Least-Squares

Stephen Boyd

EE103
Stanford University

October 30, 2014

Outline

Least-squares problem

Solution of least-squares problem

Least-squares data fitting

Least-squares classification

Validation

Least-squares problem

- suppose $m \times n$ matrix A is tall, so $Ax = b$ is over-determined
- for most choices of b, there is no x that satisfies $Ax = b$
- *residual* $r = Ax - b$
- *least-squares problem*: choose x to minimize $\|Ax - b\|^2$
- $\|Ax - b\|^2$ is the *objective function*
- $\hat{x}$ is a *solution* of least-squares problem if

$$\|A\hat{x} - b\|^2 \leq \|Ax - b\|^2$$

  for any n-vector x
- idea: $\hat{x}$ makes residual as small as possible, if not 0
- also called *regression* (in data fitting context)

Least-squares problem

- $\hat{x}$ called *least-squares approximate solution* of $Ax = b$
- $\hat{x}$ is sometimes called, very confusingly, 'the solution of $Ax = b$ in the least-squares sense'
  - never say this
  - do not associate with people who say this

- $\hat{x}$ need not (and usually does not) satisfy $A\hat{x} = b$
- but if $\hat{x}$ does satisfy $A\hat{x} = b$, then it solves least-squares problem

Column interpretation

- suppose $a_1, \dots, a_n$ are columns of A
- then
$$\|Ax - b\|^2 = \|(x_1 a_1 + \cdots + x_n a_n) - b\|^2$$
- so least-squares problem is to find a linear combination of columns of A that is closest to b
- if $\hat{x}$ is a solution of least-squares problem, the m-vector
$$A\hat{x} = \hat{x}_1 a_1 + \cdots + \hat{x}_n a_n$$
is closest to b among all linear combinations of columns of A

Row interpretation

- suppose $\tilde{a}_1^T, \ldots, \tilde{a}_m^T$ are rows of A
- residual components are $r_i = \tilde{a}_i^T x - b_i$
- least-squares objective is

$$\|Ax - b\|^2 = (\tilde{a}_1^T x - b_1)^2 + \cdots + (\tilde{a}_m^T x - b_m)^2$$

  the sum of squares of the residuals
- so least-squares minimizes sum of squares of residuals
  - solving $Ax = b$ is making all residuals zero
  - least-squares attempts to make them all small

Outline

Least-squares problem

Solution of least-squares problem

Least-squares data fitting

Least-squares classification

Validation

Solution of least-squares problem

- we make one assumption: A *has independent columns*
- this implies that Gram matrix $A^T A$ is invertible
- unique solution of least-squares problem is

$$\hat{x} = (A^T A)^{-1} A^T b = A^\dagger b$$

- cf. $x = A^{-1} b$, solution of square invertible system $Ax = b$

Derivation via calculus

- define

$$f(x) = \|Ax - b\|^2 = \sum_{i=1}^{m} \left(\sum_{j=1}^{n} A_{ij} x_j - b_i \right)^2$$

- solution $\hat{x}$ satisfies

$$\frac{\partial f}{\partial x_k}(\hat{x}) = \nabla f(\hat{x})_k = 0, \quad k = 1, \ldots, n$$

- taking partial derivatives we get $\nabla f(x)_k = \big(2A^T(Ax - b)\big)_k$
- in matrix-vector notation: $\nabla f(\hat{x}) = 2A^T(A\hat{x} - b) = 0$
- so $\hat{x}$ satisfies *normal equations* $(A^T A)\hat{x} = A^T b$
- and therefore $\hat{x} = (A^T A)^{-1} A^T b$

Direct verification

- let $\hat{x} = (A^TA)^{-1}A^Tb$, so $A^T(A\hat{x} - b) = 0$
- for any n-vector x we have

$$\begin{array}{rcl} \|Ax - b\|^2 & = & \|(Ax - A\hat{x}) + (A\hat{x} - b)\|^2 \\ & = & \|A(x - \hat{x})\|^2 + \|A\hat{x} - b\|^2 + 2(A(x - \hat{x}))^T(A\hat{x} - b) \\ & = & \|A(x - \hat{x})\|^2 + \|A\hat{x} - b\|^2 + 2(x - \hat{x})^TA^T(A\hat{x} - b) \\ & = & \|A(x - \hat{x})\|^2 + \|A\hat{x} - b\|^2 \end{array}$$

- so for any x, $\|Ax - b\|^2 \geq \|A\hat{x} - b\|^2$
- if equality holds, $A(x - \hat{x}) = 0$, which implies $x = \hat{x}$ since columns of A are independent

