メディア情報学実験
情報検索，認識（画像）

2022/01/20

庄野 逸 (shouno@uec.ac.jp)

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

- 寺内 健人（柳井研M1）
- 本部 勇真（柳井研M1）
- 村上 諒　（庄野研M2）
- 樋口 陽光（庄野研M1）

実験場所

- 2限は実験概要の説明（on Zoom）
  https://bit.ly/UEC-MEDIAEX
  3限は各自実験，質問場所用として上記Zoom開きます.

- オンラインなので席の取り決めなどは当然ありません.
- 通信環境などが不安な場合は IED を使うこともOKですが，各自の安全に留意してください.

実験場所（続き）

- 計算機実験は IED/CED の計算リソースを使います．
  （自前で計算リソースがあれば，それでもOK）
- IED/CED の計算リソース(GPU)は，学外からは見えないので，使う際には VPNを使って学内ネットワークの傘下に入る必要があります．

VPN設定について

- VPNに関しては情報基盤センターの指針に従ってください．VPNの設定は下記参照
  https://www.cc.uec.ac.jp/ug/ja/remote/vpn/

- VPNの傘下に入った場合，proxy を設定しないと学外が見えません(zoom などがアウトに)
- しかし，proxy を使って大学内部のホストを見ると変なことが起こります
  → proxy の除外設定を適切に

```
proxy.uec.ac.jp:8080; 172.21.0.0/16;
*.uec.ac.jp
```

解説文書置き場

- 実験自体は Jupyter Notebook のテンプレートを配りますので，それを使って
  - Python のコードを組み，
  - 機械学習や画像認識の実験に取り組む
  - 実験結果，考察をファイルとしてをアップロード

  という流れになります．
- 下記に説明などをおいておきます．
  https://uecmediaexp.github.io/IntroductionDocument/

GPUマシンの割当

- IED/CED の情報リソースは下記を参照（学内/VPN経由で見れます）
  http://gpu1.ied.inf.uec.ac.jp/
  http://gpu01.ced.cei.uec.ac.jp/
- IED {gpu1~gpu4}.ied.inf.uec.ac.jp#0~#7 の32枚
  CED {gpu01~gpu33}.ced.cei.uec.ac.jp の30枚
  なので1人1枚の割り当てはないです．　（25, 27, 30は不調）
  ゆるい割り当てとして下記の原則を指定しておきます
  - 学籍番号の下３桁が450未満
    CED の GPU 30 枚 {gpu02～gpu33}.ced.cei.uec.ac.jp
  - 学籍番号の下３桁が450以上
    IED　の GPU 32 枚 {gpu1~gpu4}.ied.inf.uec.ac.jp#0~#7

GPUマシンの使用時のマナー

- 空いているGPU の Jupyter Hub を用いてください
  リソース管理のページで確認できます
  - 他の人とバッティングした場合，メモリ不足（Out of Memory）と怒られるケースがあります.
- 計算を終えた場合，あるいは動かさない場合は
  Jupyter Notebook の “Running” タブから
  要らないものを Terminate してください.
- 長時間占有している計算は殺すこともあります.

やってもらうこと（パターン認識）

- 2022/01/20
  回帰・パターン認識問題，ニューラルネットワーク

- 2022/01/27
  Keras を用いたパターン認識，CNN

- 2022/02/03
  発展課題

提出課題

- 各教員の指示に従ってください
- 提出物
  Jupyter 上で作成した ノートブック(.ipynb拡張子)
  を PDF に変換してください.
- 提出方法
  - 提出場所は https://mm.cs.uec.ac.jp/media/

課題１（2022/01/20出題）

- 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.
  - 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.
  - 演習課題も解いておくこと（解答例ついています）.
    解いた（写経して）ものも提出課題に含みます.

- 提出期限は 2週間後（2022/02/03） とします.

