Topic 3: Classification

Eric B. Laber

Department of Statistical Science, Duke University

Statistics 561

On choices

I'd rather be rich than stupid.
—Pierre-Joseph Proudhon

On having to cover classification in two lectures

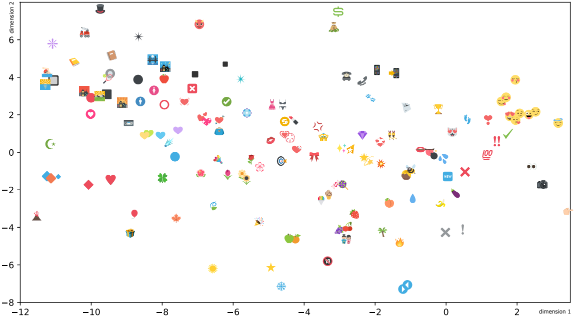
I needed to think last night. So I galloped into a wooded glen, and after punch dancing out my rage and suffering an extremely long and very painful fall, I realized what has to be done.
—Rod

Warm-up (5 minutes)

- Explain to your group
  - What is classification? What are come canonical examples?
  - What is the separability problem?
  - What is zero-one loss?
- True or false
  - Regression is a misnomer in ‘logistic regression’
  - Classification is in the ”inner loop” of many RL algorithms
  - McDonald’s is the most prevalent fast-food chain in NC

Classification: quick overview

- Input-output pairs where output is one of finitely many categories
  - High-risk v low-risk for complications in surgery
  - Spam v not spam
  - Handwritten digit recognition
- Example from sex-trafficking classification problem



Setup: binary classification

- Observe $\{(\boldsymbol{X}_i, Y_i)\}_{i=1}^n$ comprising i.i.d. draws from P
  - Inputs: $\boldsymbol{X} \in \mathbb{R}^p$
  - Outputs: $Y \in \{-1, 1\}$, aka, label
- Classifier $c : \mathbb{R}^p \to \{-1, 1\}$ so that $c(\boldsymbol{x})$ is the predicted label at input $\boldsymbol{X} = \boldsymbol{x}$

0-1 loss

- Natural measure of classification performance is 0-1 loss

$$\ell_0(\boldsymbol{x}, y; c) \triangleq 1_{y \neq c(\boldsymbol{x})}$$

so that the expected loss (risk) is

$$\tau(c) \triangleq P\ell_0(\boldsymbol{X}, Y, c) = P1_{Y \neq c(\boldsymbol{X})} = P\{Y \neq c(\boldsymbol{X})\}$$

- E.g., linear classifier $c(\boldsymbol{x}; \boldsymbol{\beta}) = \mathrm{sign}(\boldsymbol{X}^\intercal \boldsymbol{\beta})$ which has loss

$$\tau(\boldsymbol{\beta}) \triangleq P1_{Y \neq \mathrm{sign}(\boldsymbol{X}^\intercal \boldsymbol{\beta})} = P1_{Y\boldsymbol{X}^\intercal \boldsymbol{\beta} < 0}$$

Bayes classifier

- Let $\mathcal{C}$ be class of all (msbl) maps from $\mathbb{R}^p$ into $\{-1, 1\}$ and

$$c^{\mathrm{opt}} = \arg\min_{c \in \mathcal{C}} \tau(c) = \arg\min_{c \in \mathcal{C}} P\{Y \neq c(\boldsymbol{X})\}$$

then

$$c^{\mathrm{opt}}(\boldsymbol{x}) = \begin{cases} 1 & \text{if } P(Y = 1 | \boldsymbol{X} = \boldsymbol{x}) \geq 1/2 \\ -1 & \text{otherwise} \end{cases}$$

i.e., $c^{\mathrm{opt}}(\boldsymbol{x}) = \mathrm{sign}\{2q(\boldsymbol{x}) - 1\}$ w/ $q(\boldsymbol{x}) \triangleq P(Y = 1 | \boldsymbol{X} = \boldsymbol{x})$

Blank page for notes

Blank page for notes

Probabilistic classifiers

- Natural approach to classification is to estimate construct an estimator $\widehat{q}_n(\boldsymbol{x})$ of $q(\boldsymbol{x}) = P(Y = 1|\boldsymbol{X} = \boldsymbol{x})$

