機械学習入門

経済学部 BX584

第14回　多層パーセプトロン

おさらい

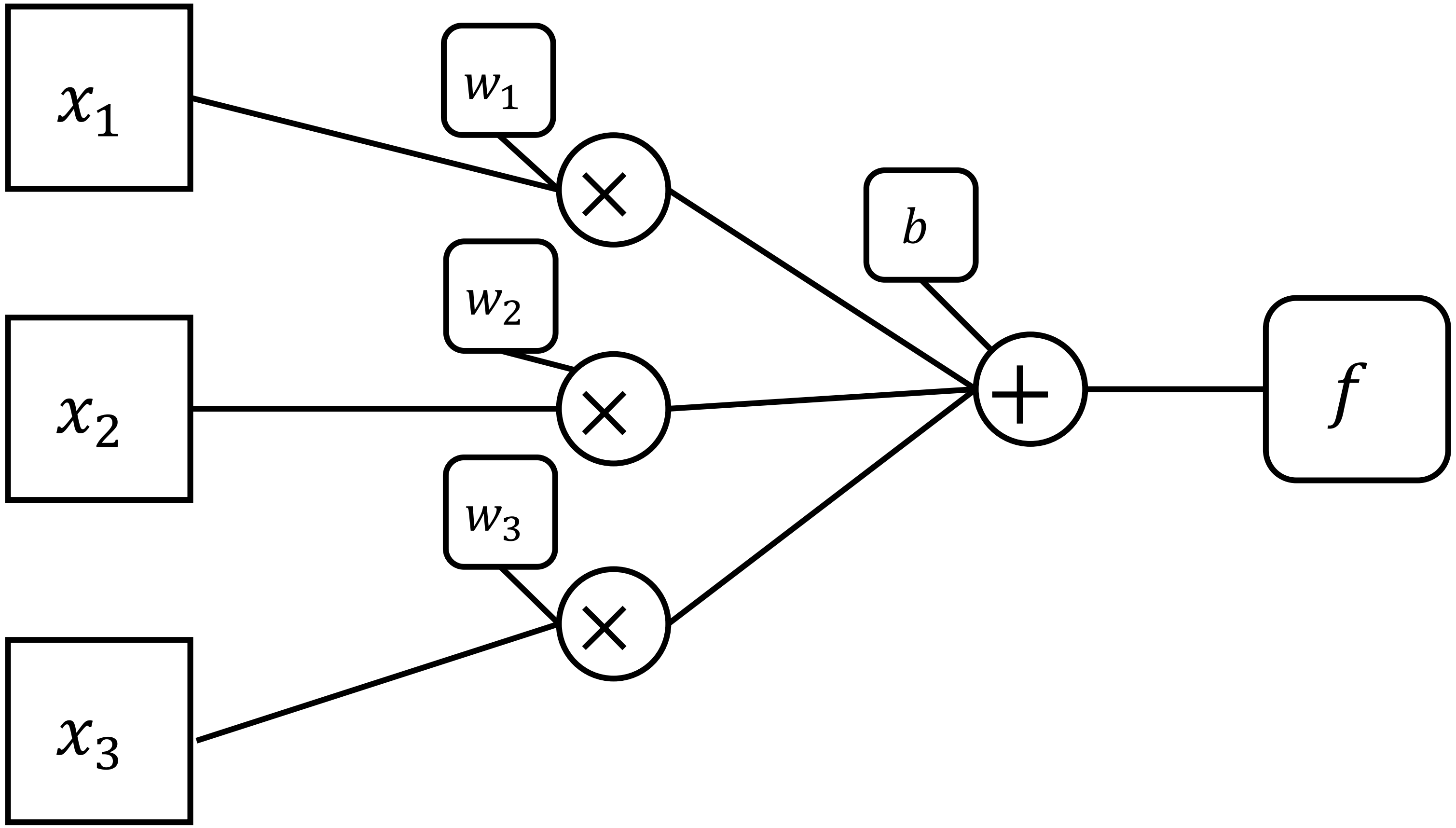
- 教師あり機械学習
  - 理想の出力値を与える「関数」を探す

    ＝関数のパラメータ設定のうち、できるだけ良いものを探す
- 損失関数の最小化として問題を定式化
  - 特定の入力値に対する、理想的な出力値と、実際の出力値との、ズレ
  - ズレを小さくする＝出力値を理想的な値に近づける
    - ズレが小さくなる方向にパラメータをうごかしていく＝より良いパラメータ設定を探す
- 最急降下法（確率的勾配降下法、ミニバッチ学習）
  - 勾配を使ってパラメータ更新

