ML勉強会
第1,2章
2020/06/12

新潟大学　自然科学研究科

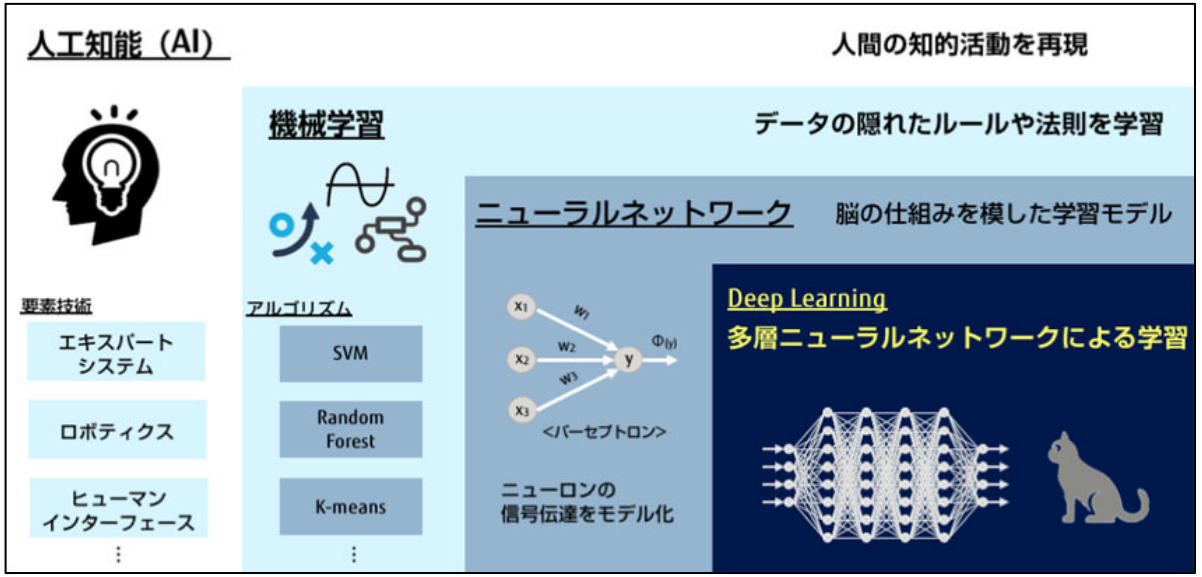
三田雅大

第1章

1. 機械学習の概念と基本用語
2. 機械学習システムをうまく設計するための構成要素
3. 環境構築

機械学習とは

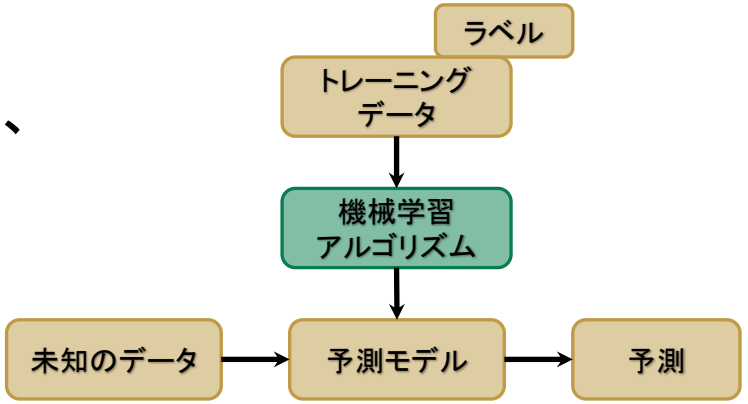
- 人口知能（AI）の一分野
  - 大量のデータから隠れたルールや法則・規則性を導き出すこと
    - 教師あり学習、教師なし学習、強化学習
- 応用例
  - Siriなどの音声認識ソフト
  - スパムフィルタ
  - 検索エンジン
  - 対戦ゲーム
  - 翻訳
  - 顔認識
  - 自動運転　など



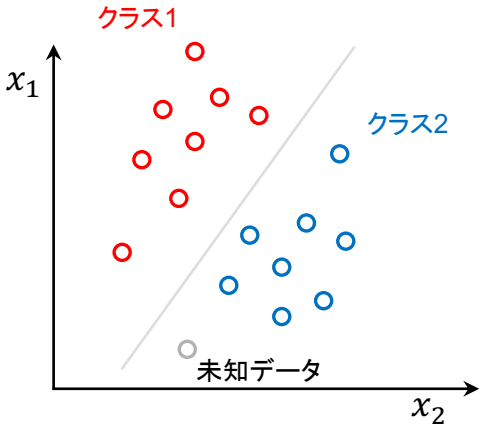
https://blog.global.fujitsu.com/jp/2018-11-29/01/

