

データサイエンス 第5回

～回帰分析（１）～

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

- 回帰分析
 - 線形回帰分析
 - リッジ回帰
 - 正則化、チューニング
-

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
	質的データ	数量化Ⅱ類

回帰分析

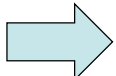
- 説明変数から目的変数を予測するモデル（関数）を構築
 - （典型的には）量的データから量的データを予測
 - 例）人口・気温から消費電力を予測

以下では

p 次元のベクトル入力 \mathbf{x} から実数値 y を出力するモデル $y = f_{\theta}(\mathbf{x})$ を、 N 個の学習サンプル $\{\mathbf{x}_i, y_i\}_{i=1}^N$ から学習する

という問題を考える

（ θ はモデルのパラメータ）

- 学習サンプルに最もフィットするモデルを求めたい
 - “フィット”するとは？  予測と真の値のズレを最小化する！

最小二乗法 (least squares)

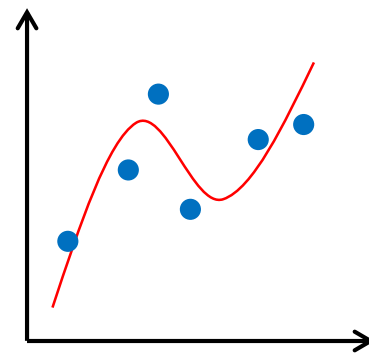
- 学習サンプルにおける、モデルの出力 $\hat{y}_i = f_{\theta}(\mathbf{x}_i)$ と真値 y_i の二乗誤差

$$E(\theta) = \sum_{i=1}^N (y_i - \hat{y}_i)^2 = \sum_{i=1}^N (y_i - f_{\theta}(\mathbf{x}_i))^2 \text{ を最小化}$$

- 誤差の分布に分散一定の正規分布を仮定
 - 他にも誤差の基準はいろいろある (異なる分布に対応)
 - 二乗誤差を基準にすると実用上はとても便利
- 例) 多項式フィッティング (xは1次元)

$$f(x) = w_0 + w_1x + w_2x^2 + w_3x^3 + \cdots + w_Mx^M$$

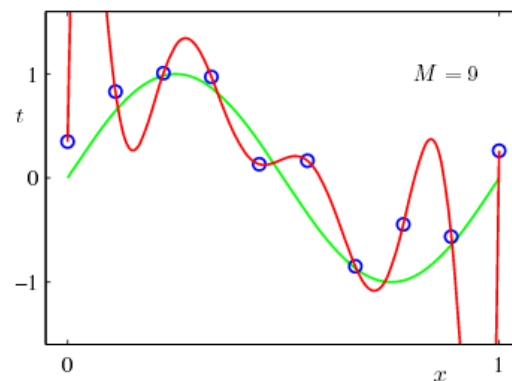
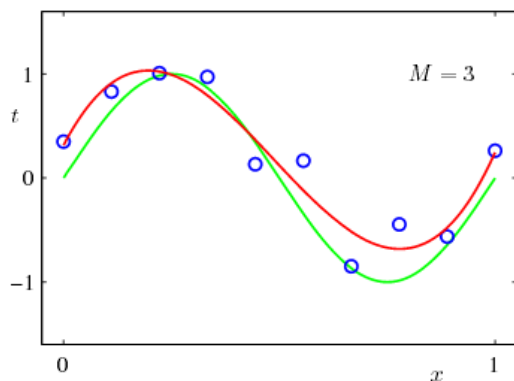
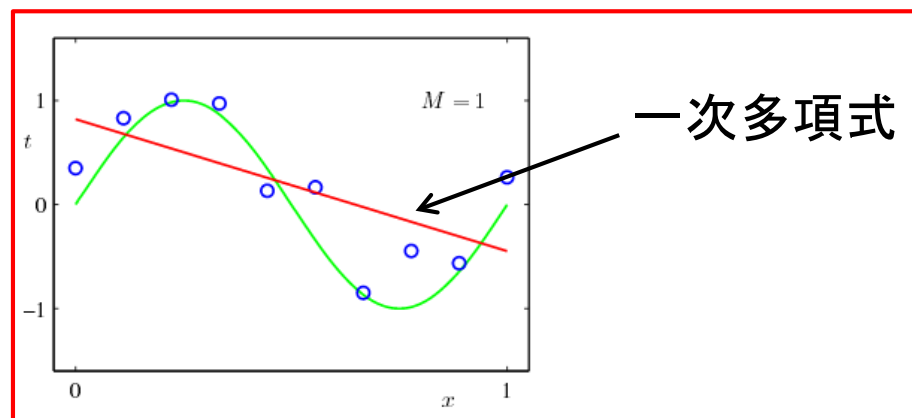
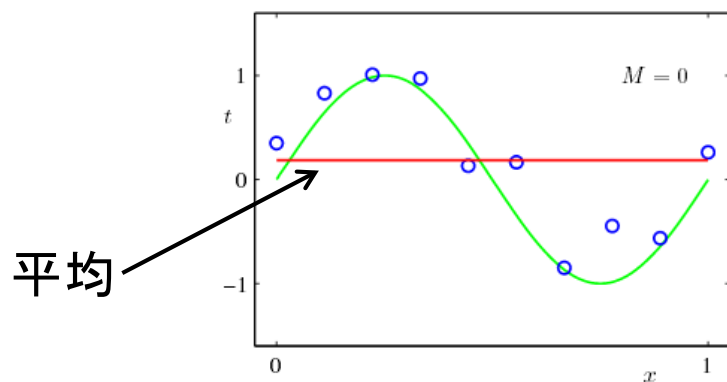
モデルパラメータ $(w_0, w_1, w_2, \dots, w_M)$ を学習