課題１をみて途方にくれた場合

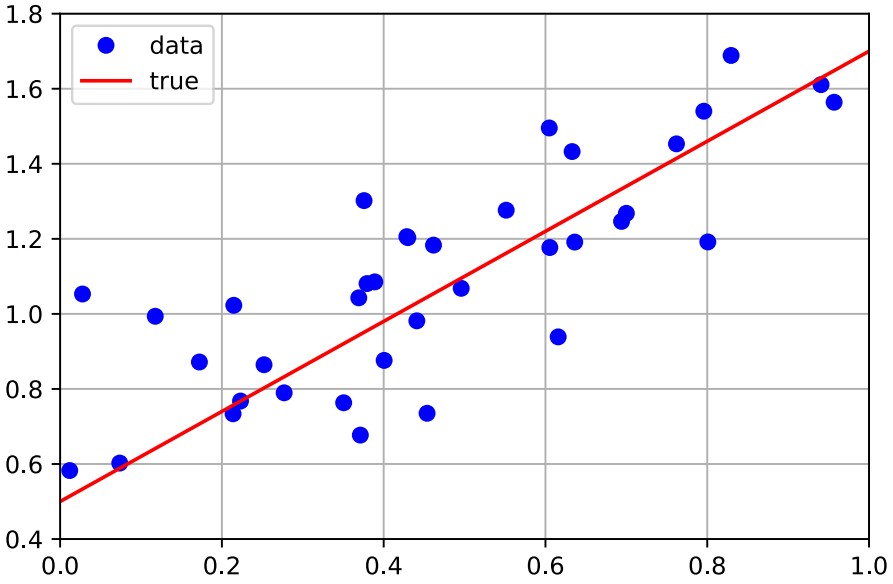
- 課題１を見て途方にくれたヒトもいるかと思います.
- でも，諦めないでください．解けるように解答例つき演習問題を設定しています.

- 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

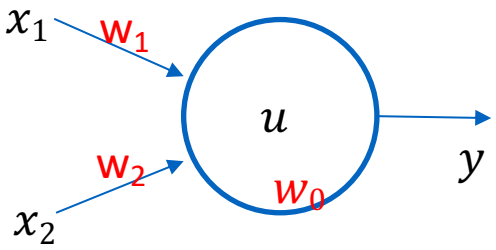
- 1変数の回帰問題
- データ点群 $\{x_n, y_n\}$ から直線を推定したい

- 修得してほしい概念:
  - 機械学習のやりたいこと
  - ロス関数
  - ニューラルネットの設計

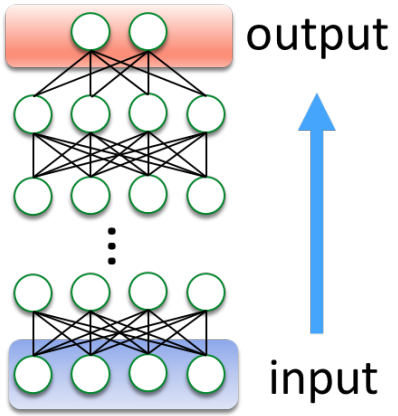


演習1-1: ニュラルネットの雑な理解

- ニューラルネットとは
  単純な演算ユニットを沢山集めて作られた計算機
- 構成ユニット
  - 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
- これが大量につながってできると
  - 多数の素子による並列，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



