Resampling Methods

Cross-validation, Bootstrapping

Marek Petrik

2/21/2017

So Far in ML

- Regression vs classification
- Linear regression
- Logistic regression
- Linear discriminant analysis, QDA
- Maximum likelihood

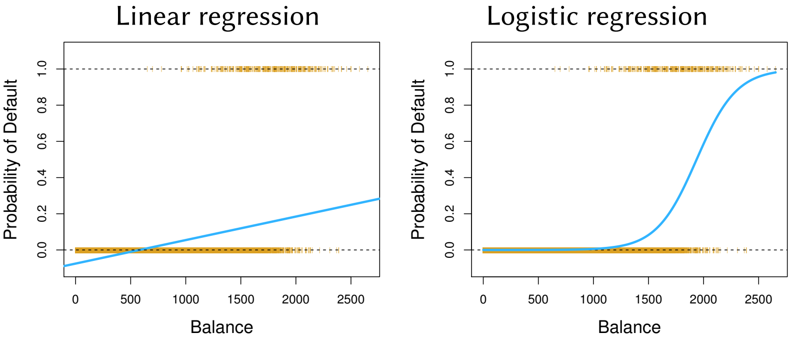
Discriminative vs Generative Models

- **Discriminative models**
  - Estimate conditional models $\Pr[Y \mid X]$
  - Linear regression
  - Logistic regression

- **Generative models**
  - Estimate joint probability $\Pr[Y, X] = \Pr[Y \mid X] \Pr[X]$
  - Estimates not only probability of labels but also the features
  - Once model is fit, can be used to generate data
  - LDA, QDA, Naive Bayes

Logistic Regression

$$Y = \begin{cases} 1 & \text{if default} \\ 0 & \text{otherwise} \end{cases}$$

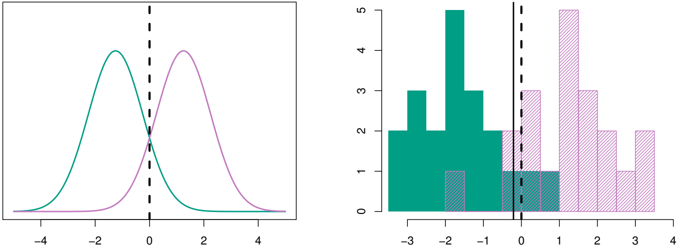


Predict:

$$\Pr[\text{default} = \text{yes} \mid \text{balance}]$$

LDA: Linear Discriminant Analysis

- **Generative model**: capture probability of predictors for each label



- Predict:
  1. $\Pr[\text{balance} \mid \text{default} = \text{yes}]$ and $\Pr[\text{default} = \text{yes}]$
  2. $\Pr[\text{balance} \mid \text{default} = \text{no}]$ and $\Pr[\text{default} = \text{no}]$
- Classes are normal: $\Pr[\text{balance} \mid \text{default} = \text{yes}]$

Bayes Theorem

- Classification from label distributions:

$$\Pr[Y = k \mid X = x] = \frac{\Pr[X = x \mid Y = k] \Pr[Y = k]}{\Pr[X = x]}$$

- Example:

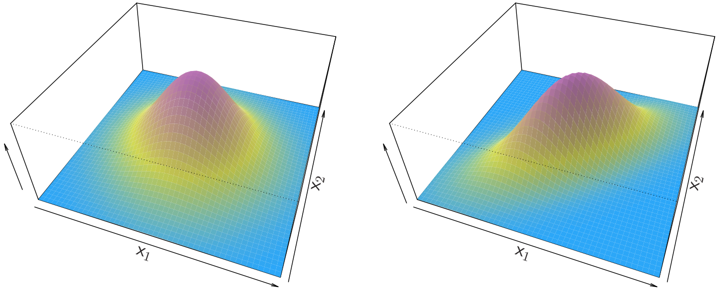
$$\Pr[\text{default} = \text{yes} \mid \text{balance} = \$100] = \frac{\Pr[\text{balance} = \$100 \mid \text{default} = \text{yes}] \Pr[\text{default} = \text{yes}]}{\Pr[\text{balance} = \$100]}$$

- Notation:

$$\Pr[Y = k \mid X = x] = \frac{\pi_k f_k(x)}{\sum_{l=1}^{K} \pi_l f_l(x)}$$

LDA with Multiple Features

- Multivariate Normal Distributions:

