慶應義塾大学理工学部

機械学習基礎

第４回　順伝播型ニューラルネット

情報工学科　教授　杉浦孔明
komei.sugiura@keio.jp

本講義の到達目標と今回の授業の狙い

本講義の到達目標

- DNNの基礎理論と実装の関係を理解する
- 種々のDNNをコーディングできる

今回の授業の狙い

- 順伝播型ニューラルネットの基礎を習得する

- 出席確認： K-LMS上の機械学習基礎のMainページへアクセス

順伝播型ニューラルネット

線形回帰
1入力1出力の場合

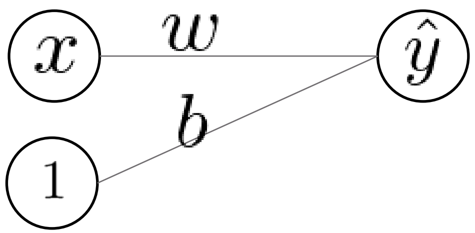
■ 前回扱った線形モデル

$$\hat{y} = wx + b$$

図で書くと↓

入力
(input)

出力
(output)



常に値が1である
ノード

線形回帰
２入力１出力の場合

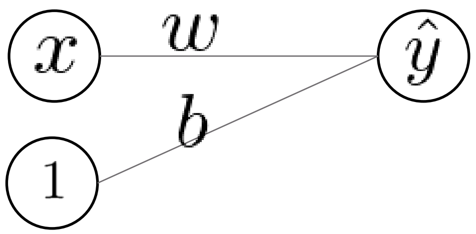
- 前回扱った線形モデル

$$\hat{y} = wx + b$$

図で書くと↓

入力
(input)

出力
(output)



常に値が１である
ノード

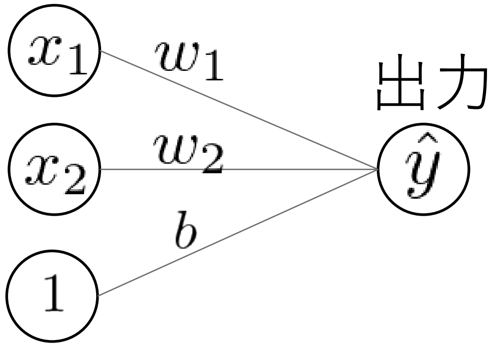
- ２次元の入力

$$\hat{y} = w_1 x_1 + w_2 x_2 + b$$

重み（weight）

バイアス（bias）

入力



出力

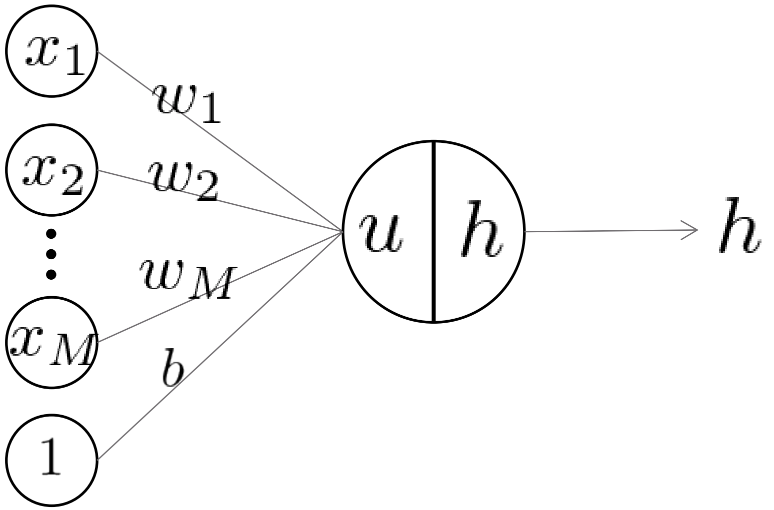
基本的なニューラルネット ユニットとは

■ ユニット

$$u = w_1 x_1 + w_2 x_2 + \cdots + w_M x_M + b$$

$$h = f(u)$$

h:活性化関数（非線形変換）

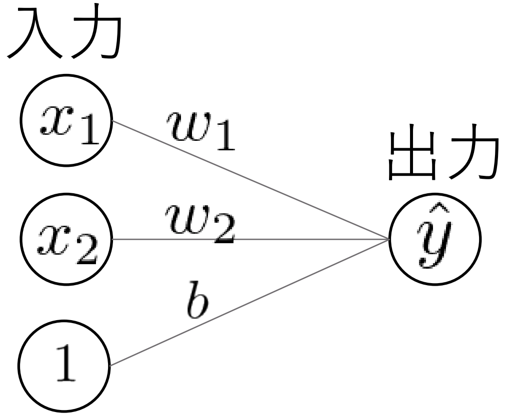


■ ２次元の入力

$$\hat{y} = w_1 x_1 + w_2 x_2 + b$$

重み（weight）

バイアス（bias）



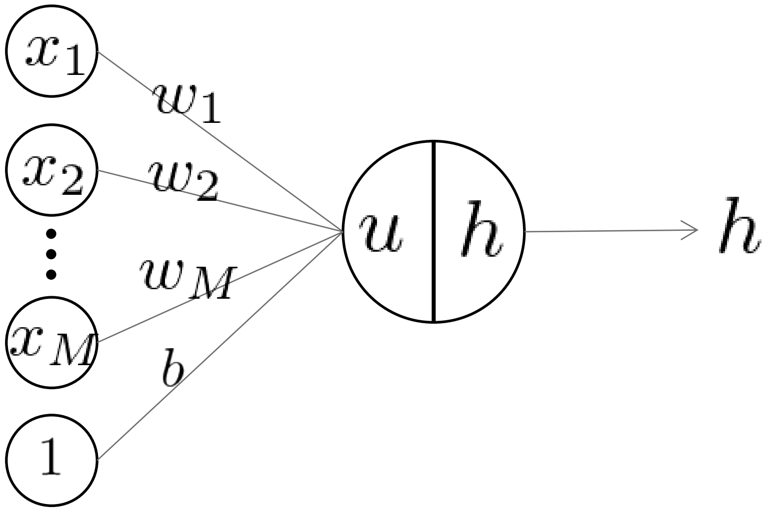
基本的なニューラルネット
活性化関数とは

- ユニット

$$u = w_1x_1 + w_2x_2 + \cdots + w_Mx_M + b$$

中間層

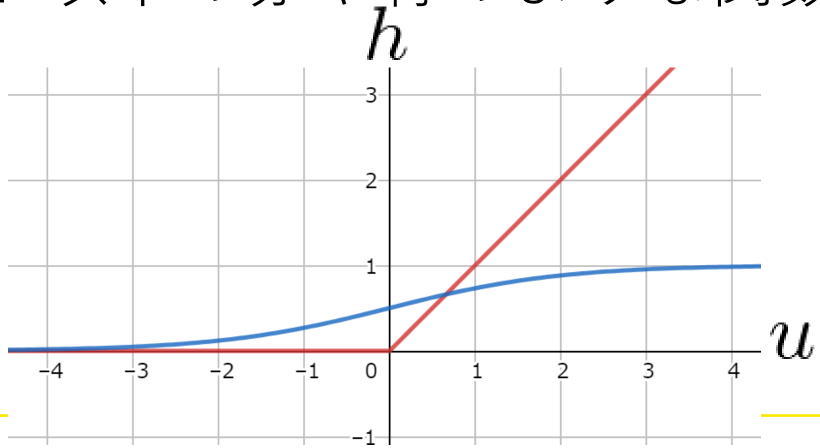
$$h = f(u)$$



$$\left.\begin{array}{l}\text{重み } w_j \\ \text{バイアス } b\end{array}\right\}\text{パラメータ}$$

