

機械学習イントロダクション + Day3 カリキュラム

ペパボ研究所 三宅悠介 / GMO PEPABO inc.

2021.07.19 機械学習研修 Day3

GMOペパボ

カリキュラム概要と目標

- **Day3: 機械学習を自分の言葉で説明できる**
 - データから知識を学習させるための方法について、初歩的な知識を導入する
 - レッスン&ハンズオンを通して、手を動かしながら知識の解釈を高める
- **Day4: 機械学習を使って課題を解決する流れを思い浮かべることができる**
 - 機械学習で課題を解決するために考慮すべき点について、基礎的な知識を導入する
 - レッスン&ハンズオンを通して、手を動かしながら知識の解釈を高める
- **Day5: 機械学習をサービスに導入するまでの流れを思い浮かべることができる**
 - 配属先のサービスで機械学習を使った施策を実現するために、知っておくべき知識を導入する
 - Day1~4までの学びを用いて、擬似的なサービス導入準備をハンズオンで体験する

Day3: 機械学習を自分の言葉で説明できる

- 座学(機械学習イントロダクション)(15min)
- レッスン&ハンズオン(160min)

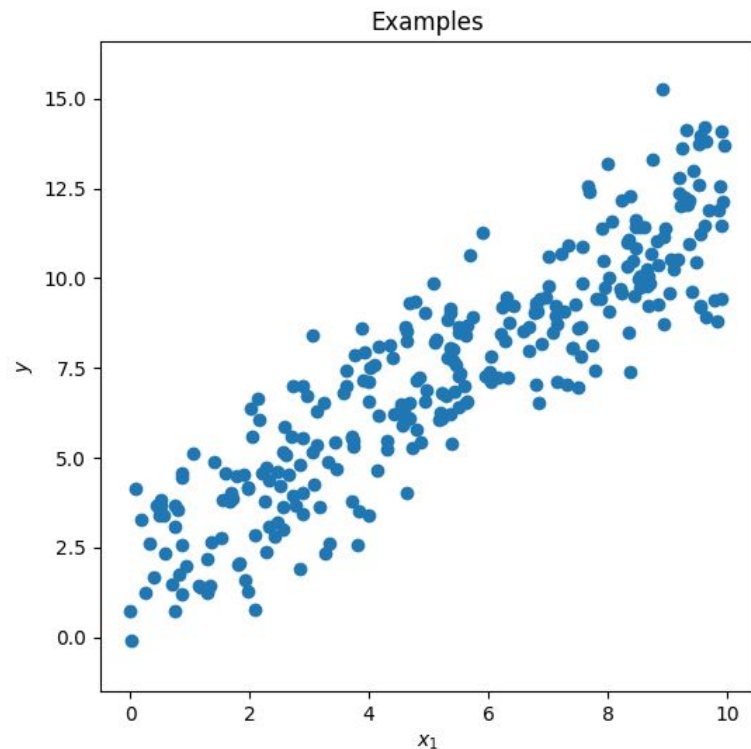
機械学習イントロダクション

“機械学習とは何か”

