Machine Learning

A. Supervised Learning
A.4. High-Dimensional Data

Lars Schmidt-Thieme

Information Systems and Machine Learning Lab (ISMLL)
Institute for Computer Science
University of Hildesheim, Germany

Outline

1. Variable Interactions and Polynomial Models

2. Parameter Variance

3. Variable Selection via Forward and Backward Search

4. Minimizing a Function via Coordinate Descent

5. L1 Regularization / The Lasso

Syllabus

Tue.	21.10.	(1)	0. Introduction
			A. Supervised Learning
Wed.	22.10.	(2)	A.1 Linear Regression
Tue.	28.10.	(3)	A.2 Linear Classification
Wed.	29.10.	(4)	A.3 Regularization
Tue.	4.11.	(5)	A.4 High-dimensional Data
Wed.	5.11.	(6)	A.5 Nearest-Neighbor Models
Tue.	11.11.	(7)	A.6 Support Vector Machines
Wed.	12.12.	(8)	A.7 Decision Trees
Tue.	18.11.	(9)	A.8 A First Look at Bayesian and Markov Networks
			B. Unsupervised Learning
Wed.	19.11.	(10)	B.1 Clustering
Tue.	25.11.	(11)	B.2 Dimensionality Reduction
Wed.	26.11.	(12)	B.3 Frequent Pattern Mining
			C. Reinforcement Learning
Tue.	2.12.	(13)	C.1 State Space Models
Wed.	3.12.	(14)	C.2 Markov Decision Processes

High-Dimensional Data

High-dimensional data occurs in different situations:

1. Data that comes naturally with many predictors.
   - e.g., text classification
     (# predictors = # words in the bag-of-words representation, e.g., 30.000)

2. Models that extract many predictor variables from objects to classify.
   - variable interactions
   - derived variables
   - complex objects such as graphs, texts, etc.
     - Situation 1 often really is a special case of this one.

3. Data with few examples compared to the number of variables ("small n, large p").
   - gene expression / microarray data

Outline

1. Variable Interactions and Polynomial Models

2. Parameter Variance

3. Variable Selection via Forward and Backward Search

4. Minimizing a Function via Coordinate Descent

5. L1 Regularization / The Lasso

Need for higher orders

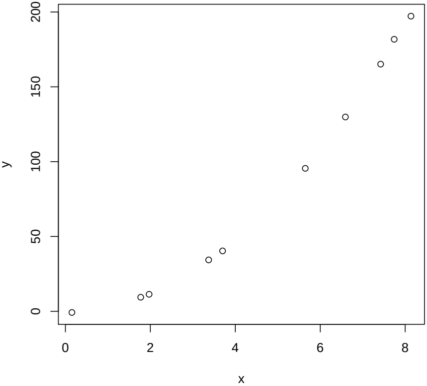
Assume a target variable does not depend linearly on a predictor variable, but say quadratic.

Example: way length vs. duration of a moving object with constant acceleration *a*.

$$s(t) = \frac{1}{2}at^2 + \epsilon$$

Can we catch such a dependency?

Can we catch it with a linear model?



Need for general transformations

To describe many phenomena, even more complex functions of the input variables are needed.

Example: the number of cells n vs. duration of growth t:

$$n = \beta e^{\alpha t} + \epsilon$$

n does not depend on t directly, but on $e^{\alpha t}$ (with a known α).

Need for variable interactions

In a linear model with two predictors

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \epsilon$$

Y depends on both, X_1 and X_2.

But changes in X_1 will affect Y the same way, regardless of X_2.

There are problems where X_2 mediates or influences the way X_1 affects Y, e.g. : the way length s of a moving object vs. its constant velocity v and duration t:

$$s = vt + \epsilon$$

Then an additional 1s duration will increase the way length not in a uniform way (regardless of the velocity), but a little for small velocities and a lot for large velocities.

v and t are said to interact: y does not depend only on each predictor separately, but also on their product.

Derived variables

All these cases can be handled by looking at **derived variables**, i.e., instead of

$$
\begin{aligned}
Y =& \beta_0 + \beta_1 X_1^2 + \epsilon \\
Y =& \beta_0 + \beta_1 e^{\alpha X_1} + \epsilon \\
Y =& \beta_0 + \beta_1 X_1 \cdot X_2 + \epsilon
\end{aligned}
$$

one looks at

$$
Y = \beta_0 + \beta_1 X_1' + \epsilon
$$

with

$$
\begin{aligned}
X_1' :=& X_1^2 \\
X_1' :=& e^{\alpha X_1} \\
X_1' :=& X_1 \cdot X_2
\end{aligned}
$$

Derived variables are computed before the fitting process and taken into account either additional to the original variables or instead of.

Polynomial Models

Polynomial models of degree *d* take into account systematically all interactions of *d* different variables (including powers up to degree *d*):

$$\hat{y}(x) := \hat{\theta}_0 + \sum_{m=1}^{M} \hat{\theta}_m x_m \qquad \text{degree 1}$$

Polynomial Models

Polynomial models of degree d take into account systematically all interactions of d different variables (including powers up to degree d):

$$\hat{y}(x) := \hat{\theta}_0 + \sum_{m=1}^{M} \hat{\theta}_m x_m \qquad \text{degree 1}$$

$$\hat{y}(x) := \hat{\theta}_0 + \sum_{m=1}^{M} \hat{\theta}_m x_m + \sum_{m=1}^{M} \sum_{l=m}^{M} \hat{\theta}_{m,l} x_m x_l \qquad \text{degree 2}$$

Polynomial Models

Polynomial models of degree d take into account systematically all interactions of d different variables (including powers up to degree d):

