

メディア情報学実験 情報検索，認識（画像）

2019/01/22

庄野 分

スタッフ

- 柳井 啓司 教授（後半 2 回分担当） yanai@cs.uec.ac.jp
- 庄野 逸（前半 1 回分担当） shouno@uec.ac.jp
- 五味 京祐 gomi-k@mm.inf.uec.ac.jp
- 岡本 開夢 okamoto-k@mm.inf.uec.ac.jp
- 川島 貴大 kawashima@uec.ac.jp
- 小林 源太 genta-kobayashi@uec.ac.jp

実験場所

- 2限は座学（西9-135）
- 3限は, CED
- 席の取り決めなどは特に考えない.

解説文書置き場

- とりあえず下記に説明などをおいてあります
<https://uecmediaexp.github.io/IntroductionDocument/>
- 課題提供と提出は github を用いて行います.

GPUマシンの割当

- 講義時間中は以下のGPUを使うこと
割当に従っていないプロセスや長時間占有しているプロセスは殺すこともあります.
- 1組 CED の GPU 21 台
gpu01.ced.cei.uec.ac.jp ~ gpu21.ced.cei.uec.ac.jp
- 2組 CED の GPU 10 台/IED の GPU 21台
gpu22.ced.cei.uec.ac.jp~gpu33.ced.cei.uec.ac.jp,
gpu1.ied.inf.uec.ac.jp, gpu2.ied.inf.uec.ac.jp#0~#2
- 3組 IED のGPU20台
gpu2.ied.inf.uec.ac.jp#3~#8, gpu3.ied.inf.uec.ac.jp,
gpu4.ied.inf.uec.ac.jp#0~#5

やってもらうこと（パターン認識）

- 1/21
回帰問題，パターン認識問題，ニューラルネットワーク
- 1/28
Keras を用いたパターン認識，CNN
- 2/4
発展課題

提出課題

- 教員の指示に従ってください
- 提出物
Jupyter hub 上で作成した ノートブック(拡張子は .ipynb)
- 提出方法
 - github 上で作成したノートブックを直接見ます.
ので, 作業ブランチ上で, 編集, 実験します.
 - 終わったらPull request を送ることで課題提出を行います.

課題 1

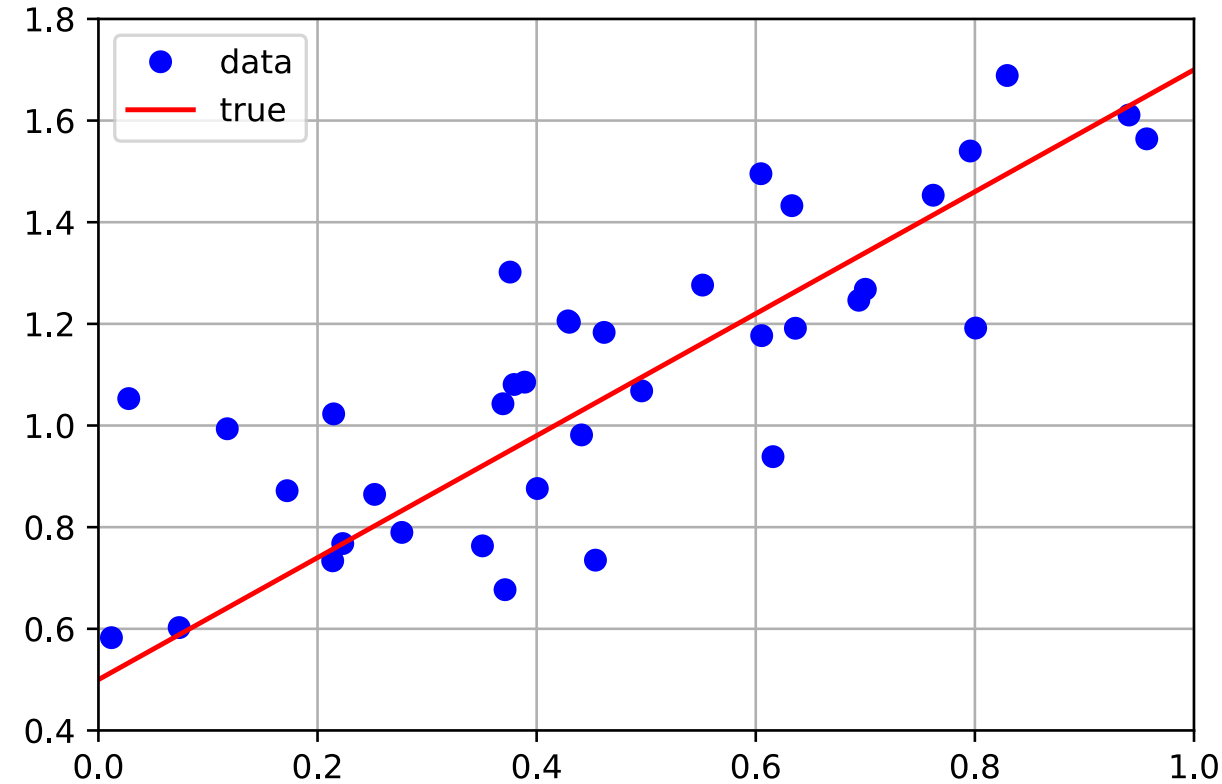
- 多層パーセプトロンを Keras を用いて実現し、MNISTデータセットの識別性能を調査する。
 - github で与えられた Jupyter notebook を用いて答えること
 - Keras を触ったことがあるなら一度は通る途です。
わかったヒトはさっくり進めてもらって構いません。
 - ただし演習課題も解くこと。解いていない場合は減点対象とします。
- 提出期限は 2週間後 (2/4) とします。

