

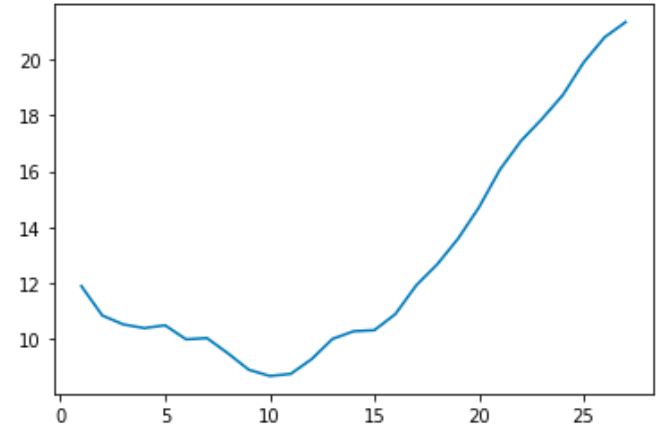
単回帰を使う

正田 備也

masada@rikkyo.ac.jp

kNNの課題の追加説明

- kを交差検証（cross validation）で決める方法
- 前回説明したcross validationの方法は、leave-one-outと呼ばれる
 - まずtest dataを除去（test setは、最後に一回、評価に使うだけ）
 - 残りのデータをtraining setとvalidation setに分割
 - 前回説明した方法では、validation setは一つの国だけ = leave-one-out
 - この一つの国を取り替えつつ、それぞれの場合の予測誤差を記録
 - これらの予測誤差の平均を求める
 - この予測誤差の平均を最小にするkの値を調べる（elbow curve）

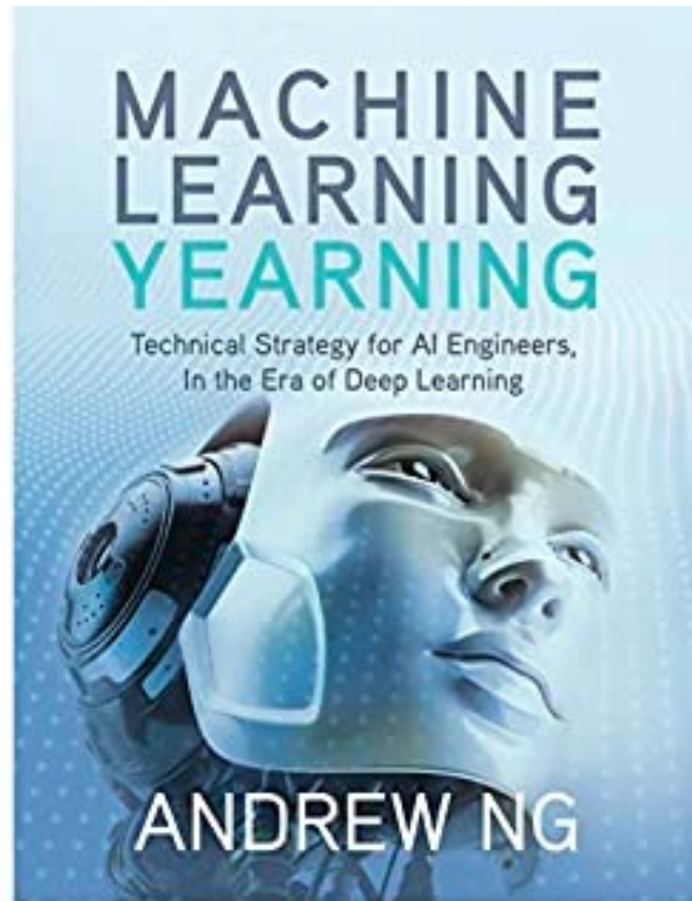


交差検証 (cross validation)

- 機械学習の実験では、しばしばデータ集合を3つに分ける

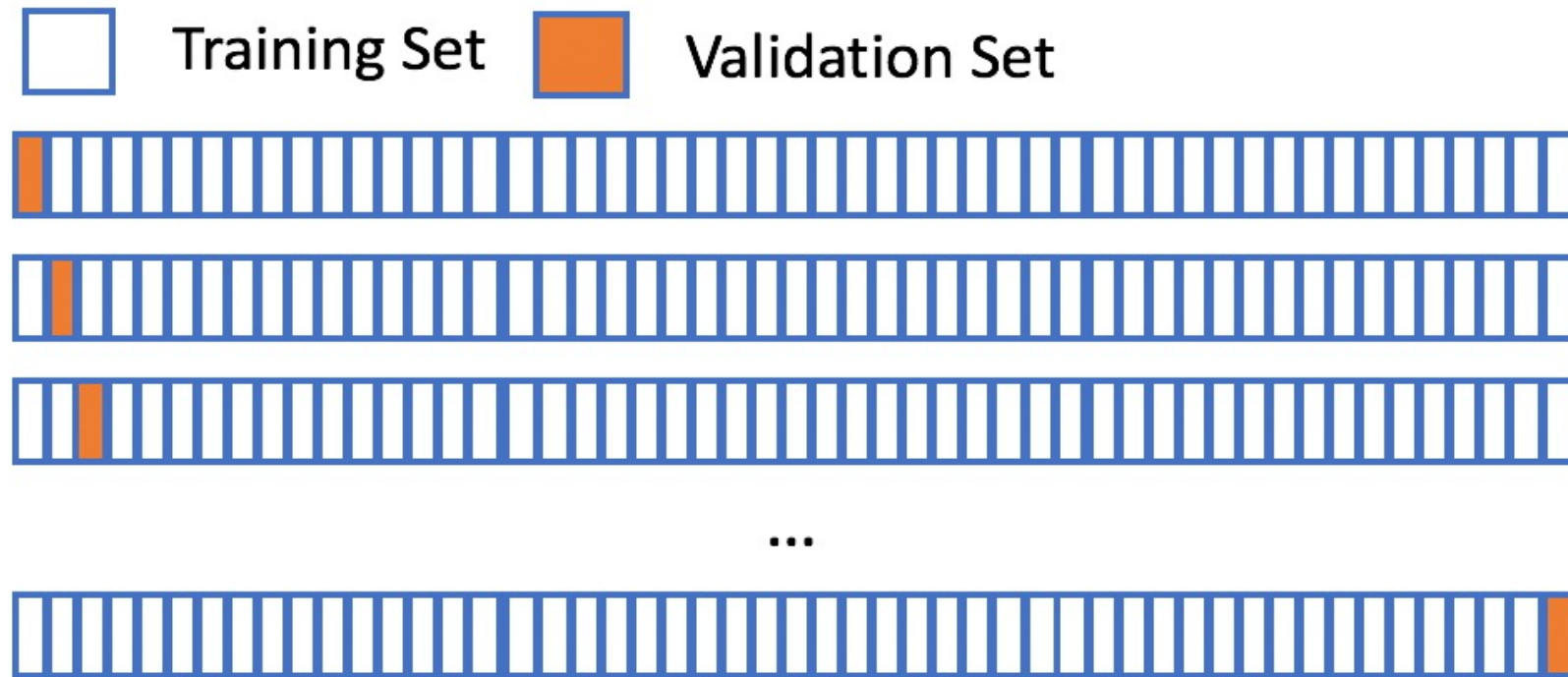
1. training set
 2. validation set (development set)
 3. test set
- } 交差検証に使う

