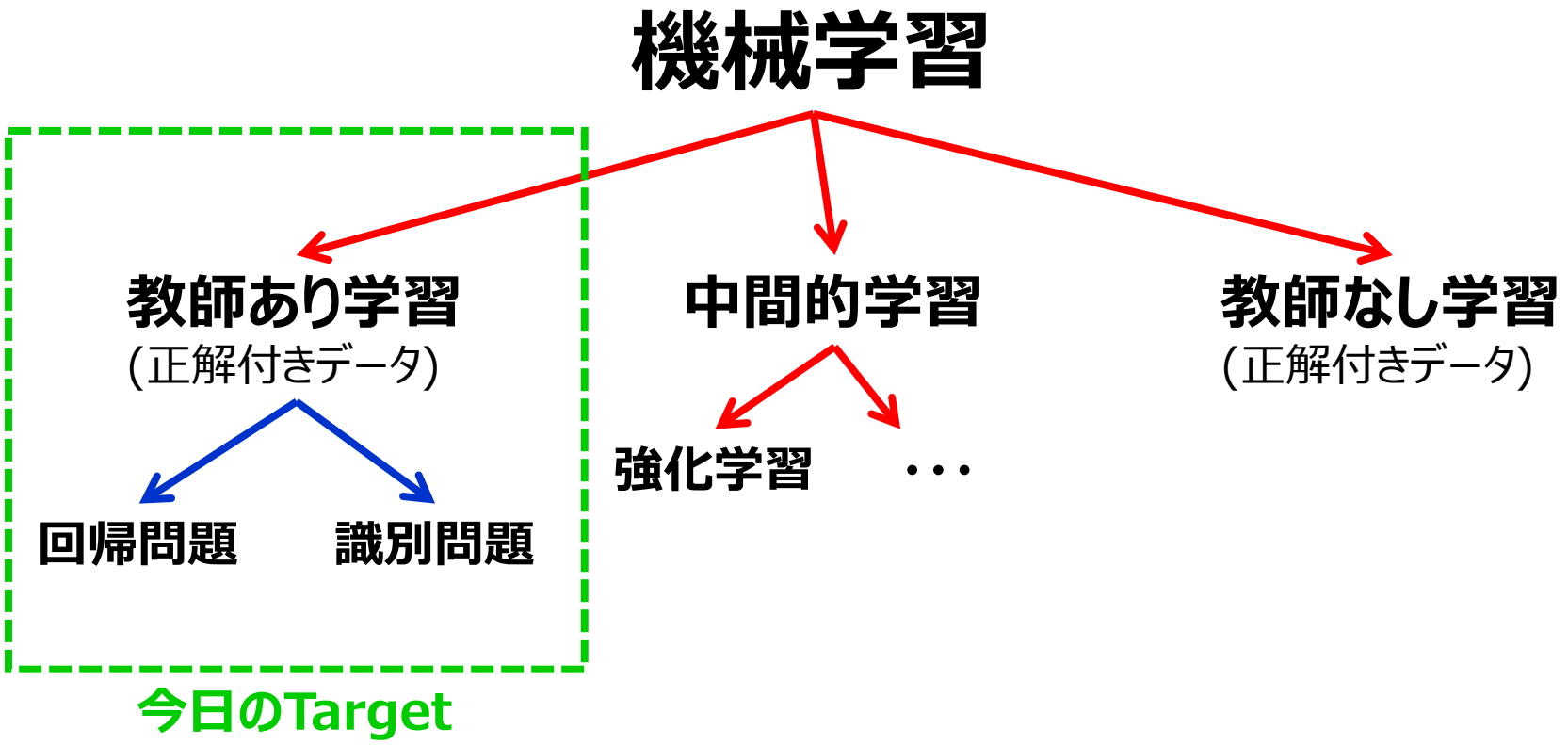
12. 機械学習 I

- Regression　回帰問題
- Overfitting　過剰適合(過学習)
- Classification　分類問題
- Example: Handwritten digit recognition
- Support vector machines (SVMs)

機械学習の種類


機械学習
教師あり学習
(正解付きデータ)
回帰問題
識別問題
今日のTarget
中間的学習
強化学習
・・・
教師なし学習
(正解付きデータ)

