Linear Regression

Wednesday, April 5, 2017 9:16 PM

Linear regression . fit

$$\hat{\beta} = \underbrace{(X^T X)^{-1}}_{} X^T y$$ } matrix multiply

1. matrix multiply
2. matrix solve

1. Matrix multiply (X is $n \times p$)

$$(X^T X)_{jk} = \sum_i X_{ij} X_{ik} \quad - \quad O(n)$$

Computational complexity:

$$O(p^2 n)$$

2. Matrix solve - Cholesky decomp.

Computational complexity:

$$O(p^3)$$

Total Regression Fit Complexity:

$$O(p^3 + p^2 n)$$

Linear Regression, predict

$$\hat{f}(x^*) = x^{*T}\hat{\beta}$$

$\mathcal{O}(p)$ time

- predict is fast $\mathcal{O}(p)$
  fit is slow $\mathcal{O}(p^3 + p^2 n)$

- $X^T X$ needs to be invertible
  (impossible if $p > n$!)

- $\hat{f}$ is linear

- Gauss-Markov theorem

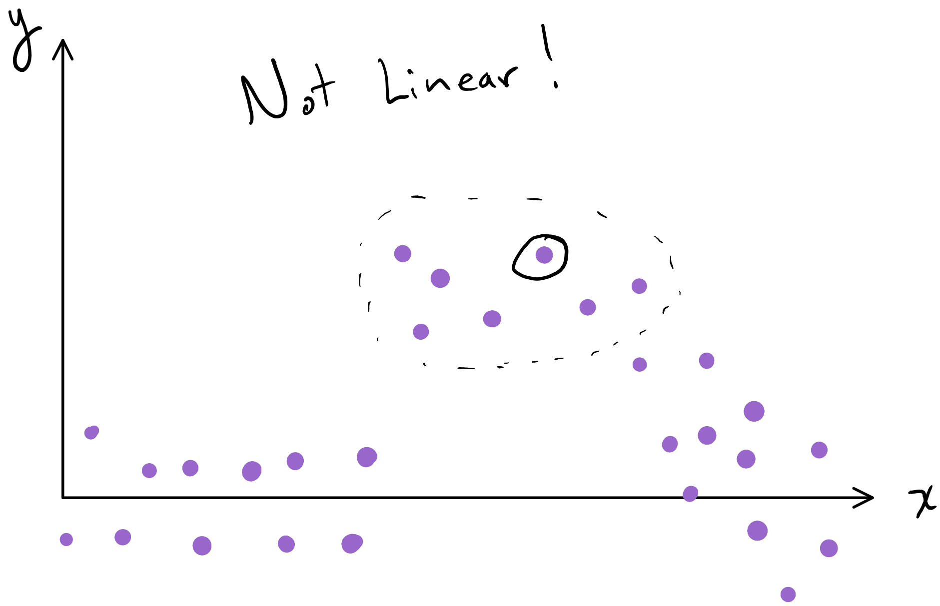
If $y_i = x_i^T \beta + \varepsilon_i$ with $\mathbb{E}\varepsilon_i = 0$
$\mathbb{V}\varepsilon_i = \sigma^2$ then OLS is

unbiased: $\mathbb{E}\hat{\beta} = \beta$

and is the unbiased estimator with minimum variance.

K Nearest Neighbors

Wednesday, April 5, 2017 10:14 PM



Given a metric $d(x, x')$ then the k-nearest neighbors of x in $\{x_i\}_{i=1}^{n}$ is $x_{m_1}, \dots, x_{m_k}$ s.t.