Computing least-squares approximate solutions

- compute QR factorization of A: $A = QR$ $\quad (2mn^2$ flops)
  (exists since columns of A are independent)
- to compute $\hat{x} = A^\dagger b = R^{-1}Q^T b$
  - form $Q^T b$ $\quad (2mn$ flops)
  - compute $\hat{x} = R^{-1}(Q^T b)$ via back substitution $\quad (n^2$ flops)
- total complexity $2mn^2$ flops

- identical to algorithm for solving $Ax = b$ for square invertible A
- but when A is tall, gives least-squares approximate solution

Outline

Least-squares problem

Solution of least-squares problem

Least-squares data fitting

Least-squares classification

Validation

Setup

- we believe a scalar y and an n-vector x are related by *model*

$$y \approx f(x)$$

- x is called the *independent variable*
- y is called the *outcome* or *response variable*
- $f : \mathbf{R}^n \to \mathbf{R}$ gives the relation between x and y
- often x is a feature vector, and y is something we want to predict
- we don't know f, which gives the 'true' relationship between x and y

Model

- choose *model* $\hat{f}$: $\mathbf{R}^n \to \mathbf{R}$, a *guess* or *approximation* of f, based on some observed data

$$(x_1, y_1), \ldots, (x_N, y_N)$$

  called *observations*, *examples*, *samples*, or *measurements*

- model form:
$$\hat{f}(x) = \theta_1 f_1(x) + \cdots + \theta_p f_p(x)$$
- f_i : $\mathbf{R}^n \to \mathbf{R}$ are *basis functions* that we choose
- θ_i are *model parameters* that we choose
- $\hat{y}_i = \hat{f}(x_i)$ is (the model's) *prediction* of y_i
- we'd like $\hat{y}_i \approx y_i$, *i.e.*, model is consistent with observed data

Least-squares data fitting

- *prediction error* or *residual* is $r_i = \hat{y}_i - y_i$
- express y, $\hat{y}$, and r as N-vectors
- $\mathbf{rms}(r)$ is *RMS prediction error*
- *least-squares data fitting*: choose θ_i to minimize RMS prediction error
- define $N \times p$ matrix A, $A_{ij} = \hat{f}_j(x_i)$ so $\hat{y} = A\theta$
- least-squares data fitting: choose θ to minimize $\|r\|^2 = \|A\theta - y\|^2$
- $\hat{\theta} = (A^T A)^{-1} A^T y$ (if columns of A are independent)
- $\|A\hat{\theta} - y\|^2/N$ is *minimum mean-square (fitting) error*

Fitting a constant model

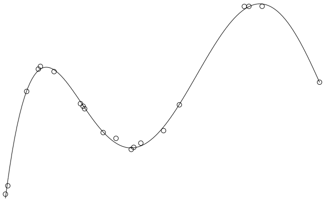
- simplest possible model: $p = 1$, $\hat{f}_1(x) = 1$, so model $\hat{f}(x) = \theta_1$ is a constant function
- $A = \mathbf{1}$, so

$$\hat{\theta}_1 = (\mathbf{1}^T\mathbf{1})^{-1}\mathbf{1}^T y = (1/N)\mathbf{1}^T y = \mathbf{avg}(y)$$

- the mean of y is the least-square fit by a constant
- MMSE is $\mathbf{std}(y)^2$; RMS error is $\mathbf{std}(y)$
- more sophisticated models are judged against the constant model

Fitting univariate functions

- when $n = 1$, we seek to approximate a function $f : \mathbf{R} \to \mathbf{R}$
- we can plot the data (x_i, y_i) and the model function $\hat{y} = \hat{f}(x)$



Straight-line fit

- $p = 2$, with $f_1(x) = 1$, $f_2(x) = x$
- model has form $\hat{f}(x) = \theta_1 + \theta_2 x$
- matrix A has form

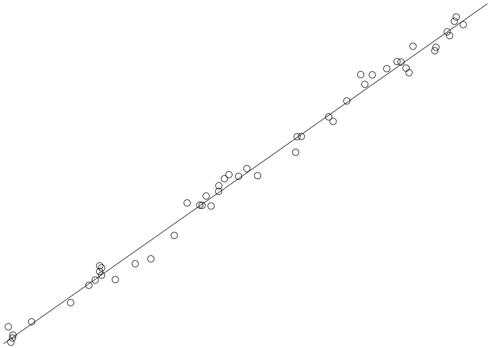
$$A = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_N \end{bmatrix}$$

- can work out $\hat{\theta}_1$ and $\hat{\theta}_2$ explicitly:

$$\hat{f}(u) = \mathbf{avg}(y) + \rho \frac{\mathbf{std}(y)}{\mathbf{std}(x)}(u - \mathbf{avg}(x))$$

(but QR works fine ...)