- test setは、最後に一回、手法の最終的な評価に使うだけ
- training setとvalidation setを使って、良いモデルを選んでおく
 - test setで評価する前に、できるだけ良いモデルを選んでおく
 - validation setの取り方をいろいろ変える→交差検証



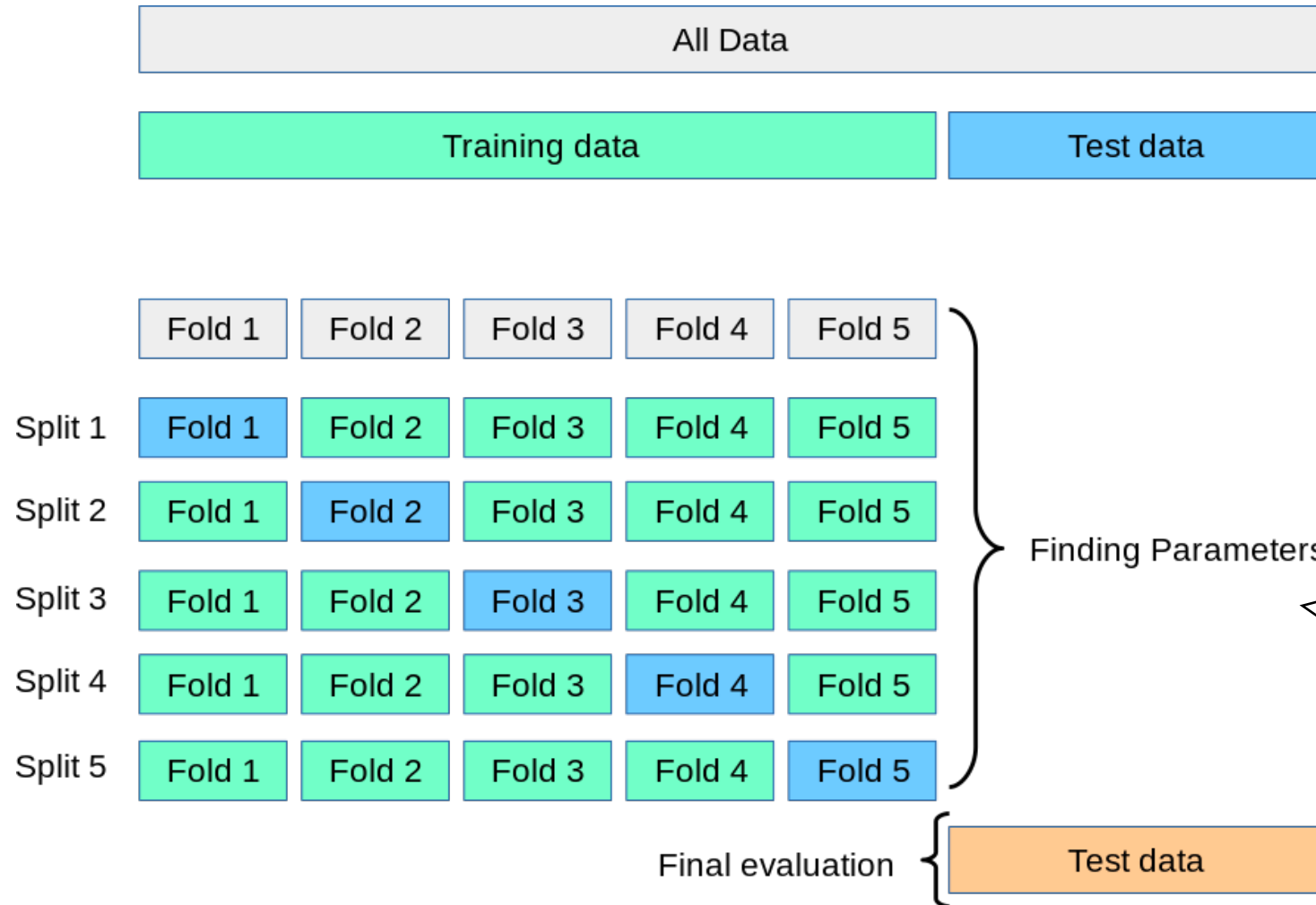
<https://www.deeplearning.ai/machine-learning-yearning/>

leave-one-out cross validation



データの個数が
少ないときに
よく使うCV

5-fold cross validation



データの個数が
多ければ、普通は
こちらを使う
(10-foldも
よく使う)

データ分割の必要性の雰囲気

- 未知データに対する予測性能を調べたい！
 - training setとは別に、test setを用意する必要がある。
 - そのtest setを最後に一回使って、手法の予測性能を測る。
- できるだけ良いモデルを探したい！
 - training setの部分集合を、validation setとして取り出す必要がある。
 - 個々のモデルをvalidation set上で評価して比較する。
 - validation setは、何通りも違う取り出し方をし、評価を繰り返す。
 - 良いモデル = 予測に良いモデル or データの「理解」に良いモデル

今日の予定

- 機械学習とは何かを説明。
- それをふまえて、単回帰を説明。
- そして、`scikit-learn`を使った単回帰の実装
 - 勾配降下法については、次回説明

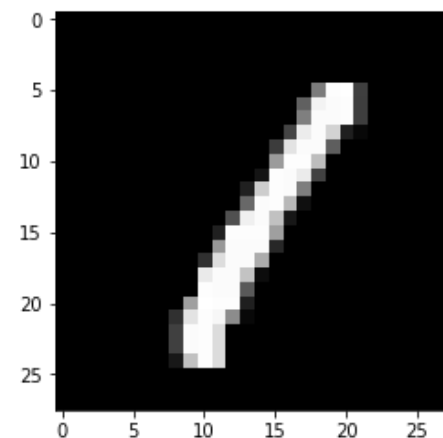
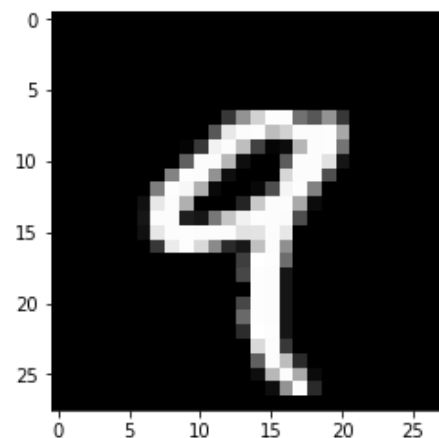
課題5

- 一人当たりの**GDP**から生活満足度を予測してみよう (p.22)
- 日本について、他の全ての国の生活満足度が既知と仮定し、生活満足度を予測しよう
- 予測は、実際の値との差の絶対値で、評価しよう
- **sklearn**の線形回帰を使おう
- 推定結果を直線のグラフとして可視化しよう
 - その際、訓練データの散布図も、グラフに重ねて描こう

