

# 機械学習と データマイニングの基礎

---

大阪大学 産業科学研究所  
原 聡

# 原担当パートの内容

---

## ■ 教師あり学習

- ・ 11/17(金) 回帰と分類
- ・ 11/24(金) スパース正則化

## ■ 教師なし学習

- ・ 12/ 1(金) 密度関数の推定
- ・ 12/ 8(金) 確率的生成モデル

## ■ 成績評価

- ・ レポート課題1回(12/1出題)にて評価

# 【参考】機械学習をより勉強するために

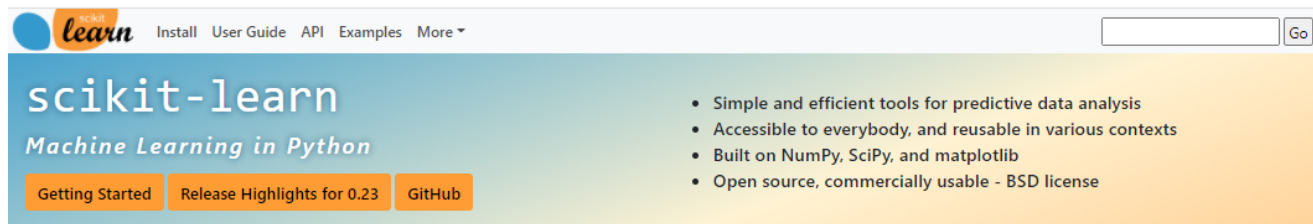
---

- 本講義の内容は機械学習技術のごく一部。
- 講義で扱わない機械学習技術の例：
  - ・ カーネル法、アンサンブルモデル、ブースティング、バイズ的機械学習
- 参考図書（例）
  - ・ 機械学習スタートアップシリーズ – 入門的な内容
  - ・ 機械学習プロフェッショナルシリーズ – 各種トピックの専門的な内容が充実
  - ・ 統計的学習の基礎 – 機械学習のアルゴリズムの紹介が充実
    - Elements of Statistical Learningの日本語訳
  - ・ パターン認識と機械学習 – 主にバイズ的機械学習の入門用
  - ・ Elements of Statistical Learning(<https://web.stanford.edu/~hastie/ElemStatLearn/>)
  - ・ Understanding Machine Learning: From Theory to Algorithms(<https://www.cs.huji.ac.il/~shais/UnderstandingMachineLearning/>)
  - ・ Deep Learning(<http://www.deeplearningbook.org/>)

# 【参考】機械学習をより勉強するために

## ■ scikit-learn (Pythonの機械学習ライブラリ)

- ・ 実用的なアルゴリズムが多数実装されている。簡単に使える。
- ・ ホームページにも各種手法の詳しい解説が充実。

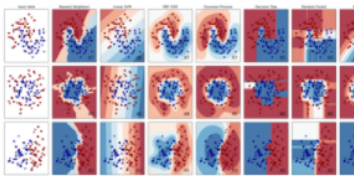


### Classification

Identifying which category an object belongs to.

**Applications:** Spam detection, image recognition.

**Algorithms:** SVM, nearest neighbors, random forest, and more...



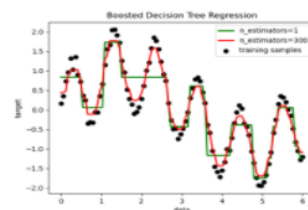
Examples

### Regression

Predicting a continuous-valued attribute associated with an object.

**Applications:** Drug response, Stock prices.

**Algorithms:** SVR, nearest neighbors, random forest, and more...



Examples

### Clustering

Automatic grouping of similar objects into sets.

**Applications:** Customer segmentation, Grouping experiment outcomes

**Algorithms:** k-Means, spectral clustering, mean-shift, and more...



Examples

### Dimensionality reduction

Reducing the number of random variables to consider.

**Applications:** Visualization, Increased efficiency

**Algorithms:** k-Means, feature selection, non-negative matrix factorization, and more...

### Model selection

Comparing, validating and choosing parameters and models.

**Applications:** Improved accuracy via parameter tuning

**Algorithms:** grid search, cross validation, metrics, and more...

### Preprocessing

Feature extraction and normalization.

**Applications:** Transforming input data such as text for use with machine learning algorithms.

**Algorithms:** preprocessing, feature extraction, and more...

<http://scikit-learn.org/stable/>

# 教師あり学習

## 回帰と分類

---

大阪大学 産業科学研究所  
原 聡

# 教師あり学習とは

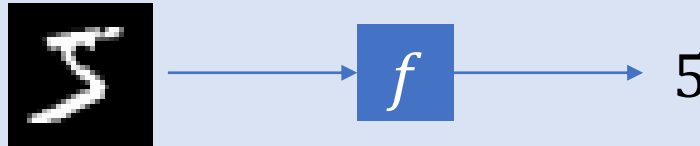
## ■ モデル $f$

- ・ 入力データを受け取り、対応する認識・予測結果を返す関数。

例. 手書き数字認識モデル

入力: 手書き数字画像

出力: 数字



例. 言語翻訳モデル

入力: 日本語文

出力: 英語文



## ■ 教師あり学習 $\hat{=}$ データから $f$ を“自動的”に作る方法

- ・ 入出力関係が単純で既知の場合は、 $f$  は人手で作れる。
- ・ 複雑で未知の関係を表現する  $f$  を人手で作るのは困難。

# 1. 回帰と 機械学習の諸概念

---

# 1. 回帰と機械学習の諸概念

## ■ データの用意

- ・ 入力 $x$ と出力 $y$ を決める。
- ・ 学習データ $D = \{x^{(n)}, y^{(n)}\}_{n=1}^N$ を集める。

## ■ モデルの学習

- ・ モデル候補(関数のクラス)を設定する。
- ・ 損失関数、正則化を設定する。
- ・ データにもっとも適合する関数をモデル候補の中から見つける。
- ・ 最適なハイパーパラメータを見つける。

線形回帰モデル

二乗損失、L2正則化

最適化  
(正規方程式)

Hold Out Validation  
Cross Validation

## ■ モデルの評価

- ・ 予測精度の評価

テストセットによる評価



# モデル $f$ の学習(経験損失最小化)

## ■ 訓練データ

- ・ 入出力データのペア $(x, y)$
- ・ データセット $D = \{x^{(n)}, y^{(n)}\}_{n=1}^N$

$$(x, y) = (\text{ペン}, 5)$$

$$(x, y) = (\text{これはペンです。}, \text{This is a pen.})$$

## ■ 経験損失最小化

- ・ 予測 $f(x)$ と出力 $y$ との乖離を損失 $\ell(y, f(x))$ で評価する。
- ・ 損失(予測の乖離)が最小になるモデル $f$ を求める。

$$\min_f \frac{1}{N} \sum_{n=1}^N \ell(y^{(n)}, f(x^{(n)}))$$

# 1. 回帰と機械学習の諸概念

## ■ データの用意

- ・ 入力 $x$ と出力 $y$ を決める。
- ・ 学習データ $D = \{x^{(n)}, y^{(n)}\}_{n=1}^N$ を集める。

## ■ モデルの学習

- ・ モデル候補(関数のクラス)を設定する。
- ・ 損失関数、正則化を設定する。
- ・ データにもっとも適合する関数をモデル候補の中から見つける。
- ・ 最適なハイパーパラメータを見つける。

線形回帰モデル

二乗損失、L2正則化

最適化  
(正規方程式)

Hold Out Validation  
Cross Validation

## ■ モデルの評価

- ・ 予測精度の評価

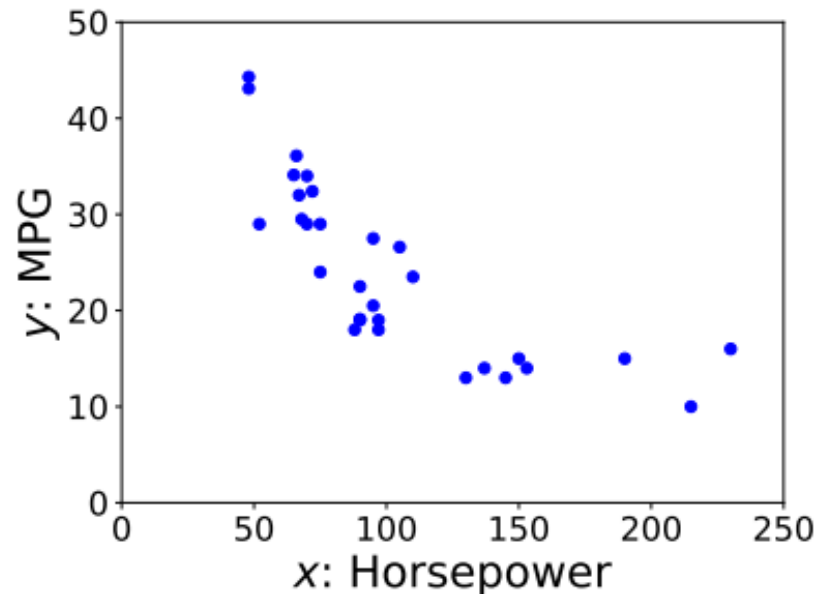
テストセットによる評価

# 【例】自動車の燃費の予測（回帰問題）

- 回帰：予測したい $y$ が実数値の場合
  - ・ 入力 $x$  = 自動車の馬力
  - ・ 出力 $y$  = MPG (Mile Per Gallon)

馬力 $x$	燃費 (MPG) $y$
66	36
230	16
95	27
130	13
$\vdots$	$\vdots$

$$D = \{x^{(n)}, y^{(n)}\}_{n=1}^N$$

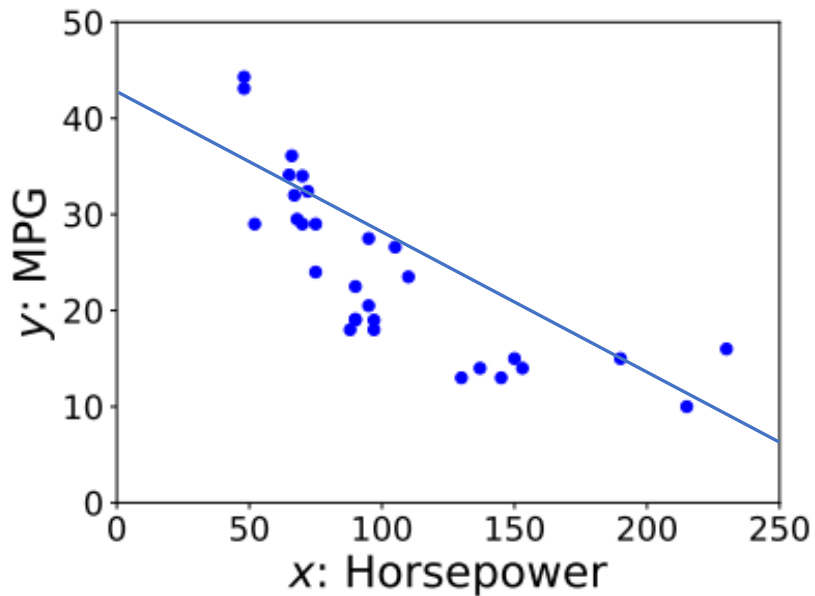


# 線形回帰モデル

## ■ 線形回帰モデル

- $f(x) = ax + b$   
 $= a \times \text{馬力} + b$

## ■ モデルのパラメータ: $a, b$



$a$ : 馬力が1増えた場合に  
MPGがどれだけ増えるか

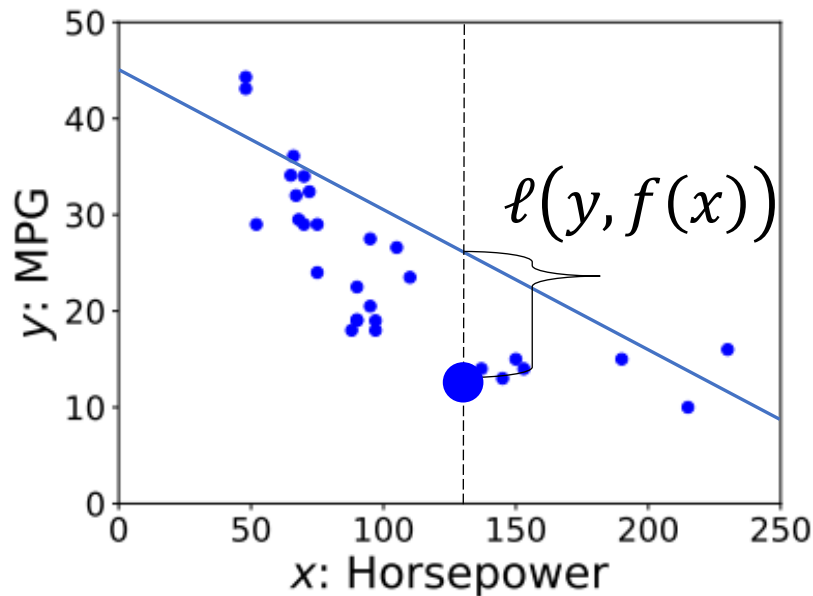
$b$ : デフォルトのMPG

# 線形回帰モデルの学習

- モデルのパラメータ $w$ をデータに適合するように最適化する。
  - ・ 二乗損失

$$\ell(y, f(x)) = \frac{1}{2} (f(x) - y)^2$$

予測値 $f(x)$ と出力 $y$ との差の二乗に応じて大きくなる損失



# 線形回帰モデルの学習

- モデルのパラメータ $w$ をデータに適合するように最適化する。
  - ・ 二乗損失

$$\ell(y, f(x)) = \frac{1}{2} (f(x) - y)^2$$

予測値 $f(x)$ と出力 $y$ との差の二乗に応じて大きくなる損失

- 経験損失最小化 (最小二乗回帰)

$$\min_w \frac{1}{N} \sum_{n=1}^N \frac{1}{2} (ax^{(n)} + b - y^{(n)})^2$$



$$f(x) = ax + b$$

$$\ell(y, f(x)) = \frac{1}{2} (f(x) - y)^2$$

$$\min_f \frac{1}{N} \sum_{n=1}^N \ell(y^{(n)}, f(x^{(n)}))$$

# 最適化(正規方程式)

$$\min_{\theta=(a,b)} L(\theta) = \frac{1}{N} \sum_{n=1}^N \frac{1}{2} (ax^{(n)} + b - y^{(n)})^2$$