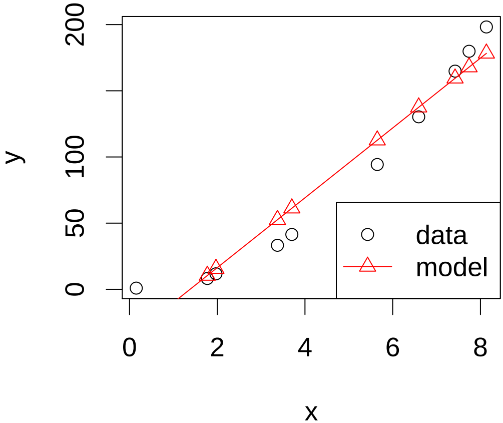
$$\hat{y}(x) := \hat{\theta}_0 + \sum_{m=1}^{M} \hat{\theta}_m x_m \qquad \text{degree } 1$$

$$\hat{y}(x) := \hat{\theta}_0 + \sum_{m=1}^{M} \hat{\theta}_m x_m + \sum_{m=1}^{M} \sum_{l=m}^{M} \hat{\theta}_{m,l} x_m x_l \qquad \text{degree } 2$$

$$\hat{y}(x) := \hat{\theta}_0 + \sum_{m=1}^{M} \hat{\theta}_m x_m + \sum_{m=1}^{M} \sum_{l=m}^{M} \hat{\theta}_{m,l} x_m x_l + \cdots$$

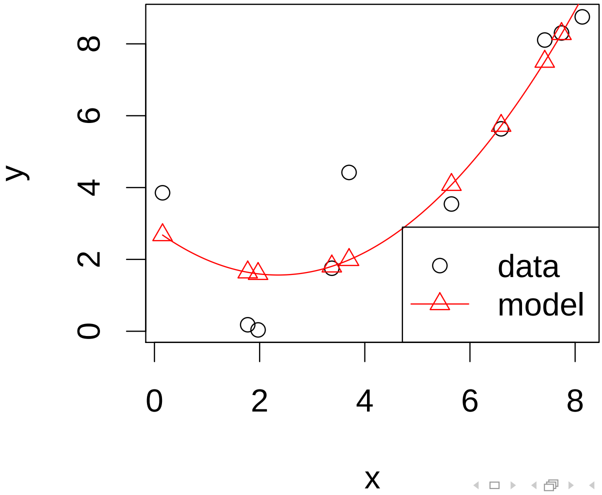
$$+ \sum_{m_1=1}^{M} \sum_{m_2=m_1}^{M} \cdots \sum_{m_d=m_{d-1}}^{M} \hat{\theta}_{m_1,m_2,\ldots,m_d} x_{m_1} x_{m_2} \cdots x_{m_d} \qquad \text{degree } d$$

High Polynomial Degress, High Model Complexity

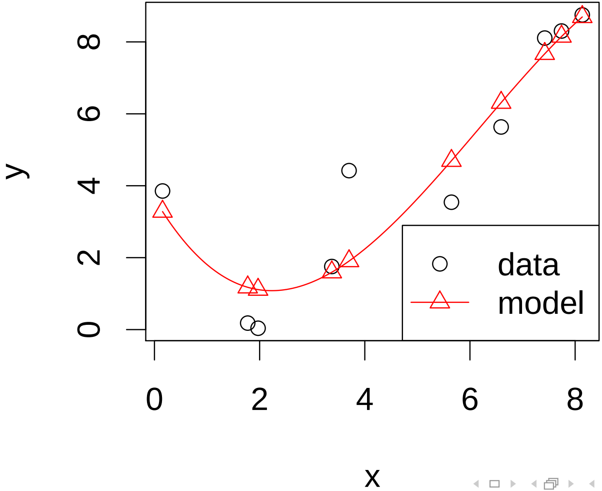


If a model does not well explain the data,
e.g., if the true model is quadratic, but we try to fit a linear model,
one says, the model **underfits**.

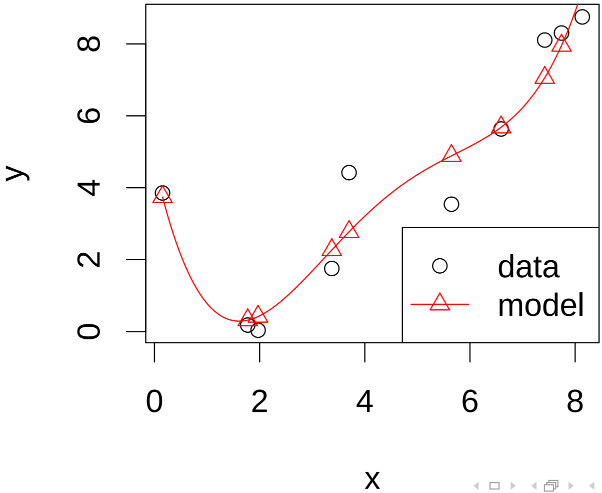
High Polynomial Degress, High Model Complexity



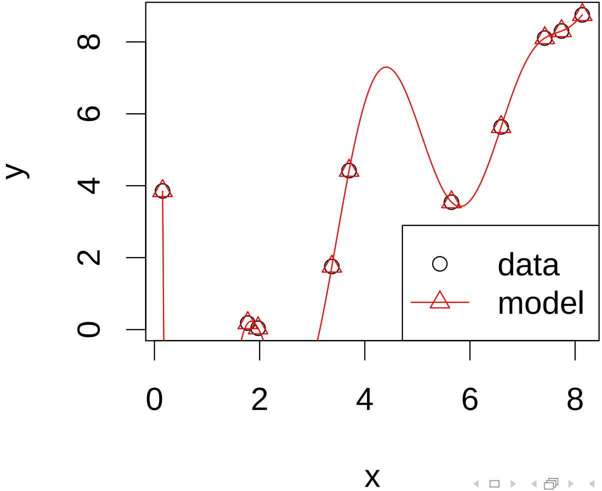
High Polynomial Degress, High Model Complexity



High Polynomial Degress, High Model Complexity



High Polynomial Degress, High Model Complexity



High Polynomial Degress, High Model Complexity

If to data

$$(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$$

consisting of n points we fit

$$\begin{aligned} X &= \beta_0 + \beta_1 X + \beta_2 X^2 + \cdots + \beta_{n-1} X^{n-1} \\ &= \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_{n-1} X_{n-1}, \quad X_i := X^i \end{aligned}$$

i.e., a polynomial with degree $n-1$, then this results in an interpolation of the data points
(if there are no repeated measurements, i.e., points with the same X.)

As the polynomial

$$\hat{y}(X) = \sum_{i=1}^{n} y_i \prod_{j \neq i} \frac{X - x_j}{x_i - x_j}$$

is of this type, and has minimal RSS $= 0$.

