Kernel methods

Lecture 14 of “Mathematics and AI”

Outline

1. The ordinary-least-squares kernel
2. Kernels
3. Representer theorems
4. Kernel methods in practice

   Support vector machines, kernel ridge regression, kernel lasso

The ordinary-least-squares kernel

Parameters are combinations of observations

- For OLS: The parameter vector $\vec{\beta}$ is a **linear combination** of the feature vectors $\vec{x}_i$ $(i = 1, \ldots, n)$:

$$\vec{\beta} = \sum_{i=1}^{n} \alpha_i \vec{x}_i$$

- Proof:

  1. $\vec{\beta}$ is the limit of the sequence $\left(\vec{\beta}^{(0)}, \vec{\beta}^{(1)}, \vec{\beta}^{(2)}, \ldots\right)$ produced by gradient descent with $\vec{\beta}^{(t+1)} = \vec{\beta}^{(t)} - \gamma \frac{\partial L}{\partial \vec{\beta}}$

Parameters are combinations of observations

2. The OLS loss function,

$$L\left(\vec{\beta}\right) = \sum_{i=1}^{n} \left(\vec{\beta} \cdot \vec{x}_i - y_i\right)^2,$$

   is convex, so wlog set $\vec{\beta}^{(0)} = 0$.

3. Proof by induction

Parameters are combinations of observations

- Base case:

$$\vec{\beta}^{(1)} = \vec{\beta}^{(0)} - \gamma \frac{\partial L}{\partial \vec{\beta}}\left(\vec{\beta}^{(0)}\right)$$

$$\vec{\beta}^{(1)} = \vec{\beta}^{(0)} - 2\gamma \sum_{i=1}^{n} \left(\vec{\beta}^{(0)} \cdot \vec{x}_i - y_i\right) \vec{x}_i$$

$$\vec{\beta}^{(1)} = 2\gamma \sum_{i=1}^{n} y_i \vec{x}_i$$

Parameters are combinations of observations

- Induction hypothesis:

$$\vec{\beta}^{(t)} = \sum_{i=1}^{n} \alpha_i^{(t)} \vec{x}_i \quad \rightarrow \quad \vec{\beta}^{(t+1)} = \sum_{i=1}^{n} \alpha_i^{(t+1)} \vec{x}_i$$

Parameters are combinations of observations

- Induction step:

$$\vec{\beta}^{(t+1)} = \vec{\beta}^{(t)} - 2\gamma \sum_{i=1}^{n} \left(\vec{\beta}^{(t)} \cdot \vec{x}_i - y_i \right) \vec{x}_i$$

$$\vec{\beta}^{(t+1)} = \sum_{j=1}^{n} \alpha_j^{(t)} \vec{x}_j - 2\gamma \sum_{i=1}^{n} \left(\sum_{j=1}^{n} \alpha_j^{(t)} \vec{x}_j \cdot \vec{x}_i - y_i \right) \vec{x}_i$$

$$\vec{\beta}^{(t+1)} = \sum_{j=1}^{n} \left(\alpha_j^{(t)} - 2\gamma \sum_{j=1}^{n} \alpha_j^{(t)} \vec{x}_j \cdot \vec{x}_i - y_i \right) \vec{x}_i$$

Implications for OLS loss and prediction

- OLS loss function rewritten:

$$L\left(\vec{\beta}\right) = \sum_{i=1}^{n} \left(\vec{\beta} \cdot \vec{x}_i - y_i\right)^2 = \sum_{i=1}^{n} \left[\left(\sum_{j=1}^{n} \alpha_j \vec{x}_j\right) \cdot \vec{x}_i - y_i\right]^2 = \sum_{i=1}^{n} \left[\sum_{j=1}^{n} \alpha_j (\vec{x}_j \cdot \vec{x}_i) - y_i\right]^2$$

- Prediction for query $\vec{x}_k$:

$$\vec{\beta} \cdot \vec{x}_k = \left(\sum_{i=1}^{n} \alpha_i \vec{x}_i\right) \cdot \vec{x}_k = \sum_{i=1}^{n} \alpha_i (\vec{x}_i \cdot \vec{x}_k)$$

Interpretation

$$\vec{\beta} \cdot \vec{x}_k = \sum_{i=1}^{n} \alpha_i (\vec{x}_i \cdot \vec{x}_k)$$

- Scalar products can be used to assess
  - similarity of vectors
  - e.g., closeness* of two feature vectors in feature space

- Predicted response y_k for a query $\vec{x}_k$ is function of the responses y_i for $\vec{x}_i$ in the training data.
- The y_i for $\vec{x}_i$ that are similar to $\vec{x}_k$ contribute more than others!