機械学習とは？

例：手書き数字画像の分類 (MNISTデータセット)

- 画像に「0」から「9」までのどの数字が書いてあるかを計算機に判定させる
 - 右の画像はそれぞれ「9」「1」が正解。
- 機械学習 = 計算機に学習させる
 - どうやって？



label = 5



label = 0



label = 4



label = 1



label = 9



label = 2



label = 1



label = 3



label = 1



label = 4



label = 3



label = 5



label = 3



label = 6



label = 1



label = 7



label = 2



label = 8



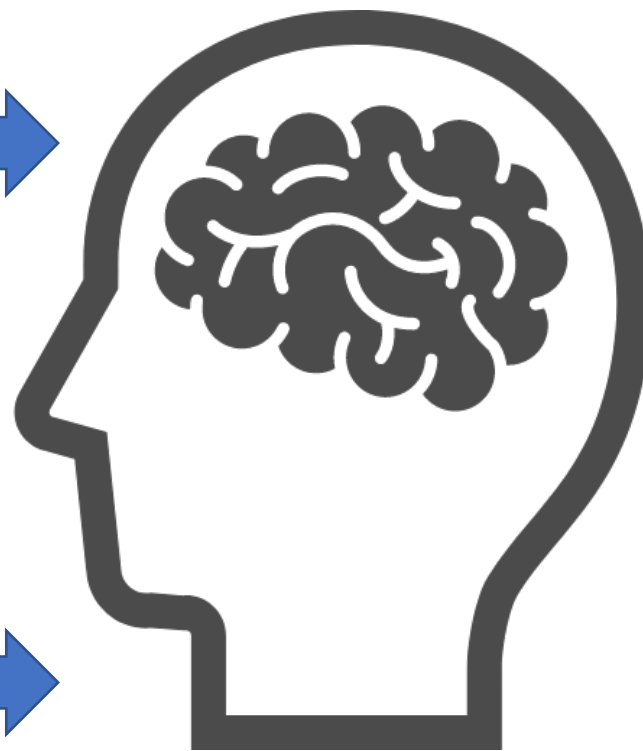
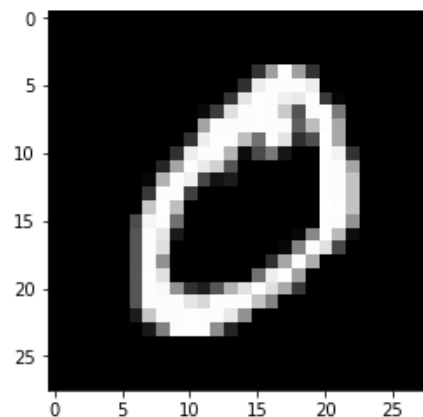
label = 6



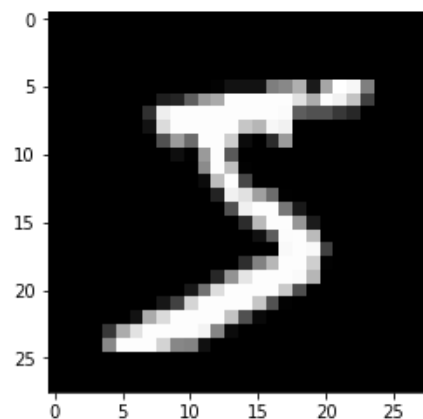
label = 9



人間はすぐ答えが分かる
(すでに学習済みなので)

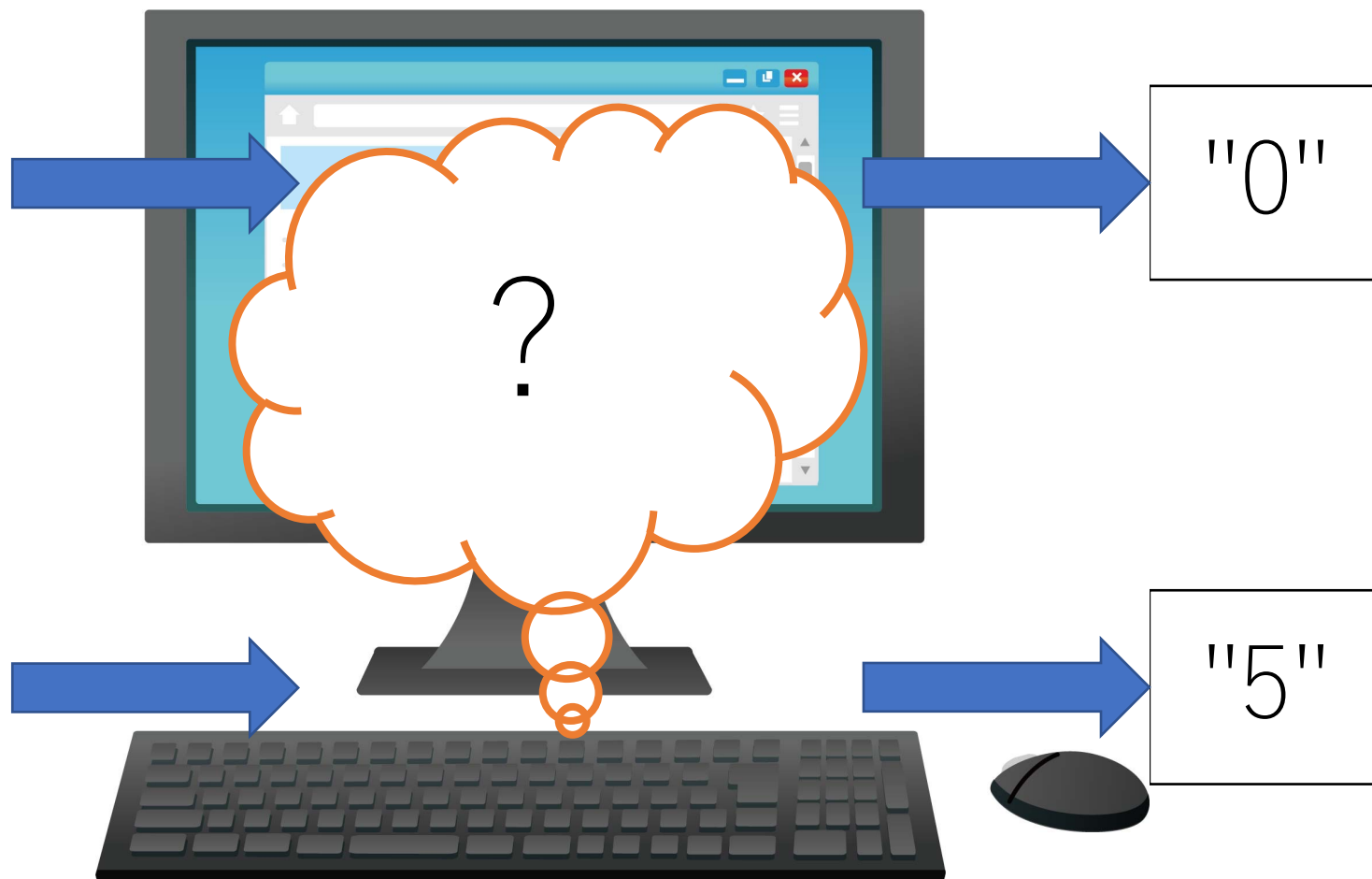
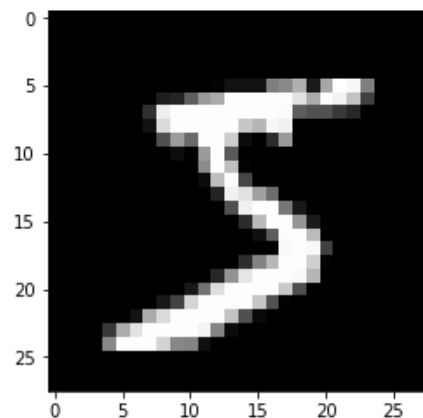
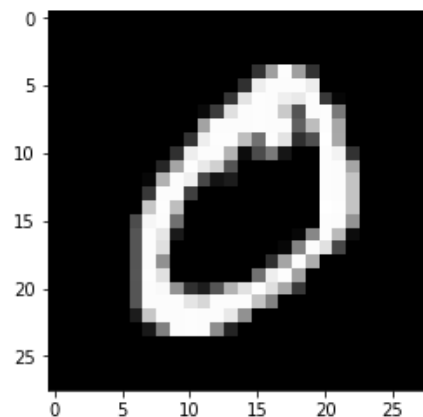