課題 1 をみて途方にくれた場合

- 課題 1 を見て途方にくれたヒトもいるかと思います.
- でも、諦めないでください. そういうヒトも解けるように演習問題を設定しています.
- 演習問題1-1 ～ 1-5 を順に解くことで課題を解くためのヒントが導かれます.
- なお演習問題の例解は PracticeHint ディレクトリにあります. 力をつけたい場合は, 解こうとしてから見ることを勧めます

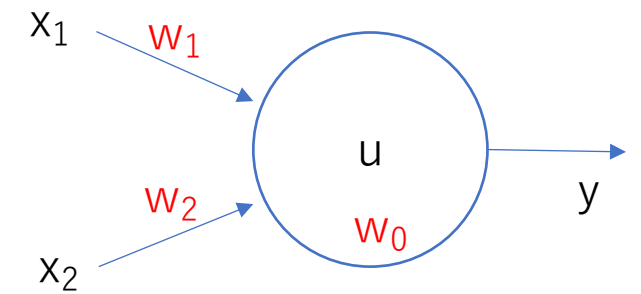
演習1-1: 回帰

- 1変数の回帰問題
- データ点群 $\{x_n, y_n\}$ から直線を推定したい
- 修得すべき概念:
 - ニューラルネットの設計
 - ロス関数



演習1-1: ニュラルネットの雑な理解

- ニューラルネットとは
 - 積和計算と非線形活性による変換関数
 - ノードとエッジによる計算
 - 多数の素子による協調計算
 - 任意精度の関数近似機械
- パラメータ w によっていろいろ化ける



$$y = f(u + w_0)$$

$$u = w_1 x_1 + w_2 x_2$$

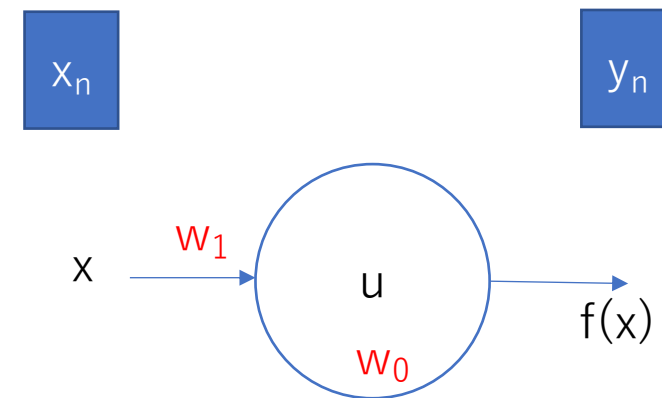
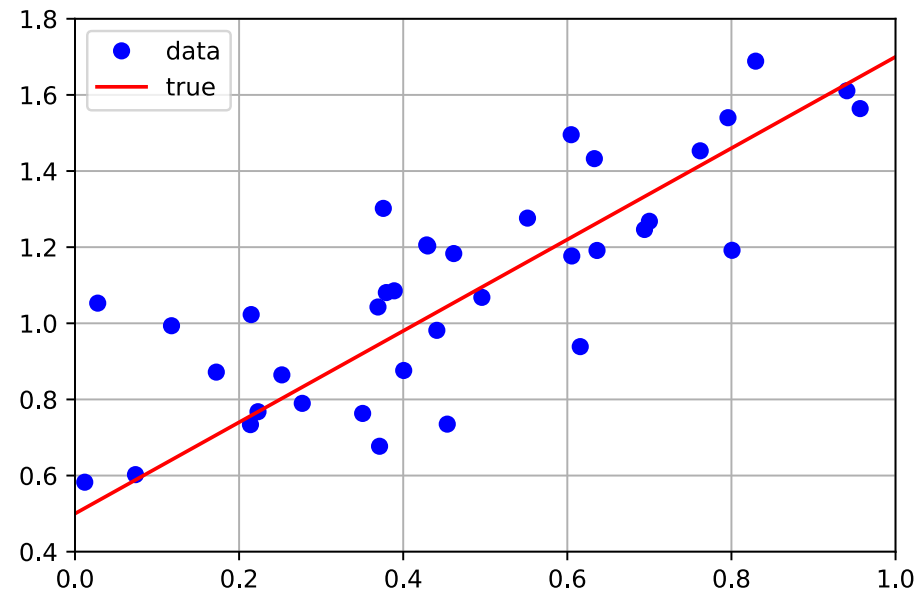
こんなのが大量につながってできる
計算機