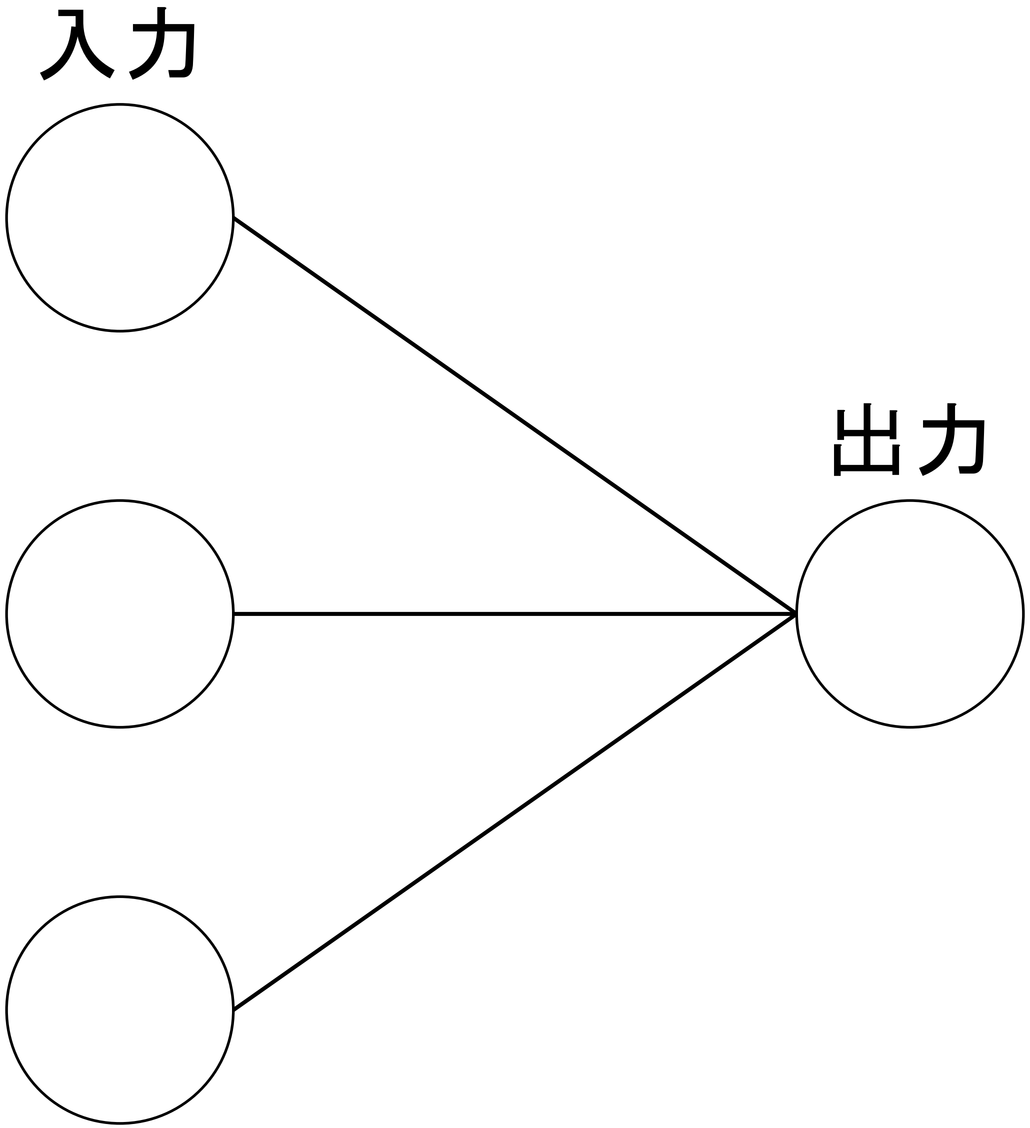
線形回帰で使った予測モデル

例）明日の終値を、今日含め過去3日分の終値を使って予測

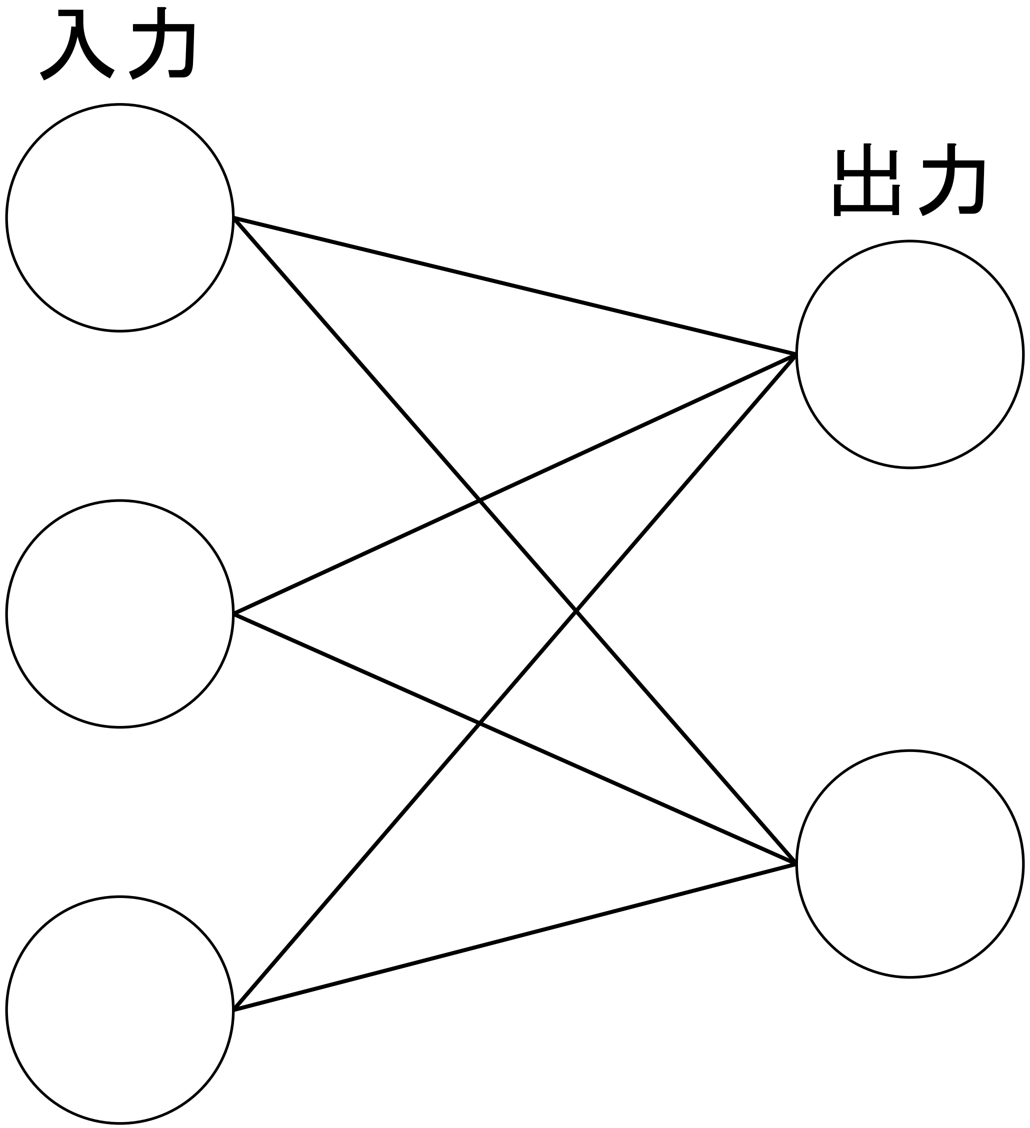
$$f = b + w_1 x_1 + w_2 x_2 + w_3 x_3$$



いろいろ略してこう書いてしまうことが多い



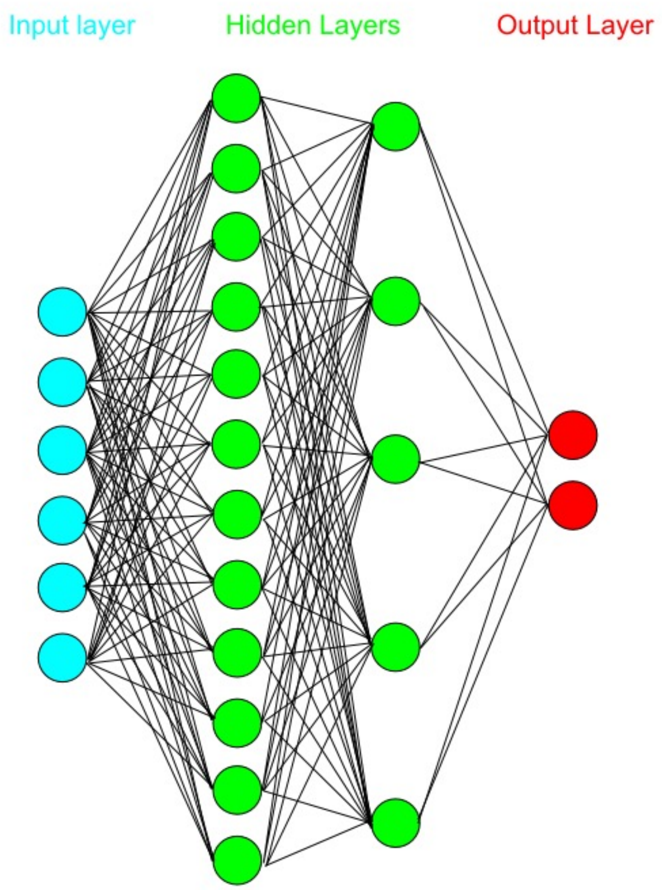
予測したい値が複数あるときは…こうすればいいだけ



ニューラルネットワーク

- 入力値に対して理想的な出力値を得るための関数
  - 人間の脳の構造を模倣して云々の話はどうでもよい。
  - 機械学習の世界では関数として使うだけなので。
- 隠れ層のノード数や隠れ層の数を増やすことで構造を複雑にできる
  - 表現力を高くすることができる。
  - ただしこの構造（アーキテクチャ）を決めるのは人間の仕事。
    - 検証データで評価しつつ決めていく。非常に時間がかかる・・・。

多層パーセプトロン(MLP)

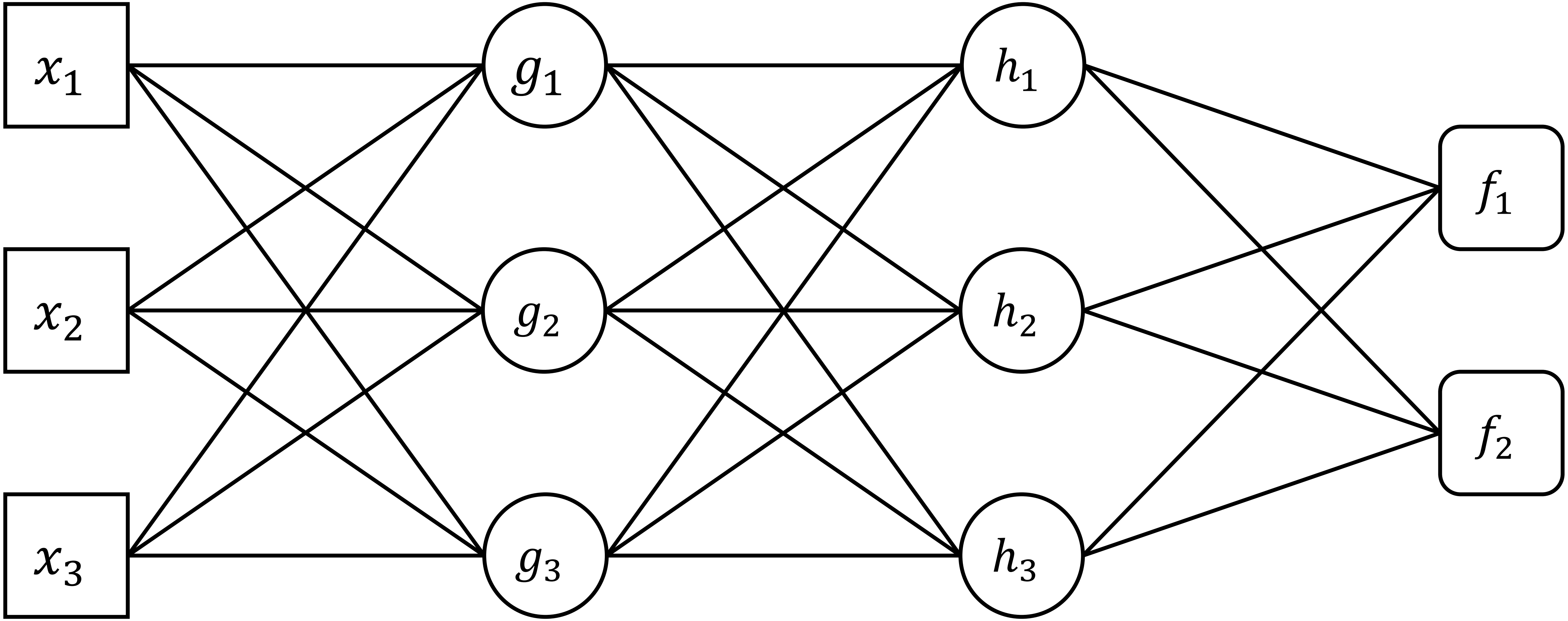


https://cs.stackexchange.com/questions/28597/difference-between-multilayer-perceptron-and-linear-regression