"0"



"5"

計算機に答えを当てさせるには？



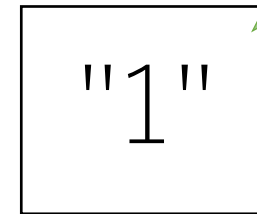
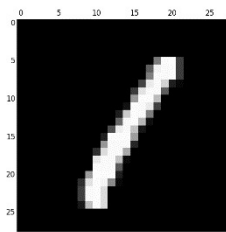
分類(classification)

- データを複数のクラスへと分ける問題
 - 入力：分類したいデータ
 - 出力：どのクラスへ属させるべきかを示す値 = 「クラスラベル」
- どんな入力データにも、ちゃんと正解のラベルを出して欲しい
 - 欲しいものは図の「？」の部分。



機械学習とは

- 学習：箱から出てくる値が目標の値になるような箱の中身（「？」の部分）を見つける
- 箱の中身：入力から出力を得るための何らかの計算方法
- 機械学習：箱の中身を計算機に見つけさせる



目標の値

機械学習とは

良い関数を計算機に見つけてもらうこと。

関数 = 値を入れると、その値に応じて何かの値が出てくる箱

良い関数 = どんな値を入れても、出てくる値が目標の値に近い箱

計算機に見つけてもらうのであって、
人間が試行錯誤して見つけるのではない。

もう少しテクニカルに言うと・・・

関数の、無数にあるパラメータ設定の中から
良い設定を計算機で見つけること。

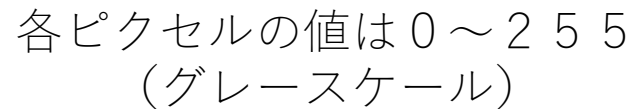
良い設定 = どんな入力に対しても（見たことがない入力に対しても）
望みどおりの出力が得られる

関数を選ぶ範囲

- 関数を選ぶ範囲は、あらかじめ決めておく
- どう決める？
 - 入力データのフォーマットを決める
 - 入力データをどのように数値化するか？
 - 出力データのフォーマットを決める
 - 出力データをどのように数値化するか？
 - 計算式の「かたち」を決める
 - 計算式はパラメータ（自由に変更できる部分）を含む。
 - このパラメータを計算機で決めるのが、機械学習。

- ベクトルにする

⇒ $28 \times 28 = 784$ 次元ベクトル



20

例：出力記号をどうやって数値化する？

- "0", "1", "2", ... はラベルであって、数値ではない
- ベクトルにする

例) 10種類のラベルがある場合 ("0", "1", "2", ..., "9" の10種類)

⇒ 10次元ベクトルとして数値化

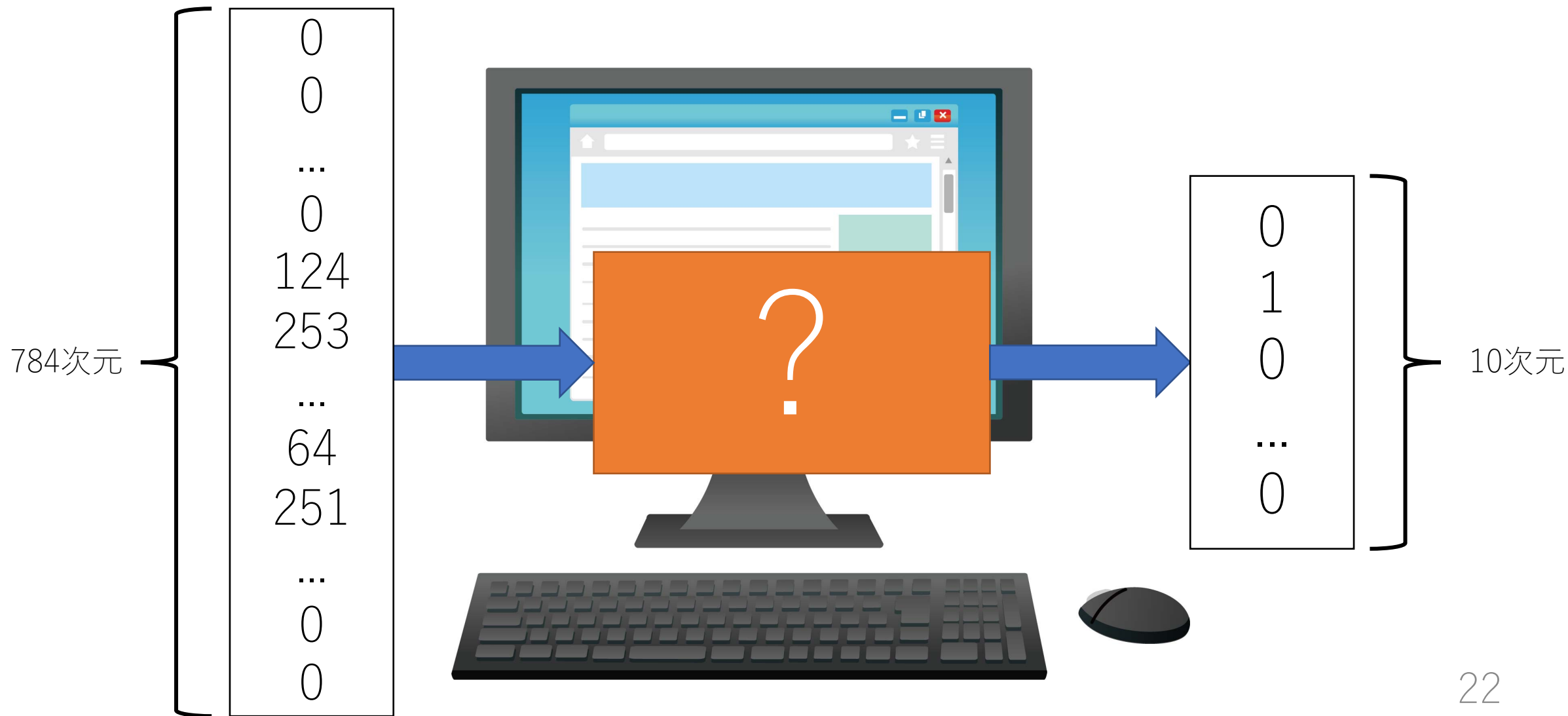
- "3" というラベルの場合:

$[0 \ 0 \ 0 \ 1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0]$

こういうものを
one-hot vector と呼ぶ。

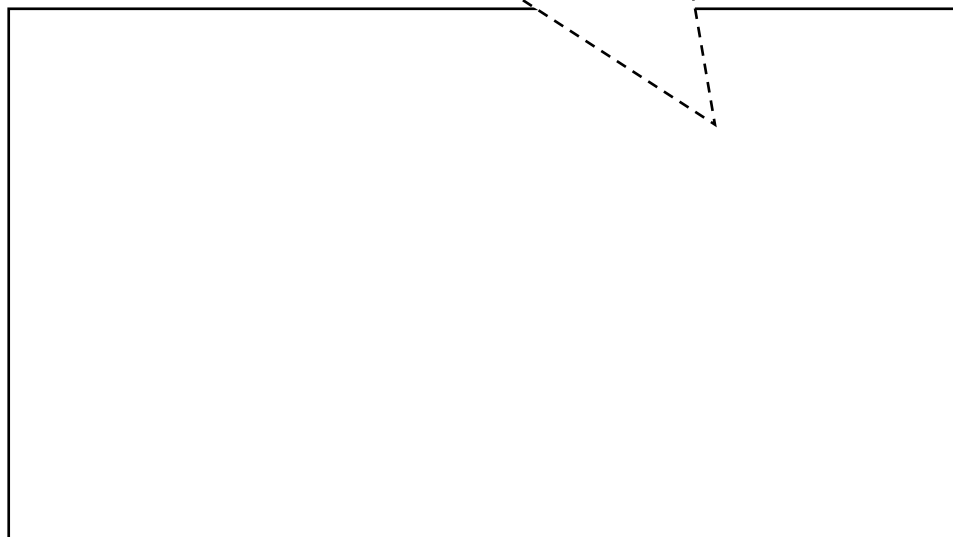
手書き数字画像の分類の問題設定

入力データと出力データをベクトル化した後の状態



(鋭い方ならこう思いつかれるかも・・・)

「こういう行列を見つければいいのでは？」



•

0
0
...
0
124
253
...
64
251
...
0
0

\equiv

0
1
0
...
0

ちょっと複雑すぎるので・・・

- 入力も出力も1個の数値である場合を考える
 - 1個の数値の入力を
 - 1個の数値の出力に
- こういう変換のなかだけから選ぶ



単回帰 (入力も出力も1次元ベクトル)

例題1



- ある箱に、
 - 1という数値を入れたら3という数値が出てきてほしい。
 - 2という数値を入れたら4という数値が出てきてほしい。
 - 3という数値を入れたら5という数値が出てきてほしい。
 - 4という数値を入れたら6という数値が出てきてほしい。
- 箱の中でどういう計算をすればいいのでしょうか？

例題2



- ある箱に、
 - 1という数値を入れたら3という数値が出てきてほしい。
 - 2という数値を入れたら8という数値が出てきてほしい。
 - 3という数値を入れたら13という数値が出てきてほしい。
 - 4という数値を入れたら18という数値が出てきてほしい。
- 箱の中でどういう計算をすればいいのでしょうか？

例題3



- ある箱に、
 - 1という数値を入れたら2という数値が出てきてほしい。
 - 2という数値を入れたら-1という数値が出てきてほしい。
 - 3という数値を入れたら-4という数値が出てきてほしい。
 - 4という数値を入れたら-7という数値が出てきてほしい。
- 箱の中でどういう計算をすればいいのでしょうか？

箱を関数だと思う

- 「 x を入れたら y が出てきてほしい」
 - $y = f(x)$ と書ける
 - $f(x)$ は x の関数（関数：行き先がひとつに決まる）
- 例題を解くことで何をしていたか？
 - 関数 $f(x)$ の式を求めている

例題4（ちょっとデータ数が多い）

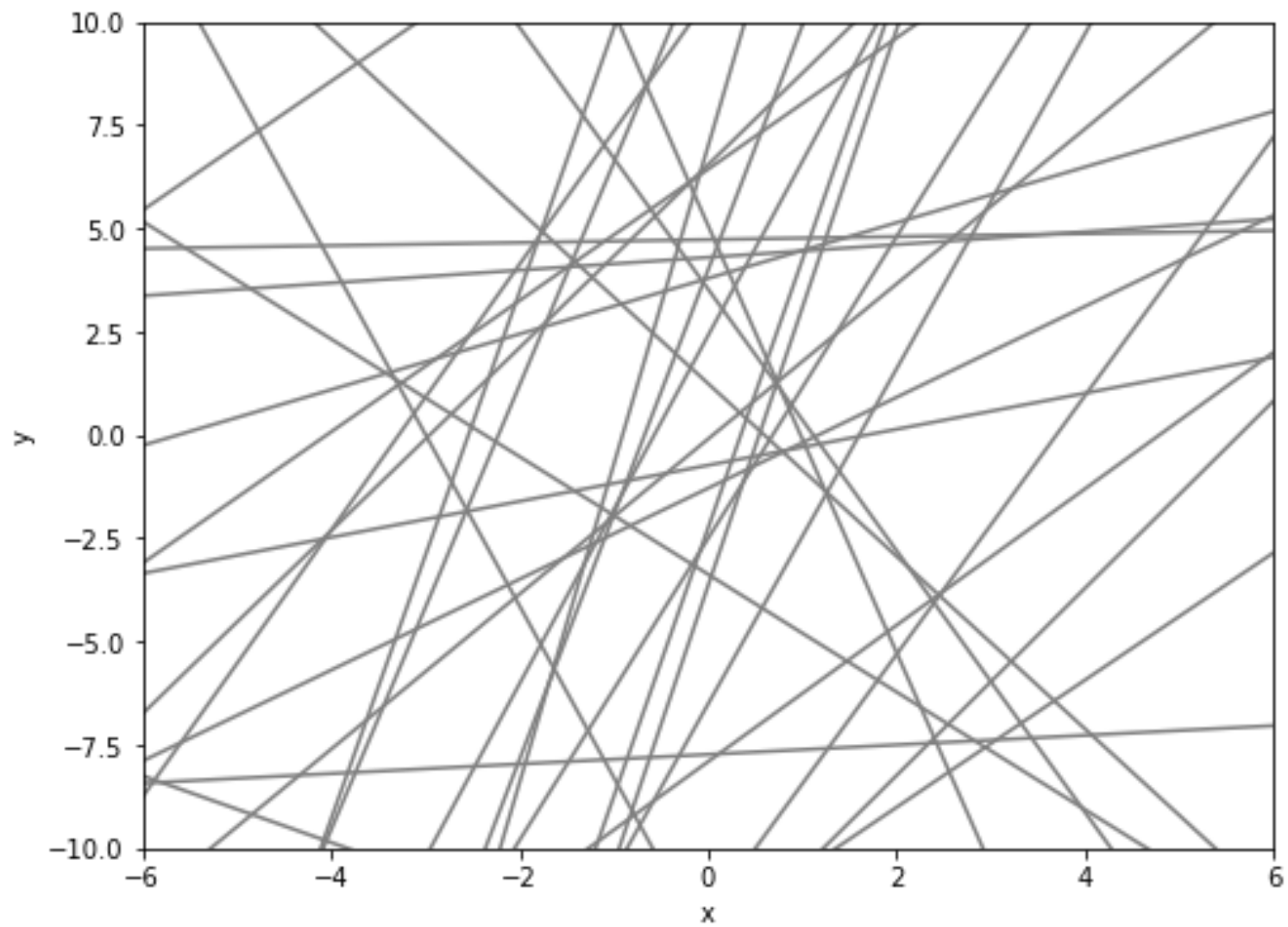
- ある箱に、
 - 2.0という数値を入れたら-4.0という数値が出てきてほしい。
 - 1.2という数値を入れたら-2.0という数値が出てきてほしい。
 - -3.0という数値を入れたら5.5という数値が出てきてほしい。
 - 0.5という数値を入れたら-0.9という数値が出てきてほしい。
 - -4.1という数値を入れたら8.3という数値が出てきてほしい。
 - -1.5という数値を入れたら2.9という数値が出てきてほしい。
 - -2.5という数値を入れたら4.9という数値が出てきてほしい。
 - 6.2という数値を入れたら-12.2という数値が出てきてほしい。
- 箱の中でどういう計算をすればいいのでしょうか？