## ■ 2変数の二次関数の最小化

### ・ 式の整理

$$ax^{(n)} + b = \begin{bmatrix} x^{(n)} \\ 1 \end{bmatrix}^T \begin{bmatrix} a \\ b \end{bmatrix} = x^{(n)T} \theta$$

  $x^{(n)} \in \mathbb{R}^2 \quad \theta \in \mathbb{R}^2$

$$L(\theta) = \frac{1}{N} \sum_{n=1}^N \frac{1}{2} (x^{(n)T} \theta - y^{(n)})^2$$

# 最適化(正規方程式)

$$\min_{\theta=(a,b)} L(\theta) = \frac{1}{N} \sum_{n=1}^N \frac{1}{2} (ax^{(n)} + b - y^{(n)})^2$$

- 2変数の二次関数の最小化
  - ・ 式の整理

$$\begin{bmatrix} x^{(1)\top} \theta - y^{(1)} \\ x^{(2)\top} \theta - y^{(2)} \\ \vdots \\ x^{(N)\top} \theta - y^{(N)} \end{bmatrix} = \begin{bmatrix} x^{(1)\top} \\ x^{(2)\top} \\ \vdots \\ x^{(N)\top} \end{bmatrix} \theta - \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(N)} \end{bmatrix} = X\theta - Y$$

$X \in \mathbb{R}^{N \times 2} \quad Y \in \mathbb{R}^N$



# 最適化(正規方程式)

$$\min_{\theta=(a,b)} L(\theta) = \frac{1}{N} \sum_{n=1}^N \frac{1}{2} (ax^{(n)} + b - y^{(n)})^2$$

- 2変数の二次関数の最小化
  - ・ 式の整理

$$L(\theta) = \frac{1}{2N} \left\| \begin{bmatrix} x^{(1)\top} \theta - y^{(1)} \\ x^{(2)\top} \theta - y^{(2)} \\ \vdots \\ x^{(N)\top} \theta - y^{(N)} \end{bmatrix} \right\|^2 = \frac{1}{2N} \|X\theta - Y\|^2$$

## 【準備】ベクトルでの微分

- $u, a \in \mathbb{R}^d$  のとき、 $u^\top a$  をベクトル  $u$  について微分する。
- $u^\top a$  を  $u$  の  $i$  番目の要素  $u_i$  について微分する場合を考える。

$$\frac{\partial}{\partial u_i} u^\top a = \frac{\partial}{\partial u_i} \sum_{j=1}^d u_j a_j = a_i$$

- この結果をベクトル状に並べたものがベクトルでの微分。

$$\frac{\partial}{\partial u} u^\top a = \begin{bmatrix} \frac{\partial}{\partial u_1} u^\top a \\ \frac{\partial}{\partial u_2} u^\top a \\ \vdots \end{bmatrix} = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \end{bmatrix} = a$$

## 【準備】ベクトルでの微分

- $v \in \mathbb{R}^d, A \in \mathbb{R}^{d \times d}$  のとき、 $v^\top A v$  をベクトル  $v$  について微分する。
- $v^\top A v$  を  $v$  の  $i$  番目の要素  $v_i$  について微分する場合を考える。

$$\begin{aligned}\frac{\partial}{\partial v_i} v^\top A v &= \frac{\partial}{\partial v_i} \sum_{j=1}^d \sum_{k=1}^d A_{jk} v_j v_k = \sum_{j=1}^d \sum_{k=1}^d A_{jk} \left( \frac{\partial v_j}{\partial v_i} v_k + v_j \frac{\partial v_k}{\partial v_i} \right) \\ &= \sum_{k=1}^d A_{ik} v_k + \sum_{j=1}^d A_{ji} v_j\end{aligned}$$

- この結果をベクトル状に並べたものがベクトルでの微分。

$$\frac{\partial}{\partial v} v^\top A v = \begin{bmatrix} \frac{\partial}{\partial v_1} v^\top A v \\ \frac{\partial}{\partial v_2} v^\top A v \\ \vdots \end{bmatrix} = \sum_{k=1}^d \begin{bmatrix} A_{1k} \\ A_{2k} \\ \vdots \end{bmatrix} v_k + \sum_{j=1}^d \begin{bmatrix} A_{j1} \\ A_{j2} \\ \vdots \end{bmatrix} v_j = A v + A^\top v$$

# 最適化(正規方程式)

$$\min_{\theta=(a,b)} L(\theta) = \frac{1}{2N} \|X\theta - Y\|^2$$

■  $L(\theta)$  を  $\theta$  で微分する。

$$\begin{aligned} \bullet L(\theta) &= \frac{1}{2N} \|X\theta - Y\|^2 = \frac{1}{2N} (X\theta - Y)^\top (X\theta - Y) \\ &= \frac{1}{2N} \theta^\top X^\top X \theta - \frac{1}{N} \theta^\top X^\top Y + \frac{1}{2N} Y^\top Y \end{aligned}$$

$$\begin{aligned} \bullet \frac{\partial L(\theta)}{\partial \theta} &= \frac{1}{2N} \frac{\partial}{\partial \theta} \theta^\top X^\top X \theta - \frac{1}{N} \frac{\partial}{\partial \theta} \theta^\top X^\top Y \\ &= \frac{1}{2N} (X^\top X \theta + X^\top X \theta) - \frac{1}{N} X^\top Y \\ &= \frac{1}{N} X^\top X \theta - \frac{1}{N} X^\top Y \end{aligned}$$

$$\bullet X^\top X \theta = X^\top Y \quad \longrightarrow \quad \hat{\theta} = (X^\top X)^{-1} X^\top Y$$

正規方程式

# 【余談】正規方程式の解法

- $\hat{\theta} = (X^T X)^{-1} X^T Y$  の計算量はどの程度か？
  - $X \in \mathbb{R}^{N \times d}, Y \in \mathbb{R}^N$  とすると、計算量は高々  $O(Nd^2 + d^3)$  。
    - 行列積  $A = X^T X$  の計算量は高々  $O(Nd^2)$ 
      - $(X^T X)_{ij} = \sum_{n=1}^N x_i^{(n)} x_j^{(n)}$
    - 行列ベクトル積  $b = X^T Y$  の計算量は高々  $O(Nd)$ 
      - $(X^T Y)_i = \sum_{n=1}^N x_i^{(n)} y^{(n)}$
    - 逆行列  $C = A^{-1}$  の計算量は高々  $O(d^3)$
    - 行列積  $Ab$  の計算量は高々  $O(d^2)$

# 【余談】正規方程式の解法

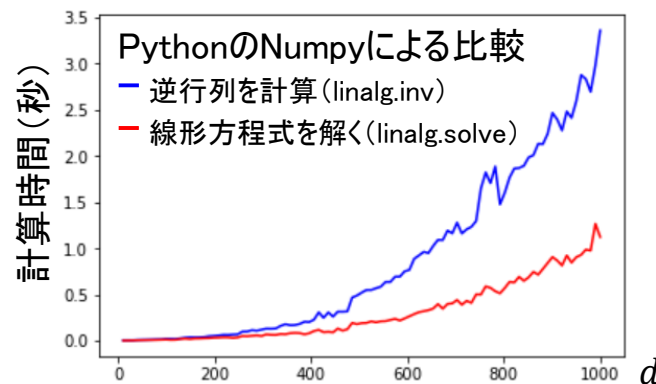
## ■ 実用上は線形方程式 $A\theta = b$ を解く。

- ・ 逆行列  $A^{-1}$  を陽に計算するのは悪手。
- ・ LU分解を使うことで、逆行列を計算せずに  $\hat{\theta}$  を計算できる。
  - LU分解:  $A = LU$  と、下三角行列  $L$  と上三角行列  $U$  の積に分解する。
  - 解きたい問題は  $LU\theta = b$ 
    - 1.  $L\eta = b$  を解く。
    - 2.  $U\theta = \eta$  を解く。

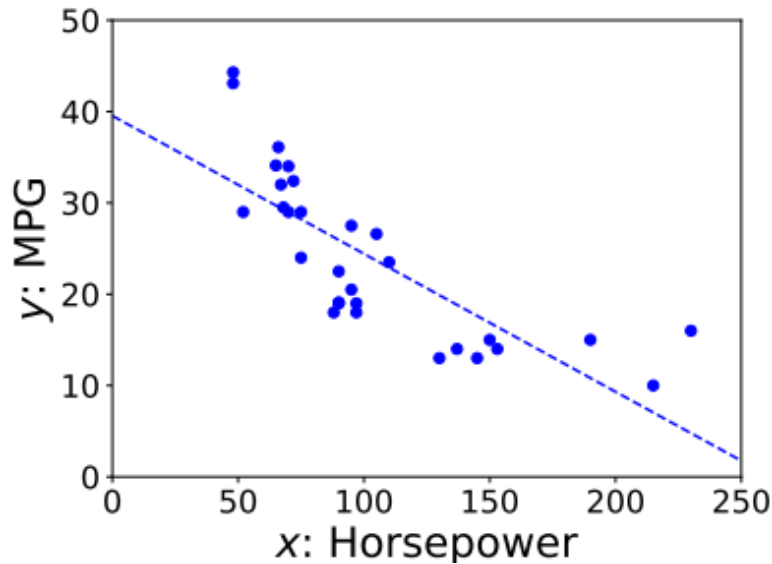
上三角行列、下三角行列の性質を使って効率的に解ける。

## ・ 主要なプログラミング言語では線形方程式の解法が提供されている。

- C, C++: LAPACK の `_gesv` (`cgesv`, `dgesv`, `sgesv`, `zgesv`)
- Python: Numpy の `linalg.solve`
- MATLAB: `linsolve`



## 【結果例】線形回帰モデル



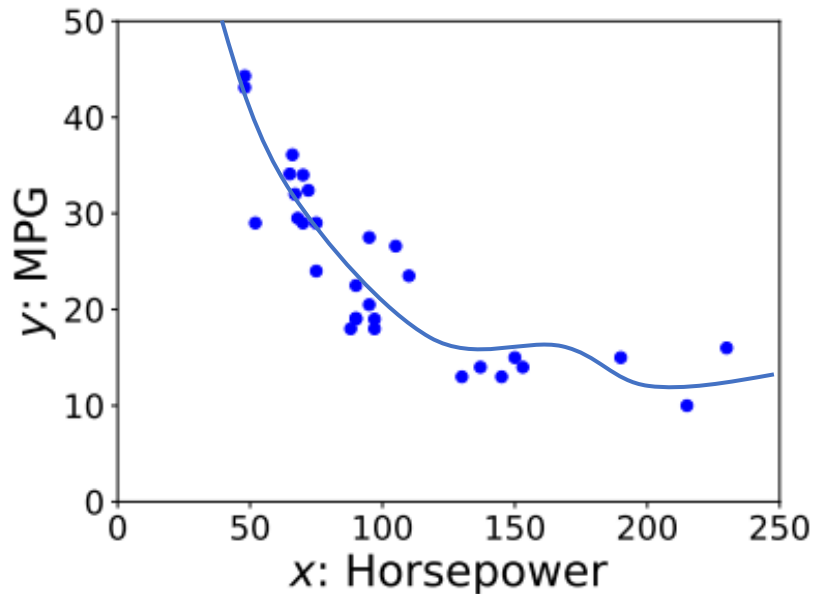
- 不満。あまり良いモデルではない。
  - ・ 馬力がとても大きい場合に、MPGが負になる可能性がある。  
→ 馬力とMPGの関係を直線で表現するのは不適切。

# 線形回帰モデル(多項式モデル)

- 線形回帰モデル( $p$ 次の多項式モデル)

- $f(x) = \sum_{t=1}^p a_t x^t + b$

- モデルのパラメータ:  $a_1, a_2, \dots, a_p, b$



$a_1, a_2, \dots, a_p$ : 多項式の係数

$b$  : デフォルトのMPG



# 【余談】多項式モデルは線形モデルか？

## ■ $p$ 次の多項式モデル

- $f(x) = \sum_{t=1}^p a_t x^t + b$
- $x$ については非線形な関数。
- パラメータ $a_1, a_2, \dots, a_p, b$ については線形な関数。

## ■ 機械学習においてはパラメータの線形性が重要。

- モデルがパラメータについて線形だと、最適化がしやすいことが多い。
  - (より本質的には)最適化の目的関数が凸関数だと最適化しやすい。
  - パラメータについて非線形なモデルは目的関数が凸にならないことが多い。

