

Introduction to Statistical learning

Omid Safarzadeh

January 18, 2022

Table of contents

- 1 Regression
- 2 Simple linear regression
 - The least squares approach
 - (Multiple) linear regression
- 3 Bias-variance tradeoff
- 4 Validation
 - Leave-one-out cross-validation
 - k-fold cross validation
- 5 Reference

***Acknowledgement:** This slide is prepared based on Murphy, 2012 and James et al., 2013

Model:

- $Y \approx f(\mathbf{X}, \beta_{true})$
- $\mathbf{X} = [X_1, \dots, X_p]^T$: independent variable vector
- Y : dependent variable
- β^{true} : unknown parameter vector
- Form of f is assumed to be known
- Usually

$$E[Y|\mathbf{X}] = f(\mathbf{X}, \beta^{true})$$

Some applications:

- Stock market prediction given macro economic variables
- Customer satisfaction vs waiting time
- Smart grid - load prediction
- Auto car sales prediction

Simple linear regression

Model:

- Y : response variable, X : predictor variable
- $Y \approx \beta_0^{true} + \beta_1^{true} X$

The least squares approach

- Data: $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$
- RSS: Residual sum of squares

$$RSS(\beta_0, \beta_1) = \sum_{i=1}^n \underbrace{(y_i - (\beta_0 + \beta_1 x_i))}_{\text{residual}}^2$$

Least squares method:

$$(\hat{\beta}_0, \hat{\beta}_1) := \arg \min_{(\beta_0, \beta_1) \in \mathbb{R}^2} RSS(\beta_0, \beta_1)$$

Properties of $\hat{\beta}_0$ and $\hat{\beta}_1$

Unbiasedness:

- $E[\hat{\beta}_0] = \beta_0^{true}$
- $E[\hat{\beta}_1] = \beta_1^{true}$

Variance:

- $\sigma^2 = \text{Var}(\epsilon)$, ϵ_i s are uncorrelated
- $\text{Var}(\hat{\beta}_0) = \sigma^2 \left[\frac{1}{n} + \frac{\bar{x}^2}{\sum_{i=1}^n (x_i - \bar{x})^2} \right]$
- $\text{Var}(\hat{\beta}_1) = \frac{\sigma^2}{\sum_{i=1}^n (x_i - \bar{x})^2}$

(Multiple) linear regression

Given $[X - 1, \dots, X_p]$ predict Y via a linear model:

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

Let $X = [1, X_1, \dots, X_p]^T$ and $\hat{\beta} = [\hat{\beta}_0, \hat{\beta}_1, \dots, \hat{\beta}_p]^T$. Then,

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

The least squares approach

- Data: $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$
- $\mathbf{x}_i = [1, X_{i1}, \dots, X_{ip}]^T$
- RSS: Residual sum of squares

$$RSS(\beta) = \sum_{i=1}^n (y_i - \mathbf{x}_i^T \beta)^2$$

Least squares method:

$$\hat{\beta}_{RSS} = \underset{\beta \in \mathbb{R}^{p+1}}{\arg \min} RSS(\beta)$$

The least squares solution

Some notation:

$$\mathbf{y}_{n \times 1} \quad \hat{\mathbf{y}}_{n \times 1} \quad \mathbf{X}_{n \times p}$$

Least squares solution:

$$\begin{aligned} \hat{\beta}_{RSS} &= \arg \min_{\beta \in \mathbb{R}^{p+1}} RSS(\beta) = \arg \min_{\beta \in \mathbb{R}^{p+1}} \sum_{i=1}^n (y_i - \mathbf{x}_i^T \beta)^2 \\ \Rightarrow \hat{\beta}_{RSS} &= (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y} \end{aligned}$$

- Assumption: \mathbf{X} has full column rank!
- Then $\hat{\mathbf{y}} = \mathbf{X} \hat{\beta}_{RSS} = \underbrace{\mathbf{X}(\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T}_{\text{projection matrix } \mathbf{H}} \mathbf{y}$