*Close in the sense of the metric/norm induced by the scalar product

Comparison to k nearest neighbors

- KNN predicts outcome of a query $\vec{x}$ by averaging the k closest observations to $\vec{x}$
- OLS linear regression predicts outcome of a query $\vec{x}$ by a weighted average of all observations with greatest weights assigned to observations closest to $\vec{x}$

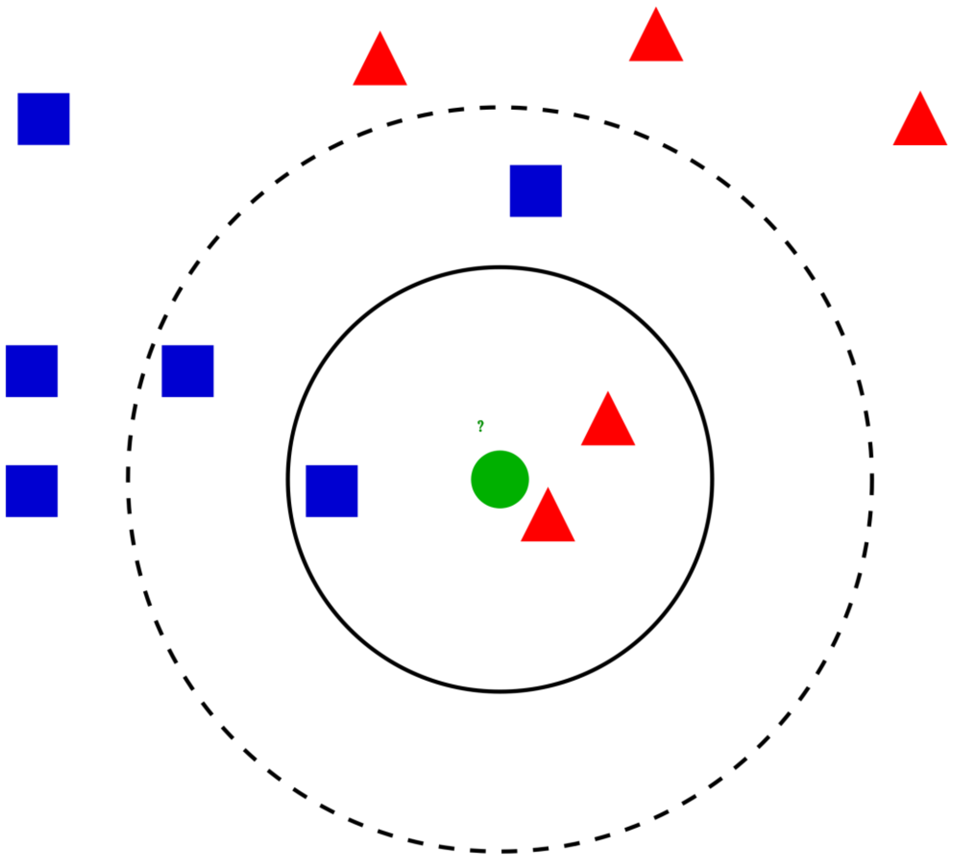

Image source: https://www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/

Comparison to k nearest neighbors

- KNN and OLS have different kernels!
- Informally: A kernel is a function that measures similarity of two vectors
- Formally: A kernel $K(\cdot,\cdot)$ is a positive semi-definite function of two inputs