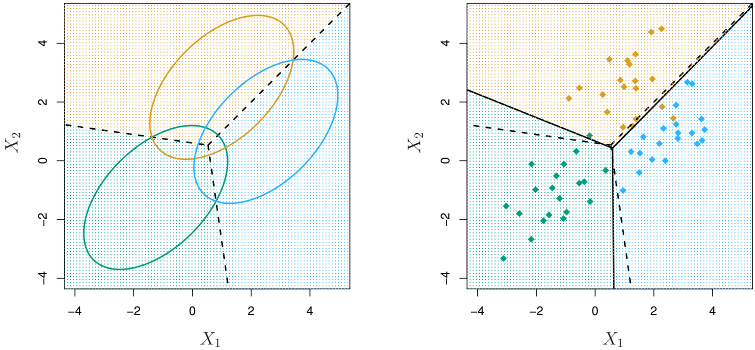


- Multivariate normal distribution density (mean vector μ, covariance matrix Σ):

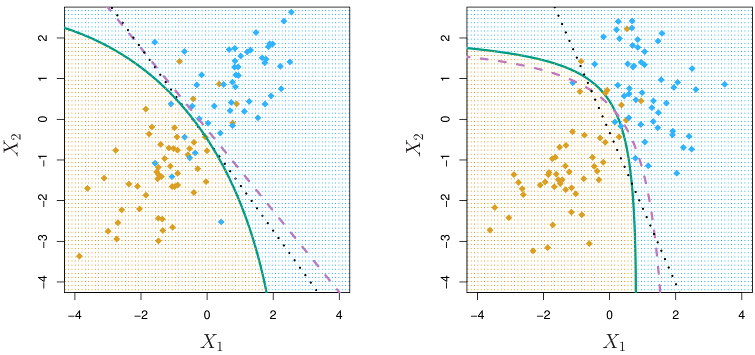
$$p(X) = \frac{1}{(2\pi)^{p/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu)^\top \Sigma^{-1}(x-\mu)\right)$$

Multivariate Classification Using LDA

- **Linear**: Decision boundaries are linear



QDA: Quadratic Discriminant Analysis



Confusion Matrix: Predict default

		True		
		Yes	No	Total
Predicted	Yes	a	b	$a+b$
	No	c	d	$c+d$
	Total	$a+c$	$b+d$	N

Result of LDA classification: Predict default if
$\Pr[\text{default} = \text{yes} \mid \text{balance}] > 1/2$

		True		
		Yes	No	Total
Predicted	Yes	81	23	104
	No	252	9 644	9 896
	Total	333	9 667	10 000

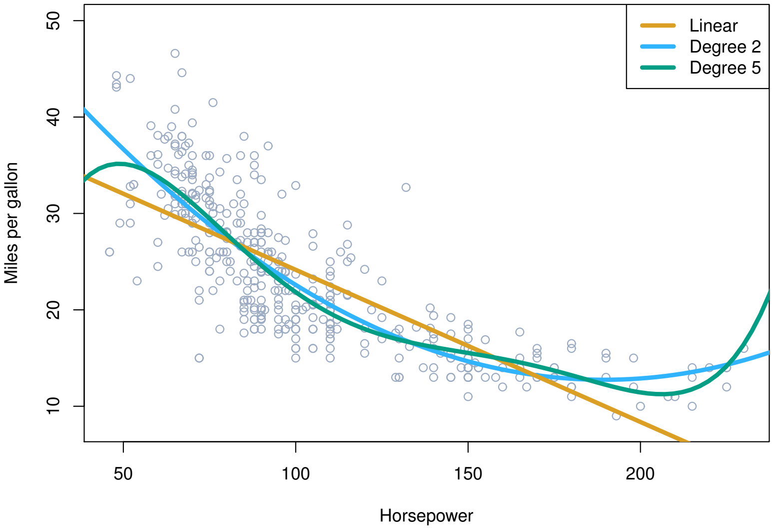
Today

- Successfully using basic machine learning methods
- Problems:
  1. How well is the machine learning method doing
  2. Which method is best for my problem?
  3. How many features (and which ones) to use?
  4. What is the uncertainty in the learned parameters?