教師あり学習

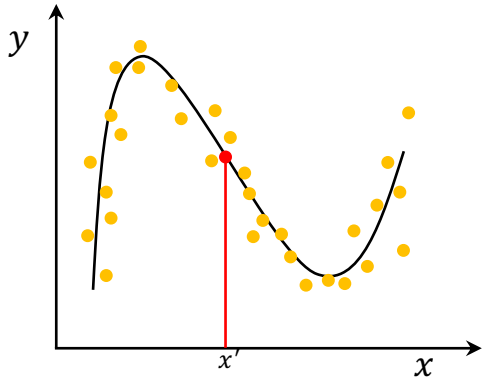
- 教師あり学習とは
  - ラベル付されたデータからモデルを学習し、未知データを予測できるようにすること
- 分類
  - クラスラベルを予測
    - 二値分類
    - 多クラス分類
- 回帰
  - 連続値を予測



Figure：教師あり学習



Figure：分類問題



Figure：回帰問題

強化学習

- 強化学習とは
  - ある環境の中で、目的として設定された報酬を最大化するように学習
- 応用例
  - ゲームのCP
    - 格ゲー
    - AlphaGo
  - ロボットの歩行制御

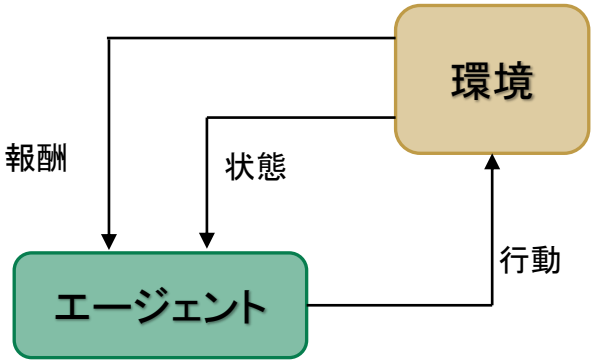


Figure : 強化学習

教師なし学習

- 教師なし学習とは
  - ラベル付けされていないデータや構造が不明なデータを扱う
  - データ内の未知のパターンを見つけ出す
- クラスタリング
  - データの類似性に基づいて、データセットをグループに分類
- 次元削除
  - 高次元のデータから情報を維持した状態で、データをより低い次元の部分空間に圧縮

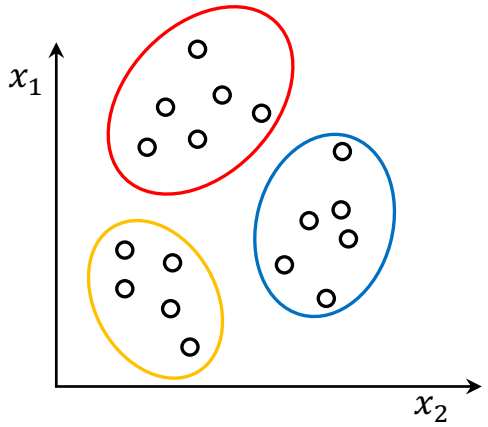


Figure : クラスタリング

第2章

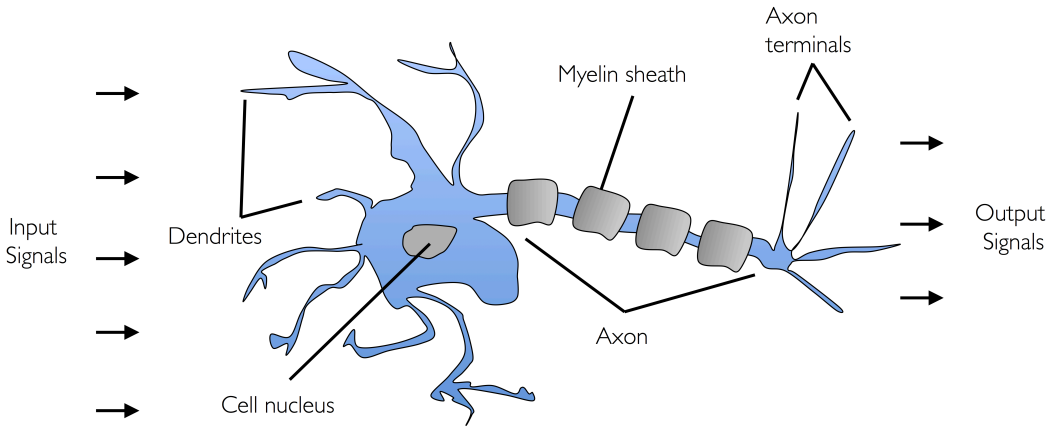
1. 人工ニューロン（パーセプトロン）
2. パーセプトロンの学習アルゴリズムをPythonで実装する
3. Irisデータセットでのパーセプトロンモデルのトレーニング
4. ADALINEと学習の収束
5. 勾配降下法によるコスト関数の最小化
6. 大規模な機械学習と確率的勾配降下法