- Logistic regression posits a model of the form

$$q(\boldsymbol{x};\boldsymbol{\beta}^*) = \mathrm{expit}(\boldsymbol{x}^\intercal\boldsymbol{\beta}^*) = \frac{\exp(\boldsymbol{x}^\intercal\boldsymbol{\beta}^*)}{1+\exp(\boldsymbol{x}^\intercal\boldsymbol{\beta}^*)}$$

Derive log-likelihood for logistic regression

Blank page for notes

In-class exercise (5 min)

- In your group derive a stochastic gradient descent algorithm for logistic regression: then on to `logistic_regression.R`

Separability problem

- If classes linearly separable estimates diverge
- Use ridge with logistic regression[1]
- Penalized negative log-likelihood (0-1 coding for simplicity)

$$\ell_n(\boldsymbol{\beta}) = -\mathbb{P}_n \left[\boldsymbol{X}^\intercal \boldsymbol{\beta} Y + \{1 - q(\boldsymbol{X}; \boldsymbol{\beta})\}\right] + \lambda ||\boldsymbol{\beta}||^2$$

differentiate wrt $\boldsymbol{\beta}$ to obtain

$$-\mathbb{P}_n \boldsymbol{X} \{Y - q(\boldsymbol{X}; \boldsymbol{\beta})\} + 2\lambda \boldsymbol{\beta}$$

[1] That's right, I'm stating this without qualification of any kind. What's it to you? That's what I thought.

Warm-up (5 min)

- What's the definition of a convex function?
- Show that if $\phi : \mathbb{R} \to \mathbb{R}$ is convex then $\boldsymbol{\beta} \mapsto P\phi(Y\boldsymbol{X}^{\mathsf{T}}\boldsymbol{\beta})$ is convex (note: we're not assuming ϕ is differentiable)

Large-margin classifiers

- Decision boundary $\{\boldsymbol{x} : q(\boldsymbol{x}) = 1/2\}$
- Don't need $P(Y|\boldsymbol{X} = \boldsymbol{x})$ just $\mathrm{sign}\{2q(\boldsymbol{x}) - 1\}$, i.e., we don't need to know the boundary just which side we're on
- Idea! Construct loss functions that penalize distance from correct side of boundary

Surrogate loss functions: laber draws a picture

Surrogate loss functions

- Common surrogate loss functions include
  - Squared error loss: $\ell(\boldsymbol{x}, y; \boldsymbol{\beta}) = (1 - y\boldsymbol{x}^{\intercal}\boldsymbol{\beta})^2$
  - Exponential loss: $\ell(\boldsymbol{x}, y; \boldsymbol{\beta}) = \exp(-y\boldsymbol{x}^{\intercal}\boldsymbol{\beta})$
  - Hinge loss: $\ell(\boldsymbol{x}, y; \boldsymbol{\beta}) = (1 - y\boldsymbol{x}^{\intercal}\boldsymbol{\beta})_{+}$
  - Logistic loss: $\ell(\boldsymbol{x}, y : \boldsymbol{\beta}) = \log\left\{1 + \exp(-y\boldsymbol{x}^{\intercal}\boldsymbol{\beta})\right\}$
- Can also consider penalized versions of these loss functions

Nice and relaxing derivation

- Show $\widehat{\beta}_n = \arg\min_{\boldsymbol{\beta}} \mathbb{P}_n \log\left\{1 + \exp(-Y\boldsymbol{X}^\intercal\beta)\right\}$ recovers the logistic regression estimator

Blank page for notes

Price of using surrogates

- Pro tip: don't google "price of using surrogates," it will lead you down a fascinating (but time consuming) rabbit hole about surrogate mothers and the ethics of paying them
- We replaced the loss of interest (0-1) with a convex surrogate what are the statistical consequences?
  - Will we still recover Bayes classifier?
  - Can any surrogate be used? Are some better than others?
  - Are there any additional benefits/drawbacks of using a surrogate?