- **活性化関数**（activation function）
  - 非線形変換を行う
  - 以下の赤や青のような関数

例

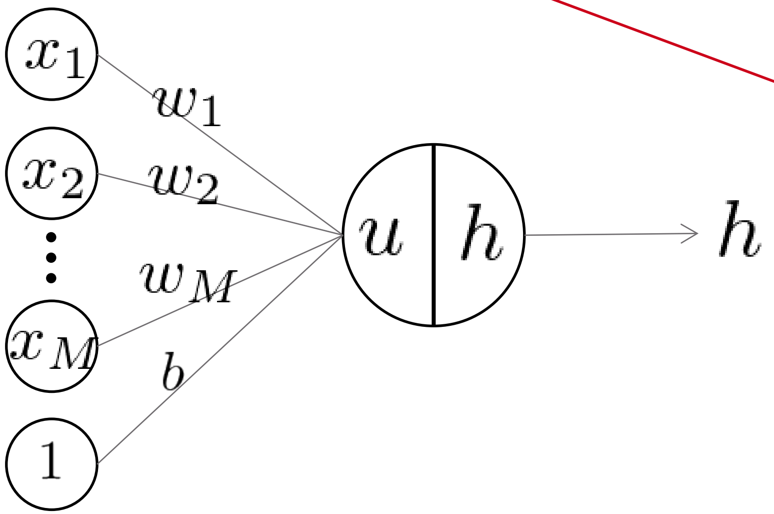


基本的なニューラルネット 複数のユニットを持つ場合

■ ユニット

$$u = w_1 x_1 + w_2 x_2 + \cdots + w_M x_M + b$$

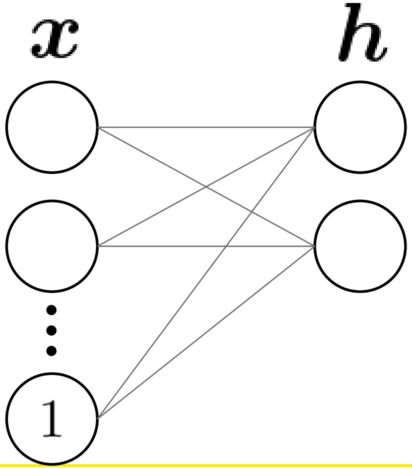
$$h = f(u)$$



■ ユニットが 2 つの場合

$$\begin{cases} u_1 = w_{11} x_1 + w_{12} x_2 + \cdots + w_{1M} x_M + b_1 \\ u_2 = w_{21} x_1 + w_{22} x_2 + \cdots + w_{2M} x_M + b_2 \end{cases}$$

$$\begin{cases} h_1 = f(u_1) \\ h_2 = f(u_2) \end{cases}$$



基本的なニューラルネット
入出力関係の行列表現

行列表現

$$\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1M} & b_1 \\ w_{21} & w_{22} & \cdots & w_{2M} & b_2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_M \\ 1 \end{bmatrix}$$

$$\boldsymbol{u} = \underline{W\boldsymbol{x}}$$

$\boldsymbol{x}$ に 1 が入っているものと考えて、
バイアスを陽に書かない

$$\boldsymbol{h} = f(\boldsymbol{u})$$

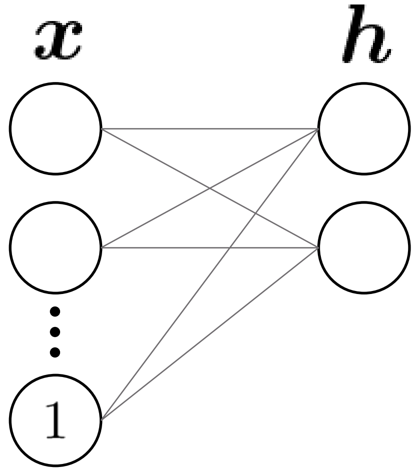
まとめて書く

■ ユニットが 2 つの場合

$$u_1 = w_{11}x_1 + w_{12}x_2 + \cdots + w_{1M}x_M + b_1$$

$$u_2 = w_{21}x_1 + w_{22}x_2 + \cdots + w_{2M}x_M + b_2$$

$$\left\{ \begin{array}{l} h_1 = f(u_1) \\ h_2 = f(u_2) \end{array} \right.$$



基本的なニューラルネット
３層ニューラルネット

行列表現

$$\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1M} & b_1 \\ w_{21} & w_{22} & \cdots & w_{2M} & b_2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_M \\ 1 \end{bmatrix}$$

$$\boldsymbol{u} = \underline{W\boldsymbol{x}}$$

Wを重みという

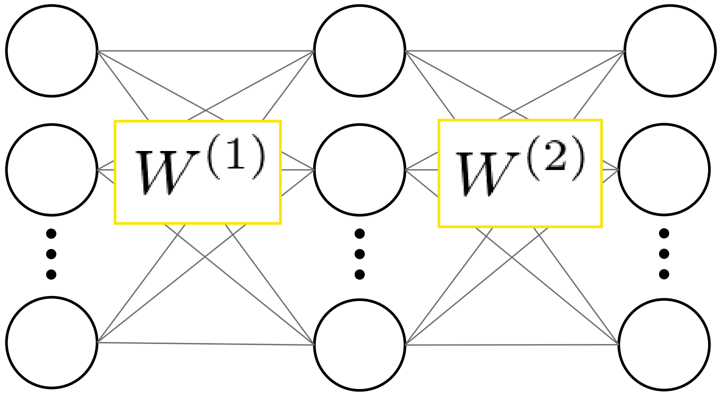
$\boldsymbol{x}$ に１が入っているものと考えて、
バイアスを陽に書かない

$$\boldsymbol{h} = f(\boldsymbol{u})$$

- ３層ニューラルネット

$$\boldsymbol{h} = f^{(1)}(W^{(1)}\boldsymbol{x})$$
$$\hat{\boldsymbol{y}} = f^{(2)}(W^{(2)}\boldsymbol{h})$$

