The elements of statistical learning notes

Frank Ji

June 8, 2018

1. Introdution

Learning problems
Supervised:
$features$
Outcome: Quantitative or categorical
Training set
Model or *Learner* to predict *outcome* based on *input* $features$
Unsupervised:
No outcome measure, describe the association or pattern
Examples:
Email Spam(SL), Prostate Cancer(SL), Handwritten Digit Recognition(SL), DNA Expression Microarrays(USL)

2. Overview of Supervised Learning

2.1 Introduction

2.1 Variable Types and Terminolgy

Variable type
Qualitative *dummy variables*, labels
Quantitative
Ordered Categorical (See Chapter 4.)
Inputs and outcomes
Predictors and *Responses*
Independent variables and *dependent variables*
X and Y *or* G
The prediction tasks
regression for quantitative outputs
classification for qualitative outputs
Notations
$N \times p$ matrix X for inputs, its components as X_j
lowercase for observed data Vectors will not be bold unless they have N components
$\hat{Y}$ or $\hat{G}$ for outputs
A set of measurements (x_i, y_i) or (x_i, g_i), $i = 1, ..., N$ known as training data

2.3 Two Simple Approaches to Prediction: Least Squares and Nearest Neighbors

2.3.1 Linear Models and Least Squares

$$\hat{Y} = \hat{\beta}_0 + \sum_{j=1}^{p} X_j \hat{\beta}_j$$

$$\hat{Y} = X^T \hat{\beta}$$

$\hat{\beta}_0$, namely intercept or bias in machine learning, is included in $\hat{\beta}$.

$f'(\mathbf{X}) = \beta$ indicates the steepest uphill direction.

$RSS(\beta) = (\mathbf{y} - \mathbf{X}\beta)^T(\mathbf{y} - \mathbf{X}\beta)$
Normal equation: $\mathbf{X}^T(\mathbf{y} - \mathbf{X}\beta) = 0$
Solution: $\hat{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$

2.3.2 Nearest-Neighbor Methods

$$\hat{Y}(x) = \frac{1}{k}\sum_{x_i \in N_k(x)} y_i$$

Error on training data increases with k. Effective number of parameters of k-nearest neighbors is N/k.

2.3.3 From Least Squares to Nearest Neighbors

Least Square:
Validity of Linear decision boundary.
Low variance and potentially high bias.
kNN:
No stringent assumptions about data.
high variance and low bias
Enhancement:
Kernel methods
Emphasis on some variables
Local regression and locally weighted least squares
Linear models fit to a basis expansion
Projection pursuit and neural network models

2.4 Statistical Decision Theory

Loss functions like *squared loss* $L(Y, f(X)) = (Y - f(X))^2$
A criterior for choosing f, the expected prediction error

$$\begin{aligned} EPE(f) &= E(Y - f(X))^2 \\ &= \int [y - f(x)]^2 Pr(dx, dy) \end{aligned}$$

$$EPE(f) = E_X E_{Y|X}([Y - f(x)]^2|x)$$
$$f(X) = argmin_c E_{Y|X}([Y - c]^2|X = x)$$
$$f(x) = E(Y|X = x)$$

For nearest neighbors,

$$\hat{f}(x) = Ave(y_i|x_i \in N_k(x))$$

As $N, k \to \infty$ *s.t.* $k/N \to 0$,

$$\hat{f}(x) \to E(Y|X = x)$$

For linear regression,

$$\beta = [E(XX^T)]^{-1}E(XY)$$

Both methods approximate conditional expectations by averages, although model assumptions are different, namely global and local.

Additive models

$$f(X) = \sum_{j=1}^{p} f_j(X_j)$$

High dimensionality can be dealt with by additivity assumption

In $L_1 : E|Y - f(x)|$ case,

$$\hat{f}(x) = median(Y|X = x)$$

which is more robust.

For Categorical outcome, loss function can be representedby a $K \times K$ matrix L, where, $K = card(G)$.

$$EPE = E[L(G, \hat{G}(x))]$$

$$EPE = E_X \sum_{k=1}^{K} L[G_k, \hat{G}(X)] Pr(G_k|X)$$

$$\hat{G}(x) = argmin_{g \in G} \sum_{k=1}^{K} L(G_k, g) Pr(G_k|X = x)$$

With the 0-1 loss function:

$$\hat{G}(x) = argmin_{g \in G}[1 - Pr(g|X = x)]$$

$$\hat{G}(x) = G_k \text{ if } Pr(G_k|X = x) = \max_{g \in G} Pr(g|X = x)$$

namely fetch the label with the highest posterior probability. For kNN classifier, it will just take the major vote. More details modeling $Pr(G|X)$ in Chapter 4.