機械学習の文脈では多項式モデルも  
「パラメータについて線形なモデル」という意味で  
「線形モデル」と呼ぶことがある。

## ■ モデルを使うユーザの視点では $x$ と $y$ の関係が重要。

- 多項式モデルを線形モデルと呼ぶと誤解が生じるので注意。

# 線形回帰モデル(多項式モデル)の学習

## ■ 経験損失最小化(最小二乗回帰)

$$\min_{\theta=(a_1, a_2, \dots, a_p, b)} L(\theta) = \frac{1}{N} \sum_{n=1}^N \frac{1}{2} \left( \sum_{t=1}^p a_t (x^{(n)})^t + b - y^{(n)} \right)^2$$



$$f(x) = \sum_{t=1}^p a_t x^t + b$$

$$\ell(y, f(x)) = \frac{1}{2} (f(x) - y)^2$$

$$\min_f \frac{1}{N} \sum_{n=1}^N \ell(y^{(n)}, f(x^{(n)}))$$

# 線形回帰モデル(多項式モデル)の学習


## ■ 経験損失最小化(最小二乗回帰)

$$\min_{\theta=(a_1, a_2, \dots, a_p, b)} L(\theta) = \frac{1}{N} \sum_{n=1}^N \frac{1}{2} \left( \sum_{t=1}^p a_t (x^{(n)})^t + b - y^{(n)} \right)^2$$

## ■ $p + 1$ 変数の二次関数の最小化

### ・ 式の整理

$$\sum_{t=1}^p w_t (x^{(n)})^t + b = \begin{bmatrix} x_1^{(n)} \\ (x_1^{(n)})^2 \\ \vdots \\ (x_1^{(n)})^p \\ 1 \end{bmatrix}^\top \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_p \\ b \end{bmatrix} = \phi^{(n)\top} \theta$$



$\phi^{(n)} \in \mathbb{R}^{p+1}$        $\theta \in \mathbb{R}^{p+1}$

$$L(\theta) = \sum_{n=1}^N \frac{1}{2} \left( \phi^{(n)\top} \theta - y^{(n)} \right)^2$$

# 線形回帰モデル(多項式モデル)の学習

## ■ 経験損失最小化(最小二乗回帰)

$$\min_{\theta=(a_1, a_2, \dots, a_p, b)} L(\theta) = \frac{1}{N} \sum_{n=1}^N \frac{1}{2} \left( \sum_{t=1}^p a_t (x^{(n)})^t + b - y^{(n)} \right)^2$$

## ■ $p + 1$ 変数の二次関数の最小化

### ・ 式の整理

$$\begin{bmatrix} \phi^{(1)\top} \theta - y^{(1)} \\ \phi^{(2)\top} \theta - y^{(2)} \\ \vdots \\ \phi^{(N)\top} \theta - y^{(N)} \end{bmatrix} = \begin{bmatrix} \phi^{(1)\top} \\ \phi^{(2)\top} \\ \vdots \\ \phi^{(N)\top} \end{bmatrix} \theta - \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(N)} \end{bmatrix} = \Phi \theta - Y$$

$$\Phi \in \mathbb{R}^{N \times (p+1)}$$


# 線形回帰モデル(多項式モデル)の学習

## ■ 経験損失最小化(最小二乗回帰)

$$\min_{\theta=(a_1, a_2, \dots, a_p, b)} L(\theta) = \frac{1}{N} \sum_{n=1}^N \frac{1}{2} \left( \sum_{t=1}^p a_t (x^{(n)})^t + b - y^{(n)} \right)^2$$

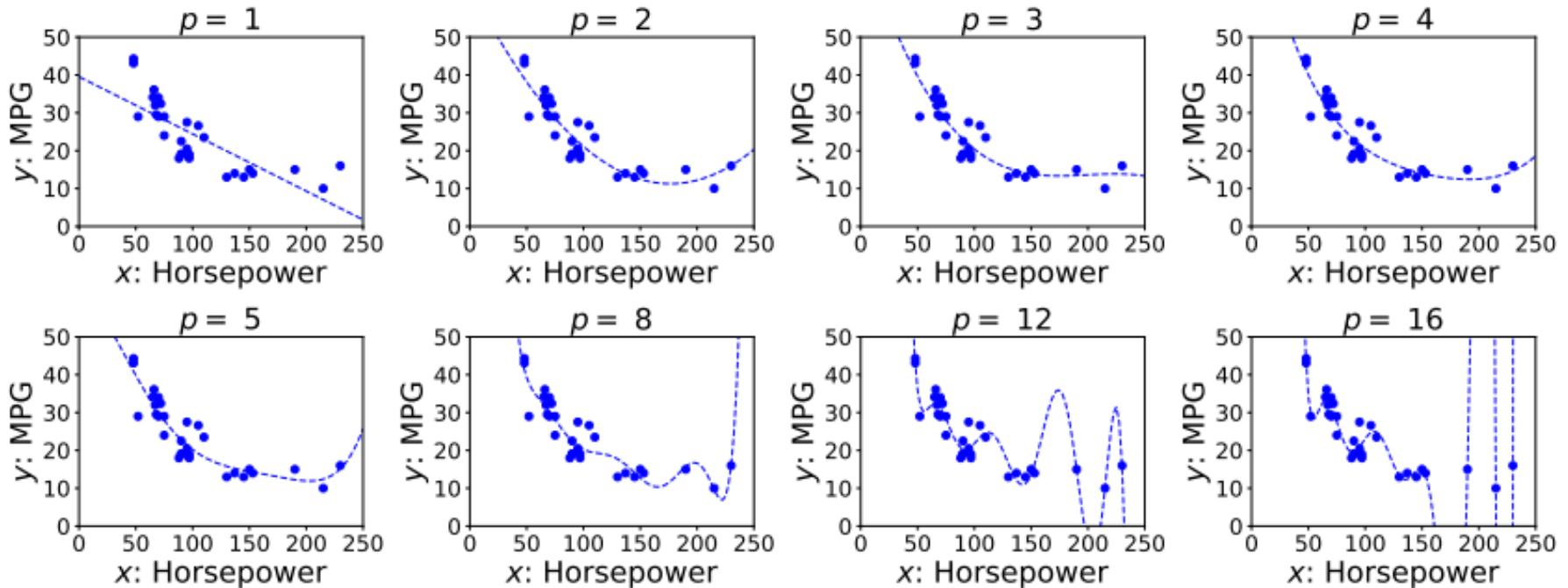
## ■ $p + 1$ 変数の二次関数の最小化

$$L(\theta) = \frac{1}{2N} \left\| \begin{bmatrix} \phi^{(1)\top} \theta - y^{(1)} \\ \phi^{(2)\top} \theta - y^{(2)} \\ \vdots \\ \phi^{(N)\top} \theta - y^{(N)} \end{bmatrix} \right\|^2 = \frac{1}{2N} \|\Phi\theta - Y\|^2$$

 
$$\hat{\theta} = (\Phi^\top \Phi)^{-1} \Phi^\top Y$$

## 【結果例】線形回帰モデル(多項式モデル)

- 多項式の次数 $p$ を増やすと、馬力とMPGの間の関係を複雑な曲線で表現できる。



# 1. 回帰と機械学習の諸概念

## ■ データの用意

- ・ 入力 $x$ と出力 $y$ を決める。
- ・ 学習データ $D = \{x^{(n)}, y^{(n)}\}_{n=1}^N$ を集める。

## ■ モデルの学習

- ・ モデル候補(関数のクラス)を設定する。
- ・ 損失関数、正則化を設定する。
- ・ データにもっとも適合する関数をモデル候補の中から見つける。
- ・ 最適なハイパーパラメータを見つける。

線形回帰モデル

二乗損失、L2正則化

最適化  
(正規方程式)

Hold Out Validation  
Cross Validation

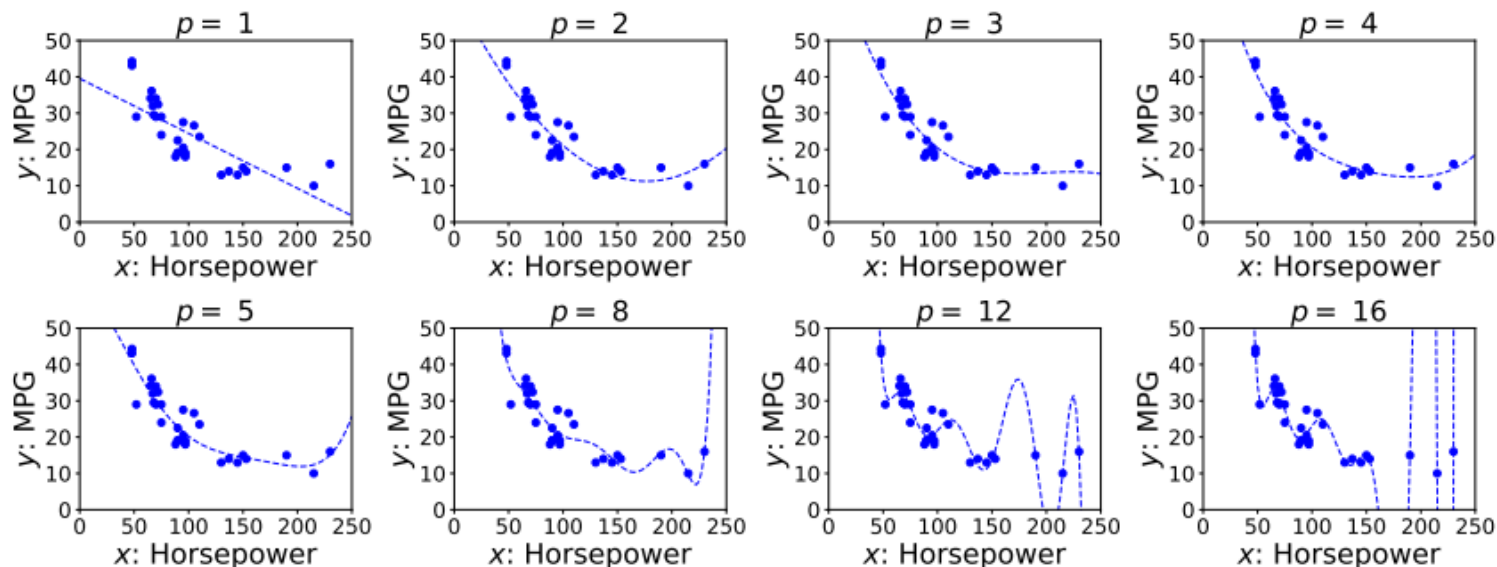
## ■ モデルの評価

- ・ 予測精度の評価

テストセットによる評価

# モデルの評価

- 多項式の次数 $p$ を変えると、いろいろなモデルが作れる。



- Q. どれが一番良いモデルだろうか？

- ・ 目視による直感的な評価の場合
  - $p = 3, 4$ では馬力200以降でMPGが落ちる傾向があり、妥当そうに見える。
  - $p \geq 8$  大きな馬力についてMPGの予測の変動が大きく不自然。

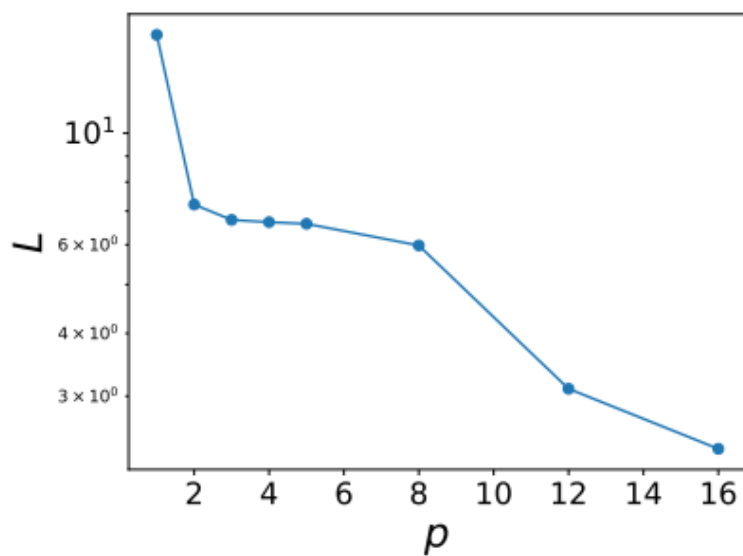
→ これらの直感を定量化したい。



# モデルの評価

## ■ 評価方法の一案

- モデルの学習ではデータとモデルの誤差  $L(\theta) = \frac{1}{2N} \|\Phi\theta - Y\|^2$  を最小にするパラメータ  $\hat{\theta}$  を求めた。
- 誤差  $L(\hat{\theta})$  が小さいモデルが良いモデルのはず。