Pointwise consistency aka Fisher Consistency

- Idea: compare population minimizer of surrogate with minimizer of 0-1 loss, if these agree then we say that the surrogate classifier is Fisher consistent

- Note all the surrogates we considered are functions of $yf(\boldsymbol{x})$ for some real-valued function f, e.g., in linear case $y\boldsymbol{x}^{\mathsf{T}}\beta$
  - Let $f : \mathbb{R}^p \rightarrow \mathbb{R}$ be generic fn indexing $\boldsymbol{x} \mapsto \mathrm{sign}\left\{f(\boldsymbol{x})\right\}$
  - Let $\phi : \mathbb{R} \rightarrow \mathbb{R}$ denote a surrogate acting on the margin
    - Squared error loss: $\phi(\alpha) = (1 - \alpha)^2$
    - Exponential loss: $\phi(\alpha) = \exp(-\alpha)$
    - Hinge loss: $\phi(\alpha) = (1 - \alpha)_+$
    - Logistic loss: $\phi(\alpha) = \log\left\{1 + \exp(-\alpha)\right\}$

Pointwise consistency aka Fisher Consistency: defns

- $R(f) \triangleq P1_{Yf(\boldsymbol{X})<0}$ the misclassification error at f
- R^* the Bayes error and define excess risk at f as $R(f) - R^*$
- $R_\phi(f) \triangleq P\phi\{Yf(\boldsymbol{X})\}$ to be the ϕ-risk at f
- $R_\phi^* \triangleq \inf_f R_\phi(f)$ and define the excess ϕ-risk at f as $R_\phi(f) - R_\phi^*$

Relating excess risks

- Goal is to find a function ψ such that $\psi(\tau) \to 0$ iff $\tau \to 0$ and

$$\psi\left\{R(f) - R*\right\} \leq R_\phi(f) - R_\phi^*$$

  What does such a result buy us?

Conditional ϕ-risk

- Conditional ϕ-risk

$$\mathbb{E}\left[\phi\left\{Yf(\boldsymbol{X})\right\}\middle|\boldsymbol{X}=\boldsymbol{x}\right]=q(\boldsymbol{x})\phi\left\{f(\boldsymbol{x})\right\}+\left\{1-q(\boldsymbol{x})\right\}\phi\left\{-f(\boldsymbol{x})\right\}$$

- It will be convenient to define

$$C_q(\alpha)\triangleq q\phi(\alpha)+(1-\alpha)\phi(-\alpha)$$

and

$$H(q)\triangleq\inf_{\alpha}C_q(\alpha)=\inf_{\alpha}\left\{q\phi(\alpha)+(1-q)\phi(-\alpha)\right\}$$

Conditional ϕ-risk cont'd

- Practice with the notation: show that

$$R_\phi^* = \inf_f R_\phi(f)) = PH\left\{q(\boldsymbol{X})\right\}$$

Blank page for notes

Blank page for notes

Classification-calibration

- For $q \in [0, 1]$ define

$$H^{-}(q) \triangleq \inf_{\alpha : \alpha(2q-1) \leq 0} C_q(\alpha)$$

to be optimal conditional ϕ-risk if you disagree with Bayes rule

- We say that a loss function ϕ is classification calibrated if

$$H^{-}(q) > H(q)$$

for all $q \neq 1/2$. Thus, ϕ is CC if disagreeing with Bayes rule increases ϕ-risk.

Ex. classification-calibrated loss

- Show exponential loss is classification-calibrated

Blank page for notes

Blank page for notes

Bounding the excess ϕ-risk

Theorem

Let ϕ be convex. Then ϕ is classification calibrated iff it is differentiable at 0 and $\phi'(0) < 0$. Furthermore, if ϕ is classification calibrated then

$$\psi(\theta) = \phi(0) - H\left(\frac{1+\theta}{2}\right)$$

satisfies

$$\psi\left\{R(f) - R^*\right\} \leq R_\phi(f) - R_\phi^*,$$

where $\psi(\theta) \to 0$ iff $\theta \to 0$.