Q. 複雑なモデル(パラメータが多い)ほど性能がよいのだろうか？

過学習（オーバーフィッティング）

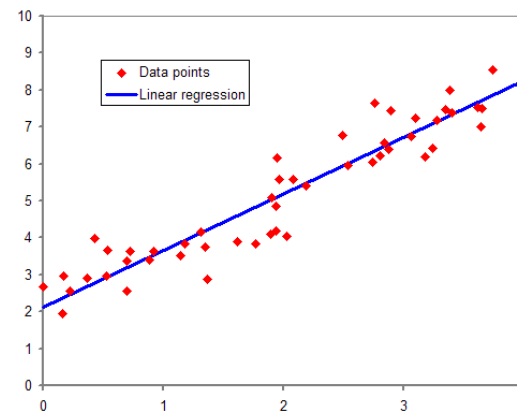
- 訓練誤差は小さいが、汎化誤差（予測誤差）が著しく大きい状態
 - データ数や実際の構造に対して複雑すぎるモデルを用いることで生じる
- 多項式回帰の次数を上げていけば、全ての学習データを通る線は引ける
 - 実際には意味のない解！（訓練誤差はゼロだが汎化誤差は非常に大きくなる）
 - 学習サンプルは有限



C.M.ビショップ
「パターン認識と
機械学習」より

線形回帰: linear regression

- 説明変数の一次式（線形結合）によるモデル
 - 単回帰：説明変数が一つ $y = ax + b$
 - 重回帰：説明変数が複数 $y = a_0x_0 + a_1x_1 + a_2x_2 + \cdots + a_px_p + b = \mathbf{a}^T \mathbf{x} + b$
 - 説明変数は互いに無相関であることを仮定
- 対象が線形な関係を有する場合に力を発揮
 - 実用上は厳密に線形であることは少ない（そもそも分からないことの方が多い）
 - 予測性能が実用に耐えるかの評価が大事
 - 前処理（説明変数の対数をとったり）は重要
- シンプルなモデルであり、学習は簡単



線形回帰

- 最小二乗法によるモデルフィッティング

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

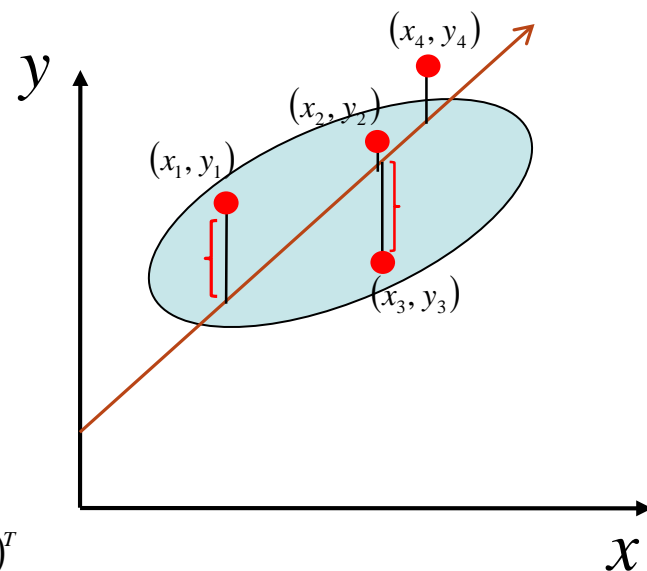
\mathbf{a}, b でそれぞれ偏微分して0とおくと

$$\frac{\partial E}{\partial \mathbf{a}} = -2 \sum_{i=1}^N \mathbf{x}_i (y_i - \mathbf{a}^T \mathbf{x}_i) = 0 \quad \Rightarrow \quad \hat{\mathbf{a}} = \left(\sum_{i=1}^N \mathbf{x}_i \mathbf{x}_i^T \right)^{-1} \left(\sum_{i=1}^N \mathbf{x}_i y_i \right)$$
$$= \underbrace{(X X^T)^{-1}}_{\text{自己相関行列}} X \mathbf{y}$$