入力層 $\boldsymbol{x}$　中間層 $\boldsymbol{h}$　出力層 $\hat{\boldsymbol{y}}$



基本的なニューラルネット
中間層とは

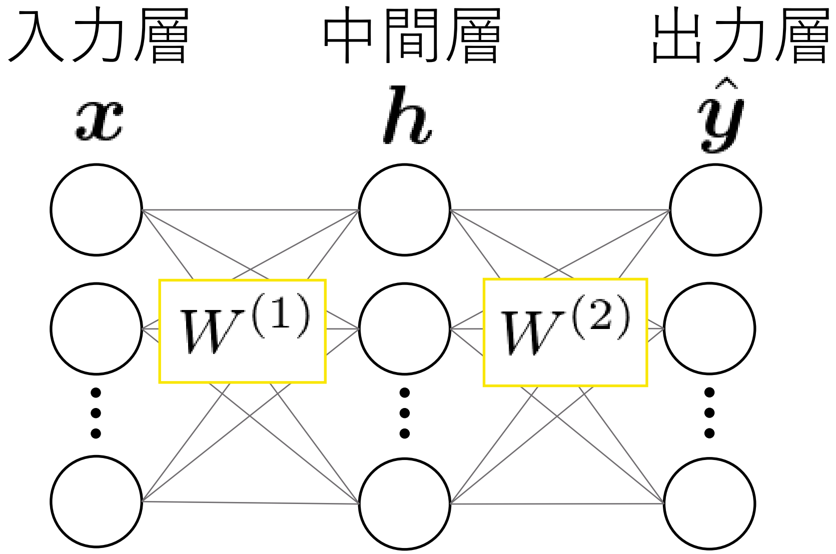
- **中間層**（隠れ層, hidden layer）

- $W^{(1)}$：１つ目の中間層への重み
- $f^{(1)}$ ：１つ目の中間層の活性化関数

- $W^{(2)}, f^{(2)}$は出力層に関するもの

- ３層ニューラルネット

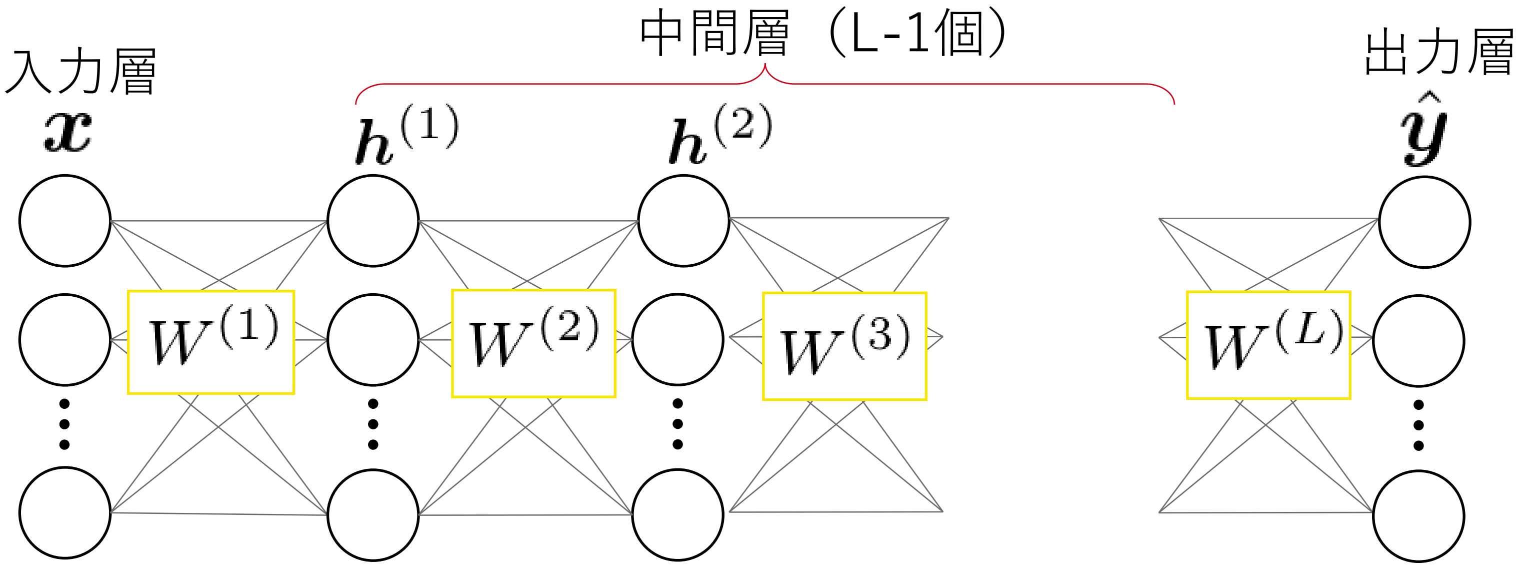
$$\boldsymbol{h} = f^{(1)}(W^{(1)}\boldsymbol{x})$$
$$\hat{\boldsymbol{y}} = f^{(2)}(W^{(2)}\boldsymbol{h})$$



順伝播型ニューラルネット

■ **順伝播型ニューラルネット**（feed-forward neural network; FFNN）

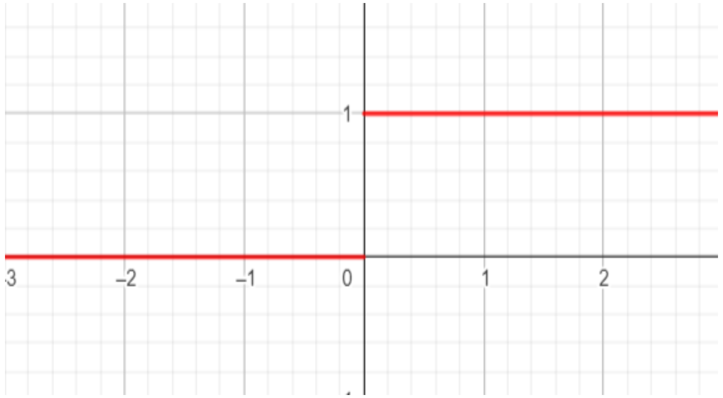
$$\boldsymbol{h}^{(1)} = f^{(1)}(W^{(1)}\boldsymbol{x})$$
$$\boldsymbol{h}^{(2)} = f^{(2)}(W^{(2)}\boldsymbol{h}^{(1)}) \xrightarrow{\text{一般化すると}} \boldsymbol{h}^{(l)} = f^{(l)}(W^{(l)}\boldsymbol{h}^{(l-1)})$$
$$\hat{\boldsymbol{y}} = f^{(L)}(W^{(L)}\boldsymbol{h}^{(L-1)})$$