Variable Types and Coding

The most common variable types:

numerical / interval-scaled / quantitative

where differences and quotients etc. are meaningful,
usually with domain $\mathcal{X} := \mathbb{R}$,
e.g., temperature, size, weight.

nominal / discrete / categorical / qualitative / factor

where differences and quotients are not defined,
usually with a finite, enumerated domain,
e.g., $\mathcal{X} := \{\text{red}, \text{green}, \text{blue}\}$
or $\mathcal{X} := \{\text{a}, \text{b}, \text{c}, \ldots, \text{y}, \text{z}\}$.

ordinal / ordered categorical

where levels are ordered, but differences and quotients are not defined,
usually with a finite, enumerated domain,
e.g., $\mathcal{X} := \{\text{small}, \text{medium}, \text{large}\}$

Variable Types and Coding

Nominals are usually encoded as binary **dummy variables**:

$$\delta_{x_0}(X) := \begin{cases} 1, & \text{if } X = x_0, \\ 0, & \text{else} \end{cases}$$

one for each $x_0 \in \mathcal{X}$ (but one).

Example: $\mathcal{X} := \{\text{red}, \text{green}, \text{blue}\}$

Replace

one variable X with 3 levels: $\text{red}, \text{green}, \text{blue}$

by

two variables $\delta_{\text{red}}(X)$ and $\delta_{\text{green}}(X)$ with 2 levels each: 0, 1

X	$\delta_{\text{red}}(X)$	$\delta_{\text{green}}(X)$
red	1	0
green	0	1
blue	0	0
—	1	1

Outline

1. Variable Interactions and Polynomial Models

2. Parameter Variance

3. Variable Selection via Forward and Backward Search

4. Minimizing a Function via Coordinate Descent

5. L1 Regularization / The Lasso

The Normal Distribution (also Gaussian)

written as:

$$X \sim \mathcal{N}(\mu, \sigma^2)$$

with parameters:

μ mean,

σ standard deviance.

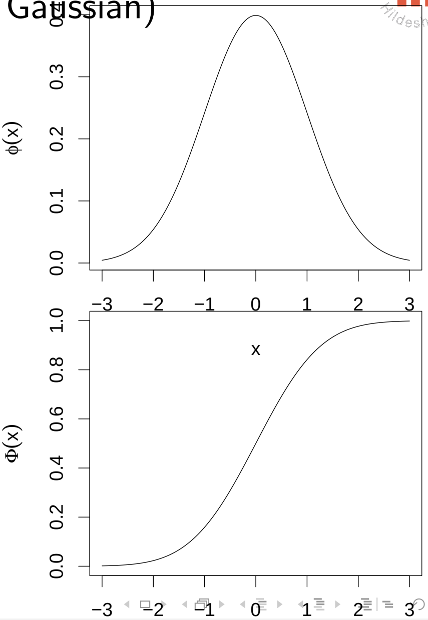
probability density function (pdf):

$$\phi(x) := \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

cumulative distribution function (cdf):

$$\Phi(x) := \int_{-\infty}^{x} \phi(t) dt$$

Φ^{-1} is called **quantile function**.



The t Distribution

written as:

$$X \sim t_p$$

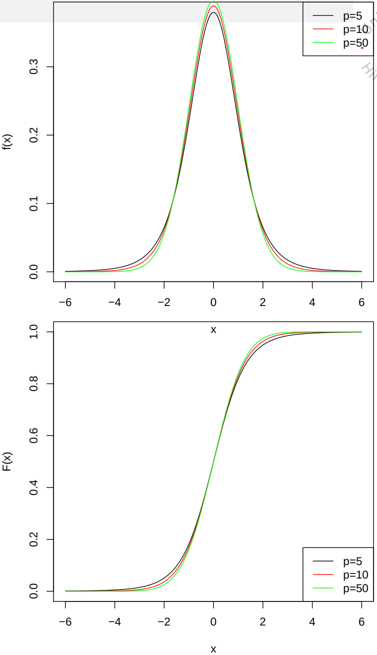
with parameter:

p degrees of freedom.

probability density function (pdf):

$$p(x) := \frac{\Gamma(\frac{p+1}{2})}{\sqrt{p\,\pi}\,\Gamma(\frac{p}{2})}(1+\frac{x^2}{p})^{-\frac{p+1}{2}}$$

$$t_p \overset{p\to\infty}{\longrightarrow} \mathcal{N}(0,1)$$



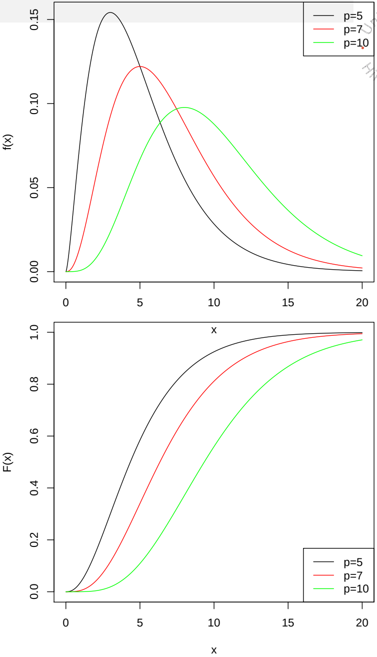
The χ^2 Distribution

written as:

$$X \sim \chi^2_p$$

with parameter:

p degrees of freedom.



probability density function (pdf):

$$p(x) := \frac{1}{\Gamma(p/2)2^{p/2}} x^{\frac{p}{2}-1} e^{-\frac{x}{2}}, \quad x \geq 0$$

If $X_1, \ldots, X_p \sim \mathcal{N}(0,1)$, then

$$Y := \sum_{i=1}^{p} X_i^2 \sim \chi^2_p$$

Parameter Variance

$\hat{\beta} = (X^T X)^{-1} X^T y$ is an unbiased estimator for β (i.e., $E(\hat{\beta}) = \beta$).
Its variance is

$$V(\hat{\beta}) = (X^T X)^{-1} \sigma^2$$

proof:

$$\hat{\beta} = (X^T X)^{-1} X^T y = (X^T X)^{-1} X^T (X\beta + \epsilon) = \beta + (X^T X)^{-1} X^T \epsilon$$