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

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

定数項 b を用意する代わりに、特徴ベクトル \mathbf{x} に常に1の要素を付与してもよい



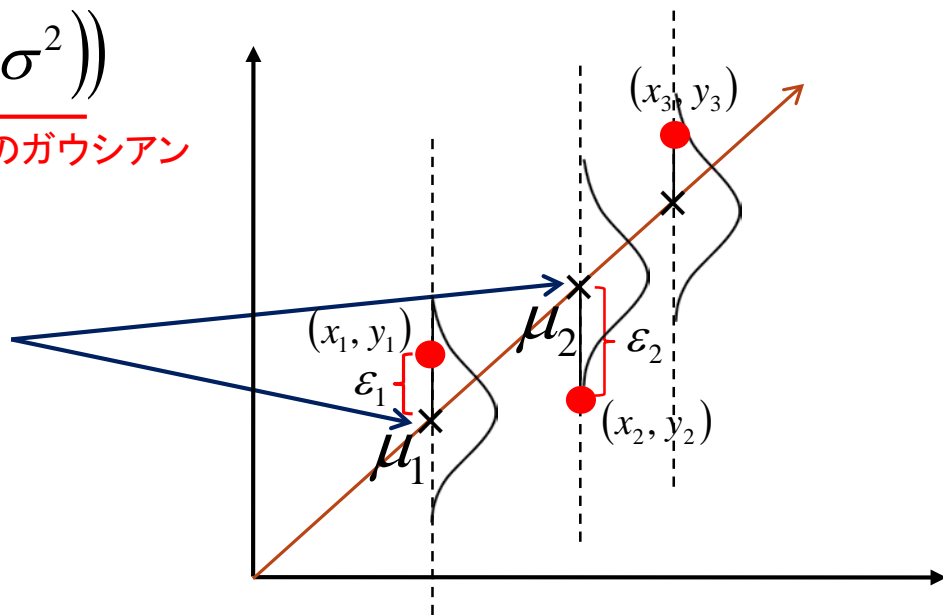
線形回帰

- 目的変数（の残差）が等分散の正規分布に従うことを仮定している

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

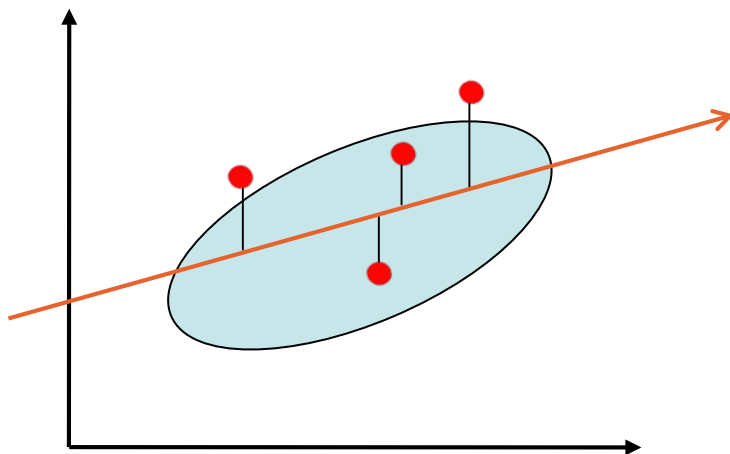
平均0、分散 σ^2 のガウシアン

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

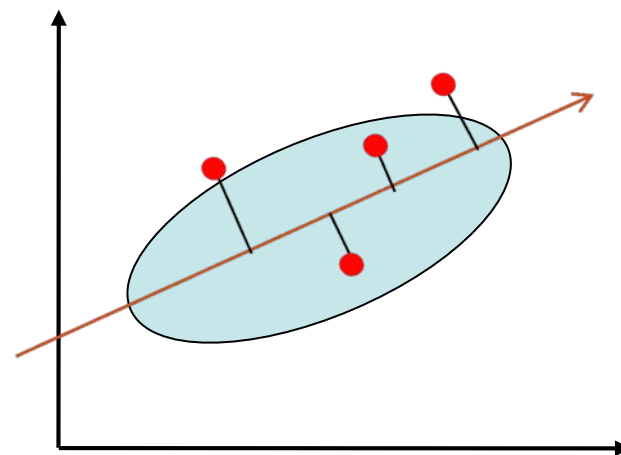


主成分分析(PCA)との違い

- 誤差の測り方が違う



線形回帰



PCA

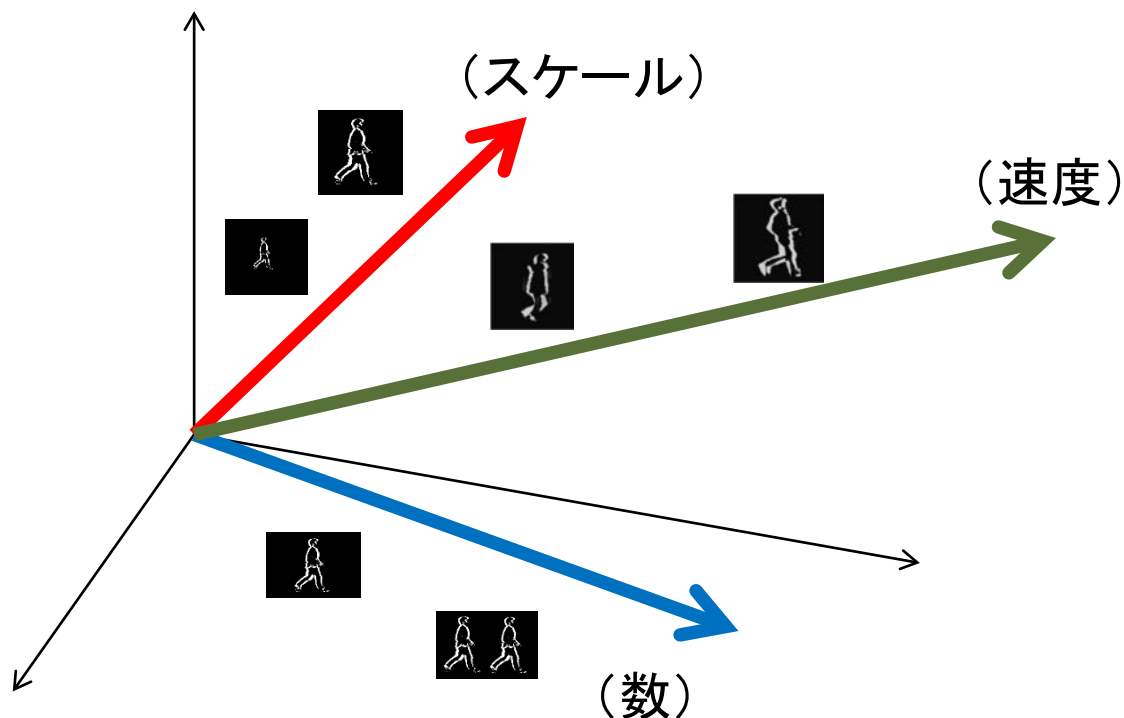
例) 線形回帰で歩行者数を計測

- [Shimohata and Otsu, 2006]



例) 線形回帰で歩行者数を計測

- 特徴空間の線形構造を上手に利用 (CHLAC特徴)

