Data Mining
W4240 Section 001

Yixin Wang

Columbia University, Department of Statistics

November 4, 2015

Outline

Outline

Generalization

Modeling for prediction:

1. get data
2. choose a model
3. fit the model
4. make predictions for new data

Generalization: making high quality predictions for new data

Expected Predictive Error

Tunable parameters α

Model with parameters α, $\hat{f}_\alpha(x)$

Goals for expected predictive error:

- **Model selection:** estimating the performance of different models in order to choose the best one (best α).
- **Model assessment:** having chosen a final model, estimating its prediction error (generalization error) on new data.

Types of Error Estimators

Training data: $\mathcal{T} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$

New data: X^0, Y^0

Types of Error Estimators

Training data: $\mathcal{T} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$

New data: X^0, Y^0

Training error (error, given a training set):

$$\mathrm{Err}_{\mathrm{train}} = \frac{1}{n} \sum_{i=1}^{n} L(y_i, \hat{f}(x_i))$$

Types of Error Estimators

Training data: $\mathcal{T} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$

New data: X^0, Y^0

Training error (error, given a training set):

$$\mathrm{Err}_{\mathrm{train}} = \frac{1}{n}\sum_{i=1}^{n} L(y_i, \hat{f}(x_i))$$

Generalization error (expected testing error, given a training set):

$$\mathrm{Err}_{\mathcal{T}} = \mathbb{E}_{X^0,Y^0}[L(Y^0, \hat{f}(X^0)) \mid \mathcal{T}]$$

Types of Error Estimators

Training data: $\mathcal{T} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$

New data: X^0, Y^0

Training error (error, given a training set):

$$\mathrm{Err}_{\mathrm{train}} = \frac{1}{n} \sum_{i=1}^{n} L(y_i, \hat{f}(x_i))$$

Generalization error (expected testing error, given a training set):

$$\mathrm{Err}_{\mathcal{T}} = \mathbb{E}_{X^0, Y^0}[L(Y^0, \hat{f}(X^0)) \,|\, \mathcal{T}]$$

Expected error (expected generalization error):

$$\mathrm{Err} = \mathbb{E}_{\mathcal{T}} \mathbb{E}_{X^0, Y^0}[L(Y^0, \hat{f}(X^0)) \,|\, \mathcal{T}]$$

Cross-Validation

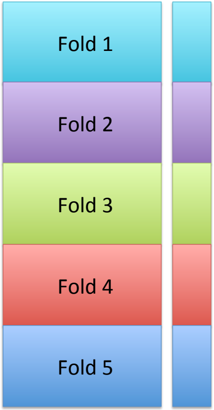
Let's revisit cross-validation: estimating expected testing error, for use in model selection and assessment

K-fold cross-validation

- separate training set into K different, equally sized sets (folds)
- for each tunable parameter value $\alpha = \alpha_1, \ldots, \alpha_M$:
  - for $k = 1, \ldots, K$:
    - use all of the data except fold k as a training set to fit the function with parameter α
    - use fold k as a testing set
    - estimate squared error on fold k
  - average errors to approximate expected predictive error
- compare error values; pick parameter with lowest error

Cross-Validation

Training Set



$$MSE_k = \frac{1}{n}\sum_j SE_k(\text{fold } j)$$

Test or validate fitted function on fold j

$$SE_k(\text{fold } j) = \sum_{i=1}^{n_j}\left(y_i - \hat{f}_{k,j}(x_i)\right)^2$$

$$\hat{f}_{k,j}(x) = \frac{1}{k}\sum_{i \text{ is in } \mathrm{N}_{k,j}(x)} y_i$$

Fit function on all folds except j

j=1,…,5

Cross-Validation

Cross-validation is a good estimator for generalization error,

$$\mathrm{Err}_{\mathcal{T}} = \mathbb{E}[L(Y), \hat{f}(X) \,|\, \mathcal{T}]$$

How can we get a good estimate of extra-sample prediction error,

$$\mathrm{Err} = \mathbb{E}_{\mathcal{T}}\mathbb{E}[L(Y), \hat{f}(X) \,|\, \mathcal{T}]$$

Cross-Validation

Cross-validation is a good estimator for generalization error,

$$\mathrm{Err}_{\mathcal{T}} = \mathbb{E}[L(Y), \hat{f}(X) \,|\, \mathcal{T}]$$

How can we get a good estimate of extra-sample prediction error,

$$\mathrm{Err} = \mathbb{E}_{\mathcal{T}}\mathbb{E}[L(Y), \hat{f}(X) \,|\, \mathcal{T}]$$

- What if we want the distribution of the estimator?