そして何ではないのか

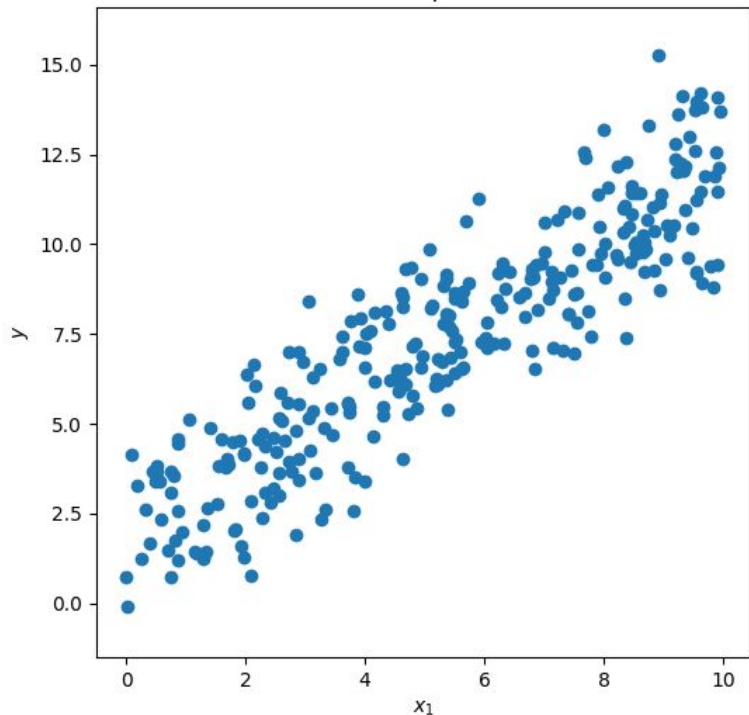
機械学習とは

プログラムが、データから知識を学習すること、またそれを用いてタスクを実行する手法のこと



まずはデータがある
(ここでは x の値に対応する y
の値の集合)

Examples



ここで、 x と y の対応関係を関数 f と置き、


$$y = f(x)$$

与えられたデータの対応性をできるだけ再現するような関数 f の**中身**を求める

例えばこう考えてみよう

プログラム開発において、仕様に従う関数を開発する。

多くの場合、仕様は入出力の対応関係であり、その実装は、開発者に任せられている。



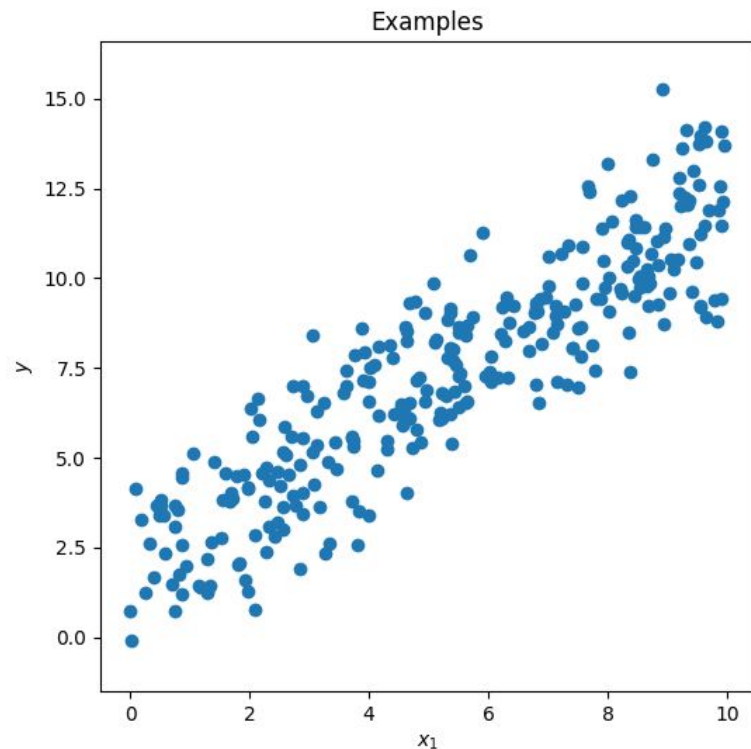
```
def f(x):  
    return x  
  
def test_f():  
    x1 = 1.0  
    y1 = f(x1)  
    assert y1 == 2.0, 'Test failed'
```

例えばこう考えてみよう

テスト駆動開発などであれば、先に仕様(に沿った入出力のパターン)をテストとして定め、次に関数の中身を実装する。

```
def f(x):  
    return x  
  
def test_f():  
    x1 = 1.0  
    y1 = f(x1)  
    assert y1 == 2.0, 'Test failed'
```