演習1-1: ロス関数

- 解きたい課題は x_n を入力したときにできるだけ y_n に近い出力を吐き出す関数 $f(x)$ を求めること
- デザインとしては下のようなものを考えればよいかも

$$J(w) = \frac{1}{N} \sum_n (y_n - f(x_n))^2$$

y 座標値 NNの答え



$$y = u + w_0$$

$$u = w_1 x$$

演習1-1: Keras による実現

- 普通はこんな簡単な問題に用いないが,
1 入力 1 出力, 非線形変換なしのニューラルネット

```
import keras
from keras.models import Sequential
from keras.layers.core import Dense, Activation

model = Sequential()    # 階層型のモデルを選択
model.add(Dense(1, input_shape=(1,), use_bias=True)) # 素子が一個の改装モデル
```

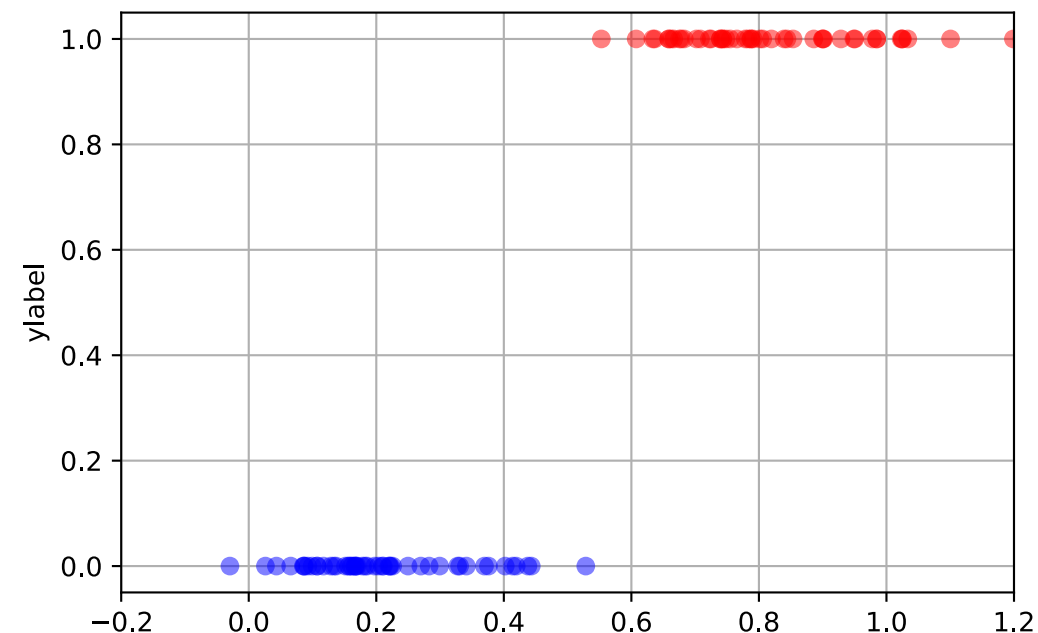
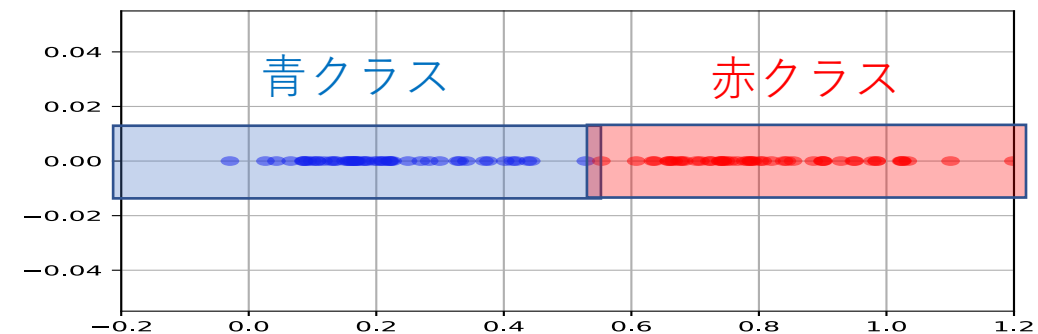
- ロスは平均二乗誤差なのでそれを選んで fit

```
model.compile(loss='mean_squared_error', optimizer='sgd') #最適化手法を指定

# 学習によるパラメータフィット
hist = model.fit(x, y, epochs=512, batch_size=10, verbose=1)
```

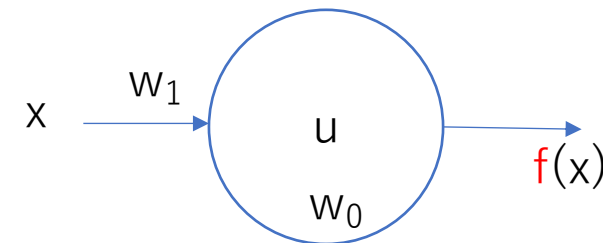
演習1-2: パターン認識課題 (1変数)

- 1変数のパターン認識
- ラベルつきデータ点群 $\{x_n, y_n\}$ から、境界を推定したい
- 修得すべき概念:
 - 活性化関数
 - ロジスティック回帰