$$d(x_{m_1}, x) \leq d(x_{m_2}, x) \leq \dots \leq d(x_{m_n}, x)$$

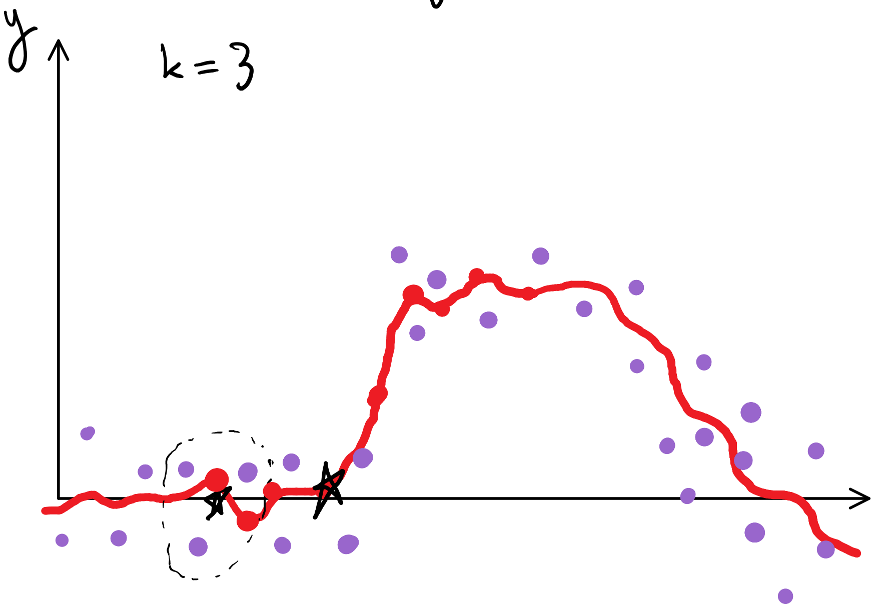
kNN methods fit a model for predicting x using the kNN dataset $\{x_{m_j}, y_{m_j}\}_{j=1}^{k}$.

Regression : $\hat{f}(x) = \frac{1}{k} \sum_{j=1}^{k} y_{m_j}$

Classification : $\hat{f}(x) = \mathbb{1}\left\{\frac{1}{k} \sum_{j=1}^{k} y_{m_j} > \frac{1}{2}\right\}$

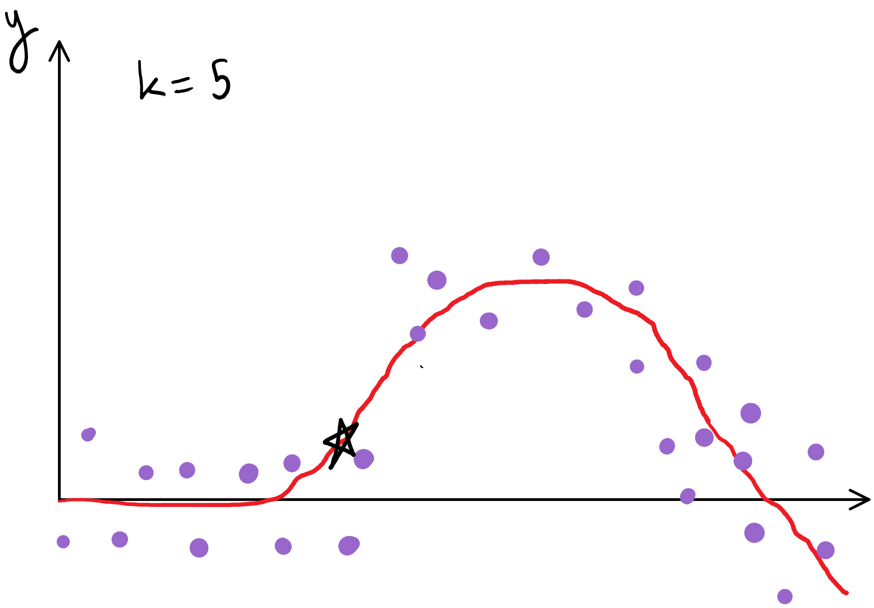
▷ randomized rounding if $= \frac{1}{2}$