第2章

1. 人工ニューロン（パーセプトロン）
2. パーセプトロンの学習アルゴリズムをPythonで実装する
3. Irisデータセットでのパーセプトロンモデルのトレーニング
4. ADALINEと学習の収束
5. 勾配降下法によるコスト関数の最小化
6. 大規模な機械学習と確率的勾配降下法

人工ニューロン（パーセプトロン）

- 人工ニューロンとは
  - 複数の入力信号を受け取り、一つの信号を出力
    - 出力は二値のうちどちらか
  - 出力信号
    - 総入力が、ある閾値を超えた場合：1（陽性クラス）
    - 閾値が超えない場合：-1（陰性クラス）



Figure：ニューロン

パーセプトロン

■パーセプトロン

- □人工ニューロンに基づいて考案された学習規則
  - ➢最適な重み係数を学習
  - ➢入力信号と掛け合わせる
  - ➢ニューロンが発火するかどうか判断

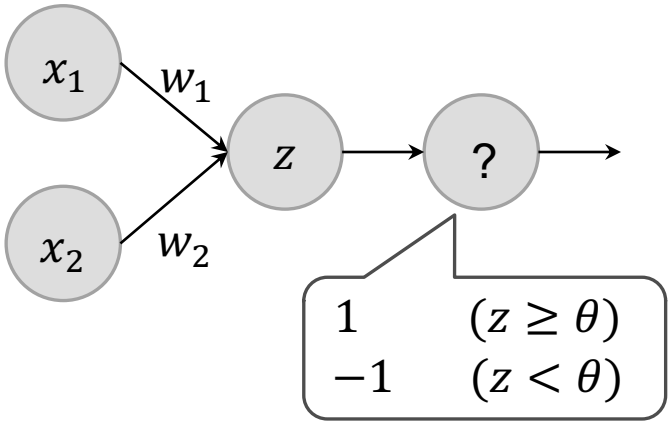
■簡単な例

- □総入力：z

$$z = w_1 x_1 + w_2 x_2$$

- □出力：y

$$y = \begin{cases} 1 & (z \geq \theta) \\ -1 & (z < \theta) \end{cases} \qquad \theta：\text{閾値}$$



Figure：単純なパーセプトロン

- □もっと単純にして…(p.19)