As $E(\epsilon) = 0$: $E(\hat{\beta}) = \beta$

$$\begin{aligned} V(\hat{\beta}) &= E((\hat{\beta} - E(\hat{\beta}))(\hat{\beta} - E(\hat{\beta}))^T) \\ &= E((X^T X)^{-1} X^T \epsilon \epsilon^T X (X^T X)^{-1}) \\ &= (X^T X)^{-1} \sigma^2 \end{aligned}$$

Parameter Variance

An unbiased estimator for σ^2 is

$$\hat{\sigma}^2 = \frac{1}{n-p}\sum_{i=1}^{n}\hat{\epsilon}_i^2 = \frac{1}{n-p}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

If $\epsilon \sim \mathcal{N}(0, \sigma^2)$, then

$$\hat{\beta} \sim \mathcal{N}(\beta, (X^TX)^{-1}\sigma^2)$$

Furthermore

$$(n-p)\hat{\sigma}^2 \sim \sigma^2\chi^2_{n-p}$$

Parameter Variance / Standardized coefficient

standardized coefficient ("z-score"):

$$z_i := \frac{\hat\beta_i}{\widehat{\text{se}}(\hat\beta_i)}, \quad \text{with } \widehat{\text{se}}^2(\hat\beta_i) \text{ the } i\text{-th diagonal element of } (X^T X)^{-1}\hat\sigma^2$$

z_i would be $z_i \sim \mathcal{N}(0,1)$ if σ is known (under $H_0 : \beta_i = 0$).
With estimated $\hat\sigma$ it is $z_i \sim t_{n-p}$.

The Wald test for $H_0 : \beta_i = 0$ with size α is:

$$\text{reject } H_0 \text{ if } |z_i| = |\frac{\hat\beta_i}{\widehat{\text{se}}(\hat\beta_i)}| > F^{-1}_{t_{n-p}}(1 - \frac{\alpha}{2})$$

i.e., its p-value is

$$p\text{-value}(H_0 : \beta_i = 0) = 2(1 - F_{t_{n-p}}(|z_i|)) = 2(1 - F_{t_{n-p}}(|\frac{\hat\beta_i}{\widehat{\text{se}}(\hat\beta_i)}|))$$

and small p-values such as 0.01 and 0.05 are good.

Confidence interval

The $1-\alpha$ confidence interval for β_i:

$$\beta_i \pm F^{-1}_{t_{n-p}}(1-\frac{\alpha}{2})\widehat{\text{se}}(\hat{\beta}_i)$$

For large n, $F_{t_{n-p}}$ converges to the standard normal cdf Φ.

As $\Phi^{-1}(1-\frac{0.05}{2}) \approx 1.95996 \approx 2$, the rule-of-thumb for a 5% confidence interval is

$$\beta_i \pm 2\widehat{\text{se}}(\hat{\beta}_i)$$

Example

We have already fitted

$$\begin{aligned}\hat{y} =& \hat{\beta}_0 + \hat{\beta}_1 x_1 + \hat{\beta}_2 x_2 \\ =& 5.583 + 0.779 x_1 - 1.699 x_2\end{aligned}$$

to the data:

x_1	x_2	y	$\hat{y}$	$\hat{\epsilon}^2 = (y - \hat{y})^2$
1	2	3	2.965	0.00122
2	3	2	2.045	0.00207
4	1	7	7.003	0.0000122
5	5	1	0.986	0.000196
RSS				0.00350

$$\hat{\sigma}^2 = \frac{1}{n-p} \sum_{i=1}^{n} \hat{\epsilon}_i^2 = \frac{1}{4-3} 0.00350 = 0.00350$$

$$(X^T X)^{-1} \hat{\sigma}^2 = \begin{pmatrix} 0.00520 & -0.00075 & -0.00076 \\ -0.00075 & 0.00043 & -0.00020 \\ -0.00076 & -0.00020 & 0.00049 \end{pmatrix}$$

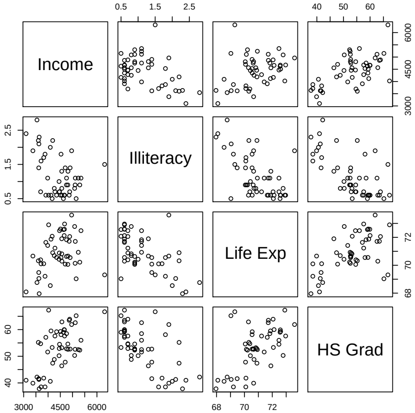
covariate	$\hat{\beta}_i$	$\widehat{\text{se}}(\hat{\beta}_i)$	z-score	p-value
(intercept)	5.583	0.0721	77.5	0.0082
X_1	0.779	0.0207	37.7	0.0169
X_2	−1.699	0.0221	−76.8	0.0083

Example 2

Example: sociographic data of the 50 US states in 1977.

state dataset:

- income (per capita, 1974),
- illiteracy (percent of population, 1970),
- life expectancy (in years, 1969–71),
- percent high-school graduates (1970).
- population (July 1, 1975)
- murder rate per 100,000 population (1976)
- mean number of days with minimum temperature below freezing (1931–1960) in capital or large city
- land area in square miles



Example 2

$$\begin{aligned}\text{Murder} = &\beta_0 + \beta_1\text{Population} + \beta_2\text{Income} + \beta_3\text{Illiteracy}\\ &+ \beta_4\text{LifeExp} + \beta_5\text{HSGrad} + \beta_6\text{Frost} + \beta_7\text{Area}\end{aligned}$$

$n = 50$ states, $p = 8$ parameters, $n - p = 42$ degrees of freedom.