y



low bias

high variance



high bias

low variance

Bias-Variance Tradeoff

Wednesday, April 5, 2017 11:08 PM

$\{y_i, x_i\}$ iid

$$\mathbb{E}\left[\frac{1}{k}\sum_{j=1}^{k} y_{m_j} \mid x\right] = \frac{1}{k}\sum_{j=1}^{k} \mathbb{E}[y \mid x_{m_j}]$$

$$\mathbb{V}\left[\frac{1}{k}\sum_{j=1}^{k} y_{m_j} \mid x\right] = \frac{1}{k^2}\sum_{j=1}^{k} \mathbb{V}[y \mid x_{m_j}] = \frac{1}{k}\sigma^2$$

Bias: $\mathbb{E}[\hat{f}(x)] - \mathbb{E}[y \mid x] = \frac{1}{k}\sum_{j=1}^{k} \mathbb{E}[y \mid x_{m_j}] - \mathbb{E}[y \mid x]$

as $d(x_{m_j}, x) \nearrow$ $\left|\mathbb{E}[y \mid x_{m_j}] - \mathbb{E}[y \mid x]\right| \nearrow$

So, typically as $k \nearrow$ bias $\nearrow$

and

as $k \nearrow$ Variance $\searrow$

this is the bias-variance tradeoff

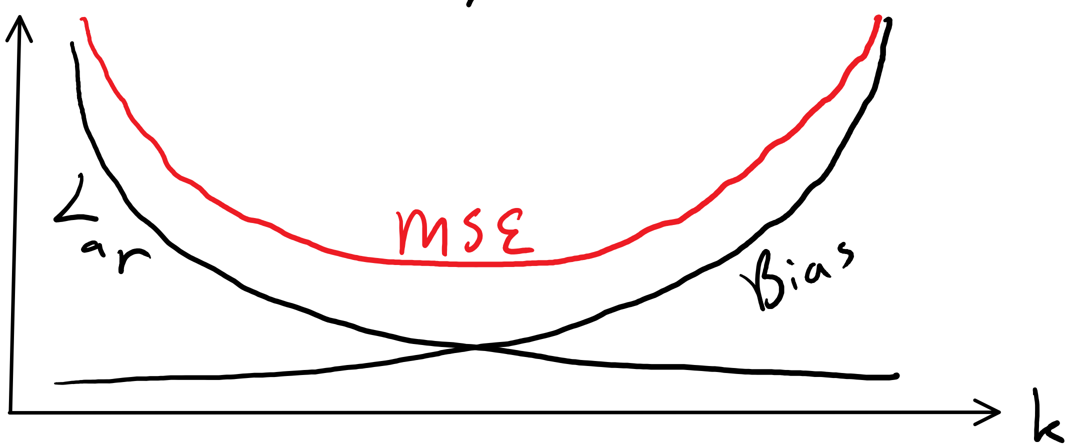
$$MSE = \mathbb{E}\left(\hat{f}(x) - \mathbb{E}[y \mid x]\right)^2 \quad \text{(True Risk)}$$

$$= \mathbb{E}\left(\hat{f}(x) - \mathbb{E}\hat{f}(x) + \mathbb{E}\hat{f}(x) - \mathbb{E}[y \mid x]\right)^2$$