$p$ が増えるほど $L(\hat{\theta})$ は減少。

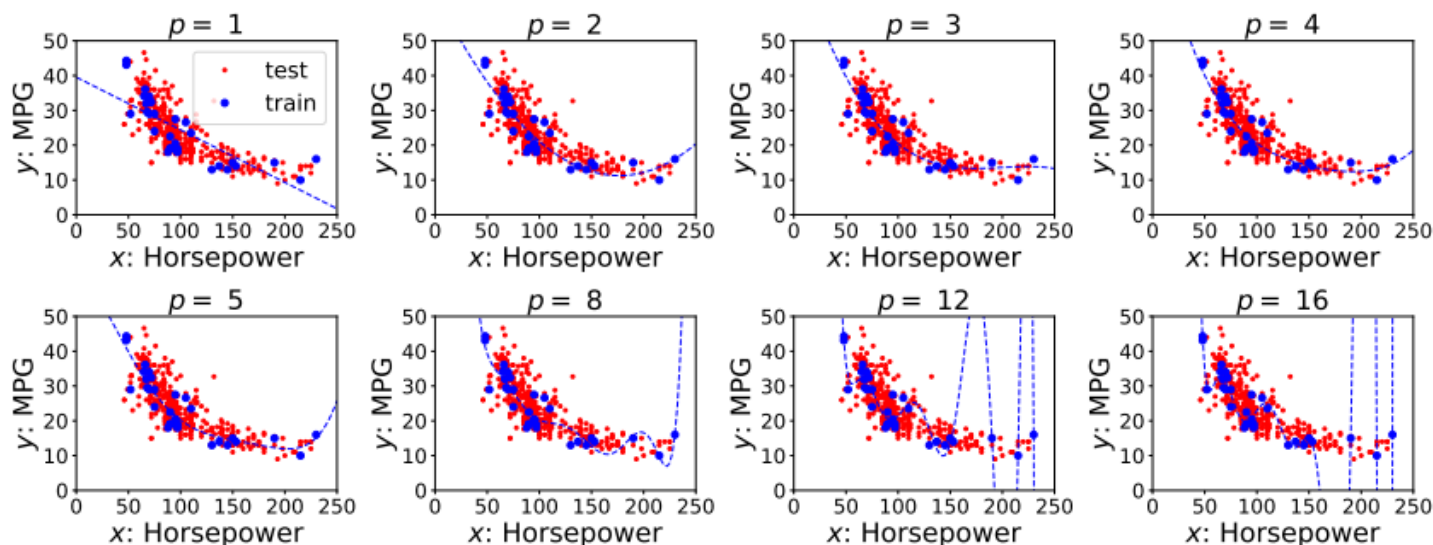
$p \geq 8$ で場合に誤差 $L(\hat{\theta})$ は大きく減少。

$p$ は大きければ大きいほど良い？

# モデルの評価

## ■ 汎化

- ・ 未知のデータに対しても良い予測ができるモデルが良いモデル。
- ・ ただのデータ丸暗記は駄目。
  - モデルの学習に使ったデータを丸暗記すれば、学習データ中のデータについては誤差なしで予測できる。
  - しかし、他の馬力に対するMPGは丸暗記では予測できない。



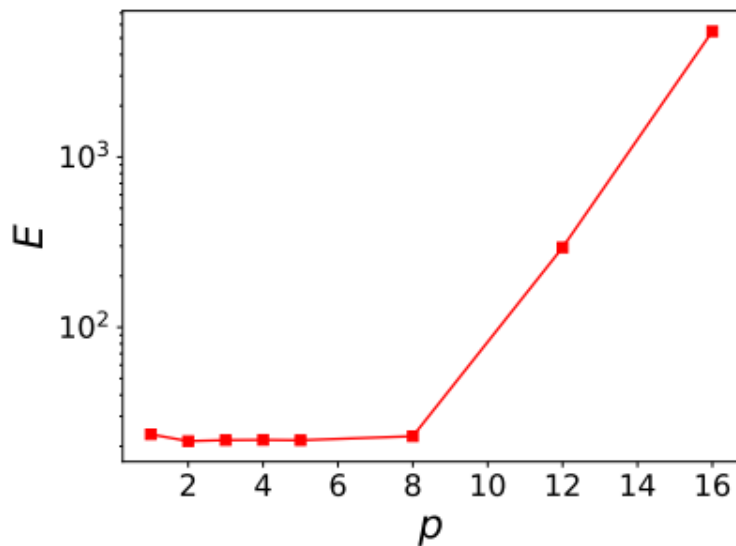
● 学習データ  $D = \{x^{(n)}, y^{(n)}\}_{n=1}^N$

● テストデータ (未知データ)  $D_{\text{test}} = \{x_{\text{test}}^{(m)}, y_{\text{test}}^{(m)}\}_{m=1}^M$

# テストデータによるモデルの評価

## ■ テストデータでの二乗誤差による評価

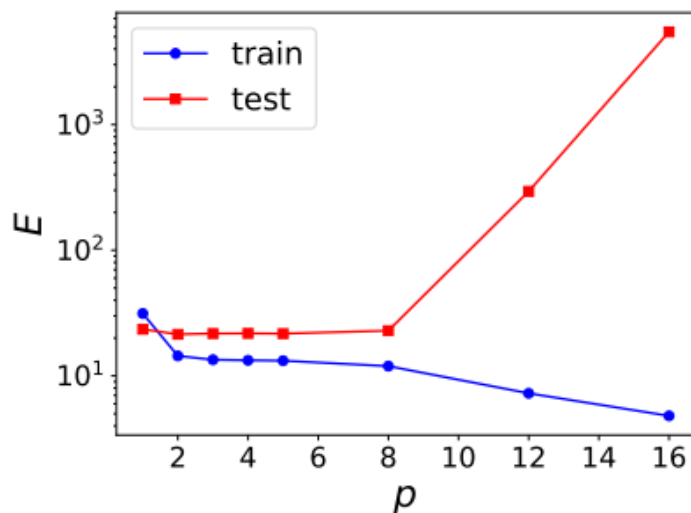
$$E_{\text{test}}(\hat{\theta}) = \frac{1}{M} \sum_{m=1}^M \left( \phi_{\text{test}}^{(m)\top} \hat{\theta} - y_{\text{test}}^{(m)} \right)^2$$



直感は確かに妥当だった。

- $p = 2 \sim 4$  のモデルが二乗誤差が小さい。
- $p \geq 8$  のモデルの二乗誤差は大きい。

# 過学習



$$E_{\text{train}}(\hat{\theta}) = \frac{1}{N} \sum_{n=1}^N \left( \phi^{(n)\top} \hat{\theta} - y^{(n)} \right)^2$$

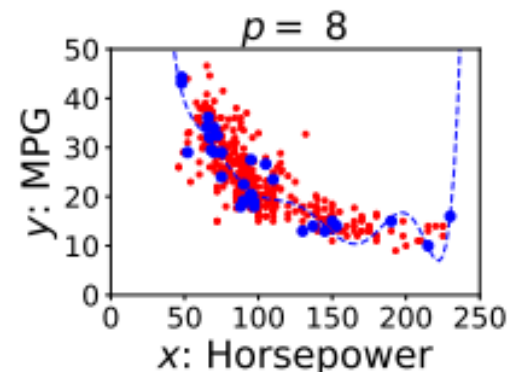
- 学習データとテストデータの二乗誤差には乖離がある。
  - ・ 学習データの誤差は $p$ の増加に従って単調に減少。
  - ・ テストデータの誤差は $p = 2 \sim 4$ で小。
- 過学習
  - ・ モデルが学習データの誤差を減らすことに特化しすぎて、未知のデータに対する予測の性能が低下している状態のこと。

# 過学習

## ■ モデルを見てみる。

- $p = 8$ の場合

- $$f(x) = 1630 - 28845x + 221392x^2 - 935634x^3 + 2371075x^4 - 3690211x^5 + 345031x^6 - 1776103x^7 + 386401x^8$$



- $[0, 50]$ の範囲の出力を計算するのに、係数は1万を超える大きな値を取っている。
- パラメータを大きい値に設定して多項式を複雑にすることで、学習データでの誤差が小さくなるようにしている。

# 1. 回帰と機械学習の諸概念

## ■ データの用意

- ・ 入力 $x$ と出力 $y$ を決める。
- ・ 学習データ $D = \{x^{(n)}, y^{(n)}\}_{n=1}^N$ を集める。

## ■ モデルの学習

- ・ モデル候補(関数のクラス)を設定する。
- ・ 損失関数、正則化を設定する。
- ・ データにもっとも適合する関数をモデル候補の中から見つける。
- ・ 最適なハイパーパラメータを見つける。

線形回帰モデル

二乗損失、L2正則化

最適化  
(正規方程式)

Hold Out Validation  
Cross Validation

## ■ モデルの評価

- ・ 予測精度の評価

テストセットによる評価

# 正則化: 過学習を抑制する方法の一つ

## ■ 過学習の一要因

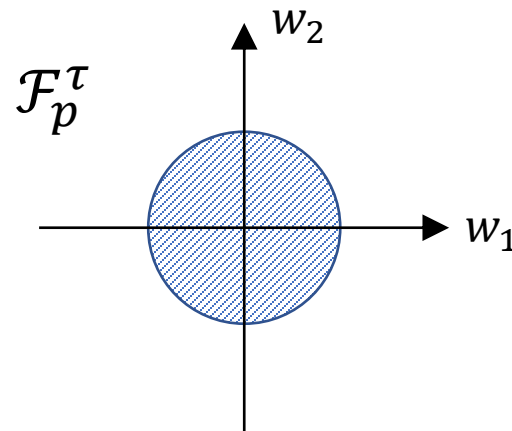
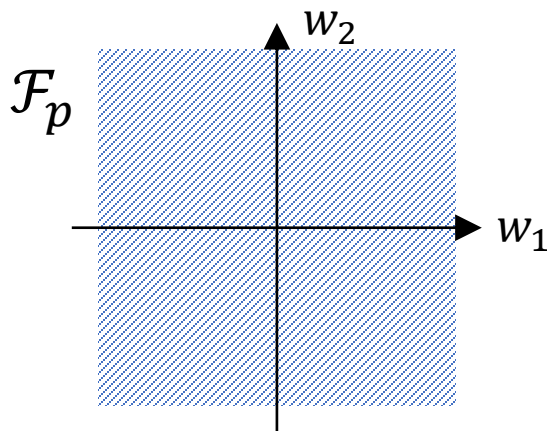
- ・ 学習時にパラメータを自由な値に設定できる。
  - 関数のクラスが大きすぎる。

$$\mathcal{F}_p = \left\{ \sum_{t=1}^p a_t x^t + b \mid a_1, a_2, \dots, a_p, b \in \mathbb{R}^{p+1} \right\}$$

## ■ 過学習を抑制するために

- ・ 関数のクラスを小さくする。

$$\mathcal{F}_p^\tau = \left\{ \sum_{t=1}^p a_t x^t + b \mid \sum_{t=1}^p w_t^2 \leq \tau \right\}$$



# 正則化: 過学習を抑制する方法の一つ

$$\min_{\theta=(a_1, a_2, \dots, a_p, b)} \frac{1}{2N} \|\Phi\theta - Y\|^2 \quad \text{subject to} \quad \sum_{t=1}^p a_t^2 \leq \tau$$

## ■ 制約をペナルティとして目的関数に追加する。

### ・ ラグランジュ乗数法

- 適当な  $\lambda \geq 0$  を設定すると、制約付きの問題と等価。

$$\min_{\theta} \frac{1}{2N} \|\Phi\theta - Y\|^2 + \frac{\lambda}{2} (\sum_{t=1}^p a_t^2 - \tau)$$

### ・ $\lambda$ をペナルティの重みとして設定する。

-  $\tau$  を明示的に設定する代わりに  $\lambda$  を設定する。

$$\min_{\theta} \frac{1}{2N} \|\Phi\theta - Y\|^2 + \frac{\lambda}{2} \sum_{t=1}^p a_t^2$$



# 最適化

$$\min_{\theta=(a_1, a_2, \dots, a_p, b)} L(\theta) = \frac{1}{2N} \|\Phi\theta - Y\|^2 + \frac{\lambda}{2} \sum_{t=1}^p a_t^2$$

## ■ 正則化項の整理

$$\cdot \sum_{t=1}^p a_t^2 = \underbrace{\begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_p \\ b \end{bmatrix}}_{\theta}^T \underbrace{\begin{bmatrix} 1 & & & & \\ & 1 & & & \\ & & \ddots & & \\ & & & 1 & \\ & & & & 0 \end{bmatrix}}_{J = \text{diag}(1, 1, \dots, 1, 0)} \underbrace{\begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_p \\ b \end{bmatrix}}_{\theta} = \theta^T J \theta$$

$$\min_{\theta} L(\theta) = \frac{1}{2N} \|\Phi\theta - Y\|^2 + \frac{\lambda}{2} \theta^T J \theta$$

# 最適化

$$\min_{\theta} L(\theta) = \frac{1}{2N} \|\Phi\theta - Y\|^2 + \frac{\lambda}{2} \theta^\top J \theta$$

■  $L(\theta)$  を  $\theta$  で微分する。

$$\begin{aligned} \cdot \frac{\partial L(\theta)}{\partial \theta} &= \frac{1}{2N} \frac{\partial}{\partial \theta} \theta^\top \Phi^\top \Phi \theta - \frac{1}{N} \frac{\partial}{\partial \theta} \theta^\top \Phi^\top Y + \frac{\lambda}{2} \frac{\partial}{\partial \theta} \theta^\top J \theta \\ &= \frac{1}{2N} (\Phi^\top \Phi \theta + \Phi^\top \Phi \theta) - \frac{1}{N} \Phi^\top Y + \lambda J \theta \\ &= \left( \frac{1}{N} \Phi^\top \Phi + \lambda J \right) \theta - \frac{1}{N} \Phi^\top Y \end{aligned}$$