回帰問題

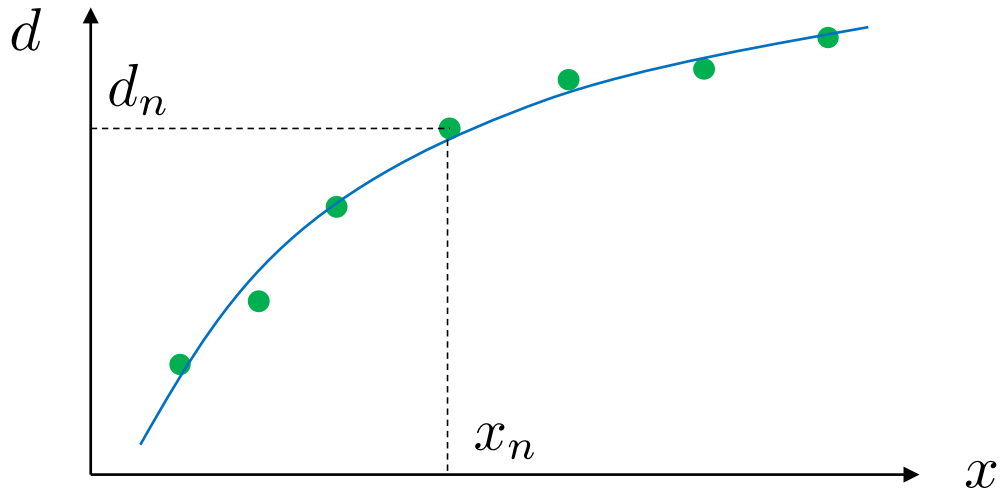
- N組のベクトル**x**とスカラーdの組が与えられたとする。

$$\{\mathbf{x}_n\}(n = 1, \ldots, N) \qquad \{d_n\}(n = 1, \ldots, N)$$

- 新しい入力**x**に対するdを予想したい。
  - **x** は独立変数と呼ばれ、対象を予測するために使われる。
  - *d* は従属変数と呼ばれ、予測対象である。
- 次のような近似を考える。

$$y(\mathbf{x}_n) \sim d_n$$

  - y(x)を記述するために任意の関数を利用可能とする。



多項式近似

- n階の多項式による近似を考える。(線形近似の代わりに)

$$y = a_0 + a_1 x + a_2 x^2 + \cdots a_n x^n$$

$$\sum_{i=1}^{N} \| y_i - (a_0 + a_1 x_i + a_2 x_i^2 + \cdots a_n x_i^n) \|^2 \to \min$$

- `polyfit` performs this
  - 例； `pinv`(逆行列を求める機能)の代わりに次の形式で線形近似できる。

```
>> p=polyfit(x,y,1);
```

```
>> p=pinv(X)*y;
```

  - 例； 3階の多項式による近似

```
>> p=polyfit(x,y,3)
ans =
   -2.2455    3.8778   -1.3517    0.4603
```