Outline

"Pull yourself up by your bootstraps!"

Bootstrap Methods

Problem: Estimate

$$\mathrm{Err} = \mathbb{E}_{\mathcal{T}}\mathbb{E}[L(Y), \hat{f}(X) \,|\, \mathcal{T}]$$

need to sample from the distribution of $\mathcal{T}$

What has the same distribution as $\mathcal{T}$?

Bootstrap Methods

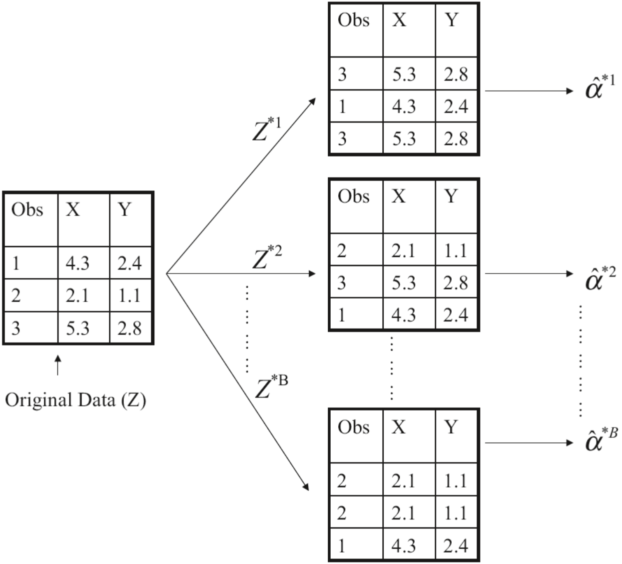
To get a bootstrap estimate,

1. resample from the original data n times *with replacement*
2. use new dataset to compute bootstrap estimate
3. create B new datasets
4. (draw a picture for why this is the right thing to do)

(Bootstrap dataset contains $\sim 63.2\%$ of the original data)

$$\begin{aligned}\mathbb{P}\{\text{observation } i \in \text{ bootstrap sample } b\} &= 1 - \left(1 - \frac{1}{n}\right)^n \\ &\approx 1 - e^{-1}\end{aligned}$$

Bootstrap Methods



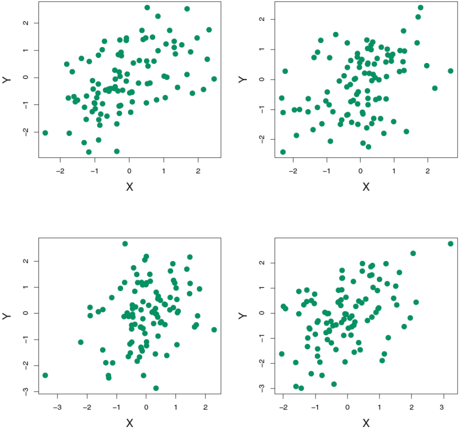
Example: two financial returns

- We invest a fixed sum of money in two financial assets that yield returns of X and Y (X and Y are random).
- Invest a fraction α of our money in X, and the remaining $(1-\alpha)$ in Y
- We choose α to minimize the total risk, or variance, of our investment: $\mathrm{Var}[\alpha X + (1-\alpha)Y]$.
- The value that minimizes the risk is given by

$$\alpha = \frac{\sigma_Y^2 \sigma_{XY}}{\sigma_X^2 + \sigma_Y^2 - 2\sigma_{XY}}$$