パーセプトロンの学習規則

活性化関数

- 入力信号の総和を出力信号に変換する関数：$\phi(z)$
  - パーセプトロンではステップ関数

$$\phi(z) = \begin{cases} 1 & (z \geq 0) \\ -1 & (z < 0) \end{cases}$$

$$y = \phi(z)$$

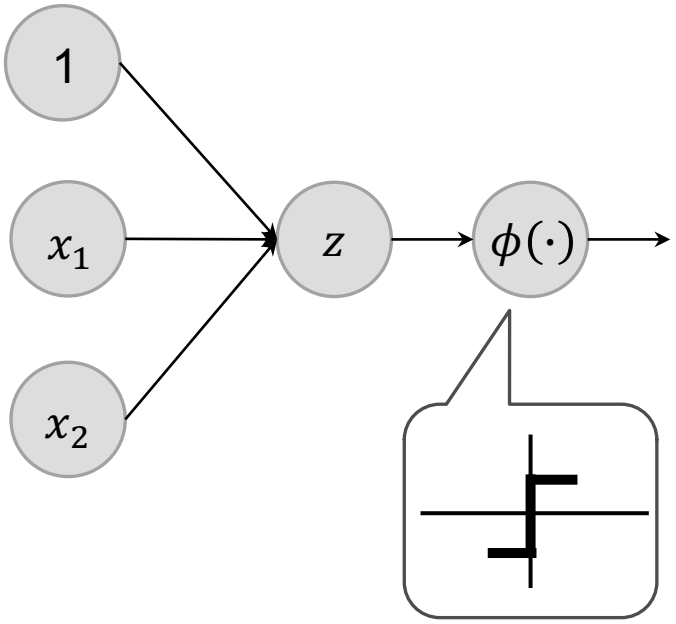


Figure : パーセプトロン

パーセプトロンの学習規則

■学習規則（p.21参照）

□重みの更新

- ➢重みを0、または値の小さい乱数で初期化
- ➢トレーニングサンプルごとに、

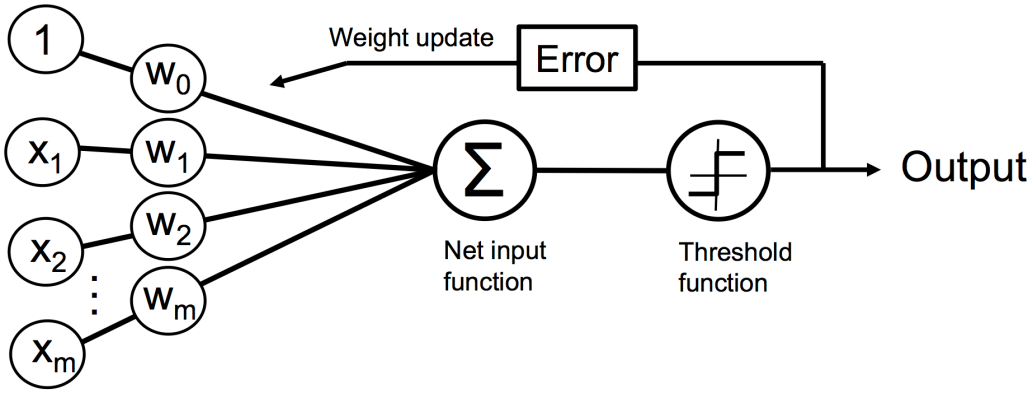
  1. 出力値$\hat{y}$を計算する
  2. 重みを更新する

  を実行する

$$\Delta w_j = \eta\left(y^{(i)} - \hat{y}^{(i)}\right)x_j^{(i)}$$

$$w_j := w_j + \Delta w_j$$

η：学習率
$y^{(i)}$：i番目の正解ラベル
$\hat{y}^{(i)}$：i番目の正解ラベル



Figure：パーセプトロンの重み更新

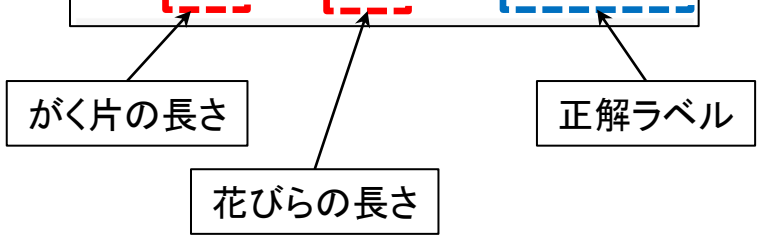
第2章

1. 人工ニューロン（パーセプトロン）
2. パーセプトロンの学習アルゴリズムをPythonで実装する
3. Irisデータセットでのパーセプトロンモデルのトレーニング
4. ADALINEと学習の収束
5. 勾配降下法によるコスト関数の最小化
6. 大規模な機械学習と確率的勾配降下法

Pythonによる実装

- Irisデータセット（p.9参照）
  - SetosaとVersicolorの品種クラスを分類
  - Versicolor：1、Setosa：-1
  - 特徴量
    - がく片の長さ
    - 花びらの長さ

	0	1	2	3	4
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa
5	5.4	3.9	1.7	0.4	Iris-setosa



データについて

- Irisデータの可視化（散布図）
  - 横軸：がく片の長さ
  - 縦軸：花びらの長さ
- 決定境界
  - クラスを分類する境界
  - パーセプトロンにより境界線を見つける