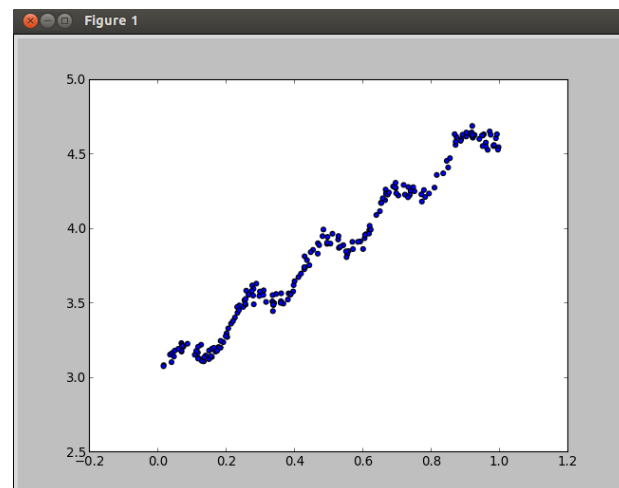


対象と特徴(説明変数)の性質をうまく使えば十分実用的

実行例 (練習問題1)

```
w = standRegres(X,Y)
print(w)
matrix([[1.69532264], ←  $y = 1.6953 * x + 3.0077$ 
        [3.00774324]])
```

yHat = ... ?



※数式では一つのサンプルは通常列ベクトルで表記されるが、実際の実装では行ベクトルの形で格納されることが多いので注意

局所重み付線形回帰

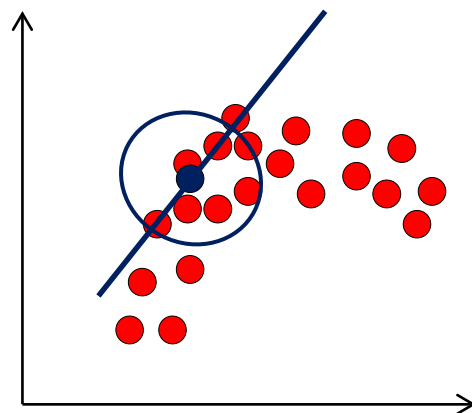
- 入力されたテストデータの近傍学習サンプルに大きな重みを付与して線形回帰を実行する

$$\hat{\mathbf{a}} = \left(X W X^T \right)^{-1} X W \mathbf{y} \quad \leftarrow \text{これを入力されるテストデータごとに解く}$$

W は入力 \mathbf{x} に対する各学習サンプルの重みをもつ対角行列
(非対角要素はゼロ)

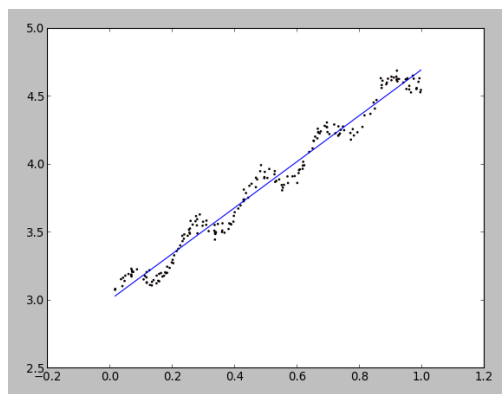
$$W_{i,i} = \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}_i\|^2}{2k^2}\right)$$

- 非線形回帰手法の一つ（局所的には線形）

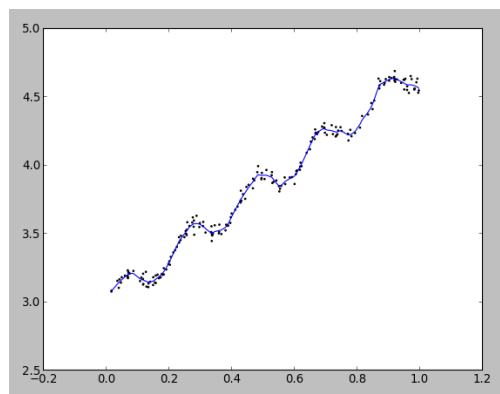


実行例（練習問題1）

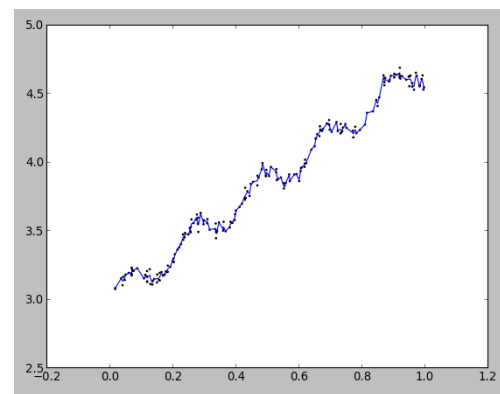
`yHat = lwLRTest(xMat,xMat,yMat,k=0.005)` #全サンプルで実行



$k = 1.0$



$k = 0.01$



$k = 0.003$

UCI abalone データ

- アワビ貝の年齢予測

#性別	長さ	直径	厚さ	総重量	...			年齢
1	0.455	0.365	0.095	0.514	0.2245	0.101	0.15	15
1	0.35	0.265	0.09	0.2255	0.0995	0.0485	0.07	7
-1	0.53	0.42	0.135	0.677	0.2565	0.1415	0.21	9
1	0.44	0.365	0.125	0.516	0.2155	0.114	0.155	10
0	0.33	0.255	0.08	0.205	0.0895	0.0395	0.055	7
0	0.425	0.3	0.095	0.3515	0.141	0.0775	0.12	8
-1	0.53	0.415	0.15	0.7775	0.237	0.1415	0.33	20
-1	0.545	0.425	0.125	0.768	0.294	0.1495	0.26	16
1	0.475	0.37	0.125	0.5095	0.2165	0.1125	0.165	9
-1	0.55	0.44	0.15	0.8945	0.3145	0.151	0.32	19
-1	0.525	0.38	0.14	0.6065	0.194	0.1475	0.21	14
1	0.43	0.35	0.11	0.406	0.1675	0.081	0.135	10
1	0.49	0.38	0.135	0.5415	0.2175	0.095	0.19	11

