データサイエンス
第6回

~回帰分析（２）~

情報理工学系研究科
創造情報学専攻
中山 英樹

本日の内容

- 先週の復習（+α）
  - 線形回帰分析
  - 正則化
  - パラメータチューニング
- 一般化線形モデル
  - 最尤推定
  - ニューラルネットワークとの関係
  - 勾配降下法
- 第一回レポート課題発表
- 来週（11月15日）は休講

復習

- 機械学習の言葉でいうと…
  - データから何かを発見したい → 教師なし学習

説明変数	手法
量的データ(比尺度)	主成分分析、因子分析、LPP
量的データ(間隔尺度)	クラスター分析、多次元尺度構成法、数量化Ⅳ類
質的データ	数量化Ⅲ類、対応分析

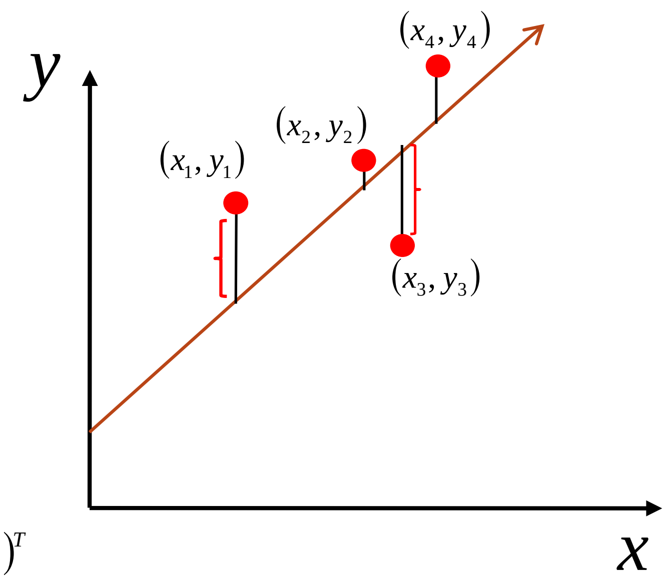
  - データを使って何かを予測したい → 教師あり学習

目的変数	説明変数	手法
量的データ	量的データ	回帰分析
	質的データ	数量化Ⅰ類
質的データ	量的データ	判別分析、ロジスティック回帰
	質的データ	数量化Ⅱ類

線形回帰

- 最小二乗法によるモデルフィッティング

$$E(\mathbf{a}, b) = \sum_{i=1}^{N} (y_i - \hat{y}_i)^2 = \sum_{i=1}^{N} \left(y_i - \left(\mathbf{a}^T \mathbf{x}_i + b\right)\right)^2$$



$\mathbf{a}, b$ でそれぞれ偏微分して0とおくと

$$\frac{\partial E}{\partial \mathbf{a}} = -2 \sum_{i=1}^{N} \mathbf{x}_i \left(y_i - \mathbf{a}^T \mathbf{x}_i\right) = 0 \;\Rightarrow\; \hat{\mathbf{a}} = \left(\sum_{i=1}^{N} \mathbf{x}_i \mathbf{x}_i^T\right)^{-1} \left(\sum_{i=1}^{N} \mathbf{x}_i y_i\right)$$

$$= \underbrace{\left(X X^T\right)}_{\text{自己相関行列}}^{-1} X\, \mathbf{y}$$

ただし $X = (\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N)$, $\mathbf{y} = (y_1, y_2, \cdots, y_N)^T$

$$\frac{\partial E}{\partial b} = -2 \sum_{i=1}^{N} \left(y_i - \left(\mathbf{a}^T \mathbf{x}_i + b\right)\right) = 0 \;\Rightarrow\; \hat{b} = \frac{1}{N} \sum_{i=1}^{N} \left(y_i - \hat{\mathbf{a}}^T \mathbf{x}_i\right)$$

定数項bを用意する代わりに、特徴ベクトルx
に常に1の要素を付与してもよい

リッジ回帰

- 自己相関行列に単位行列を微小な重みをつけて加算し、安定的に逆行列を計算（正則化）

$$\hat{\mathbf{a}} = \left(XX^T + \gamma I\right)^{-1} X\,\mathbf{y}$$

- いろんな解釈が可能
  - 過学習を抑える効果（人工的なノイズを加えている）
  - 最小化する目的関数として、誤差項に二乗ノルムを加えている = L2正則化

$$E'(\mathbf{a}, b) = \sum_{i=1}^{N}\left(y_i - \left(\mathbf{a}^T\mathbf{x}_i + b\right)\right)^2 + \gamma\left\|\mathbf{a}\right\|^2$$

係数（の絶対値）をできるだけ小さくするような作用

ことば

- パラメータ
  - 学習アルゴリズム自身によって最適化される変数
  - 例）線形回帰の重み係数ベクトル

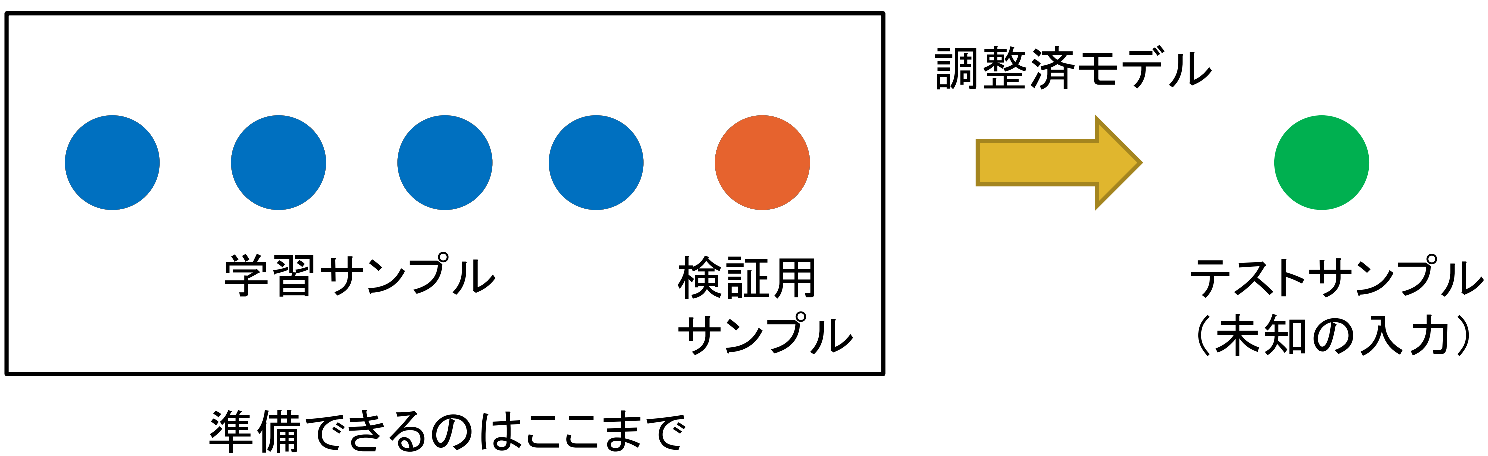
$$y = \mathbf{a}^T \mathbf{x} \qquad \hat{\mathbf{a}} = \left(X X^T \right)^{-1} X \, \mathbf{y}$$