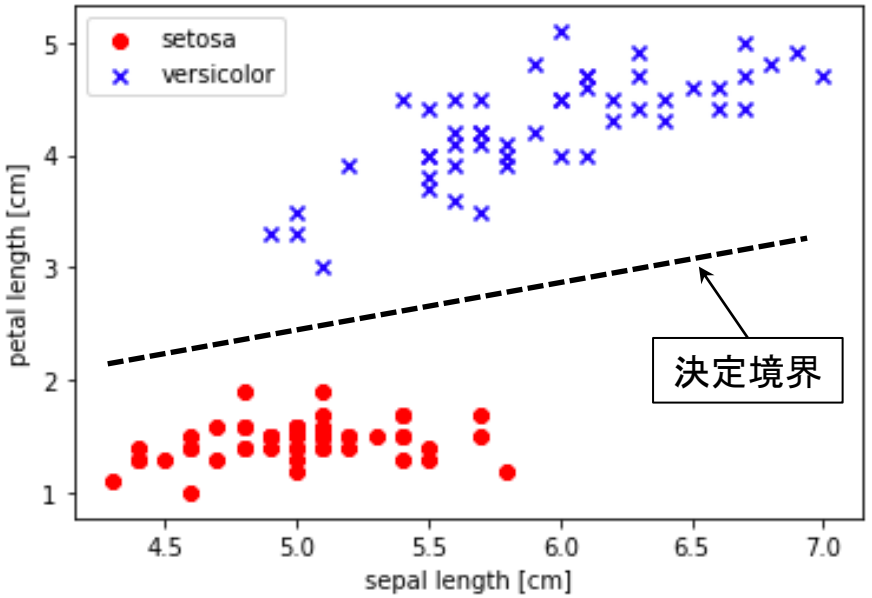
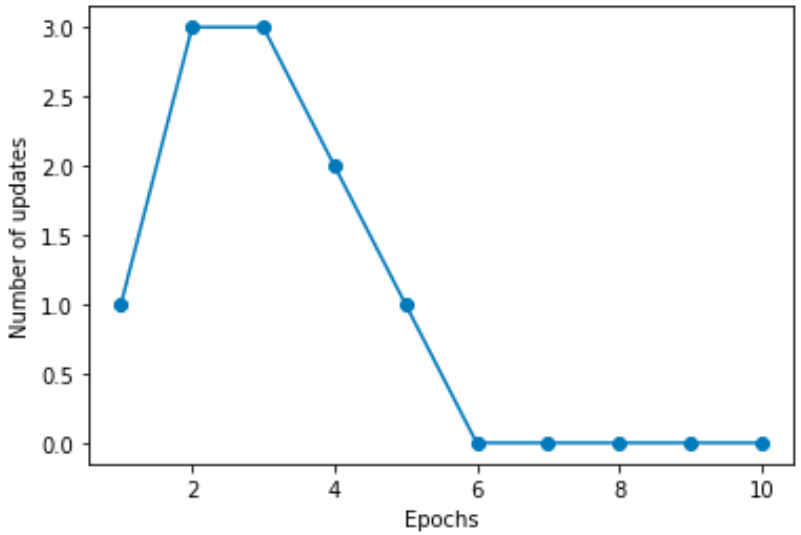