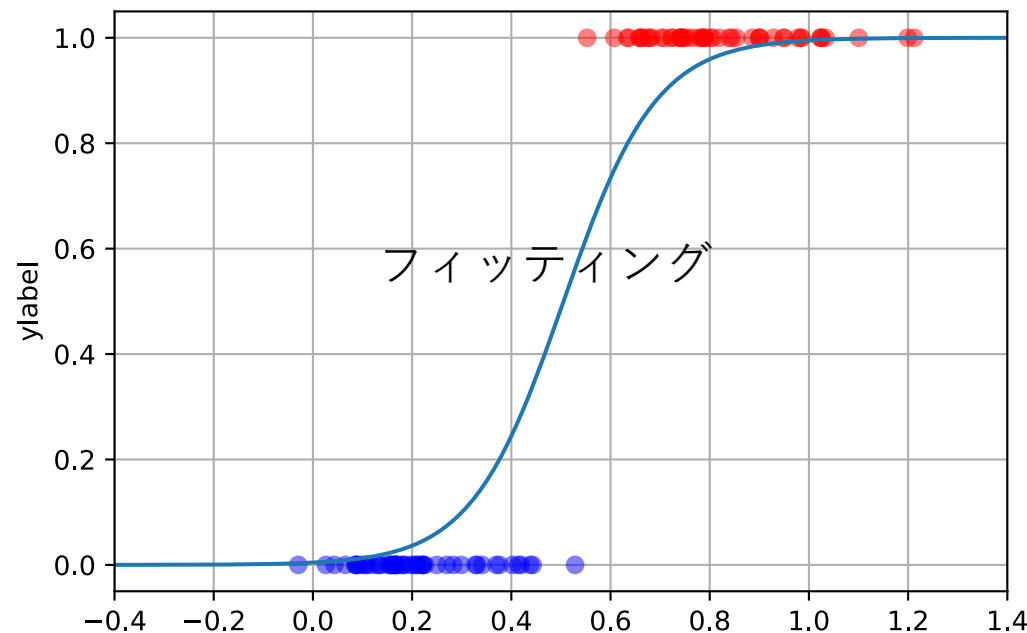
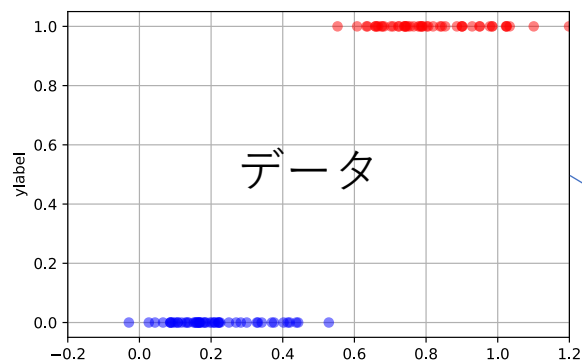
演習1-2: 活性化関数

