R Programming and Data Analysis

Evaluating Variability and Uncertainty

Introduction

1. Studying properties by simulation
   a. Sampling distribution of MLE
   b. Empirical coverage study for a credible interval
2. Dimension Reduction Application
3. Cross-validation
4. Bootstrapping

Sampling Distribution of an Estimator

Sampling Distribution of an Estimator

- The Beta-Binomial (BB) distribution is an extension of Binomial which allows for extra variation.
- $Z \sim \text{Binomial}(m, p)$ measures the number of successes out of m independent success/failure trials, each having success probability p.
- Suppose

$$
\begin{aligned}
Y &\sim \text{Binomial}(m, p) \\
p &\sim \text{Beta}(\pi\rho^2(1-\rho^2), (1-\pi)\rho^2(1-\rho^2)),
\end{aligned}
$$

where $a = \pi(1-\rho^2)/\rho^2$ and $b = (1-\pi)(1-\rho^2)/\rho^2$.

- Marginally, $Y \overset{\text{iid}}{\sim} \text{BB}(\pi, \rho)$, with density

$$
\begin{aligned}
f(y \mid m, \pi, \rho) &= \frac{\Gamma(m+1)}{\Gamma(y+1)\Gamma(m-y+1)} \frac{\Gamma(a+y)\Gamma(b+m-y)\Gamma(a+b)}{\Gamma(a+b+m)\Gamma(a)\Gamma(b)}, \\
\text{E}(Y) &= m\pi \\
\text{Var}(Y) &= m\pi(1-\pi)\{1+\rho(m-1)\}
\end{aligned}
$$

Sampling Distribution of an Estimator

- Given fixed $m_1, \ldots, m_n$ and a random sample $y_1, \ldots, y_n$ from $\mathrm{BB}(\pi, \rho)$, the maximum likelihood estimator (MLE) is obtained by

$$\hat{\boldsymbol{\theta}} = \underset{(\pi,\rho)}{\operatorname{argmax}} \left\{ \log \left[\prod_{i=1}^{n} f(y_i \mid m_i, \pi, \rho) \right] \right\}.$$

- Large sample theory says that $\hat{\boldsymbol{\theta}} \sim \mathrm{N}(\boldsymbol{\theta}, \mathcal{I}^{-1}(\boldsymbol{\theta}))$, approximately, for large n.

- **Problem**: Use simulation study to determine the distribution of $\hat{\boldsymbol{\theta}}$ for a fixed n.

Sampling Distribution of an Estimator

- Use `optim` to maximize the likelihood numerically.
- π and ρ are probabilities, therefore must be constrained to $(0, 1)$.
  a. `optim` supports simple bound constraints
  b. `constrOptim` supports linear inequality constraints
  c. We will transform the problem to an unconstrained optimization.
- Let $G(x) = 1/\{1 + e^{-x}\}$ denote the cdf of the standard logistic distribution. G is a (one-to-one and onto) mapping from $\mathbb{R}$ to $(0, 1)$.
- Transformed problem is

$$\hat{\phi} = \underset{(\phi_1, \phi_2)}{\operatorname{argmax}} \left\{ \log \left[\prod_{i=1}^{n} f(y_i \mid m_i, \pi = G(\phi_1), \rho = G(\phi_2)) \right] \right\}.$$

- MLE for the original space is $\hat{\boldsymbol{\theta}} = (G(\hat{\phi}_1), G(\hat{\phi}_2))$.

Sampling Distribution of an Estimator

. . . Demonstration . . .

(See `sampling.Rmd`)

Empirical Coverage Study for a Credible Interval

Empirical Coverage Study for a Credible Interval

- In a Bayesian analysis, we assume a model for the observed data $\boldsymbol{y}$ and a distribution for $\boldsymbol{\theta} \in \Theta$ which describes our prior belief,

$$\boldsymbol{y} \sim f(\boldsymbol{y} \mid \boldsymbol{\theta}), \quad \boldsymbol{\theta} \sim f(\boldsymbol{\theta}).$$

- All inference on $\boldsymbol{\theta}$ is then based on the posterior distribution

$$f(\boldsymbol{\theta} \mid \boldsymbol{y}) = \frac{f(\boldsymbol{y} \mid \boldsymbol{\theta}) f(\boldsymbol{\theta})}{\int f(\boldsymbol{y} \mid \boldsymbol{\vartheta}) f(\boldsymbol{\vartheta}) d\boldsymbol{\vartheta}}.$$