活性化関数の例

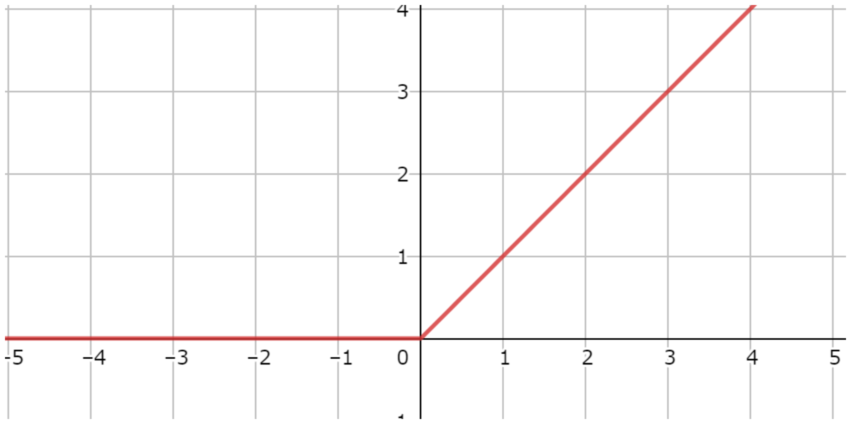
- ステップ関数

$$f(x)=\begin{cases}1 & (x\geq 0)\\ 0 & (x<0)\end{cases}$$



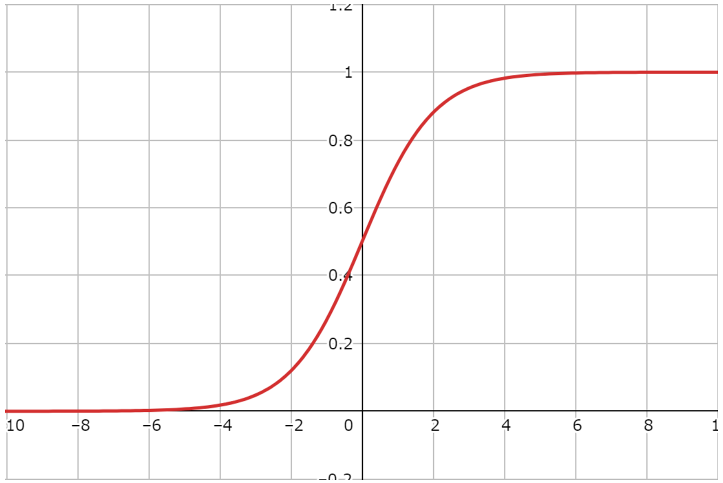
- 正規化線形関数 (ReLU)
  - 「レル」と発音

$$\mathrm{ReLU}(x)=\max\{x,0\}$$



- ロジスティックシグモイド関数

$$\sigma(x)=\frac{1}{1+e^{-x}}$$



口語ではシグモイド関数と呼ばれるが、シグモイド関数とは本来S字関数（tanhなどを含む）を意味する

ニューラルネットによる回帰
例題：大気汚染物質の濃度を予測したい

- 観測データを集める

ID	濃度（今）	風速（今）	濃度（未来）
	$\boldsymbol{x}_n$		y_n
1	5	2.0	4
2	7	1.2	5
3	10	1.6	11
	…	…	…
999	10	1.8	10
1000	9	2.6	10
新規	**8**	**1.8**	**???**

1. 訓練集合を構築する

$$\mathcal{D} = \{(\boldsymbol{x}_n, y_n) | n = 1, 2, ..., N)\}$$

2. 損失関数を最小化するパラメータを反復的に求める

$$E(\underline{\boldsymbol{w}}) = \frac{1}{2}\sum_{n=1}^{N}(y_n - \hat{y}_n)^2 \to \min$$

重みやバイアスをまとめたもの

ニューラルネットによる２値分類
例題

- 画像を「かぼちゃ」か「かぼちゃ以外」に分けたい

ラベル：１

ラベル：０

- 正解ラベルは１または０
- $\hat{\boldsymbol{y}}$ を予測するのではなく、$P(\hat{y}=1|\boldsymbol{x})$ を予測する

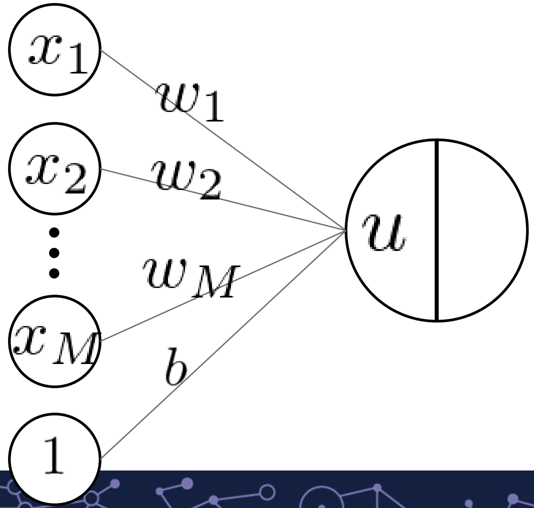
入力された画像に対し、予測ラベルが1である確率の予測値

$$\begin{cases} P(\text{“pumpkin”}) = 0.98 \\ P(\text{“not pumpkin”}) = 0.02 \end{cases}$$

ロジスティック回帰との関係

ロジスティック回帰
（logistic regression）：
ロジット（logit）uをxの線形関数としてモデル化

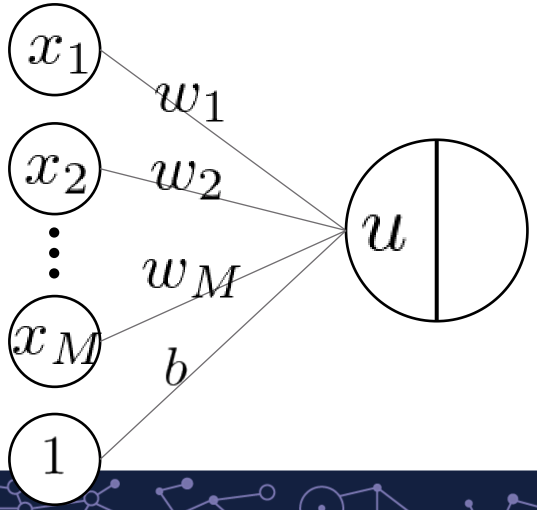
$$u = w_1 x_1 + w_2 x_2 + \cdots + w_M x_M + b$$



ロジスティック回帰との関係

ロジスティック回帰
（logistic regression）：
ロジット（logit）uをxの線形関数としてモデル化

$$u = w_1 x_1 + w_2 x_2 + \cdots + w_M x_M + b$$



用語

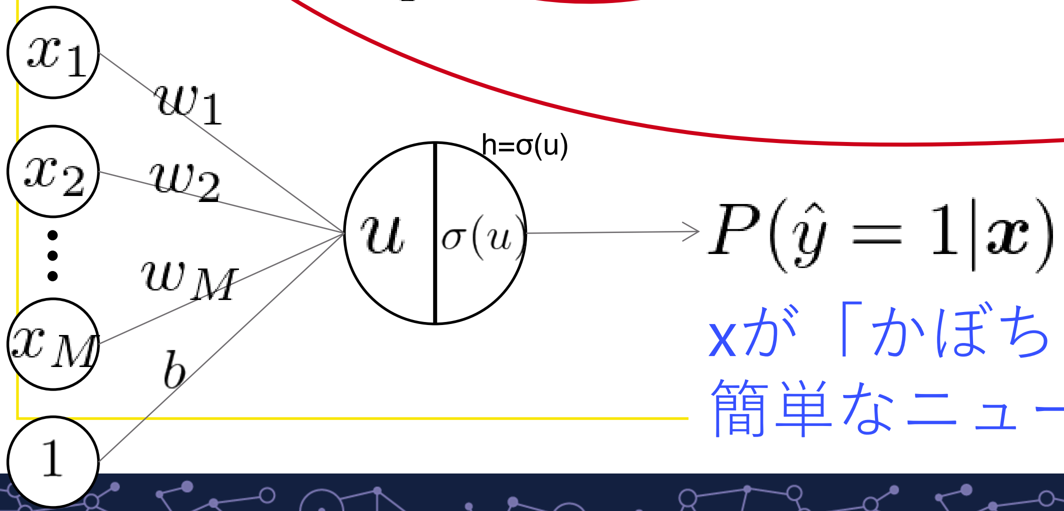
- pの例：画像xが「かぼちゃ」である確率の予測値
  $p = P(\hat{y} = 1|\boldsymbol{x})$
- オッズ $\frac{p}{1-p}$
- pのロジット $u = \log \frac{p}{1-p}$