- ハイパーパラメータ ⇦ こっちを決めるのが大変！（何らかの仮定を置かない限り定まらない）
  - 学習アルゴリズムに天下り的に与える変数
  - 例）正則化の大きさ、多項式回帰の次数

$$\hat{\mathbf{a}} = \left(X X^T + \gamma I \right)^{-1} X \, \mathbf{y}$$

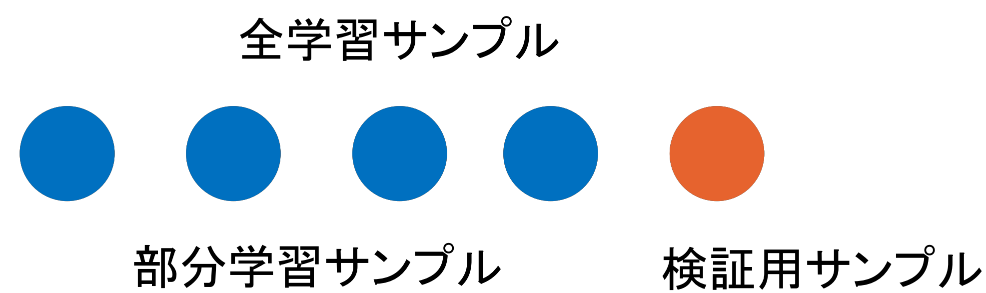
学習アルゴリズムの検証と評価

- 一般的な手順
  1．一定数の学習サンプル、検証用サンプルを用意する。
  2．学習サンプルで、候補となるモデル（ハイパーパラメータごと）を学習し、検証用データの予測誤差を調べる。
  3．検証用サンプルでの予測誤差が最も小さかったモデルを採用する。
  4．**<u>別に用意したテストサンプル</u>**での予測誤差を評価し、最終的な評価指標とする。



テストデータがない場合

- 交差検定法（クロスバリデーション）
  1. もともとの学習サンプルを分割し、新しい学習サンプルと検証用サンプルに分割する（例えば4:1などに）
  2. 新しい学習サンプルで、候補となるモデル（この場合パラメータごとに）を学習し、検証用データの予測誤差を調べる
  3. 学習サンプルと検証用サンプルを順番に入れ替え、２を繰り返す
  4. 全試行の平均予測誤差が最も小さかったものを採用する



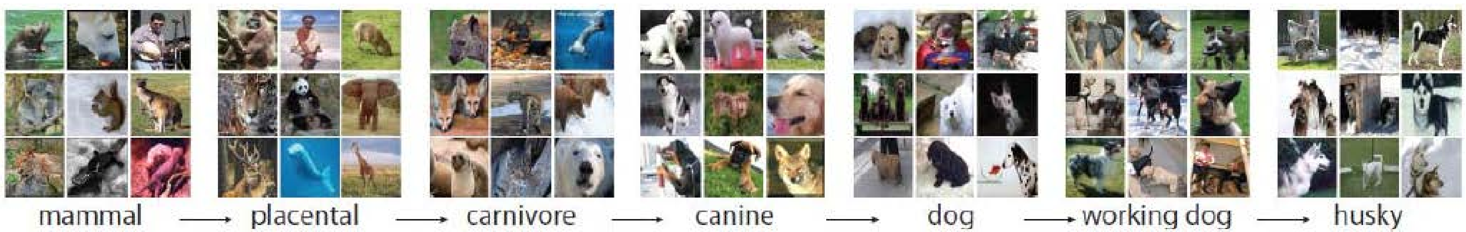
Baidu事件 @ILSVRC

- BaiduのHPCチームが、画像認識コンペティションであるImageNet large-scale visual recognition challenge で不正行為を行い、一年間の出禁処分を受ける
- 評価手順に関しての不正行為
- チームの責任者は解雇

ImageNet Large-scale Visual Recognition Challenge (ILSVRC)

Russakovsky et al., “ImageNet Large Scale Visual Recognition Challenge”, 2014.

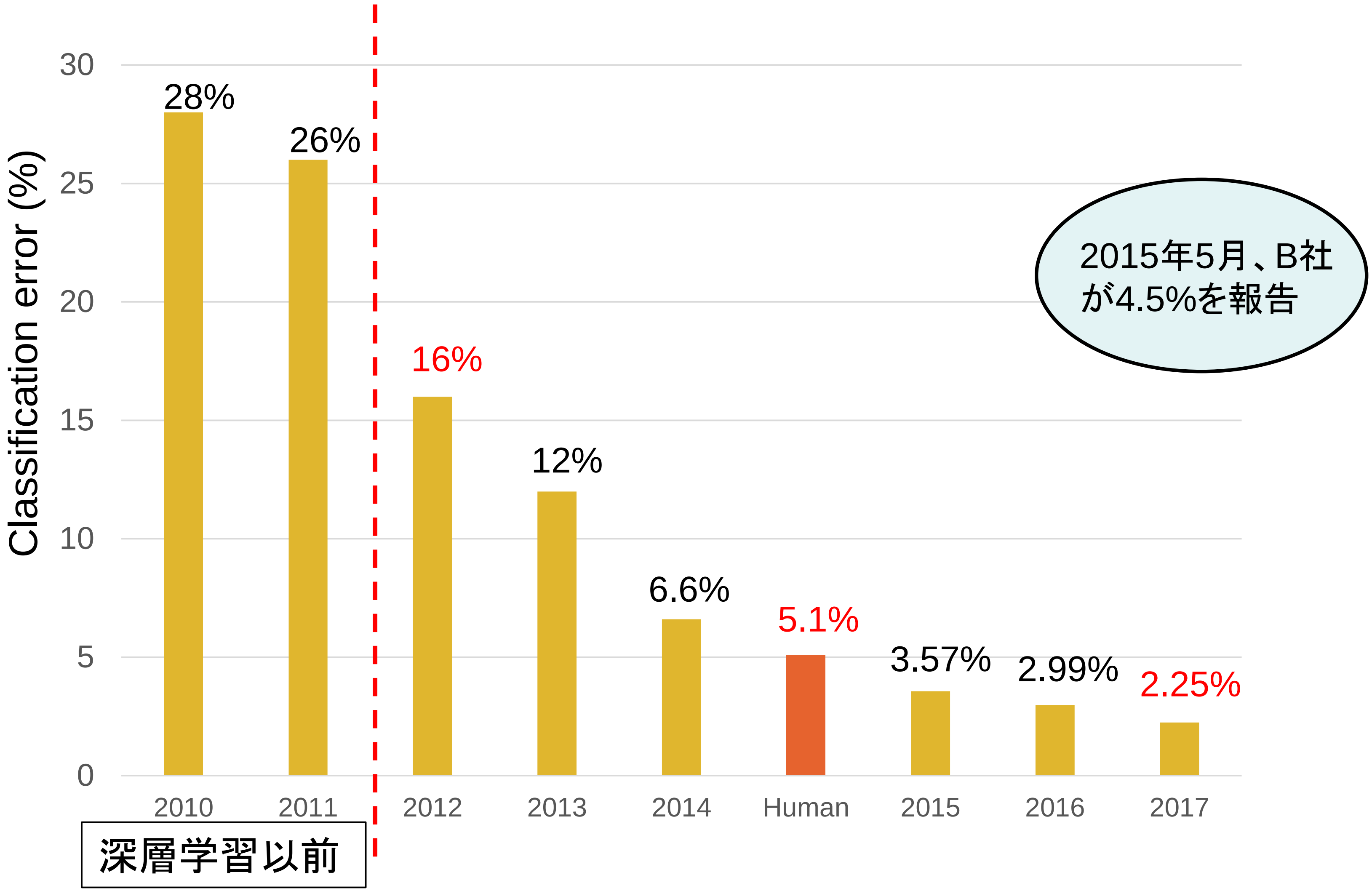
- ImageNetのデータの一部を用いたフラッグシップコンペティション (2010年から2017年まで開催)
  - ImageNet [Deng et al., 2009]
    - クラウドソーシングにより構築中の大規模画像データセット
    - **1400万枚、2万2千カテゴリ**（WordNetに従って構築）



