機械学習特論

~理論とアルゴリズム~

(Introduction to Machine Learning and Octave)

講師：西郷浩人
saigo@inf.kyushu-u.ac.jp

授業の進め方

- **受講者は講義情報 moodleサイトに必ず登録すること**

  https://moodle.s.kyushu-u.ac.jp

  後期・火4・機械学習特論（西郷　浩人）
- プログラミング言語Octaveを演習に用いる。使用経験は問わないが、プログラミングの課題を課す。
- **疑問があれば積極的に質問して講義に参加すること。**

参考書

- **Pattern Recognition and Machine Learning**
  - **C.Bishop, Springer**
- Pattern Classification
  - R.Duda, P.Hart, D.Stork, Wiley Interscience
- 統計的機械学習
  - 杉山将、オーム社
- Elements of Statistical Learning
  - T.Hastie, R.Tibshirani, J.Friedman, Springer
- All of Statistics
  - L.Wassermann, Springer

成績評価

- 演習課題（40%）
- プロジェクト（60%）

The goal of this lecture series

- Understanding of machine learning algorithms
- Reading and writing octave codes
- Reading and understanding texts in english
  - Assignments can be described in Japanese.

What is Machine Learning ?

- Also known as
  - Pattern recognition
  - Artificial intelligence
  - Statistical learning