対数オッズとも呼ばれる

ロジスティック回帰との関係

ロジスティックシグモイド関数によるuの変換を考える

$$\sigma(u) = \frac{1}{1+e^{-u}}$$
$$= \frac{1}{1+e^{\log\frac{1-p}{p}}}$$
$$= p$$



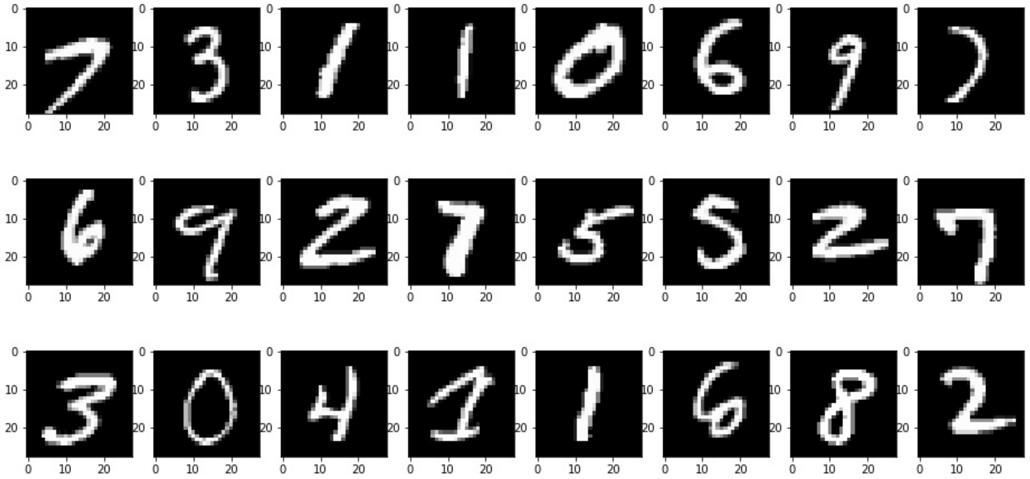
xが「かぼちゃ」である確率を予測する
簡単なニューラルネットと等価

用語

- pの例：画像xが「かぼちゃ」である確率の予測値
  $p = P(\hat{y}=1|\boldsymbol{x})$
- オッズ $\dfrac{p}{1-p}$
- pのロジット $u = \log\dfrac{p}{1-p}$

多クラス分類
例題：MNIST

- 手書き数字のデータセット
  - 深層学習分野でMNISTを知らない人はいないはず
- 28×28ピクセル画像
- 訓練集合：6万枚
  テスト集合：1万枚



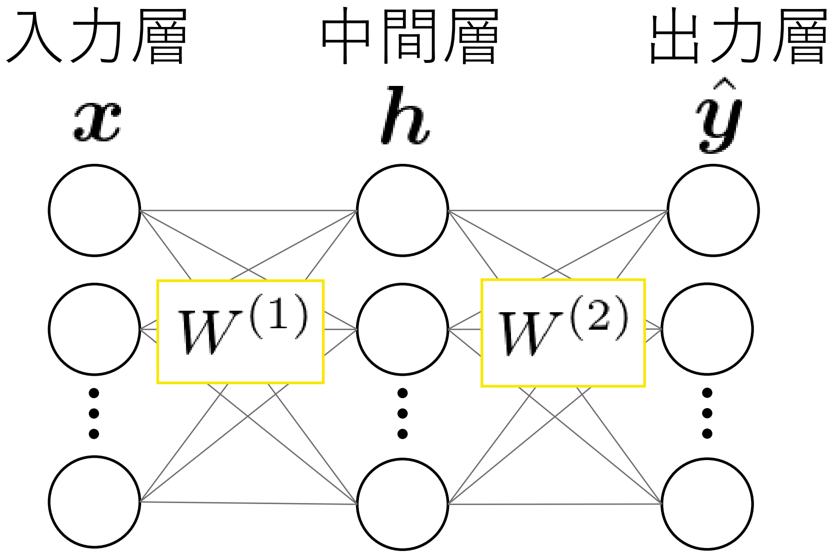
- 1-of-K表現
  - 特定の次元のみ１であり、残りの次元は０
  - テキスト処理において単語を表現する方法でもある

- Zero: (1, 0, 0, 0, 0, 0, 0, 0, 0, 0)
- One: (0, 1, 0, 0, 0, 0, 0, 0, 0, 0)
- Two: (0, 0, 1, 0, 0, 0, 0, 0, 0, 0)

3層ニューラルネットによる多クラス分類
回帰と分類の違い

■ 3層ニューラルネット（再）

$$\boldsymbol{h} = f^{(1)}(W^{(1)}\boldsymbol{x})$$
$$\hat{\boldsymbol{y}} = f^{(2)}(W^{(2)}\boldsymbol{h})$$



■ 分類

$$\boldsymbol{h} = f^{(1)}(W^{(1)}\boldsymbol{x})$$
$$P(\hat{y}_k|\boldsymbol{x}) = \mathrm{softmax}_k(W^{(2)}\boldsymbol{h})$$

■ 出力例

(0.8, 0.1, 0.1, 0, 0, 0, 0, 0, 0, 0)

３層ニューラルネットによる多クラス分類
ソフトマックス関数とは

■ **ソフトマックス関数**（softmax function）

$$\mathrm{softmax}_k(u_1, u_2, \ldots, u_K) = \frac{\exp(u_k)}{\sum_j \exp(u_j)}$$

■ $\boldsymbol{u} = (3, 1, 4)$ の例

$$\mathrm{softmax}_1(\boldsymbol{u}) = \frac{e^3}{e^3 + e^1 + e^4} \simeq 0.259$$

$$\mathrm{softmax}_2(\boldsymbol{u}) = \frac{e^1}{e^3 + e^1 + e^4} \simeq 0.035$$

$$\mathrm{softmax}_3(\boldsymbol{u}) = \frac{e^4}{e^3 + e^1 + e^4} \simeq 0.705$$

