机器学习

主讲教师：刘峤

第6章 实例学习

Instance-based Learning

Instance-based Learning

- A.K.A. memory-based, case-based, or distance-based.
- The idea is extremely simple:
  - given a new example **x**
  - find the most **similar** training example(s) and
  - **predict** a similar output
- Examples:
  - Nearest Neighbor Methods
  - K-Nearest Neighbors
  - Kernel Regression

Norms

Norms

- a map that assigns a length or size to a mathematical object
  - Vector norm, a map that assigns a length or size to any vector in a vector space
  - Matrix norm, a map that assigns a length or size to a matrix

$$L_p \text{ norm: } (\Sigma_i |x_i|^p)^{1/p}$$

https://ccrma.stanford.edu/~jos/gradient/Vector_Space_Concepts.html

Norms

- For all $a \in R$ and all $u, v \in V$,
  - $L_p(v) \geq 0$, and $L_p(v) = 0$ iff v is the zero vector
  - $L_p(kv) = |k| L_p(v)$
    - homogeneity
  - $L_p(u + v) \leq L_p(u) + L_p(v)$
    - triangle inequality or subadditivity (次可加性)

vector norm

ℓ_0 norm : $||x||_0 = \sum_i^n (x_i \neq 0)$ 向量非零元素的个数

ℓ_1 norm : $||x||_1 = \sum_i^n |x_i|$ 向量所有元素的绝对值之和

ℓ_2 norm : $||x||_2 = (\sum_{i=1}^n |x_i|^2)^{\frac{1}{2}}$ 向量的欧式距离

ℓ_∞ norm : $||x||_\infty = \max_i |x_i|$ 向量元素中的最大值

$\ell_{-\infty}$ norm : $||x||_{-\infty} = \min_i |x_i|$ 向量元素中的最小值

ℓ_p norm : $||x||_p = (\sum_{i=1}^n |x_i|^p)^{\frac{1}{p}}$

matrix norm

列和范数：矩阵列向量中绝对值之和的最大值

$$\ell_1 \text{ norm}: \quad ||A||_1 = \max_j \sum_{i=1}^{m} |a_{ij}|$$

谱范数：A^TA矩阵的最大特征值的开平方

$$\ell_2 \text{ norm}: \quad ||A||_2 = \sqrt{\lambda_1}$$

其中：λ_1为A^TA的最大特征值

行和范数：矩阵行向量中绝对值之和的最大值

$$\ell_\infty \text{ norm}: \quad ||A||_\infty = \max_j \sum_{i=1}^{n} |a_{ij}|$$

Frobenius范数：矩阵元素的绝对值的平方和再开方

$$\ell_F \text{ norm}: \quad ||A||_F = \left(\sum_{i=1}^{m} \sum_{j=1}^{n} |a_{ij}|^2 \right)^{\frac{1}{2}}$$

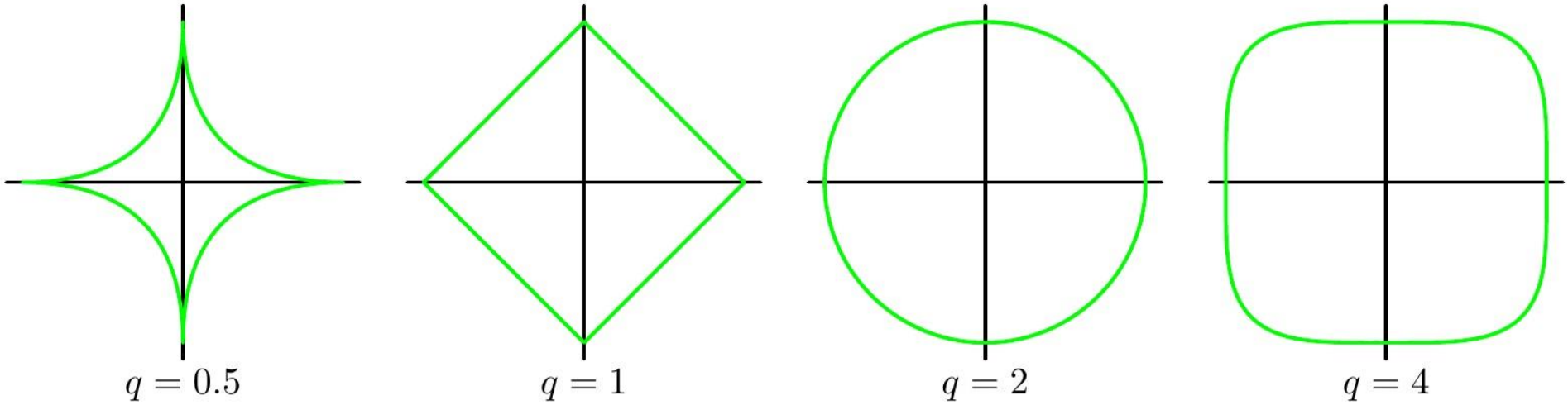


Figure 3.3 Contours of the regularization term in (3.29) for various values of the parameter q.