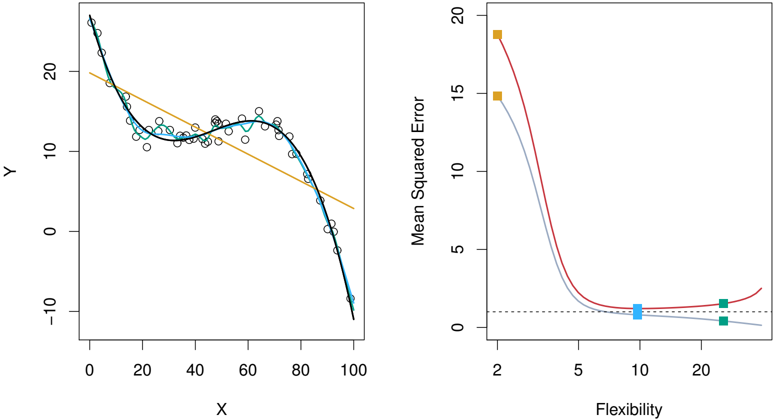
Today

- Successfully using basic machine learning methods
- Problems:
    1. How well is the machine learning method doing
    2. Which method is best for my problem?
    3. How many features (and which ones) to use?
    4. What is the uncertainty in the learned parameters?
- Methods:
    1. Validation set
    2. Leave one out cross-validation
    3. k-fold cross validation
    4. Bootstrapping

Problem: How to design features?



Benefit of Good Features



gray: training error　　red: test error

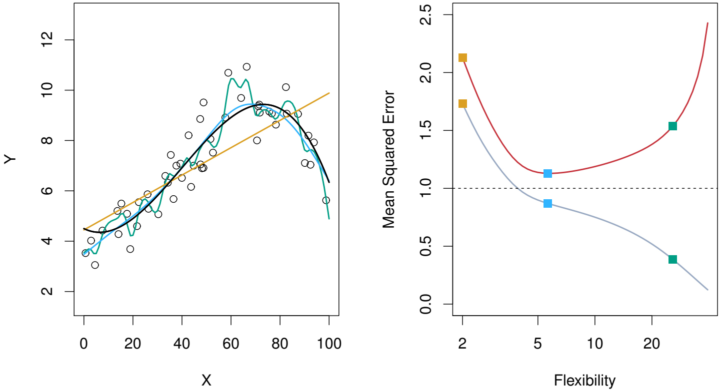
Just Use Training Data?

Just Use Training Data?

- Using more features will always reduce MSE

Just Use Training Data?

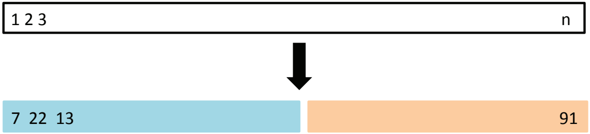
- Using more features will always reduce MSE
- Error on the test set will be greater



gray: training error red: test error

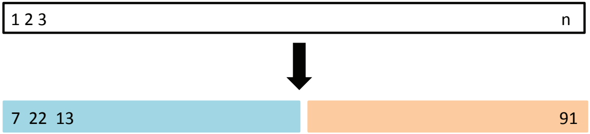
Solution 1: Validation Set

- Just evaluate how well the method works on the test set
- **Randomly** split data to:
  1. Training set: about half of all data
  2. Validation set (AKA hold-out set): remaining half



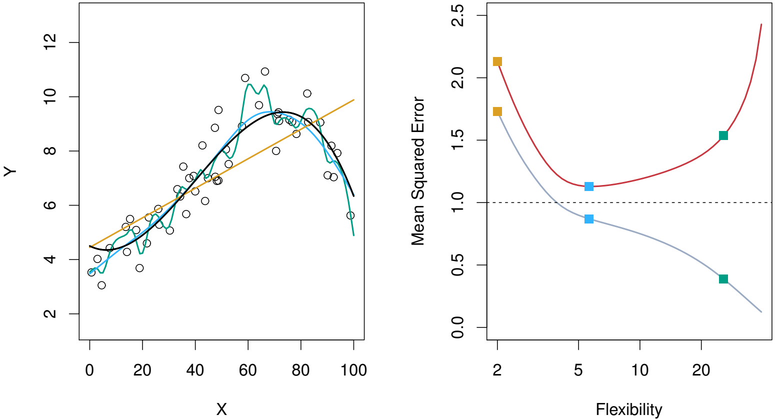
Solution 1: Validation Set

- Just evaluate how well the method works on the test set
- **Randomly** split data to:
  1. Training set: about half of all data
  2. Validation set (AKA hold-out set): remaining half