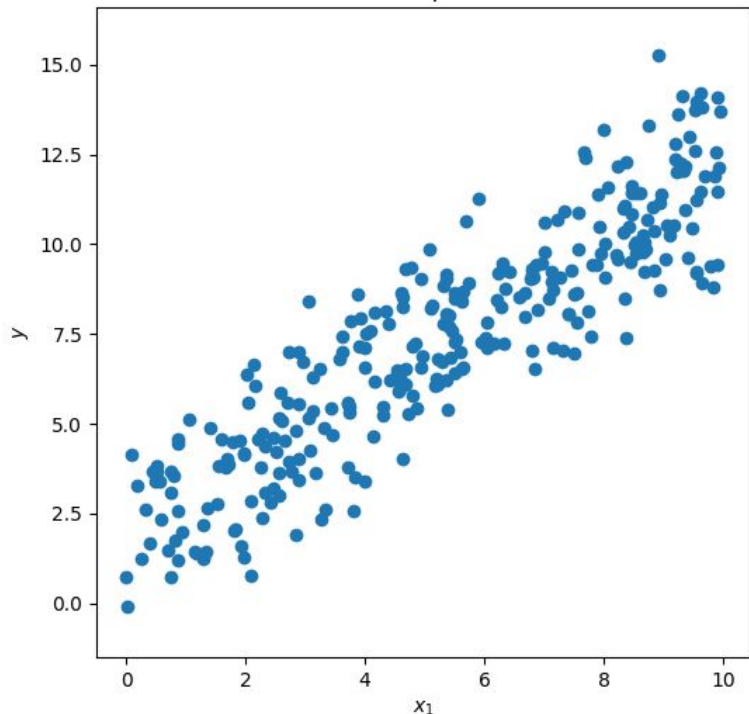
実装の自由度と入出力の対応関係の柔軟性はトレードオフの関係。以下、機械的に求めるために、実装の自由度を制限する。



入力から出力を推測したい

$$\boldsymbol{x} = \begin{pmatrix} x_0 = 1 \\ x_1 \end{pmatrix} \longrightarrow y'$$

Examples



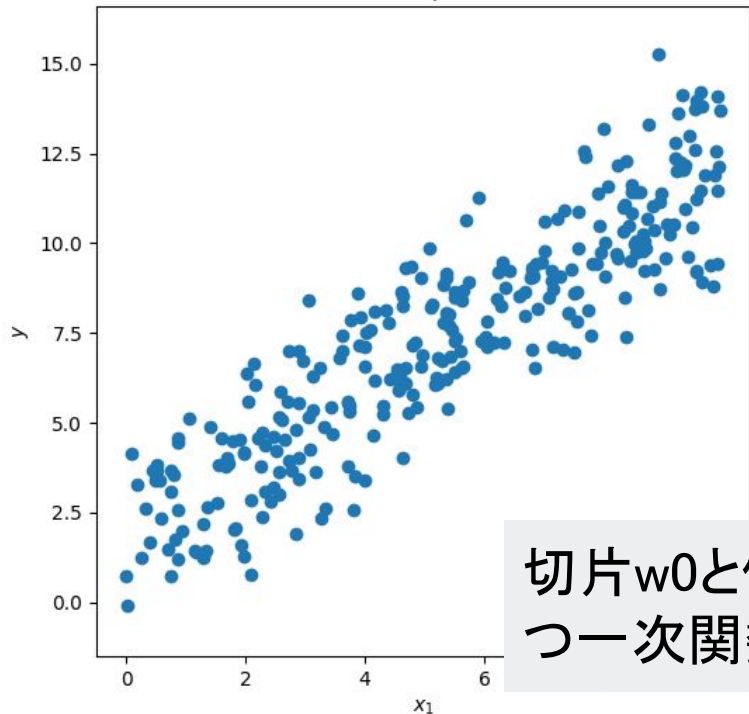
入力から出力を推測したい

$$x \longrightarrow f(x) \longrightarrow y'$$

入力から出力を予測
する関数を考える

= 与えられたデータの対応性をできるだけ再現するような関数 f の**中身**を求める

Examples



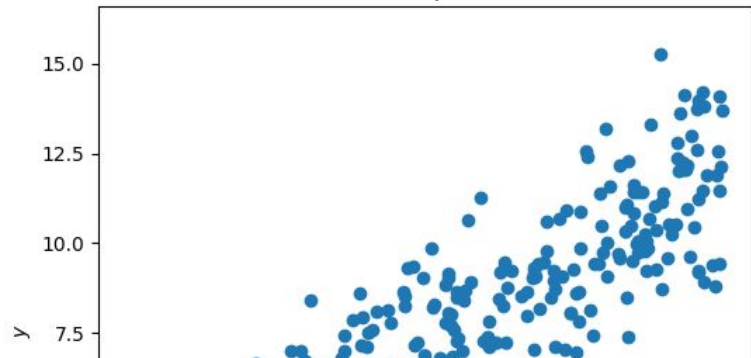
切片 w_0 と傾き w_1 を持つ一次関数で表す

= 実装の自由度の話

入力から出力を推測したい

$$\begin{aligned} \boldsymbol{x} &\longrightarrow f_{\boldsymbol{w}}(\boldsymbol{x}) \longrightarrow y' \\ &= \boldsymbol{w}^{\top} \boldsymbol{x} \\ &= w_0 x_0 + w_1 x_1 \end{aligned}$$