- 欲しいのは x_n を入力したときに
妥当なラベル y_n を吐き出す関数 $f(x)$



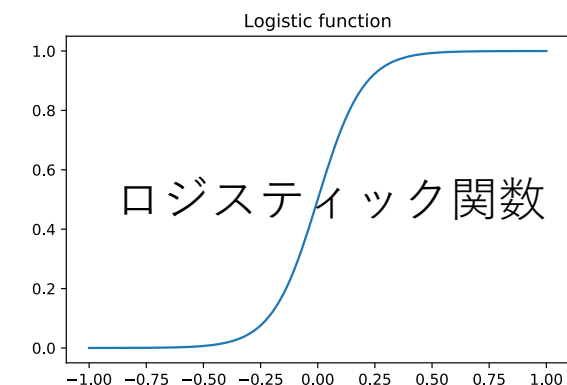
$$y = u + w_0$$

$$u = w_1 x$$



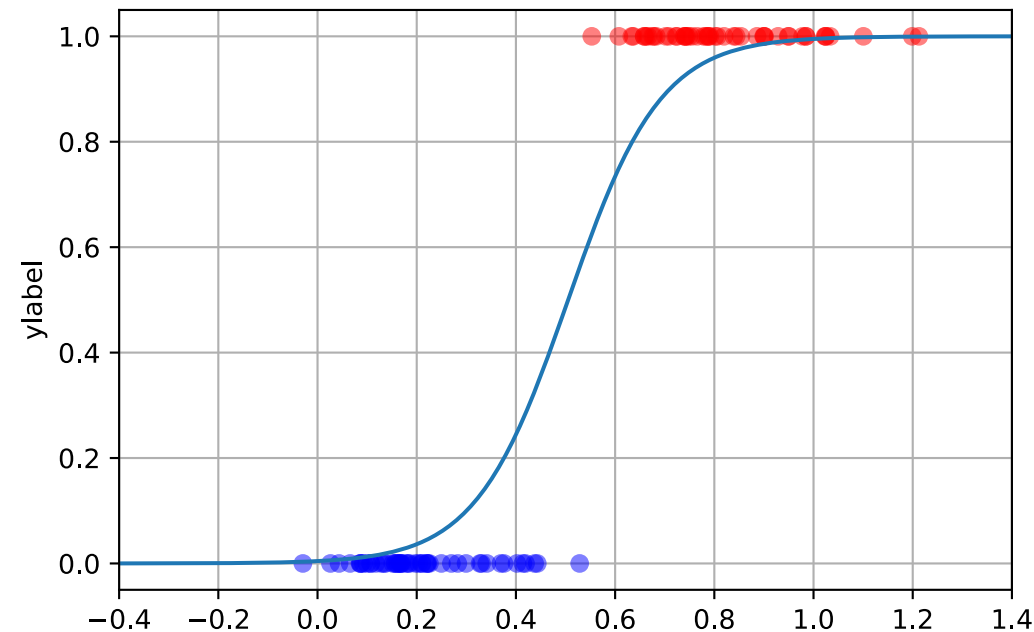
$f()$ をロジスティック関数
としてフィットする

1 に近ければ赤,
0 に近ければ青と判断



演習1-2: ロス関数

- クラス分類の場合は交差エントロピー関数をロス関数に使うのが一般的



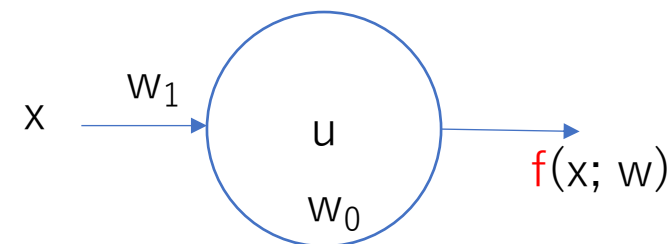
$$J(\mathbf{w}) = -\frac{1}{N} \sum_n y_n \log f(x_n) + (1 - y_n) \log(1 - f(x_n))$$

y 座標値

NNの答え

x_n

y_n



$$y = f(u + w_0)$$

$$u = w_1 x$$

演習1-2: Keras による実現

- 普通はこんな簡単な問題に用いないが,
1 入力 1 出力のニューラルネット

```
model = Sequential() # 階層型のモデルを選択
model.add(Dense(1, input_shape=(1,), use_bias=True))
model.add(Activation('sigmoid'))
```

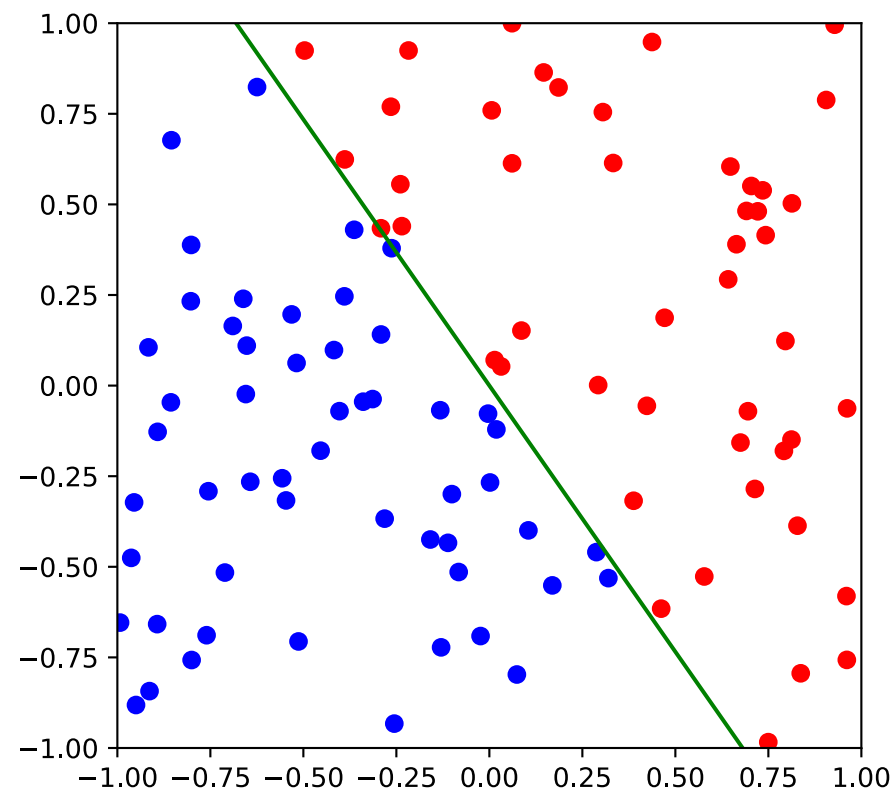
- ロスは交差エントロピーなのでそれを選んで fit

```
model.compile(loss='binary_crossentropy', optimizer='sgd') #最適化手法を指定

# 学習によるパラメータフィット
hist = model.fit(x, y, epochs=4096, batch_size=10, verbose=1)
```