- Choose the number of features/representation based on minimizing error on **validation set**

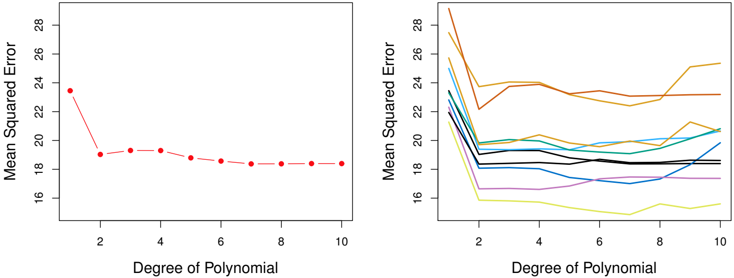
Feature Selection Using Validation Set



gray: training error   red: test error (validation set)

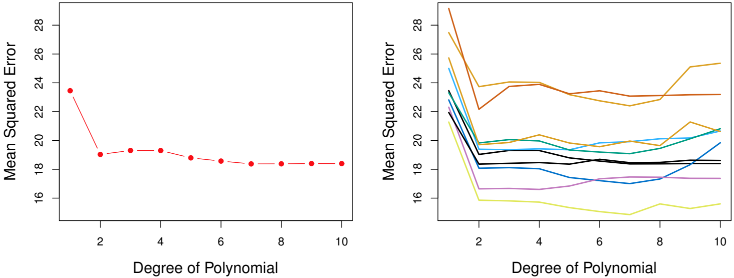
Problems using Validation Set

1. **Highly variable (imprecise) estimates**: Each line shows validation error for one possible division of data



Problems using Validation Set

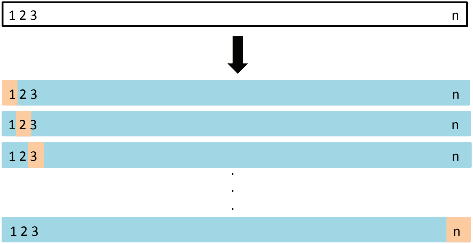
1. **Highly variable (imprecise) estimates**: Each line shows validation error for one possible division of data



2. **Only subset of data is used** (validation set is excluded – only about half of data is used)

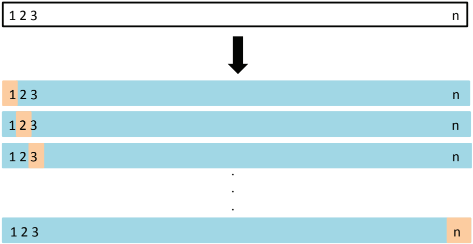
Solution 2: Leave-one-out

- Addresses problems with validation set
- Split the data set into 2 parts:
  1. Training: Size $n - 1$
  2. Validation: Size 1
- Repeat n times: to get n learning problems



Leave-one-out

- Get n learning problems:



- Train on $n-1$ instances (blue)
- Test on 1 instance (red)

$$\text{MSE}_i = (y_i - \hat{y}_i)^2$$

- LOOCV estimate

$$\text{CV}_{(n)} = \frac{1}{n}\sum_{i=1}^{n} \text{MSE}_i$$

Leave-one-out vs Validation Set

- Advantages

Leave-one-out vs Validation Set

- Advantages
  1. Using almost all data not just half

Leave-one-out vs Validation Set

- Advantages
  1. Using almost all data not just half
  2. Stable results: Does not have any randomness

Leave-one-out vs Validation Set

- Advantages
  1. Using almost all data not just half
  2. Stable results: Does not have any randomness
  3. Evaluation is performed with more test data

Leave-one-out vs Validation Set

- Advantages
  1. Using almost all data not just half
  2. Stable results: Does not have any randomness
  3. Evaluation is performed with more test data
- Disadvantages

Leave-one-out vs Validation Set

- Advantages
  1. Using almost all data not just half
  2. Stable results: Does not have any randomness
  3. Evaluation is performed with more test data
- Disadvantages
  - Can be very computationally expensive: Fits the model n times

Speeding Up Leave-One-Out

1. Solve each fit independently and distribute the computation

Speeding Up Leave-One-Out

1. Solve each fit independently and distribute the computation

2. **Linear regression**:

Speeding Up Leave-One-Out

1. Solve each fit independently and distribute the computation

2. **Linear regression**:
   - Solve only one linear regression using **all** data

Speeding Up Leave-One-Out

1. Solve each fit independently and distribute the computation

2. **Linear regression**:
   - Solve only one linear regression using **all** data
   - Compute leave-one-out error as:

$$\mathrm{CV}_{(n)} = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \hat{y}_i}{1 - h_i}\right)^2$$

Speeding Up Leave-One-Out

1. Solve each fit independently and distribute the computation

2. **Linear regression**:
   - Solve only one linear regression using **all** data
   - Compute leave-one-out error as:

$$\mathrm{CV}_{(n)} = \frac{1}{n} \sum_{i=1}^{n} \left(\frac{y_i - \hat{y}_i}{1 - h_i} \right)^2$$

   - True value: y_i, Prediction: $\hat{y}_i$

Speeding Up Leave-One-Out

1. Solve each fit independently and distribute the computation

2. **Linear regression**:
   - Solve only one linear regression using **all** data
   - Compute leave-one-out error as:

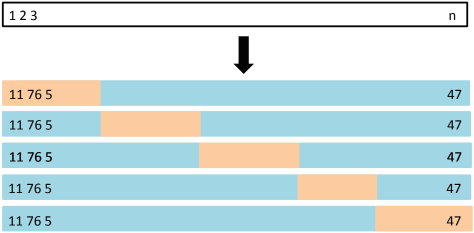
$$\mathrm{CV}_{(n)} = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \hat{y}_i}{1 - h_i}\right)^2$$

   - True value: y_i, Prediction: $\hat{y}_i$
   - h_i is the leverage of data point i:

$$h_i = \frac{1}{n} + \frac{(x_i - \bar{x})^2}{\sum_{j=1}^{n}(x_j - \bar{x})^2}$$

Solution 3: k-fold Cross-validation

- Hybrid between validation set and LOO
- Split training set into k subsets
  1. Training set: $n - {}^{n}/_{k}$
  2. Test set: ${}^{n}/_{k}$
- k learning problems

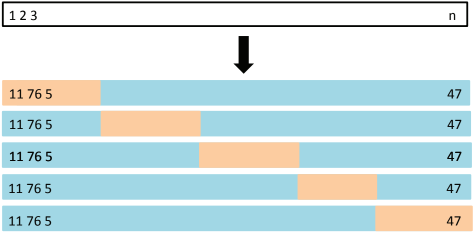


- Cross-validation error:

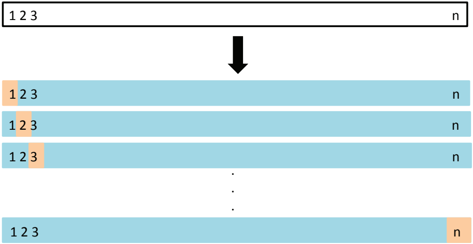
$$\mathrm{CV}_{(k)} = \frac{1}{k} \sum_{i=1}^{k} \mathrm{MSE}_i$$