ダミー変数

- 名義尺度データを説明変数に使用したい場合
 - カテゴリ情報を二値(0,1など)で表す
 - 数量化 I 類と呼ばれる方法と実質的に同等

#性別	長さ	直径	厚さ	総重量	...			年齢
1	0.455	0.365	0.095	0.514	0.2245	0.101	0.15	15
1	0.35	0.265	0.09	0.2255	0.0995	0.0485	0.07	7
-1	0.53	0.42	0.135	0.677	0.2565	0.1415	0.21	9
1	0.44	0.365	0.125	0.516	0.2155	0.114	0.155	10
0	0.33	0.255	0.08	0.205	0.0895	0.0395	0.055	7
0	0.425	0.3	0.095	0.3515	0.141	0.0775	0.12	8



本当は0は欠損値扱いにしないとよろしくないが...

実行してみる

```
def rssError(yArr,yHatArr): # 二乗誤差を測る関数
    return ((yArr-yHatArr)**2).sum()
```

```
data = loadData('abalone.txt',sep='¥t', header=None)
xMat = np.mat(data[:, :8]).T
yMat = np.mat(data[:, 8]).T
```

```
for k in [10,1,.5]:
    yHat=lwLRTest(xMat[:, :100],xMat[:, :100],yMat[:100],k)
    #学習・テストとも同じデータ（最初の100サンプル）で局所線形回帰
    print('k={0}: {1}'.format(k, rssError(yMat[:100].T.A[0],yHat))) #訓練
誤差
```

k=10: 532.9037383272165

k=1: 411.97877254924856

k=0.5: 314.49778461048305

k = 0.5が一番いいのかな… ?

汎化誤差の評価

```
for k in [10,1,.5]:  
    yHat=lwLRTest(xMat[:,100:200],xMat[:, :100],yMat[:100],k)  
    #テストデータを別にとる  
    print('k={0}: {1}'.format(k, rssError(yMat[100:200].T.A[0],yHat)))
```

k=10: 609.9244790868446

k=1: 664.2491584972063

k=0.5: 744.0252058875878 #過学習！

線形回帰の弱点（１）

- 説明変数の間に強い相関や線形従属の関係があると正しい解がでない
(= 多重共線性, multicollinearity)

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

相関行列がランク落ちし、逆行列が求まらない

- 実データでは、説明変数の間に相関がある方がふつう
 - 解けないだけならまだいいが、不安定な解を出すことがある
(行列式がゼロに近いので、係数 \mathbf{a} が異常に大きな値になる)
 - あらかじめ相関の高い説明変数は除外しておくことが望ましい
- サンプルが次元数に対して少ない場合も不安定になりやすい
 - 過学習が起こる

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）

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

- いろんな解釈が可能
 - 学習データにホワイト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**

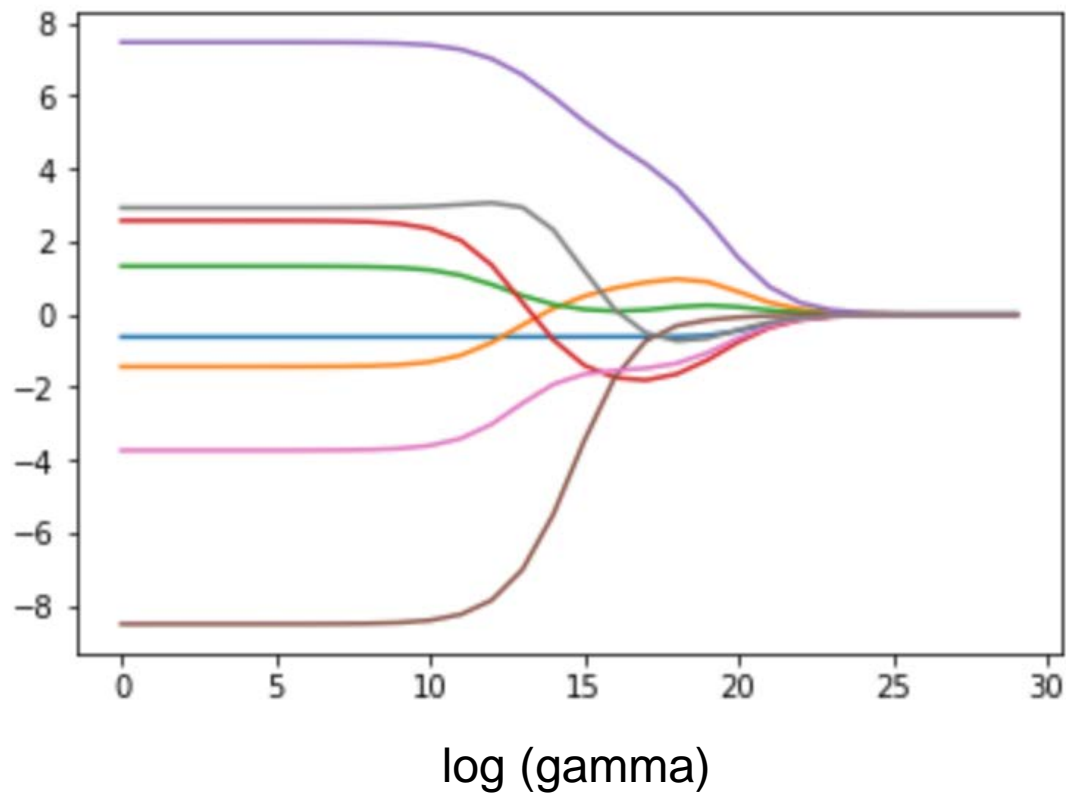
$$E'(\mathbf{a}, b) = \sum_{i=1}^N (y_i - (\mathbf{a}^T \mathbf{x}_i + b))^2 + \gamma \|\mathbf{a}\|^2$$



