Day 15 - 1 November - Cross Validation and K-Nearest Neighbor

Agenda:

- Cross Validation
- K-Nearest Neighbor

Model Validation

Suppose you have multiple trained predictors you are choosing between. How do you select the best one?

Eg Ridge regression

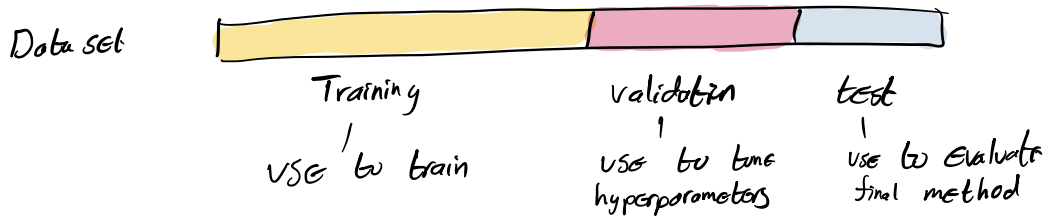
parameter

$$\min_{\theta} \|y - X\theta\|^2 + \lambda \|\theta\|^2$$

Which value of λ should you choose?

λ: hyperparameter

Ideally, use validation data



$$\min_{\lambda} \frac{1}{m} \sum_{i=1}^{m} \ell\left(y_i^{val}, f_{\lambda}(x_i^{val})\right)$$

f_{λ}: predictor w/ hyperparameter λ

Challenges: Need data for validation,
Need independent data for test
data is often expensive

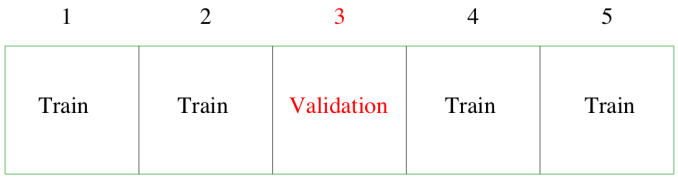
K-fold Cross Validation

Chap 7 of Hastie

7.10.1 K-Fold Cross-Validation

Ideally, if we had enough data, we would set aside a validation set and use it to assess the performance of our prediction model. Since data are often scarce, this is usually not possible. To finesse the problem, K-fold cross-validation uses part of the available data to fit the model, and a different part to test it. We split the data into K roughly equal-sized parts; for example, when $K = 5$, the scenario looks like this:

"folds"



For the kth part (third above), we fit the model to the other $K-1$ parts of the data, and calculate the prediction error of the fitted model when predicting the kth part of the data. We do this for $k = 1, 2, \ldots, K$ and combine the K estimates of prediction error.

Here are more details. Let $\kappa : \{1, \ldots, N\} \mapsto \{1, \ldots, K\}$ be an indexing function that indicates the partition to which observation i is allocated by the randomization. Denote by $\hat{f}^{-k}(x)$ the fitted function, computed with the kth part of the data removed. Then the cross-validation estimate of prediction error is

$$\mathrm{CV}(\hat{f}) = \frac{1}{N}\sum_{i=1}^{N} L(y_i, \hat{f}^{-\kappa(i)}(x_i)). \tag{7.48}$$

avg over each datapoint

train on all data not in the fold containing i^{th} data point

Equivalently: average over K folds of the average loss on each fold when trained on all other folds

Typical choices of K are 5 or 10 (see below). The case $K = N$ is known as *leave-one-out* cross-validation. In this case $\kappa(i) = i$, and for the ith observation the fit is computed using all the data except the ith.

Given a set of models $f(x, \alpha)$ indexed by a tuning parameter α, denote by $\hat{f}^{-k}(x, \alpha)$ the αth model fit with the kth part of the data removed. Then for this set of models we define

$$\mathrm{CV}(\hat{f}, \alpha) = \frac{1}{N} \sum_{i=1}^{N} L(y_i, \hat{f}^{-\kappa(i)}(x_i, \alpha)). \tag{7.49}$$

The function $\mathrm{CV}(\hat{f}, \alpha)$ provides an estimate of the test error curve, and we find the tuning parameter $\hat{\alpha}$ that minimizes it. Our final chosen model is $f(x, \hat{\alpha})$, which we then fit to all the data.