Empirical Coverage Study for a Credible Interval

- If $Y_i \overset{\text{ind}}{\sim} \text{Binomial}(m_i, p)$ and $p \sim \text{Beta}(a, b)$, for given a and b, then

$$f(p \mid \boldsymbol{y}) \propto \left\{ \prod_{i=1}^{n} \binom{m_i}{y_i} p^{y_i} (1-p)^{m_i - y_i} \right\} \frac{p^{a-1}(1-p)^{b-1}}{B(a,b)}$$
$$\propto p^{\sum_{i=1}^{n} y_i + a - 1} (1-p)^{\sum_{i=1}^{n} m_i - \sum_{i=1}^{n} y_i + b - 1},$$

  so that $p \mid \boldsymbol{y} \sim \text{Beta}(\sum_{i=1}^{n} y_i + a, \sum_{i=1}^{n} m_i - \sum_{i=1}^{n} y_i + b)$.

- A level $1 - \alpha$ credible interval for p uses the $\alpha/2$ and $1 - \alpha/2$ quantiles.

- **Problem**: What is the expected width and coverage probability?

Sampling Distribution of an Estimator

. . . Demonstration . . .

(See `credible.Rmd`)

Dimension Reduction Application

Dimension Reduction Application

- We will now consider some dimension reduction methods to use with the `diamonds` dataset (in the `ggplot2` package).
- **?** propose several methods for dimension reduction in regression models when the form of the regression function is not known.
- Suppose $\{(\boldsymbol{x}_i, y_i) : i = 1, \dots, n\}$ is an iid sample from

$$Y = \tilde{g}(\boldsymbol{X}) + \epsilon, \quad \mathrm{E}(\epsilon \mid \boldsymbol{X}) = 0, \quad \boldsymbol{X} \in \mathbb{R}^p$$

  where $\tilde{g}$ is unknown regression function.

- It is often conceivable that, for some $d < p$,

$$\begin{aligned}\tilde{g}(\boldsymbol{x}) &= g(\boldsymbol{B}^\top \boldsymbol{x}) \\ &= g(\boldsymbol{b}_1^\top \boldsymbol{x}, \dots, \boldsymbol{b}_d^\top \boldsymbol{x}), \quad \boldsymbol{B} = (\boldsymbol{b}_1 \cdots \boldsymbol{b}_d) \in \mathbb{R}^{p \times d}.\end{aligned}$$

  Here, $\boldsymbol{B}$ is the basis of a d-dimensional subspace of $\mathbb{R}^p$ which preserves the regression relationship between y and $\boldsymbol{x}$.

Dimension Reduction Application

- The Minimum Average Variance Estimation (MAVE) method (**?**) estimates $\boldsymbol{B}$ by minimizing the objective function

$$Q(\boldsymbol{B}) = \sum_{j=1}^{n} \min_{\alpha_j, \boldsymbol{\gamma}_j} \left\{ \sum_{i=1}^{n} \left(y_i - \alpha_j - \boldsymbol{\gamma}_j^\top \boldsymbol{B}^\top (\boldsymbol{x}_i - \boldsymbol{x}_j) \right)^2 w_{ij} \right\}$$

- The model is based on Taylor series expansions

$$\begin{aligned} g(\boldsymbol{B}^\top \boldsymbol{x}_i) &\approx \underbrace{g(\boldsymbol{B}^\top \boldsymbol{x}_0)}_{\alpha_0} + \underbrace{[\nabla g(\boldsymbol{B}^\top \boldsymbol{x}_0)]^\top}_{\boldsymbol{\gamma}_0^\top} (\boldsymbol{x}_i - \boldsymbol{x}_0), \quad i = 1, \ldots, n, \\ &= \alpha_0 + \boldsymbol{\gamma}_0^\top (\boldsymbol{x}_i - \boldsymbol{x}_0), \end{aligned}$$

  around a given point $\boldsymbol{x}_0$.