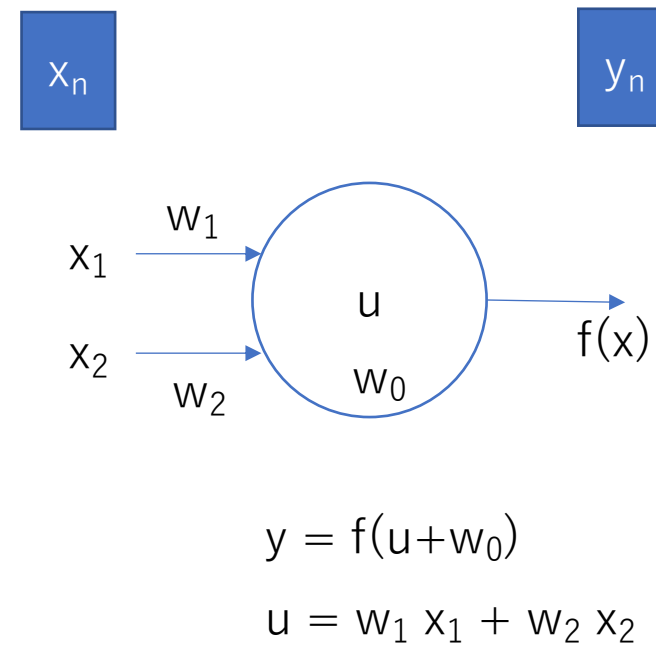
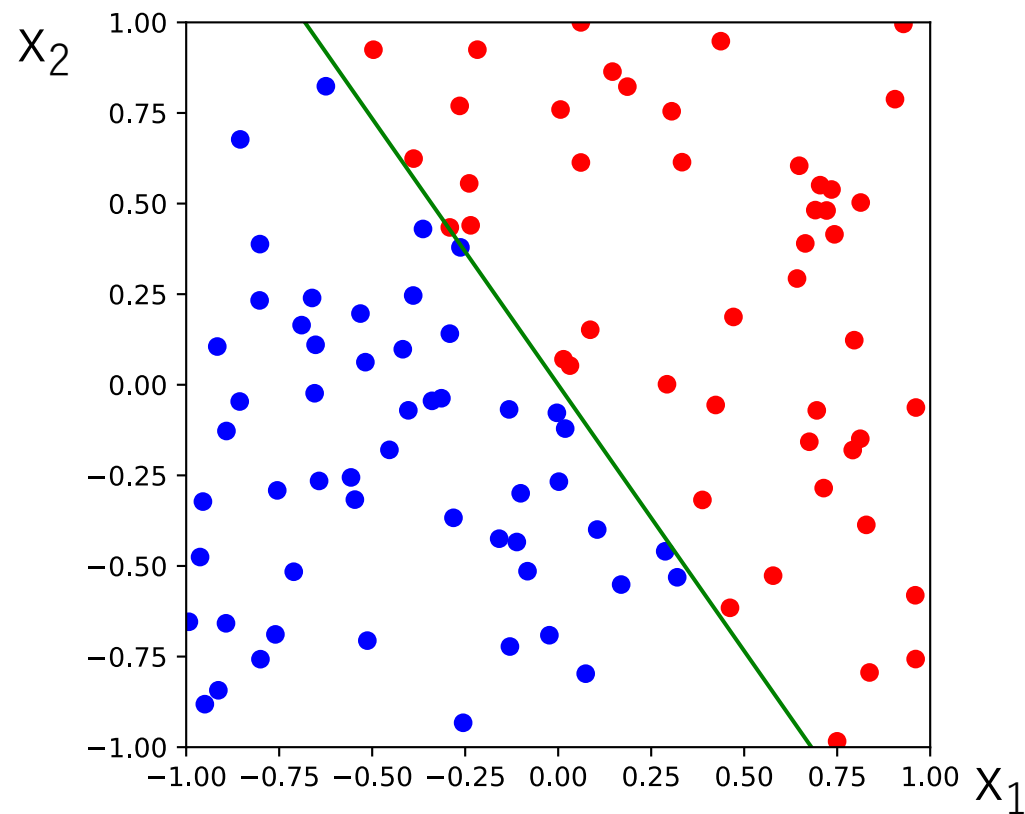
演習1-3: パターン認識課題 (多変数)

- 多変数のパターン認識
- ラベルつきデータ点群 $\{x_n, y_n\}$ から, 境界を推定したい
 - x_n がベクトルになる
- 修得すべき概念:
 - 多変数の場合の概念