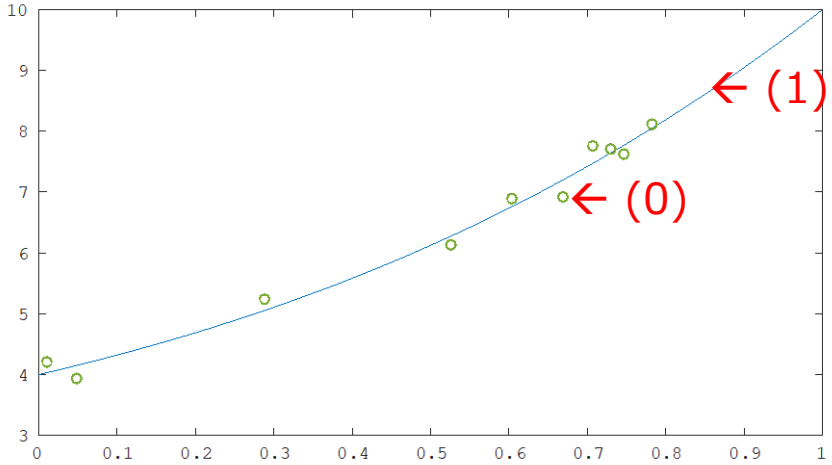
a_3 a_2 a_1 a_0

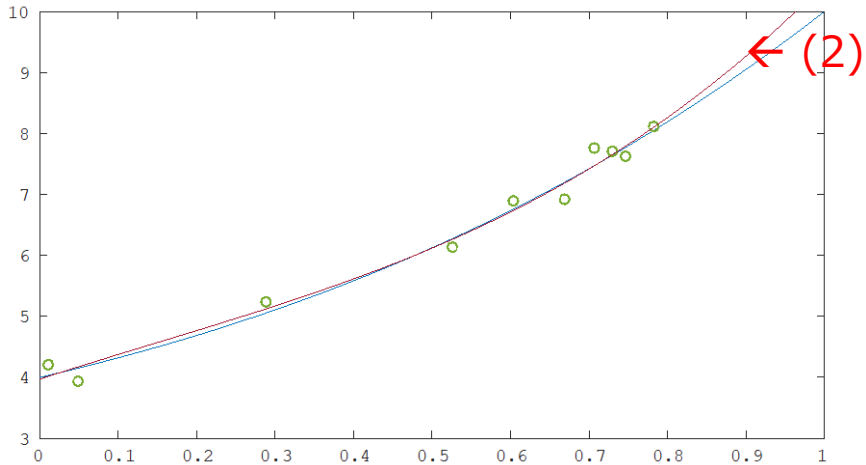
多項式近似の実例

```
>> x=rand(10,1);
>> p0=[1.0,2.0,3.0,4.0];
>> y=p0(1)*x.^3+p0(2)*x.^2+p0(3)*x+p0(4)+0.2*randn(10,1);
>> plot(x,y,'o') ← (0)
>>
>> hold on
>> xx=0:0.01:1;
>> yy=p0(1)*xx.^3+p0(2)*xx.^2+p0(3)*xx+p0(4);
>> plot(xx,yy) ← (1)
>>
>> p=polyfit(x,y,3);
>> plot(xx,polyval(p,xx)) ← (2)
```

フィッティングカーブからのバラつきを表現

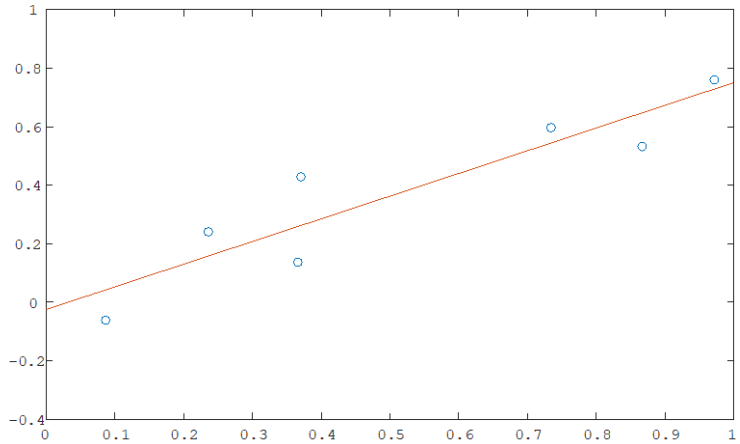
Data are synthesized here for the purpose of explanation



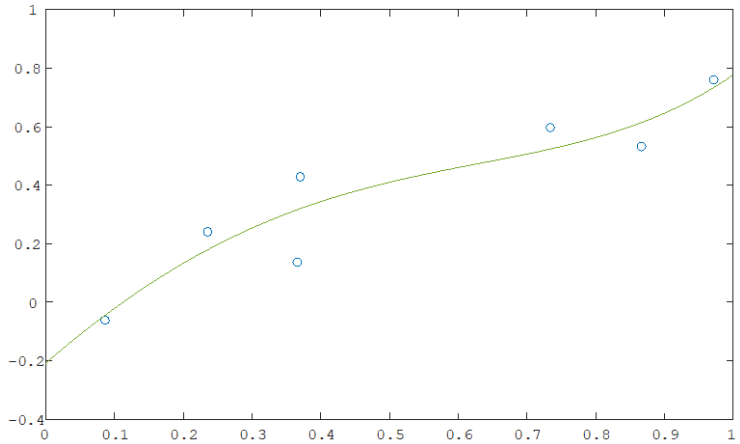


過剰適合 (過学習)