Example: two financial returns, $M = 4$

Simulated data ($n = 100$, $\sigma_X^2 = 1$, $\sigma_Y^2 = 1.25$, $\sigma_{XY} = 0.5$).



$$\hat{\boldsymbol{\alpha}}_M = \begin{bmatrix} \hat{\alpha}_1 & \hat{\alpha}_2 \\ \hat{\alpha}_3 & \hat{\alpha}_4 \end{bmatrix} = \begin{bmatrix} 0.576 & 0.532 \\ 0.657 & 0.651 \end{bmatrix}$$

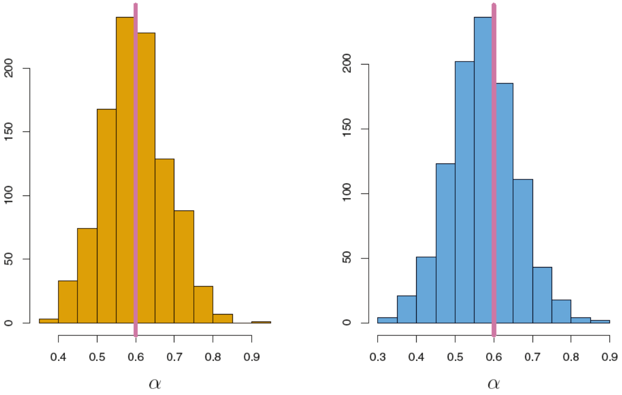
Example: two financial returns, $M = B = 1,000$

$$\begin{aligned}
\bar{\alpha}_M &= \frac{1}{M}\sum_{m=1}^{M}\hat{\alpha}_m = 0.59996 \\
\mathrm{SE}_M(\hat{\alpha}) &= \sqrt{\frac{1}{M-1}\sum_{m=1}^{M}(\hat{\alpha}_m - \bar{\alpha}_M)^2} = 0.083
\end{aligned}$$

$$\begin{aligned}
\bar{\alpha}^*_B &= \frac{1}{B}\sum_{b=1}^{B}\hat{\alpha}^*_b \\
\mathrm{SE}_B(\hat{\alpha}^*) &= \sqrt{\frac{1}{B-1}\sum_{b=1}^{B}(\hat{\alpha}^*_b - \bar{\alpha}^*_B)^2} = 0.087
\end{aligned}$$

Bootstrap Methods

Left: simulated data ($n = 100$, $M = 1,000$,).
Right: bootstrap ($n = 100$, $B = 1,000$).



Bootstrap Methods

Bootstrap method:

1. for $b = 1, \ldots, B$
   - create new dataset $(x_i^{(b)}, y_i^{(b)})_{i=1}^n$ by sampling from original dataset *with replacement*
   - estimate error (or other values like variance) with new dataset
2. average estimated errors

Expected predictive error:

$$\begin{aligned} \mathrm{Err} &= \mathbb{E}_{\mathcal{T}} \mathbb{E}[L(Y, \hat{f}(X)) \,|\, \mathcal{T}] \\ &\approx \frac{1}{B} \frac{1}{n} \sum_{b=1}^{B} \sum_{i=1}^{n} L(y_i, \hat{f}^b(x_i)) \end{aligned}$$

Outline

Bootstrap MSE: Example

100 data points, $X_1, \ldots, X_{100} \sim N(\mu, 1)$

Bootstrap MSE: Example

100 data points, $X_1, \ldots, X_{100} \sim N(\mu, 1)$

- What is a good estimator for μ?

Bootstrap MSE: Example

100 data points, $X_1, \ldots, X_{100} \sim N(\mu, 1)$

- What is a good estimator for μ?