Cross-validation vs Leave-One-Out

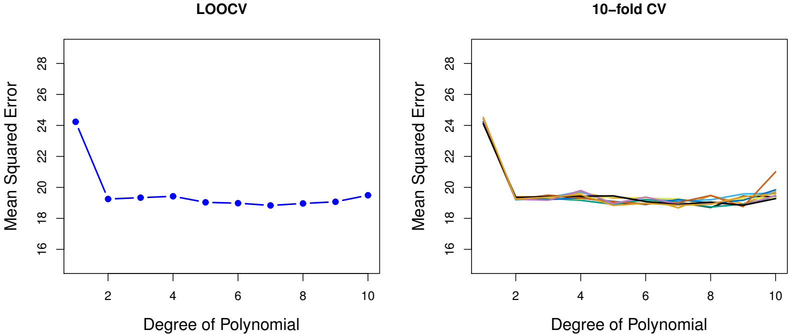
- k-fold Cross-validation



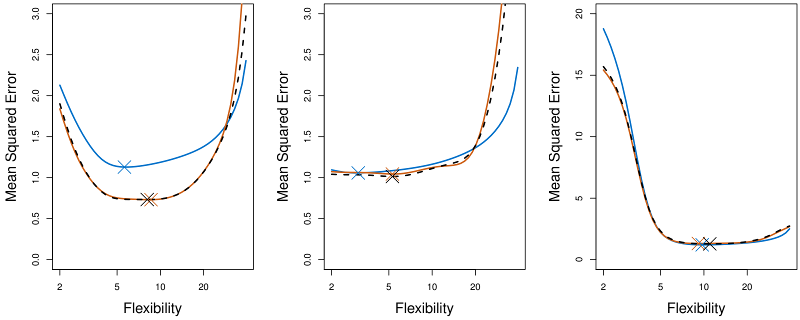
- Leave-one-out



Cross-validation vs Leave-One-Out



Empirical Evaluation: 3 Examples



Blue True error

Dashed LOOCV estimate

Orange 10-fold CV

How to Choose k in CV?

- As k increases we have:
    1. Increasing computational complexity
    2. Decreasing bias (more training data)
    3. Increasing variance (bigger overlap between training sets)
- Empirically good values: 5 - 10

Cross-validation in Classification

Logistic Regression

- Predict **probability** of a class: $p(X)$
- Example: $p(\text{balance})$ probability of default for person with balance
- **Linear regression**:

$$p(X) = \beta_0 + \beta_1$$

- **Logistic regression**:

$$p(X) = \frac{e^{\beta_0 + \beta_1 X}}{1 + e^{\beta_0 + \beta_1 X}}$$

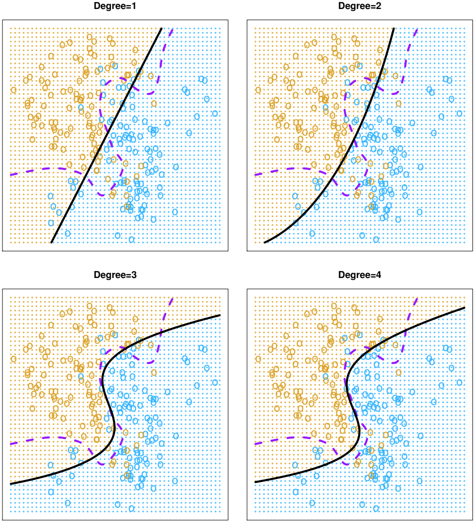
- the same as:

$$\log \left(\frac{p(X)}{1 - p(X)} \right) = \beta_0 + \beta_1 X$$

- Linear decision boundary (derive from log odds: $p(x_1) \geq p(x_2)$)

Features in Logistic Regression

Logistic regression decision boundary is also linear … non-linear decisions?



Logistic Regression with Nonlinear Features

- Linear:

$$\log\left(\frac{p(X)}{1-p(X)}\right) = \beta_0 + \beta_1\, X$$

- Nonlinear odds:

$$\log\left(\frac{p(X)}{1-p(X)}\right) = \beta_0 + \beta_1\, X + \beta_2\, X^2 + \beta_3\, X^3$$

- Nonlinear probability:

$$p(X) = \frac{e^{\beta_0+\beta_1\, X+\beta_2\, X^2+\beta_3\, X^3}}{1+e^{\beta_0+\beta_1\, X+\beta_2\, X^2+\beta_3\, X^3}}$$

Cross-validation in Classification

- Works the same as for regression
- Do not use MSE but:

$$\mathrm{CV}_{(n)} = \frac{1}{n}\sum_{i=1}^{n}\mathrm{Err}_i$$

- Error is an indicator function:

$$\mathrm{Err}_i = I(y_i \neq \hat{y}_i)$$

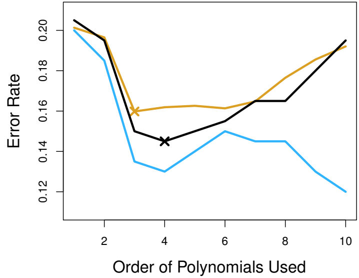
K in KNN

- How to decide on the right k to use in KNN?

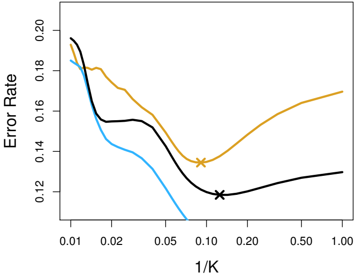
K in KNN

- How to decide on the right k to use in KNN?
- Cross-validation!