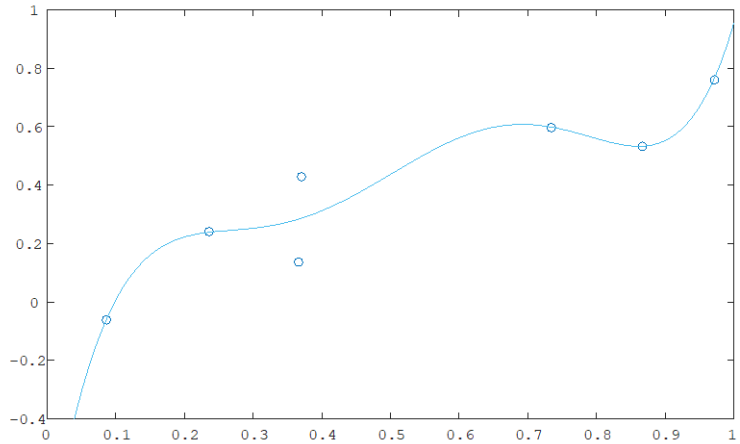
- 7点のデータに対して1階、3階、5階、6階それぞれで近似してみると…。
  - 過剰な自由度を有するモデルはノイズも含めてしまって(無意味に)表記してしまう。



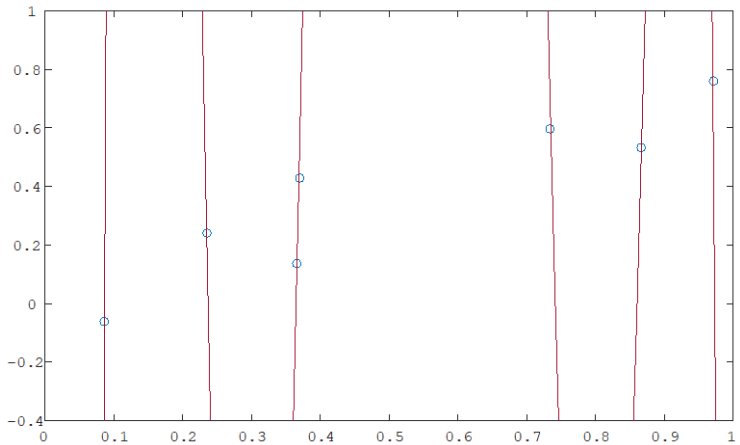
1st order (a line)



3rd-order



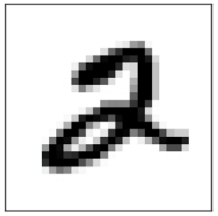
5th-order



6th-order

分類(識別)

- 変数**x** がk個に分けられたクラスの内、いずれかに属するとする。
- Classification(分類) ： 入力**x**を*K* 個のクラスのいずれかに割り当てること。
  - 例； **x** は1桁の数字を示す画像で、それがどの数字なのかを知りたい。



$\mathbf{x}$

⟹ 0,1,2,3,4,5,6,7,8,9

- *N*組の入力 **x** と1対1に対応する分類ラベル *d* が与えられたと仮定する。

$$\{\mathbf{x}_n\}(n = 1, \ldots, N) \qquad \{d_n\}(n = 1, \ldots, N)$$

新しい入力 **x** に対しての分類を推測したい。

例: 手書き文字(数字)の識別

- 手書き文字識別のデータセットしてよく知られている *MNIST* をこの演習では用いる。

  参考URL `http://yann.lecun.com/exdb/mnist/`

  - 講義のページから以下のファイルをダウンロードする。

    `mnist-data.zip`

  - 今日は以下のファイルを使用する。

    `t10k-images-idx3-ubyte` & `t10k-labels-idx1-ubyte`

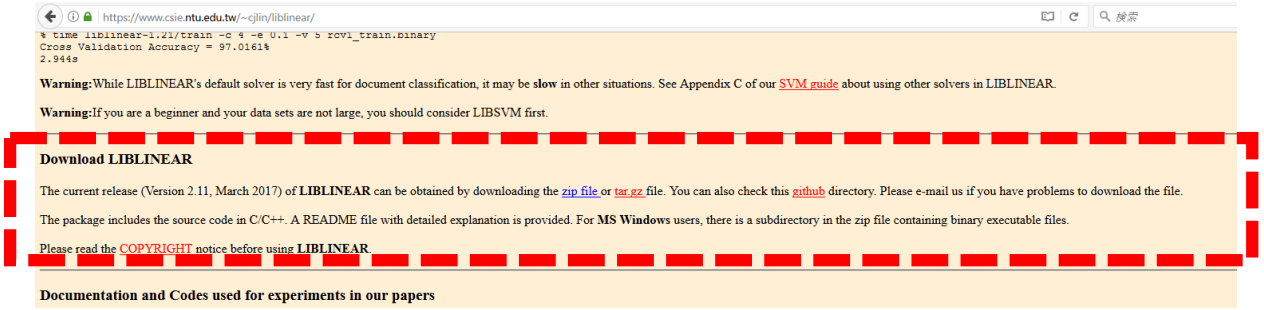
- 分類には*support vector machines* (*SVMs*) を用いる。
- 本演習では *liblinear* というソフトウェアライブラリを用いてSVMを実行し、文字識別を行う。

