernels

Inner product in Hilbert space (Mercer 1909)

A function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a valid kernel if and only if there exists a feature map $\phi : \mathcal{X} \to \mathcal{H}$ into a Hilbert space $\mathcal{H}$, such that

$$k(\mathbf{x}, \mathbf{x}') = \phi(\mathbf{x})^\top \phi(\mathbf{x}') \quad \forall \mathbf{x}, \mathbf{x}' \in \mathcal{X} \, .$$

Kernel composition rules

Sum rule

If k_1 and k_2 are valid kernels on $\mathcal{X}$, then $k_1 + k_2$ is a valid kernel on $\mathcal{X}$.

Scaling rule

If $\lambda > 0$ and k is a valid kernel on $\mathcal{X}$, then λk is a valid kernel on $\mathcal{X}$.

Product rule

If k_1 and k_2 are valid kernels on $\mathcal{X}$, then $k_1 k_2$ is a valid kernel on $\mathcal{X}$. If k_1 is a valid kernel on $\mathcal{X}_1$ and k_2 is a valid kernel on $\mathcal{X}_2$, then $k_1 k_2$ is a valid kernel on $\mathcal{X}_1 \times \mathcal{X}_2$.

Proving kernel validity

1. Proving that a kernel is valid:
   - 1.1 Prove symmetry (easy) and positive definiteness (usually harder)
   - 1.2 Find an explicit feature map $\phi(\mathbf{x})$, such that $k(\mathbf{x}, \mathbf{x}') = \phi(\mathbf{x})^\top \phi(\mathbf{x}')$
   - 1.3 Derive the kernel from other valid ones using the composition rules
2. Proving that a kernel is invalid:
   - 2.1 Find a counterexample against symmetry (might be easy)
   - 2.2 Find a counterexample against positive definiteness (might be harder)

Exercises

For $\mathbf{x}, \mathbf{x}' \in \mathbb{R}^d$ and $k(\mathbf{x}, \mathbf{x}') = (\mathbf{x}^\top \mathbf{x}' + 1)^2$, show that $k(\cdot, \cdot)$ is a valid kernel.

Exercises

$$
\begin{aligned}
(\mathbf{x}^\top \mathbf{x}' + 1)^2 &= (\sum_{i=1}^{d} x_i x_i' + 1)^2 \\
&= 1 + 2\sum_i x_i x_i' + \sum_i \sum_j x_i x_j x_i' x_j' \\
&= 1 + \sum_i (\sqrt{2} x_i)(\sqrt{2} x_i') + \sum_i \sum_j (x_i x_j)(x_i' x_j')
\end{aligned}
$$

Thus $k(\mathbf{x}, \mathbf{x}') = \phi(\mathbf{x})^\top \phi(\mathbf{x}')$ with

$$
\phi(\mathbf{x}) = \left[1, \sqrt{2}x_1, \ldots, \sqrt{2}x_d, x_1 x_1, x_1 x_2, \ldots, x_1 x_d, x_2 x_1, \ldots, x_d x_d\right]^\top
$$

Exercises

For $\tilde{k}(\mathbf{x}, \mathbf{x}') = f(k(\mathbf{x}, \mathbf{x}'))$, show that $\tilde{k}(\cdot, \cdot)$ is a valid kernel if $k(\cdot, \cdot)$ is a valid kernel and f is a polynomial with non-negative coefficients.

Exercises

$$\tilde{k}(\mathbf{x}, \mathbf{x}') = a_1 k(\mathbf{x}, \mathbf{x}')^{e_1} + a_2 k(\mathbf{x}, \mathbf{x}')^{e_2} + \dots$$

Proof:

- All the $k(\mathbf{x}, \mathbf{x}')^{e_i}$ are valid kernels by the product rule.
- Thus, all the $a_i k(\mathbf{x}, \mathbf{x}')^{e_i}$ are valid kernels by the scaling rule.
- Thus, $\tilde{k}(\mathbf{x}, \mathbf{x}')$ is a valid kernel by the sum rule.

Exercises

For $\tilde{k}(\mathbf{x}, \mathbf{x}') = f(\mathbf{x})k(\mathbf{x}, \mathbf{x}')f(\mathbf{x}')$ with $f : \mathcal{X} \to \mathbb{R}$, show that $\tilde{k}(\cdot, \cdot)$ is a valid kernel if $k(\cdot, \cdot)$ is a valid kernel.

Exercises

If $k(\cdot, \cdot)$ is a valid kernel, we can write it as $k(\mathbf{x}, \mathbf{x}') = \phi(\mathbf{x})^\top \phi(\mathbf{x}')$ for some $\phi(\mathbf{x})$.
Thus

$$\begin{aligned}\tilde{k}(\mathbf{x}, \mathbf{x}') &= f(\mathbf{x})\phi(\mathbf{x})^\top \phi(\mathbf{x}') f(\mathbf{x}') \\ &= \left(f(\mathbf{x})\phi(\mathbf{x})\right)^\top \left(f(\mathbf{x}')\phi(\mathbf{x}')\right).\end{aligned}$$

With $\tilde{\phi}(\mathbf{x}) = f(\mathbf{x})\phi(\mathbf{x})$, we can write $\tilde{k}(\mathbf{x}, \mathbf{x}') = \tilde{\phi}(\mathbf{x})^\top \tilde{\phi}(\mathbf{x}')$, which makes it a valid kernel.

Exercises

For $\tilde{k}(\mathbf{x}, \mathbf{x}') = k(f(\mathbf{x}), f(\mathbf{x}'))$ with $f : \mathcal{X} \to \mathcal{X}$, show that $\tilde{k}(\cdot, \cdot)$ is a valid kernel if $k(\cdot, \cdot)$ is a valid kernel.

Exercises

If $k(\cdot, \cdot)$ is a valid kernel, we can write it as $k(\mathbf{x}, \mathbf{x}') = \phi(\mathbf{x})^\top \phi(\mathbf{x}')$ for some $\phi(\mathbf{x})$.
Thus

$$\tilde{k}(\mathbf{x}, \mathbf{x}') = \phi(f(\mathbf{x}))^\top \phi(f(\mathbf{x}'))\,.$$

With $\tilde{\phi}(\mathbf{x}) = \phi(f(\mathbf{x}))$, we can write $\tilde{k}(\mathbf{x}, \mathbf{x}') = \tilde{\phi}(\mathbf{x})^\top \tilde{\phi}(\mathbf{x}')$, which makes it a valid kernel.

Exercises

For the data set $X = [(-3, 4), (1, 0)]$ and the feature map $\phi(\mathbf{x}) = [x_1, x_2, \|\mathbf{x}\|]^\top$, compute the kernel matrix K.

Exercises

$$\phi((-3,4)) = [-3,4,5]^\top$$
$$\phi((1,0)) = [1,0,1]^\top$$
$$\phi((-3,4))^\top \phi((-3,4)) = 50$$
$$\phi((-3,4))^\top \phi((1,0)) = 2$$
$$\phi((1,0))^\top \phi((1,0)) = 2$$
$$K = \begin{bmatrix} 50 & 2 \\ 2 & 2 \end{bmatrix}$$

References

1. *Convolutional kitchen sinks for transcription factor binding site prediction*. Morrow, Alyssa and Shankar, Vaishaal and Petersohn, Devin and Joseph, Anthony and Recht, Benjamin and Yosef, Nir, NeurIPS 2016.

END OF PRESENTATION
BEGININNG OF Q&A