■ 分類

$$\boldsymbol{h} = f^{(1)}(W^{(1)}\boldsymbol{x})$$

$$P(\hat{y}_k|\boldsymbol{x}) = \mathrm{softmax}_k(W^{(2)}\boldsymbol{h})$$

どのクラスに分類されそうかの確率

指数関数で変換したのち、
規格化している

３層ニューラルネットによる多クラス分類 交差エントロピー誤差関数とは

- 情報理論における離散分布 p, q 間の交差エントロピー

$$H(p,q) = -\sum_i p_i \log q_i$$

正解の確率は大きく、yで0,1をちゃんと予測できてるとその正解の確率（一番大きい）の要素の依存を強く受け-logでEが小さくなる

（普通の）エントロピー

$$H(p) = -\sum_i p_i \log p_i$$

- **交差エントロピー誤差関数**（cross-entropy error function）

Eが小さいほど予測確率が正解に近く良いモデル

正解ラベル y_{nk} は固定値なので確率で表す必要がない

モデルが「サンプルnはクラスkである」と予測した確率

$$E(\boldsymbol{w}) = -\sum_{n=1}^{N}\sum_{k=1}^{K} y_{nk} \log P(\hat{y}_{nk})$$

p
p（本当の分布）と
q
q（モデルの予測分布）の間の差を測る

サンプル番号 n のラベルの k 次元目の値（クラスkであれば１であり、そうでなければ０）

３層ニューラルネットによる多クラス分類
２値分類の場合の交差エントロピー誤差関数

■ Kクラス

$$E(\boldsymbol{w}) = -\sum_{n=1}^{N}\sum_{k=1}^{K} y_{nk} \log P(\hat{y}_{nk})$$

■ ２クラス（K=2）

$$E(\boldsymbol{w}) = -\sum_{n=1}^{N}[y_{n1} \log P(\hat{y}_{n1}) + y_{n2} \log P(\hat{y}_{n2})]$$

$$= -\sum_{n=1}^{N}[\underline{y_n} \log P(\hat{y}_n) + (1 - y_n) \log(1 - P(\hat{y}_n))]$$

高校数学で言うと
余事象の考え方

サンプル番号nのラベル（１または０）

理解度確認

理解度確認
以下について周りと相談して１分以内に答えよ

1. 訓練集合とテスト集合の違いは何か？
2. 訓練集合と訓練サンプルの違いは何か？
3. ミニバッチ確率的勾配降下法の英語名は何か？
4. 損失関数の例を挙げよ。

※LLMに聞いても良いが、ハルシネーションの場合に「LLMが誤った（のであって自分は悪くない）」という回答は不適切　＝検証が必要

交差エントロピー誤差関数と最尤推定

ベルヌーイ分布（Bernoulli distribution）

ひしゃげたコインの分布

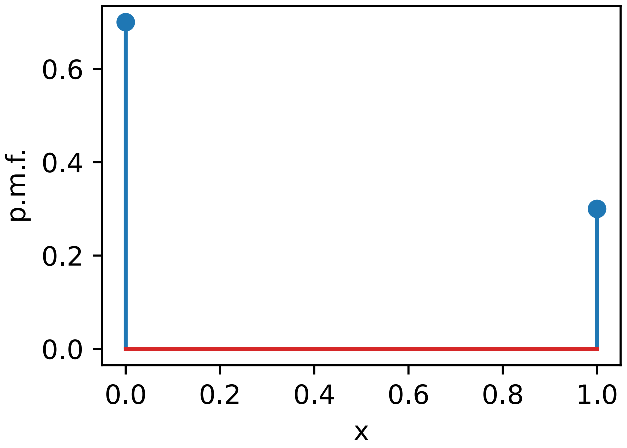
- $P(x) = \mathrm{Bern}(\mu) = \underline{\mu^x (1-\mu)^{1-x}}$
  - 2値をとる実現値 $x \in \{0, 1\}$を生成するための確率分布
  - 1個のパラメータ（母数）$0 \le \mu \le 1$によって分布の性質が決まる

  例：$\mu = 0.3$ のとき

  $\{x_1, x_2, ..., x_{20}\} = \{1,0,0,0,1,0,1,0,0,1,0,0,0,0,1,0,0,0,1,0\}$

  - 期待値：$1 \times \mu + 0 \times (1-\mu) = \mu$
  - 分散：$E[x^2] - (E[x])^2 = E[x] - (E[x])^2 = \mu - \mu^2 = \mu(1-\mu)$
- 同時確率

$$P(x_1, x_2, ..., x_N; \mu) = \prod_{n=1}^{N} \mu^{x_n} (1-\mu)^{1-x_n}$$



べき乗で場合分けを表現するトリック
xが1の確率μ
xが0の確率$(1-\mu)$

最尤推定

- 観測値 $x_1, x_2, \ldots, x_N$ の**同時確率を最大化**したい

- サンプルは母集団から**独立同分布**で抽出されたものとする（**i.i.d.**; independent and identically distributed）

$$P(x_1, x_2, \ldots, x_N; \mu) = \prod_{n=1}^{N} P(x_n; \mu)$$

尤度とは

- 観測値 $x_1, x_2, \ldots, x_N$ の**同時確率を最大化**したい

- サンプルは母集団から**独立同分布**で抽出されたものとする（**i.i.d.**; independent and identically distributed）

$$P(x_1, x_2, \ldots, x_N; \mu) = \prod_{n=1}^{N} P(x_n; \mu)$$

- 「μ が既知で、$\{x_n\}$ が未知」から「$\{x_n\}$が既知で、μ が未知」に見方を変える

- **尤度**（**likelihood, ゆうど**）
  - **データが与えられたうえでのモデルの尤もらしさ**
  - 規格化（＝足して１）されていないので**確率ではない**

$$L(\mu) = P(x_1, x_2, \ldots, x_N; \mu) \to \max$$

交差エントロピー誤差の最小化は尤度最大化を意味する

- 2値分類の場合の尤度関数

$$L(\boldsymbol{w}) = \prod_{n=1}^{N} \{P(\hat{y}_n)\}^{y_n}\{1 - P(\hat{y}_n)\}^{1-y_n}$$

- 尤度最大化
  ＝対数尤度最大化
  ＝負の対数尤度最小化
  →損失関数として最小化

- 「μ が既知で、$\{x_n\}$ が未知」から「$\{x_n\}$が既知で、μ が未知」に見方を変える

- **尤度**（**likelihood, ゆうど**）：
  - **データが与えられたうえでのモデルの尤もらしさ**
  - 規格化（＝足して１）されていないので**確率ではない**

$$L(\mu) = P(x_1, x_2, \ldots, x_N; \mu) \rightarrow \max$$

交差エントロピー誤差の最小化は尤度最大化を意味する

- ２値分類の場合の尤度関数

$$L(\boldsymbol{w}) = \prod_{n=1}^{N} \{P(\hat{y}_n)\}^{y_n} \{1 - P(\hat{y}_n)\}^{1-y_n}$$

確率のように小さい数を何度も
掛け合わせるより、対数をとって
足し算にしたほうが楽

- 尤度最大化
＝対数尤度最大化
＝負の対数尤度最小化
→損失関数として最小化

$$E(\boldsymbol{w}) = -\sum_{n=1}^{N} [y_n \log P(\hat{y}_n) + (1 - y_n) \log(1 - P(\hat{y}_n))]$$