係数(の絶対値)をできるだけ小さくするような作用

正則化パラメータの影響

coefficients



正則化項を大きくすると係数はゼロに収束

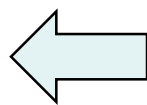
ことば

- 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

$$\mathbf{y} = \mathbf{a}^T \mathbf{x} \quad \hat{\mathbf{a}} = (XX^T)^{-1} X \mathbf{y}$$

- 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)

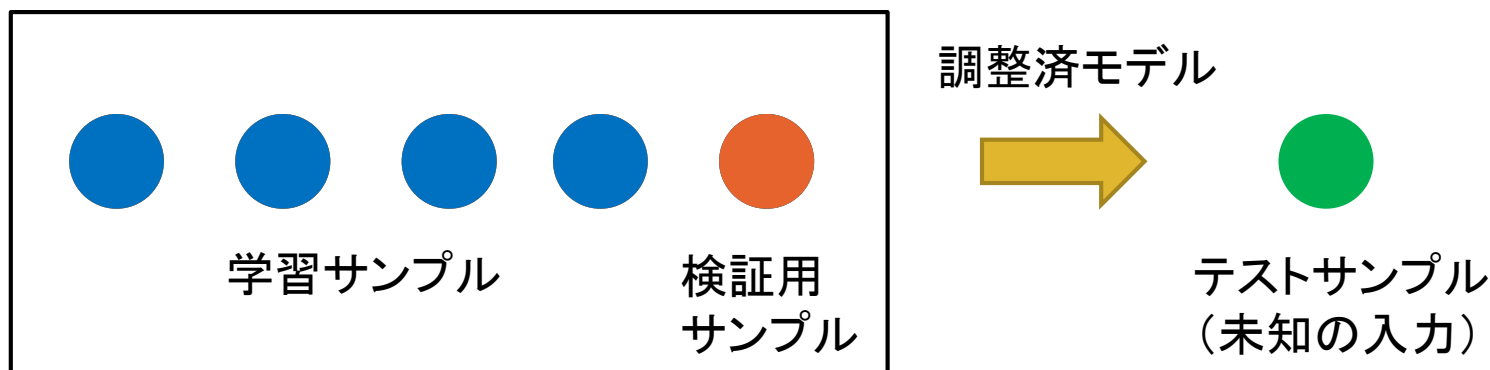
- 学習アルゴリズムに天下りの与える変数
- 例) 正則化の大きさ、多項式回帰の次数

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

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. **別に用意したテストサンプル**での予測誤差を評価し、最終的な評価指標とする。



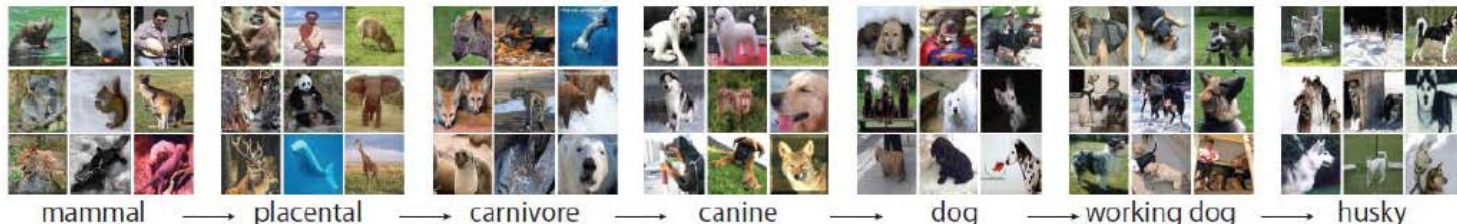
Baidu事件 @ILSVRC

- 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.