Examples



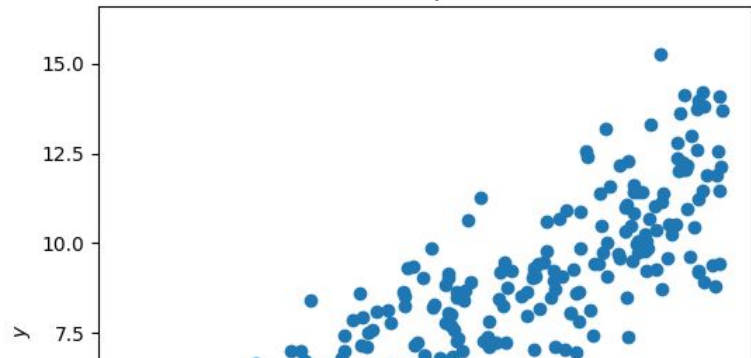
① w_0 と w_1 を自分で決める

② w_0 と w_1 を発見させるプログラムを書く

入力から出力を予測したい

$$\begin{aligned} \boldsymbol{x} &\longrightarrow f_{\boldsymbol{w}}(\boldsymbol{x}) \longrightarrow y' \\ &= \boldsymbol{w}^{\top} \boldsymbol{x} \\ &= w_0 x_0 + w_1 x_1 \end{aligned}$$

Examples



① w_0 と w_1 を自分で決める

② w_0 と w_1 を発見させるプログラムを書く

入力から出力を予測したい

$$\begin{aligned} \boldsymbol{x} &\longrightarrow f_{\boldsymbol{w}}(\boldsymbol{x}) \longrightarrow y' \\ &= \boldsymbol{w}^T \boldsymbol{x} \\ &= w_0 x_0 + w_1 x_1 \end{aligned}$$

データから学習する、とは
プログラムによって
入力に対して
望ましい出力を返すよう
数式のパラメータを調整すること

データから学習する、ためには

① データの生成規則をモデルとして記述する

$x \rightarrow f_w(x) \rightarrow y'$ ← 入出力(変数)

$= w^T x$

$= w_0 x_0 + w_1 x_1$ ← 数式の構造

← 数式のパラメータ

データから学習する、ためには

- ① データの生成規則をモデルとして記述する
- ② モデルの良し悪しを定式化する

$$L(\boldsymbol{w}) = \frac{1}{2} \sum_{i=0}^n (y_i - y'_i)^2$$

データから学習する、ためには

- ① データの生成規則をモデルとして記述する
- ② モデルの良し悪しを定式化する
- ③ 現在のパラメータの良し悪しを②式で評価

データから学習する、ためには

- ① データの生成規則をモデルとして記述する
- ② モデルの良し悪しを定式化する
- ③ 現在のパラメータの良し悪しを②式で評価
- ④ ③の評価に基づきパラメータを調整

データから学習する、ためには

- ① データの生成規則をモデルとして記述する
- ② モデルの良し悪しを定式化する
- ③ 現在のパラメータの良し悪しを②式で評価
- ④ ③の評価に基づきパラメータを調整

データから学習する、ためには

- ① データの生成規則をモデルとして記述する
- ② モデルの良し悪しを定式化する
- ③ 現在のパラメータの良し悪しを②式で評価
- ④ ③の評価に基づきパラメータを調整
- ⑤ モデルを用いてタスクを実行する