- How can we estimate $\mathrm{Var}(\hat{\mu})$
  1. for $b = 1 : 1000$:
     - resample $x_1, \ldots, x_{100}$ *with replacement* to get $x_1^{(b)}, \ldots, x_{100}^{(b)}$
     - compute $\hat{\mu}(x^{(b)}) = \frac{1}{100} \sum_{i=1}^{100} x_i^{(b)}$
  2. set $\hat{\mu} = \frac{1}{1000} \sum_{b=1}^{1000} \hat{\mu}(x^{(b)})$
  3. set $\hat{\mathrm{Var}} = \frac{1}{1000} \sum_{b=1}^{1000} \left(\hat{\mu}(x^{(b)}) - \hat{\mu}\right)^2 - \hat{\mu}^2$

Bootstrap MSE: Example

100 data points, $X_1, \ldots, X_{100} \sim N(\mu, 1)$

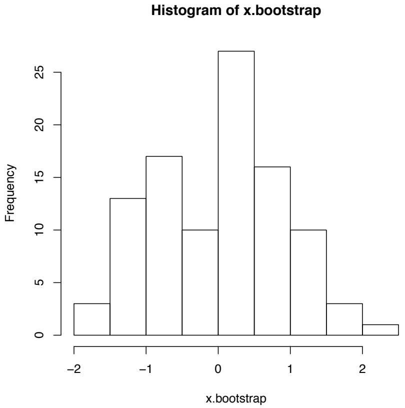
- What is a good estimator for μ?

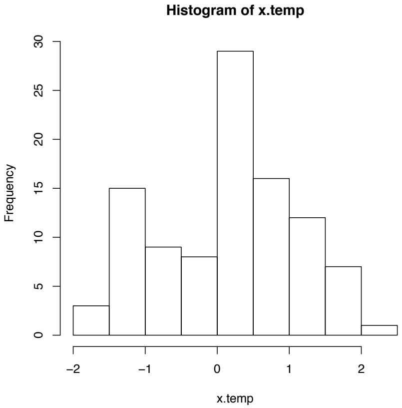
- How can we estimate $\mathrm{Var}(\hat{\mu})$
  1. for $b = 1 : 1000$:
     - resample $x_1, \ldots, x_{100}$ *with replacement* to get $x_1^{(b)}, \ldots, x_{100}^{(b)}$
     - compute $\hat{\mu}(x^{(b)}) = \frac{1}{100} \sum_{i=1}^{100} x_i^{(b)}$
  2. set $\hat{\mu} = \frac{1}{1000} \sum_{b=1}^{1000} \hat{\mu}(x^{(b)})$
  3. set $\hat{\mathrm{Var}} = \frac{1}{1000} \sum_{b=1}^{1000} \left(\hat{\mu}(x^{(b)}) - \hat{\mu}\right)^2 - \hat{\mu}^2$

- True MSE: $\left(\frac{\sigma}{\sqrt{n}}\right)^2 = \frac{1}{100} = 0.01$

Bootstrap: Example

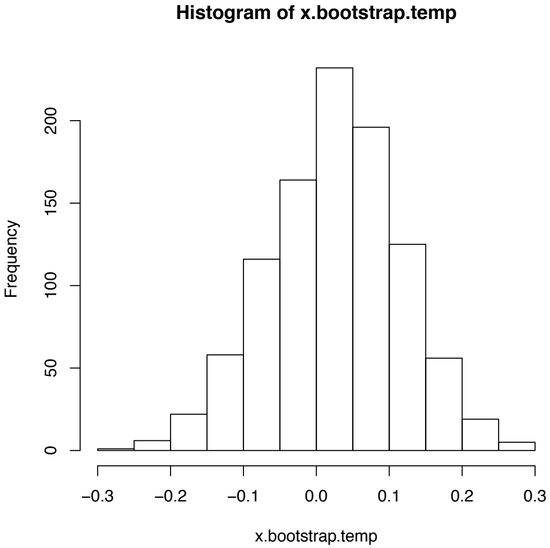
100 data points, $X_1, \ldots, X_{100} \sim N(0,1)$: original and bootstrap sample





Bootstrap: Example

100 data points, $X_1, \ldots, X_{100} \sim N(0,1)$: bootstrapped means



Bootstrap: Example

Let's code this and see if we get what we expect:

```
> n <- 100
> B <- 1000
> x.temp <- rnorm(n)
> bootstrap.mean <- rep(0,B)
> for (i in 1:B){
>   x.bootstrap <- sample(x.temp,n,replace=T)
>  bootstrap.mean[i] <- mean(x.bootstrap)
> }
> mu.bar <- mean(bootstrap.mean)
> mean.var <- mean((bootstrap.mean-mu.bar)^2)
```