Least squares estimators:

```
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.222e+02  1.789e+01   6.831 2.54e-08 ***
Population   1.880e-04  6.474e-05   2.905  0.00584 **
Income      -1.592e-04  5.725e-04  -0.278  0.78232
Illiteracy   1.373e+00  8.322e-01   1.650  0.10641
‘Life Exp‘  -1.655e+00  2.562e-01  -6.459 8.68e-08 ***
‘HS Grad‘    3.234e-02  5.725e-02   0.565  0.57519
Frost       -1.288e-02  7.392e-03  -1.743  0.08867 .
Area         5.967e-06  3.801e-06   1.570  0.12391
```

Outline

1. Variable Interactions and Polynomial Models

2. Parameter Variance

3. Variable Selection via Forward and Backward Search

4. Minimizing a Function via Coordinate Descent

5. L1 Regularization / The Lasso

The Variable Selection Problem

Given a data set $\mathcal{D}^{\text{train}} \subseteq \mathbb{R}^M \times \mathcal{Y}$,
an error measure err,
a model class with a learning algorithm $\mathcal{A}$,

find the subset $V \subseteq \{1, 2, \ldots, M\}$ of (relevant) variables s.t. the model

$$\hat{y} := \mathcal{A}(\pi_V(\mathcal{D}^{\text{train}}))$$

learned on this subset V is best, i.e., for new test data $\mathcal{D}^{\text{test}}$ it's test error

$$\text{err}(\hat{y}, \mathcal{D}^{\text{test}}),$$

is minimal.

Projection onto predictors V:

$$\pi_V(x, y) := (x_{i_1}, x_{i_2}, \ldots, x_{i_{\tilde{M}}}, y), \quad \text{for } V := \{i_1, i_2, \ldots, i_{\tilde{M}}\}$$

Greedy Search

- All 2^M subsets are too many to test (for larger M).
- Use a simple greedy search.
- **forward search:**
    - start with no variables.
    - test adding one more variable not yet in the model.
    - add the one leading to lowest validation error.
- **backward search:**
    - start with all variables.
    - test removing one more variable still in the model.
    - remove the one leading to lowest validation error.
- Does not guarantee to find the best variables subset.
  (But usually finds a useful one.)

Forward Search

```
procedure SELECTVARS-FORWARD($\mathcal{D}^{\text{train}\prime} \subseteq \mathbb{R}^M \times \mathcal{Y}, \text{err}, \mathcal{A}$)
    $(\mathcal{D}^{\text{train}}, \mathcal{D}^{\text{val}}) := \text{split}(\mathcal{D}^{\text{train}\prime})$
    $V := \emptyset$
    $e_{\text{allbest}} := \text{err}(\mathcal{A}(\pi_V(\mathcal{D}^{\text{train}})), \pi_V(\mathcal{D}^{\text{val}}))$
    $v_{\text{best}} := 1$
    while $v_{\text{best}} \neq 0$ do
        $v_{\text{best}} := 0$
        $e_{\text{best}} := e_{\text{allbest}}$
        for $v \in \{1, 2, \ldots, M\} \setminus V$ do
            $V' := V \cup \{v\}$
            $\hat{y} := \mathcal{A}(\pi_{V'}(\mathcal{D}^{\text{train}}))$
            $e := \text{err}(\hat{y}, \pi_{V'}(\mathcal{D}^{\text{val}}))$
            if $e < e_{\text{best}}$ then
                $v_{\text{best}} := v$
                $e_{\text{best}} := e$
        if $e_{\text{best}} < e_{\text{allbest}}$ then
            $V := V \cup \{v_{\text{best}}\}$
            $e_{\text{allbest}} := e_{\text{best}}$
    return $V$
```

Backward Search

```
procedure SELECTVARS-BACKWARD($\mathcal{D}^{\text{train}\prime} \subseteq \mathbb{R}^M \times \mathcal{Y}, \text{err}, \mathcal{A}$)
    $(\mathcal{D}^{\text{train}}, \mathcal{D}^{\text{val}}) := \text{split}(\mathcal{D}^{\text{train}\prime})$
    $V := \{1, 2, \ldots, M\}$
    $e_{\text{allbest}} := \text{err}(\mathcal{A}(\pi_V(\mathcal{D}^{\text{train}})), \pi_V(\mathcal{D}^{\text{val}}))$
    $v_{\text{best}} := 1$
    while $v_{\text{best}} \neq 0$ do
        $v_{\text{best}} := 0$
        $e_{\text{best}} := e_{\text{allbest}}$
        for $v \in V$ do
            $V' := V \setminus \{v\}$
            $\hat{y} := \mathcal{A}(\pi_{V'}(\mathcal{D}^{\text{train}}))$
            $e := \text{err}(\hat{y}, \pi_{V'}(\mathcal{D}^{\text{val}}))$
            if $e < e_{\text{best}}$ then
                $v_{\text{best}} := v$
                $e_{\text{best}} := e$
        if $e_{\text{best}} < e_{\text{allbest}}$ then
            $V := V \setminus \{v_{\text{best}}\}$
            $e_{\text{allbest}} := e_{\text{best}}$
    return $V$
```

Sequential Search with Variable Importance Heuristics

- Forward and backward search has to learn many models.
  - forward search: 1, 2, 3, . . .
  - backward search: M, M-1, M-2, . . .
- Further simplication: use a sequential search.
- Use a heuristics to assess **variable importance** once (without context)
  - e.g., the error of the single-variable model:

$$\text{imp}(m) := \text{err}(\mathcal{A}(\pi_{\{m\}}(\mathcal{D}^{\text{train}})), \mathcal{D}^{\text{val}})$$

- Add variables in order of increasing heuristics.
- Usually a full sequential sweep through all variables is done.
  - No difference between Forward and Backward Search.
- Faster, but even less reliable than forward/backward search.

Sequential Search

```
procedure SELECTVARS-SEQ($\mathcal{D}^{\text{train}\prime} \subseteq \mathbb{R}^M \times \mathcal{Y}$, err, $\mathcal{A}$, imp)
    $(\mathcal{D}^{\text{train}}, \mathcal{D}^{\text{val}}) := \text{split}(\mathcal{D}^{\text{train}\prime})$
    $\mathcal{V} := \text{sort-increasing}(\{1, 2, \ldots, M\}, \text{imp})$
    $V := \emptyset$
    $e_{\text{best}} := \text{err}(\mathcal{A}(\pi_V(\mathcal{D}^{\text{train}})), \pi_V(\mathcal{D}^{\text{val}}))$
    $m_{\text{best}} := 1$
    for $m = 1, \ldots, M$ do
        $v := \mathcal{V}_m$
        $V := V \cup \{v\}$
        $\hat{y} := \mathcal{A}(\pi_V(\mathcal{D}^{\text{train}}))$
        $e := \text{err}(\hat{y}, \pi_V(\mathcal{D}^{\text{val}}))$
        if $e < e_{\text{best}}$ then
            $m_{\text{best}} := m$
            $e_{\text{best}} := e$
    $V := \{1, 2, \ldots, m_{\text{best}}\}$
    return $V$
```

Outline

1. Variable Interactions and Polynomial Models

2. Parameter Variance

3. Variable Selection via Forward and Backward Search

4. Minimizing a Function via Coordinate Descent

5. L1 Regularization / The Lasso

Minimizing a Function via Coordinate Descent (CD)

Given a function $f : \mathbb{R}^N \rightarrow \mathbb{R}$, find x with minimal $f(x)$.