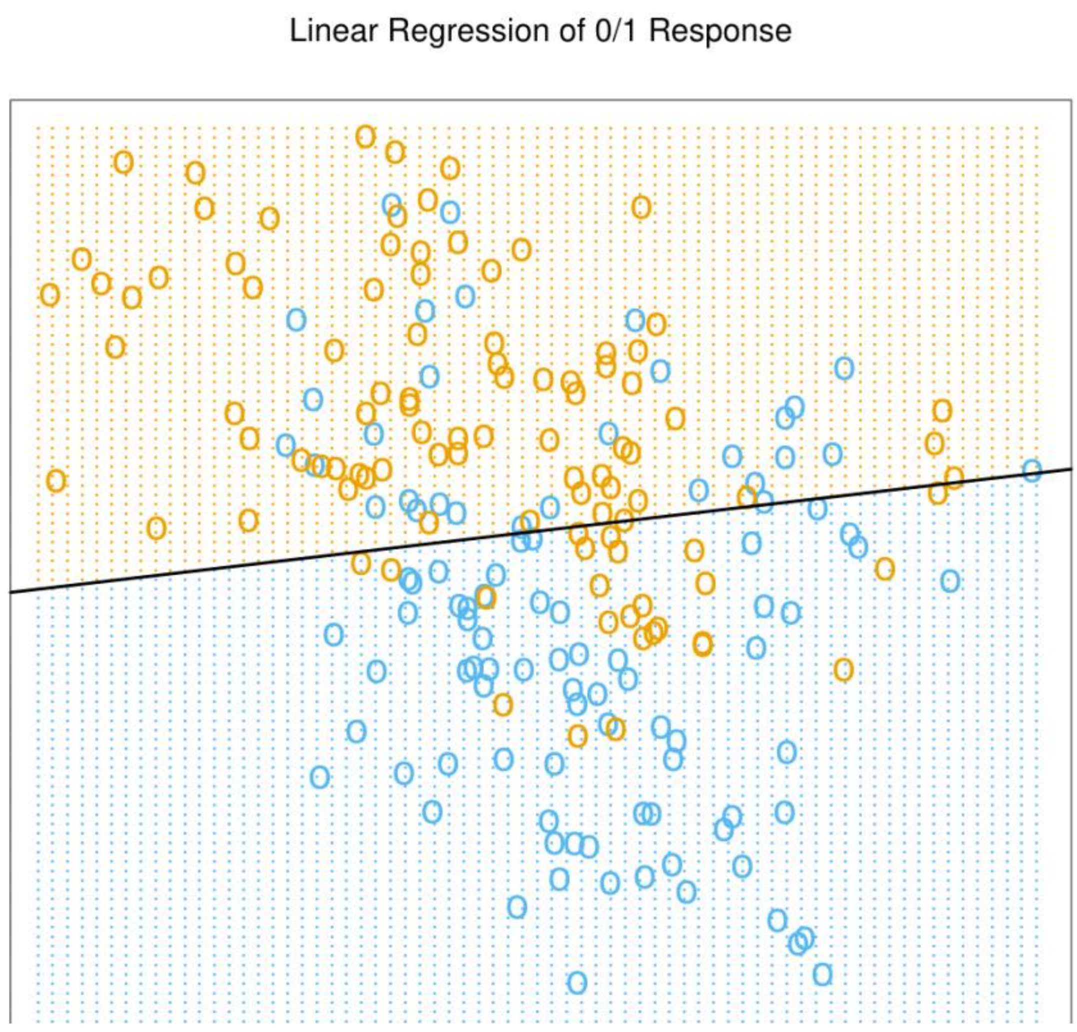
$$= \mathbb{E}\left(\hat{f}(x) - \mathbb{E}\hat{f}(x)\right)^2 + \left(\mathbb{E}\hat{f}(x) - \mathbb{E}[y \mid x]\right)^2$$

$$+ 2\,\underbrace{\mathbb{E}\left(\hat{f}(x) - \mathbb{E}\hat{f}(x)\right)}_{0} \cdot \left(\mathbb{E}\hat{f}(x) - \mathbb{E}[y|x]\right)$$