Statistical inference

- Convexity buys you a lot, many derivations much simpler
- High-level overview of assumptions
  - (A1) $\ell(\boldsymbol{x}, y; \boldsymbol{\beta})$ is convex in $\boldsymbol{\beta}$ for each $(\boldsymbol{x}, y)$
  - (A2) $Q(\boldsymbol{\beta}) = P\ell(\boldsymbol{X}, Y; \boldsymbol{\beta})$ exists and is finite for all $\boldsymbol{\beta}$
  - (A3) $\boldsymbol{\beta}^* = \arg\min Q(\boldsymbol{\beta})$ exists an is unique
  - (A4) $Q(\boldsymbol{\beta})$ is twice continuously differentiable in a nbrhd of $\boldsymbol{\beta}^*$ and $H = \nabla^2 Q(\boldsymbol{\beta}^*)$ is positive definite

Theorem

Assume (A1)-(A4) and let $\Omega = P\nabla\ell(\boldsymbol{X}, Y; \boldsymbol{\beta}^*)\nabla\ell(\boldsymbol{X}, Y; \boldsymbol{\beta}^*)^{\intercal}$ *then*

$$\sqrt{n}(\widehat{\boldsymbol{\beta}}_n - \boldsymbol{\beta}^*) \rightsquigarrow \mathrm{Normal}(0, H^{-1}\Omega H^{-1}).$$

Statistical inference cont'd

- Asymptotic normality $\Rightarrow$ we can use standard methods of inference for $\boldsymbol{\beta}^*$ as in the regression case, e.g., Wald-type intervals etc.
- Unlike regression case, measures of performance are not well-behaved
- Recall our measures of performance
  - Population-level error: $R \triangleq P1_{Y\boldsymbol{X}^\intercal\boldsymbol{\beta}^*<0}$
  - Conditional error: $C(\widehat{\boldsymbol{\beta}}_n) \triangleq P1_{Y\boldsymbol{X}^\intercal\widehat{\boldsymbol{\beta}}_n<0}$
  - Average error: $A_n \triangleq \mathbb{E}C(\widehat{\boldsymbol{\beta}}_n)$

Comparing measures of performance

- Fact: the three measures of performance need not coincide even in infinite samples

Blank page for notes

Blank page for notes

Potpourri: local linear models

- Let $K_\sigma : \mathbb{R}^p \times \mathbb{R}^p \to \mathbb{R}_+$ be a kernel fn, e.g., $K_\sigma(\boldsymbol{x}, \boldsymbol{x}') = \exp\left\{-||\boldsymbol{x} - \boldsymbol{x}'||^2/\sigma^2\right\}$
- Local linear model: estimate coefficient fn $\widehat{\boldsymbol{\beta}}_n(\boldsymbol{x})$ for each $\boldsymbol{x}$ via

$$\widehat{\boldsymbol{\beta}}_n(\boldsymbol{x}) = \arg\min_{\boldsymbol{\beta}} \mathbb{P}_n \left(Y - \boldsymbol{X}^{\intercal}\beta\right)^2 K_\sigma(\boldsymbol{X}, \boldsymbol{x}),$$

  tune σ using CV etc.
  - Natural (and easy!) extension of linear model
  - Studying coeff fn can generate insights about how the mean of Y changes in different regions of the input space (but this is not trivial if p is large)

Potpourri: local large-margin classifiers

- Let K_σ be a kernel and ϕ a surrogate loss function, define

$$\widehat{\beta}_n(\boldsymbol{x}) = \arg\min_{\boldsymbol{\beta}} \mathbb{P}_n \phi(Y\boldsymbol{X}^\intercal \beta) K_\sigma(\boldsymbol{X}, \boldsymbol{x})$$

- E.g., local logistic regression with ridge penalty

$$\widehat{\boldsymbol{\beta}}_n^\lambda(\boldsymbol{x}) = \arg\min_{\boldsymbol{\beta}} \mathbb{P}_n \log\left\{1 + \exp(-YX^\intercal \boldsymbol{\beta})\right\} K_\sigma(\boldsymbol{X}, \boldsymbol{x}) + \lambda ||\boldsymbol{\beta}||^2$$

Potpourri: trees

- Canonical local model: classification and regression trees
- laber draws a tree:
- Tree is an additive model with form $f(\boldsymbol{x}) = \sum_{m=1}^{M} \beta_m 1_{\boldsymbol{x} \in R_m}$ where $R_1, \ldots, R_M$ partition the input space