演習1-3: ニューラルネットの形状

- 欲しいのは x_n を入力したときに
妥当なラベル y_n を吐き出す関数 $f(x)$



演習1-3: Keras による実現

- 普通はこんな簡単な問題に用いないが,
2入力 1 出力のニューラルネット

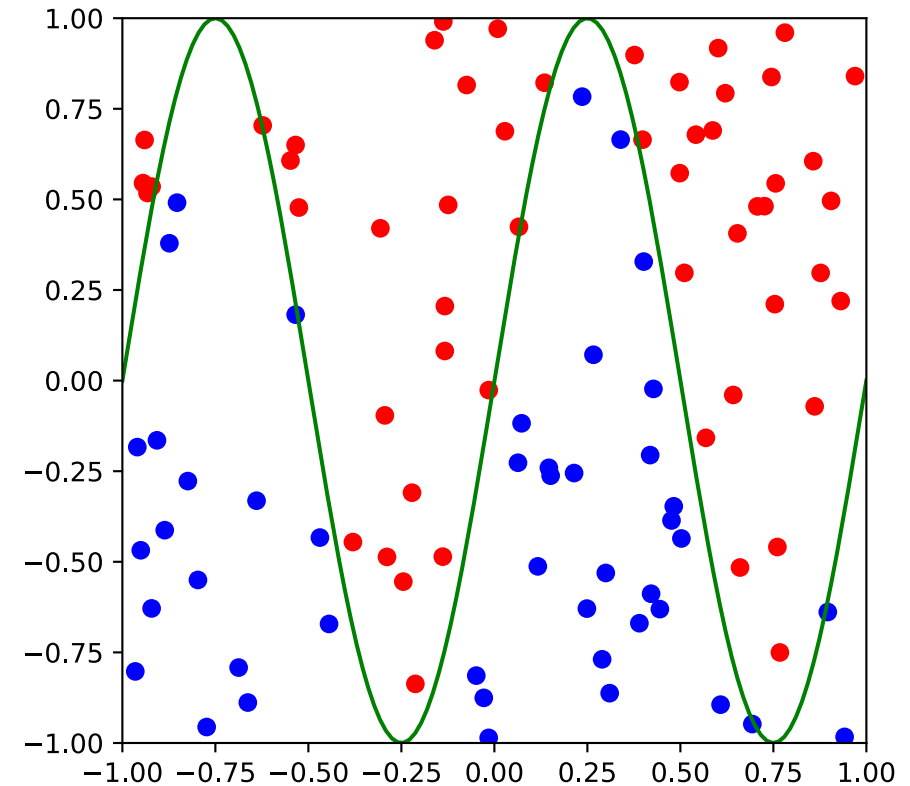
```
model = Sequential() # 階層型のモデルを選択
model.add(Dense(1, input_shape=(2,), use_bias=True)) # 入力が2次元になるところだけが違う
model.add(Activation('sigmoid'))
```

- ロスは交差エントロピーで fit

```
# 学習によるパラメータフィット
hist = model.fit(x, y, epochs=4096, batch_size=10, verbose=1)
```

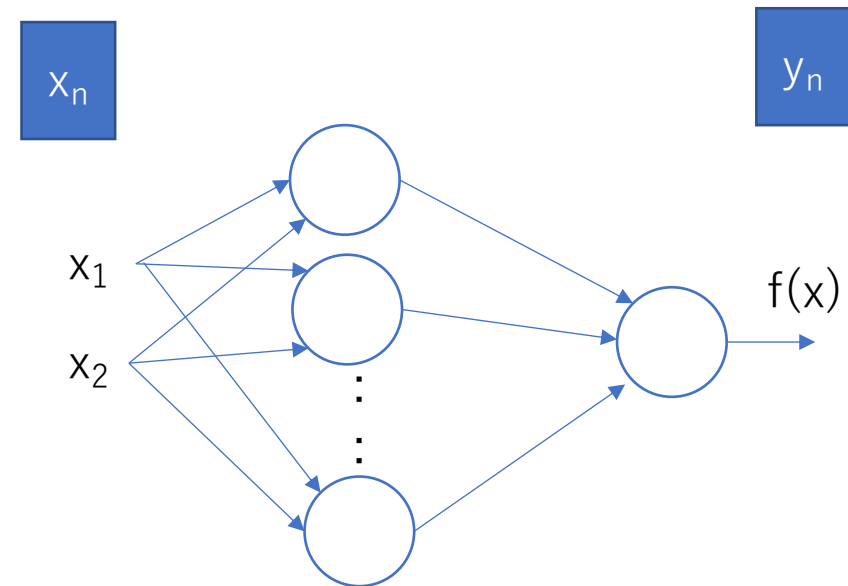
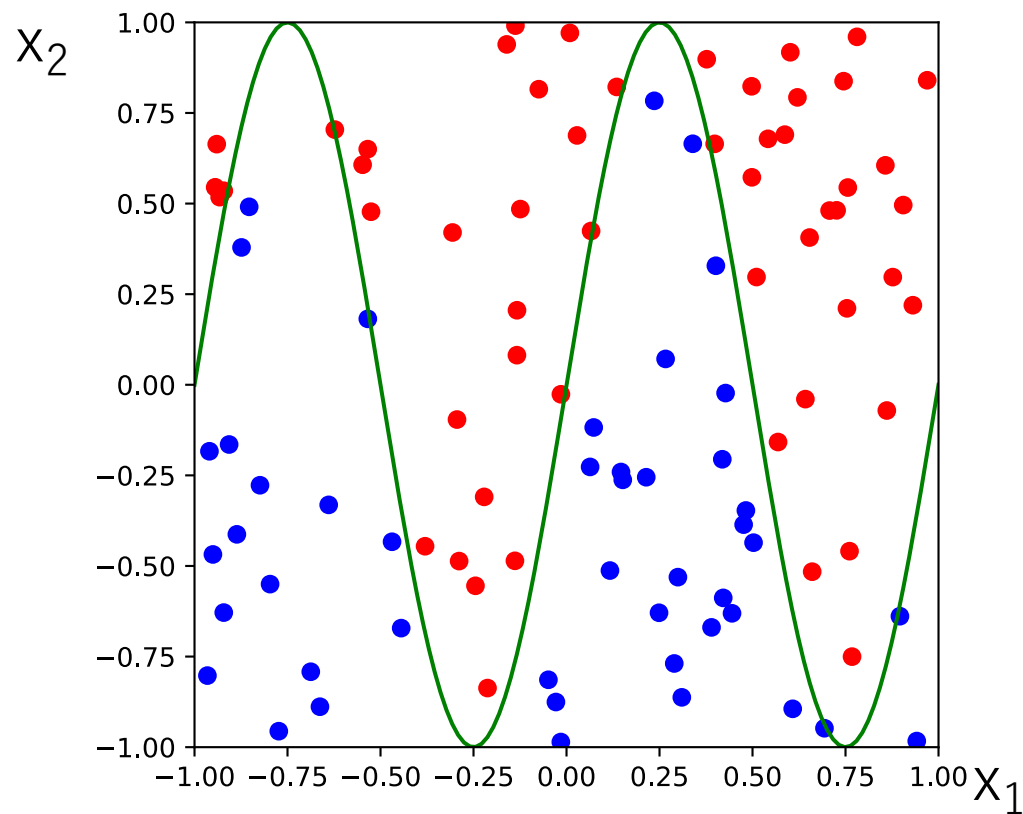
演習1-4: パターン認識課題: MLP