- コンペでのタスク
  - 1000クラスの物体カテゴリ分類
    - 学習データ120万枚、検証用データ5万枚、テストデータ10万枚
  - 200クラスの物体検出
    - 学習データ45万枚、検証用データ2万枚、テストデータ4万枚

ILSVRCにおけるブレークスルー

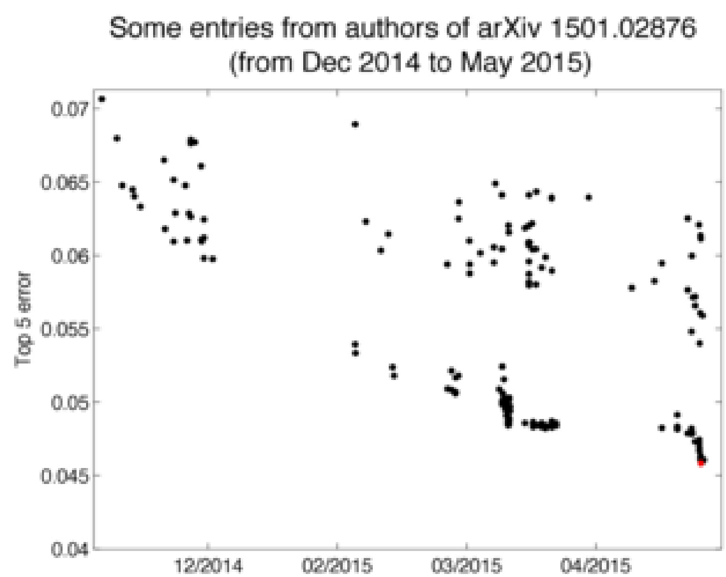
- エラー率が 16% (2012) → 2.3% (2017)



何がまずかったか？

- 評価手順のルール
  - 主催者が用意しているサーバにテストデータの識別結果を送信し、採点してもらう（答えは公開されない）
  - サーバに送信していいのは週2度まで（ユーザアカウントごとに管理される）
  - 過度にテストデータに適応することを防ぐ

- B社の罪
  - 複数のアカウントを作成し、合計200回結果提出
  - 一番良かったスコアを論文化
  - 要するに、答をクローリングしたようなもの



教訓

- この業界ならではの新しいタイプの研究不正
  - 大なり小なり似たようなことをみんなやっているという噂も…
  - コンペではなく、公開されているベンチマークだとさらにルールが曖昧
- 悪意なくルール違反してしまうことも
  - 機械学習の原理原則と評価に関する十分な理解が必要
  - スコアを追いかけるだけでなく、そもそも何のためにやっているかを考える

特徴（説明変数）の選択

- 一般的な方法
  - 基本的には、変数選択とモデル構築の段階が別れている

- １．ひとつずつ、目的変数への寄与を見る
  - 相関、エントロピー利得など
  - 寄与が低いものを捨てる

- ２．特徴の部分集合でモデルを作り評価
  - 汎化誤差（検証誤差）、AICなど
  - 一つずつ特徴を追加 or 一つずつ外していく

モデル選択の規準（×基準）

- 赤池情報量規準 (AIC) （尤度推定）
  - 以下を最小とするモデルを選択する

$$-2\underbrace{\ln(L)}_{\text{対数尤度}}+2\underbrace{M}_{\text{ハイパーパラメータ数}}$$

- BIC (Bayesian information criterion)（ベイズ推定）

$$-2\ln(L)+M\ln(N)$$

- MDL (minimal description length)（情報理論）

$$-\ln(L)+\frac{M\ln(N)}{2}$$

- 訓練データだけから、「そこそこいいモデル」が得られる
  - 実際には、過度に単純なものが得られる場合が多いらしい？
  - 変数選択などに使える

スパース線形回帰

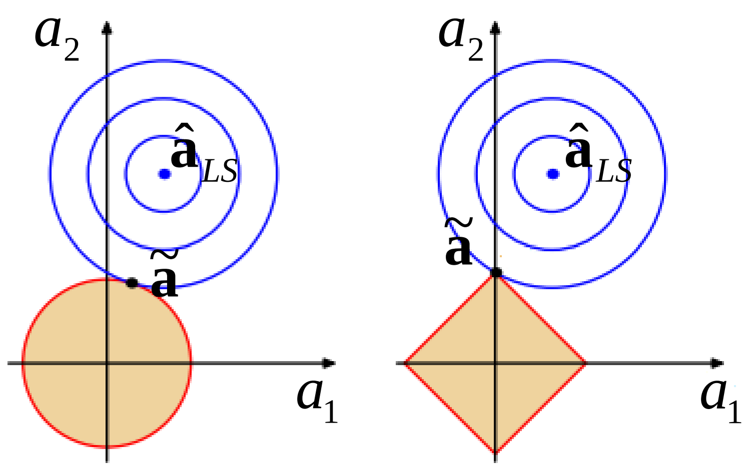
- L1正則化(LASSO)を用いた線形回帰 **(least absolute shrinkage and selection operator)**

$$E''(\mathbf{a},b)=\sum_{i=1}^{N}\left(y_i-\left(\mathbf{a}^T\mathbf{x}_i+b\right)\right)^2+\gamma\underline{\|\mathbf{a}\|_1}$$