Example: Fitting a Model to Stock Returns

When working with stock data:

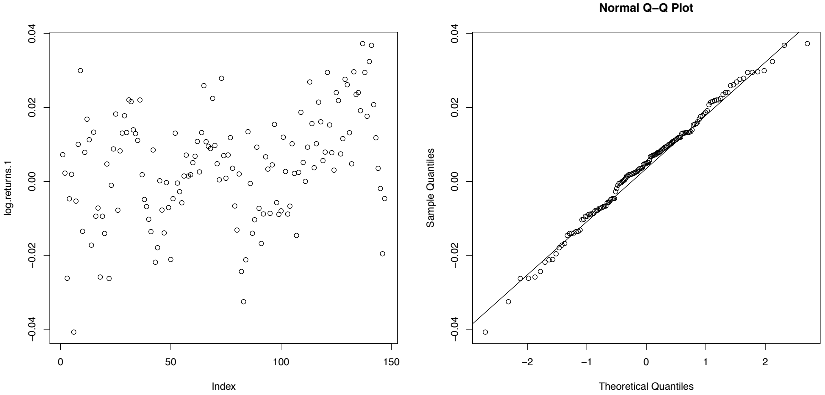
- all stock values should be positive ($S_t \geq 0$)
- and returns are multiplicative (if I invest $X and a stock as a Y% return, I now have $X(1.0Y))
- so usually we work with **log returns**

$$\text{log return at time } t = \log\left(\frac{S_t}{S_{t-1}}\right)$$

Under some financial models, log returns are assumed to have a Gaussian distribution with drift μ and volatility σ (implies prices follow a geometric Brownian motion)

Example: Fitting a Model to Stock Returns

Here is an example set of log returns and their Q-Q plot:



Example: Fitting a Model to Stock Returns

To fit a Gaussian distribution to the data, we need to find a mean μ and variance σ^2. Let's use the bootstrap to look at the distribution of those estimators.

```
> # I have loaded the returns as log.returns.1
> n <- length(log.returns.1)
> B <- 1000

> mu.vec <- rep(0,B)
> sigma2.vec <- rep(0,B)

> for (i in 1:B){
>  x.bootstrap <- sample(log.returns.1,n,replace=T)
>  mu.vec[i] <- mean(x.bootstrap)
>  sigma2.vec[i] <- var(x.bootstrap)
>}
```

Example: Fitting a Model to Stock Returns

Let's use the bootstrapped mean and variance samples to make some confidence intervals for those parameters:

```
> # Let's make some empirical confidence intervals
> mu.sort <- sort(mu.vec)
> sigma2.sort <- sort(sigma2.vec)

> mu.95.CI <- c(mu.sort[25],mu.sort[975])
> sigma2.95.CI <- c(sigma2.sort[25],sigma2.sort[975])
```

Bootstrap Methods for Model Selection and Assessment

We can also use bootstrap methods for model selection and assessment:

- averages over your *training set* as well as new potential observations
- can use observations not selected in training set as a validation set

Expected predictive error:

$$\begin{aligned}\mathrm{Err} &= \mathbb{E}_{\mathcal{T}}\mathbb{E}[L(Y, \hat{f}(X)) \mid \mathcal{T}] \\ &\approx \frac{1}{B}\sum_{b=1}^{B}\frac{1}{|B_b^{(-)}|}\sum_{i \in B_b^{(-)}} L(y_i, \hat{f}^b(x_i))\end{aligned}$$

Outline

Example: Assessing 1NN

Suppose we have:

- $X \sim Unif[-5, 5]$
- $Y = \text{sine}(X) + \epsilon$
- $\epsilon \sim N(0, 0.5^2)$

```
> n.train <- 500
> x.train <- runif(n.train,-5,5)
> y.train <- sin(x.train) + 0.5*rnorm(n.train)

> x.test <- seq(-5,5,by=0.01)
> n.test <- length(x.test)
> y.test <- sin(x.test)
> # For comparison
> y.test.noisy <- y.test + 0.5*rnorm(n.test)
> B <- 100
> test.estimation.mat <- mat.or.vec(n.test,B)
> test.err <- rep(0,B)
```