Gauss-Markov Theorem

Assumptions:

- $y_i = (\beta^{true})^T \mathbf{x}_i + \epsilon_i, i = 1, \dots, n$
- $E[\epsilon_i] = 0, \text{Var}(\epsilon_i) = \sigma^2, \text{Cov}(\epsilon_i, \epsilon_j) = 0, i \neq j$

Linear estimator of β_j^{true} :

- $\hat{\beta}_j = c_{1j}y_1 + \dots + c_{nj}y_n$
- c_{kj} : possibly non-linear function of \mathbf{X}

Theorem 2.1

Least squares estimate of β^{true} true have the smallest variance among all linear unbiased estimators.

$$\text{Unbiased estimator: } E[\hat{\beta}] = \beta^{true}$$

$$\text{"Best linear unbiased estimator" (BLUE)} = \hat{\beta}_{RSS}$$

MLE of linear regression

- $y_i = (\beta^{\text{true}})^T \mathbf{x}_i + \epsilon_i$, (unknown)
- i.i.d. $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$
- We have access to $\mathcal{D} := \{(\mathbf{x}_i, y_i)\}_{i=1}^n$

Likelihood:

$$L(\beta) = \prod_{i=1}^n \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(y_i - \beta^T \mathbf{x}_i)^2}{2\sigma^2}\right)$$

Log likelihood:

$$l(\beta) := \log L(\beta) = n \log \frac{1}{\sqrt{2\pi}\sigma} - \frac{1}{2\sigma^2} \sum_{i=1}^n (y_i - \beta^T \mathbf{x}_i)^2$$

Probabilistic interpretation of the solution

- Maximizing $L(\beta)$ is equivalent to maximizing $l(\beta)$ since $l(\beta)$ is a strictly increasing function of $L(\beta)$!

$$\arg \max_{\beta} l(\beta) = \arg \min_{\beta} RSS(\beta) \Rightarrow \hat{\beta}_{MLE} = \hat{\beta}_{RSS}$$

- Note: MLE is independent of σ^2

Measuring the quality of fit

- Data: $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$, $\mathbf{x}_i \in \mathbb{R}^p$, $y_i \in \mathbb{R}$
- MSE: mean square error

$$MSE := \frac{1}{n} \sum_{i=1}^n (y_i - \hat{f}(\mathbf{x}_i))^2$$

- Example: linear regression $\hat{f}(\mathbf{x}_i)$?

Goal: We want linear regression to perform well on unseen data. Not on \mathcal{D} !

Can we get $RSS = 0$ in \mathcal{D} ?

Measuring the quality of fit

- Usually we are provided with a test set \mathcal{D}_{test} to measure how well our algorithm performs
- \mathcal{D}_{train} : dataset in which we trained (and fixed) our model.

$$MSE_{train} := \frac{1}{n_{train}} \sum_{(\mathbf{x}_i, y_i) \in \mathcal{D}_{train}} (y_i - \hat{f}(\mathbf{x}_i))^2$$

$$MSE_{test} := \frac{1}{n_{test}} \sum_{(\mathbf{x}_i, y_i) \in \mathcal{D}_{test}} (y_i - \hat{f}(\mathbf{x}_i))^2$$

Measuring the quality of fit

Example 2.1

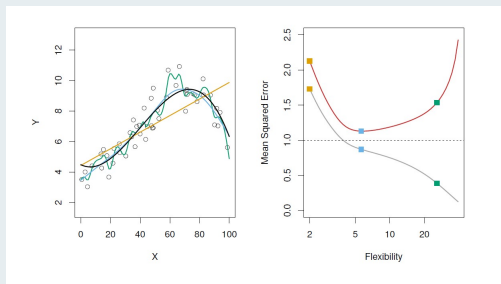


Figure: James et al., 2013

- Data generated by non-linear model
- LHS: fit to \mathcal{D}_{train}
- Red curve: MSE_{test} , gray curve: MSE_{train}