- 2変数のパターン認識
- ラベルつきデータ点群 $\{x_n, y_n\}$ から, 境界を推定したい
 - 境界が複雑で直線で分離出来ない
- 修得すべき概念:
 - 多層化の概念



演習1-4: ニューラルネットの形状

- 欲しいのは x_n を入力したときに
 妥当なラベル y_n を吐き出す関数 $f(x)$



直線では分離できなさそうなので中間層を増やす

演習1-4: Keras による実現

- 中間層があるので MLP となる

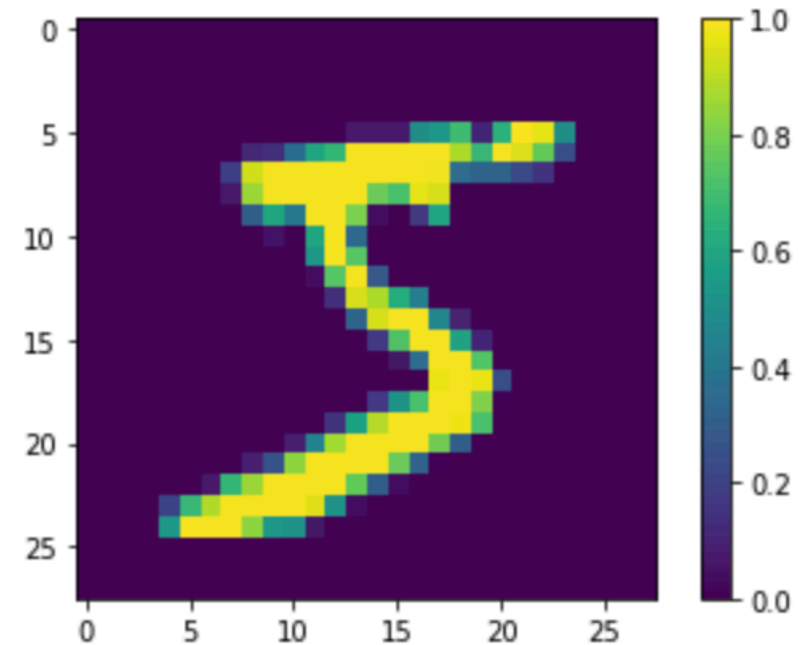
```
model = Sequential() # 階層型のモデルを選択
model.add(Dense(10, input_shape=(2,), use_bias=True)) # まず10個からなる中間層を構成
model.add(Activation('relu')) # 1層目を relu で非線形変換しておく(多分sigmoid でもok)
model.add(Dense(1)) # 前層の10個の表現を1個にまとめる(この部分は logistic 回帰のまま)
model.add(Activation('sigmoid'))
```

- ロスは交差エントロピーで fit

```
# 学習によるパラメータフィット
hist = model.fit(x, y, epochs=8192, batch_size=10, verbose=1)
```

演習1-5: MNISTをロジスティック回帰

- 2変数のパターン認識
- ラベルつきデータ点群 $\{x_n, y_n\}$ から、境界を推定したい
- 修得すべき概念:
 - MNISTの取扱



28 x 28 = 784 次元の入力

課題 1

- 多層パーセプトロンを Keras を用いて実現し、MNISTデータセットの識別性能を調査する。
 - github で与えられた Jupyter notebook を用いて答えること
 - Keras を触ったことがあるなら一度は通る途です。
わかったヒトはさっくり進めてもらって構いません。
 - ただし演習課題も解くこと。解いていない場合は減点対象とします。
- 提出期限は 2/4 とします。