Hyper parameter tuning

How to choose # of folds?

If $K=N$,
- have to solve N problems
  $\Rightarrow$ expensive.
- low bias, high variance

If K too small, not enough data used for training model.

Compromise $k = 5$ or 10

7.10.2 The Wrong and Right Way to Do Cross-validation

Consider a classification problem with a large number of predictors, as may arise, for example, in genomic or proteomic applications. A typical strategy for analysis might be as follows:

1. Screen the predictors: find a subset of "good" predictors that show fairly strong (univariate) correlation with the class labels
2. Using just this subset of predictors, build a multivariate classifier.
3. Use cross-validation to estimate the unknown tuning parameters and to estimate the prediction error of the final model.

Is this a correct application of cross-validation?

The predictor screening step uses all of the data. When training each model for cross-validation, this means each of those models will have trained on the test data that was left out.

Here is the correct way to carry out cross-validation in this example:

1. Divide the samples into K cross-validation folds (groups) at random.
2. For each fold $k = 1, 2, \ldots, K$
    (a) Find a subset of "good" predictors that show fairly strong (univariate) correlation with the class labels, using all of the samples except those in fold k.
    (b) Using just this subset of predictors, build a multivariate classifier, using all of the samples except those in fold k.
    (c) Use the classifier to predict the class labels for the samples in fold k.

The error estimates from step 2(c) are then accumulated over all K folds, to produce the cross-validation estimate of prediction error. The lower panel

In general, with a multistep modeling procedure, cross-validation must be applied to the entire sequence of modeling steps. In particular, samples must be "left out" before any selection or filtering steps are applied. There is one qualification: initial *unsupervised* screening steps can be done before samples are left out. For example, we could select the 1000 predictors with highest variance across all 50 samples, before starting cross-validation. Since this filtering does not involve the class labels, it does not give the predictors an unfair advantage.

K nearest neighbors (KNN)

- Method for regression & classification
- Idea: Find k closest training examples and average them / do a majority vote

Classification

Training data: $\{(X_i, y_i)\}$ w/ $X_i \in \mathbb{R}^d$, $y_i \in \{1 \dots M\}$

Predictor:

$$P(y=c \mid \mathcal{X}, S, k) = \frac{1}{k} \sum_{i \in N_k} 1_{y_i = c}$$

S = train data

N_k = set of indices i of the k nearest X_i to $\mathcal{X}$

$\hat{y}(\mathcal{X})$ = most common value of y_i among k nearest values of x_i to $\mathcal{X}$.

Distance measured by ℓ_2 norm (or alternative of your choice)

2 class example:

K=1

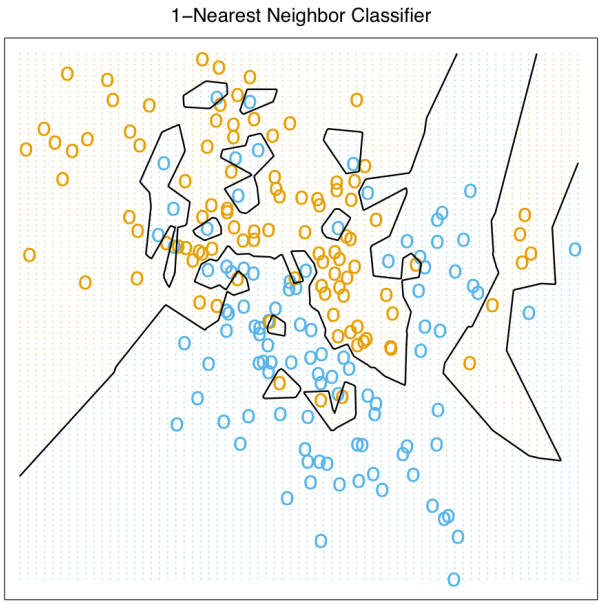
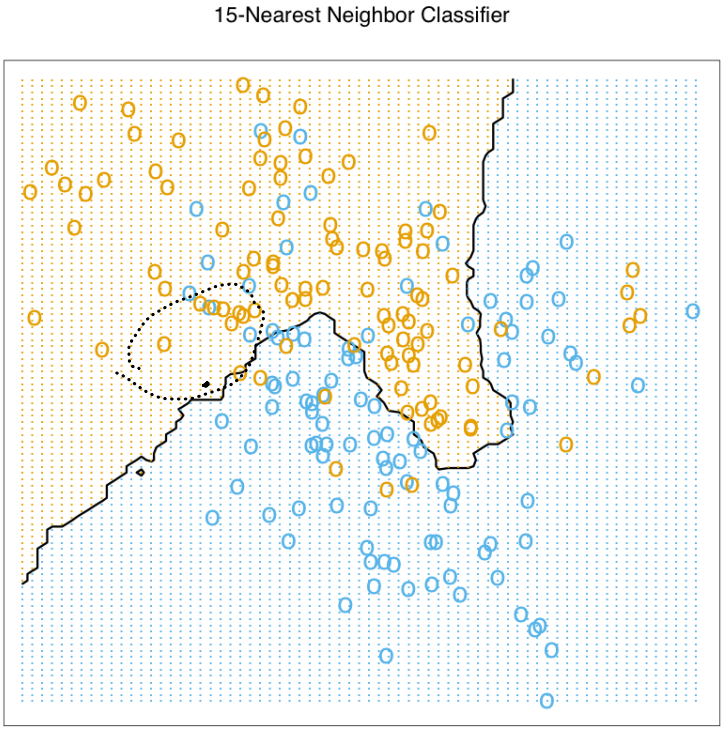
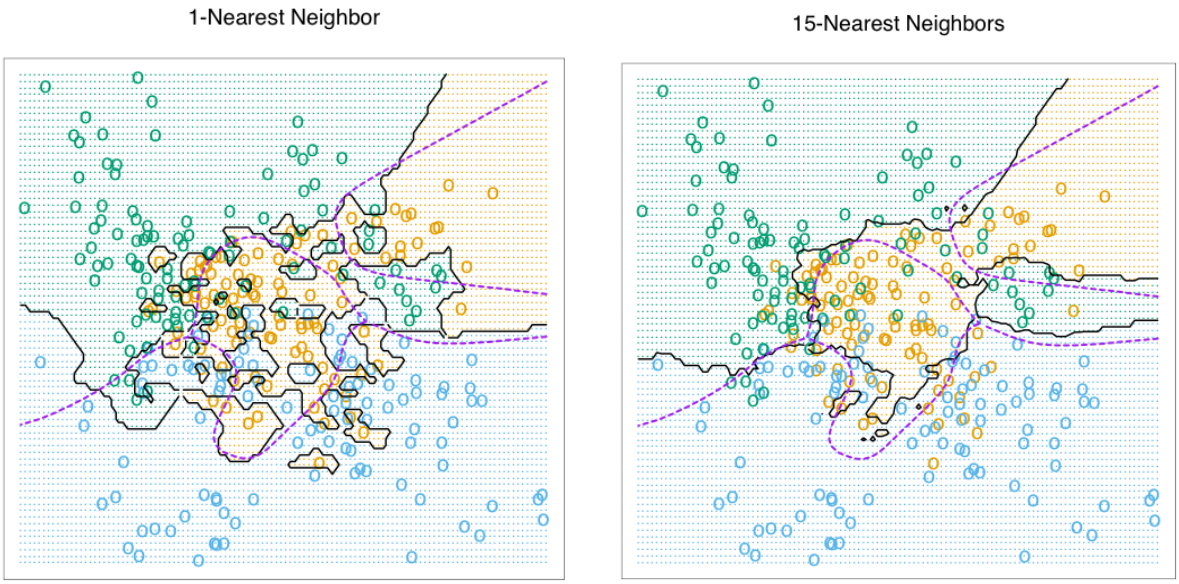


FIGURE 2.3. *The same classification example in two dimensions as in Figure 2.1. The classes are coded as a binary variable* (BLUE $= 0$, ORANGE $= 1$)*, and then predicted by* 1*-nearest-neighbor classification.*

K=15



3 Class Example



— Decision boundary for Bayes Classifier

Bayes Classifier

Assume (X, y) follows a joint distribution

$$\hat{y}(x) = \underset{y}{\operatorname{argmax}} \; P(y \mid x)$$

/

Bayes Classifier gives most likely class of y given x. Can't compute this as underlying dist is unknown.