Figure : Irisデータの分布

データの可視化

■誤分類

- □横軸：エポック数
  - ➢エポック数：トレーニングデータ全体を学習した回数
- □縦軸：誤分類
- □1エポックごとの誤分類の推移を見る



Figure：エポックごとの学習の推移

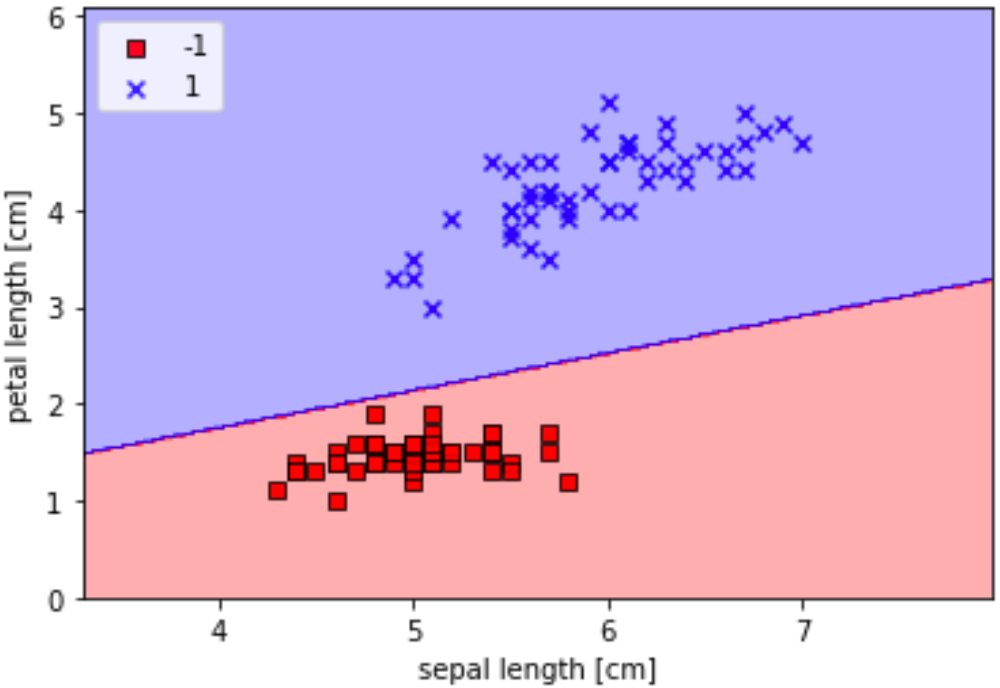
6エポック目の学習で
誤分類が0

トレーニングデータを
完全に分類している

データの可視化

決定境界の可視化

- plot_decision_regions 関数を使用（p.32参照）



Versicolor

Setosa

Figure : Irisデータの決定境界

第2章

1. 人工ニューロン（パーセプトロン）
2. パーセプトロンの学習アルゴリズムをPythonで実装する
3. Irisデータセットでのパーセプトロンモデルのトレーニング
4. ADALINEと学習の収束
5. 勾配降下法によるコスト関数の最小化
6. 大規模な機械学習と確率的勾配降下法

ADALINE（ADAptive LIner NEuron）

- ADALINEとは
  - パーセプトロンの改良
- パーセプトロンとの違い
  - 活性化関数に恒等関数を使用（総入力をそのまま出力）
    - $\phi(z) = \phi(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}) = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}$
  - 最終出力としてクォンタライザーで二値分類
  - 重みの更新に勾配降下法を使用
    - 勾配降下法により、コスト関数を最小化する
    - 全サンプルの予測値と真の値の誤差を用いて重みを更新

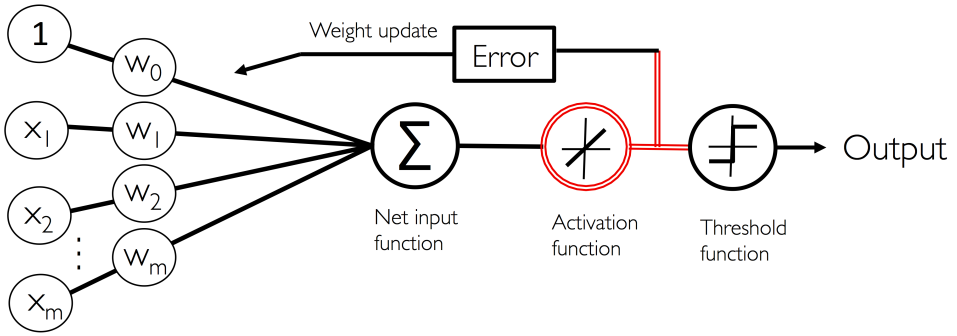


Figure : ADALINE

コスト関数

コスト関数とは

- 推定の精度の悪さを表す指標
- 0に近いほど性能が良い
  - 予測値と真の値が近い
- ADALINEでは「誤差平方和」が使用される

誤差平方和

- 正解となる教師データと出力の差の2乗を計算し、その総和を求める

$$J(\boldsymbol{w}) = \frac{1}{2}\sum_{i}\left(y^{(i)} - \phi\left(z^{(i)}\right)\right)^2$$