モデルを設定する

- モデル = 箱の中身を数式で表したもの
- ここでは、関数のかたちを一次式に設定（一番簡単なので）

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

- これで関数を選ぶ範囲が決まった。（線形回帰）
- そして「 a と b をいくらにすればいいか？」という問題を解く
 - あとはこの範囲の中で、できるだけいい関数を選べばよい。



$f(x) = ax + b$ は様々でありうる

例題4の続き

$$2a + b = -4$$

$$1.2a + b = -2$$

$$-3a + b = 5.5$$

$$0.5a + b = -0.9$$

$$-4.1a + b = 8.3$$

$$-1.5a + b = 2.9$$

$$-2.5a + b = 4.9$$

$$6.2a + b = -12.2$$

解けない方程式

- 未知数は a と b の二つだけ
- なのに等式がたくさんある
 - 式が2つだったら解ける
- つまり・・・解はない
- 困った！

例題4で式が2つだけだったら・・・

$$2a + b = -4$$

$$1.2a + b = -2$$

- これは普通の連立一次方程式
 - 答えは $a = -2.5, b = 1$

問題を変える

目標の値に、一致させるのではなく、近づける

- 未知数は a と b の二つだけ
- なのに等式がたくさんある（式が2つだったら解ける）
- つまり・・・解はない
- そこで・・・値がズレていてもいいことにする
 - 一致しなくても、十分に近ければ良い、と考える

例題4の続き (簡単のために式を3つにした)

$$2a + b \approx -4$$

$$1.2a + b \approx -2$$

$$-3a + b \approx 5.5$$

完全に一致しなくてもいい、
という意味。

残差

- $2a + b \approx -4$
 - 残差は $(2a + b) - (-4)$
- $1.2a + b \approx -2$
 - 残差は $(1.2a + b) - (-2)$
- $-3a + b \approx 5.5$
 - 残差は $(-3a + b) - 5.5$

