LECTURER: PROF. HOMRIGHAUSEN SCRIBE: ZEKE WANG

First, we talk about the Setup.

Suppose we have data $\mathcal{D}$= $(X_1, Y_1), (X_2, Y_2), \ldots, (X_n, Y_n)\}$, where $X_i \in \mathbb{R}^p$ are the features (or explanatory variables or predictors or covariates, not the independent variables); $Y_i \in \mathbb{R}$ are the response variables.(not dependent variable). Our goal for this class is to find a way to explain (at least approximately) the relationship between X and Y.

Then we talk about the prediction risk for regression. Given the training data $\mathcal{D}$, we want to predict some independent test data $Z = (X, Y)$. This means forming a $\hat{f}$, which is a function of both the range of X and the training data $\mathcal{D}$, which provides predictions $\hat{Y} = \hat{f}(X)$.

The quality of this prediction is measured via the prediction risk[1]

$$R(\hat{f}) = ¶_{D,Z}(Y - \hat{f}(X))^2.$$

We know that the regression function, $f_*(X) = ¶[Y|X]$, is the best possible predictor. We should note that f_* is *unknown*

There are some notations recap: X is a vector of measurements for each subject; x is a vector of subjects for each measurement,; X_i^j is the j^{th} measurement on the i^{th} subject.

Imposing Linearity

Multiple Regression

If we specify the model: $f_*(X) = X^\top \beta = \sum_{j=1}^p x_j \beta_j$

$$\Rightarrow Y_i = X_i^\top \beta + \epsilon_i$$

Then we recover the usual linear regression formulation

$$= \begin{bmatrix} x_1 & \cdots & x_p \end{bmatrix} = \begin{bmatrix} X_1^\top \\ X_2^\top \\ \vdots \\ X_n^\top \end{bmatrix}.$$

When referring to j^{th} entry of any X_i, we write X_i^j.

Commonly, a column $x_0^\top = \underbrace{(1, \ldots, 1)}_{n \text{ times}}$ is included. This encodes an intercept term, with intercept parameter β_0 . We could seek to find a β such that $Y \approx \beta$.

[1] Note: sometimes we integrate with respect to D only, Z only, neither (loss), or both.

Polynomial effects

Instead, we may believe

$$f_*(X) = \beta_0 + \sum_{j=1}^{p} X^j \beta_j + \sum_{j=1}^{p} \sum_{j'=1}^{p} X^j X^{j'} \alpha_{j,j'}$$

Then the feature matrix is

$$= \left[\begin{array}{ccccccccc} x_0 & x_1 & \cdots & x_p & x_1^2 & x_1 x_2 & \cdots & x_p^2 \end{array} \right]$$

Here, interpret vector multiplication in the entrywise sense, as in R: `x * y`.

General form

Specify

functions $\phi_k :^p \to, \; k = 1, \ldots, K$

$$= [\phi_k(X_i)] = \left[\begin{array}{c} \Phi(X_1)^\top \\ \Phi(X_2)^\top \\ \vdots \\ \Phi(X_n)^\top \end{array} \right] \in R^{n \times K},$$

where $\Phi(\cdot)^\top = (\phi_1(\cdot), \ldots, \phi_K(\cdot))$.

Example

$$\phi_k(X) = X^j X^{j'}$$

is an interaction for the j^{th} and j'^{th} covariates.

In this case $K = \binom{p}{2} + p = p(p-1)/2 + p = (p^2 + p)/2$

We don't know if f_* can actually be expressed as a linear function. Hence, write

$$\Phi = \{f : \exists (\beta_k)_{k=1}^K \text{ such that } f = \sum_{k=1}^{K} \beta_k \phi_k = \beta^\top \Phi\}$$

and

$$f_{*,\Phi} =_{f \in \Phi} \P \ell_f.$$

The function $f_{*,\Phi}$ is known as the linear oracle. This is the object we are estimating when using a linear model.(Alternatively, we are assuming $f_* \in \Phi$)

Multiple regression redux

Let $K = p$ and define ϕ_k to be the coordinate projection map

That is,

$$\phi_k(X_i) \equiv X_i^k$$

We recover the usual linear regression formulation

$$= [\phi_k(X_i)] = \begin{bmatrix} \Phi(X_1)^\top \\ \Phi(X_2)^\top \\ \vdots \\ \Phi(X_n)^\top \end{bmatrix} = \begin{bmatrix} X_1^1 & X_1^2 & \cdots & X_1^p \\ X_2^1 & X_2^2 & \cdots & X_2^p \\ \vdots & & & \\ X_n^1 & X_n^2 & \cdots & X_n^p \end{bmatrix} = \begin{bmatrix} X_1^\top \\ X_2^\top \\ \vdots \\ X_n^\top \end{bmatrix}.$$

Orthogonal basis expansion

Suppose $f_* \in$, where is a Hilbert space with norm induced by the inner product $\langle \cdot, \cdot \rangle$.