liblinear のインストール

- liblinear

  -National Taiwan University のグループによって作られたよく知られる機械学習のためのライブラリの一つ。

- 以下のURLからダウンロードする。
  - https://www.csie.ntu.edu.tw/~cjlin/liblinear



- ファイルを作業ディレクトリで解凍(展開)し、解凍されたフォルダ内の `liblinear-x.xx/matlab` に移動する。
  - `>> cd ./liblinear-2.30/matlab` (移動するためのコマンド)
- Run make.m
  - `>> make`
- Add the folder to search paths
  - `>> addpath('/Users/xxxx/Octave/liblinear-2.30/matlab')`

これは人によってことなります。
作業ディレクトリのパスを確認してください。

Support vector machines (SVMs) (1/2)*

- 次の2つの分類をかんがえる： $d_n = 1 \ \text{or} \ -1$
- 次のデータの組を与える： $(\boldsymbol{x}_1, d_1), (\boldsymbol{x}_2, d_2), \cdots, (\boldsymbol{x}_N, d_N)$
- 以下の方法で分類を行う。：

$$y(\mathbf{x}) = \begin{cases} 1 & \text{if } u(\mathbf{x}) > 0 \\ -1 & \text{otherwise} \end{cases}$$

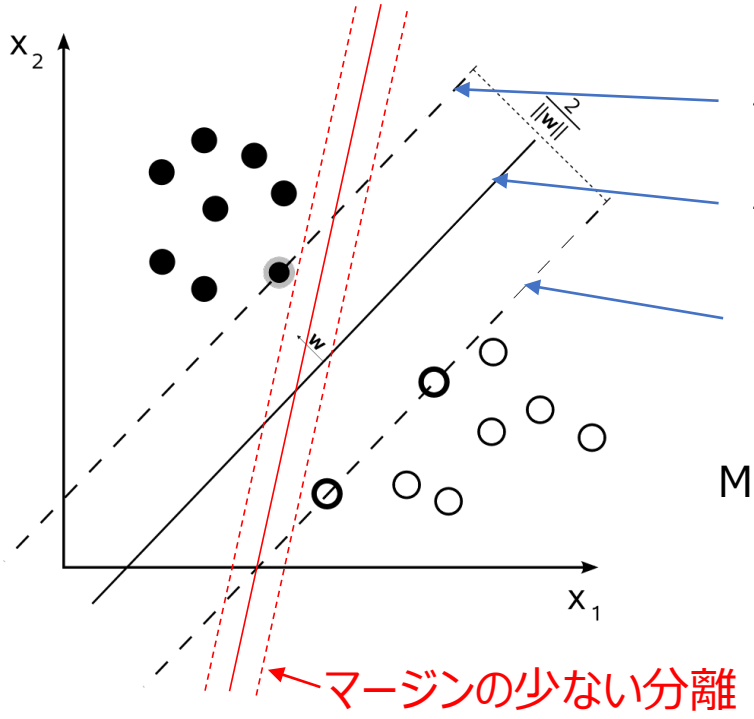
where $u(\mathbf{x}, \mathbf{w}) = w_0 + w_1 x_1 + \cdots + w_I x_I = w_0 + \mathbf{w}^\top \mathbf{x}$

- **w** は重み(係数)と呼ばれ、分類決定のためのパラメータ。
- 例えば次の条件でwを決定する。：
  - Known as a *hard-margin SVM*

Minimize $\|\mathbf{w}\| \quad \text{subject to} \quad d_n(w_0 + \mathbf{w}^\top \mathbf{x}) \geq 1$

Support vector machines (SVMs) (2/2)*

- データポイントを正確に２つのクラスに分離する２つの平行な平面を考える。分離可能な２つの平行な平面は色々考えられるが、平面間の距離が最大であるものを選ぶ。
  - F問題の単純化のために、データポイントを（曲面でなく）平面で分離することができるものとする。（線形分離可能と呼ぶ）
- **w**とw0で表すことができるこの２つの平行平面の間の平面を選択します。
  - なぜこのような手順を踏む? → 大きなマージンをもって分類するため。