例）入力が3変数、出力が2変数、隠れ層は2層

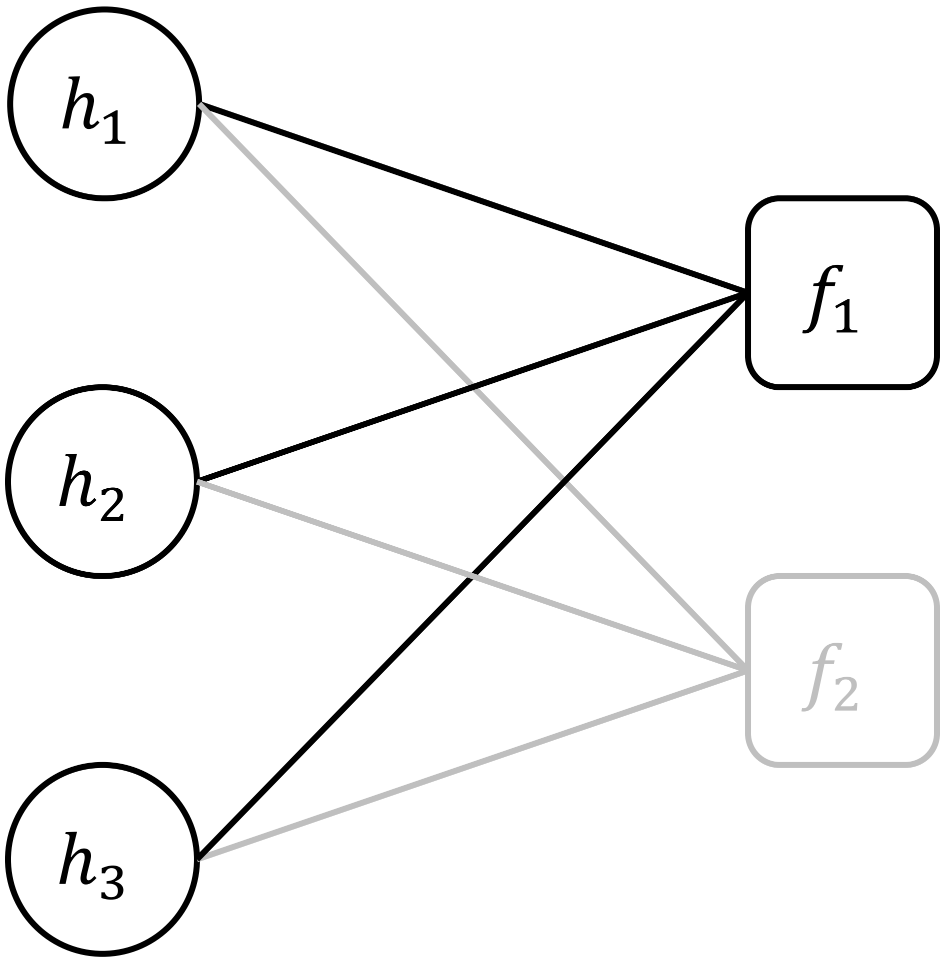
- 例えば、年齢、体重、BMI値からコレステロール値と血糖値を予測
  - 適当に考えた問題設定です･･･。
- 損失関数は2乗誤差とする
  - （損失関数）

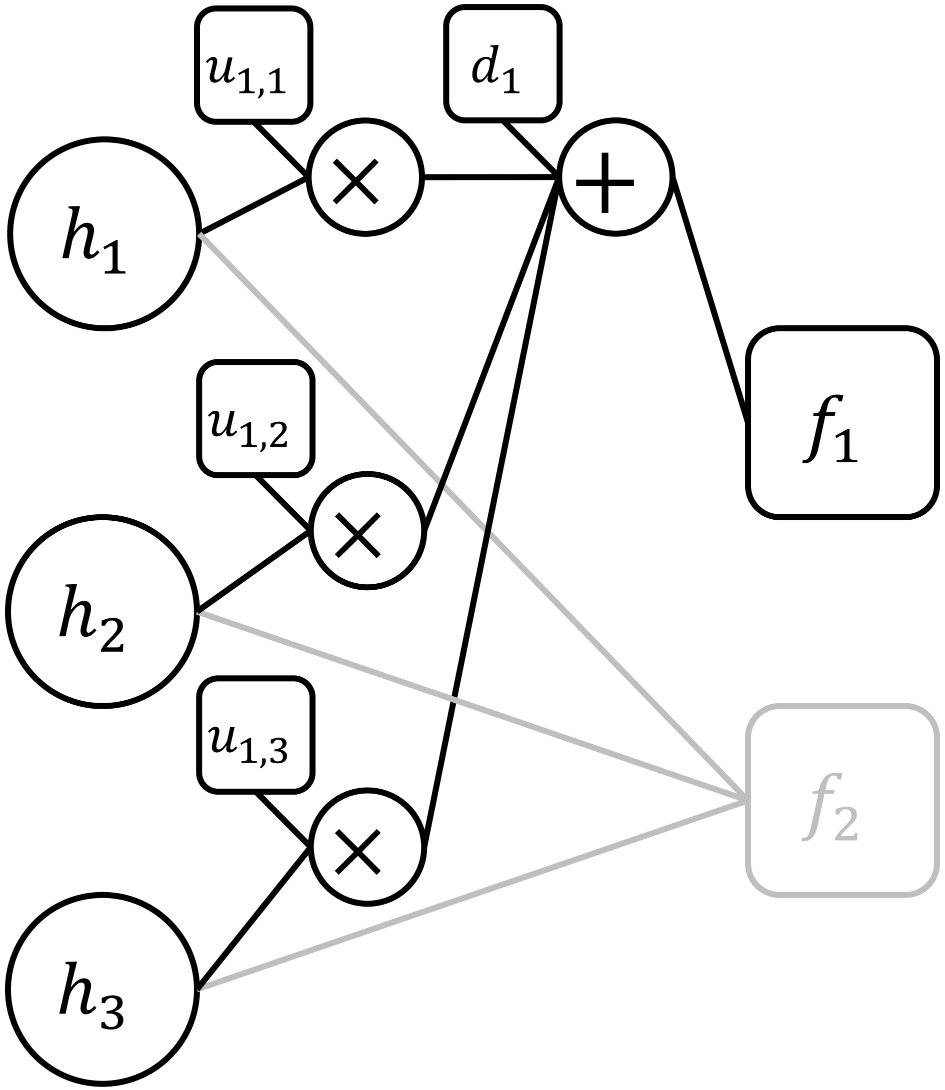
$$
\begin{aligned}
&= (\text{実際のコレステロール値} - \text{MLPが出力したコレステロール値})^2 \\
&\quad + (\text{実際の血糖値} - \text{MLPが出力した血糖値})^2
\end{aligned}
$$