Potpourri: trees cont'd

- Note that if regions $R_1, \ldots, R_m$ were given and Y were continuous then

$$\widehat{\boldsymbol{\beta}}_n = \arg\min_{\boldsymbol{\beta} \in \mathbb{R}^M} \mathbb{P}_n \left\{ Y - \mathcal{I}(\boldsymbol{X})^{\intercal} \boldsymbol{\beta} \right\}^2 ,$$

where $\mathcal{I}(\boldsymbol{x}) = (1_{\boldsymbol{x} \in R_1}, \ldots, 1_{\boldsymbol{x} \in R_M})^{\intercal}$ and $\widehat{\beta}_j = \mathbb{P}_n Y 1_{\boldsymbol{X} \in R_j} / \mathbb{P}_n 1_{\boldsymbol{X} \in R_j}$

- Note that this is a local constant model with

$$\widehat{\beta}_n(\boldsymbol{x}) = \arg\min_{\beta} \mathbb{P}_n (Y - \beta)^2 K_{\sigma}(\boldsymbol{X}, \boldsymbol{x})$$

and $K_{\sigma}(\boldsymbol{x}, \boldsymbol{x}') = 1_{\boldsymbol{x}, \boldsymbol{x}' \in R_j}$,for some j

Potpourri: trees cont'd

- Estimating optimal partition $R_1, \ldots, R_M$ generally combinatorially hard
- Two approaches:
  - A1: restrict regions to rectangles and optimize greedily
  - A2: randomly generate a bunch of (overlapping) regions aka tile coding

Potpourri notes: greedy optimization

Potpourri notes: tile coding

Warm-up (5 minutes)

- Explain to your stats group
  - What are random forests?
  - What is the Gram matrix?
  - What is a Voronoi partition?
- True or false
  - Voronoi partitions are credited to Georgy Voronoy
  - Voronoi partitions were a key part of a CSI-Cyber episode
  - Voldemort should have done a better job hiding his horcruxes

Kernel Based Random Forests (KeRFs)

- Focus on regression setting
  - Goals for today:
    - Formally define a class of random forests
    - Show random forests representable as kernels

Setup

- Observe $\{(\boldsymbol{X}_i, Y_i)\}_{i=1}^n$ w/ $\boldsymbol{X} \in \mathbb{R}^p$ and $Y \in \mathbb{R}$
- Creating a forest
  - Sequence of trees $\boldsymbol{x} \mapsto \mu_n(\boldsymbol{x}; \boldsymbol{\Theta}_j), j = 1, \ldots, M$, where $\boldsymbol{\Theta}_1, \ldots, \boldsymbol{\Theta}_M$ drawn i.i.d. from $P_{\boldsymbol{\Theta}}$
  - $\boldsymbol{\Theta}$ encodes actions like randomly subsetting/bootstrapping, selecting variables for splits, choosing split points, etc.
  - Prediction at $\boldsymbol{X} = \boldsymbol{x}$

$$\mu_{n,M}(\boldsymbol{x}; \overline{\boldsymbol{\Theta}}_M) = \frac{1}{M} \sum_{j=1}^{M} \mu_n(\boldsymbol{x}; \boldsymbol{\Theta}_j),$$

where $\overline{\boldsymbol{\Theta}}_M = (\boldsymbol{\Theta}_1, \ldots, \boldsymbol{\Theta}_M)$

Some forests

- Breiman: bootstrap data, randomly select set of variables to split on, choose splits to minimize variance
- Extremely randomized trees (ERTs): randomly select set of variables to split on, small set of candidate splits chosen at random, best chosen in terms of minimizing variance
- Gump: simple man with a big heart lives an extraordinary life. I'm not crying, you're crying. FINE, I AM CRYING. SORRY FOR HAVING EMOTIONS YOU CYLON.

Formalizing random forests

- Let $A_n(\boldsymbol{x}, \boldsymbol{\Theta}_j)$ be the node (subset of $\mathbb{R}^p$) in the tree indexed by $\boldsymbol{\Theta}_j$ to which $\boldsymbol{x}$ belongs
- Random forest estimator

$$\mu_{M,n}(\boldsymbol{x}; \overline{\boldsymbol{\Theta}}_M) = \frac{1}{M} \sum_{j=1}^{M} \left\{ \sum_{i=1}^{n} \frac{Y_i 1_{\boldsymbol{X}_i \in A_n(\boldsymbol{x}, \boldsymbol{\Theta}_j}}{\sum_{\ell=1}^{n} 1_{\boldsymbol{X}_\ell \in A_n(\boldsymbol{x}, \boldsymbol{\Theta}_j)}} \right\}$$