残差の2乗

2乗するとプラスかマイナスかが関係なくなる

- $2a + b \approx -4$

- 残差の2乗は $\{(2a + b) - (-4)\}^2$

- $1.2a + b \approx -2$

- 残差の2乗は $\{(1.2a + b) - (-2)\}^2$

- $-3a + b \approx 5.5$

- 残差の2乗は $\{(-3a + b) - 5.5\}^2$

問題を解く方針

- 残差の2乗の和を最小にすることで、関数

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

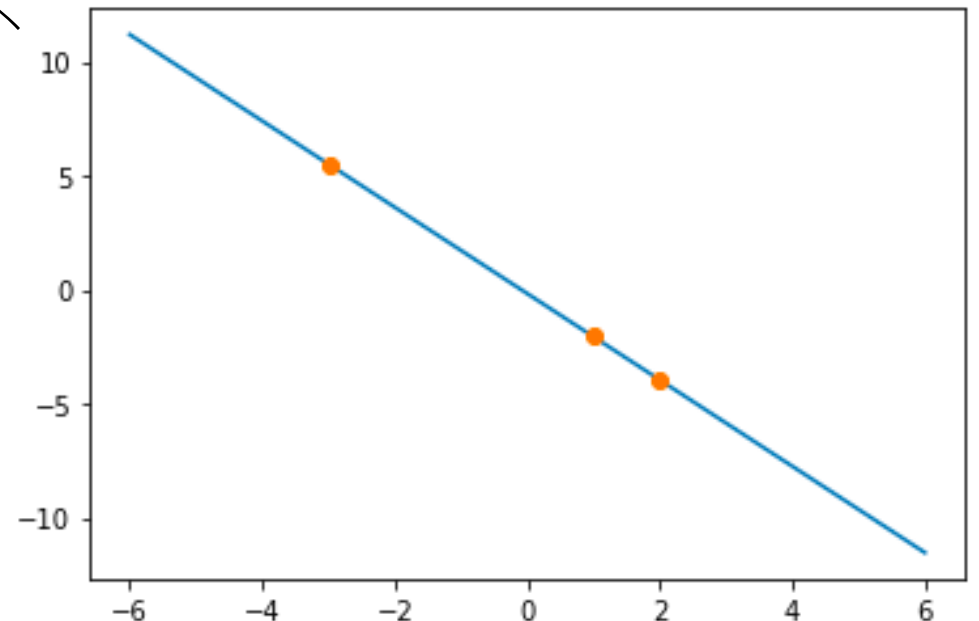
の a と b を求める

- つまり、 $ax + b$ というモデルのパラメータを求める
 - 例題4の場合なら、次の式の値を最小にする a と b を求める

$$\{(2a + b) - (-4)\}^2 + \{(1.2a + b) - (-2)\}^2 + \{(-3a + b) - 5.5\}^2$$

問題の解き方のイメージ

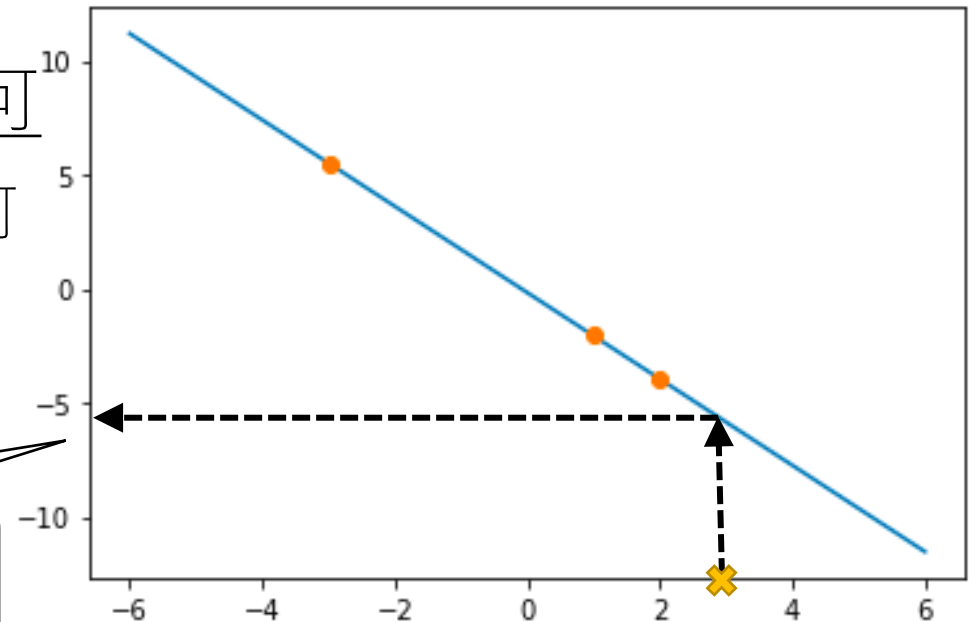
- 入力値と出力値のペアを表す点がたくさんある
 - 入力値が x 座標、出力値が y 座標。
- それらの点にぴったり合う直線を引く
- こういう問題を「線形回帰」という
 - 「線形」 \equiv 「まっすぐ」



線形回帰による予測

- 直線が求まれば . . .
 - つまり関数の一つ選べば . . .
- 任意の入力値について出力値を予測可
 - 見たことない入力値でも出力値を計算可
 - 機械学習は予測に使える！

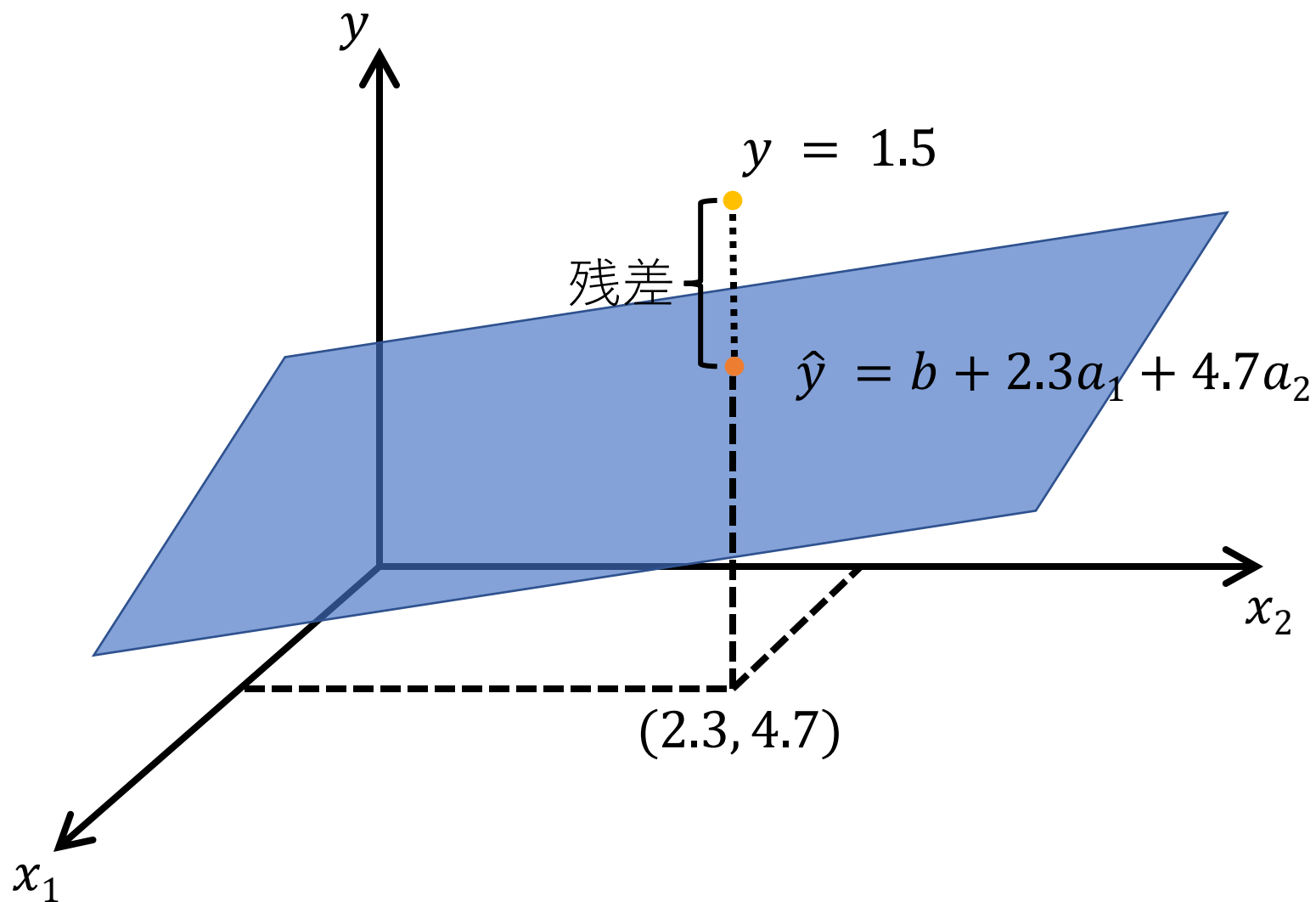
未知の入力値について
予測された出力値



単回帰

- 回帰のうち入力値が1つのものを単回帰と呼ぶ
 - 入力値が2つの場合だと . . .
 - 線形回帰は、平面をデータ点にフィットさせる
 - 「線形」 = 「まったいら（曲がっていない）」

入力値が2つの線形回帰



機械学習とは . . .