- Given an estimate $\hat{\boldsymbol{B}}$, a prediction of y_0 for a given $\boldsymbol{x}_0$ is

$$\hat{\alpha}_0 = g(\hat{\boldsymbol{B}}^\top \boldsymbol{x}_0).$$

Dimension Reduction Application

- MAVE objective function

$$Q(\boldsymbol{B}) = \sum_{j=1}^{n} \min_{\alpha_j, \gamma_j} \left\{ \sum_{i=1}^{n} \left(y_i - \alpha_j - \gamma_j^\top \boldsymbol{B}^\top (\boldsymbol{x}_i - \boldsymbol{x}_j) \right)^2 w_{ij} \right\}$$

- MAVE uses kernel weights to avoid assumptions on the distribution of $\boldsymbol{X}$. Relative to a given $\boldsymbol{x}_0$, and given a bandwidth h, the weights

$$w_{i0} = \frac{K_h(\boldsymbol{x}_i - \boldsymbol{x}_0)}{\sum_{\ell=1}^{n} K_h(\boldsymbol{x}_\ell - \boldsymbol{x}_0)}, \quad i = 1, \ldots, n,$$

are based on a kernel function $K_h(\boldsymbol{u}) = p^{-1} K(\boldsymbol{u}/\sqrt{h})$.

- An example is the Gaussian kernel $K_h(\boldsymbol{u}) = (2\pi)^{p/2} \exp(-u^\top u/2)$.

- Choice of bandwidth h has a major influence on the model.

Dimension Reduction Application

- MAVE objective function

$$Q(\boldsymbol{B}) = \sum_{j=1}^{n} \min_{\alpha_j, \boldsymbol{\gamma}_j} \left\{ \sum_{i=1}^{n} \left(y_i - \alpha_j - \boldsymbol{\gamma}_j^\top \boldsymbol{B}^\top (\boldsymbol{x}_i - \boldsymbol{x}_j) \right)^2 w_{ij} \right\}$$

- Since $\boldsymbol{B}$ is the basis of a subspace, $\boldsymbol{B}^\top \boldsymbol{B} = \boldsymbol{I}_{d \times d}$. Furthermore, for any orthogonal matrix $\boldsymbol{A}$,

$$\boldsymbol{\gamma}_j^\top \boldsymbol{A}\boldsymbol{A}^\top \boldsymbol{B}^\top = \boldsymbol{\gamma}_j^\top \boldsymbol{B}^\top \quad \Longrightarrow \quad Q(\boldsymbol{BA}) = Q(\boldsymbol{B}).$$

  Therefore, $Q(\boldsymbol{B})$ is an optimization problem over the d-dimensional subspaces in $\mathbb{R}^p$ (the “Grassmann manifold”).

Dimension Reduction Application

- MAVE objective function

$$Q(\boldsymbol{B}) = \sum_{j=1}^{n} \min_{\alpha_j,\gamma_j} \left\{ \sum_{i=1}^{n} \left(y_i - \alpha_j - \boldsymbol{\gamma}_j^\top \boldsymbol{B}^\top (\boldsymbol{x}_i - \boldsymbol{x}_j) \right)^2 w_{ij} \right\}$$

- Given a particular $\boldsymbol{B}$ and $\boldsymbol{x}_0$, the inner minimization over

$$Q_{\boldsymbol{B}}(\alpha_0, \boldsymbol{\gamma}_0) = \sum_{i=1}^{n} \left(y_i - \alpha_0 - \boldsymbol{\gamma}_0^\top \boldsymbol{B}^\top (\boldsymbol{x}_i - \boldsymbol{x}_0) \right)^2 w_{i0}$$

corresponds to a local regression estimate at $\boldsymbol{x}_j$ (**?**).

- Minimize $Q_{\boldsymbol{B}}(\alpha_0, \boldsymbol{\gamma}_0)$ in one step via weighted least squares (WLS),

$$\begin{pmatrix} \hat{\alpha}_0 \\ \hat{\boldsymbol{\gamma}}_0 \end{pmatrix} = [\boldsymbol{Z}^\top \boldsymbol{W}_0 \boldsymbol{Z}]^{-1} \boldsymbol{Z}^\top \boldsymbol{W}_0 \boldsymbol{y}, \quad \text{where} \quad \boldsymbol{Z} = \begin{pmatrix} 1 & (\boldsymbol{x}_1 - \boldsymbol{x}_0)^\top \boldsymbol{B} \\ \vdots & \vdots \\ 1 & (\boldsymbol{x}_n - \boldsymbol{x}_0)^\top \boldsymbol{B} \end{pmatrix},$$

$$\boldsymbol{W}_0 = \mathrm{Diag}(w_{10}, \ldots, w_{n0}), \quad \boldsymbol{y} = (y_1, \ldots, y_n)^\top.$$