$$= \frac{1}{M} \sum_{j=1}^{M} \left\{ \sum_{i=1}^{n} \frac{Y_i 1_{\boldsymbol{X}_i \in A_n(\boldsymbol{x}; \boldsymbol{\Theta}_j)}}{N_n(\boldsymbol{x}; \boldsymbol{\Theta}_j)} \right\},$$

where $N_n(\boldsymbol{x}; \boldsymbol{\Theta}_j) = \sum_{\ell=1}^{n} 1_{\boldsymbol{X}_i \in A_n(\boldsymbol{x}; \boldsymbol{\Theta}_j)}$ is the number of cellmates, i.e., cellies, of $\boldsymbol{x}$ in tree j

Formalizing random forests cont'd

- Weights for Y_i are $1_{\mathbf{X}_i \in A_n(\mathbf{x};\Theta_j)}$ which are large when $N_n(\mathbf{x};\Theta_j)$ is small, i.e., cells with little data influential
- Scornet et al., proposed KeRF (Kernels based on Random Forests)

$$
\begin{aligned}
\widetilde{\mu}_{M,n}(\mathbf{x};\overline{\Theta_j}) &\triangleq \frac{\sum_{j=1}^{M}\sum_{i=1}^{n} Y_i 1_{\mathbf{X}_i \in A_n(\mathbf{x};\Theta_j)}}{\sum_{j=1}^{M} N_n(\mathbf{x};\Theta_j)} \\
&= \frac{\sum_{j=1}^{M}\sum_{i=1}^{n} Y_i 1_{\mathbf{X}_i \in A_n(\mathbf{x};\Theta_j)}}{\sum_{j=1}^{M}\sum_{i=1}^{n} 1_{\mathbf{X}_i \in A_n(\mathbf{x};\Theta_j)}}
\end{aligned}
$$

Random Forests as kernel estimators

- Note that

$$\widetilde{\mu}_{M,n}(\boldsymbol{x}; \overline{\Theta_j}) = \frac{\sum_{i=1}^{n} Y_i K_{M,n}(\boldsymbol{x}, \boldsymbol{X}_i)}{\sum_{i=1}^{n} K_{M,n}(\boldsymbol{x}, \boldsymbol{X}_i)},$$

where $K_{M,n}(\boldsymbol{x}, \boldsymbol{z}) = M^{-1} \sum_{j=1}^{M} 1_{\boldsymbol{z} \in A_n(\boldsymbol{x}; \Theta_j)}$ is the fraction of trees in which $\boldsymbol{x}$, $\boldsymbol{z}$ are in the same cell

Comments on Kernels based on Random Forests

- Kernel selection is a major problem esp. in high-dim settings
- Selection of a high-quality kernel can drastically improve predictive performance
- KeRF is a data-adaptive kernel
  - Variants that use outcome are especially appealing in that they may identify important variables for prediction
  - Interesting benefit: use variables that exist in training data to form kernel (laber discusses search engine example)

Blank page for notes

Blank page for notes

Exercise (2 min)

- How might one use the preceding idea for evil?!

Relating KeRF and RF

(A1) Suppose there exists sequences $\{a_n\}_{n\geq 1}$ and $\{b_n\}_{n\geq 1}$ such that $a_n \leq N_n(\mathbf{x};\Theta) \leq b_n$ almost surely for all n

Theorem

Let $\mu_{M,n}(\mathbf{x};\overline{\Theta}_M)$ be a random forest algorithm that satisfies (A1). Then for each $\mathbf{x}$

$$\left|\frac{\mu_{M,n}(\mathbf{x};\overline{\Theta}_j)}{\widetilde{\mu}_{M,n}(\mathbf{x};\overline{\Theta}_j)} - 1\right| \leq \frac{b_n - a_n}{a_n},$$

almost surely.

Sparsity of the RF Kernel

- RF Kernel often induces a sparse Gram matrix (why?)
- Useful in very large prediction problems esp as an emulator for more complex (and expensive) models at runtime

Thank you.

eric.laber@duke.edu

laber-labs.com