Measuring the quality of fit

Example 2.2

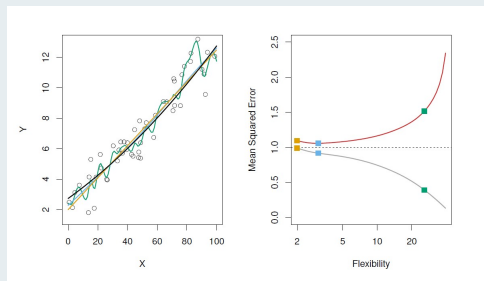


Figure: James et al., 2013

- Data generated by noisy linear model
- LHS: fit to \mathcal{D}_{train}
- Red curve: MSE_{test} , gray curve: MSE_{train}

Bias-variance tradeoff

- $(\mathbf{x}_i, y_i) \sim p(\mathbf{X}, Y)$
- $(\mathbf{x}_i, y_i), i = 1, \dots, n$ is i.i.d.
- $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ (random dataset).
 $\mathcal{D} \sim J$ where $J := \prod_{i=1}^n p(\mathbf{X}_i, Y_i)$

Expected label:

$$\bar{y}(\mathbf{x}) = E[Y|\mathbf{X}=\mathbf{x}] = \int_y y p(y|\mathbf{x}) dy$$

Statistical learning algorithm \mathcal{A} :

$$\hat{f}_{\mathcal{D}} = \mathcal{A}(\mathcal{D})$$

Then, given $\mathbf{x} \in \mathbb{R}^p$, we have the prediction $\hat{y} = \hat{f}_{\mathcal{D}}(\mathbf{x})$

Tradeoff:

$$\begin{aligned} & \underbrace{E_{(\mathbf{X}, Y) \sim p, \mathcal{D} \sim J}[(\hat{f}_{\mathcal{D}}(\mathbf{X}) - Y)^2]}_{\text{expected squared test error}} \\ &= \underbrace{E_{\mathbf{X}}[(\bar{f}(\mathbf{X}) - \bar{y}(\mathbf{X}))^2]}_{\text{bias}^2} + \underbrace{E_{\mathbf{X}, \mathcal{D}}[(\hat{f}_{\mathcal{D}}(\mathbf{X}) - \bar{f}(\mathbf{X}))^2]}_{\text{variance}} \\ & \quad + \underbrace{E_{\mathbf{X}, Y}[(\bar{y}(\mathbf{X}) - Y)^2]}_{\text{noise}} \end{aligned}$$

Bias-variance tradeoff

- Bias: Inherent error in your model (ex: due to the linearity assumption)
- Variance: Change in the model if trained on a different training set (over-specialization).
- Noise: Intrinsic noise in the data generation process (property of the data).

Which terms can you change by changing the learning algorithm \mathcal{A} ?

Illustration of bias-variance tradeoff

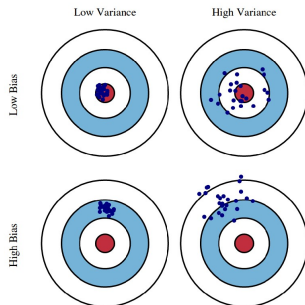


Figure: Ref.

- Each point represents the test accuracy given a random realization of the training set and the test set.
- Points close to the center imply good test accuracy.

Illustration of bias-variance tradeoff

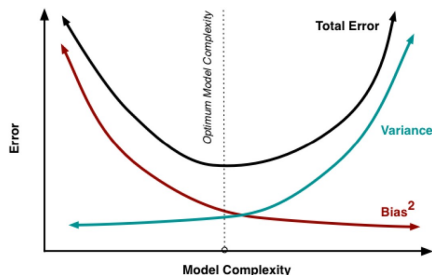


Figure: <http://scott.fortmann-roe.com/docs/BiasVariance.html>

- Black line is the test error
- High bias = underfit
- High variance = overfit

Illustration of bias-variance tradeoff

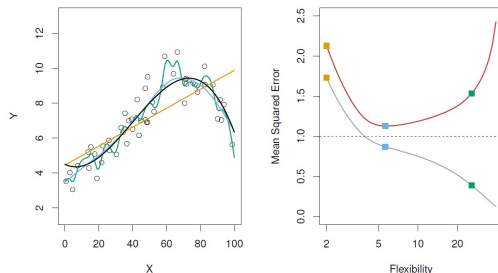


Figure: James et al., 2013

- Flexibility \approx model complexity
- Data generated by non-linear model
- Red curve: MSE_{test} , gray curve: MSE_{train}

What if we don't have a separate test set?