Dimension Reduction Application

- **?** propose an iterative algorithm based on quadratic programming to minimize the objective function

$$Q(\boldsymbol{B}) = \sum_{j=1}^{n} \min_{\alpha_j, \boldsymbol{\gamma}_j} \left\{ \sum_{i=1}^{n} \left(y_i - \alpha_j - \boldsymbol{\gamma}_j^\top \boldsymbol{B}^\top (\boldsymbol{x}_i - \boldsymbol{x}_j) \right)^2 w_{ij} \right\}$$

- **?** also propose a simpler method called Outer Product of Gradients (OPG). The rate of consistency of OPG is slower than MAVE, but it works for our purposes and it does not require multiple iterations.

- OPG can provide a good starting value for MAVE.

Outer Product of Gradients (OPG) Algorithm

Inputs: dimension d and bandwidth h.

```
for $j = 1$ to $n$ do
    Compute weights $w_{ij} = \frac{K_h(\boldsymbol{x}_i - \boldsymbol{x}_0)}{\sum_{\ell=1}^{n} K_h(\boldsymbol{x}_\ell - \boldsymbol{x}_j)}$ for $i = 1, \ldots, n$.
    Compute $(\hat{\alpha}_j, \hat{\gamma}_j)$ by minimizing $Q_{\boldsymbol{B}}(\alpha_j, \gamma_j)$ via WLS.
end for
return $\hat{\boldsymbol{B}}$ as matrix of the first $d$ eigenvectors of $\hat{\boldsymbol{\Sigma}} = \frac{1}{n} \sum_{j=1}^{n} \hat{\gamma}_j \hat{\gamma}_j^\top$.
```

Dimension Reduction Application

- There are two factors that must be selected by the user: the dimension $d \in \{0, 1, \ldots, p\}$ of the reduction, and the bandwidth $h > 0$. We will assume $d = 1$, but select h via **cross-validation**.
- The estimated regression function is based on a random sample, and should be expressed with uncertainty. We will use the **bootstrap** to provide upper and lower bounds to go with the estimated function.

Cross-Validation

Cross-Validation

- In statistics and machine learning, we assume that observed data is generated from an underlying process. The process, not necessarily the data, is our primary interest. But we learn about the process through the data.
- Overfitting occurs when a model is trained to capture all features of the observed data, even spurious ones, but fails to capture important features of the process.
- Using the data at hand, we would like to select a good fitting model which does not overfit.
- Cross-validation (**?**, §7.10) is a method to assess model fit and ensure that we are not overfitting or oversmoothing the process of interest.

Cross-Validation

- To illustrate cross-validation concretely, consider the following classification problem.
- Suppose $\boldsymbol{x}_i \in \mathbb{R}^d$ and $\mathcal{R}$ is a subregion of $\mathbb{R}^d$ for $i = 1, \ldots, n$. We observe the class assignment

$$y_i = \begin{cases} 1 & \text{if } \boldsymbol{x}_i \in \mathcal{R}, \\ 0 & \text{otherwise.} \end{cases}$$

  Having observed $\mathcal{D} = \{(\boldsymbol{x}_i, y_i) : i = 1, \ldots, n\}$, but without explicitly knowing $\mathcal{R}$, how to assign the class y^* for a new $\boldsymbol{x}^* \in \mathbb{R}$?
- We could allow some randomness in the observed y_i, but this is not necessary for our illustration.
- The N-Nearest Neighbors algorithm finds the closest N points in $\mathcal{D}$ to $\boldsymbol{x}^*$ and takes y^* to be the one most frequently occurring in this set (**?**, Chapter 2).
- Taking N too small will overfit, but N too large will oversmooth.

Cross-Validation

- The most basic way to assess overfitting is to partition the sample into three parts: a **training set**, a **validation set**, and a **test set**.
- For each N, fit the model using observations in the training set, and evaluate the model using observations in the validation set.
- Select the N that gave the best fit on the validation set, and let this be the final model.
- The fit of the final model may be evaluated using the test set.
- This method assumes that there are no systematic differences between the three datasets. This can be avoided by partitioning at random.