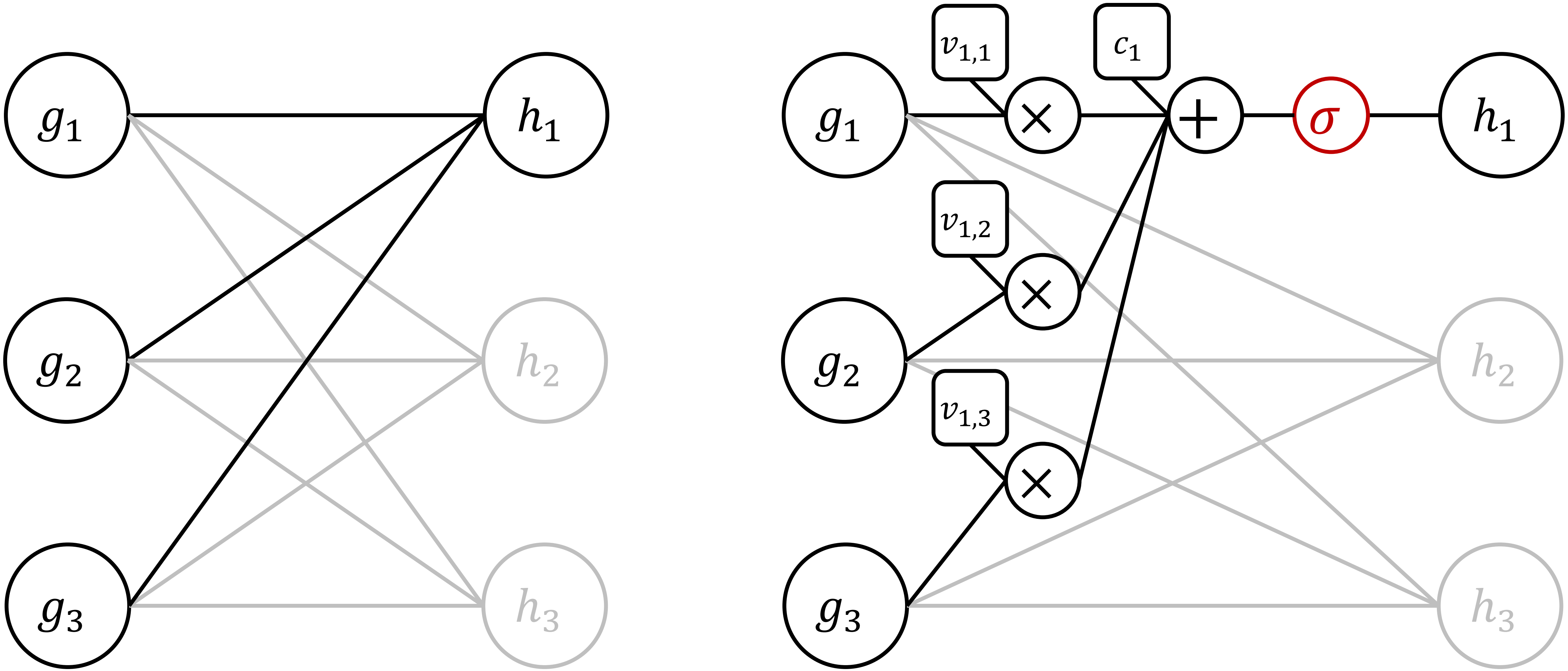
こういう図、よく目にするけど、どう読むの？

よくあるMLPのこの図は右図の計算グラフの略記！





よくあるMLPのこの図は右図の計算グラフの略記！
（隠れ層の活性化関数も略されている）

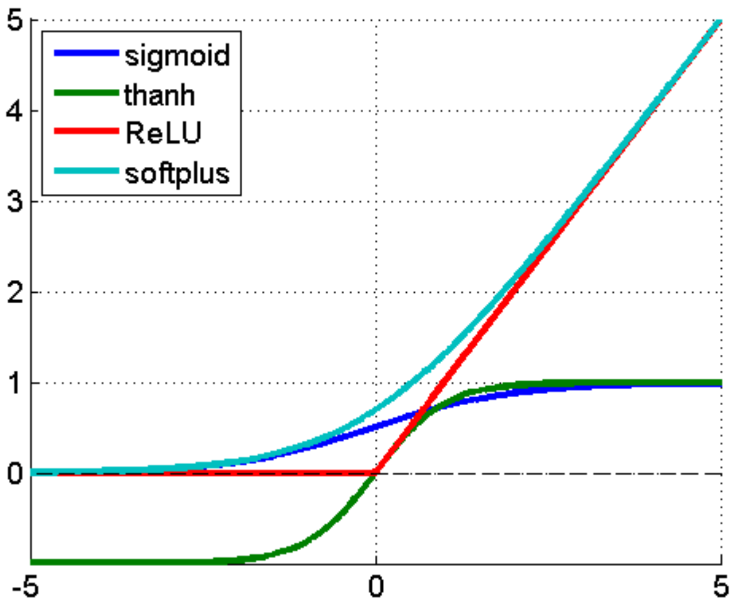


隠れ層の出力に非線形関数（活性化関数）を適用

- なぜ？
  - そうしないと、単層のパーセプトロンとして書けてしまう
    - 行列の積は行列になるため。
  - つまり、多層にしている意味があまりなくなる
    - ただし、非線形関数を使わなくても、中間層の次元が小さい場合は次元圧縮になる。
    - これはこれで意味はあるが、全体が（行列）×（入力ベクトル）という一回の掛け算で書けてしまうことに変わりはない。

活性化関数(activation function)

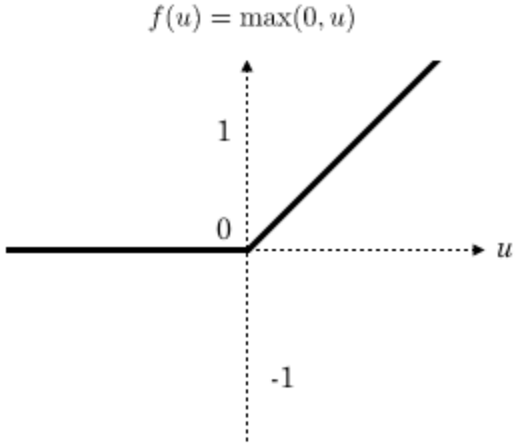
- 隠れ層のactivation functionにはReLUを使うのが最近は一般的
  - シグモイド関数はほぼ使わない
  - ReLUの亜種はよく使う
    - 特にLeakyReLUは使う。



ReLU (rectified linear unit)

$$y = \begin{cases} x & x \geq 0 \\ 0 & x < 0 \end{cases}$$

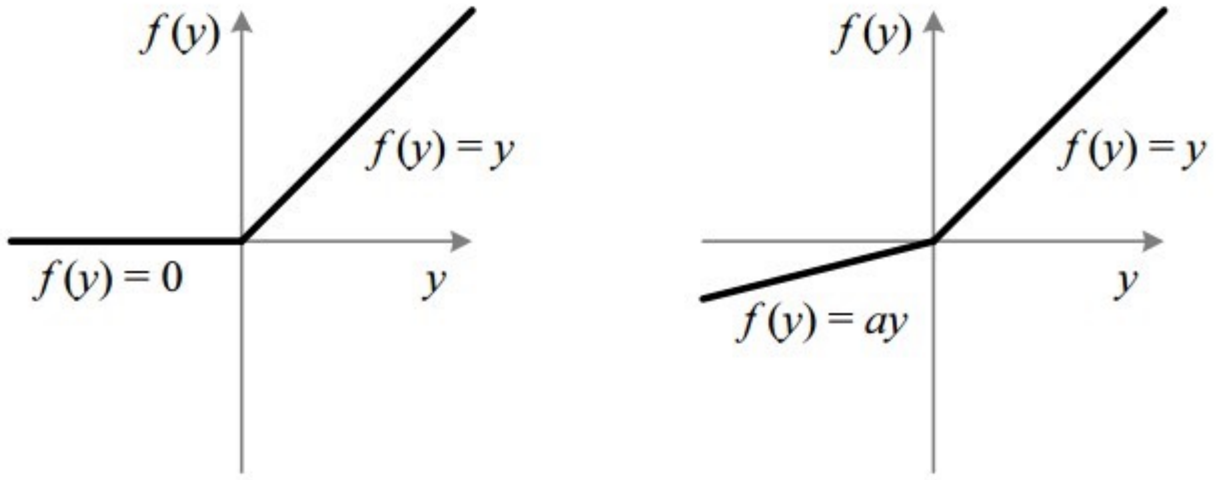
$$\frac{dy}{dx} = \begin{cases} 1 & x \geq 0 \\ 0 & x < 0 \end{cases}$$



- 微分が1か0かのどちらか
  - 伝播された誤差をそのまま伝えるか、伝えないかのどちらか

LeakyReLU

- 入力が負の場合も勾配がゼロにならないようにしてある

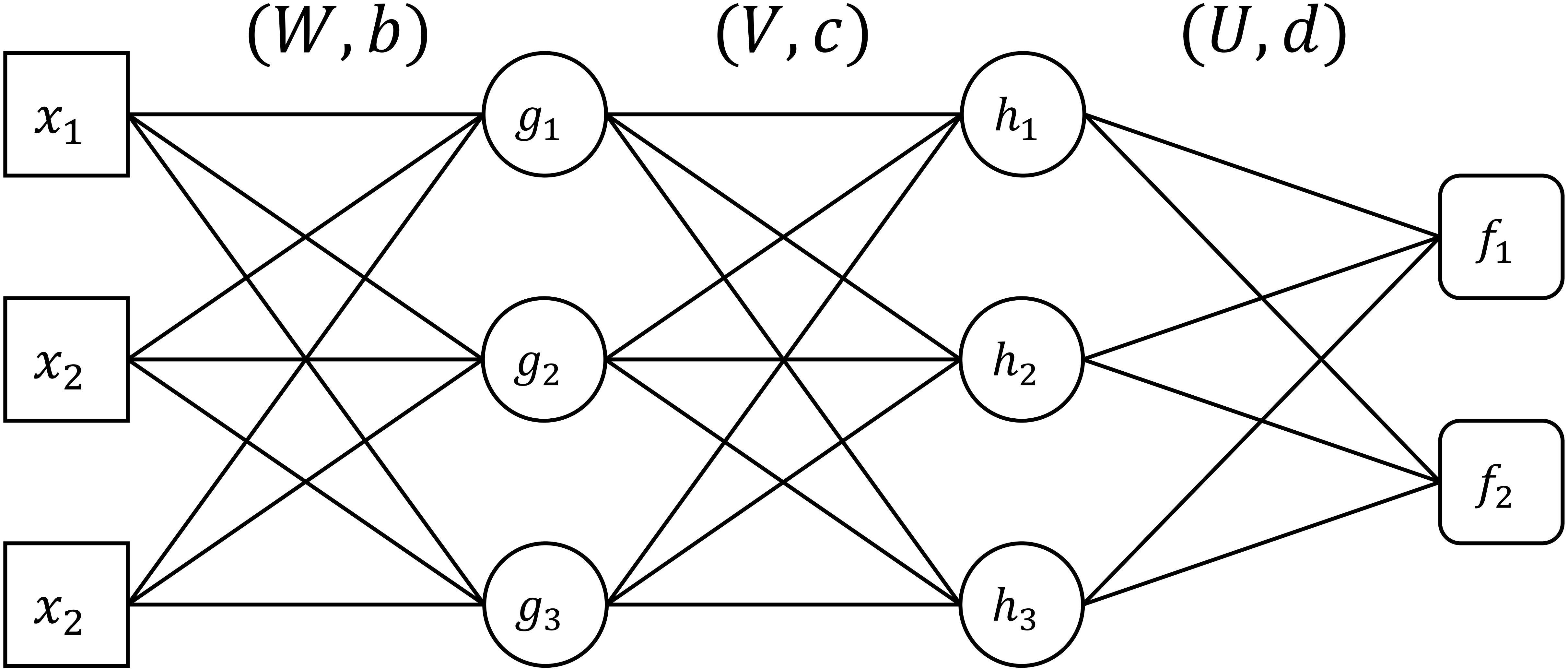


隠れ層の活性化関数になぜシグモイド関数を使わないか？

$$y = \frac{1}{1 + e^{-x}}$$

$$\frac{dy}{dx} = -\frac{-e^{-x}}{(1 + e^{-x})^2} = \frac{(1 + e^{-x}) - 1}{(1 + e^{-x})^2} = y - y^2 = y(1 - y)$$

- この微分が誤差逆伝播に良くない（勾配消失問題）
  - $1 \geq y \geq 0$だから$\frac{1}{4} \geq \frac{dy}{dx} \geq 0$
  - 微分の値が1よりかなり小さい（最大でも0.25）＝パラメータが少ししか更新されない
- ON/OFF（1か0か）の"近似"としての使いみちのほうが大きいかも
  - cf. ロジスティック回帰
  - cf. LSTM