$$y = f(u + w_0)$$
$$u = w_1 x_1 + w_2 x_2$$

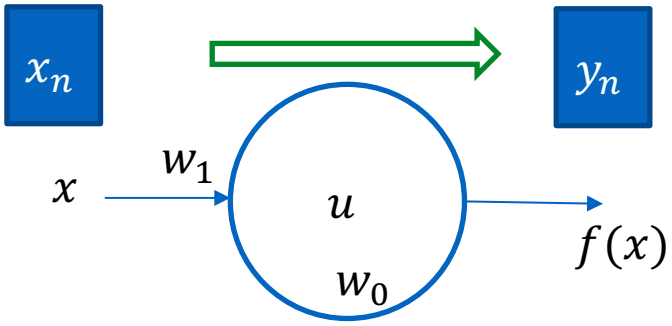


演習1-1: ロス関数

- 解きたい課題は，x_n を入力したときに
  y_n に近い出力を吐き出す関数 f(x) を構成すること
- デザインとしては下のようなものを考えればよい

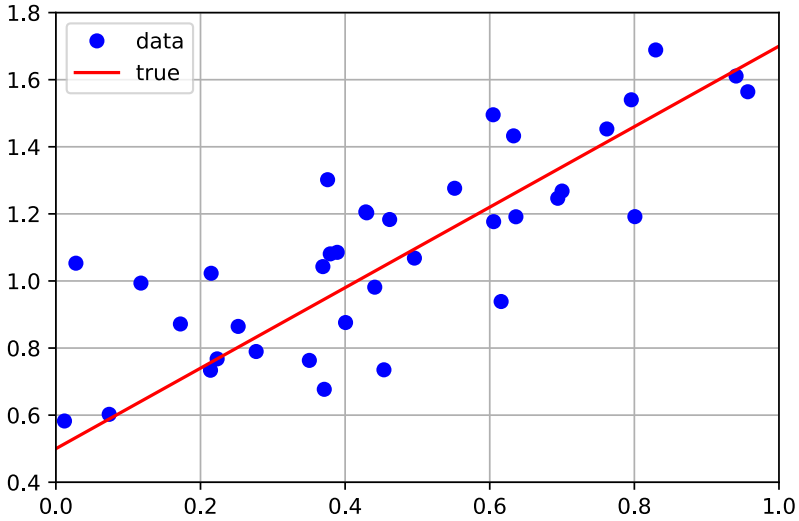
$$J(w) = \frac{1}{N}\sum_n (y_n - f(x_n))^2$$