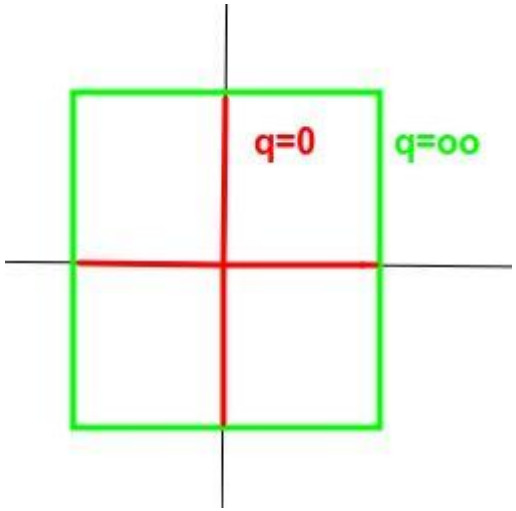
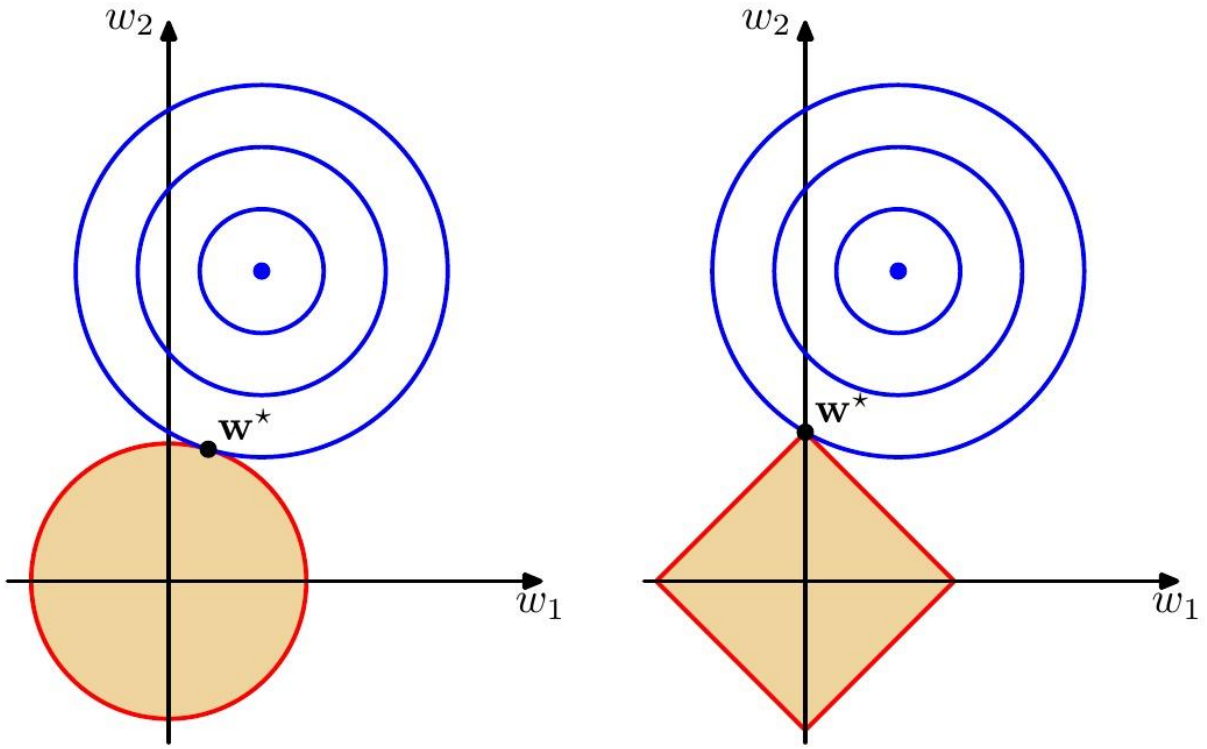
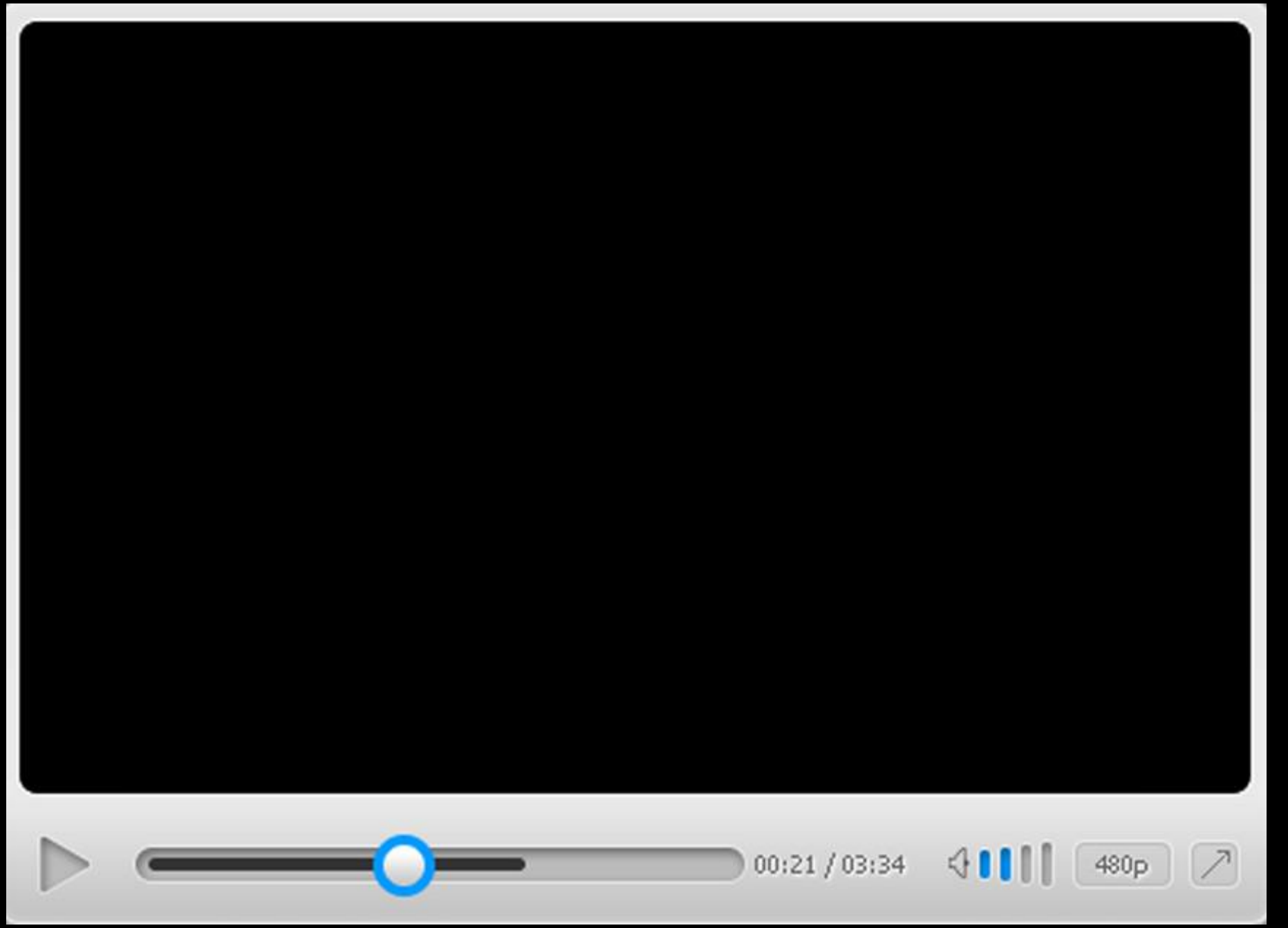


Figure 3.4 Plot of the contours of the unregularized error function (blue) along with the constraint region (3.30) for the quadratic regularizer $q = 2$ on the left and the lasso regularizer $q = 1$ on the right, in which the optimum value for the parameter vector $\mathbf{w}$ is denoted by $\mathbf{w}^\star$. The lasso gives a sparse solution in which $w_1^\star = 0$.