KNN and bias-variance tradeoff

What is the effect of different values of k?

If k too small:

Bias: High or low

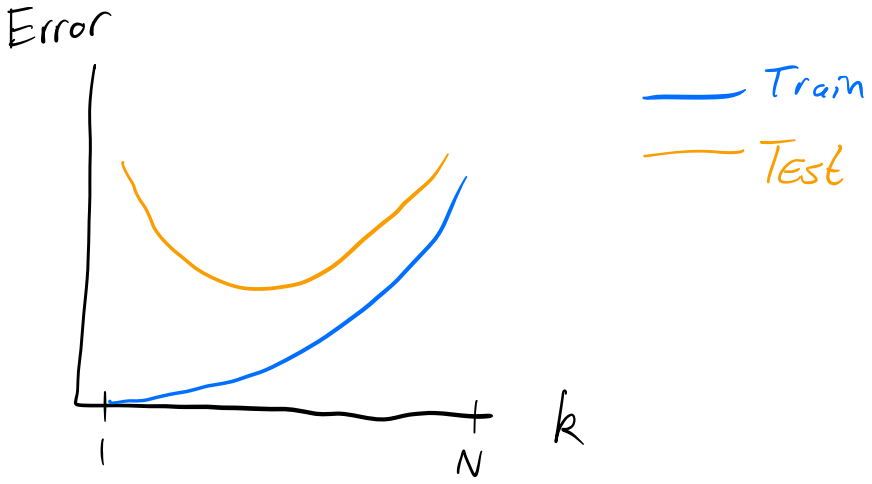
Variance: High or low

If k too large?

Bias: High or low

Variance: High or low

Plot: Train Error vs k & Test Error vs k



How do you find k?

Regression with k nearest neighbors

Training data: $\{(x_i, y_i)\}_{i=1,\dots,n}$ w/ $x_i \in \mathbb{R}^d$, $y_i \in \mathbb{R}^m$

Prediction: $\hat{y} = \frac{1}{k} \sum_{i \in N_k} y_i$

N_k = set of indices i of the k nearest x_i to x

Bias-Variance tradeoff for regression:

Consider a model $y_i = h(x_i) + \varepsilon_i$ w/ $\varepsilon_i \sim N(0, \sigma^2)$

(h = true response function)

Bias: $\left(\mathbb{E}_y \hat{y}(x) - h(x) \right)^2 = \left(\frac{1}{k} \sum_{i \in N_k} \mathbb{E}_y(y_i) - h(x) \right)^2$

$= \left(\frac{1}{k} \sum_{i \in N_k} h(x_i) - h(x) \right)^2$

If k fixed and $n \to \infty$,

$\frac{1}{k} \sum_{i \in N_k} h(x_i) \approx h(x)$ low bias

If $k = n$ and $n \to \infty$ high bias

$$\frac{1}{k} \sum_{i \in N_k} h(x_i) \approx \text{average of } h$$

Variance:

$$Var(\hat{y}(x)) = Var\left[\frac{1}{k} \sum_{i \in N_k} y_i \right]$$

$$= \frac{1}{k^2} Var \sum_{i \in N_k} y_i$$

$$= \frac{1}{k^2} \; k \; Var(y_i)$$

$$= \frac{\sigma^2}{k}$$

Small k: high variance

Large k: low variance

ISSUE: KNN may fail w/ high dimensional data.

Why? Because "all points in high dimensional space are roughly equidistant"

CURSE OF DIMENSIONALITY

Illustration:

Consider a sphere of radius 1 in $\mathbb{R}^d$. What fraction of the sphere's volume is located inside radius $1-\varepsilon$?

$$\frac{(1-\varepsilon)^d}{1^d} = (1-\varepsilon)^d$$

which is very small for large d!

A randomly generated point in a ball of radius r in high dim space is almost certainly near its boundary.