Cross-Validation

- K-fold cross-validation makes more efficient use of the data at the expense of more computing.
- Partition the observations indices $\mathcal{S} = \{1, \ldots, n\}$ into K subsets $\mathcal{S}_1, \ldots, \mathcal{S}_K$.

Algorithm 1 Cross-validation for N-Nearest Neighbors

```
for $N = 1, \ldots, n$ do
    for $k = 1, \ldots, K$ do
        Predict classes $\hat{y}_{\mathrm{CV},i}$ for $i \in \mathcal{S}_k$ using $\mathcal{S} \setminus \mathcal{S}_k$ as training set.
    end for
    Let $\mathrm{ERROR}_{\mathrm{CV}(N)} = \sum_{i=1}^{n} I(y_i \neq \hat{y}_{\mathrm{CV},i})$
end for
```

- We could also set aside a test set, which would be left out of cross-validation and reserved for evaluating the final model.

Simple Cross-Validation Example

. . . Demonstration . . .

(See `cv.Rmd`)

Cross-validation

- Now consider bandwidth selection for the OPG method, and suppose we have a set of candidate bandwidths $\mathcal{H}$.
- Selecting h too small will use too much information from immediate neighbors and overfit.
- Selecting h too large will fail to capture important local features and oversmooth.
- It is reasonable to select h by minimizing a prediction error such as

$$\text{SAPE} = \sum_{i=1}^{n} |y_i - \hat{y}_i|.$$

- We could compute SAPE for each $h \in \mathcal{H}$ using all the observations, and pick an h that gives the smallest value. But this does not reflect if our selected model has overfit.

Cross-validation

Algorithm 2 Cross-validation for OPG

```
for $h \in \mathcal{H}$ do
    for $k = 1, \ldots, K$ do
        Fit model to obs in $\mathcal{S} \setminus \mathcal{S}_k$ to estimate $\hat{B}_{\mathrm{CV}}$.
        Use $\hat{B}_{\mathrm{CV}}$ to get predictions for the obs in $\mathcal{S}_k$.
    end for
    Compute $\mathrm{SAPE}_{\mathrm{CV}(h)}$
end for
```

Cross-Validation with OPG

. . . Demonstration . . .

(See `sim.Rmd` and `diamonds.Rmd`)

Bootstrap

Bootstrap

- Bootstrap is a technique to estimate the distribution of a statistic, and thereby obtain estimates for properties such as variance and bias (**?**).
- Generally, let $\boldsymbol{X} = (X_1, \ldots, X_n)$ be a random sample and $\hat{\theta}(\boldsymbol{X})$ be an estimator for a quantity $\boldsymbol{\theta}$ of interest.

Algorithm 3 Nonparametric Bootstrap

```
for $b = 1, \ldots, R$ do
    $s \leftarrow$ simple random sample with replacement of size $n$ from $\{1, \ldots, n\}$
    $\boldsymbol{X}_{\text{boot}} \leftarrow (X_{s_1}, \ldots, X_{s_n})$
    $\hat{\boldsymbol{\theta}}_r \leftarrow \hat{\boldsymbol{\theta}}(\boldsymbol{X}_{\text{boot}})$
end for
return $\hat{\boldsymbol{\theta}}_1, \ldots, \hat{\boldsymbol{\theta}}_R$
```

- Bootstrap is very general and can be used in complicated situations where other methods are not available or are too difficult to apply.
- This flexibility comes at a price; bootstrap becomes heavily computational when $\hat{\theta}(\boldsymbol{X})$ takes effort to compute.

Explanation of Bootstrap

- From lecture Larry Wasserman's lecture notes (`http://www.stat.cmu.edu/~larry/=stat705/Lecture13.pdf`).
- Suppose $\boldsymbol{X} = (X_1, \ldots, X_n)$ is an iid sample from distribution P, and we want to estimate a functional $T_n(P)$.
- Concretely, suppose $T_n(P) = \text{Var}_P(\hat{\boldsymbol{\theta}}(\boldsymbol{X}))$ for illustration.
- If we knew P, we could approximate $T_n(P)$ by simulation:
  1. Draw $\boldsymbol{X}^{(1)}$ from P and compute $\hat{\boldsymbol{\theta}}^{(1)} = \hat{\boldsymbol{\theta}}(\boldsymbol{X}^{(1)})$.