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

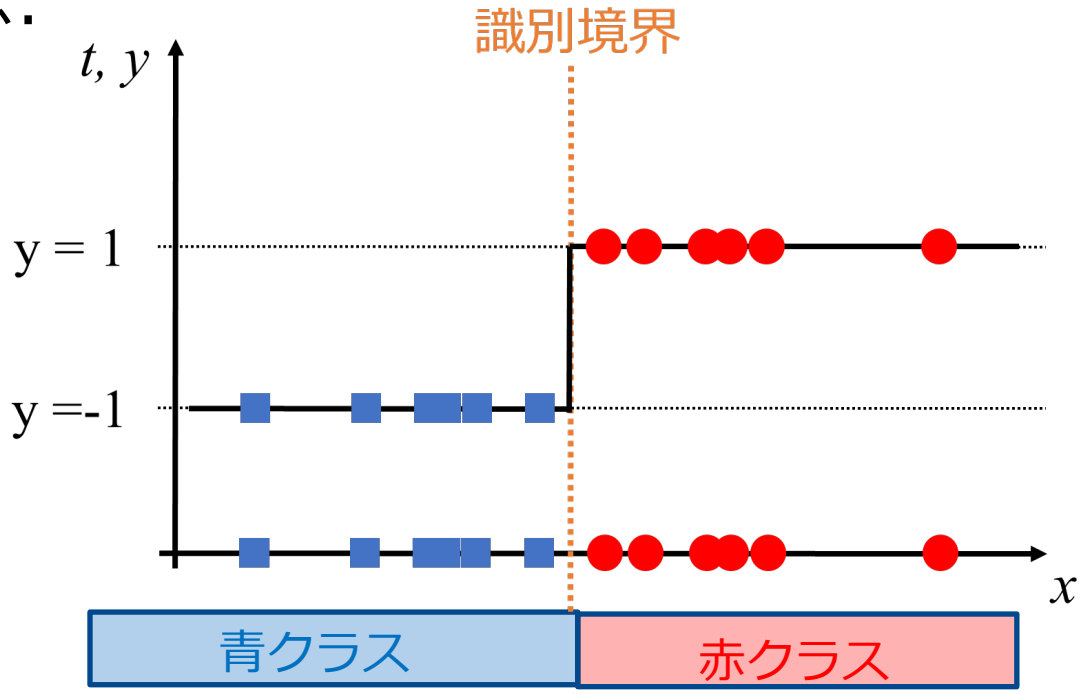
```
model.compile(loss='mean_squared_error',  optimizer='sgd') #最適化手法を指定

# 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）

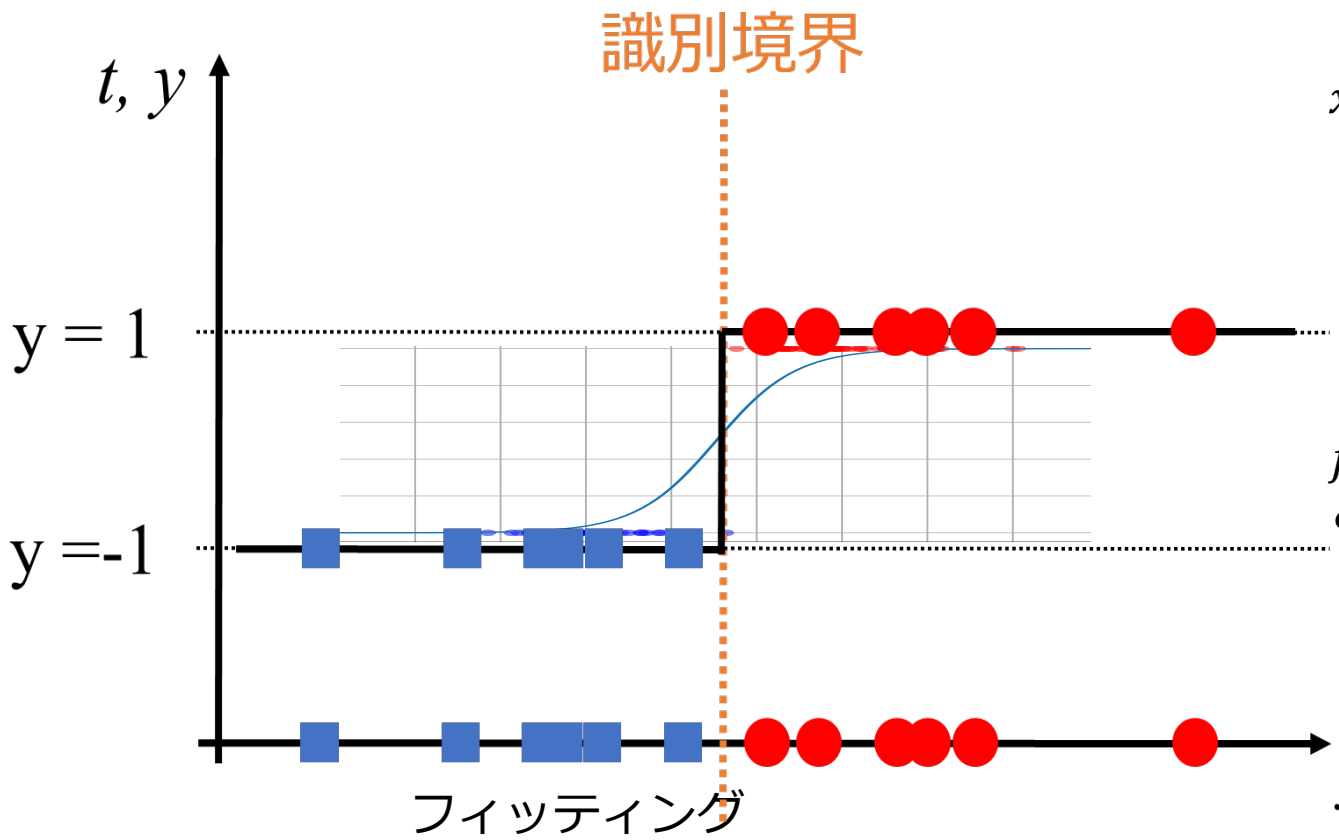
- 1変数のパターン認識
- ラベル付データ点群 $\{x_n, y_n\}$ から，境界を推定したい

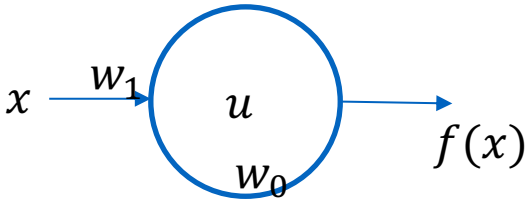
- 修得してほしい概念:
  - 活性化関数
  - ロジスティック回帰



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

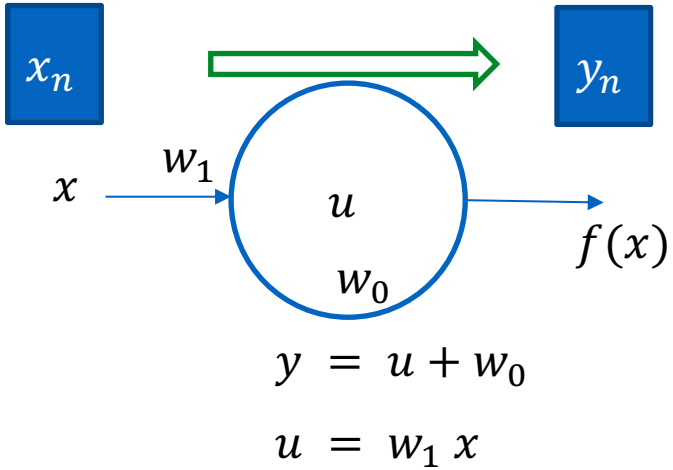
1 に近ければ赤,
0 に近ければ青と判断

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

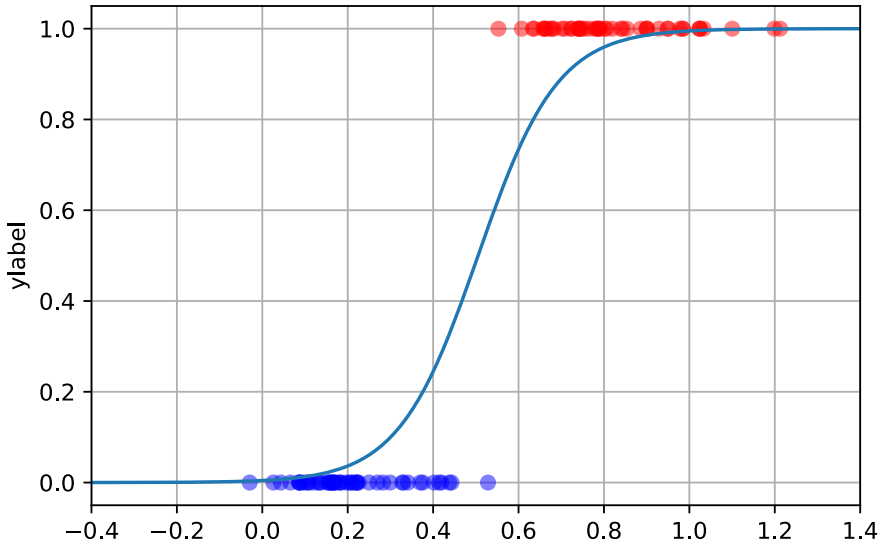
$$J(\boldsymbol{w}) = -\frac{1}{N}\sum_{n} y_n \log f(x_n) + (1 - y_n)\log(1 - f(x_n))$$

y 座標値　　NNの答え



$$y = u + w_0$$

$$u = w_1 x$$



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'))
```