L1ノルム $\sum_i |a_i|$

  - スパース（ゼロ要素が多い）な係数ベクトルが出やすい
  - 汎化性能が向上する場合がある。特徴選択にも使われる。
  - データ容量や計算コストの削減につながる

sklearn.linear_model.Lasso など



L2正則化　　L1正則化

Lpノルム正則化

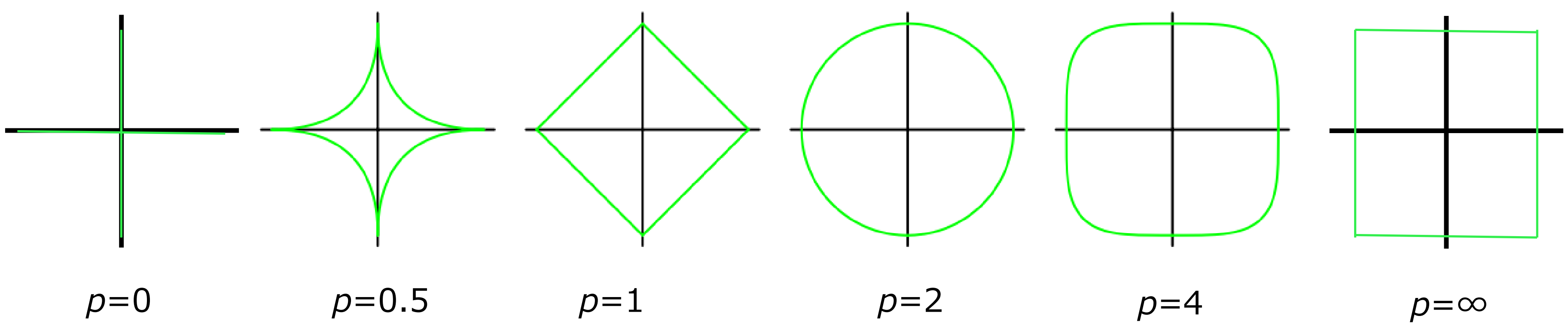
- スパースネスの程度をコントロールできる

$$\|\mathbf{a}\|_p = \left(|a_1|^p + |a_2|^p + \cdots + |a_d|^p\right)^{\frac{1}{p}}$$

ただし $$\|\mathbf{a}\|_0 = \sum_{i=1}^{d} \delta(a_i) \qquad \delta(a_i) = \begin{cases} 1 & (a_i \neq 0) \\ 0 & (a_i = 0) \end{cases}$$

$$\|\mathbf{a}\|_\infty = \max\left(|a_1|, |a_2|, \cdots, |a_d|\right)$$

(一般には、p=1,2以外について解くのは難しい…)



c.f. Yukawa & Amari, “Lp-regularized least squares (0<p<1) and critical path”, 2013.

Elastic Net [Zhou et al., 2005]

- L1正則化(Lasso) + L2正則化(Ridge regression)

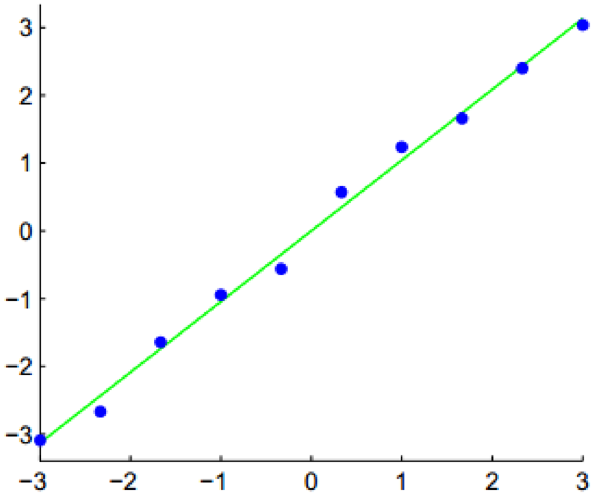
$$E''(\mathbf{a}, b) = \sum_{i=1}^{N} \left(y_i - \left(\mathbf{a}^T \mathbf{x}_i + b \right) \right)^2 + \gamma_1 \|\mathbf{a}\|_1 + \gamma_2 \|\mathbf{a}\|_2$$

  - リッジ回帰を解いたあと、Lassoの最適化を行う

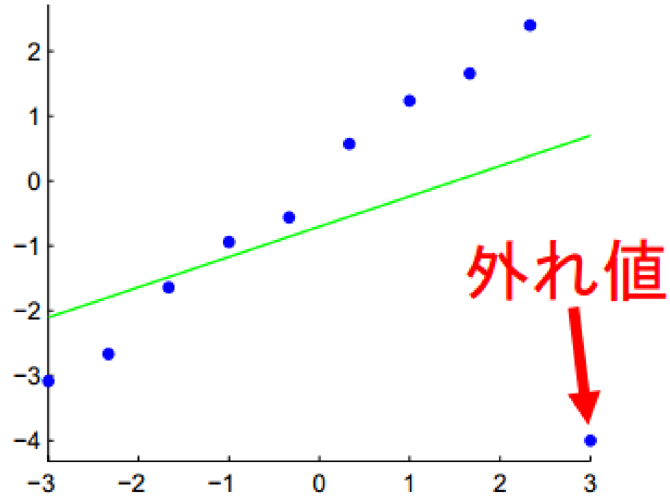
sklearn.linear_model.ElasticNet　など
http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.ElasticNet.html

線形回帰の弱点（２）

- 二乗誤差基準の場合、外れ値（異常値）に弱い
  - 測定のノイズなど
  - 値のズレが二乗で効いてくる！



通常の線形回帰
（外れ値なし）



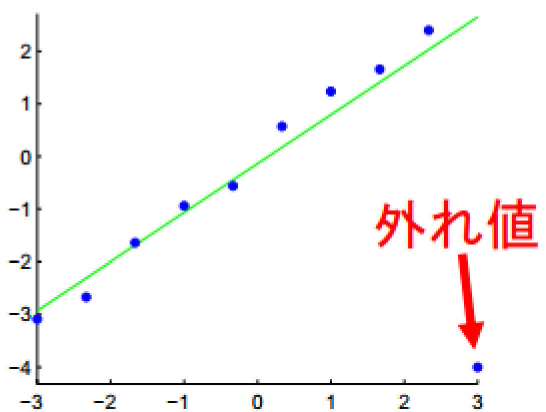
通常の線形回帰
（外れ値あり）

（図：杉山将）

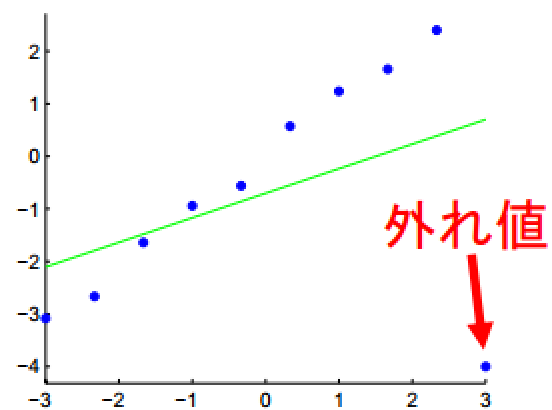
ロバスト線形回帰

- L1損失関数による誤差基準
  - 線形計画問題で解を求める
    （あるいはRANSACという乱数ベースアルゴリズム）

$$\hat{\mathbf{a}}_{l1} = \arg\min_{\mathbf{a}} \sum_{i=1}^{N} \left| y_i - \left(\mathbf{a}^T \mathbf{x}_i + b \right) \right| \qquad \hat{\mathbf{a}}_{l2} = \arg\min_{\mathbf{a}} \sum_{i=1}^{N} \left(y_i - \left(\mathbf{a}^T \mathbf{x}_i + b \right) \right)^2$$