     $\vdots$

  R. Draw $\boldsymbol{X}^{(R)}$ from P and compute $\hat{\boldsymbol{\theta}}^{(R)} = \hat{\boldsymbol{\theta}}(\boldsymbol{X}^{(R)})$.

  Take s^2 to be the sample variance of $\hat{\boldsymbol{\theta}}^{(1)}, \ldots, \hat{\boldsymbol{\theta}}^{(R)}$.
- By the strong law of large numbers $s^2 \overset{\text{a.s.}}{\to} T_n(P)$.
- In a data analysis situation with an observed $x_1, \ldots, x_n$, P is unknown.

Explanation of Bootstrap

- Bootstrap uses the empirical distribution P_n of $x_1, \ldots, x_n$ to estimate P;

$$P_n(x) = \begin{cases} \frac{1}{n} & \text{if } x \in \{x_1, \ldots, x_n\}, \\ 0 & \text{o.w.} \end{cases}$$

  This is the same as drawing a simple random sample with replacement of size n from $\{x_1, \ldots, x_n\}$.

- In Bootstrap,
  1. Draw $\tilde{\boldsymbol{X}}^{(1)}$ from P_n and compute $\hat{\boldsymbol{\theta}}^{(1)} = \hat{\boldsymbol{\theta}}(\tilde{\boldsymbol{X}}^{(1)})$.

     $\vdots$

  R. Draw $\tilde{\boldsymbol{X}}^{(R)}$ from P_n and compute $\hat{\boldsymbol{\theta}}^{(R)} = \hat{\boldsymbol{\theta}}(\tilde{\boldsymbol{X}}^{(R)})$.

  Take s^2_{boot} to be the sample variance of $\hat{\boldsymbol{\theta}}^{(1)}, \ldots, \hat{\boldsymbol{\theta}}^{(R)}$.

- By the strong law of large numbers $s^2_{\text{boot}} \overset{\text{a.s.}}{\to} T_n(P_n)$ as $R \to \infty$.

- $T_n(P_n)$ estimates $T_n(P)$ as $n \to \infty$, under some regularity conditions.

Simple Bootstrap Example

. . . Demonstration . . .

(See `boot.Rmd`)

Bootstrap

- The basic MAVE and OPG algorithms do not produce estimates of variability, but these can be obtained via the bootstrap.
- Recall that $\alpha_0 = g(\boldsymbol{B}^\top \boldsymbol{x}_0)$ is the quantity we wished to estimate. Let us compute $1 - \delta$ level confidence intervals for each $\boldsymbol{x}_0$ in a given set $\mathcal{X}$.

Algorithm 4 Nonparametric Bootstrap with OPG Algorithm

```
for $r = 1, \ldots, R$ do
    $s \leftarrow$ simple random sample with replacement of size $n$ from $\{1, \ldots, n\}$
    $\boldsymbol{y}_{\text{boot}} \leftarrow (y_{s_1}, \ldots, y_{s_n})$
    $\boldsymbol{X}_{\text{boot}} \leftarrow (\boldsymbol{x}_{s_1} \cdots \boldsymbol{x}_{s_n})^\top$
    $\hat{B}_r \leftarrow \text{OPG}(\boldsymbol{y}_{\text{boot}}, \boldsymbol{X}_{\text{boot}}, h)$
    $\hat{\boldsymbol{\alpha}}_r \leftarrow$ estimates $\hat{\alpha}_0$ for each $\boldsymbol{x}_0 \in \mathcal{X}$ based on $\hat{B}_r$
end for
return $\hat{\boldsymbol{\alpha}}_1, \ldots, \hat{\boldsymbol{\alpha}}_R$
```

- A bootstrap CI for $\boldsymbol{x}_0 \in \mathcal{X}$ is obtained using the $\delta/2$ and $1 - \delta/2$ quantiles of $\hat{\boldsymbol{\alpha}}_{10}, \ldots, \hat{\boldsymbol{\alpha}}_{R0}$.
- Other bootstrap CIs have been proposed (**?**).

Bootstrap with OPG

. . . Demonstration . . .

(See `sim.Rmd` and `diamonds.Rmd`)

References I