⇕

$$w_0 + \mathbf{w}^\top \mathbf{x} = 1$$

$$w_0 + \mathbf{w}^\top \mathbf{x} = 0$$

$$w_0 + \mathbf{w}^\top \mathbf{x} = -1$$

&

Minimize $\|\mathbf{w}\|$ subject to $d_n(w_0 + \mathbf{w}^\top \mathbf{x}) \geq 1$

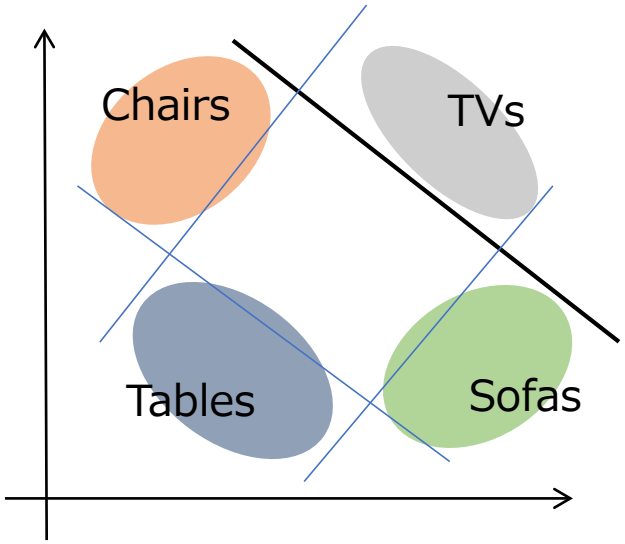
SVMが発表された当初は線形分類器にしか適用できなかったが、非線形分類問題に対しても特徴空間への写像により線形分類問題に落とし込むことができるようになり、機械学習の大きな発展につながった。

複数クラスへの分類

対他分類器（*one-versus-the-rest* classifier)

1. あるクラス*k*に対してモデル $y_k(x)$ が*k* に属するかそうでないかを学習させる。*(*クラス*k*についての分類器を学習させる。*)*
2. それぞれのモデルに対して各クラスの分類器を適用して、最も大きなスコアを出力した分類器のクラスに分類させる。

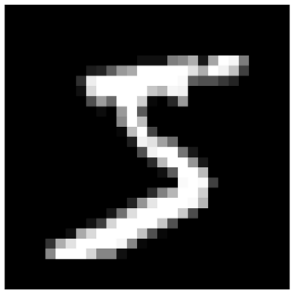
$$\underset{k}{\operatorname{argmax}}\, y_k(\mathbf{x})$$



MNIST ファイルの読み込み

- Octaveにイメージをロードさせる:
  - ‘test-images-idx3-ubyte’ には28x28 pixelsイメージデータが10,000個格納されている。
  - 最初の4個の整数(32bits)を飛ばして、残りの数字データを変数’`data`’に読み込む。
  - 適当なサイズのテンソルに整形して、’`imshow(matrix, [brightness_min, brightness_max])`’を用いて画像を表示する。

```
>> fid=fopen('t10k-images-idx3-ubyte','r','b');
>> fread(fid,4,'int32')
>> data=fread(fid,[28*28,10000],'uint8');
>> fclose(fid);
>> img=reshape(data,28,28,10000);
>> imshow(img(:,:,1)',[0,255])
>> imshow(img(:,:,100)',[0,255])
```



- ラベルを読み込む:
  - ‘test-labels-idx1-ubyte’ にはイメージのラベルが格納されている。
  - 最初の4個の整数(32bits)を飛ばして、残りの数字データを変数’`label`’に読み込む。

```
>> fid=fopen('t10k-labels-idx1-ubyte','r','b');
>> fread(fid,2,'int32')
>> label=fread(fid,10000,'uint8');
```

Check the contents of this variable

識別の学習と評価

- 5,000個の画像サンプルを用いて学習させる。
  - Train a model (SVM) using samples with indices 1,…,5000:

```
>> tr_label = label(1:5000);
>> tr_data = data(:,1:5000);
>> model = train(tr_label,sparse(tr_data)');
...
Objective value = -0.081903
nSV = 910
```

Status of training, which you can ignore (as long as the training went well)