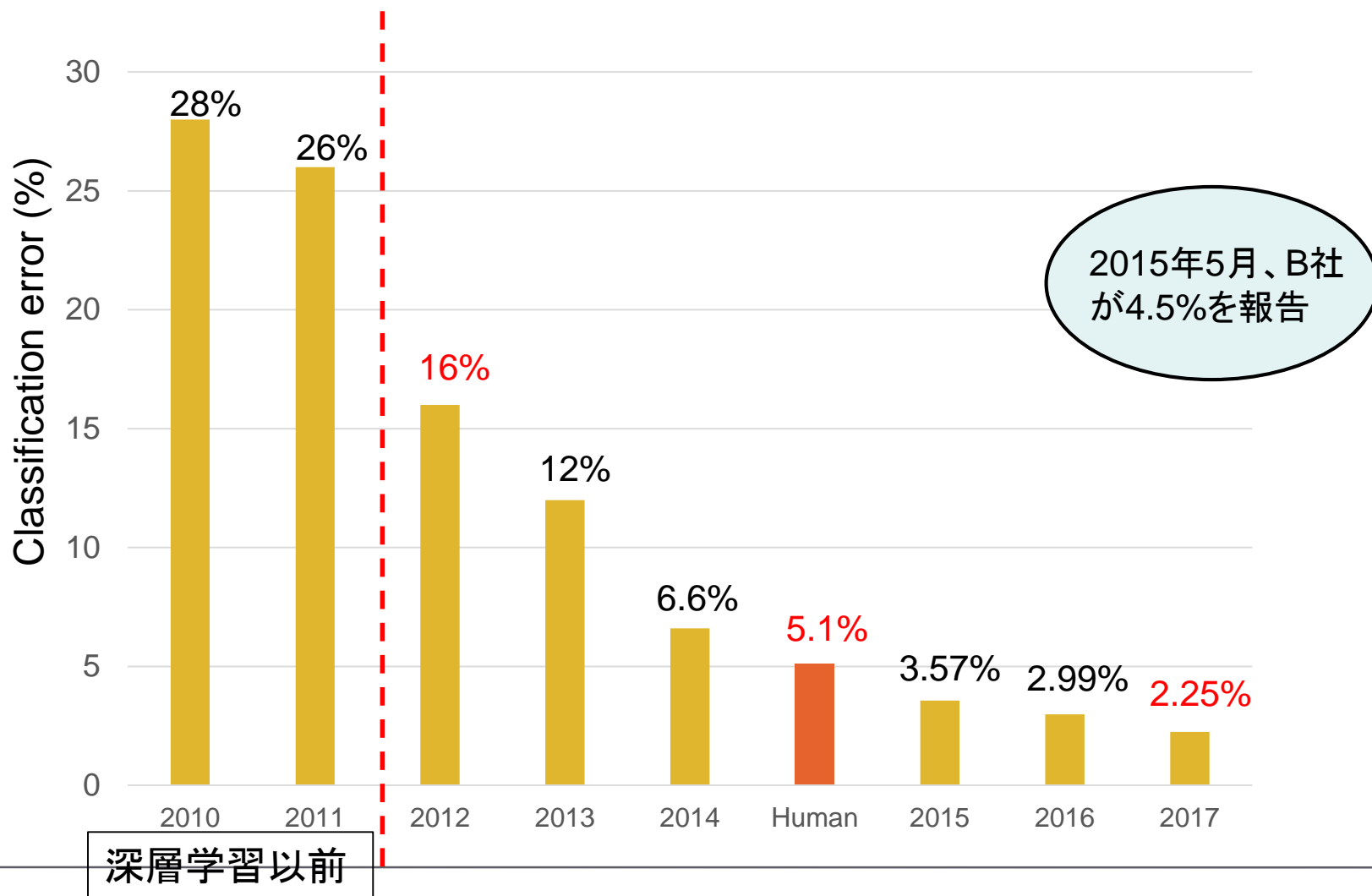
- ImageNetのデータの一部を用いたフラッグシップコンペティション (2010年から2017年まで開催)
 - ImageNet [Deng et al., 2009]
 - クラウドソーシングにより構築中の大規模画像データセット
 - **1400万枚、2万2千カテゴリ** (WordNetに従って構築)



- 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)