このコスト関数を最小化するようにして、重みを学習する

ADALINEの学習規則

重みの更新

勾配降下法

- 勾配を利用してコスト関数が最小化するように重みを更新

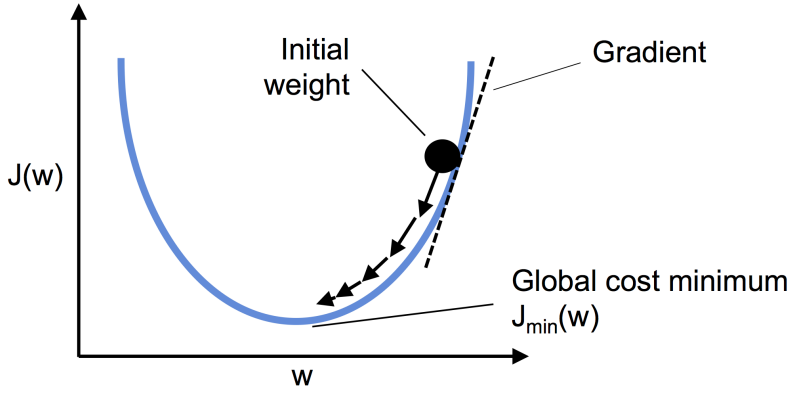
$$\boldsymbol{w} := \boldsymbol{w} + \Delta\boldsymbol{w}$$

$$\Delta\boldsymbol{w} = -\eta\nabla J(\boldsymbol{w})$$

$$\frac{\partial J}{\partial w_j} = -\sum_i \left(y^{(i)} - \phi\left(z^{(i)}\right)\right) x_j^{(i)}$$

$$w_j := w_j + \sum_i \left(y^{(i)} - \phi\left(z^{(i)}\right)\right) x_j^{(i)}$$

- 計算の詳細はp.37参照



Figure：勾配降下法

コスト関数の変動

学習率に対するコスト関数の変動

- 学習率：$\eta = 0.01$ と 学習率：$\eta = 0.0001$ を比較
- 横軸：エポック数
- 縦軸：コスト関数の大きさ

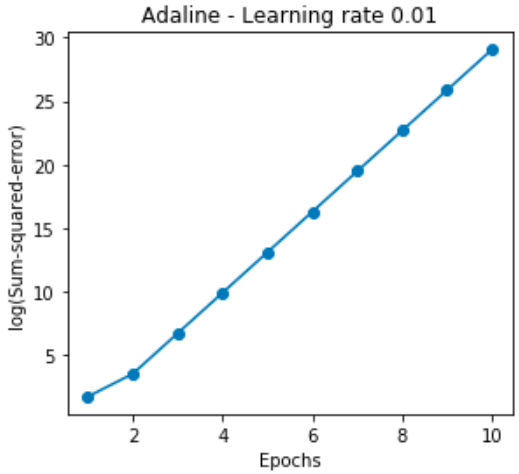


Figure : $\eta = 0.01$

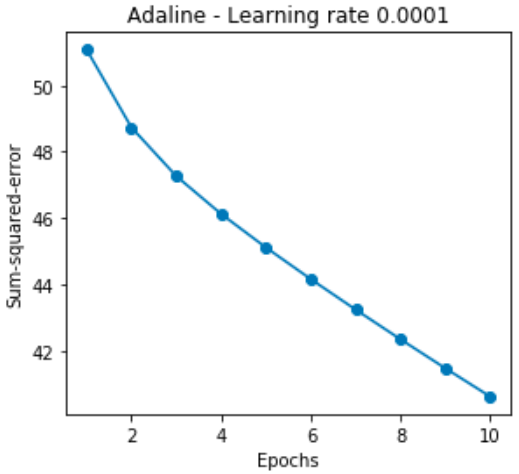


Figure : $\eta = 0.0001$

- 学習率が高すぎるとコスト関数が増加
- 学習率が低すぎるとコスト関数の減少が遅い（学習が進まない）

データセットの標準化

標準化

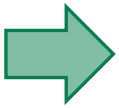
- 各特徴量を一定のルールでスケーリングすることで、特徴量を捉えやすくする
- 各特徴量の平均を0、標準偏差を1にする

$$x'_j = \frac{x_j - \mu_j}{\sigma_j}$$

x_j : j番目の特徴量
μ_j : サンプルの平均
σ_j : 標準偏差

```
[[5.1 1.4]
 [4.9 1.4]
 [4.7 1.3]
 [4.6 1.5]
 [5.  1.4]
 [5.4 1.7]
 [4.6 1.4]
 [5.  1.5]]
```

標準化前



```
[[-0.5810659  -1.01435952]
 [-0.89430898 -1.01435952]
 [-1.20755205 -1.08374115]
 [-1.36417359 -0.94497788]
 [-0.73768744 -1.01435952]
 [-0.11120129 -0.80621461]
 [-1.36417359 -1.01435952]
 [-0.73768744 -0.94497788]
```

標準化後

データの可視化

- 標準化後に学習
  - 学習率$\eta = 0.01$の場合でも正しく分類されている
    - 標準化の効果
  - 正しく分類されていてもコスト関数は0にならない