- Use the coordinate axes as descent direction
  - first x_1-axis, then x_2-axis, etc. (cyclic)
  - **one-dimensional subproblems**:

$$g_n(x) := \arg\min_{x_n \in \mathbb{R}} f(x_n; x_{-n}) := \arg\min_{x' \in \mathbb{R}} f(x_1, x_2, \ldots, x_{n-1}, x', x_{n+1}, \ldots, x_N)$$

- Coordinate Descent can be fast if solving the one-dimensional subproblems can be done analytically.
  - For smooth f, one needs to solve

$$\frac{\partial f(x_n; x_{-n})}{\partial x_n} \overset{!}{=} 0$$

  - Then also no step length is required !

Note: $x_{-n} := (x_1, \ldots, x_2, \ldots, x_{n-1}, x_{n+1}, \ldots, x_N)$ is the vector without element n for a vector $x \in \mathbb{R}^N$.

Coordinate Descent

```
procedure MINIMIZE-CD($f : \mathbb{R}^N \to \mathbb{R}, g, x^{(0)} \in \mathbb{R}^N, i_{\max} \in \mathbb{N}, \epsilon \in \mathbb{R}^+$)
    for $i := 1, \ldots, i_{\max}$ do
        $x^{(i)} := x^{(i-1)}$
        for $n := 1, \ldots, N$ do
            $x_n^{(i)} := g_n(x_{-n}^{(i)})$
        if $f(x^{(i-1)}) - f(x^{(i)}) < \epsilon$ then
            return $x^{(i)}$
    error "not converged in $i_{\max}$ iterations"
```

g solvers g_n for the n-th one-dimensional subproblem

$$g_n(x_1, x_2, \ldots, x_{n-1}, x_{n+1}, \ldots, x_N) := \arg\min_{x' \in \mathbb{R}} f(x_1, \ldots, x_{n-1}, x', x_{n+1}, \ldots, x_N)$$

Example: Simple Quadratic Function

Minimize

$$f(x_1, x_2) := x_1^2 + x_2^2 + x_1 x_2$$

One dimensional problem for x_1:

$$\begin{aligned} f(x_1; x_2) =& x_1^2 + x_2^2 + x_1 x_2 \\ \frac{\partial f}{\partial x_1}(x_1; x_2) =& 2x_1 + x_2 \overset{!}{=} 0 \\ \rightsquigarrow x_1 =& -\frac{1}{2} x_2 \\ \text{i.e., } g_1(x_2) :=& -\frac{1}{2} x_2 \end{aligned}$$

and analogous for x_2:

$$g_2(x_1) := -\frac{1}{2} x_1$$

Example: Simple Quadratic Function

Minimize

$$f(x_1, x_2) := x_1^2 + x_2^2 + x_1 x_2, \quad x^{(0)} := (1, 1)$$

$$g_1(x_2) := -\frac{1}{2}x_2, \quad g_2(x_1) := -\frac{1}{2}x_1$$

i	$x^{(i)}$ before	n	$g_n(x^{(i)})$	$x^{(i-1)}$ after
1	$(1, 1)$	1	$-1/2$	$(-1/2, 1)$
	$(-1/2, 1)$	2	$1/4$	$(-1/2, 1/4)$
2	$(-1/2, 1/4)$	1	$-1/8$	$(-1/8, 1/4)$
	$(-1/8, 1/4)$	2	$1/16$	$(-1/8, 1/16)$

$\vdots$

Note: Minimize $f(x_1, x_2) := x_1^2 + x_2^2$ via CD yourself. What is different? Why?

Learn Linear Regression via CD

Minimize

$$\begin{aligned} f(\hat\beta) &:= ||y - X\hat\beta||^2 \propto \hat\beta^T X^T X\hat\beta - 2y^T X\hat\beta \\ f(\hat\beta_m; \hat\beta_{-m}) &= x_m^T x_m \hat\beta_m^2 + 2\hat\beta_{-m}^T X_{-m}^T x_m \hat\beta_m + \hat\beta_{-m}^T X_{-m}^T X_{-m}\hat\beta_{-m} \\ &\quad - 2y^T x_m \hat\beta_m - 2y^T X_{-m}\hat\beta_{-m} \\ &\propto x_m^T x_m \hat\beta_m^2 - 2(y - X_{-m}\hat\beta_{-m})^T x_m \hat\beta_m \end{aligned}$$

$$\frac{\partial f(\hat\beta_m; \hat\beta_{-m})}{\partial \hat\beta_m} \overset{!}{=} 0 \rightsquigarrow \hat\beta_m = \frac{(y - X_{-m}\hat\beta_{-m})^T x_m}{x_m^T x_m}$$

Note: $x_m := X_{.,m}$ denotes the m-th column of X,
X_{-m} denotes the matrix X without column m.

Learn Linear Regression via CD

```
procedure LEARN-LINREG-CD($\mathcal{D}^{\text{train}} := \{(x_1, y_1), \ldots, (x_N, y_N)\}, i_{\max} \in \mathbb{N}, \epsilon \in \mathbb{R}^+$)
    $X := (x_1, x_2, \ldots, x_N)^T$
    $y := (y_1, y_2, \ldots, y_N)^T$
    $\hat{\beta}_0 := (0, \ldots, 0)$
    $\hat{\beta} :=$ MINIMIZE-CD( $f(\hat{\beta}) := (y - X\hat{\beta})^T (y - X\hat{\beta})$,
                             $g(\hat{\beta}_m; \hat{\beta}_{-m}) := \frac{(y - X_{-m}\hat{\beta}_{-m})^T x_m}{x_m^T x_m}$
                             $\hat{\beta}_0, \alpha, i_{\max}, \epsilon$)
    return $\hat{\beta}$
```

Note: $x_m := X_{.,m}$ denotes the m-th column of X,
X_{-m} denotes the matrix X without column m.