Logistic regression



KNN



Brown	Test error
Blue	Training error
Black	CV error

Overfitting and CV

- Is it possible to overfit when using cross-validation?

Overfitting and CV

- Is it possible to overfit when using cross-validation?
- **Yes!**

Overfitting and CV

- Is it possible to overfit when using cross-validation?
- **Yes!**
- Inferring k in KNN using cross-validation is learning

Overfitting and CV

- Is it possible to overfit when using cross-validation?
- **Yes!**
- Inferring k in KNN using cross-validation is learning
- Insightful theoretical analysis: Probably Approximately Correct (PAC) Learning

Overfitting and CV

- Is it possible to overfit when using cross-validation?
- **Yes!**
- Inferring k in KNN using cross-validation is learning
- Insightful theoretical analysis: Probably Approximately Correct (PAC) Learning
  - Cross-validation will not overfit when learning simple concepts

Overfitting with Cross-validation

- Task: Predict mpg $\sim$ power
- Define a new feature for some βs:

$$\mathsf{f} = \beta_0 + \beta_1\, \mathsf{power} + \beta_2\, \mathsf{power}^2 + \beta_3\, \mathsf{power}^3 + \beta_4\, \mathsf{power}^4 + \dots$$

- **Linear regression**: Find α such that:

$$\mathsf{mpg} = \alpha\, \mathsf{f}$$

- **Cross-validation**: Find values of βs

Overfitting with Cross-validation

- Task: Predict $\mathsf{mpg} \sim \mathsf{power}$
- Define a new feature for some βs:

$$\mathsf{f} = \beta_0 + \beta_1\, \mathsf{power} + \beta_2\, \mathsf{power}^2 + \beta_3\, \mathsf{power}^3 + \beta_4\, \mathsf{power}^4 + \ldots$$

- **Linear regression**: Find α such that:

$$\mathsf{mpg} = \alpha\, \mathsf{f}$$

- **Cross-validation**: Find values of βs
- Will overfit
- Same solution as using linear regression on entire data (no cross-validation)

Preventing Overfitting

- **Gold standard**: Have a test set that is used **only once**
- Rarely possible
- $1M Netflix prize design:
  1. Publicly available training set
  2. Leader-board results using a test set
  3. Private data set used to determine the final winner

Bootstrap

- **Goal**: Understand the confidence in learned parameters
- Most useful in inference
- How confident are we in learned values of β:

$$\mathsf{mpg} = \beta_0 + \beta_1\ \mathsf{power}$$

Bootstrap

- **Goal**: Understand the confidence in learned parameters
- Most useful in inference
- How confident are we in learned values of β:

$$\text{mpg} = \beta_0 + \beta_1 \,\text{power}$$

- **Approach**: Run learning algorithm multiple times with different data sets:

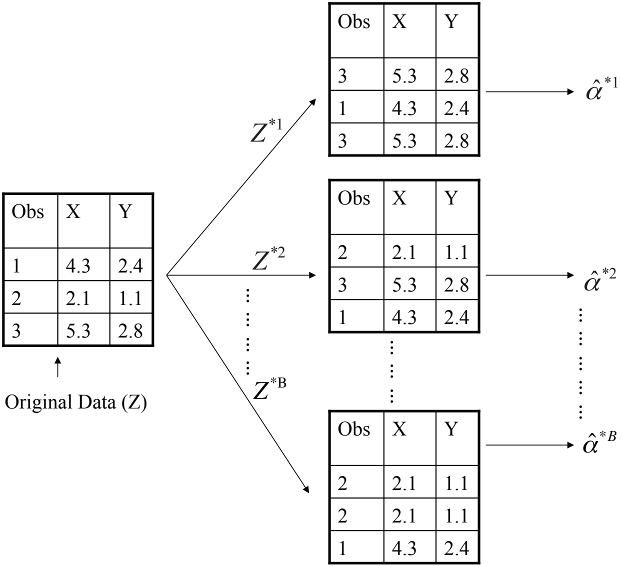
Bootstrap

- **Goal**: Understand the confidence in learned parameters
- Most useful in inference
- How confident are we in learned values of β:

$$\text{mpg} = \beta_0 + \beta_1 \, \text{power}$$

- **Approach**: Run learning algorithm multiple times with different data sets:
- Create a new data-set by sampling **with replacement** from the original one

Bootstrap Illustration



Bootstrap Results