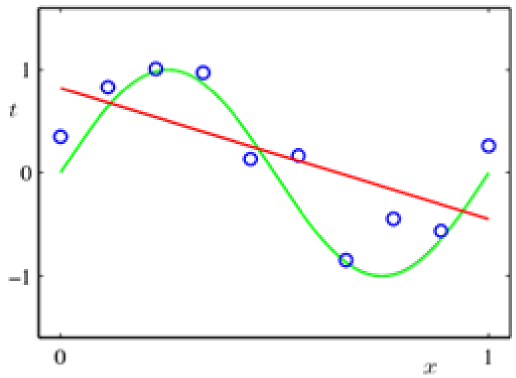
ロバスト学習



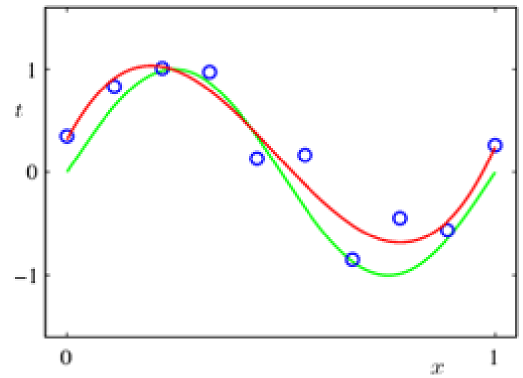
通常の
最小二乗学習

非線形回帰分析

- 目的変数と説明変数が非線形な関係である場合の回帰分析
  - 局所線形モデル（先週の資料参照）
  - 一般化線形モデル
  - 多層パーセプトロン
  - カーネル回帰分析（後日. 多様体学習と密接に関連.）



線形回帰



多項式回帰（次数３）

一般化線形モデル (generalized linear model)

- 目的変数が正規分布以外の指数分布族に従うモデル
  - 目的変数にある関数で非線形変換を加えると、説明変数の線形結合で表現できる

$$g(\underline{\mu_i}) = \mathbf{a}^T \mathbf{x}_i + b$$

$\mu_i = E[y_i]$: 目的変数の分布の平均（期待値）　$g()$ をリンク関数と呼ぶ

  - Rのstatsパッケージの関数glm

> glm(formula, family, data)

回帰の種類	分布族 (family)	リンク関数	
線形回帰	正規分布 (gaussian)	μ	link="identity"
ポアソン回帰	ポアソン分布 (poisson)	log(μ)	link="log"
ロジスティック回帰	二項分布 (binomial)	log(μ/(1-μ))	link="logit"

線形回帰（基本）

- 目的変数（の残差）は等分散の正規分布に従う

$$y_i = \mathbf{a}^T \mathbf{x}_i + b + \varepsilon_i \qquad \left(\varepsilon_i \sim N\left(0, \sigma^2\right)\right)$$

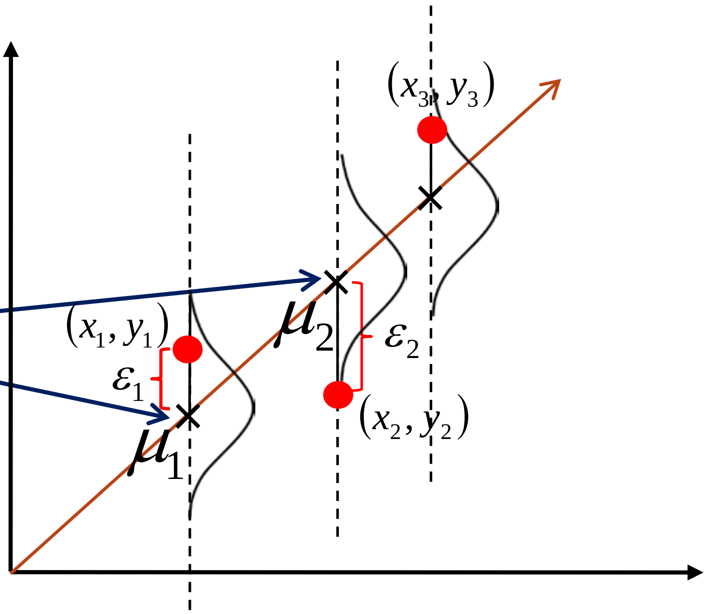
平均0、分散σ^2のガウシアン

$$\mu_i = E\left[y_i\right] = \mathbf{a}^T \mathbf{x}_i + b$$

$$\Updownarrow$$

$$g_{LR}\left(\mu_i\right)$$

リンク関数は恒等変換



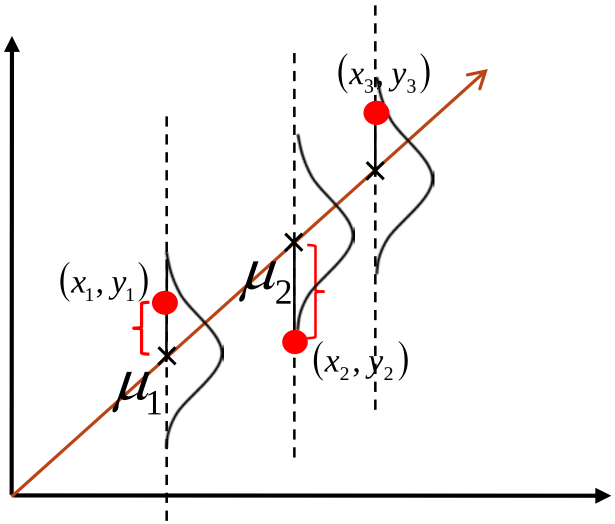
最尤推定との関係性

- 尤度：ある仮説（モデル）のもとで観察されたデータが生じる確率
  - 尤度（もっともらしさ）を最大とするパラメータを求める

$$L = \prod_{i=1}^{N} \underline{p(y_i \mid \mathbf{x}_i, \mathbf{a}, b)}$$

パラメータa,bのもとでのyiの事後確率
(各ｘiは独立試行)

$$= \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{\left(y_i - \left(\mathbf{a}^T \mathbf{x}_i + b\right)\right)^2}{2\sigma^2} \right)$$



対数尤度

$$\log L = -\frac{N}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2} \boxed{\sum_{i=1}^{N} \left(y_i - \left(\mathbf{a}^T \mathbf{x}_i + b\right)\right)^2}$$

結局これを最小化

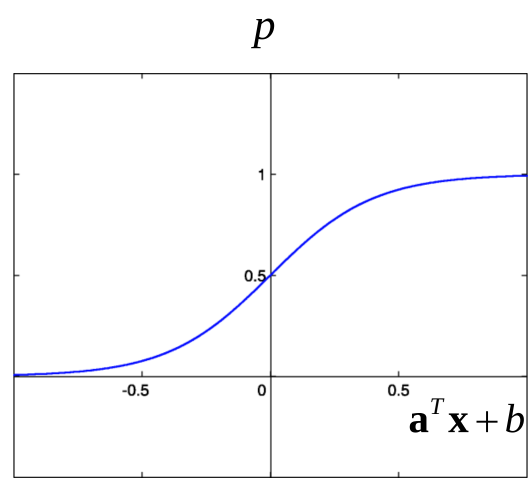
ロジスティック回帰

- 目的変数がベルヌーイ分布に従う場合（e.g., コインの裏表)
- 二値データ（質的データ）の回帰
  - 実用上はクラス識別の手法として解釈される場合が多い

y_i は0か1の二値（ベルヌーイ分布）

$$P(y_i = 1 \mid \mathbf{x}_i) = p_i, \quad P(y_i = 0 \mid \mathbf{x}_i) = 1 - p_i$$

$$p_i = \frac{\exp(\mathbf{a}^T \mathbf{x}_i + b)}{1 + \exp(\mathbf{a}^T \mathbf{x}_i + b)}$$ とおく（ロジスティック関数）