June 8, 2018

2.5 Local Methods in High Dimensions

The *curse of dimensionality*

Hypercube example:

if we want to get a fraction r of local subspace in p-dimension unit space using hypercubes to estimate the outcomes. The expected edge length will be like. $e_p(r) = r^{\frac{1}{p}}$

When p is large, it goes to 1. If reduce the r, the variance of estimate will be high.

p-dimension ball example: Uniformly distributed points in p-dimension ball. The median of the closest point to origin will be:

$$d(p, N) = (1 - \frac{1}{2}^{\frac{1}{N}})^{\frac{1}{p}}$$

Most data points will be close to boundary. Prediction is much more difficult near the edges.

The sampling density is propotional to $N^{\frac{1}{p}}$. N^p samples will be needed for single input problem.

Another example:

$$Y = f(x) = e^{-8||X||^2}$$

Denoting the training set as τ
By the $bias - variance\ decomposition$:

$$\begin{aligned} MSE(x_0) &= E_\tau[f(x_0) - \hat{y}_0]^2 \\ &= E_\tau[\hat{y}_0 - E_\tau(\hat{y}_0)]^2 + [E_\tau(\hat{y}_0) - f(x_0)]^2 \\ &= Var_\tau(\hat{y}_0) + Bias^2(\hat{y}_0) \end{aligned}$$

As p increases, estimate tends to be 0 (most samples will be far from origin), the variance will drop (an artifact of this problem).

The complexity of function goes exponentially with variables and we need to exponentially increase data size to achieve the same performance.

If we know,

$$Y = X^T\beta + \epsilon, \quad \epsilon \sim N(0, \sigma^2)$$
$$EPE(x_0) = \sigma^2 + E_\tau x_0^T (X^T X)^{-1} x_0 \sigma^2 + 0^2$$

Hint: Decompose first and decompose the bias term again on y_0.

Assume $E(X) = 0$, $X^T X \to NCov(X)$

$$\begin{aligned} E_{x_0} EPE(x_0) &\sim E_{x_0} x_0^T Cov(X)^{-1} x_0 \sigma^2/N + \sigma^2 \\ &= tr(Cov(X)^{-1} Cov(x_0))\sigma^2/N + \sigma^2 \\ &= \sigma^2(p/N) + \sigma^2 \end{aligned}$$

Hint: tr(BA) = tr(AB) and for 1×1 matrix C, tr(C) = C.

The results highly depend on assumptions. For example, Bias for linear model is zero and variance can be negligible under right assumption but kNN may also dominate if the assumption is wrong. See (Figure 2.9)

2.6 Statistical Models, Supervised Learning and Function Approximation

if the dimension is high, the nearest neighbors may be close to target point, which can result in large errors.
if special structure exists, this can be used to reduce both the bias and the variance of estimates.

2.6.1 A Statistical Model for the Joint Distribution Pr(X, Y)

Note (X, Y) may not have a deterministic form like $Y = f(X)$. The additive model assume, we can model the departures to form a deterministic relationship via the error ϵ.
Some problems will have a deterministic relationship like some classification problems in machine learning.
Assumptions in $Y = f(X) + \epsilon$ may not hold, for example, errors are $i.i.d.$ $Pr(Y|X)$ can depend on X. Additive error models are typically not used for qualitative outputs G.

2.6.2 Supervised Learning

Learn f by example through a *teacher*. Input data to a learning program and $\hat{f}(x_i)$ will also be generated. The *learningbyexample* process will modify $\hat{f}$ in response to $y_i - \hat{f}(x_i)$ to make artificial and real outputs similar.

2.6.3 Function Approximation

Assume the domain of $f(x)$ is $\mathbb{R}^p$. The function approximation in supervised larning encourages the geometrical concepts of Euclidean spaces and mathematical comcepts of probablistic

inference to be applied to the problem. For a *linearbasisexpansions* given paramter θ and suitable transformations $h(x)$:

$$f_\theta(x) = \sum_{k=1}^{K} h_k(x)\theta_k$$

$$RSS(\theta) = \sum_{i=1}^{N}(y_i - f_\theta(x_i))^2$$

$f_\theta(x)$ can be treated as a surface in $p+1$ space. To approximate the true space by getting closer to the data through minimizing $RSS(\theta)$.
A general principle for estimation is maximum likelihood estimation. The log-probability of the observed sample is:

$$L(\theta) = \sum_{i=1}^{N} logPr_\theta(y_i)$$

For $Y = f_\theta(X) + \epsilon$,

$$Pr(Y|X,\theta) = N(f\theta(X), \sigma^2)$$

$$L(\theta) = -\frac{N}{2}log(2\pi) - Nlog(\sigma) - \frac{1}{2\sigma^2}\sum_{i=1}^{N}(y_i - f_\theta(x_i))^2$$

For multinomial likelihood for $Pr(G|X)$,

$$L(\theta) = \sum_{i=1} Nlogp_{g_i\theta}(x_i)$$

June 8, 2018

2.7 Structured Regression Models

More structured approaches may dominate in performance at specific setting.