$$= \text{bias}^2 + \mathbb{V}\hat{f}(x)$$



ESL Ch. 2



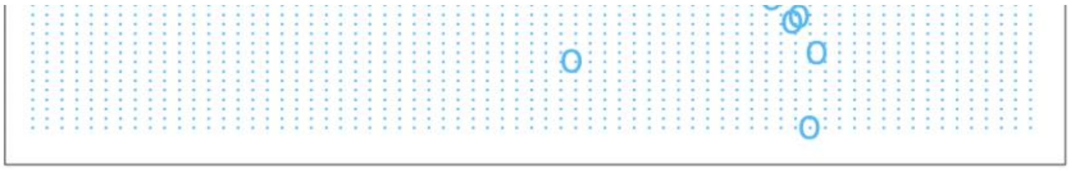


FIGURE 2.1. *A classification example in two dimensions. The classes are coded as a binary variable (*`BLUE` $= 0$*,* `ORANGE` $= 1$*), and then fit by linear regression. The line is the decision boundary defined by* $x^T\hat{\beta} = 0.5$*. The orange shaded region denotes that part of input space classified as* `ORANGE`*, while the blue region is classified as* `BLUE`*.*

1−Nearest Neighbor Classifier

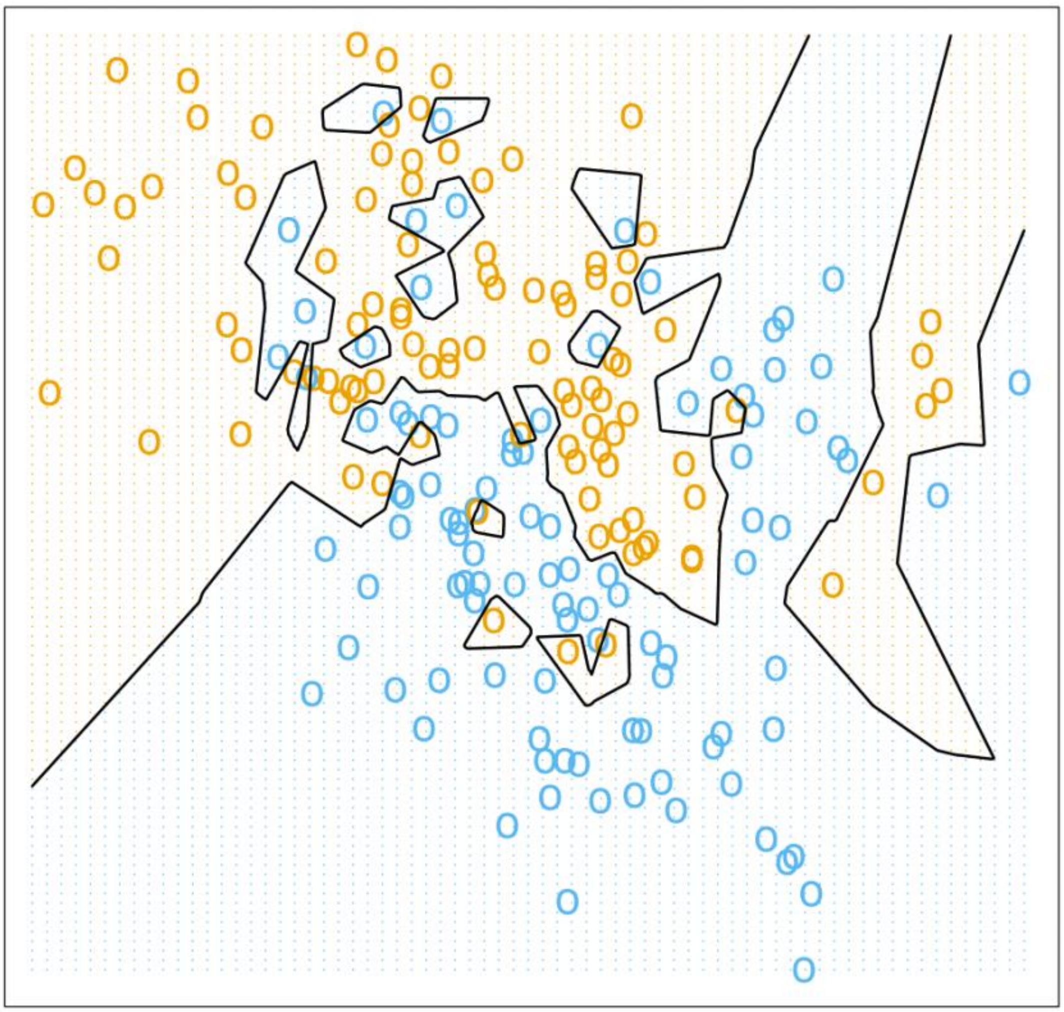