（yの分布ではないので注意）

$$E[y_i] = \mu_i = 1 \times p_i + 0 \times (1 - p_i) = \frac{\exp(\mathbf{a}^T \mathbf{x}_i + b)}{1 + \exp(\mathbf{a}^T \mathbf{x}_i + b)}$$

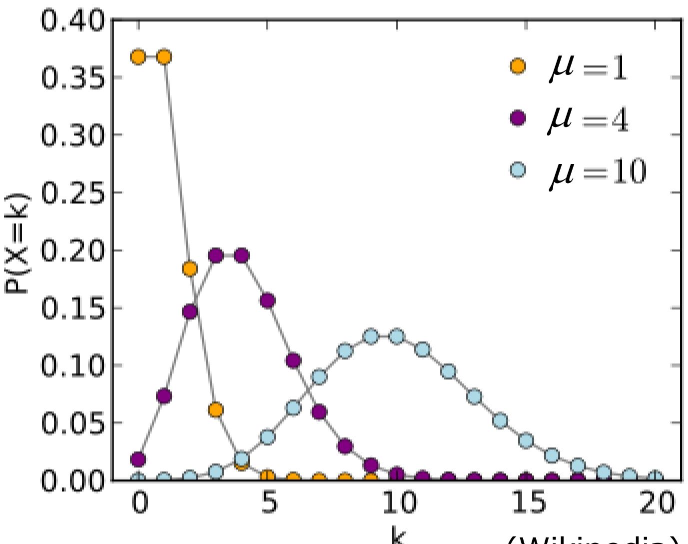
$$g(\mu_i) = \log\left(\frac{\mu_i}{1 - \mu_i}\right) = \mathbf{a}^T \mathbf{x}_i + b$$ リンク関数はロジット関数

ポアソン回帰

- 目的変数がポアソン分布に従う場合
  - ある一定の時間内に平均μ回発生する事象がk回発生する確率

- 計数（カウント）データに使う

$$P(y_i = k) = \frac{\mu_i^k e^{-\mu_i}}{k!}$$



(Wikipedia)

$$E[y_i] = \mu_i = \exp(\mathbf{a}^T \mathbf{x}_i + b)$$ （と仮定する）

$$g_{Po}(\mu_i) = \log(\mu_i) = \mathbf{a}^T \mathbf{x}_i + b$$

対数線形モデル

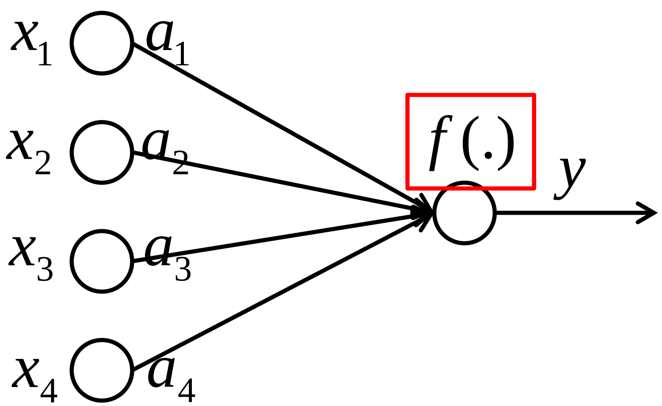
例）Large-scale behavioral targeting [Chen et al., KDD'09]

- 広告CTRをユーザの行動データから予測

$\mu_i = \mathbf{a}^T \mathbf{x}_i$ で解いている

ニューラルネットワークとの関係

- 単純パーセプトロン
  - 一般化線形モデルと本質的には同じ（最尤推定で解けば）



$$y = \boxed{f(\eta)}$$ 活性化関数

$$\eta = \mathbf{a}^T \mathbf{x}$$

$$f(\eta) = \eta$$ → 線形回帰と等価

$$f(\eta) = \frac{1}{1+\exp(-\eta)}$$ → ロジスティック回帰と等価

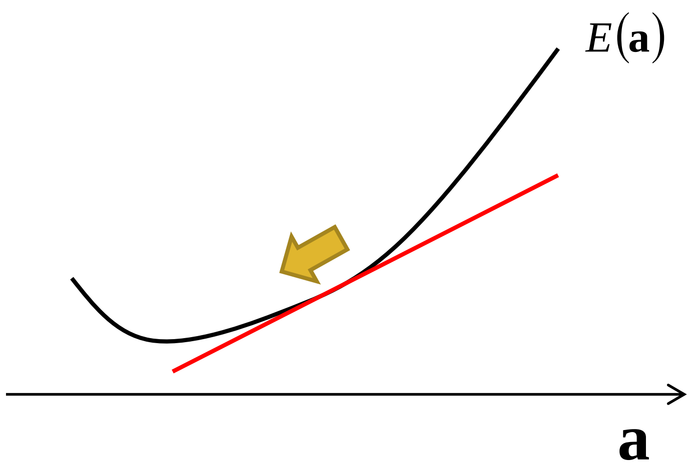
$$f(\eta) = \exp(\eta)$$ → ポアソン回帰と等価

※ただしそれぞれ出力yに適切な確率分布を仮定すれば

勾配降下法

- 収束するまで以下のようにパラメータを更新
  - 目的関数の勾配を下る方向へ少しずつ移動
  - α は微小な正のハイパーパラメータ

$$\mathbf{a} \Leftarrow \mathbf{a} - \alpha \frac{\partial E(\mathbf{a})}{\partial \mathbf{a}}$$

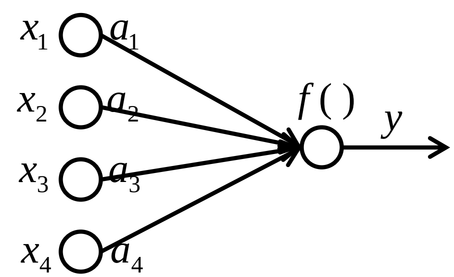


- 解析的に解けない問題で威力を発揮

線形回帰の場合

- 二乗誤差最小化（ $f(\eta)=\eta$ の場合）

$$\varepsilon^2 = \sum_{i=1}^{N}(y_i - \hat{y}_i)^2 = \sum_{i=1}^{N}(y_i - \mathbf{a}^T\mathbf{x}_i)^2$$



$$\frac{\partial \varepsilon^2}{\partial a_k} = -2\sum_{i=1}^{N}(y_i - \hat{y}_i)\,x_{ik}$$