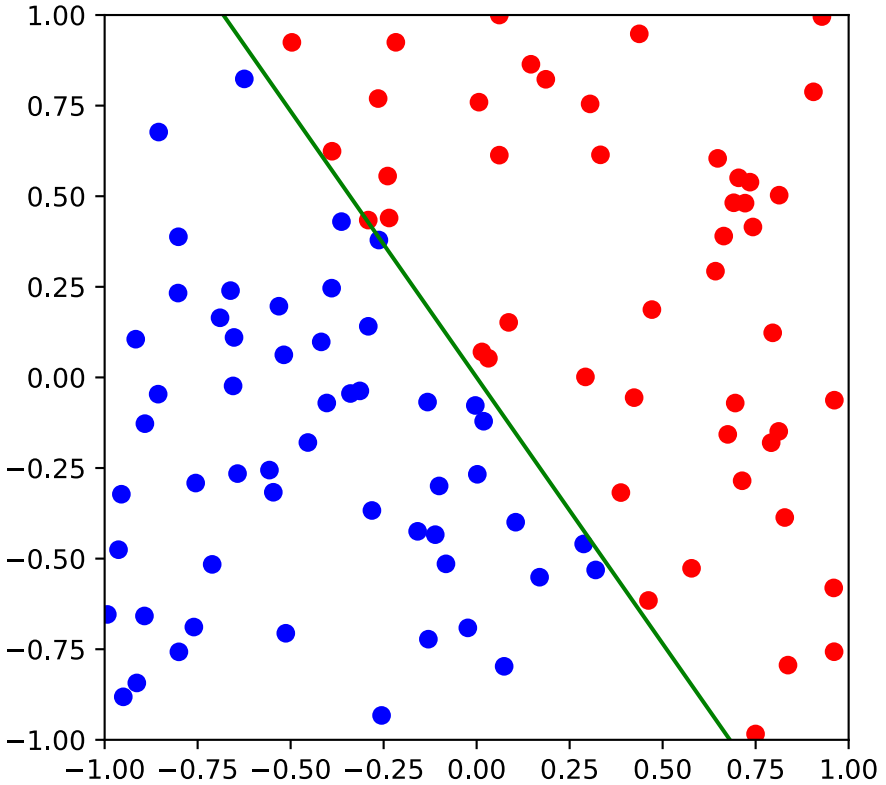
- ロスは交差エントロピーなのでそれを選んで fit

```
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)
```