Example: Assessing 1NN

```
> for(i in 1:B){
>   ind.boot <- sample(1:n.train,n.train,replace=T)
>   x.boot <- x.train[ind.boot]
>   y.boot <- y.train[ind.boot]
>   ind.validation <- setdiff(1:n.train,unique(ind.boot))
>   x.val <- x.train[ind.validation]
>   y.val <- y.train[ind.validation]
>   n.val <- length(y.val)
>   err.val <- rep(0,n.val)
>   for (j in 1:n.val){
>     ind.closest <- which.min(abs(x.boot-x.val[j]))
>     err.val[j] <- (y.val[j]-y.boot[ind.closest])^2
>   }
>   bootstrap.err[i] <- mean(err.val)
>   for (j in 1:n.test){
>     ind.closest <- which.min(abs(x.boot-x.test[j]))
>     test.estimation.mat[j,i] <- y.boot[ind.closest]
>   }
>   test.err[i] <- mean((test.estimation.mat[,i]-y.test.noisy)^2)
> }
```

Example: Assessing 1NN

Let's compare the MSE predicted by the bootstrap with the average MSE on the test set.

- average bootstrap MSE
- average MSE of each bootstrap estimator on test set
- lower bound on error

Recall:

$$MSE = \mathrm{Bias}^2(\hat{f}(X)) + \mathrm{Var}(\hat{f}(X)) + \mathrm{Var}(\epsilon)$$

Let's think about these B 1NN estimators for a moment:

- what is the bias of each?
- what is the variance?

Example: Assessing 1NN

What if we averaged the estimators?

Suppose $\hat{f}_1$ and $\hat{f}_2$ that are both 1NN using different datasets. Set

$$\hat{f}_{avg} = \frac{1}{2}\hat{f}_1 + \frac{1}{2}\hat{f}_2$$

Then the MSE of $\hat{f}_{avg}$ is

$$\begin{aligned} MSE &= \mathrm{Bias}^2(\frac{1}{2}\hat{f}_1 + \frac{1}{2}\hat{f}_2) + \mathrm{Var}(\frac{1}{2}\hat{f}_1 + \frac{1}{2}\hat{f}_2) + \mathrm{Var}(\epsilon) \\ &= \mathrm{Bias}^2(\hat{f}_1) + \frac{2}{2^2}\mathrm{Var}(\hat{f}_1) + \frac{2}{2^2}\mathrm{Cov}(\hat{f}_1, \hat{f}_2) + \mathrm{Var}(\epsilon) \\ &= \mathrm{Bias}^2(\hat{f}_1) + \frac{1}{B}\mathrm{Var}(\hat{f}_1) + \sum_{i \neq j} \frac{1}{B^2}\mathrm{Cov}(\hat{f}_i, \hat{f}_j) + \mathrm{Var}(\epsilon) \end{aligned}$$

So averaging these estimators keeps the bias the same, but reduces the variance! The more uncorrelated the estimators the better!

Averaging bootstrapped estimators is called **bagging**.

Outline

Bootstrap Methods

Bootstrapping is very flexible:

- bootstrapping gives you a distribution over estimators
  - approximate more complicated metrics (medians, quantiles, etc)
  - approximate distributional properties
- create confidence intervals
- average bootstrapped estimators to produce new, superior estimator (bagging)

Bootstrap Methods

Reasons to use the bootstrap:

- very simple
- very flexible
- consistent (estimates are correct) as n goes to ∞
- one of the few methods that works with limited data

Cautions about the bootstrap:

- estimates are optimistic (estimated MSE smaller than true)
- you are limited to your data
  - bootstrap housing price changes from 1970's to 2007: will not capture wild price changes afterwards
- no theoretical guarantees for finite samples
- assumes independence of samples