この手の図については、辺の重みも、書かれていないバイアスも、
省略されている活性化関数も、想像しつつ眺められるようにしておく。

前向き計算

- 入力から最初の隠れ層へ

$$\boldsymbol{g} = \mathrm{ReLU}(\boldsymbol{W}\boldsymbol{x} + \boldsymbol{b})$$

- 最初の隠れ層から次の隠れ層へ

$$\boldsymbol{h} = \mathrm{ReLU}(\boldsymbol{V}\boldsymbol{g} + \boldsymbol{c})$$

- 最後の隠れ層から出力へ

$$\boldsymbol{f} = \boldsymbol{U}\boldsymbol{h} + \boldsymbol{d}$$

損失関数（回帰の場合）

- 2乗誤差
  - MLPの出力値を(f_1, f_2)目標となる出力値を(y_1, y_2)とすると

$$L = (y_1 - f_1)^2 + (y_2 - f_2)^2$$

  - 出力が多数（K個）あっても同様

$$L = \sum_{k=1}^{K} (y_k - f_k)^2$$

損失関数（多値分類の場合）

- クロスエントロピー
  - まずMLPの出力値をソフトマックス関数に通す

$$s_k = \mathrm{SoftMax}(f_k) = \frac{\exp(f_k)}{\sum_{k'} \exp(f_{k'})}$$

  - クロスエントロピーは下記のとおり（正解ラベルの項しか残らない）

$$-\sum_{k=1}^{K} t_k \log s_k$$

BP計算例: ReLUを使ったMLPで2乗誤差の場合

- 2乗誤差の場合、目標となる出力値を(y_1, y_2)とすると

$$L = (y_1 - f_1)^2 + (y_2 - f_2)^2$$

- MLPの出力値で偏微分しておく（BPの最初の段階）

$$\frac{\partial L}{\partial f_1} = 2(f_1 - y_1)$$

$$\frac{\partial L}{\partial f_2} = 2(f_2 - y_2)$$

２番目の隠れ層～出力層

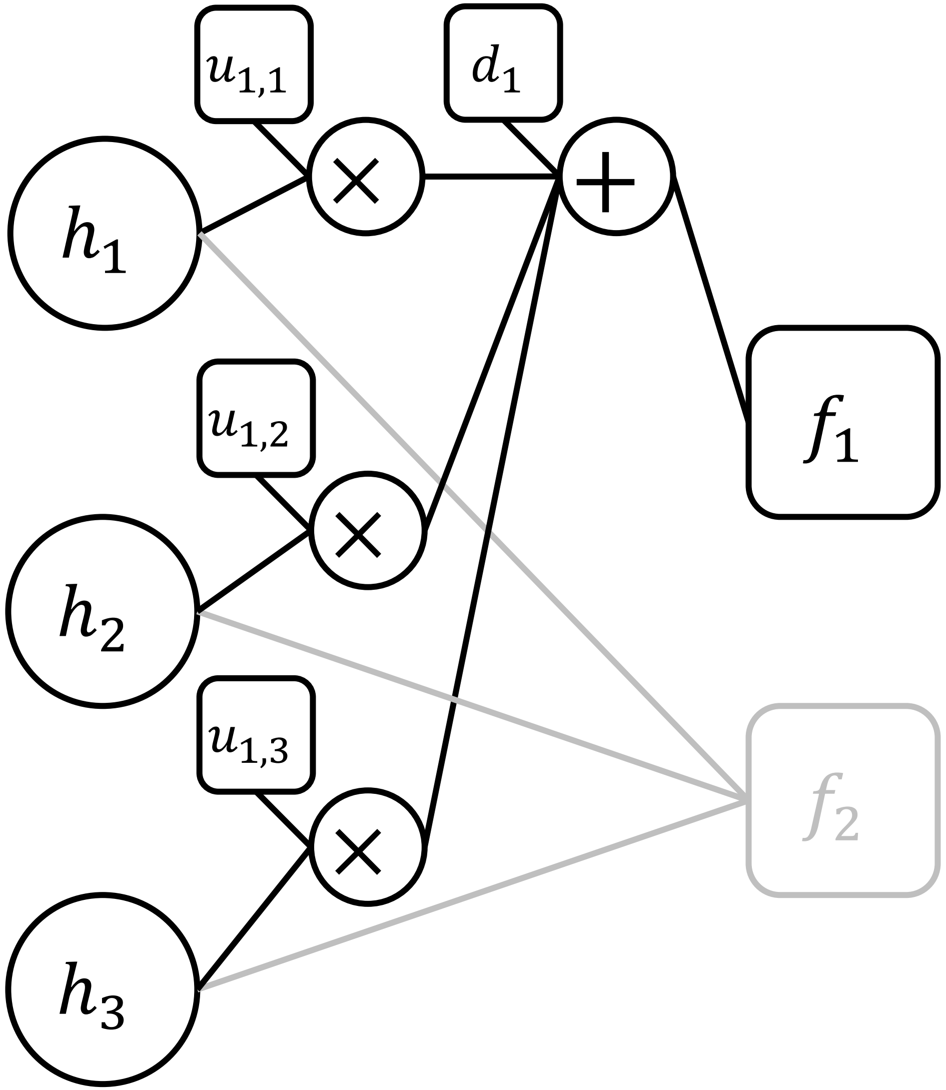
$$f_1 = u_{1,1}h_1 + u_{1,2}h_2 + u_{1,3}h_3 + d_1$$

$$\frac{\partial L}{\partial u_{1,1}} = \frac{\partial L}{\partial f_1}\frac{\partial f_1}{\partial u_{1,1}} = 2(f_1 - y_1)h_1$$

$$\frac{\partial L}{\partial u_{1,2}} = \frac{\partial L}{\partial f_1}\frac{\partial f_1}{\partial u_{1,2}} = 2(f_1 - y_1)h_2$$

$$\frac{\partial L}{\partial u_{1,3}} = \frac{\partial L}{\partial f_1}\frac{\partial f_1}{\partial u_{1,3}} = 2(f_1 - y_1)h_3$$

$$\frac{\partial L}{\partial d_1} = \frac{\partial L}{\partial f_1}\frac{\partial f_1}{\partial d_1} = 2(f_1 - y_1)$$



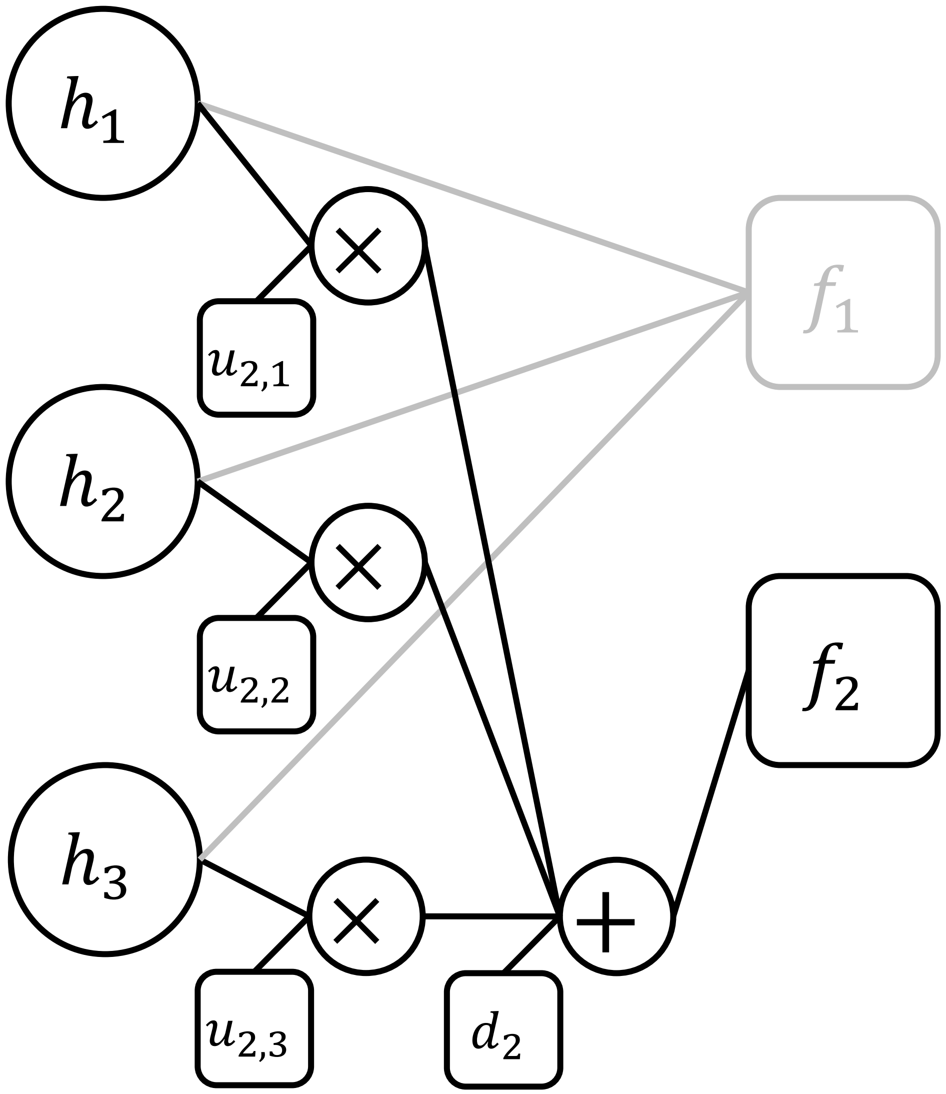
$$f_2 = u_{2,1}h_1 + u_{2,2}h_2 + u_{2,3}h_3 + d_2$$

$$\frac{\partial L}{\partial u_{2,1}} = \frac{\partial L}{\partial f_2}\frac{\partial f_2}{\partial u_{2,1}} = 2(f_2 - y_2)h_1$$

$$\frac{\partial L}{\partial u_{2,2}} = \frac{\partial L}{\partial f_2}\frac{\partial f_2}{\partial u_{2,2}} = 2(f_2 - y_2)h_2$$

$$\frac{\partial L}{\partial u_{2,3}} = \frac{\partial L}{\partial f_2}\frac{\partial f_2}{\partial u_{2,3}} = 2(f_2 - y_2)h_3$$

$$\frac{\partial L}{\partial d_2} = \frac{\partial L}{\partial f_2}\frac{\partial f_2}{\partial d_2} = 2(f_2 - y_2)$$