- 入力値と出力値のペアが大量に与えられているとき
- 入力値から出力値を計算する方法（関数）を計算機に探させる。
 - 関数は特定の形式を持っていると仮定する（例：1次式）
 - 関数を探す範囲は、すでに決まっていると仮定
 - 目標の出力値からのズレの測り方を決める（例：残差の2乗の和）
 - そしてズレを機械に小さくさせる = 機械に関数のパラメータを推定させる
 - 例：一次式の係数を、残差平方和ができるだけ小さくなるように決める

逆に言えば・・・

「”良い関数を見つける”という問題へ
落としこめない問題は、
機械学習では解けません」

- 解きたい問題を、「良い関数を見つける」問題として、どうにかして、言い換えてみてください

機械学習の問題として解くのに必要なもの

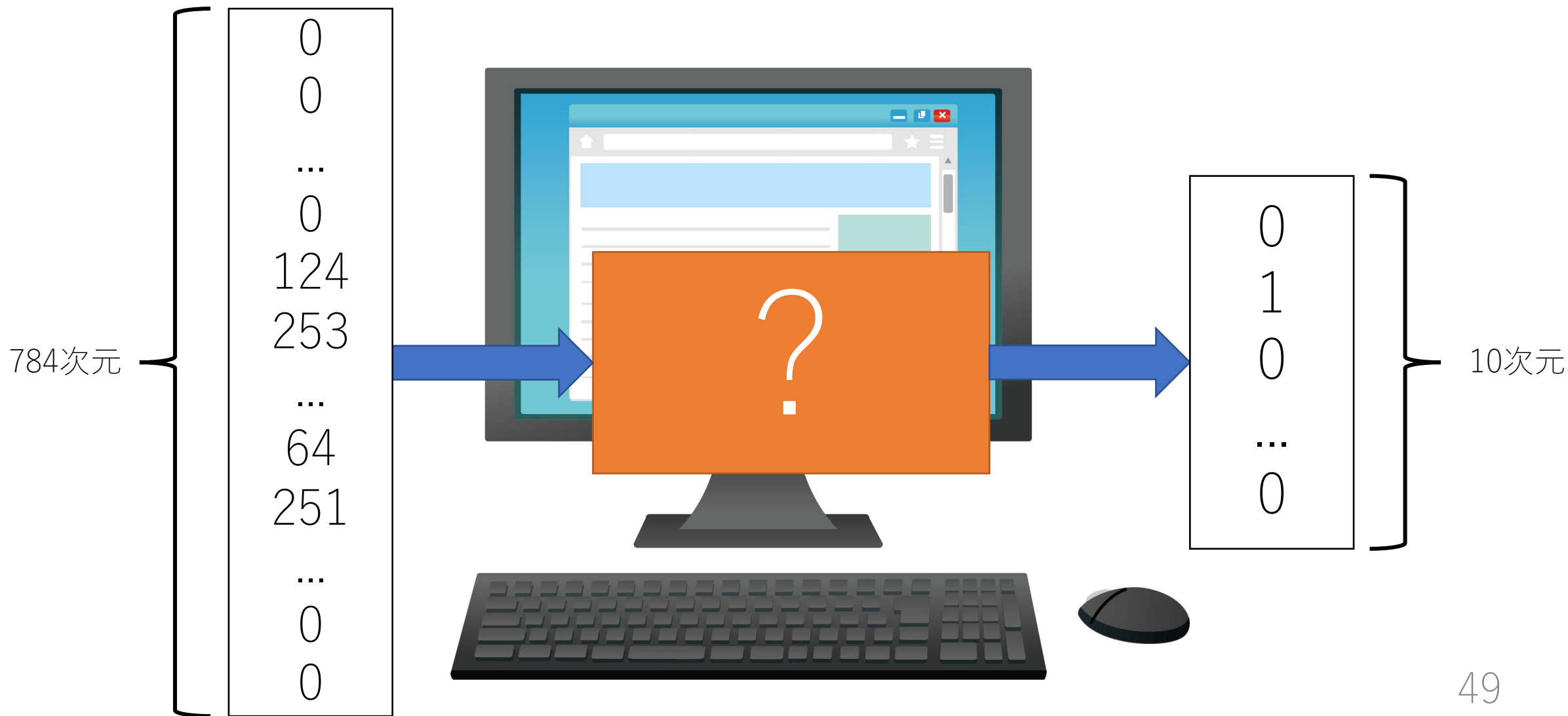
- 数値化された入力とそれに対応する目標値としての出力の、ペア
 - このペアは、たくさんあればあるほど、良い
- 入力を目標の出力へ変換する関数の式の形
 - 式の形を固定すれば、あとはパラメータの値をうまく決めればよい
- 関数の出力値と目標値とのズレの測り方
 - 残差の2乗の和は、一つの候補
 - このズレが損失関数。モデルのパラメータを変えると、ズレも変わる。

損失関数

- モデルの出力値とターゲットとのズレを表す関数
 - 線形回帰では残差の2乗の和をよく使う
- モデルパラメータを動かすと損失関数の値も動く
 - 単回帰では、傾きと切片がパラメータ
- モデルパラメータを動かして損失関数を最小化する
 - これがいわゆる「学習」

手書き数字画像の分類の問題設定

入力データと出力データをベクトル化した後の状態



演習

$$\{(2a + b) - (-4)\}^2 + \{(1.2a + b) - (-2)\}^2 + \{(-3a + b) - 5.5\}^2$$

上の残差の 2 乗和を最小にする a と b の値を求めよ。

实践

scikit-learnを使う

- 機械学習のライブラリ
- 線形回帰だけでなく、いろいろ入っている
 - SVMやk-meansなど
 - 手法だけでなく、実験に使えるデータセットも入っている
- データの型としてnumpyのarrayが使える
 - 便利！

説明用の例題

- 関数 $y = ax + b$ によってモデル化される「箱」に
 - 2.0を入れたら-4.0が出てきてほしい
 - 1.2を入れたら-2.0が出てきてほしい
 - -3.0を入れたら5.5が出てきてほしい
 - -1.0を入れたら2.0が出てきてほしい
- パラメータ a と b をいくらにすればいいのでしょうか？

最小二乗法による線形回帰（エラーが出る）

```
from sklearn import linear_model

reg = linear_model.LinearRegression() #線形回帰を準備
x = [2.0, 1.2, -3.0, -1.0]
y = [-4.0, -2.0, 5.5, 2.0]
reg.fit(x, y) #最小2乗法を実行
a = reg.coef_
b = reg.intercept_
print(a, b)
```

最小二乗法による線形回帰（エラーが出ない）

```
from sklearn import linear_model

reg = linear_model.LinearRegression() #線形回帰を準備
x = [[2.0], [1.0], [-3.0], [-1.0]]
y = [-4.0, -2.0, 5.5, 2.0]
reg.fit(x, y) #最小2乗法を実行
a = reg.coef_
b = reg.intercept_
print(a, b)
```

なぜこうなっている？

- scikit-learnでは入力がベクトルであると想定
 - 入力が1個の数値であっても、1次元のベクトルとして扱わないといけない



最小二乗法による線形回帰

```
import numpy as np
from sklearn import linear_model

reg = linear_model.LinearRegression() #線形回帰を準備
x = [2.0, 1.0, -3.0, -1.0]
x = np.array(x).reshape(-1, 1)
y = [-4.0, -2.0, 5.5, 2.0]
reg.fit(x, y) #最小2乗法を実行
a = reg.coef_
b = reg.intercept_
print(a, b)
```