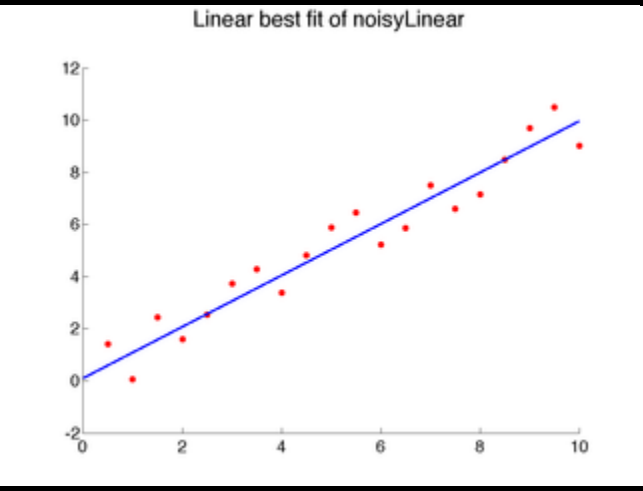


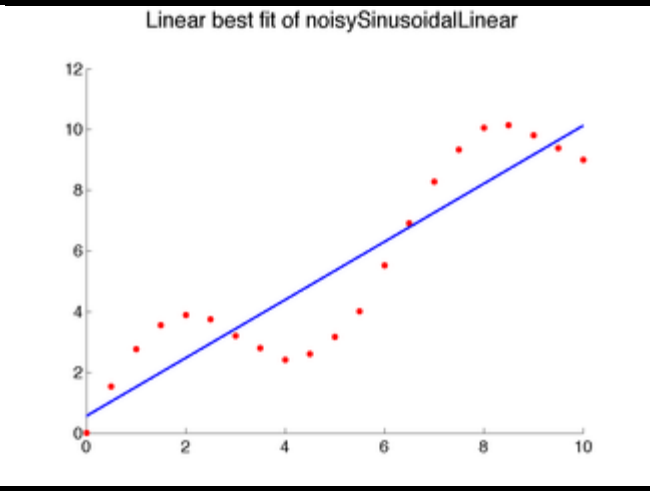
Nearest Neighbor Methods

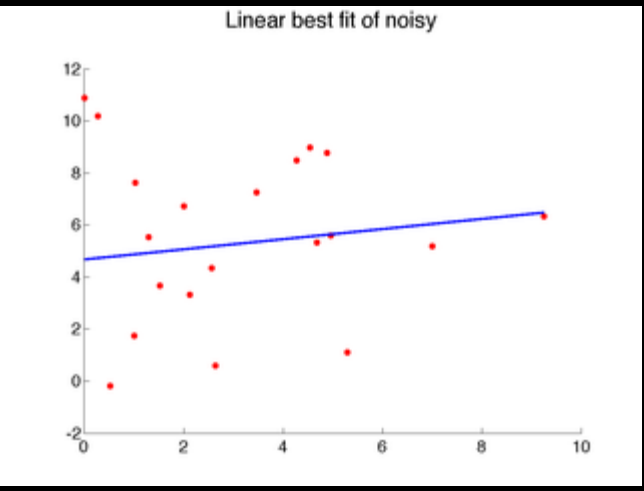

00:21 / 03:34
480p

Nearest Neighbor Methods

- The nearest neighbor idea are not limited to classification.
- Consider the following regression problems:





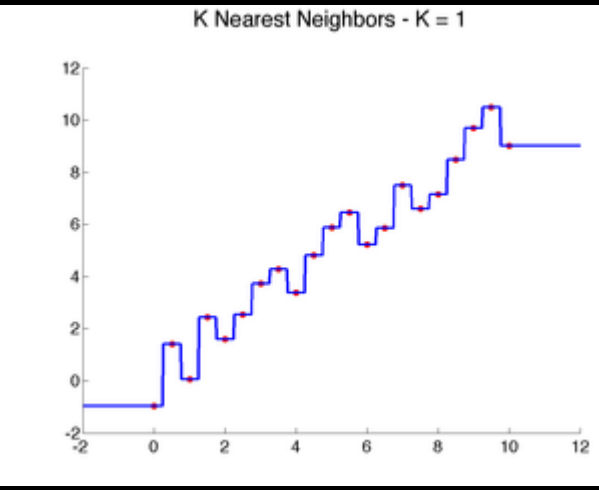


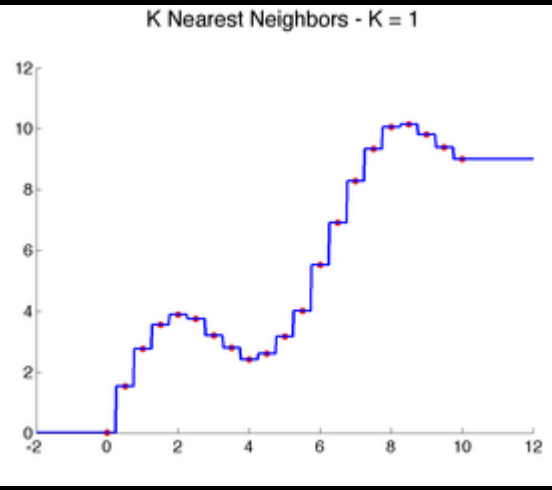
Clearly, linear models do not capture the data well.