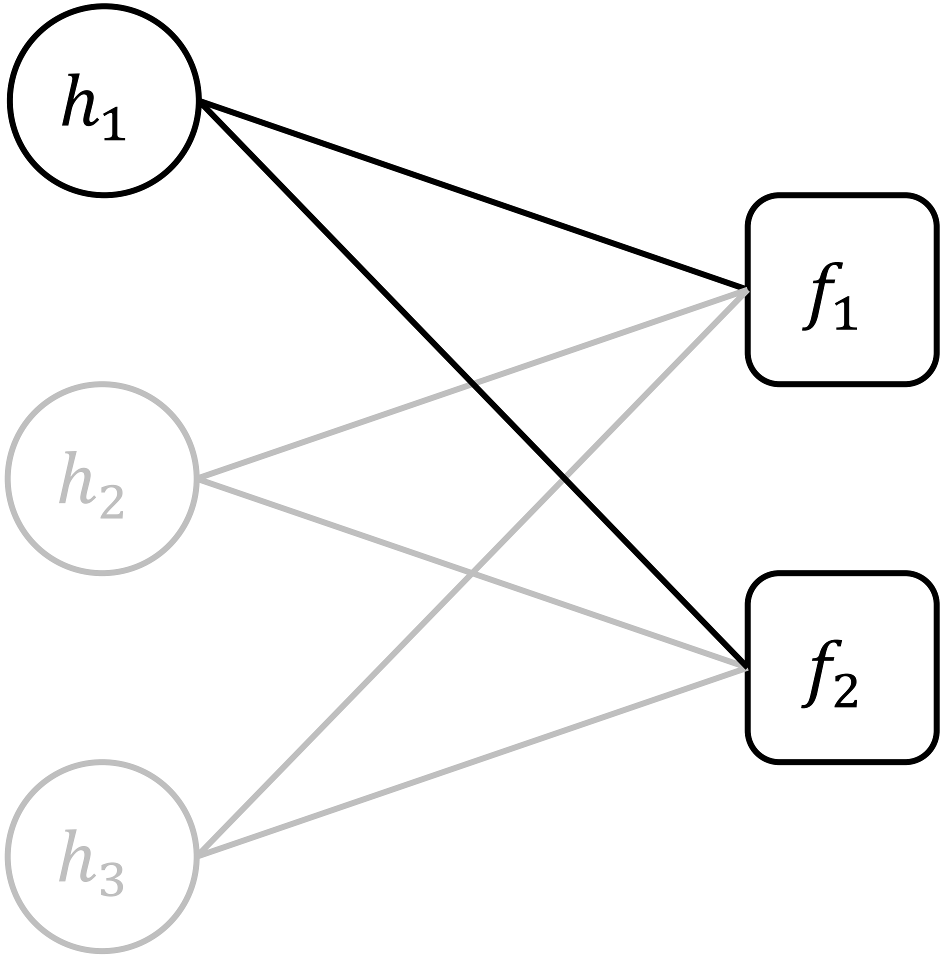


誤差の合流

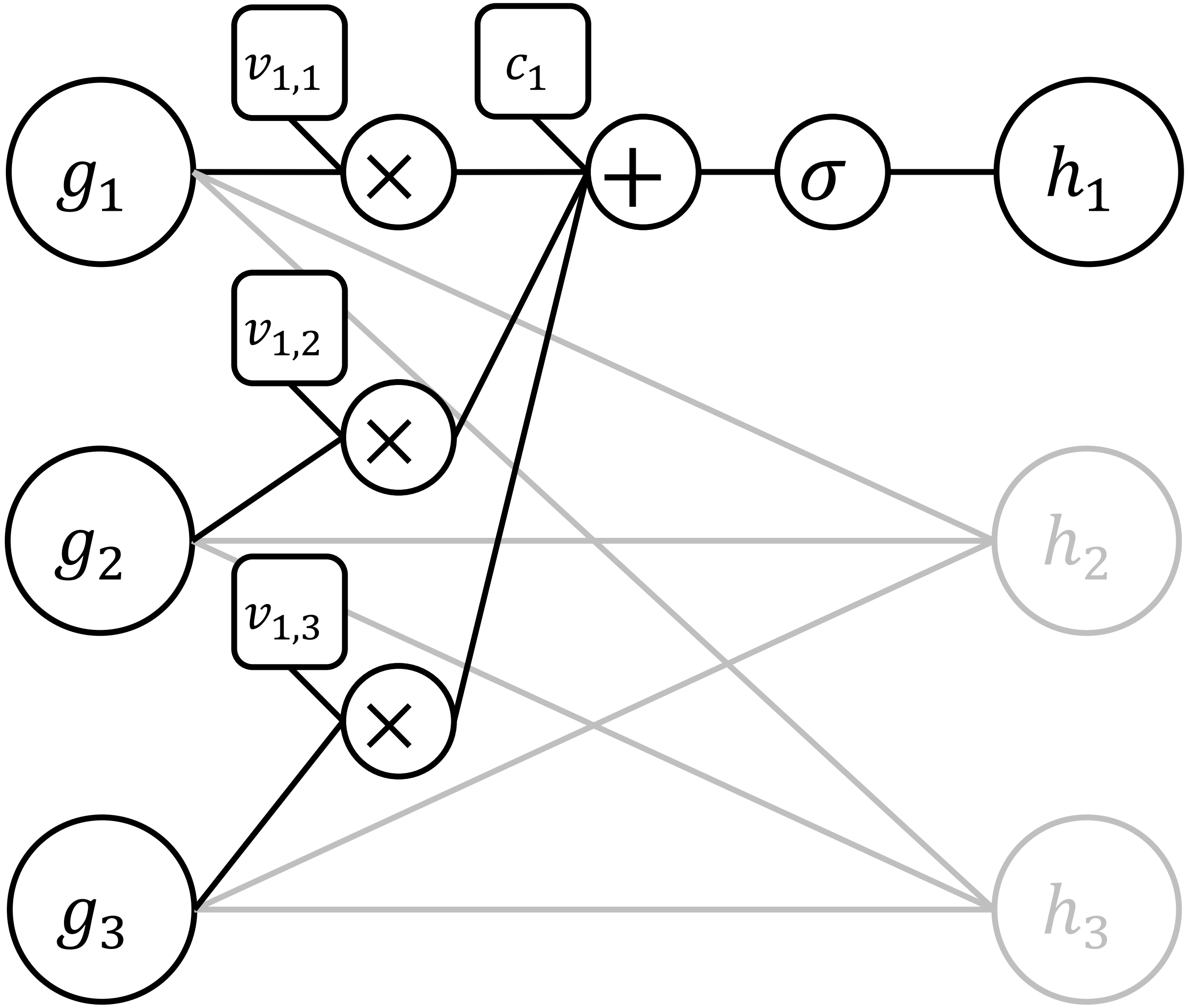
$$\frac{\partial L}{\partial h_1} = \frac{\partial L}{\partial f_1}\frac{\partial f_1}{\partial h_1} + \frac{\partial L}{\partial f_2}\frac{\partial f_2}{\partial h_1}$$
$$= 2(f_1 - y_1)u_{1,1} + 2(f_2 - y_2)u_{2,1}$$

$$\frac{\partial L}{\partial h_2} = \frac{\partial L}{\partial f_1}\frac{\partial f_1}{\partial h_2} + \frac{\partial L}{\partial f_2}\frac{\partial f_2}{\partial h_2}$$
$$= 2(f_1 - y_1)u_{1,2} + 2(f_2 - y_2)u_{2,2}$$

$$\frac{\partial L}{\partial h_3} = \frac{\partial L}{\partial f_1}\frac{\partial f_1}{\partial h_3} + \frac{\partial L}{\partial f_2}\frac{\partial f_2}{\partial h_3}$$
$$= 2(f_1 - y_1)u_{1,3} + 2(f_2 - y_2)u_{2,3}$$