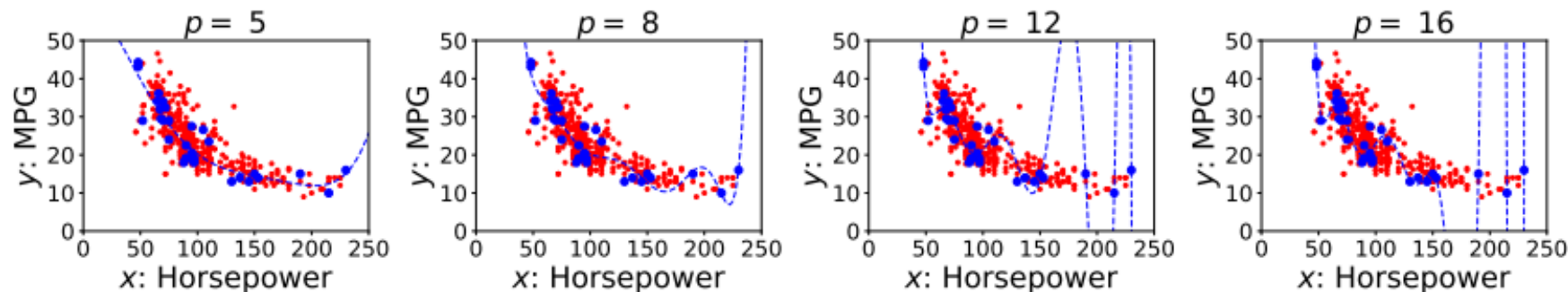
$$\cdot \left( \frac{1}{N} \Phi^\top \Phi + \lambda J \right) \theta - \frac{1}{N} \Phi^\top Y = 0$$



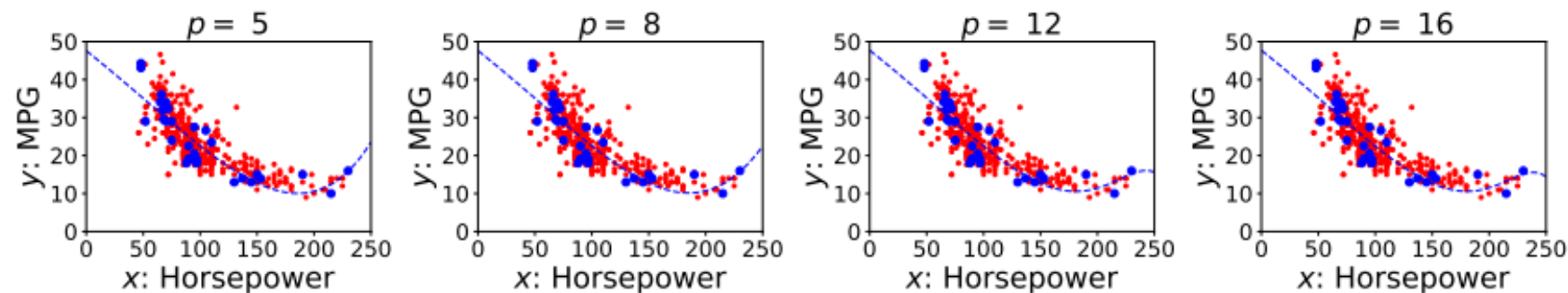
$$\hat{\theta} = (\Phi^\top \Phi + N\lambda J)^{-1} \Phi^\top Y$$

# 結果例

## ■ 正則化なし( $\lambda = 0$ )の場合



## ■ $\lambda = 0.001$ に設定した場合



# 1. 回帰と機械学習の諸概念

## ■ データの用意

- ・ 入力 $x$ と出力 $y$ を決める。
- ・ 学習データ $D = \{x^{(n)}, y^{(n)}\}_{n=1}^N$ を集める。

## ■ モデルの学習

- ・ モデル候補(関数のクラス)を設定する。
- ・ 損失関数、正則化を設定する。
- ・ データにもっとも適合する関数をモデル候補の中から見つける。
- ・ 最適なハイパーパラメータを見つける。

線形回帰モデル

二乗損失、L2正則化

最適化  
(正規方程式)

Hold Out Validation  
Cross Validation

## ■ モデルの評価

- ・ 予測精度の評価

テストセットによる評価

# ハイパーパラメータの選択

---

## ■ ここまでのまとめ

- ・ 多項式モデルを使うと「馬力・MPG間の関係」がうまく表現できる。
- ・ そのためには「多項式の次数 $p$ 」と「正則化の大きさ $\lambda$ 」を適切に設定する必要がある。
  - $p$ や $\lambda$ はモデルのクラスを決めるハイパーパラメータ
  - 設定を間違えると過学習が起きて、予測に使えないモデルになってしまう。

## ■ ハイパーパラメータを決める適当な方法が必要。

- ・ テストデータがあれば、テストデータ上での誤差を評価して、誤差が小さくなる $p$ や $\lambda$ を決めれば良い。
- ・ 問題：モデルの学習段階ではテストデータ（未知データ）を持っていない。

# ホールドアウト検証 (hold out validation)

- 学習データを学習用とテスト用に分ける。

馬力 $x$	燃費 (MPG) $y$
66	36
230	16
95	27
130	13
$\vdots$	$\vdots$

馬力 $x$	燃費 (MPG) $y$
66	36
230	16
$\vdots$	$\vdots$

馬力 $x$	燃費 (MPG) $y$
95	27
130	13
$\vdots$	$\vdots$

## 学習用

色々な $p$ 及び $\lambda$ について  
 $\hat{\theta}$ を推定する。

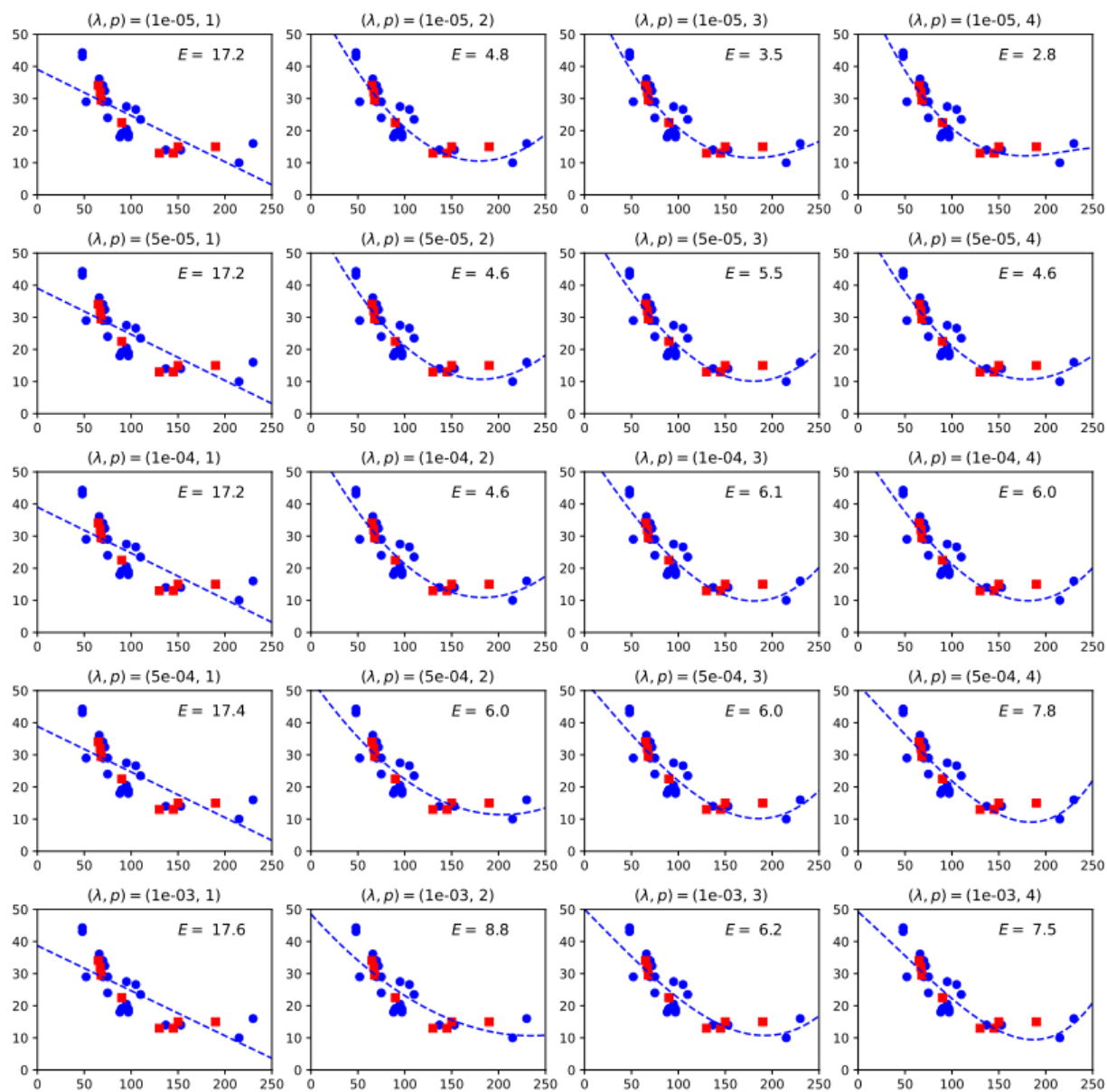
## テスト用

$E_{\text{test}}(\hat{\theta})$ を計算する。



$E_{\text{test}}(\hat{\theta})$ が最小になった  
 $p$ 及び $\lambda$ を採用する。

# 結果例



$p = 4, \lambda = 10^{-5}$ で  
 $E_{\text{test}}(\hat{\theta})$ が最小



$p = 4, \lambda = 10^{-5}$ を採用

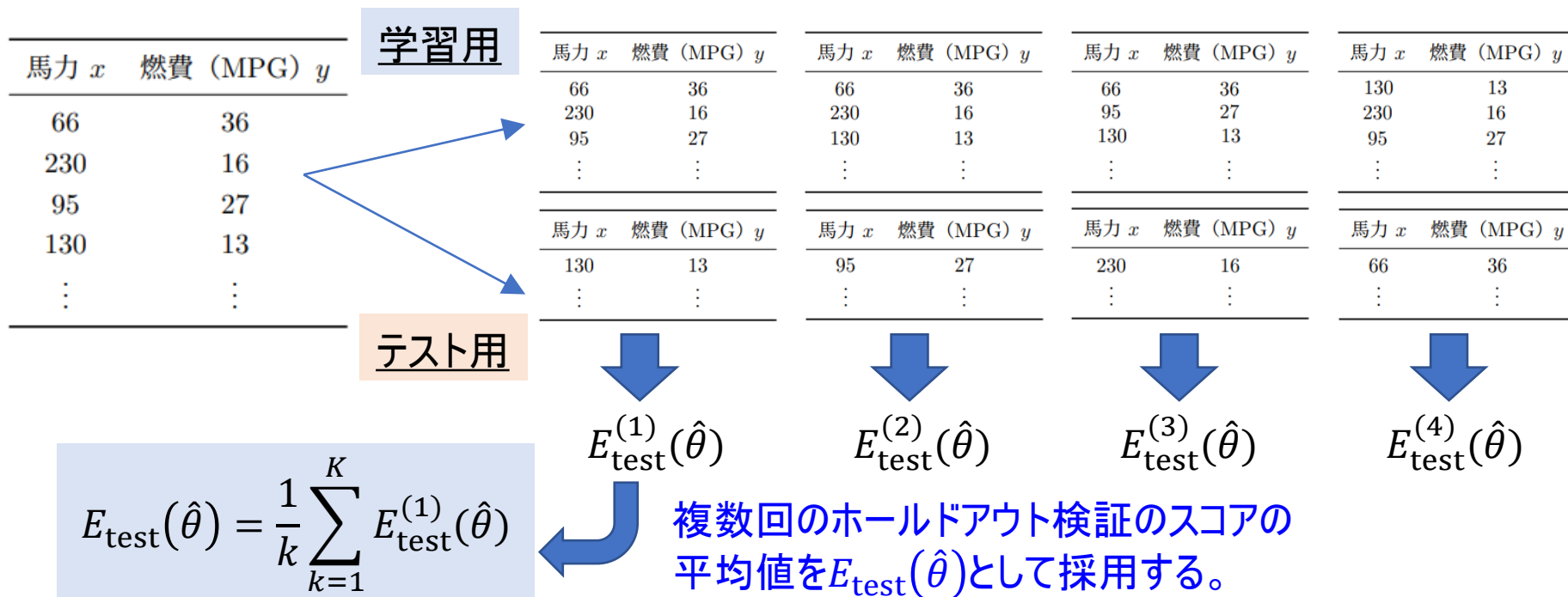
# 交差検証 (Cross Validation)