Outline

1. Variable Interactions and Polynomial Models

2. Parameter Variance

3. Variable Selection via Forward and Backward Search

4. Minimizing a Function via Coordinate Descent

5. L1 Regularization / The Lasso

L1 Regularization

Let X the predictor matrix and y the target vector,
$\hat{\theta}$ the model parameters,
$\hat{y}$ the model predictions and
ℓ the loss/error.

L2 regularization:

$$f(\hat{\theta}) := \ell(y, \hat{y}(\hat{\theta}, X)) + \lambda ||\hat{\theta}||_2^2 = \ldots + \lambda \sum_{p=1}^{P} \theta_p^2$$

L1 regularization:

$$f(\hat{\theta}) := \ell(y, \hat{y}(\hat{\theta}, X)) + \lambda ||\hat{\theta}||_1 = \ldots + \lambda \sum_{p=1}^{P} |\hat{\theta}_p|$$

Why L1 Regularization?

$$\begin{aligned} \text{min. } & f(\hat{\theta}) := \ell(y, \hat{y}(\hat{\theta}, X)) + \lambda ||\hat{\theta}||_1 \\ & \hat{\theta} \in \mathbb{R}^P \end{aligned}$$

is equivalent to

$$\begin{aligned} \text{min. } & f(\hat{\theta}) := \ell(y, \hat{y}(\hat{\theta}, X)) \\ & ||\hat{\theta}||_1 \leq B \\ & \hat{\theta} \in \mathbb{R}^P \end{aligned}$$

with

$$B := ||\hat{\theta}^*||_1$$

Note: $\hat{\theta}^*$ denotes the optimal parameters. Thus this equivalence provides insight, but cannot (yet) be used to solve the problem.

Why L1 Regularization?

$$\begin{aligned} \text{min.}\ & f(\hat{\theta}) := \ell(y, \hat{y}(\hat{\theta}, X)) + \lambda \|\hat{\theta}\|_1 \\ & \hat{\theta} \in \mathbb{R}^P \end{aligned}$$

is equivalent to

$$\begin{aligned} \text{min.}\ & f(\hat{\theta}) := \ell(y, \hat{y}(\hat{\theta}, X)) \\ & \|\hat{\theta}\|_1 \leq B \\ & \hat{\theta} \in \mathbb{R}^P \end{aligned}$$

with

$$B := \|\hat{\theta}^*\|_1$$

$$\begin{aligned} \text{min.}\ & f(\hat{\theta}) := \ell(y, \hat{y}(\hat{\theta}, X)) + \lambda \|\hat{\theta}\|_2^2 \\ & \hat{\theta} \in \mathbb{R}^P \end{aligned}$$

is equivalent to

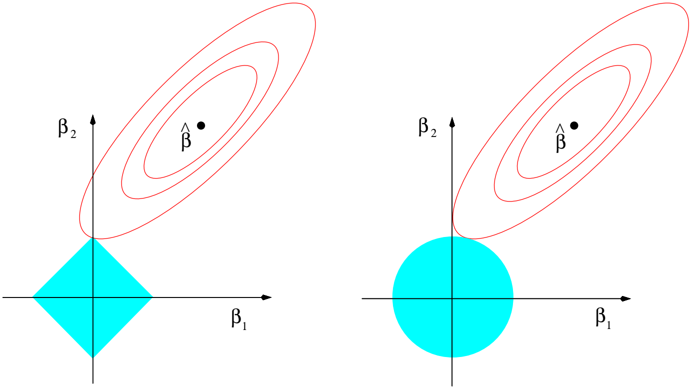
$$\begin{aligned} \text{min.}\ & f(\hat{\theta}) := \ell(y, \hat{y}(\hat{\theta}, X)) \\ & \|\hat{\theta}\|_2^2 \leq B \\ & \hat{\theta} \in \mathbb{R}^P \end{aligned}$$

with

$$B := \|\hat{\theta}^*\|_2^2$$

Note: $\hat{\theta}^*$ denotes the optimal parameters. Thus this equivalence provides insight, but cannot (yet) be used to solve the problem.

Why L1 Regularization?



source: [HTFF05, p. 90]

Regularized Linear Regression

Let X the predictor matrix and y the target vector,
$\hat{\beta}$ the linear regression model parameters,
$\hat{y} := X\hat{\beta}$ the linear regression model predictions and
$\ell(y, \hat{y}) := ||y - \hat{y}||_2^2$ the RSS loss/error.

L2 Regularized Linear Regression (**Ridge Regression**):

$$\begin{aligned} f(\hat{\beta}) &:= \ell(y, \hat{y}(\hat{\beta}, X)) + \lambda||\hat{\beta}||_2^2 \\ &\propto \hat{\beta}^T X^T X\hat{\beta} - 2y^T X\hat{\beta} + \lambda\hat{\beta}^T\hat{\beta} \\ &= \hat{\beta}^T(X^T + \lambda^{\frac{1}{2}} I)(X + \lambda^{\frac{1}{2}} I)\hat{\beta} - 2y^T X\hat{\beta} \end{aligned}$$

- L2 regularized problem has same structure as unregularized one.
- All learning algorithms work seamlessly.

Regularized Linear Regression

Let X the predictor matrix and y the target vector,
$\hat{\beta}$ the linear regression model parameters,
$\hat{y} := X\hat{\beta}$ the linear regression model predictions and
$\ell(y, \hat{y}) := ||y - \hat{y}||_2^2$ the RSS loss/error.