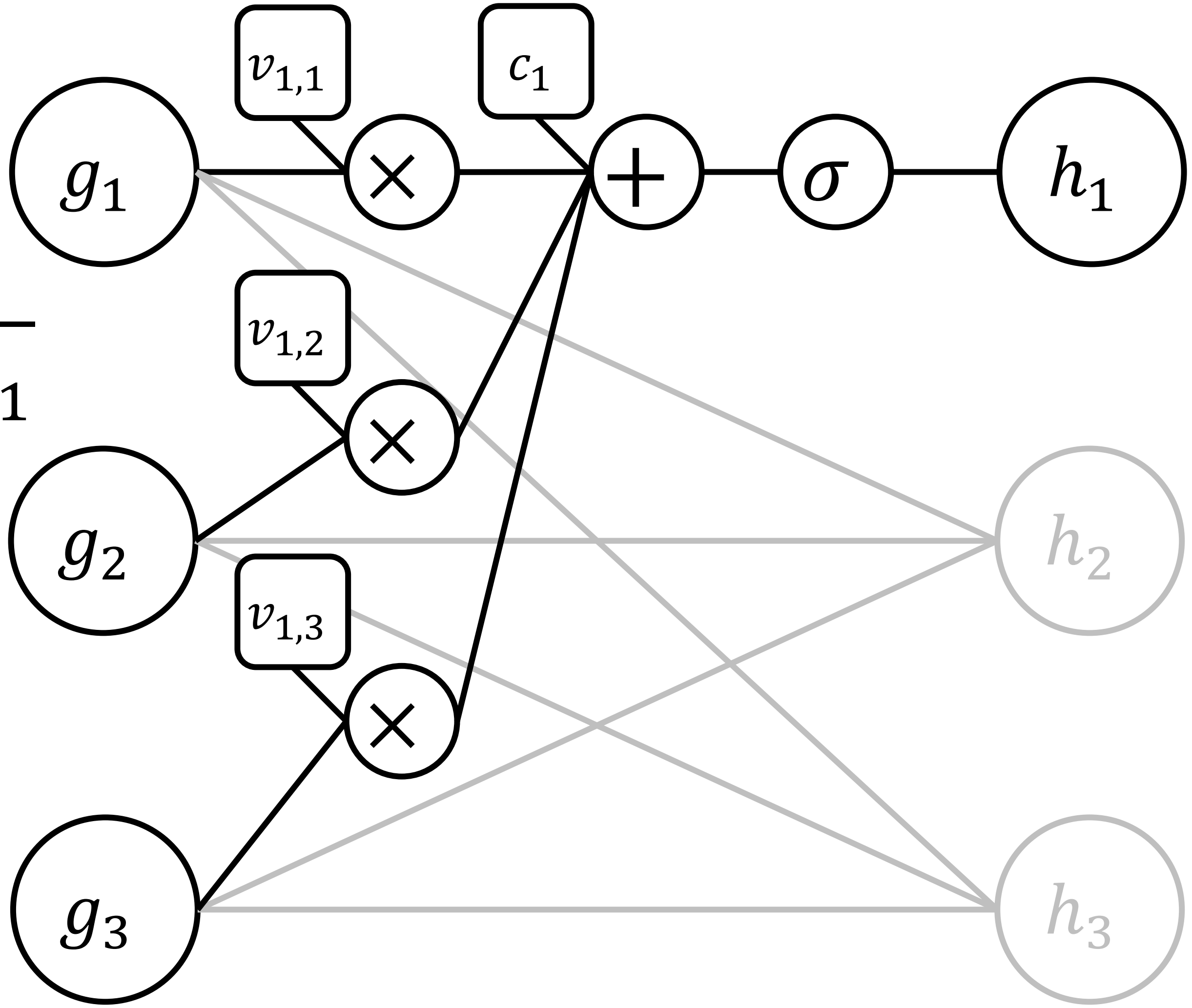
1番目の隠れ層～2番目の隠れ層



1番目の隠れ層～2番目の隠れ層（続き）

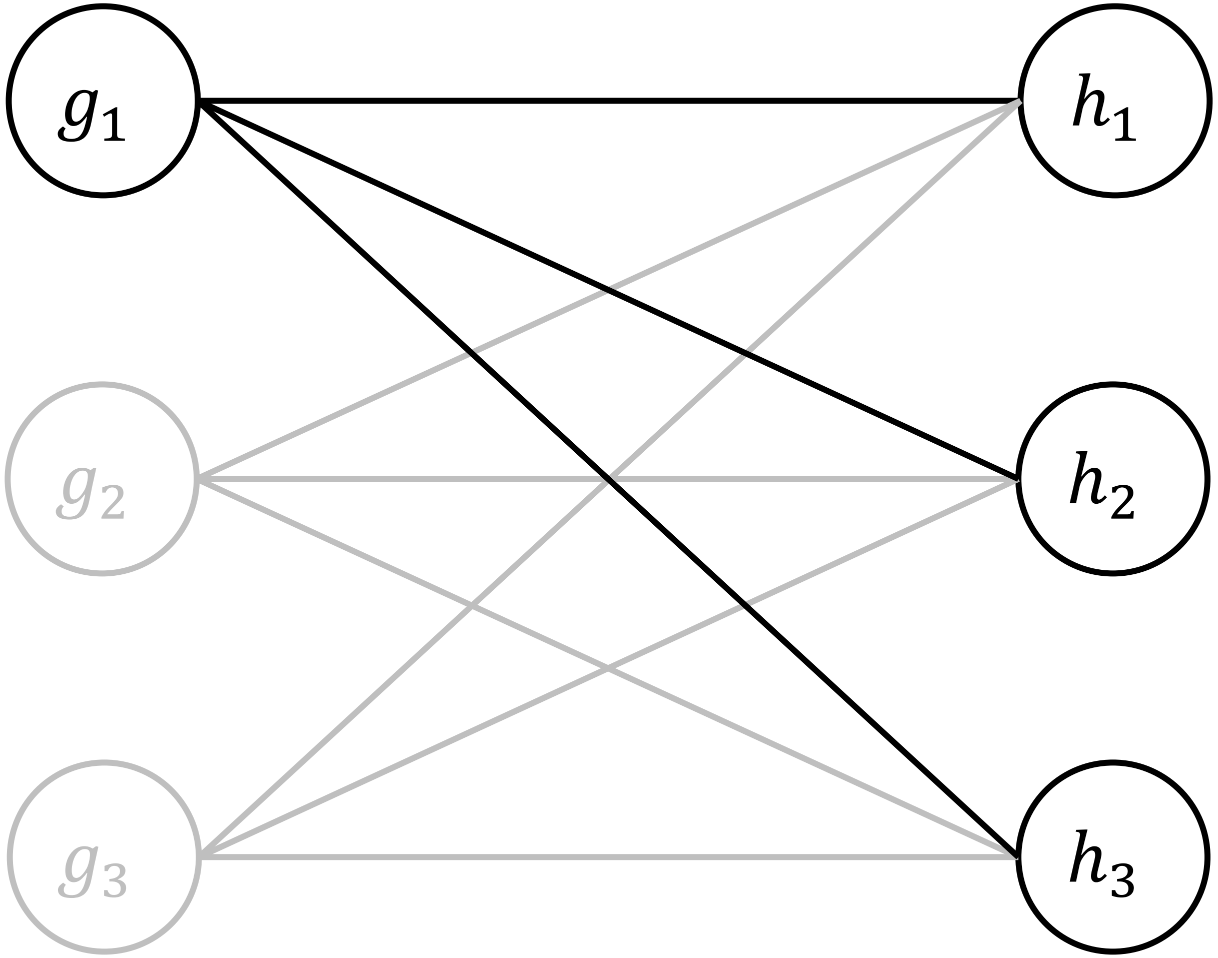
$$r_1 = v_{1,1}g_1 + v_{1,2}g + v_{1,3}g + c_1$$
$$h_1 = \sigma(r_1)$$