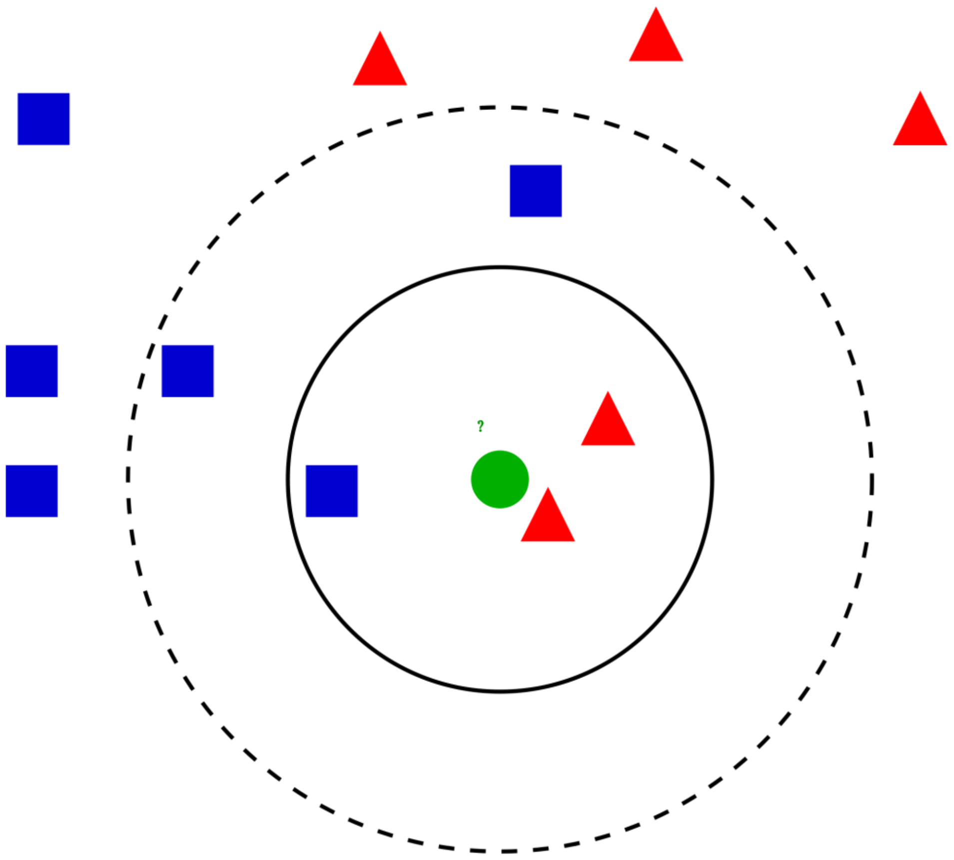

Image source: https://www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/

Nonlinear kernels

Examples of (non-)linear kernels

- Linear kernel: $$K(x,z) = x \cdot z$$
- Polynomial kernel: $$K(x,z) = (1 + x \cdot z)^d$$
- Gaussian kernel: $$K(x,z) = \exp(-\frac{1}{\sigma^2}\|x - z\|^2)$$
- Exponential kernel: $$K(x,z) = \exp(-\frac{1}{2\sigma^2}\|x - z\|)$$

Hilbert spaces for machine learning

- Consider a feature space $\mathcal{X}$. Define an associated vector space $\mathcal{H}$ (Hilbert space) of "nice" functions $f(\mathcal{X}) \to \mathbb{R}$, in particular:
  - $\mathcal{H}$ has vector addition:
$$\forall f, g \in \mathcal{H}: f + g \in \mathcal{H}$$
  - $\mathcal{H}$ has scalar multiplication:
$$\forall f \in \mathcal{H}, c \in \mathbb{R}: cf \in \mathcal{H}$$
  - $\mathcal{H}$ has (real-valued) scalar product:
$$\forall f, g \in \mathcal{H}: f \cdot g \in \mathbb{R}$$

Reproducing-kernel Hilbert spaces (RHKS)

- An RHKS is a Hilbert space in which

closeness of functions (i.e., $||f - g||$ small)

implies

pointwise closeness (i.e., small $f(x)g(x)$ for all $x \in \mathcal{X}$)

and vice versa.

- Such a Hilbert space has some special functions

$$K_x \in \mathcal{H} \text{ s.t. } \forall f\colon\ f \cdot K_x = f(x)$$

Reproducing-kernel Hilbert spaces (RHKS)

- For any $\in \mathcal{X}$ with associated $K_x, K_z \in \mathcal{H}$:

$$K(x,z) = K_z(x) = K_x(z) = K_x \cdot K_z$$

- When points x, z are close in $\mathcal{X}$, their associated functions K_x, K_z are close $\mathcal{H}$.
- The functions $K_x, K_z \in \mathcal{H}$ ***reproduce the kernel*** $K(x,z)$!