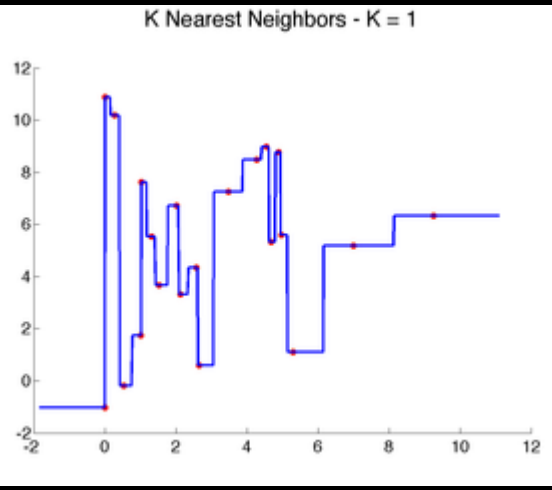
1-Nearest Neighbor Algorithm

1. Given training data $D=\{x_i, y_i\}$, distance function $d(\cdot,\cdot)$ and input x
2. Find $j = \text{argmin}_i\, d(x, x_i)$ and return y_j

- Here are a few examples with $d(x, x_i) = |x - x_i|$ as x varies:

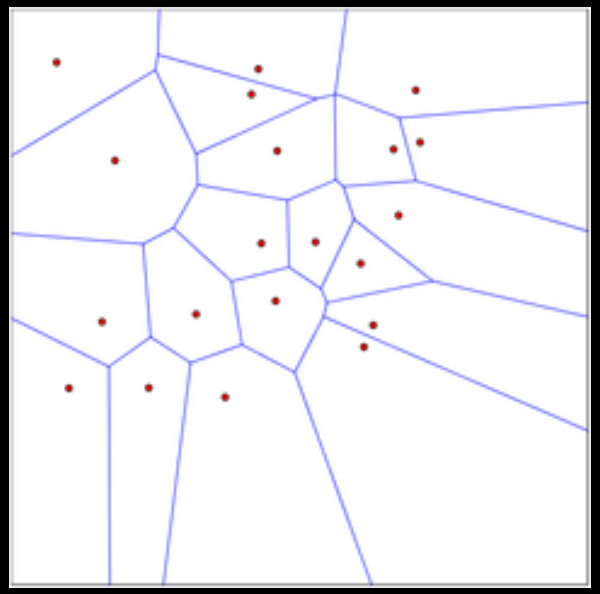






1-Nearest Neighbor Algorithm

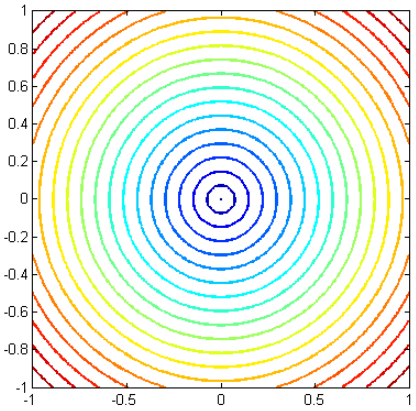
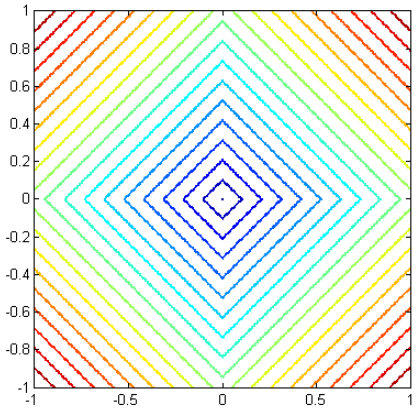
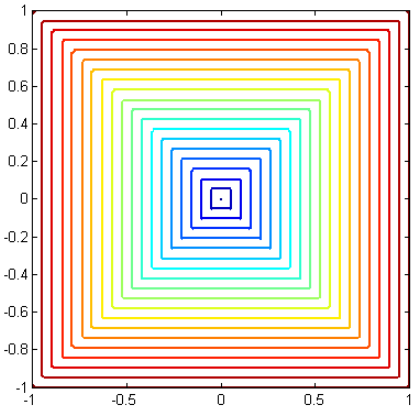
- if we use Euclidean distance with $|| x - xi ||_2$ as x varies



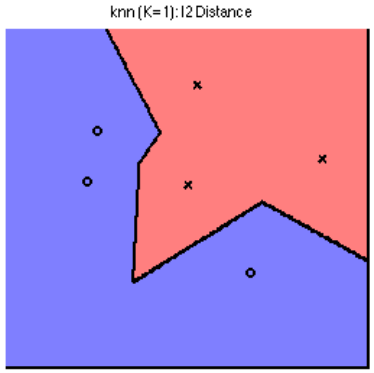
Voronoi diagram

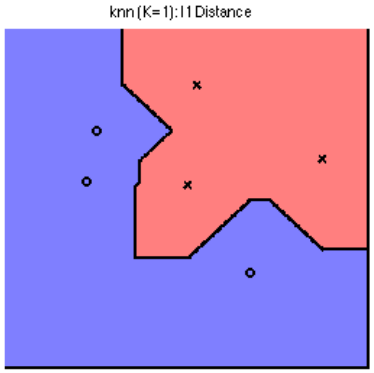
1-Nearest Neighbor Algorithm

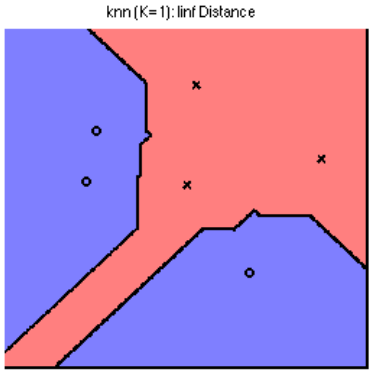
- if we use Euclidean distance with $|| x-xi ||_p$ as x varies

contours of L_2, L_1 and L_∞ norms







Voronoi diagram

Non-parametric models

- 1-NN is one of the simplest examples of a non-parametric method
- in a non-parametric approach
  - the model structure is determined by the training data.
  - The model usually still has some parameters.
  - But their number or type grows with the data.
  - Example: decision trees
- Non-parametric models are much more flexible and expressive than parametric ones, and thus overfitting is a major concern.
- One nice property of non-parametric prediction algorithms is that : with enough data, non-parametric methods are able to model (nearly) anything and therefore achieve (close to) zero bias for any distribution.