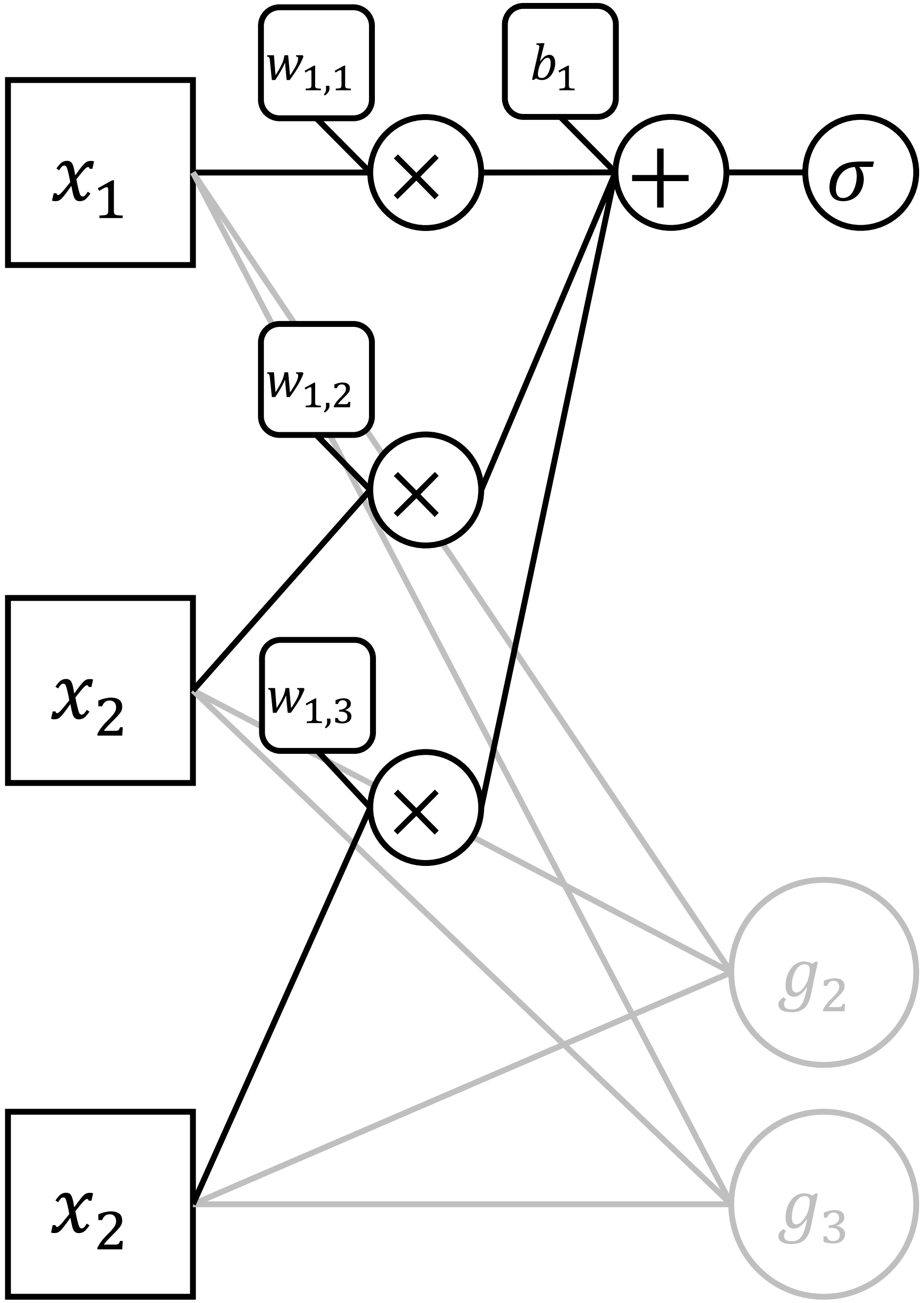
$$\frac{\partial L}{\partial v_{1,1}} = \frac{\partial L}{\partial h_1}\frac{\partial h_1}{\partial v_{1,1}} = \frac{\partial L}{\partial h_1}\frac{\partial h_1}{\partial r_1}\frac{\partial r_1}{\partial v_{1,1}}$$
$$= \begin{cases} \dfrac{\partial L}{\partial h_1} g_1 & r_1 \geq 0 \\ 0 & r_1 < 0 \end{cases}$$



誤差の合流

$$\frac{\partial L}{\partial g_1} = \frac{\partial L}{\partial h_1}\frac{\partial h_1}{\partial g_1} + \frac{\partial L}{\partial h_2}\frac{\partial h_2}{\partial g_1} + \frac{\partial L}{\partial h_3}\frac{\partial h_3}{\partial g_1}$$





$$q_1 = w_{1,1}x_1 + w_{1,2}x_2 + w_{1,3}x_3 + b_1$$
$$g_1 = \sigma(q_1)$$

$$\frac{\partial L}{\partial w_{1,1}} = \frac{\partial L}{\partial g_1}\frac{\partial g_1}{\partial w_{1,1}} = \frac{\partial L}{\partial g_1}\frac{\partial g_1}{\partial q_1}\frac{\partial q_1}{\partial w_{1,1}}$$
$$= \begin{cases} \frac{\partial L}{\partial g_1} x_1 & q_1 \geq 0 \\ 0 & q_1 < 0 \end{cases}$$

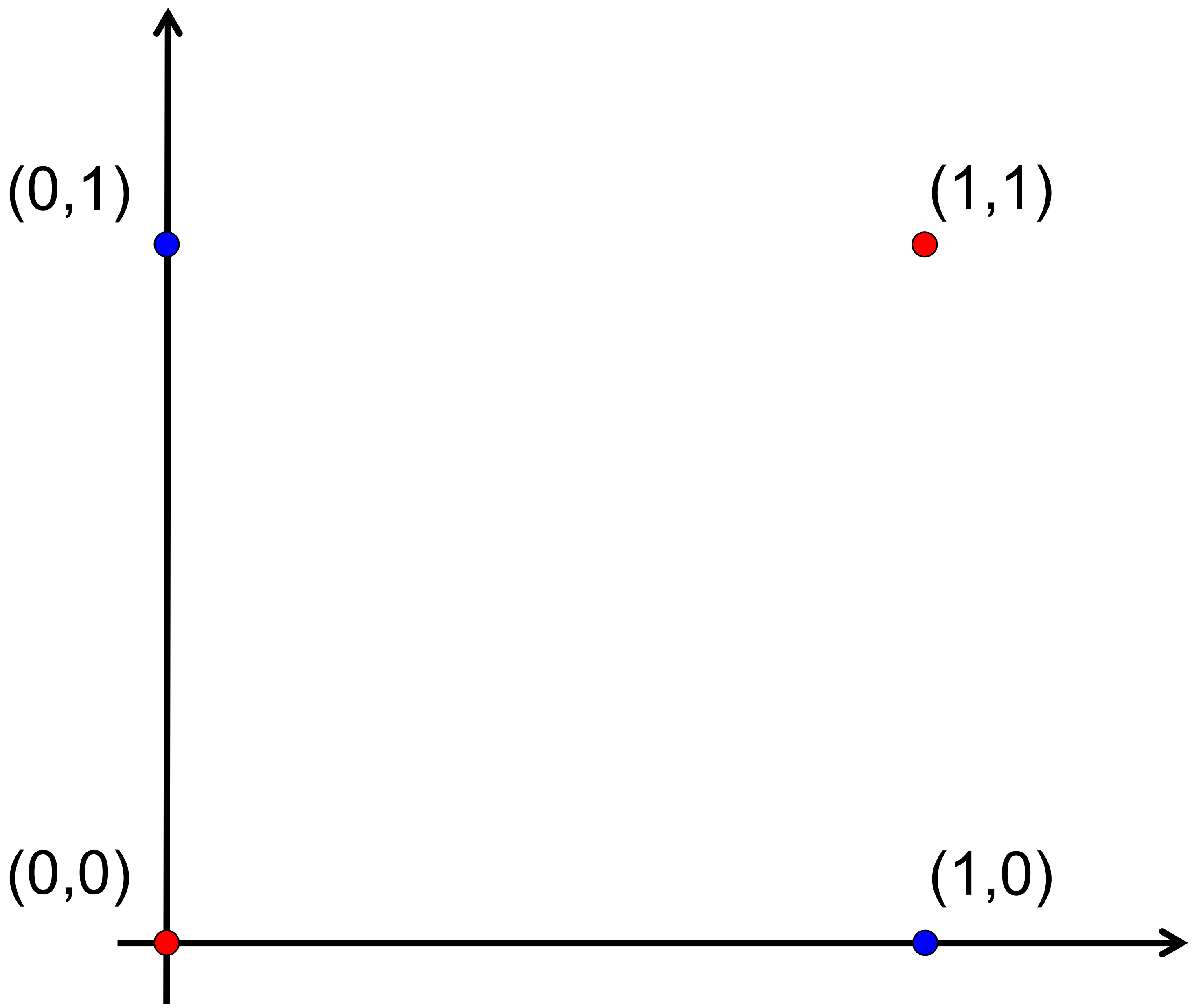
自動微分はフレームワークにまかせる

- 上述のBPは深層学習フレームワーク(PyTorchなど)に任せればよい
- 我々が考えるべきことは・・・（この試行錯誤に手間がかかる！）
    - 学習率をいくらにする？
    - ミニバッチに含まれる訓練データの個数を何個にする？
    - 隠れ層の数を何層にする？
    - 各層のサイズをいくらにする？
    - 活性化関数をどれにする？（基本的にReLUでよい）
    - Dropoutを使うか使わないか
    - 様々なNormalizationを使うか使わないか

scikit-learnのMLPを使ってみる

```
from sklearn.neural_network import MLPClassifier

X = [[0., 0.], [1., 1.], [0., 1.], [1., 0.]]
y = [0, 0, 1, 1]
clf = MLPClassifier(solver='adam', alpha=1e-5, max_iter=10000,
                    hidden_layer_sizes=(8, 8, 8), random_state=1)
clf.fit(X, y)
```


(0,1)
(1,1)
(0,0)
(1,0)

パラメータの個数を確認

```
print([c.shape for c in clf.coefs_])
print([i.shape for i in clf.intercepts_])

# coefs_は隠れ層と隠れ層の間の重み
# intercepts_は各隠れ層でのバイアス
```

分類境界の可視化

```
import numpy as np
import matplotlib.pyplot as plt

x1, x2 = np.mgrid[0:1:101j, 0:1:101j]
X = np.c_[x1.ravel(), x2.ravel()]
fig, ax = plt.subplots()
im = ax.imshow(clf.predict(X).reshape(101,101))
plt.show()
```

課題

- 多層パーセプトロンを使って、MNISTデータを0から9までの10種類の画像に分類してみよう
- 検証データによる評価を使って、良いアーキテクチャを探そう
- テストデータで予測を行い、confusion matrixを作成しよう

Confusion matrix（混同行列）

- 真のラベルと予測ラベルの対応関係を表にしたもの
  - 真のラベルがAであるテストデータについて、予測もAだったものの個数、予測がBだったものの個数、予測がCだったものの個数、等々を表にする。