## ■ ホールドアウト検証の弱点

- 学習データの個数が少ない場合に、学習用に使えるデータの個数が大幅に減ってしまう。

## ■ 交差検証

- 学習用/テスト用の分割の複数通りについてホールドアウト検証

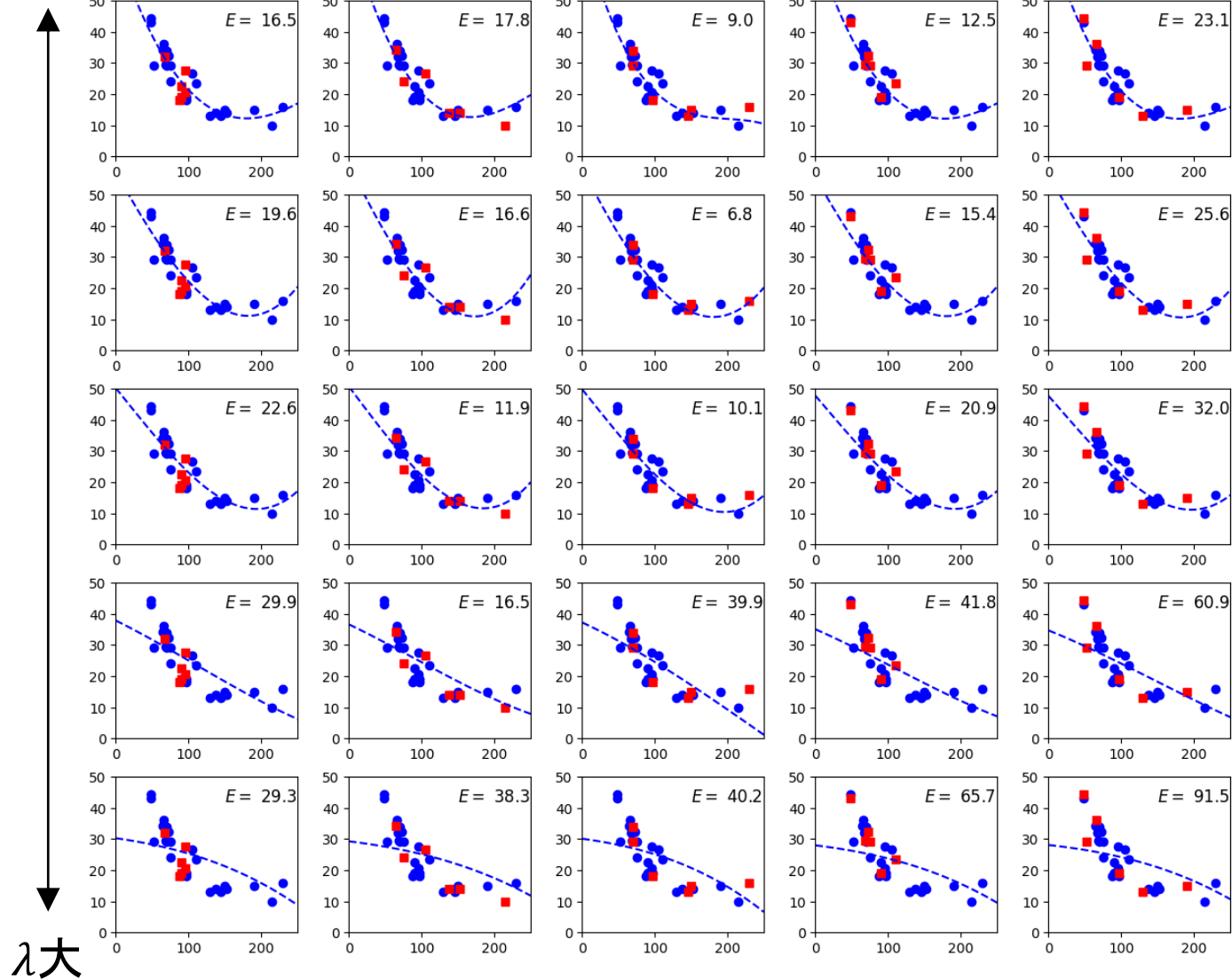




# 結果例 ( $p = 3$ の場合)

分割5通り

$\lambda$ 小



最小

$$E_{\text{test}}(\hat{\theta}) = 15.8$$

$$E_{\text{test}}(\hat{\theta}) = 16.8$$

$$E_{\text{test}}(\hat{\theta}) = 19.5$$

$$E_{\text{test}}(\hat{\theta}) = 37.8$$

$$E_{\text{test}}(\hat{\theta}) = 53.0$$

# 結果の比較

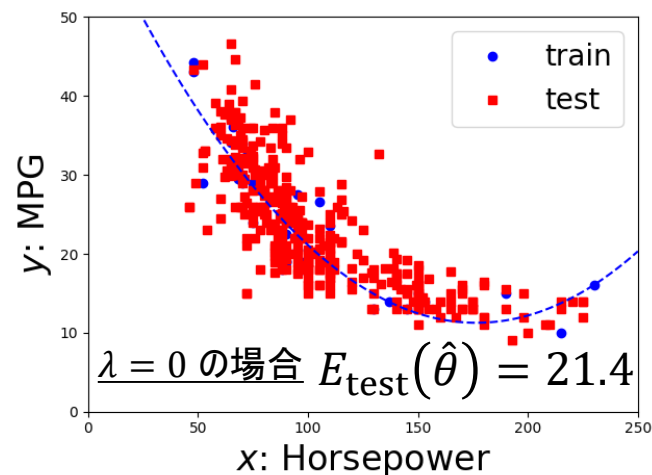
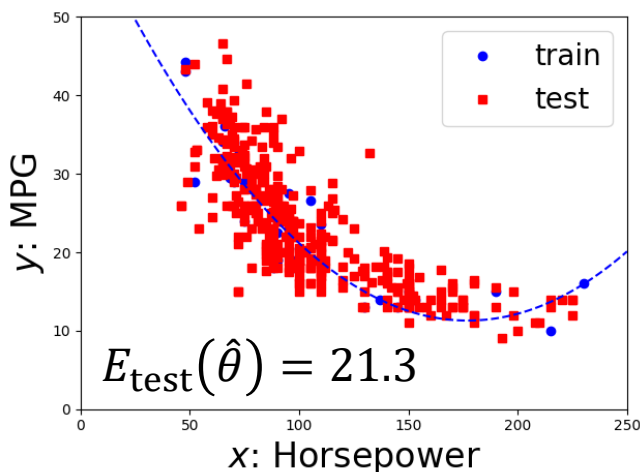
## ■ ハイパーパラメータの候補

- $p$ の候補:  $\{1, 2, 3\}$
- $\lambda$ の候補:  $\{10^{-i}\}_{i=1}^5$

## ■ Cross Validationによるハイパーパラメータの決定

- $p = 2, \lambda = 10^{-5}$

## ■ テスト誤差



# 1. 回帰と機械学習の諸概念

## ■ データの用意

- ・ 入力 $x$ と出力 $y$ を決める。
- ・ 学習データ $D = \{x^{(n)}, y^{(n)}\}_{n=1}^N$ を集める。

## ■ モデルの学習

- ・ モデル候補(関数のクラス)を設定する。
- ・ 損失関数、正則化を設定する。
- ・ データにもっとも適合する関数をモデル候補の中から見つける。
- ・ 最適なハイパーパラメータを見つける。

線形回帰モデル

二乗損失、L2正則化

最適化  
(正規方程式)

Hold Out Validation  
Cross Validation

## ■ モデルの評価

- ・ 予測精度の評価

テストセットによる評価

## 2. 分類と代理損失と 勾配降下法

---

## 2. 分類と代理損失と勾配降下法

### ■ データの用意

- ・ 入力 $x$ と出力 $y$ を決める。
- ・ 学習データ $D = \{x^{(n)}, y^{(n)}\}_{n=1}^N$ を集める。

### ■ モデルの学習

- ・ モデル候補(関数のクラス)を設定する。
- ・ 損失関数、正則化を設定する。
- ・ データにもっとも適合する関数をモデル候補の中から見つける。
- ・ 最適なハイパーパラメータを見つける。

線形分類モデル

0-1 損失  
ロジスティック損失

最適化  
(勾配降下法)

### ■ モデルの評価

- ・ 予測精度の評価

Accuracy, Precision,  
Recall, F値, AUROC

## 【例】文章トピックの予測（分類問題）

- 分類：予測したい出力がカテゴリ値の場合
  - ・ 入力：レビュー文
  - ・ 出力：レビュー文のトピック（京都観光 or グルメ）

トピック：京都観光

※ 架空の例です。

秋の京都はとても美しい。京都に比べて大阪には秋の風物詩が欠けている。京都と大阪、これほど近いのに景観が大きく異なるのはとても興味深い。

トピック：グルメ

※ 架空の例です。

ラーメンと言えば京都の天下一品総本店だろう。他の店舗と違ってメニューが多彩で色々な味が楽しめる。特に本店でしか味わえない限定メニューはとても美味しい。

# 【例】文章トピックの予測(分類問題)

## ■ 分類: 予測したい出力がカテゴリ値の場合

- ・ 入力: レビュー文
- ・ 出力: レビュー文のトピック(京都観光 or グルメ)

トピック: 京都観光

※ 架空の例です。

秋の**京**都はとても美しい。**京**都に比べて大阪には秋の風物詩が欠けている。**京**都と大阪、これほど近いのに景観が大きく異なるのはとても興**味**深い。

$$y = 1$$

$$x_1 = n_{\text{京}} = 3$$

$$x_2 = n_{\text{味}} = 1$$

トピック: グルメ

※ 架空の例です。

ラーメンと言えば**京**都の天下一品総本店だろう。他の店舗と違ってメニューが多彩で色々な**味**が楽しめる。特に本店でしか**味**わえない限定メニューはとても美**味**しい。

$$y = -1$$

$$x_1 = n_{\text{京}} = 1$$

$$x_2 = n_{\text{味}} = 3$$

## ・ データの数値化

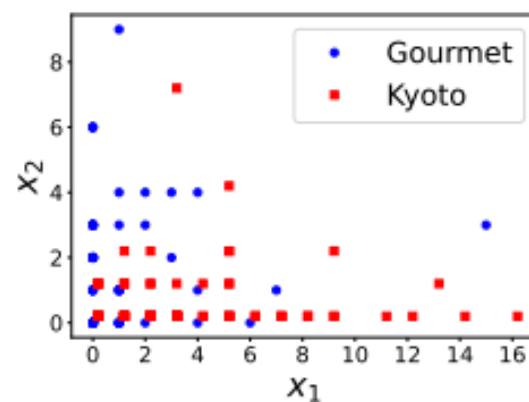
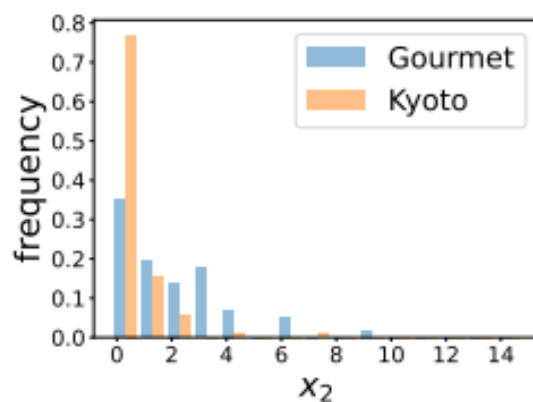
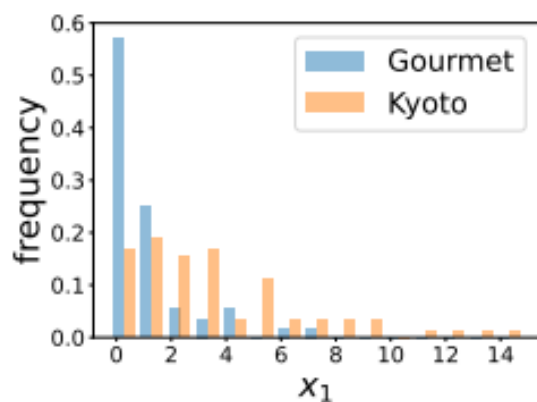
- 入力:  $x_1$  = 「京」の出現回数,  $x_2$  = 「味」の出現回数
- 出力: 京都観光  $y = 1$ , グルメ  $y = -1$

# 【例】文章トピックの予測（分類問題）

## ■ トピック予測データ

- ・ グルメ ( $y = -1$ ) : 57件
- ・ 京都観光 ( $y = 1$ ) : 91件

<http://nlp.ist.i.kyoto-u.ac.jp/kuntt/> より



$x_1, x_2$ には分類に有用な情報が含まれていそう



# 線形分類モデル

## ■ 線形分類モデル

$$\bullet f_{\theta}(x) = \sum_{i=1}^d a_i x_i + b = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \\ 1 \end{bmatrix}^{\top} \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_d \\ b \end{bmatrix} = x^{\top} \theta$$

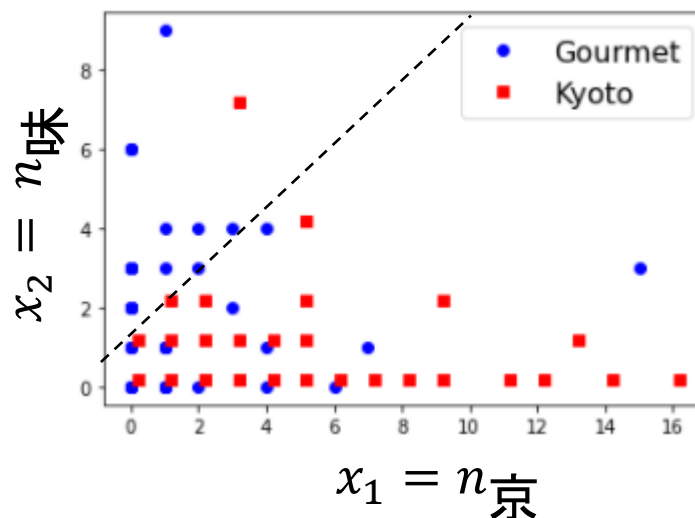
$x \in \mathbb{R}^{d+1}$        $\theta \in \mathbb{R}^{d+1}$

## ・ 符号による分類

$$\text{sign}(f_{\theta}(x)) = \begin{cases} 1, & \text{if } f_{\theta}(x) > 0 \\ -1, & \text{if } f_{\theta}(x) \leq 0 \end{cases}$$