Representer theorems

Fitting n points in d dimensions

Representer theorem

Theorem: Consider a positive-definite real-valued kernel $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ on a non-empty set $\mathcal{X}$ with a corresponding reproducing kernel Hilbert space H_k. Let there be given

- a training sample $(x_1, y_1), \dots, (x_n, y_n) \in \mathcal{X} \times \mathbb{R}$,
- a strictly increasing real-valued function $g\colon [0, \infty) \to \mathbb{R}$, and
- an arbitrary error function $E\colon (\mathcal{X} \times \mathbb{R}^2)^n \to \mathbb{R} \cup \{\infty\}$,

which together define the following regularized empirical risk functional on H_k:

$$f \mapsto E\left((x_1, y_1, f(x_1)), \dots, (x_n, y_n, f(x_n))\right) + g\left(\|f\|\right).$$

Then, any minimizer of the empirical risk

$$f^* = \operatorname*{argmin}_{f \in H_k} \left\{E\left((x_1, y_1, f(x_1)), \dots, (x_n, y_n, f(x_n))\right) + g\left(\|f\|\right)\right\}, \quad (*)$$

admits a representation of the form:

$$f^*(\cdot) = \sum_{i=1}^{n} \alpha_i k(\cdot, x_i),$$

where $\alpha_i \in \mathbb{R}$ for all $1 \le i \le n$.

Source: https://en.wikipedia.org/wiki/Representer_theorem

Kernel methods in practice

Support vector machines

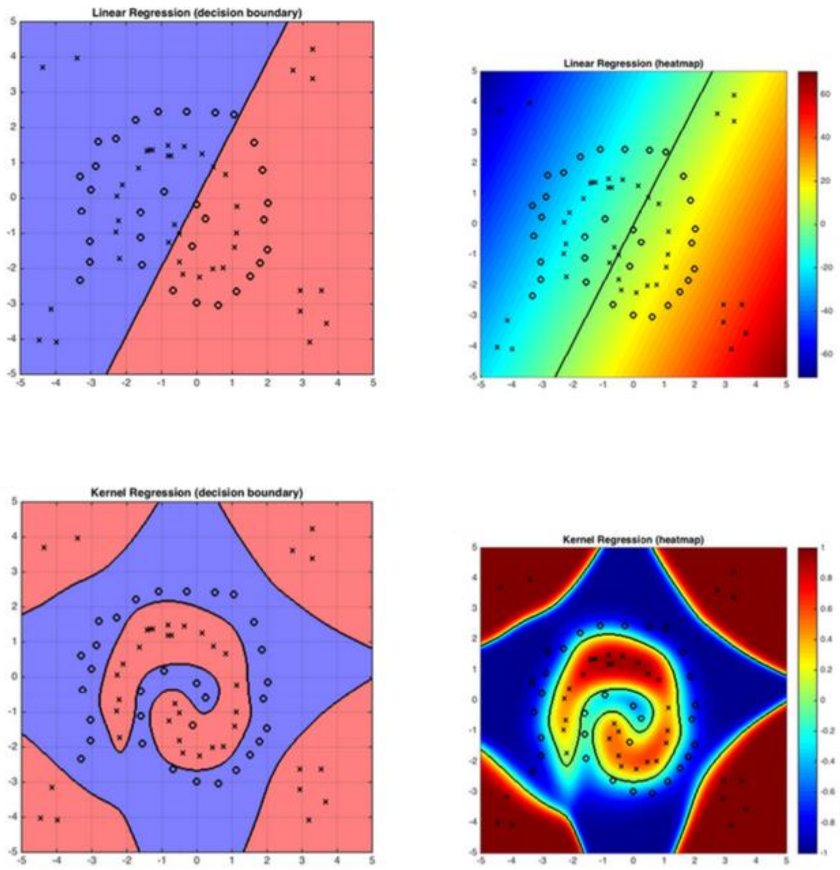

Image source: https://www.cs.cornell.edu/courses/cs4780/2018fa/lectures/lecturenote13.html