Example



Polynomial fit

- $f_i(x) = x^{i-1}, \quad i = 1, \ldots, p$
- model is a polynomial of degree less than p

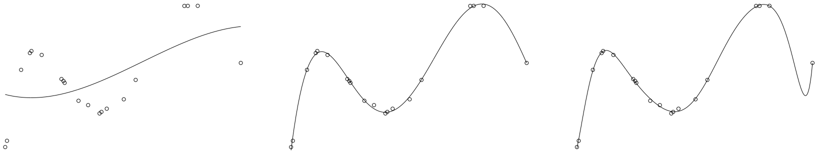
$$\hat{f}(x) = \theta_1 + \theta_2 x + \cdots + \theta_p x^{p-1}$$

- A is a Vandermonde matrix

$$A = \begin{bmatrix} 1 & x_1 & \cdots & x_1^{p-1} \\ 1 & x_2 & \cdots & x_2^{p-1} \\ \vdots & \vdots & & \vdots \\ 1 & x_N & \cdots & x_N^{p-1} \end{bmatrix}$$

Examples

- $N = 20$ data points
- fits of degree $p - 1 = 3$, 5, and 10



Outline

Least-squares problem

Solution of least-squares problem

Least-squares data fitting

Least-squares classification

Validation

Least-squares classification

- like regression, but response has only two values, *e.g.*, TRUE and FALSE
- common encoding of the two values: $y_i = +1$ and $y_i = -1$
  - email spam detection
  - transaction fraud detection
  - document classification (say, politics or not)

- *least-squares classification*:
  - fit model $\tilde{f}$ to encoded (± 1) y_i values
  - use model $\hat{f}(x) = \mathbf{sign}(\tilde{f}(x))$

  (size of $\tilde{f}(x)$ is related to our confidence in the prediction)

Confusion matrix

- the pair $(y, \hat{y})$ has only 4 values:
  - *True positive.* $y = +1$ and $\hat{y} = +1$
  - *True negative.* $y = -1$ and $\hat{y} = -1$
  - *False positive.* $y = -1$ and $\hat{y} = +1$
  - *False negative.* $y = +1$ and $\hat{y} = -1$
- numbers of each is organized into *confusion matrix*, *e.g.*

	$\hat{y} = +1$	$\hat{y} = -1$	total
$y = +1$	95	32	127
$y = -1$	19	1120	1139
total	114	1152	1266

- various error and prediction rates have names

Outline

Least-squares problem

Solution of least-squares problem

Least-squares data fitting

Least-squares classification

Validation

Generalization

basic idea:

- goal of model is *not* to predict outcome in the given data
- instead it is to *predict the outcome on new, unseen data*

- a model that makes reasonable predictions on new, unseen data has *generalization ability*
- a model that makes poor predictions on new, unseen data is said to suffer from *over-fit*

Validation

a simple and effective method to guess if a model will generalize

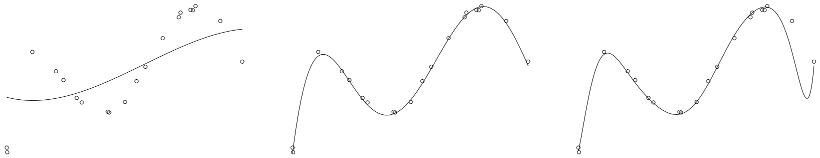
- split original data into a *training set* and a *test set*
- typical splits: 80%/20%, 90%/10%
- build ('train') model on training data set
- then *check the model's predictions on the test data set*
- (can also compare RMS prediction error on train and test data)
- if they are similar, we can *guess* the model will generalize

Validation

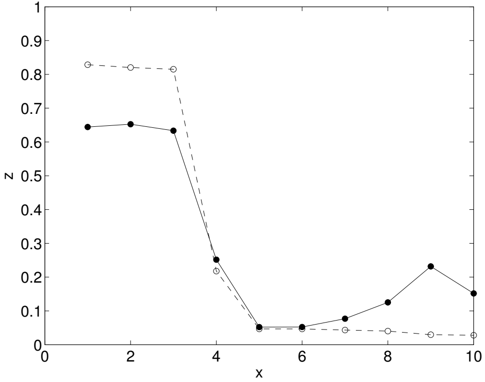
- can be used to choose among different candidate models, *e.g.*
  - polynomials of different degrees
  - regression models with different sets of regressors
- we'd use one with low, or lowest, test error

Example

- polynomial models from 20 training points above
- evaluated below with 20 new test points



Example



- suggests degree 5 or 6 polynomial would be good choice