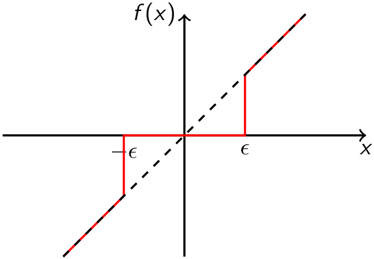
L1 regularized Linear Regression (**Lasso**):

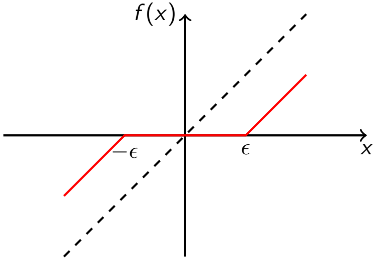
$$f(\beta) := \ell(y, \hat{y}) + \lambda||\beta||_1$$

$$\propto \hat{\beta}^T X^T X \hat{\beta} - 2y^T X \hat{\beta} + \lambda \sum_{m=1}^{M} |\beta_m|$$

- L1 regularized problem has new terms $|\beta_m|$.
  - Esp. non-differentiable at 0.
- All learning algorithms seen so far do not work.
  - Solving SLE is not applicable.
  - Gradient Descent does not work.

Hard & Soft Thresholding





$$\text{hard}(x,\epsilon) := \begin{cases} x, & \text{if } |x| > \epsilon \\ 0, & \text{else} \end{cases}$$

$$\text{soft}(x,\epsilon) := \begin{cases} x-\epsilon, & \text{if } x > \epsilon \\ 0, & \text{if } |x| \leq \epsilon \\ x+\epsilon, & \text{if } x < -\epsilon \end{cases}$$

Coordinate Gradient for L1 Regularized Linear Regression

$$f(\hat{\beta}) := \hat{\beta}^T X^T X \hat{\beta} - 2y^T X \hat{\beta} + \lambda \sum_{m=1}^{M} |\beta_m|$$

$$f(\hat{\beta}_m; \hat{\beta}_{-m}) \propto x_m^T x_m \hat{\beta}_m^2 - 2(y - X_{-m}\hat{\beta}_{-m})^T x_m \hat{\beta}_m + \lambda |\beta_m|$$

$$\frac{\partial f(\hat{\beta}_m; \hat{\beta}_{-m})}{\partial \hat{\beta}_m} \overset{!}{=} 0 \rightsquigarrow \hat{\beta}_m = \frac{(y - X_{-m}\hat{\beta}_{-m})^T x_m - \frac{1}{2}\lambda}{x_m^T x_m}, \quad \hat{\beta}_m > 0$$

$$\hat{\beta}_m = \frac{(y - X_{-m}\hat{\beta}_{-m})^T x_m + \frac{1}{2}\lambda}{x_m^T x_m}, \quad \hat{\beta}_m < 0$$

$$\rightsquigarrow \hat{\beta}_m = \text{soft}\left(\frac{(y - X_{-m}\hat{\beta}_{-m})^T x_m}{x_m^T x_m}, \frac{\frac{1}{2}\lambda}{x_m^T x_m}\right)$$

Note: LASSO = Least Absolute Selection and Shrinkage Operator.

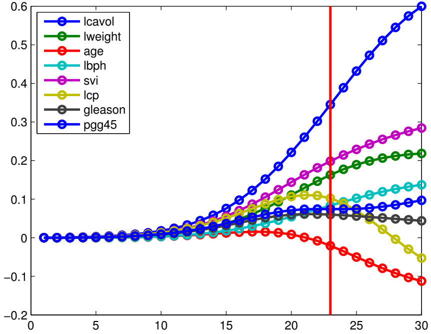
Learn L1-regularized Linear Regression via CD (Shooting Algorithm)

```
procedure LEARN-LINREG-L1REG-CD($\mathcal{D}^{\text{train}} := \{(x_1, y_1), \ldots, (x_N, y_N)\}, \lambda \in \mathbb{R}^+, i_{\max} \in \mathbb{N}, \epsilon \in \mathbb{R}^+$)
    $X := (x_1, x_2, \ldots, x_N)^T$
    $y := (y_1, y_2, \ldots, y_N)^T$
    $\hat{\beta}_0 := (0, \ldots, 0)$
    $\hat{\beta} :=$ MINIMIZE-CD( $f(\hat{\beta}) := (y - X\hat{\beta})^T(y - X\hat{\beta}) + \lambda||\beta||_1$,
                                  $g(\hat{\beta}_m; \hat{\beta}_{-m}) := \text{soft}(\frac{(y-X_{-m}\hat{\beta}_{-m})^T x_m}{x_m^T x_m}, \frac{\frac{1}{2}\lambda}{x_m^T x_m})$,
                                  $\hat{\beta}_0, \alpha, i_{\max}, \epsilon$)
    return $\hat{\beta}$
```

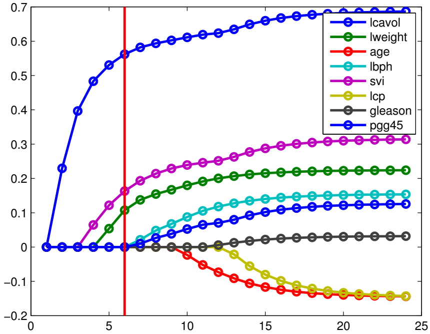
Note: $x_m := X_{.,m}$ denotes the m-th column of X,
X_{-m} denotes the matrix X without column m.

Regularization Paths

L2 regularization



L1 regularization



x-axis: bound B on parameter size.
y-axis: parameter $\hat{\theta}$.

source: [Mur12, p. 437]

Summary

- **High-dimensional data** poses problems as many parameters have to be estimated from comparable few instances.

- Non-linear effects can be captured by **derived predictor variables**.
  - e.g., in **polynomial models**.
  - making even originally low-dimensional data high-dimensional.

- Relevant variables can be searched explicitly through a greedy **forward search** and **backward search**.

- To minimize a function, **coordinate descent** cyclicly chooses the coordinate axes as descent direction.
  - efficient, if the **one-dimensional subproblems** can be solved analytically.
  - does need no step length.

- Variable selection also can be accomplished by **L1 regularization**.
  - **L1 regularized linear regression (LASSO)** can be learned by coordinate descent (**shooting algorithm**).

Further Readings

- [JWHT13, chapter 6], [Mur12, chapter 13], [HTFF05, chapter 3.3–8].

References

Trevor Hastie, Robert Tibshirani, Jerome Friedman, and James Franklin.
The elements of statistical learning: data mining, inference and prediction, volume 27.
2005.

Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani.
An introduction to statistical learning.
Springer, 2013.

Kevin P. Murphy.
Machine learning: a probabilistic perspective.
The MIT Press, 2012.