Consistency

- An algorithm whose **bias is zero** as the number of examples grows to infinity for any distribution P(y|x) is called **consistent**.
- A **consistent estimator** (or **asymptotically** consistent estimator) is an estimator — a rule for computing estimates of a parameter θ — having the property that as the number of data points used increases indefinitely, the resulting sequence of estimates **converges** in probability to the true parameter θ.

https://en.wikipedia.org/wiki/Consistent_estimator

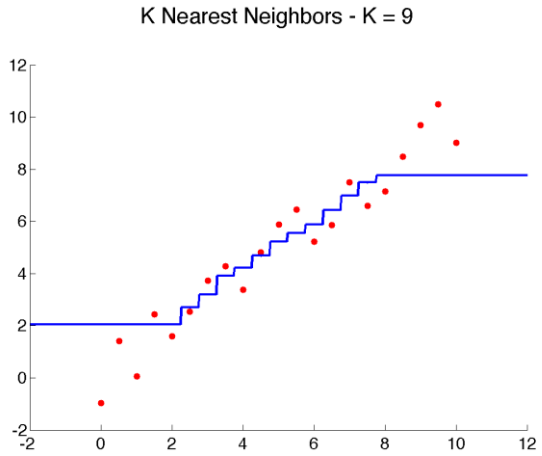
Consistency

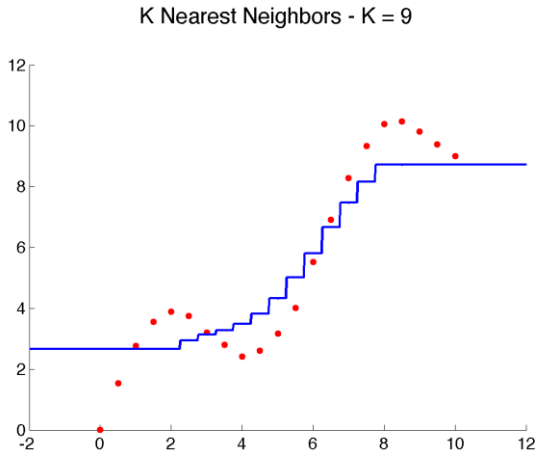
- 1-Nearest Neighbor is consistent
  - Under some reasonable regularity conditions on P(y|x).
- The problem is that the strict memorization aspect of 1-NN leads to a large variance for any finite sample size.
- One way to reduce the variance is local averaging:
  - instead of just one neighbor
  - find K and average their predictions – KNN

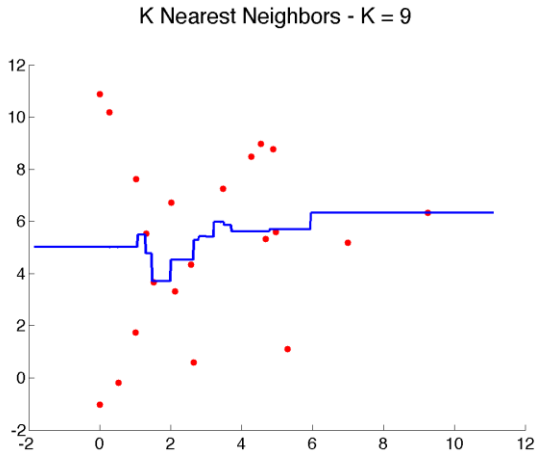
K-Nearest Neighbors

K-Nearest Neighbors Algorithm

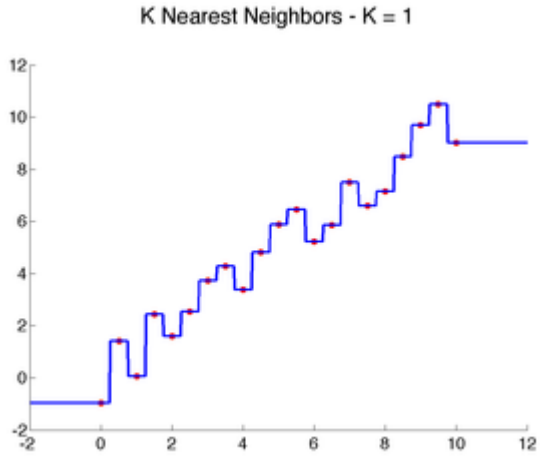
1. Given training data $D = \{\mathbf{x}_i, y_i\}$, distance function d(·,·) and input x
2. Find $\{j_1, \ldots, j_K\}$ closest examples with respect to d(x,·)
   - Regression: if y∈R, return average: $\frac{1}{K}\sum_{k=1}^{K} y_{j_k}$
   - Classification: if y∈±1, return majority: $sign(\sum_{k=1}^{K} y_{j_k})$