↑交差エントロピー誤差

本講義全体の参考図書

- ★機械学習スタートアップシリーズ これならわかる深層学習入門　瀧雅人著　講談社
- ★Dive into Deep Learning (https://d2l.ai/)
- 深層学習 改訂第2版 (機械学習プロフェッショナルシリーズ)　岡谷貴之著　講談社
- ディープラーニングを支える技術　岡野原大輔著　 技術評論社
- 画像認識 (機械学習プロフェッショナルシリーズ)　原田達也著　講談社
- 深層学習による自然言語処理 (機械学習プロフェッショナルシリーズ)　坪井祐太、海野裕也、鈴木潤 著、講談社
- IT Text 自然言語処理の基礎　岡﨑直観、荒瀬由紀、鈴木潤、鶴岡慶雅、宮尾祐介 著、オーム社
- 東京大学工学教程 情報工学 機械学習　中川 裕志著、東京大学工学教程編纂委員会編　丸善出版
- パターン認識と機械学習 上・下　C.M. ビショップ著　丸善出版
- Bishop, Christopher M. and Bishop, Hugh, "Deep Learning: Foundations and Concepts", Springer, ISBN-13:978-3031454677

小レポート①の準備

小レポート①の準備

- Kaggleとは
  - 機械学習コンペを開催するプラットフォーム

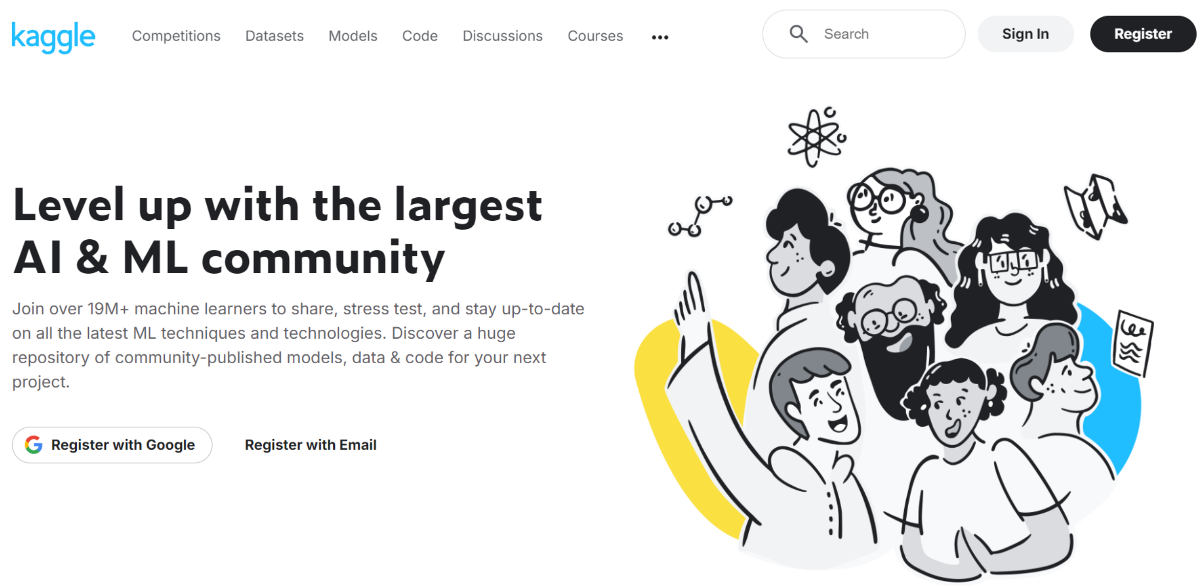
- 次回講義までの宿題
  - 「Kaggleへのユーザ登録」を行ってください

- 次回講義で行うこと
  - 小レポート①の内容・締切の説明

ユーザ登録（1/4）

① 以下へアクセス

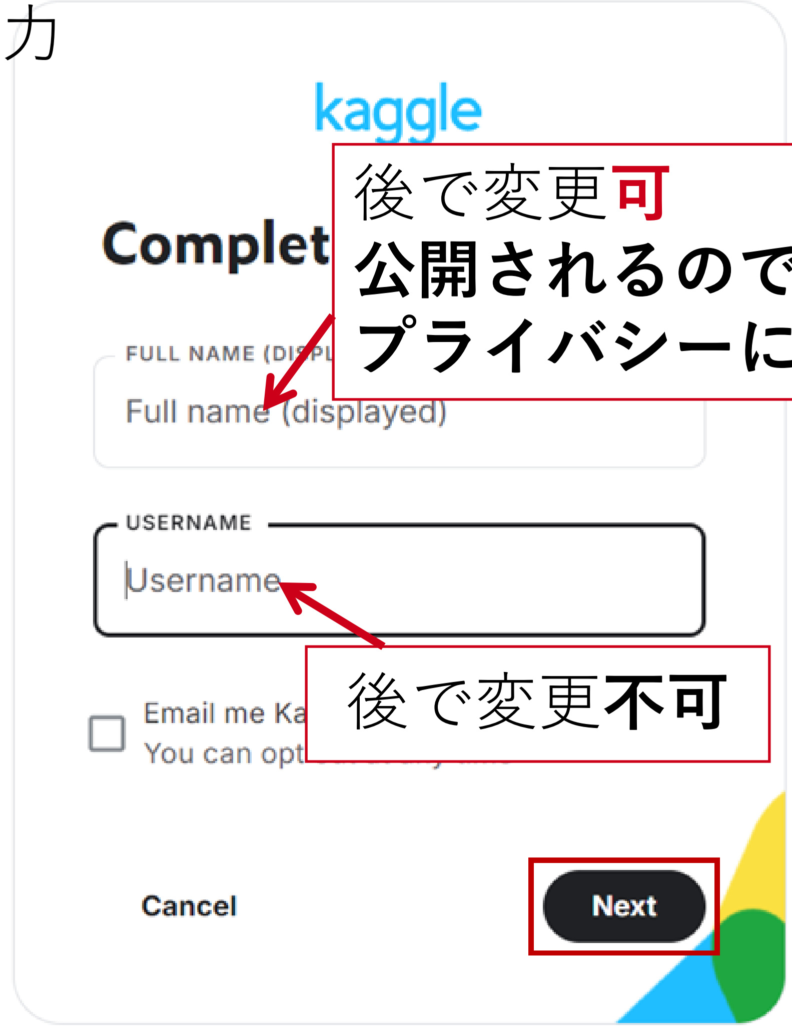
https://www.kaggle.com/



② Googleアカウント等でログイン

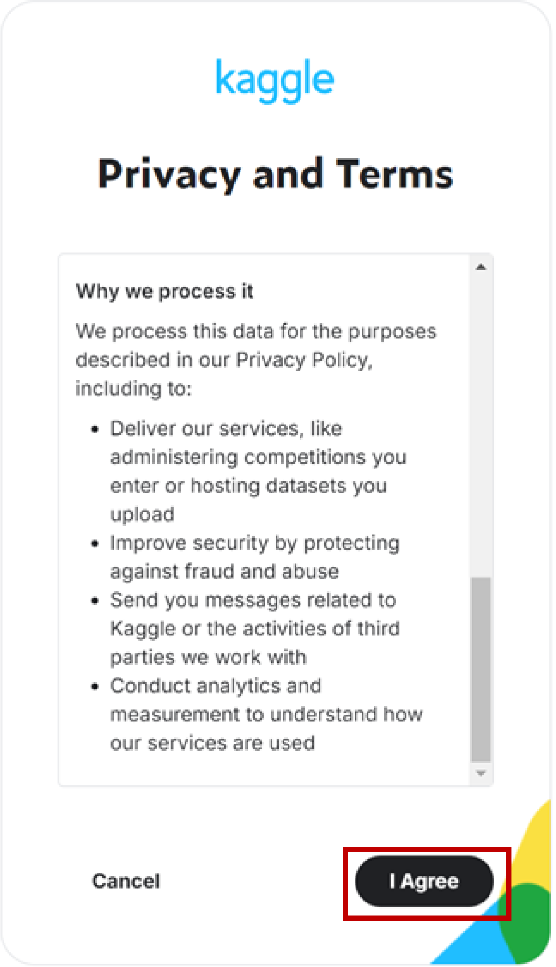
ユーザ登録（2/4）

③ ユーザ名・表示名を入力



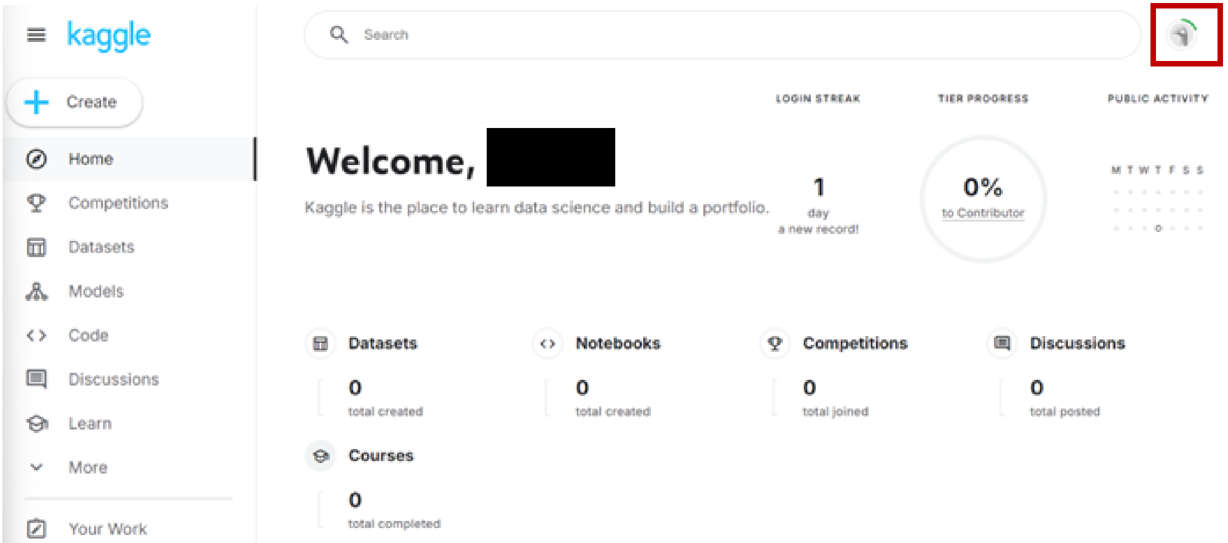
後で変更**可**
公開されるので、