Let $(\phi_k)_{k=1}^\infty$ be an orthonormal basis for

Write

$$f_* = \sum_{k=1}^\infty \langle f_*, \phi_k \rangle \phi_k = \sum_{k=1}^\infty \beta_k \phi_k$$

Then we can estimate $f_{*,\Phi}$ by finding the coefficients of the projection on Φ. By Parseval's theorem for Hilbert spaces, this induces an approximation error of $\sum_{k=K+1}^\infty \beta_k^2$.This is small if f_* is smooth. (for instance, if f_* has m derivatives, then $\beta_k \asymp k^{-m}$)

Neural Nets

Let

$$\phi_k(X) = \sigma(\alpha_k^\top X + b_k),$$

where $\sigma(t) = 1/(1 + e^{-t})$ is the sigmoid activation function. Then we can form the feature matrix

$$= \begin{bmatrix} \phi_1(X_1) & \phi_2(X_1) & \cdots \\ & \vdots & \\ \phi_1(X_n) & \phi_2(X_n) & \cdots \end{bmatrix}$$

For future reference, this is a

"single-layer feed-forward neural network model with linear output"

It is actually a bit more complicated, as the parameters in the σ map are estimated, and hence this is actually nonlinear.

Radial basis functions

Let

$$\phi_k(X) = e^{-||\mu_k - X||_2^2/\lambda_k}.$$

Then $f_{*,\Phi}$ is called an:

"Gaussian radial-basis function estimator'.

This turns out to be a parametric form of a more general technique known as Gaussian process regression.

Detour

WARNING: It is common to conflate:the number of original covariates (p) and the number of created features (K) This means we will always write $\in^{n\times p}$, regardless of the transformation Φ that generates the matrix

The reasons for this are: multiple regression comes from a particular, degenerate choice of Φ; the mapping Φ is often not explicitly created (and $K = \infty$)

Think of X as the vector after transformations and $\in^{n\times p}$ regardless of the choice of Φ.

Turning these ideas into procedures

Each of these methods have parameters to choose:

- p could be very large. Do we include all covariates?
- If we include some polynomial (or other function) terms, should be include all of them?
- For neural nets, we need to choose: the activation function σ, the directions α_k, bias terms b_k, as well as the number of units in the hidden layer

Additionally, we need to estimate the associated coefficient vector β, α, or whatever. We would like the data to inform these parameters.

Training error and risk estimation

The linear oracle is defined to be

$$f_{*,\Phi} =_{f\in\Phi} \P\ell_f.$$

Reminder: for regression, $\ell_f(Z) = (f(X) - Y)^2$

Hence, it is intuitive to use $\hat{\P}$ to form the training error

$$\hat{R}(f) = \hat{\P}\ell_f = \frac{1}{n}\sum_{i=1}^{n} \ell_f(Z_i) = \frac{1}{n}\sum_{i=1}^{n}(f(X_i) - Y_i)^2 = \frac{1}{n}Y - \beta_2^2$$

In many statistical applications, this plug-in estimator is minimized. However, this sometimes has disastrous results.

Example

Let's suppose $\mathcal{D}$is drawn from

```
n = 30
X = (0:n)/n*2*pi
Y = sin(X) + rnorm(n,0,.25)
```

Now, let's fit some polynomials to this data.

We consider the following models:

- Model 1: $f(X_i) = \beta_0 + \beta_1 X_i$
- Model 2: $f(X_i) = \beta_0 + \beta_1 X_i + \beta_2 X_i^2 + \beta_3 X_i^3$
- Model 3: $f(X_i) = \sum_{k=0}^{10} \beta_k X_i^k$
- Model 4: $f(X_i) = \sum_{k=0}^{n-1} \beta_k X_i^k$

The $\hat{R}$'s are:

$\hat{R}(\text{Model 1}) = 10.98$

$\hat{R}(\text{Model 2}) = 2.86$

$\hat{R}(\text{Model 3}) = 2.28$

$\hat{R}(\text{Model 4}) = 0$

Bias and Variance

Prediction risk for regression

Note that $R(\hat{f})$ can be written as

$$R(\hat{f}) = \int \text{bias}^2(x) d¶_X + \int \text{var}(x) d¶_X + \sigma^2$$

where

$$\begin{aligned} \text{bias}(x) &= ¶\hat{f}(x) - f_*(x) \\ \text{var}(x) &= \hat{f}(x) \\ \sigma^2 &= ¶(Y - f_*(X))^2 \end{aligned}$$

As an aside, this decomposition applies to much more general loss functions.

Bias-variance tradeoff

This can be heuristically thought of as

$$\text{Prediction risk} = \text{Bias}^2 + \text{Variance}.$$

There is a natural conservation between these quantities. Low bias → complex model → many parameters → high variance

The opposite also holds. When $\hat{f} \equiv 0$, it has low variance but high bias.

We'd like to 'balance' these quantities to get the best possible predictions