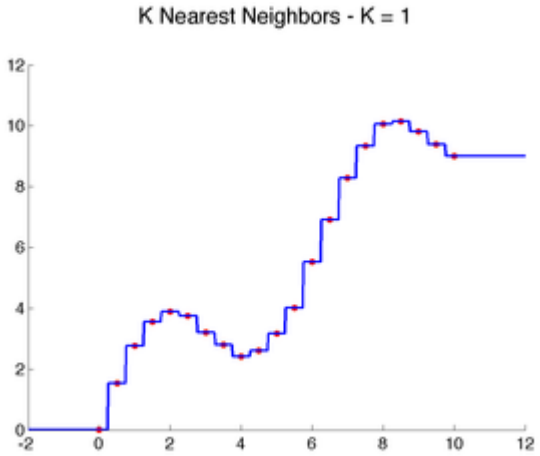


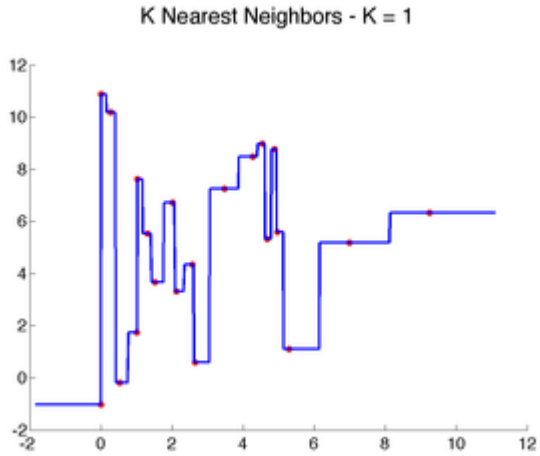


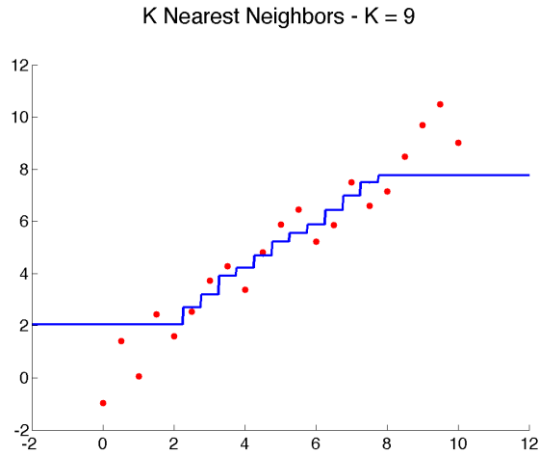


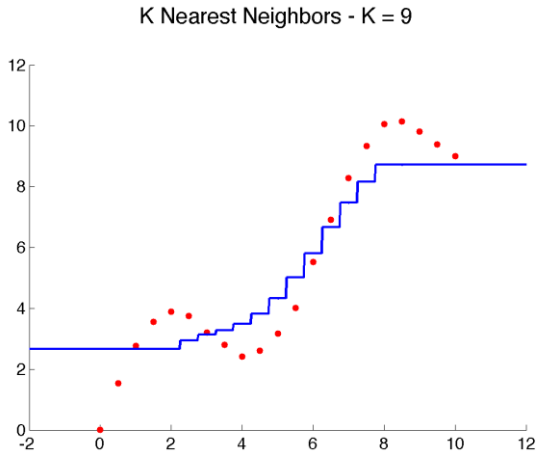
K-Nearest Neighbors

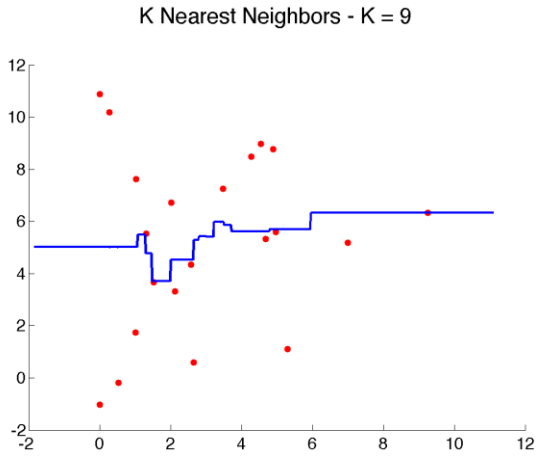




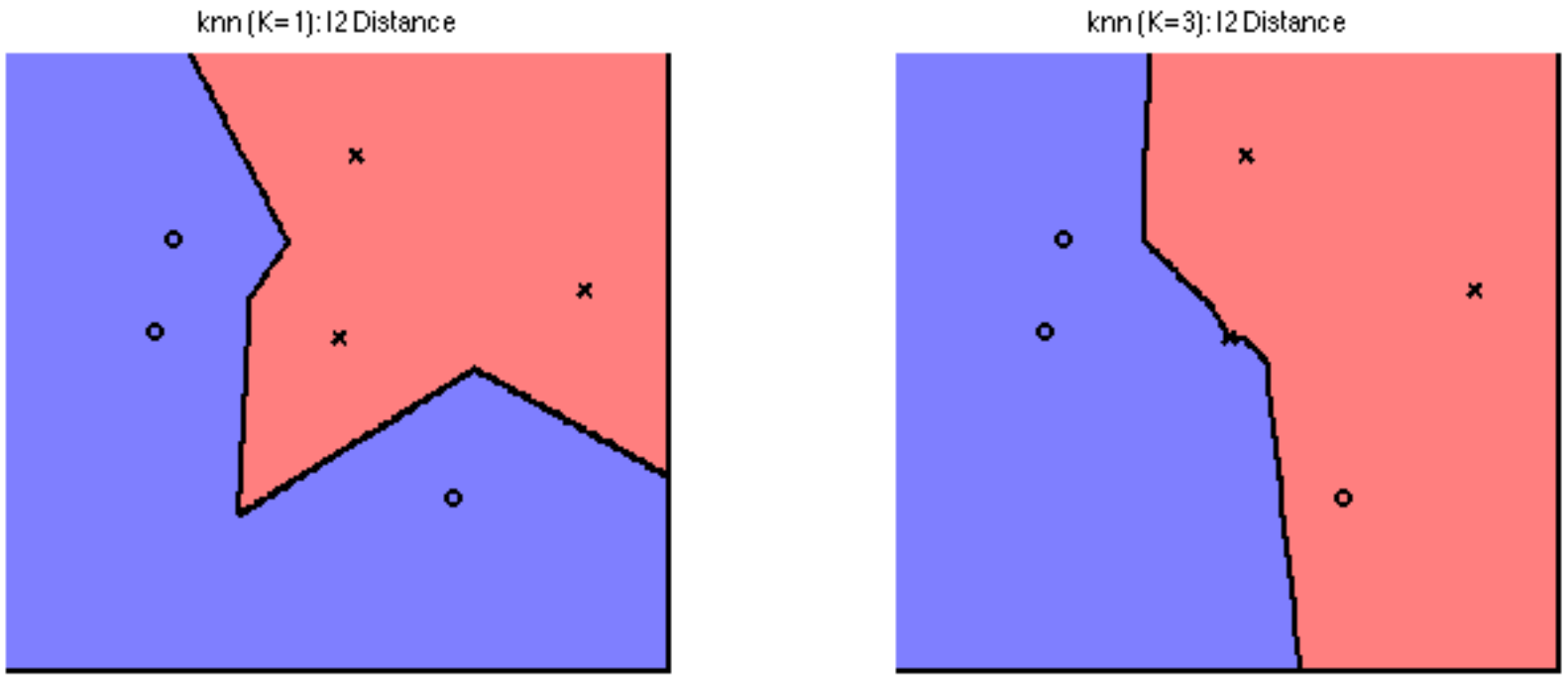








K-Nearest Neighbors



bias-variance tradeoff

Kernel Regression

Kernel Regression

- Shortcomings of the K-NN method?
  - all neighbors receive equal weight
  - the number of neighbors must be chosen globally
- Kernel regression addresses these issues by:
  - Instead of selected nearest neighbors,
  - all neighbors are used, but with different weights.
  - Closer neighbors receive higher weight.
- The weighting function is called a **kernel**
  - kernel measures similarity between examples.