なので、最急降下法による更新式は、

$$\boxed{a_k \Leftarrow a_k + \alpha\left(\sum_{i=1}^{N}(y_i - \hat{y}_k)x_{ik}\right)}$$

Widrow-Hoffの学習規則

※ $f(\eta)=\eta$ の時は、実際は解析解が求まる（線形回帰）

ロジスティック回帰の場合

- 最尤推定

訓練データ集合 $\{\mathbf{x}_i, t_n\},\ t_n \in \{0, 1\}$

尤度関数は $L = \prod_{i=1}^{N} p_i^{t_i} \{1 - p_i\}^{1-t_i}$ $\quad p_i = P(y_i = 1 | \mathbf{x}_i) = \frac{\exp(\mathbf{a}^T \mathbf{x}_i + b)}{1 + \exp(\mathbf{a}^T \mathbf{x}_i + b)}$

負の対数尤度は $E(\mathbf{a}) = -\ln L = -\sum_{i=1}^{N} \{t_i \ln p_i + (1 - t_i) \ln(1 - p_i)\}$

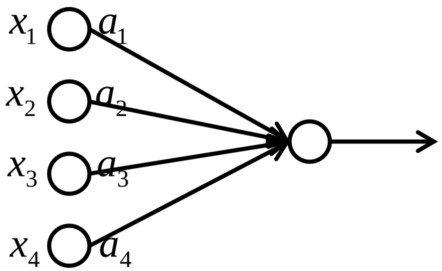
Cross-entropy loss

$$\frac{E(\mathbf{a})}{\partial \mathbf{a}} = \sum_{i=1}^{N} (p_i - t_i)\mathbf{x}_i$$

エラーに説明変数をかけたもの

一般化線形モデルとパーセプトロン

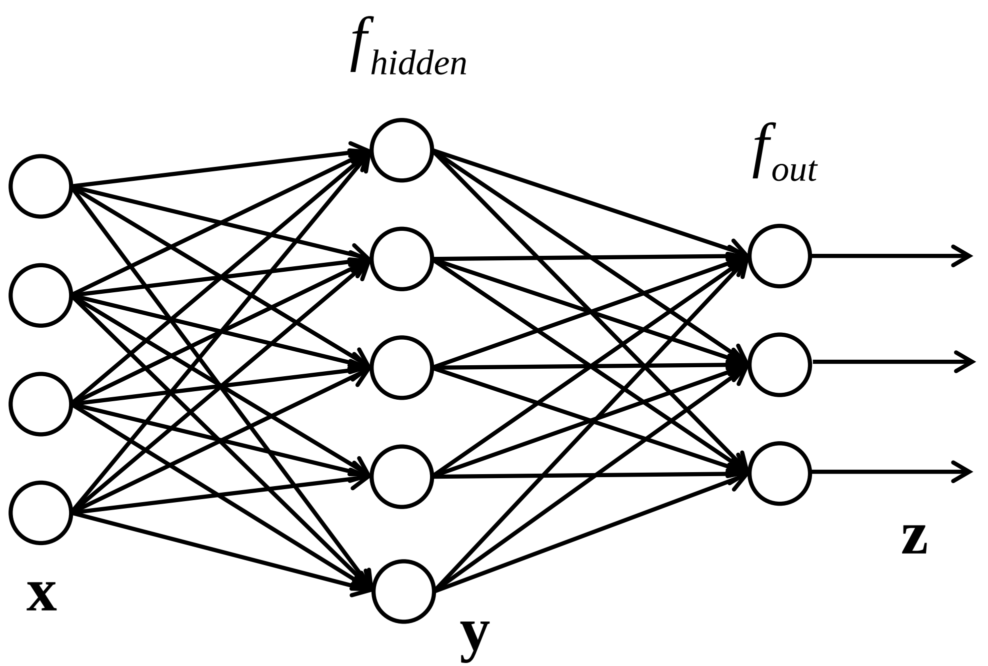
- 同じものの見方の立場の違い



	一般化線形モデル	パーセプトロン
予測	確率分布（統計的）	決定的
非線形変換	リンク関数 ⇔ 逆関数 ⇔	活性化関数
学習	最尤推定 → 背景 →	損失関数

多層パーセプトロン

- 充分な数の素子があれば、任意の連続関数は３層のニューラルネットワークで近似できる
  - 誤差逆伝播法と呼ばれる方法でパラメータを最適化
  - 最近はもっと多層にするのがはやり（deep learning)
  - 本講義後半で詳しく取り上げる予定

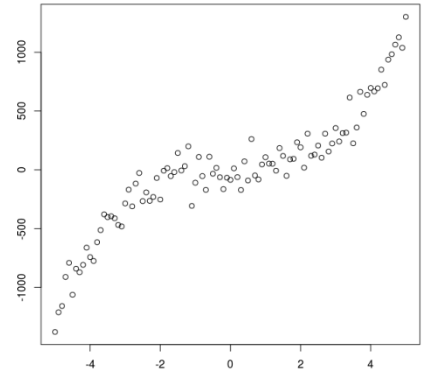


その他の非線形回帰

- 一般の関数を用いた回帰
  - Rの関数 nls など
    nls(formula, data, start, trace)

  - 多項式回帰の例

```
> set.seed(30)
> x<-seq(-5,5,0.1)
> y<-10*x^3+100*rnorm(x,0,1) #3次式に従う人口データを生成
> plot(x,y)
```



つづき

```
> fm3<-nls(y~a+b*x+c*x^2+d*x^3,start=c(a=1,b=1,c=1,d=1),trace=T)
21031980 :  1 1 1 1
1073246 :   1.3077234 13.8639457 -0.9720568  9.4123383
> summary(fm3)

Formula: y ~ a + b * x + c * x^2 + d * x^3

Parameters:
  Estimate Std. Error t value Pr(>|t|)
a   1.3077    15.7011   0.083    0.934
b  13.8639     8.9781   1.544    0.126
c  -0.9721     1.3769  -0.706    0.482
d   9.4123     0.5379  17.498   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 105.2 on 97 degrees of freedom

Number of iterations to convergence: 1
Achieved convergence tolerance: 1.103e-07

> AIC(fm3)
[1] 1233.004
```

任意に指定可能（解ければ）

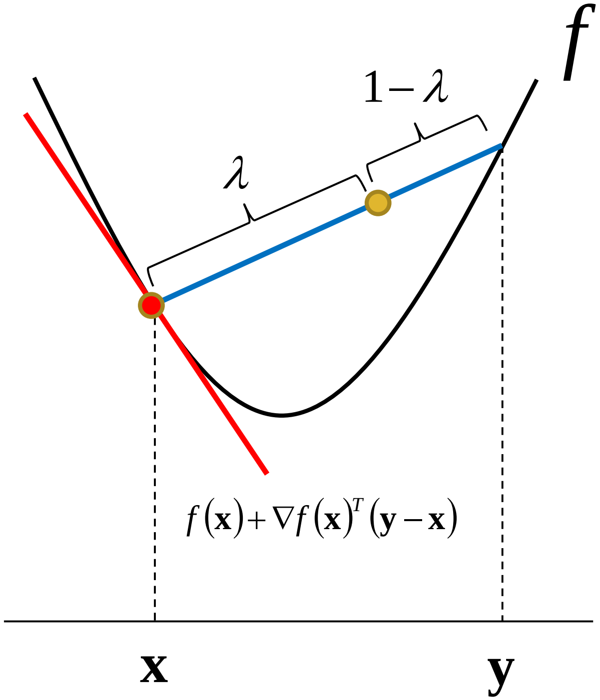
付録：LASSOの解説と実装

- 要点だけ
  - 詳しく知りたい人は最適化の教科書などを参照
- この他にもいろんなアルゴリズムがある

凸関数

- 定義：ある関数 $f:R^n \to R$ が凸である
  ≜ f 上の任意の2点の内分点が f より上に位置する

$$(1-\lambda)f(\mathbf{x})+\lambda f(\mathbf{y}) \geq f((1-\lambda)\mathbf{x}+\lambda\mathbf{y})$$