- 残りの画像サンプルを用いて学習結果を評価する。
  - Test the model using samples with indices 5001,…,6000:

```
>> te_label = label(5001:6000);
>> te_data = data(:,5001:6000);
>> pred_label = predict(te_label,sparse(te_data)',model)
Accuracy = 84.6% (846/1000)
pred_label =
    2
    3
...
```

Classification accuracy for the input 1,000 samples is shown

Predicted labels for the 1,000 samples; note that the numbers do not correspond to the true digits; these numbers correspond to the indices of model.Label, which stores the true labels of digits

重み係数の可視化

- `predict` performs the following computation

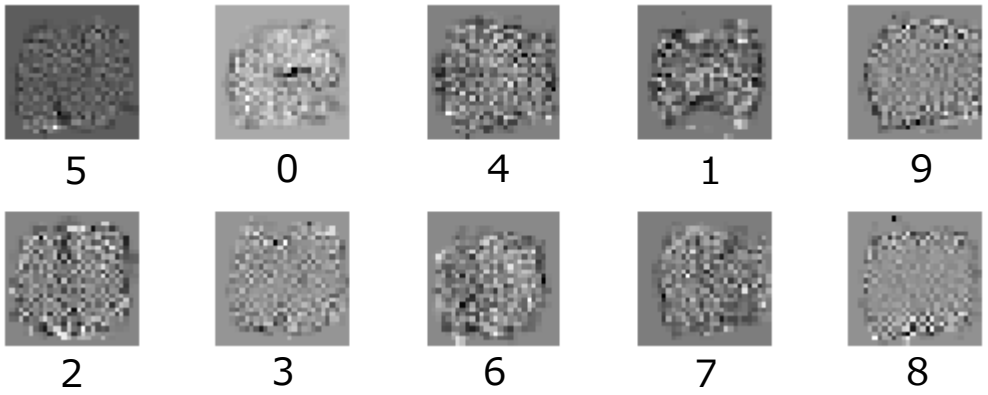
```
>> for i=1:10,model.w(i,:)*reshape(te_data(:,4),28*28,1)+model.bias,end
ans = -5.3081
...
...
ans = -17.245
ans =  2.5717
...
```

```
>> te_label(4)
ans =  6
>> model.Label
ans =

   5
   0
   4
   1
   9
   2
   3
   6
   7
   8
```

- Visualize the trained weights as images
  - それぞれのイメージがどこでそれぞれの数字を分類しているか説明できますか？

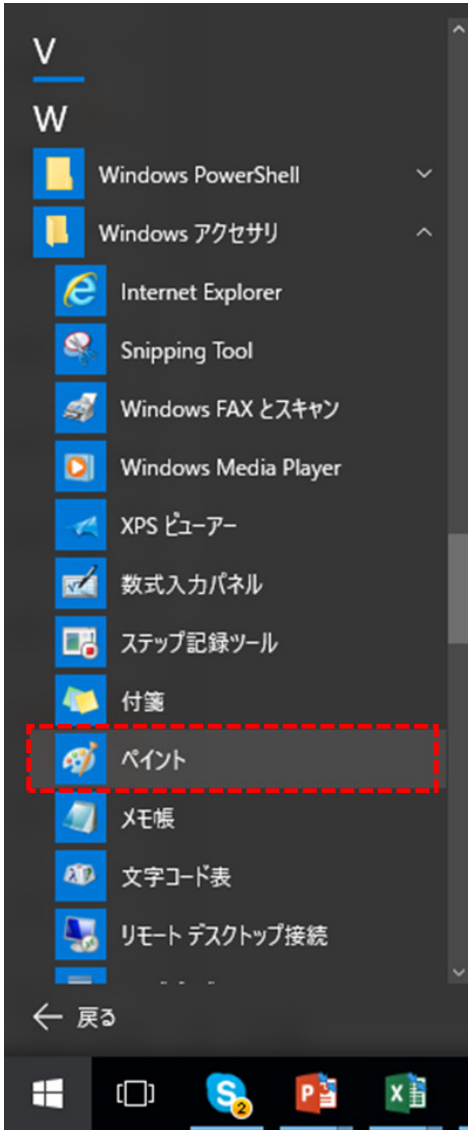
```
>> figure
>> for i=1:10,subplot(2,5,i),imshow(reshape(model.w(i,:),28,28),[min(model.w(i,:)),max(model.w(i,:))]),end
```