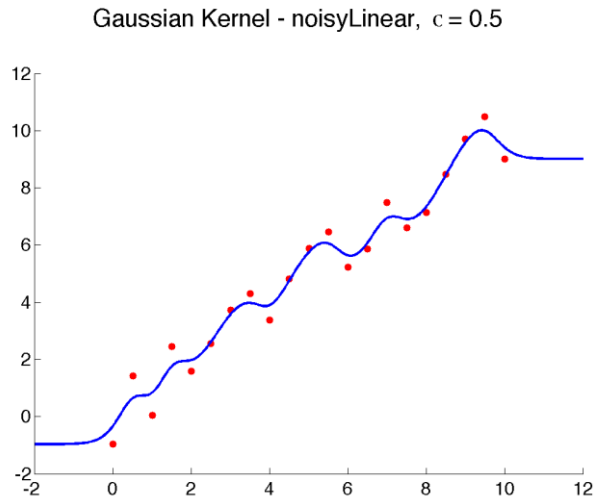
Kernel Regression

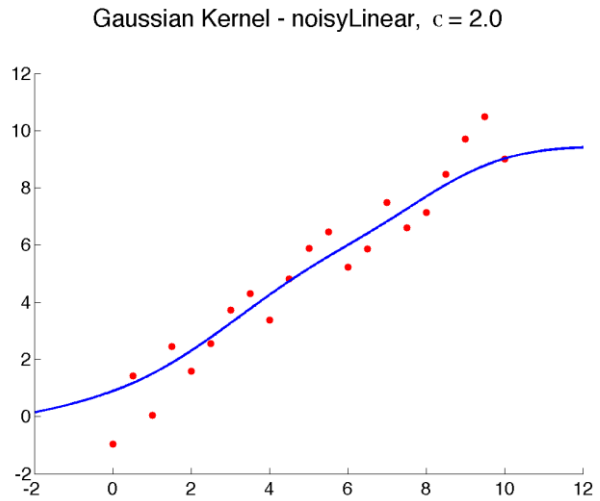
- **Gaussian-type kernel** $K(\mathbf{x}, \mathbf{x}_i) = \exp\{\frac{-d^2(\mathbf{x},\mathbf{x}_i)}{\sigma^2}\}$
  - **In which the width parameter (σ) determines how quickly the influence of neighbors falls off with distance.**
- **Kernel Regression/Classification Algorithm**
  - **Given training data** $D = \{\mathbf{x}_i, y_i\}$, **Kernel function K(·,·) and input x**
  - **regression: if y∈R, return weighted average:** $\frac{\sum_{i=1}^{n} K(\mathbf{x},\mathbf{x}_i) y_i}{\sum_{i=1}^{n} K(\mathbf{x},\mathbf{x}_i)}$
  - **classification: if y∈±1, return weighted majority:**

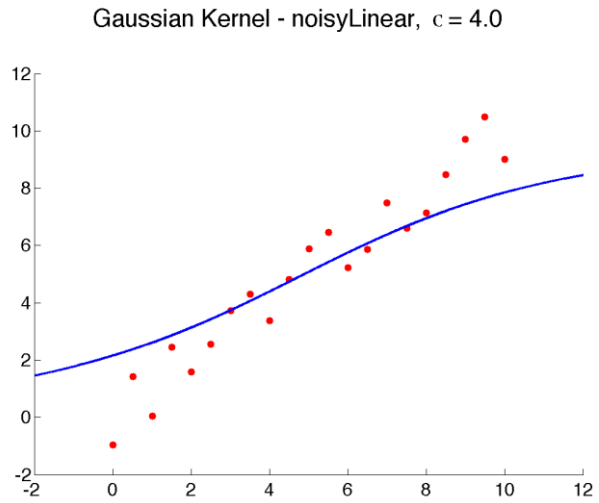
$$sign(\sum_{i=1}^{n} K(\mathbf{x}, \mathbf{x}_i) y_i)$$