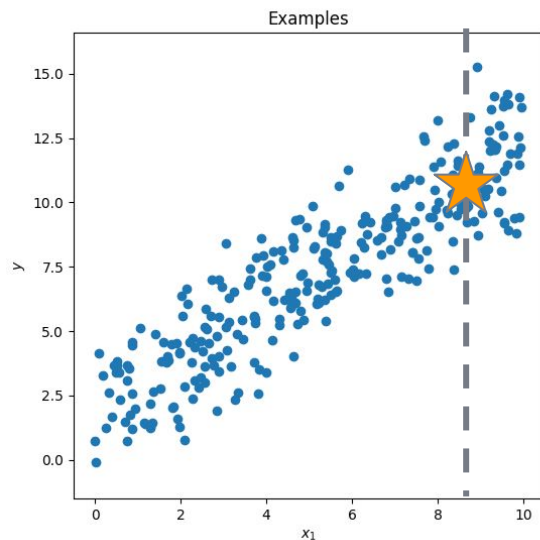
“機械学習のタスク”

何のために学習するのか

機械学習とは(再掲)

プログラムが、データから知識を学習すること、また
それを用いてタスクを実行する手法のこと

モデルとタスクと汎化



機械学習では、モデルが前提とするデータの生成規則における、**未知のデータ**に対してどれだけそのタスクをよくこなせるかが求められる。

例えばこう考えてみよう

事前に与えられたデータの
対応関係を全て満たす実装
は必ずしも、未知のデータに
対する振る舞いが正しいと
は限らない。

```
def f(x):  
    if x == 1.0:  
        return 2.0  
    elif x == 2.0:  
        return 4.0  
    elif x == 3.0:  
        return 6.0  
    ...
```

回帰 (Regression)

入力から出力を予測する問題設定。

出力は連続値として得られる。

入出力の関係性は、訓練データ(ラベル付きの例)から学習する(教師あり学習)

例) 気温から売り上げを予測、土地の特徴から家屋の価格を予測

分類 (Classification)

入力から出力を予測する問題設定。

出力は離散値（有限集合の要素）で得られる。

入出力の関係性は、訓練データ（ラベル付きの例）から学習する（教師あり学習）

例) スпамメール分類、手書き文字認識

クラスタリング (Clustering)

入力をいくつかのクラスタに分割する問題設定。

クラスタは、入力データ自身の性質から学習する
(教師なし学習)

例) 商品特徴による動的なカテゴリ生成

機械学習のタスク		機械学習の手法(モデル、アルゴリズム)
教師あり	分類	ロジスティック回帰 SVM 決定木 ニューラルネットワークなど
	回帰	線形回帰 リッジ回帰 決定木 ニューラルネットワークなど
教師なし	クラスタリング	k-meansなど
	次元削減	PCAなど
	など	—

“なぜ機械学習を学ぶのか”

なめらかなシステム

「ファンを増やす」

Webサービスの利用者の行動(や意図)はログをはじめとするデータとして観測できる。

データの背景を記述し学習する機械学習の手法を習得することで、個々の利用者の振る舞いに適応するデータ駆動なシステムを実現し、エンジニアリングによってファンを増やすことにつながる。

レッスン&ハンズオン

Machine Learning Crash Course | Google Developers

- レッスン&ハンズオンとして、Googleが公開している機械学習の短期集中コースを利用します
 - <https://developers.google.com/machine-learning/crash-course>
- Day3は、IntroからReducing Loss、そしてFirst Steps with TFのハンズオンまで行います。
 - 英語のテキストのみです
 - 各セッションのビデオは飛ばして構いません。後半のテキストにもほぼ同じ内容が含まれており冗長なためです(時間もギリギリなので)
 - 自分のペースで進めてください(休憩含む)
 - わからないところは逐次聞いてください！！!(重要)

1. Introduction to ML

3min。このコースのイントロダクションです。コースの趣旨を理解する意味も込めて動画を見ましょう

2. Framing

15min。機械学習の基本的な用語、モデルの種類などを学びます。

3. Descending into ML

20min。機械学習の入門として、線形回帰モデルを例に、重み、バイアス、損失や誤差などの概念を学びます。

4. Reducing Loss

60min。モデルを訓練するため、勾配降下法や学習率などの知識を学びます。

5. First Steps with TF

60min。Colab上で、tf.kerasを使った線形回帰モデルを動かすハンズオンです。NumpyとPandasのUltraquick TutorialはDay1,2でやったものと同じですのでスキップしてください

See you tomorrow!

GMOペパボ