FIGURE 2.3. *The same classification example in two dimensions as in Figure 2.1. The classes are coded as a binary variable (*`BLUE` $= 0$*,* `ORANGE` $= 1$*), and then predicted by* 1*-nearest-neighbor classification.*

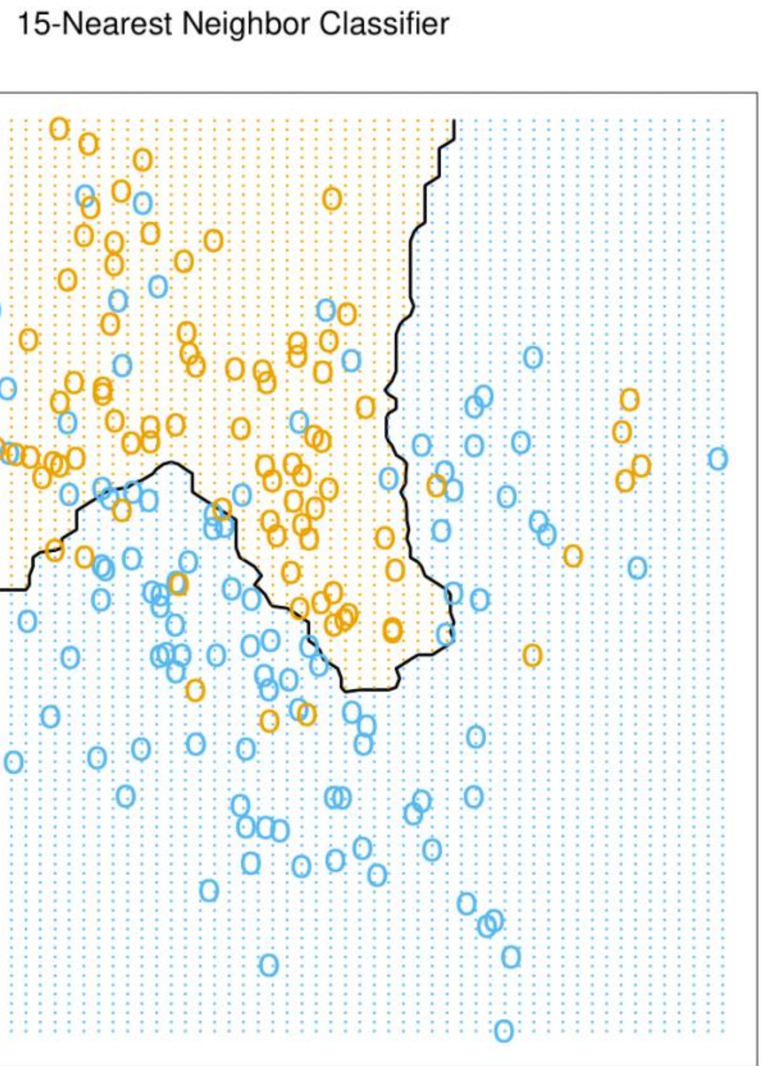


FIGURE 2.2. *The same classification example in two dimensions as in Figure 2.1. The classes are coded as a binary variable* (BLUE $= 0$, ORANGE $= 1$) *and then fit by* 15*-nearest-neighbor averaging as in (2.8). The predicted class is hence chosen by majority vote amongst the* 15*-nearest neighbors.*