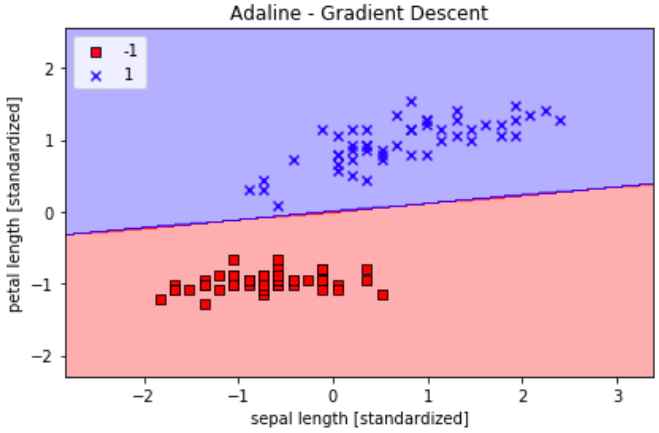


Figure : 決定境界

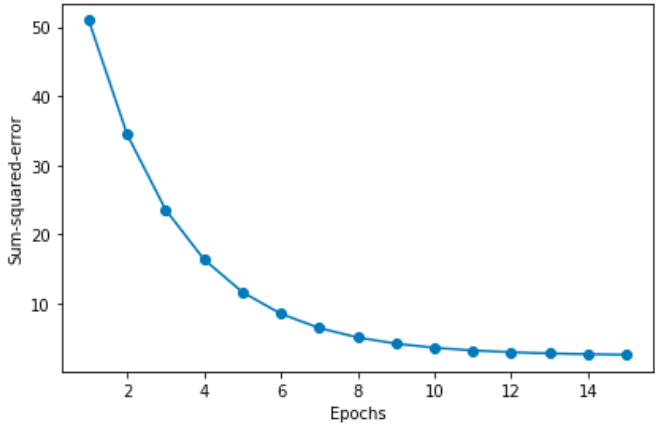


Figure : コスト関数の推移

第2章

1. 人工ニューロン（パーセプトロン）
2. パーセプトロンの学習アルゴリズムをPythonで実装する
3. Irisデータセットでのパーセプトロンモデルのトレーニング
4. ADALINEと学習の収束
5. 勾配降下法によるコスト関数の最小化
6. 大規模な機械学習と確率的勾配降下法

大規模な機械学習

- データセットが巨大な場合
  - バッチ勾配降下法→計算コストが非常に高くなる
- 確率的勾配降下法
  - トレーニングサンプルごとに重みを更新する
    - 効率よく学習を進めることができる
    - エポックごとにトレニーングデータをシャッフル必要がある

適応学習率

- 学習率をエポックごとに更新
  - 固定の学習率を使用せず、徐々に減少する適応学習率を使用

$$\eta = \frac{c_1}{[\text{イテレーションの回数}] + c_2}$$

$$c_1, c_2 : \text{定数}$$

  - より適切な大局的最小値に近づけることができる

おまけ

重みの更新(具体例)

- エポック:1、サンプル:1 を使用した場合
  - $x_1^{(0)} = 5.1$、$x_2^{(0)} = 1.4$、$y^{(0)} = -1$、(Setosa)
  - 学習率: $\eta = 0.01$、重みの初期値は全て0とする
    - $\hat{y}^{(0)} = \phi(\underbrace{0\times1 + 0\times5.1 + 0\times1.4}_{\text{総入力}z}) = \phi(0) = 1$

$$\Delta w_0 = \eta\left(y^{(0)} - \hat{y}^{(0)}\right)$$
$$\Delta w_1 = \eta\left(y^{(0)} - \hat{y}^{(0)}\right)x_1^{(0)}$$
$$\Delta w_2 = \eta\left(y^{(0)} - \hat{y}^{(0)}\right)x_2^{(0)}$$

$$\Delta w_0 = \eta(-1-1) = -0.02$$
$$\Delta w_1 = \eta(-1-1)5.1 = -0.102$$
$$\Delta w_2 = \eta(-1-1)1.4 = -0.028$$

$$w_0 := 0 - 0.02 = -0.02$$
$$w_1 := 0 - 0.102 = -0.102$$
$$w_2 := 0 - 0.028 = -0.028$$

$$z = -0.02 - 0.102x_1^{(1)} - 0.028x_2^{(1)}$$