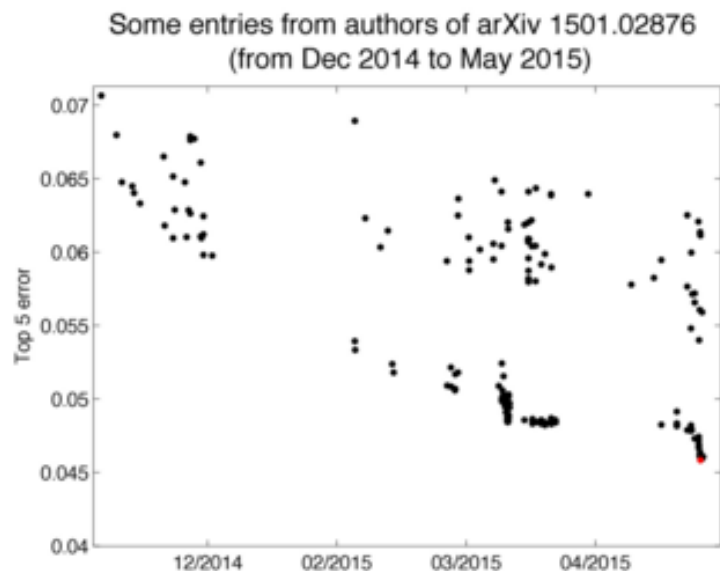
何がまずかったか？

- 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



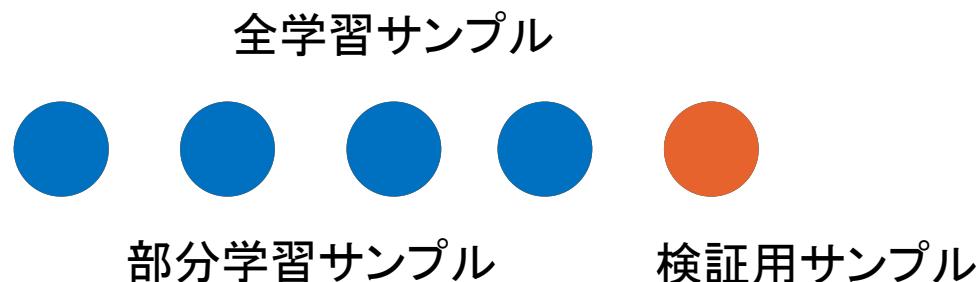
教訓

- 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

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



Lpノルム正則化

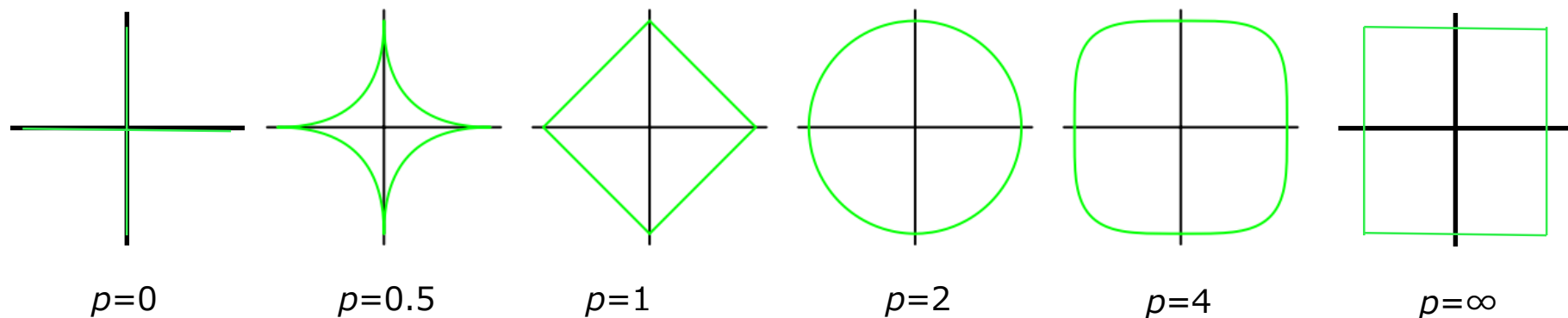
- スパースネスの程度をコントロールできる

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

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

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

(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)

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

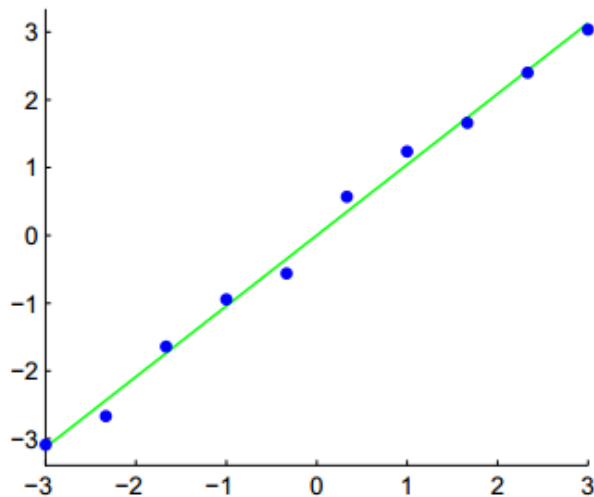
- リッジ回帰を解いたあと、Lassoの最適化を行う

sklearn.linear_model.ElasticNet など

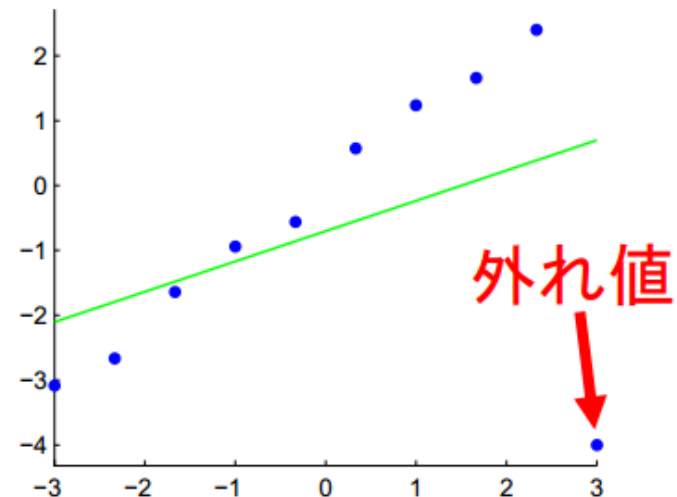
http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.ElasticNet.html

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)



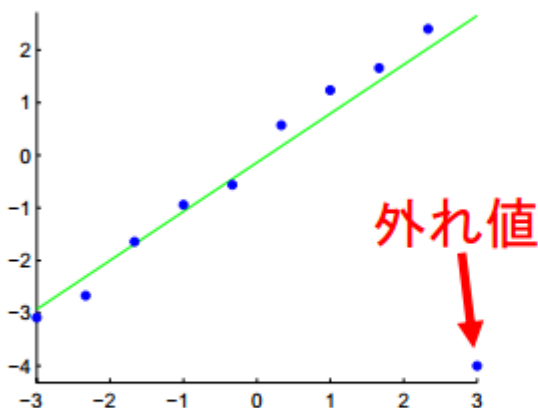
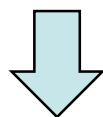
通常の線形回帰
(外れ値あり)

(図: 杉山将)

ロバスト線形回帰

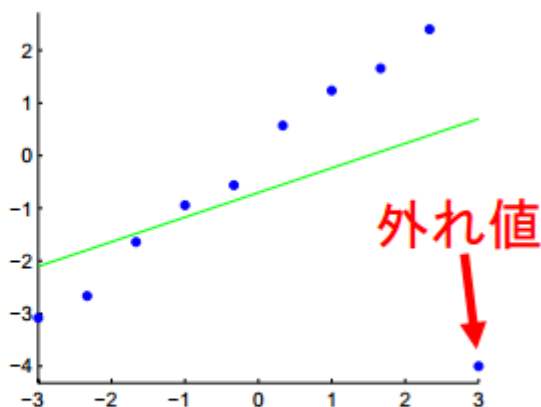
- L1損失関数による誤差基準
 - 線形計画問題で解を求める
(あるいはRANSACという乱数ベースアルゴリズム)

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



ロバスト学習

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



通常の
最小二乗学習

今日のまとめ

- 回帰入門
 - データの性質・量と、適用すべきモデルの複雑さをよく考える
 - 訓練誤差、汎化誤差の違い
 - オーバーフィッティングに注意
- 線形回帰分析
 - 単純だが全ての基本。意外と実問題にも使える。
 - 多重共線性、正則化の意味を理解すること
 - リッジ回帰を使いましょう（交差検証も）
- 来週
 - 勾配降下法
 - 一般化線形モデル（誤差の分布が正規分布以外の場合）