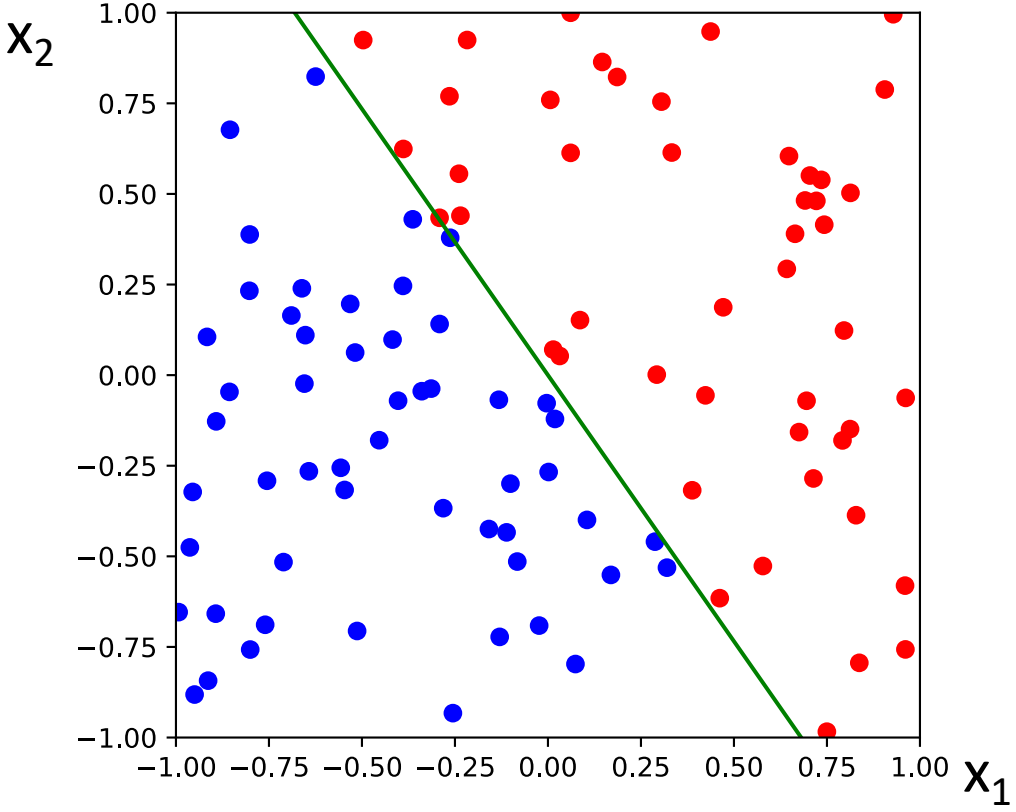
演習1-3: パターン認識課題（多変数）

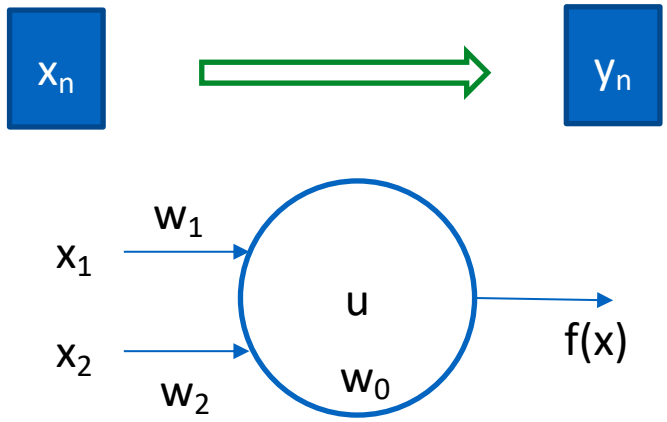
- 多変数のパターン認識
- ラベルつきデータ点群 $\{\boldsymbol{x}_n, y_n\}$ から，境界を推定
  - $\boldsymbol{x}_n$ がベクトルになる
- 修得してほしい概念:
  - 多変数の場合への拡張



演習1-3: ニューラルネットの形状

- 欲しいのは $\boldsymbol{x}_n$ 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)