## ■ 例：文章トピックの予測

- $x_1 = n_{\text{京}}$
- $x_2 = n_{\text{味}}$



## 2. 分類と代理損失と勾配降下法

### ■ データの用意

- ・ 入力 $x$ と出力 $y$ を決める。
- ・ 学習データ $D = \{x^{(n)}, y^{(n)}\}_{n=1}^N$ を集める。

### ■ モデルの学習

- ・ モデル候補(関数のクラス)を設定する。
- ・ 損失関数、正則化を設定する。
- ・ データにもっとも適合する関数をモデル候補の中から見つける。
- ・ 最適なハイパーパラメータを見つける。

線形分類モデル

0-1損失  
ロジスティック損失

最適化  
(勾配降下法)

### ■ モデルの評価

- ・ 予測精度の評価

Accuracy, Precision,  
Recall, F値, AUROC

# 問題設定

## ■ 指示関数

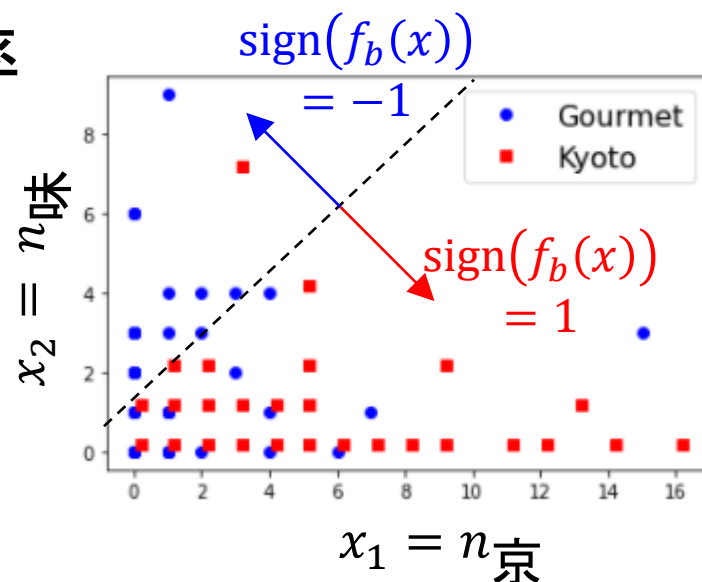
$$\bullet 1[a] = \begin{cases} 1, & \text{if } a = \text{True} \\ 0, & \text{if } a = \text{False} \end{cases}$$

## ■ 0-1損失

$$\bullet \ell_{01}(y, f(x)) = 1[y \neq \text{sign}(f(x))] = \begin{cases} 1, & \text{if } y \neq \text{sign}(f(x)) \\ 0, & \text{if } y = \text{sign}(f(x)) \end{cases}$$

## ■ モデルの性能評価指標：誤分類率

$$E_{\text{test}}(f) = \frac{1}{M} \sum_{m=1}^M \ell_{01}(y_{\text{test}}^{(m)}, f(x_{\text{test}}^{(m)}))$$



# 【余談】性能限界

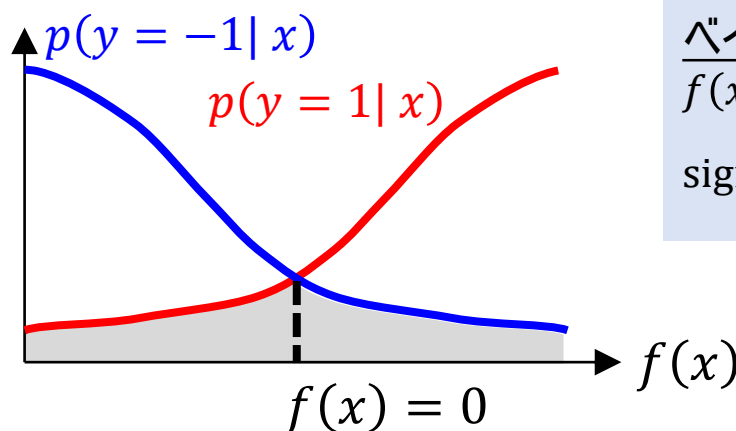
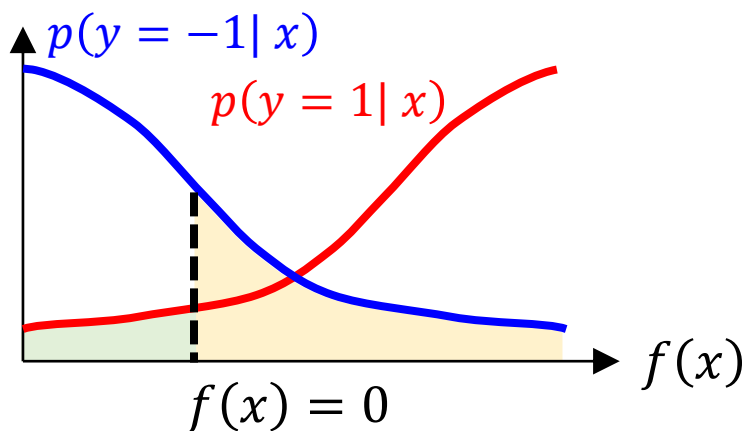
- 常に誤分類率ゼロが達成できるわけではない。
  - ・ 原理的に達成可能な誤分類率は問題(データ分布)に依存する。
- ベイズ誤差: 原理的に達成可能な最小の誤分類率

$$\mathbb{E}_{p(x,y)}[\ell_{01}(y, f(x))]$$

$$= \int 1[f(x) > 0]p(y = -1|x)p(x)dx + \int 1[f(x) \leq 0]p(y = 1|x)p(x)dx$$

$$\geq \int \min_{y \in \{-1,1\}} \ell_{01}(y, f(x))p(x)dx$$

ベイズ誤差



ベイズルール

$$f(x) = \text{sign}\left(p(y = 1|x) - \frac{1}{2}\right)$$

# 最適化: 代理損失の利用

- 目的関数を最適化しやすいものに置き換える。

$$E_{\text{train}}(f) := \frac{1}{N} \sum_{n=1}^N \ell_{01}(y^{(n)}, f(x^{(n)}))$$

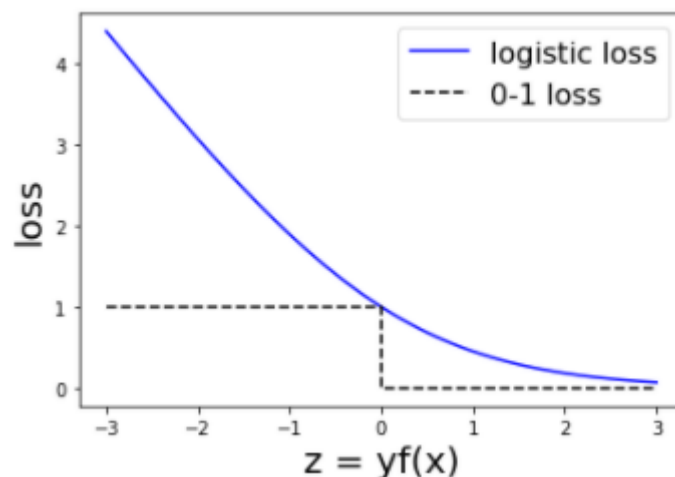
$$L_{\text{train}}(f) := \frac{1}{N} \sum_{n=1}^N \ell(y^{(n)}, f(x^{(n)}))$$

$\ell_{01}(y, f_b(x))$   
 $\rightarrow \ell(y, f_b(x))$ : 代理損失

- 代替損失の代表例: ロジスティック損失

$$\begin{aligned} \ell_{\log}(y, f_b(x)) \\ = \log(1 + \exp(-yf_b(x))) \end{aligned}$$

- 連続かつ微分可能な関数なので連続最適化の技術が使える。
- 損失関数として“良い性質”がある。



# ロジスティック損失の利点

---

$$\ell_{\log}(y, f(x)) = \log(1 + \exp(-yf(x)))$$

## ■ 利点1

- ・ 連続かつ微分可能な関数なので連続最適化の技術が使える。  
→ 勾配降下法(最急降下法)による最適化

## ■ 利点2

- ・ 損失関数として“良い性質”がある。  
→ 損失関数のCalibration (本講義では扱わない)

# 勾配降下法(最急降下法)による最適化

## ■ ロジスティック損失の最小化

$$\min_{\theta} L(\theta) := \frac{1}{N} \sum_{n=1}^N \log \left( 1 + \exp \left( -y^{(n)} (x^{(n)})^{\top} \theta \right) \right)$$

凸関数

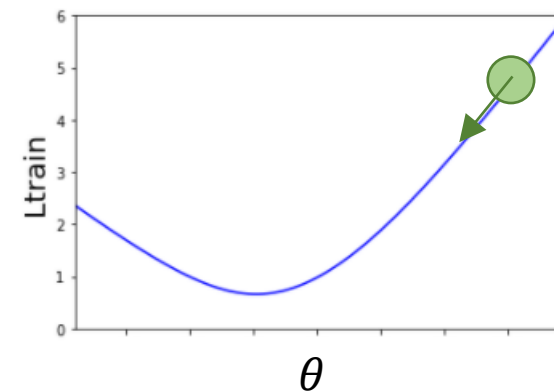
## ■ 凸関数の最小化

- ・ 微分ゼロの点 = 大域最適解

## ■ 勾配降下法(最急降下法)

- ・  $\theta^{[t+1]} \leftarrow \theta^{[t]} - \eta \frac{\partial L(\theta^{[t]})}{\partial \theta}$
- ・  $\frac{\partial L(\theta)}{\partial \theta_i} = -\frac{1}{N} \sum_{n=1}^N \frac{\exp(-y^{(n)} x^{(n)\top} \theta)}{1 + \exp(-y^{(n)} x^{(n)\top} \theta)} y^{(n)} x_i^{(n)}$

$\eta$ : ステップサイズ(学習率)



# 最適化

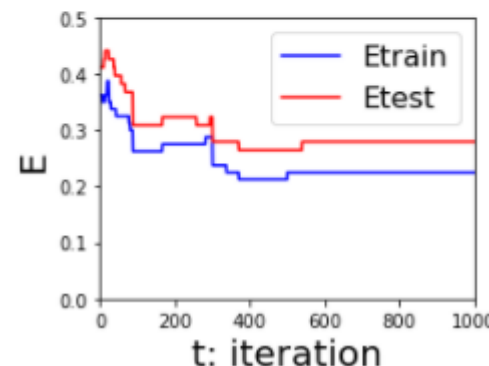
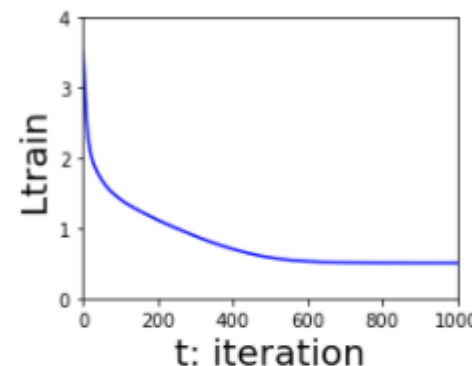
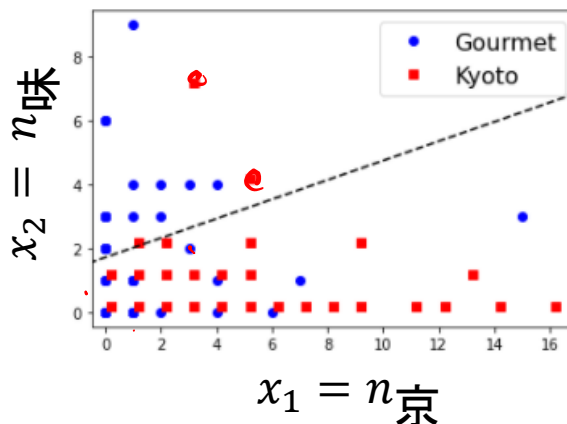
$$\min_{\theta} L(\theta) = \frac{1}{N} \sum_{n=1}^N \log \left( 1 + \exp \left( -y^{(n)} x^{(n)\top} \theta \right) \right)$$

## ■ 勾配降下法(最急降下法)による最適化

- $\theta^{[t+1]} \leftarrow \theta^{[t]} - \eta \frac{\partial L(\theta^{[t]})}{\partial \theta}$
- $\frac{\partial L(\theta)}{\partial \theta_i} = -\frac{1}{N} \sum_{n=1}^N \frac{\exp(-y^{(n)} x^{(n)\top} \theta)}{1 + \exp(-y^{(n)} x^{(n)\top} \theta)} y^{(n)} x_i^{(n)}$