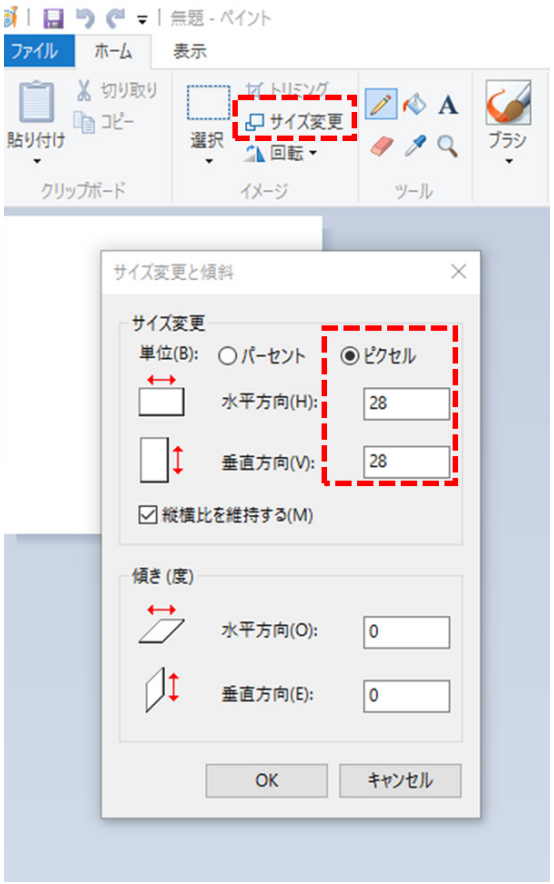


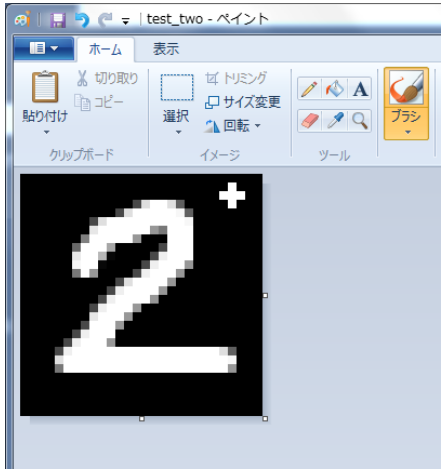
The order of weights is specified by `model.Label`

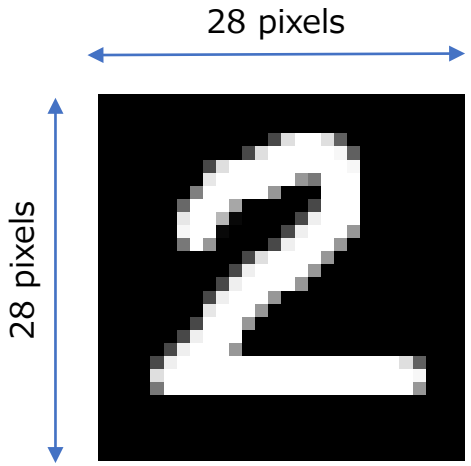
Exercise 12.1 (Make the model recognize your handwritten digit)

- 28x28 pixelsの画像を作成し、好きな数字を書いて保存してください。









Black background and white foreground

Exercise 12.1 (Make the model recognize your handwritten digit)

- 先ほどの評価方法と同様に画像を読み込んで評価してください。
  - もし、うまくいかなければ、数字を書くところから再度試してみてください。

```
>> sample = imread('a_number_I_wrote.png');
>> sample = mean(sample,3);   Convert your image into grayscale if it is a color image
                 True label
>> predict([2], sparse(reshape(sample',1,28*28)), model)
Accuracy = 100% (1/1)
ans =  2
          Predicted label; this is correct!
```

提出すべき物
1)自身で作成した手書きの数字の画像
2)実行結果が分かる物(上記のようなコマンドのコピペ、実行結果のプリントスクリーン、これまでと同様のスクリプトファイルなど)
3)もし、学習過程で工夫があれば、学習過程のスクリプトファイルを添付してください。(加点要素)

提出先：
東北大学ｲﾝﾀｰﾈｯﾄｽｸｰﾙ(ISTU)上で提出
もしくは
Email: hisashi.kino.a1＠tohoku.ac.jp
shimada@m.tohoku.ac.jp
〆切：
2019年8月2日(金)の午前8:50