$y = f(u+w_0)$

$u = w_1 x_1 + w_2 x_2$

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'))
```

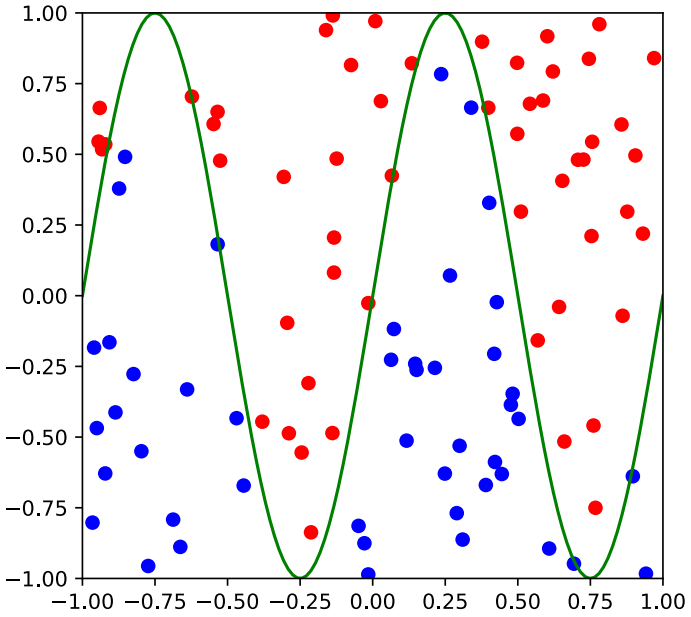
- ロスは交差エントロピーで fit

```
# 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

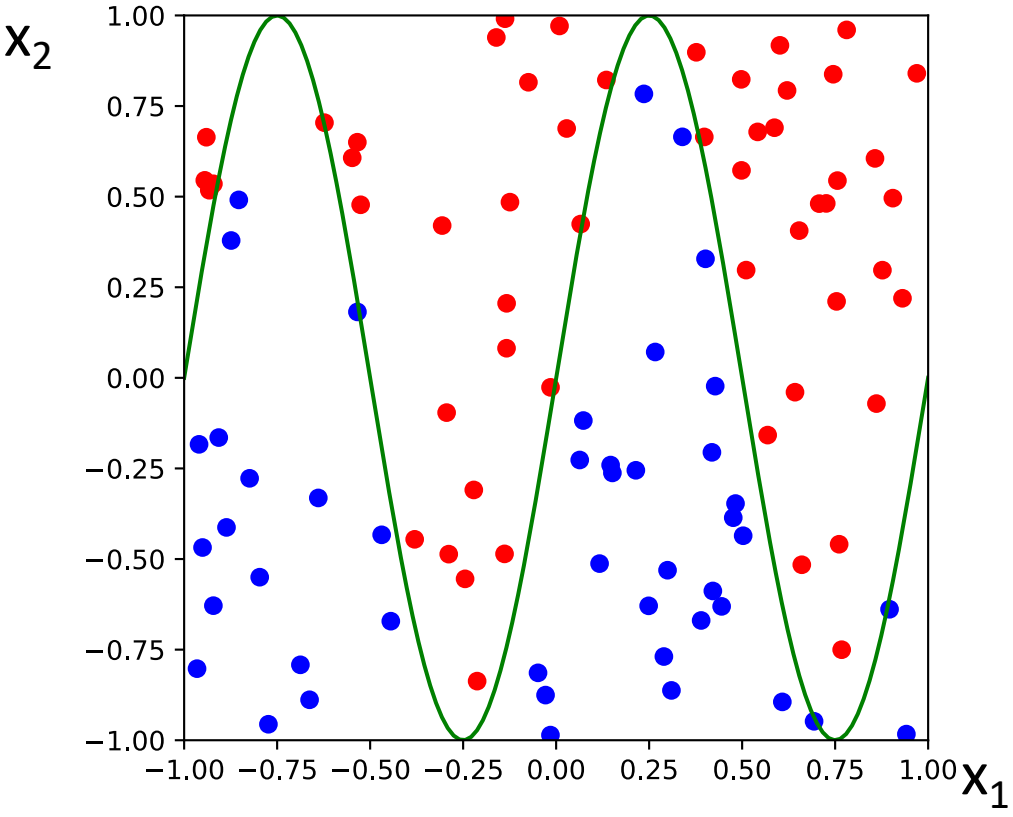
- 2変数のパターン認識
- ラベルつきデータ点群 $\{x_n, y_n\}$ から，境界を推定
  - 境界が複雑で直線で分離出来ない