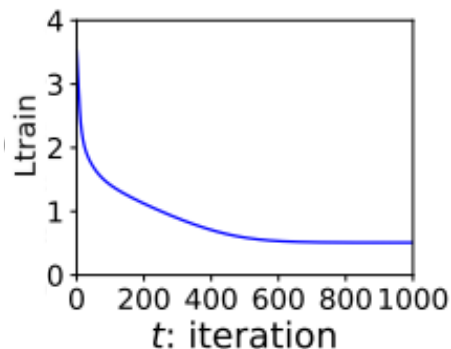
## ■ 結果例

- $f_{\theta}(x) = 0.24n_{\text{京}}$   
 $-0.70n_{\text{味}}$   
 $+0.74$

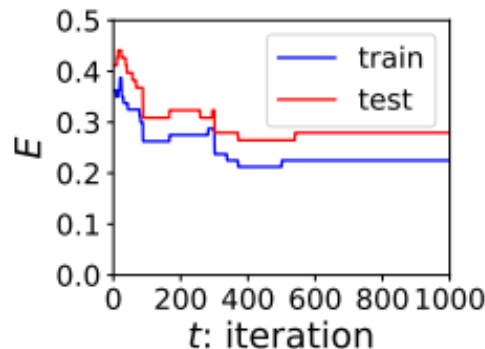




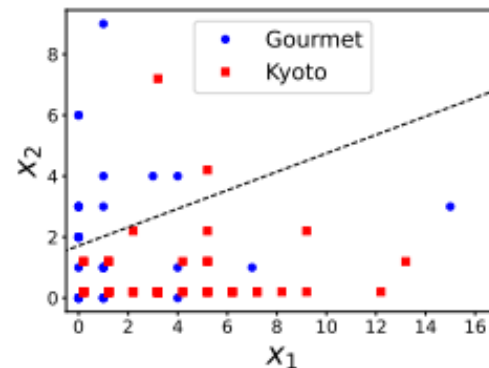
# 結果例



(a) 勾配降下法の収束



(b) 0-1 損失の推移



(c) 学習された識別境界

- 勾配降下法は600ステップ程度で収束
- 学習されたモデル
  - ・  $f_{\theta}(x) = 0.24n_{\text{京}} - 0.70n_{\text{味}} + 0.74$
- より高性能なモデルを目指すには、多項式モデルや正則化を導入する。

## 2. 分類と代理損失と勾配降下法

### ■ データの用意

- ・ 入力 $x$ と出力 $y$ を決める。
- ・ 学習データ $D = \{x^{(n)}, y^{(n)}\}_{n=1}^N$ を集める。

### ■ モデルの学習

- ・ モデル候補(関数のクラス)を設定する。
- ・ 損失関数、正則化を設定する。
- ・ データにもっとも適合する関数をモデル候補の中から見つける。
- ・ 最適なハイパーパラメータを見つける。

線形分類モデル

0-1損失  
ロジスティック損失

最適化  
(勾配降下法)

### ■ モデルの評価

- ・ 予測精度の評価

Accuracy, Precision,  
Recall, F値, AUROC

# 分類結果の良さの評価指標

## ■ 混同行列 (Confusion Matrix)

- ラベル毎の正誤数を表にまとめたもの。

	$\text{sign}(f(x)) = 1$	$\text{sign}(f(x)) = -1$
$y = 1$	True Positive (TP)	False Negative (FN)
$y = -1$	False Positive (FP)	True Negative (TN)

$$\text{Acc} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{FN} + \text{TN}}$$

## ■ 例：文書トピックの予測

	$\text{sign}(f(x)) = 1$	$\text{sign}(f(x)) = -1$
$y = 1$	39	1
$y = -1$	18	10

$$\begin{aligned}\text{Acc} &= \frac{39 + 10}{39 + 18 + 1 + 10} \\ &= \frac{49}{66}\end{aligned}$$

# 分類結果の良さの評価指標

- 混同行列を見ると、分類結果の偏りがわかる。

	$\text{sign}(f(x))$ = 1	$\text{sign}(f(x))$ = -1
$y = 1$	39	1
$y = -1$	18	10

- 「京都観光 ( $y = 1$ )」の文書は  $\frac{39}{40}$  で正しく分類できている。
- 「グルメ ( $y = -1$ )」の文書は  $\frac{10}{28}$  でしか正しく分類できてない。

## ■ ラベル毎の評価

- Precision: 予測されたラベルが正しかった割合。

- 京都観光 ( $y = 1$ ):  $\text{Prec} = \frac{39}{39+18} = \frac{39}{57}$
- グルメ ( $y = -1$ ):  $\text{Prec} = \frac{10}{10+1} = \frac{10}{11}$

- Recall: 真のラベルが正しく予測された割合。

- 京都観光 ( $y = 1$ ):  $\text{Recall} = \frac{39}{39+1} = \frac{39}{40}$
- グルメ ( $y = -1$ ):  $\text{Recall} = \frac{10}{10+18} = \frac{10}{28}$

$y = -1$  のPrecisionが高い。



-1 と予測されたデータについては、本当に  $y = -1$  である可能性が高い。

$y = -1$  のRecallが低い。



-1 という予測には取りこぼしが多い。

# 分類結果の良さの評価指標

- 混同行列を見ると、分類結果の偏りがわかる。

	$\text{sign}(f(x)) = 1$	$\text{sign}(f(x)) = -1$
$y = 1$	39	1
$y = -1$	18	10

- 「京都観光 ( $y = 1$ )」の文書は $\frac{39}{40}$ で正しく分類できている。
- 「グルメ ( $y = -1$ )」の文書は $\frac{10}{28}$ でしか正しく分類できてない。

## ■ ラベル毎の評価

- F値: PrecisionとRecallの2つを1つの値に統合した指標。

- F値 = PrecisionとRecallの調和平均

- 京都観光 ( $y = 1$ ):  $F = \left( \frac{57}{39} + \frac{40}{39} \right)^{-1} = \frac{39}{97}$

- グルメ ( $y = -1$ ):  $F = \left( \frac{11}{10} + \frac{28}{10} \right)^{-1} = \frac{10}{39}$

$$F = \left( \frac{1}{\text{Prec}} + \frac{1}{\text{Recall}} \right)^{-1} = \frac{2 \cdot \text{Prec} \cdot \text{Recall}}{\text{Prec} + \text{Recall}}$$

# 分類結果の良さの評価指標

## ■ ROC曲線とAUC (AUROC)

- 評価項目1: 真陽性率 (True Positive Rate; TPR)

- クラス $y$ のデータのうち、どれだけについて正しく $y$ と予測できたか？
- “狼少年” 常に $y$ と予測していれば真陽性率は100%

真陽性率が高いほど、  
予測の見落としが少ない

- 評価項目2: 偽陽性率 (False Positive Rate; FPR)

- クラス $y$ でないデータのうち、どれだけについて誤って $y$ と予測してしまったか？
- “狼少年” 常に $y$ でないと予測していれば偽陽性率は0%

偽陽性率が低いほど、  
誤報が少ない

- 実際に評価したいのはTPRとFPRのトレードオフ

- 「TPR高、FPR低」であるほど良い分類結果と言える。
- TPR100%、FPR0%が最も理想的。

# 分類結果の良さの評価指標

## ■ ROC曲線とAUC (AUROC)

- TPRとFPRのトレードオフの評価
- 符号による分類の拡張

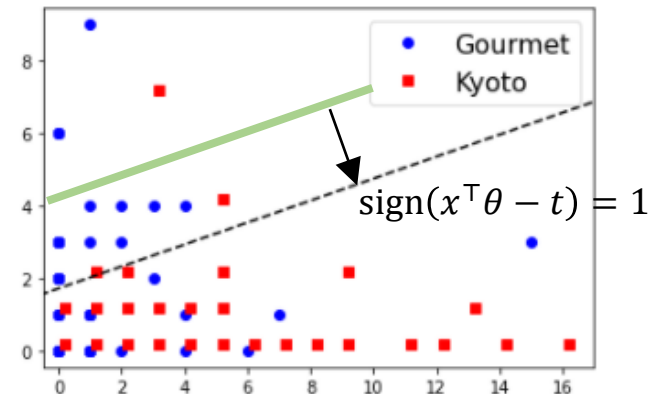
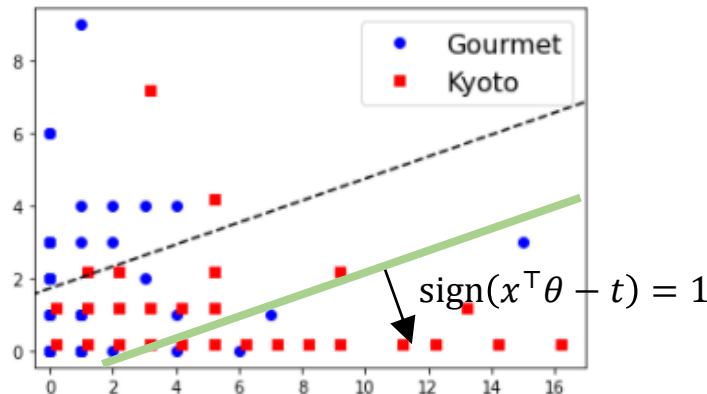
$$\text{sign}(f(x) - t) = \begin{cases} 1, & \text{if } f(x) > t \\ -1, & \text{if } f(x) \leq t \end{cases}$$

閾値 $t$ で分類結果を決める。

### • トレードオフ

- 閾値 $t$ を上げる/下げるとクラス $y = 1$ と分類されづらく/やすくなる。
  - クラス $y = 1$ のTPRが減少/増加
  - クラス $y = 1$ のFPRも減少/増加

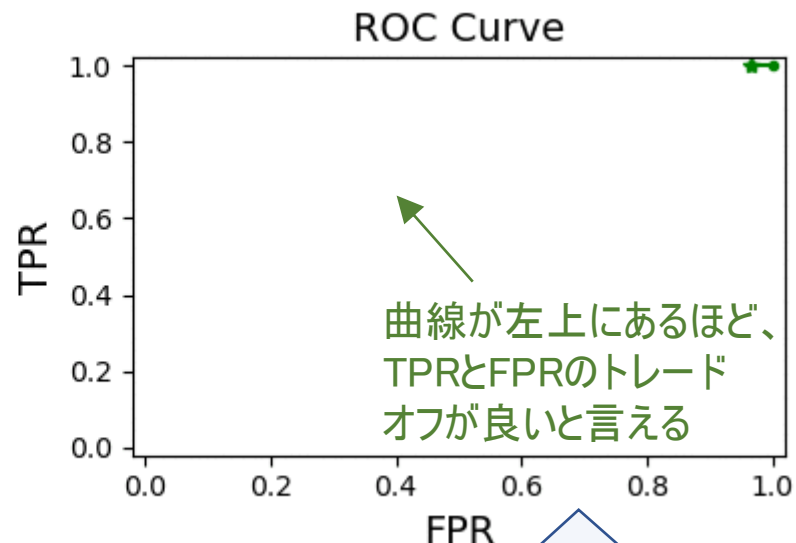
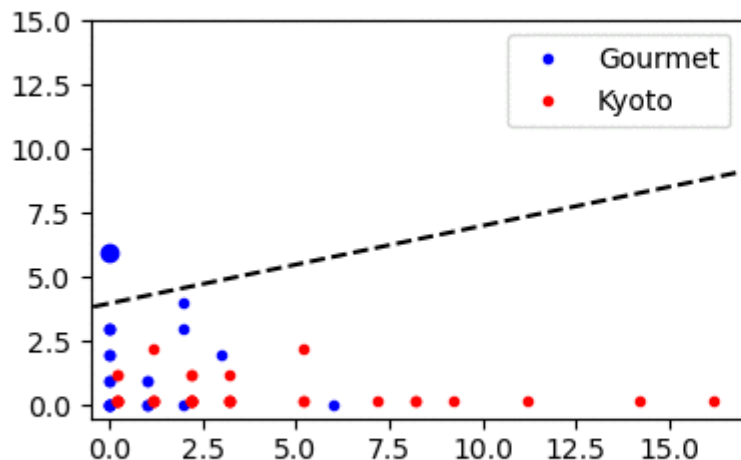
$f(x) = x^\top \theta$  の場合の例



# 分類結果の良さの評価指標

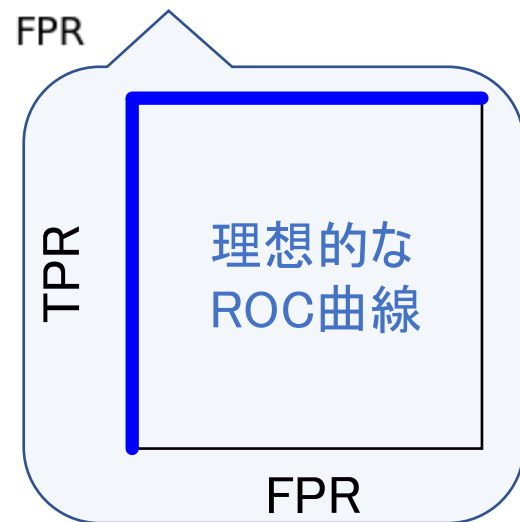
## ■ ROC曲線

- ・ 横軸にFPR、縦軸にTPRをとって描画。



## ■ AUC (Area Under ROC)

- ・ ROC曲線下の面積
  - 大きいほど良いトレードオフ。最大で1。





## 2. 分類と代理損失と勾配降下法

### ■ データの用意

- ・ 入力 $x$ と出力 $y$ を決める。
- ・ 学習データ $D = \{x^{(n)}, y^{(n)}\}_{n=1}^N$ を集める。

### ■ モデルの学習

- ・ モデル候補(関数のクラス)を設定する。
- ・ 損失関数、正則化を設定する。
- ・ データにもっとも適合する関数をモデル候補の中から見つける。
- ・ 最適なハイパーパラメータを見つける。

線形分類モデル

0-1損失  
ロジスティック損失

最適化  
(勾配降下法)

### ■ モデルの評価

- ・ 予測精度の評価

Accuracy, Precision,  
Recall, F値, AUROC

# まとめ

## ■ 1. 回帰

- ・ 最小二乗回帰
- ・ モデルの評価
- ・ L2正則化
- ・ ハイパーパラメータ選択

## ■ 2. 分類

- ・ ロジスティック回帰
- ・ 勾配降下法による最適化
- ・ 様々な評価指標