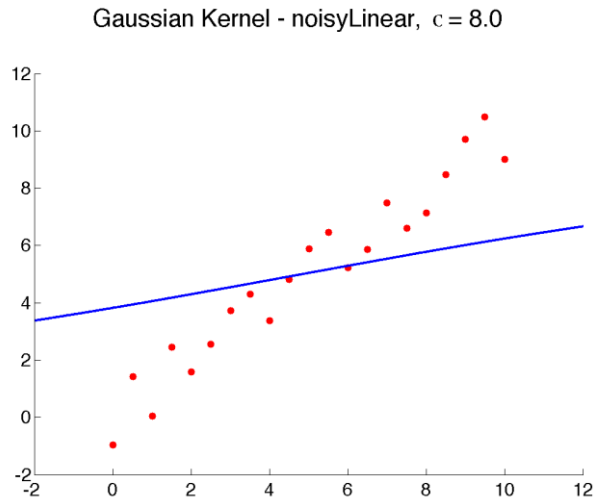
Kernel Regression

Choosing the right width is extremely important to get the bias right

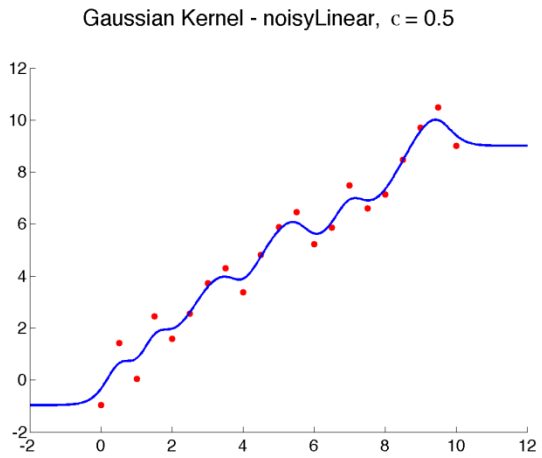


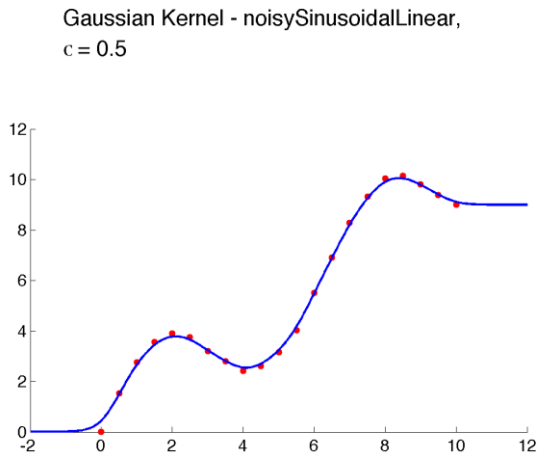


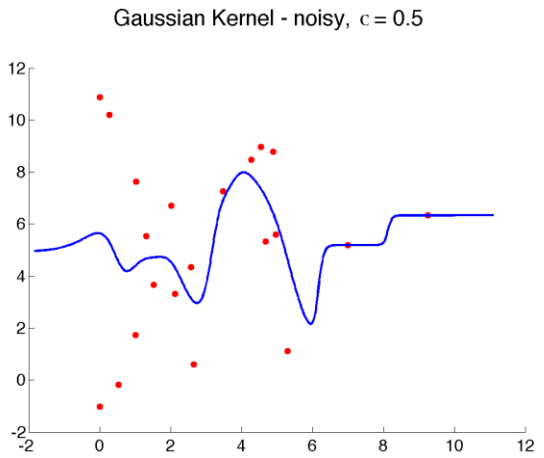


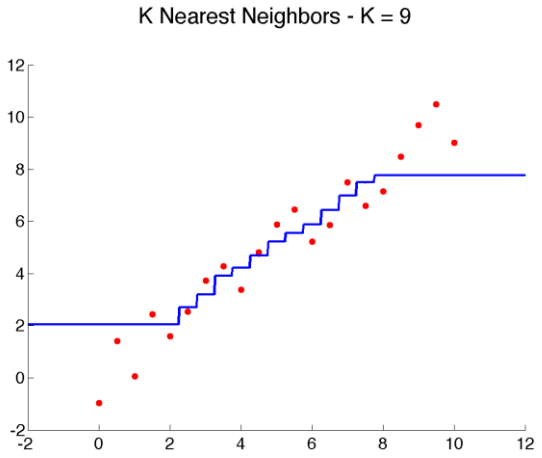


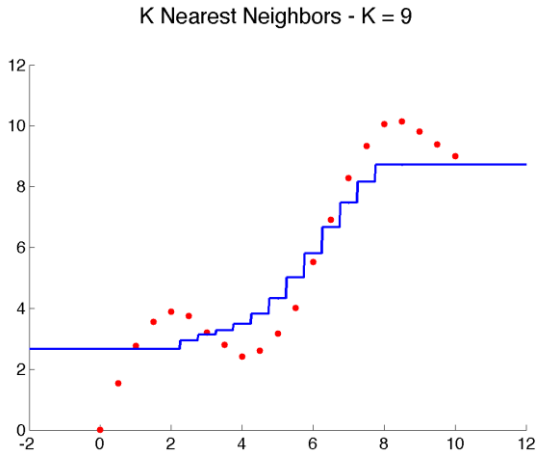
Kernel Regression

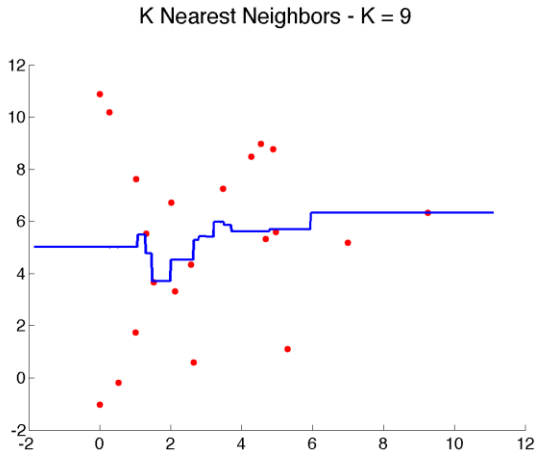




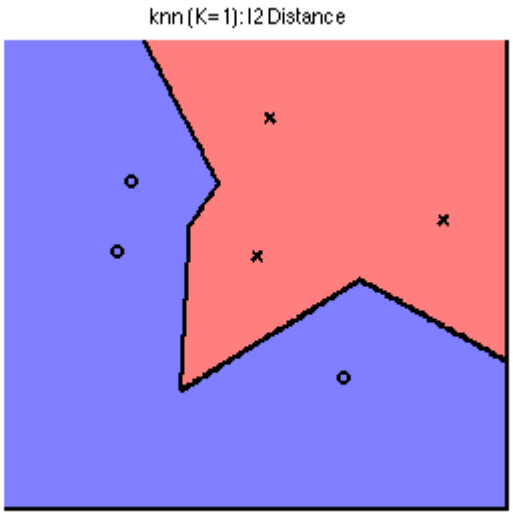


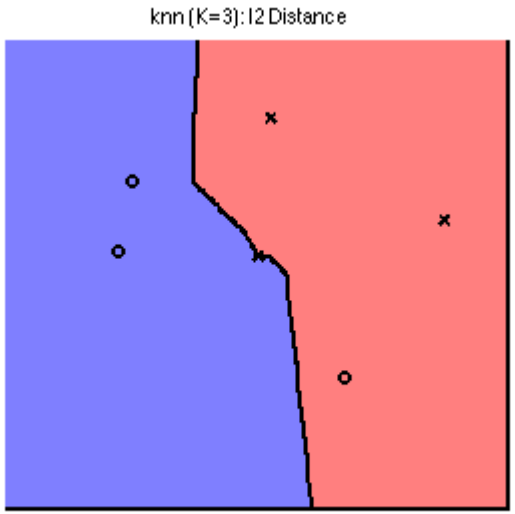


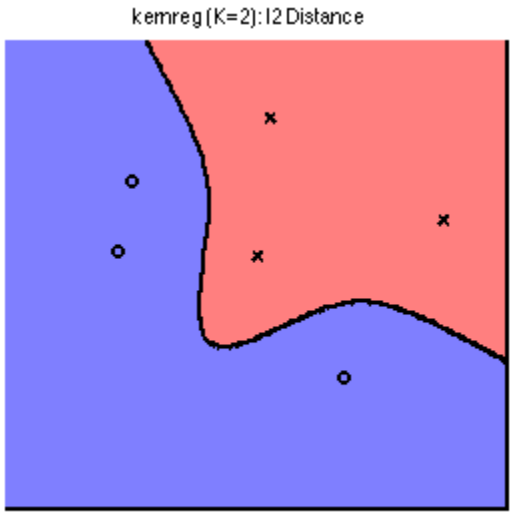




Kernel Regression







bias-variance tradeoff


UESTC
1956

Learning system model

核范数：是A的奇异值之和。

$$||A||_* = \sum_{i=1}^{n} \lambda_i$$