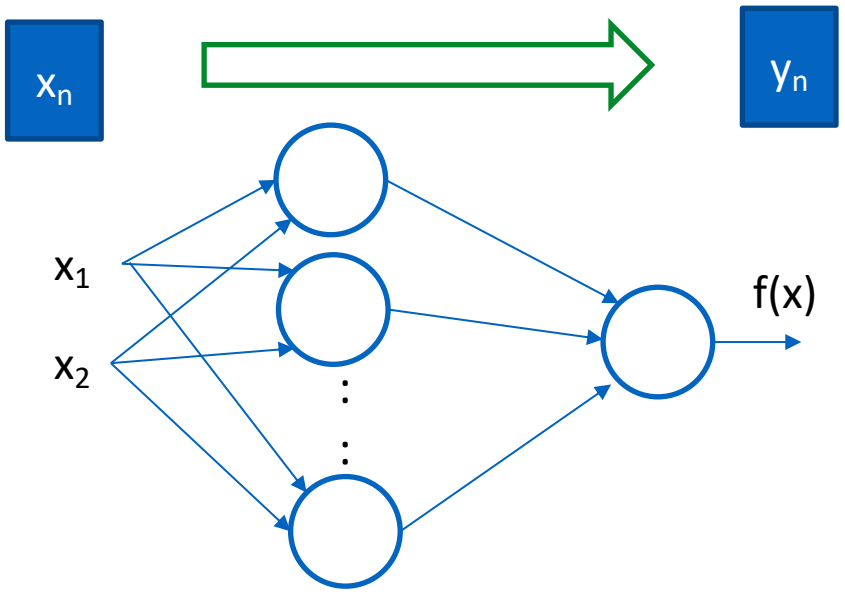
- 修得してほしい概念:
  - 多層化によってできることの確認



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

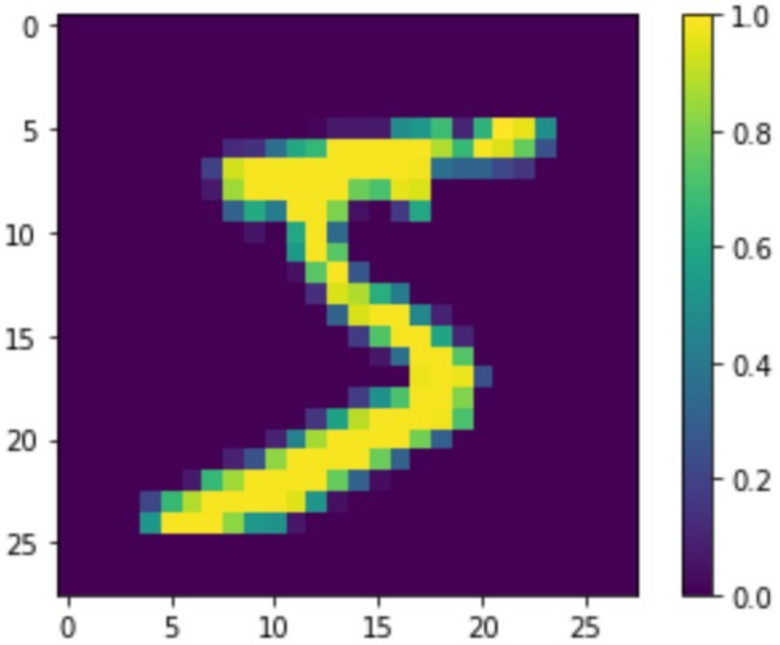
```
model = Sequential()  # 階層型のモデルを選択
model.add(Dense(10, input_shape=(2,), use_bias=True))  # まず10個からなる中間層を構成
model.add(Activation('relu')) # 1層目を relu で非線形変換しておく(多分sigmoid でもおk)
model.add(Dense(1))#前層の10個の表現を1個にまとめる(この部分は logistic 回帰のまま
model.add(Activation('sigmoid'))
```

- 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 $\{\boldsymbol{x}_n, y_n\}$ から，境界を推定したい
- 修得すべき概念:
  - MNISTの取扱
  - 多クラス分類



28 x 28 = 784 次元の入力

課題 1

- 多層パーセプトロン(MLP)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.
  - 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.