- MSE_{train} provides no information on how well our algorithm generalizes
- Divide \mathcal{D} randomly into \mathcal{D}_{train} and $\mathcal{D}_{validation}$
- Up: original dataset. Blue: training dataset. Beige: validation dataset
- Compute \hat{f} based on \mathcal{D}_{train} . Evaluate \hat{f} based on $\mathcal{D}_{validation}$



Figure: James et al., 2013

Leave-one-out cross-validation

- Train n models
- Train i th model on $\mathcal{D}_i = \mathcal{D} - \{(\mathbf{x}_i, y_i)\}$
- Test i th model on (\mathbf{x}_i, y_i)
- Blue: training dataset. Beige: validation dataset

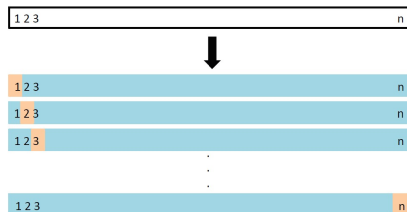


Figure: James et al., 2013

Leave-one-out cross-validation

- $MSE_i = (y_i - \hat{f}_{\mathcal{D}_i}(\mathbf{x}_i))^2$
- $CV_{(n)} = \frac{1}{n} \sum_{i=1}^n MSE_i$

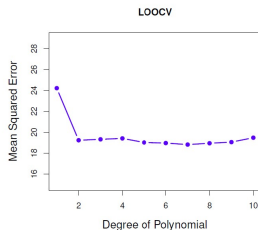


Figure: James et al., 2013

- No randomness in $CV_{(n)}$
- No over-estimation of test error

k-fold cross validation

- Randomly split D into k equal sized folds: $\mathcal{F}_1, \dots, \mathcal{F}_k$
- Train i th model on all folds except fold i : $\mathcal{D}_i := \mathcal{D} - \mathcal{F}_i$
- Test i th model on fold i : \mathcal{F}_i
- Blue: training dataset. Beige: validation dataset

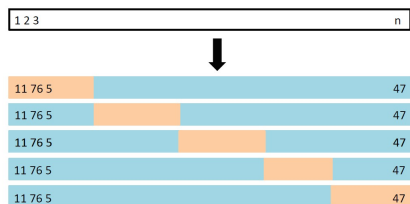


Figure: James et al., 2013

k-fold cross validation

$$MSE_i = \frac{1}{|\mathcal{F}_i|} \sum_{(\mathbf{x}_j, y_j) \in \mathcal{F}_i} (y_j - \hat{f}_{\mathcal{D}_i}(\mathbf{x}_j))^2$$

$$CV_{(k)} = \frac{1}{k} \sum_{i=1}^k MSE_i$$

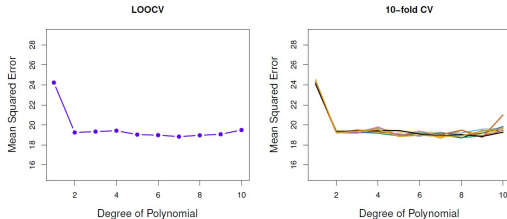


Figure: James et al., 2013

- James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). *An introduction to statistical learning: With applications in r*. Springer New York.
https://books.google.fr/books?id=qcl%5C_AAAAQBAJ
- Murphy, K. (2012). *Machine learning: A probabilistic perspective*. MIT Press.
<https://books.google.fr/books?id=NZP6AQAAQBAJ>