プライバシーに配慮

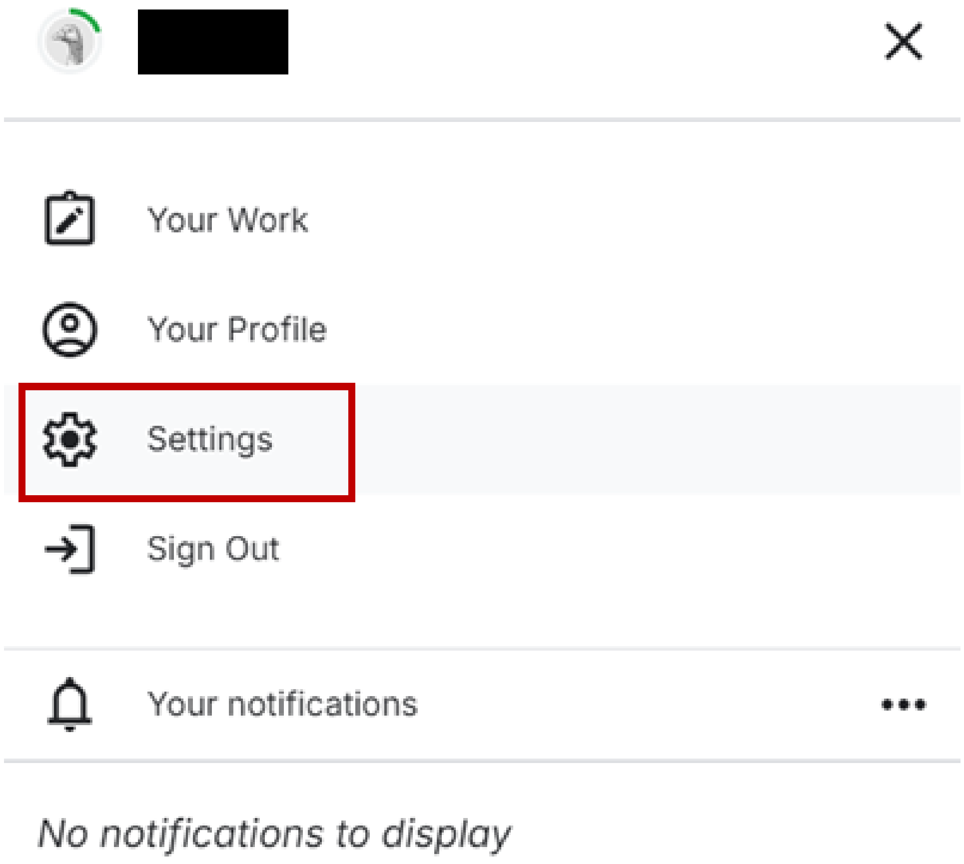
後で変更**不可**



ユーザ登録（3/4）

④アカウントの「Settings」を開く





ユーザ登録（4/4）

⑤コンペ参加するために電話番号で認証（課題提出に必要）

Phone verification

Your account is not verified. Verifying your account with a phone number allows you to do more on Kaggle, and helps prevent spam and other abuse.

Phone verify

⑥以下を入力すると、認証コードがSMSで送られてくる

Just one thing—first verify your account

Enter your phone number and we'll send you a code

COUNTRY

JP +81 JP

PHONE NUMBER

[redacted]

電話番号

I'm not a robot

reCAPTCHA

Privacy - Terms

Already have a code? Enter it now.

Cancel

Send verification code

実習

実習
MNISTへの３層ニューラルネットの適用

- ニューラルネットの出力：10次元
  - 10次元の出力のうち、最大のものを予測ラベルとする
- 損失関数：交差エントロピー誤差関数
- ミニバッチSGDで反復的に損失を最小化

- 理工学基礎実験との違いは、コーディングが多いJ科向けか否か
  - 理工学基礎実験：　視覚的にわかりやすいが自由度は低いコード
  - 機械学習基礎：　各自が改変しやすいように不要な関数を削除

実習

実習の目的

■ コーディングと基礎理論の関係を学ぶ

実習課題の場所

■ K-LMSから辿る

実習に関する質問

■ LLMに説明させる

■ 教科書で調べる・検索・周囲と相談（私語禁止ではありません）

■ 上記で解消しなければ挙手