- 一次微分が常に存在すれば…

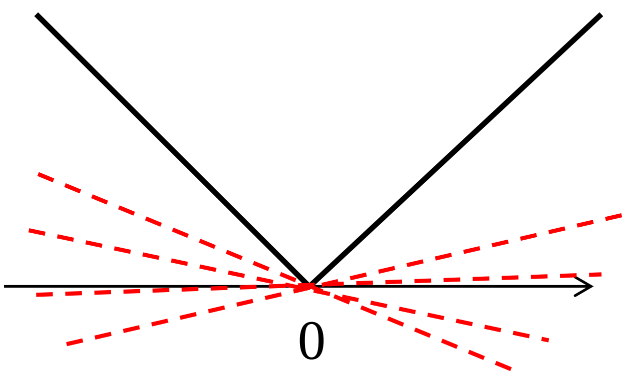
$$f(\mathbf{y}) \geq f(\mathbf{x})+\nabla f(\mathbf{x})^T(\mathbf{y}-\mathbf{x})$$

$\nabla f(\mathbf{x})=0$ なる x が大域的最適解 ← 勾配法

微分が定義できない場合…

- 劣勾配 (subgradient)
  - ある点 x における劣勾配とは任意の z について
    $$f(\mathbf{z}) \geq f(\mathbf{x}) + \mathbf{g}^T(\mathbf{y}-\mathbf{x})$$
    が成立する任意の $\mathbf{g} \in R^n$

$f(x) = |x| \quad x \in R$



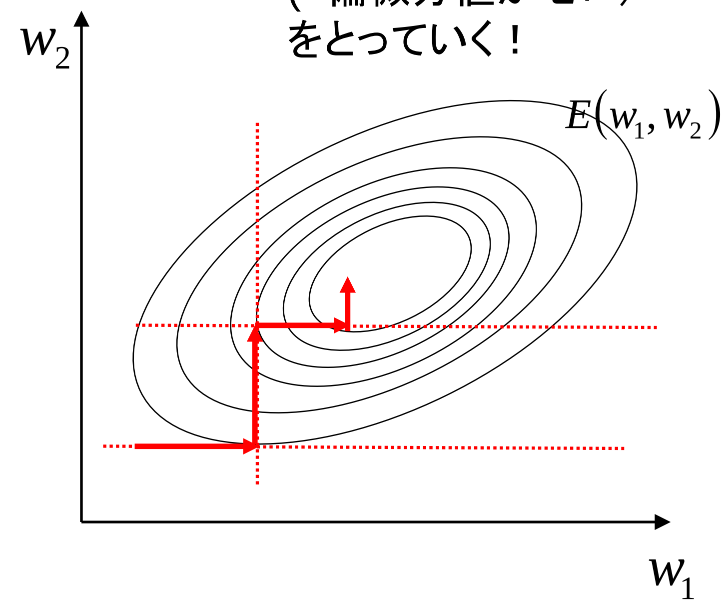
$-1 \leq c \leq 1$ はすべて
$x = 0$ における劣勾配

- 劣勾配を用いた最適化
  - 全ての微分不可能点に最低一つの劣勾配が存在すれば、これを用いて勾配降下法が実行可能
  - もちろん勾配法を必ずしも使う必要はない（e.g.,解析解）

座標降下法

- パラメータごとに偏微分をとり、更新

毎回各断面での最適点
(=偏微分値がゼロ)
をとっていく！



$$\text{initialize} \quad w_1, w_2, \ldots, w_n$$

$$\text{for } t = 1,2,\ldots,T \quad \text{(until convergence)}$$

$$\qquad \text{for } \; j = 1,2,\ldots,n$$

$$\qquad\qquad \text{set} \quad w_j = \underset{w_j}{\arg\min}\, E(w_1, \ldots, w_n)$$

- 更新の順番
  - 常に固定 (cyclic)
  - ランダム
などいろいろ（それぞれ収束性は証明されている）

- パラメータごとの解析が容易な場合に力を発揮
  - 勾配降下法におけるステップ幅の設定が必要ない

LASSOにおける実装(1)

- 目的関数： $E(\mathbf{w}) = \sum_{i=1}^{N} \left(y_i - \mathbf{w}^T \mathbf{x}_i \right)^2 + \gamma \|\mathbf{w}\|_1$

$$\frac{\partial E}{\partial w_k} = \frac{\partial}{\partial w_k} \sum_{i=1}^{N} \left(y_i - \mathbf{w}^T \mathbf{x}_i \right)^2 + \gamma \frac{\partial}{\partial w_k} \sum_{j=1}^{d} |w_j|$$

$$= a_k w_k - c_k + \gamma \frac{\partial |w_k|}{\partial w_k}$$

ただし

$$a_j = 2 \sum_{i=1}^{N} x_{i,j}^2$$

$$c_j = 2 \sum_{i=1}^{N} x_{i,j} \left(y_i - \mathbf{w}_{-j}^T \mathbf{x}_{i,-j} \right)$$

$$\frac{\partial |w_k|}{\partial w_k} = \begin{cases} -1 & w_k < 0 \\ [-1, 1] & w_k = 0 \\ 1 & w_k > 0 \end{cases}$$ ← 劣勾配

$-j$ はj番目の要素だけを除くベクトルを示す添字

LASSOにおける実装(2)

以上を代入すると
$$\frac{\partial E}{\partial w_k}=\begin{cases} a_k w_k - c_k - \gamma & w_k<0 \\ \left[-c_k-\gamma, -c_k+\gamma\right] & w_k=0 \\ a_k w_k - c_k + \gamma & w_k>0 \end{cases}$$

以下 $\frac{\partial E}{\partial w_k}=0$ を考える。$w_k<0$ の時

$$a_k w_k - c_k - \gamma = 0$$

$w_k = (c_k+\gamma)/a_k < 0$ よりこれは $c_k<-\gamma$ に対応

他の場合も同様にして、最終的に

$$w_k=\begin{cases} (c_k+\gamma)/a_k & c_k<-\gamma \\ 0 & -\gamma\le c_k\le\gamma \\ (c_k-\gamma)/a_k & c_k>\gamma \end{cases}$$

を得る。これが座標降下法における1ステップとなる。