Difficulty of the problem

$$RSS(f) = \sum_{i=1}^{N}(y_i - f(x_i))^2$$

Any function $\hat{f}$ passing trough the training points is a solution, which could be poor beyond training samples. If there are multiple observations at each x_i, the risk is limited. The solution will pass through the average values of the y_{il} at each x_i, (hints: take the derivative). If N is sufficiently large and the repeats were gauranteed and densely arranged, the solution might tend to the limiting conditional expection.
For finite N, $RSS(f)$ needs to be restricted to set of functions for eligibility and the ambiguity has simply been transferred to the choice of constrant like *complexity* restrictions. The strength of the constraint is dictated by the neighborhood size. If the local linear fits in very large neighborhoods is almost a globally linear model, and is vert restrictive. However, nny method that attempts to produce locally varying functions in small isotropic neighborhoods will run into problems in high dimensions..

June 8, 2018

2.8 Classes of Restricted Estimators

Roughness Penalty and Bayesian Methods

Regularization method
Penalizing RSS:

$$PRSS(f;\lambda) = RSS(f) + \lambda J(f)$$

For example, *cubicsmoothingspline* for one-dimensional inputs:

$$PRSS(f;\lambda) = \sum_{i=1}^{N}(y_i - f(x_i))^2 + \lambda \int [f''(x)]^2 dx$$

Kernel Methods and Local Regression

The local neighborhood will be specified by a $kernelfunction$ $K_\lambda(x_0, x)$, for example,

$$K_\lambda(x_0, x) = \frac{1}{\lambda} exp[-\frac{||x - x_0||^2}{2\lambda}]$$

in which, λ will control the width of the neigborhood. The simplest form of kernel estimate is the $Nadaraya - Watson$ weighted average,

$$\hat{f}(x_0) = \frac{\sum_{i=1}^{N} K_\lambda(x_0, x_i) y_i}{\sum_{i=1}^{N} K_\lambda(x_0, x_i)}$$

A local regression estimate of $f(x_0)$ as $f_{\hat{\theta}}(x_0)$ where $\hat{\theta}$ minimizes:

$$RSS(f_\theta, x_\theta) = \sum_{i=1} N K_\lambda(x_0, x_i)(y_i - f_\theta(x_i))^2$$

k-nearest neighbors:

$$K_k(x, x_0) = I(||x - x_0|| \leq ||x_{(k)} - x_0||)$$

2.8.3 Basis Functions and Dictionary Methods

The model for f is a linear expandsion of basis functions

$$f_\theta(x) = \sum_{m=1}^{M} \theta_m h_m(x)$$

Radial basis functions, p-dimensional kernels located at particular centroids (Gaussian, for example),

$$f_\theta(x) = \sum_{m=1}^{M} K_{\lambda m}(\mu_m, x)\theta_m$$

A single-layer feed-forward neural network model, with *activation* function $\sigma(x) = 1/(1 + e^{-x})$.

$$f_\theta(x) = \sum_{m=1}^{M} \beta_m \sigma(\alpha^T x + b_m)$$

The directions α_m and the *bias* terms b_m have to be determined. These adaptively chosen basis function methods are also known as *dictionary* methods with searchable dictionary **D**.

2.9 Model Selection and the Bias-Variance Tradeoff

Smoothing or *comlexity* parameter that has to be determined:

1. the multiplier of the penalty term
2. the width of the kernel
3. the number of basis function

In the case of the smoothing spline, RSS may be not a good choice for picking parameter. (window size) k-nearest neighbor regression fit $\hat{f}_k(x_0)$

$$\begin{aligned} EPE_k(x_0) &= E[(Y - \hat{f}_k(x_0))^2 | X = x] \\ &= \sigma^2 + [Bias^2(\hat{f}_k(x_0)) + Var_\tau(\hat{f}_k(x_0))] \\ &= \sigma^2 + [f(x_0) - \frac{1}{k}\sum_{l=1} kf(x_{(l)})]^2 + \frac{\sigma^2}{k} \end{aligned}$$

σ^2 is the *irreducible* error beyond control. Bias term will most likely increase with k if the true function is reasonably smooth. (Consider those not close but still in top k). As k varies (complexity), there is a $bias - variance\ tradeoff$. The model which is too close to training set, is very likely to fail in generalization. If it is not complex enough, it may also underfit.