Big players in AI/ML

PLATINUM

Microsoft

GOLD

Alibaba Group 阿里巴巴集团

Baidu 百度

facebook

VISA
Visa Research

SILVER

amazon

Booking.com

Disney Research

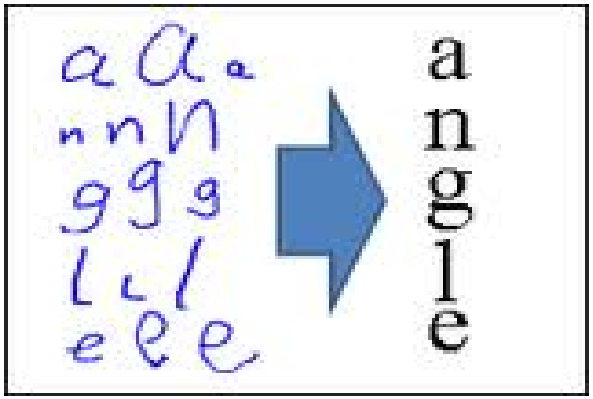
Google

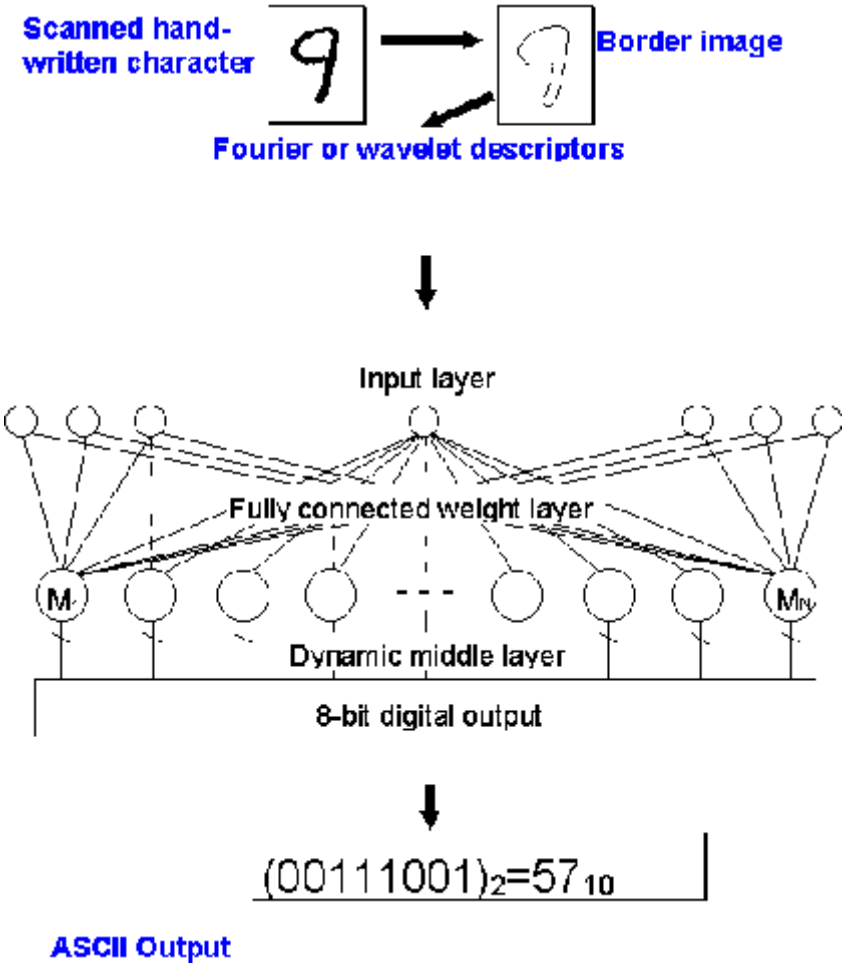
tableau

Applications

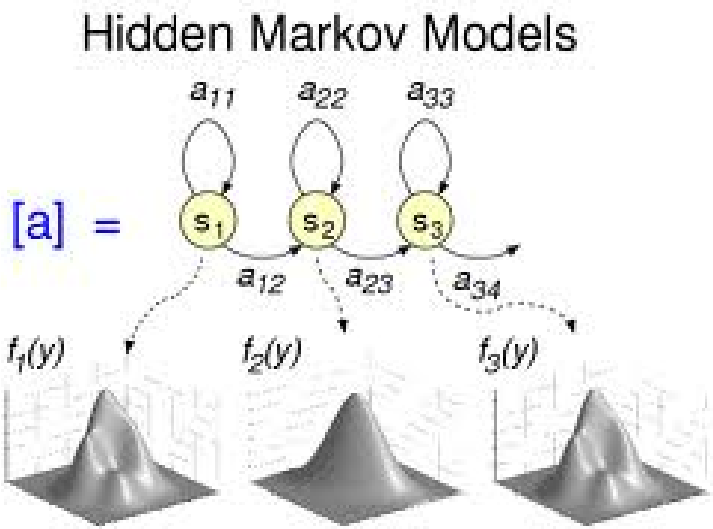
Hand written character recognition

•



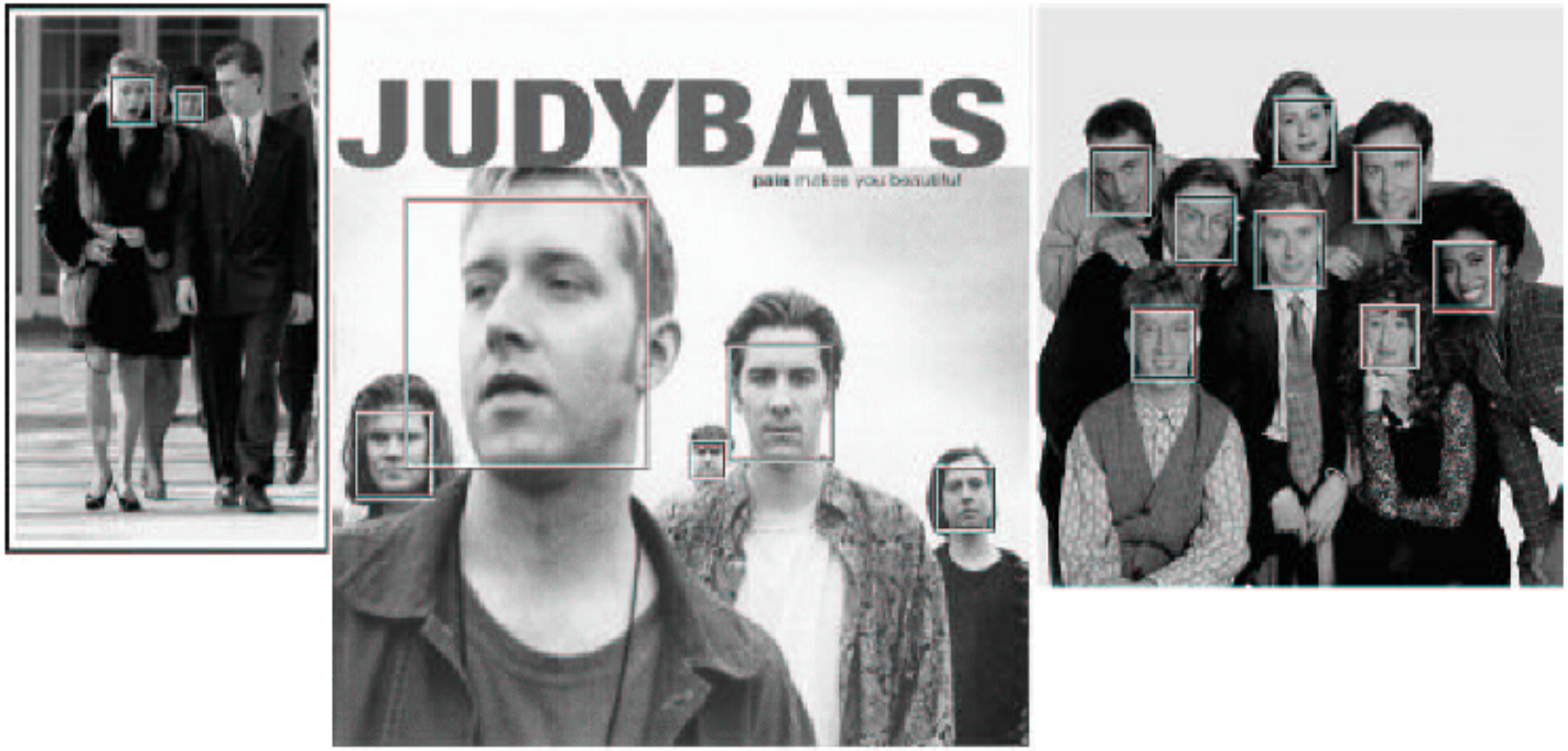


Speech recognition



Face recognition

•



bioinformatics

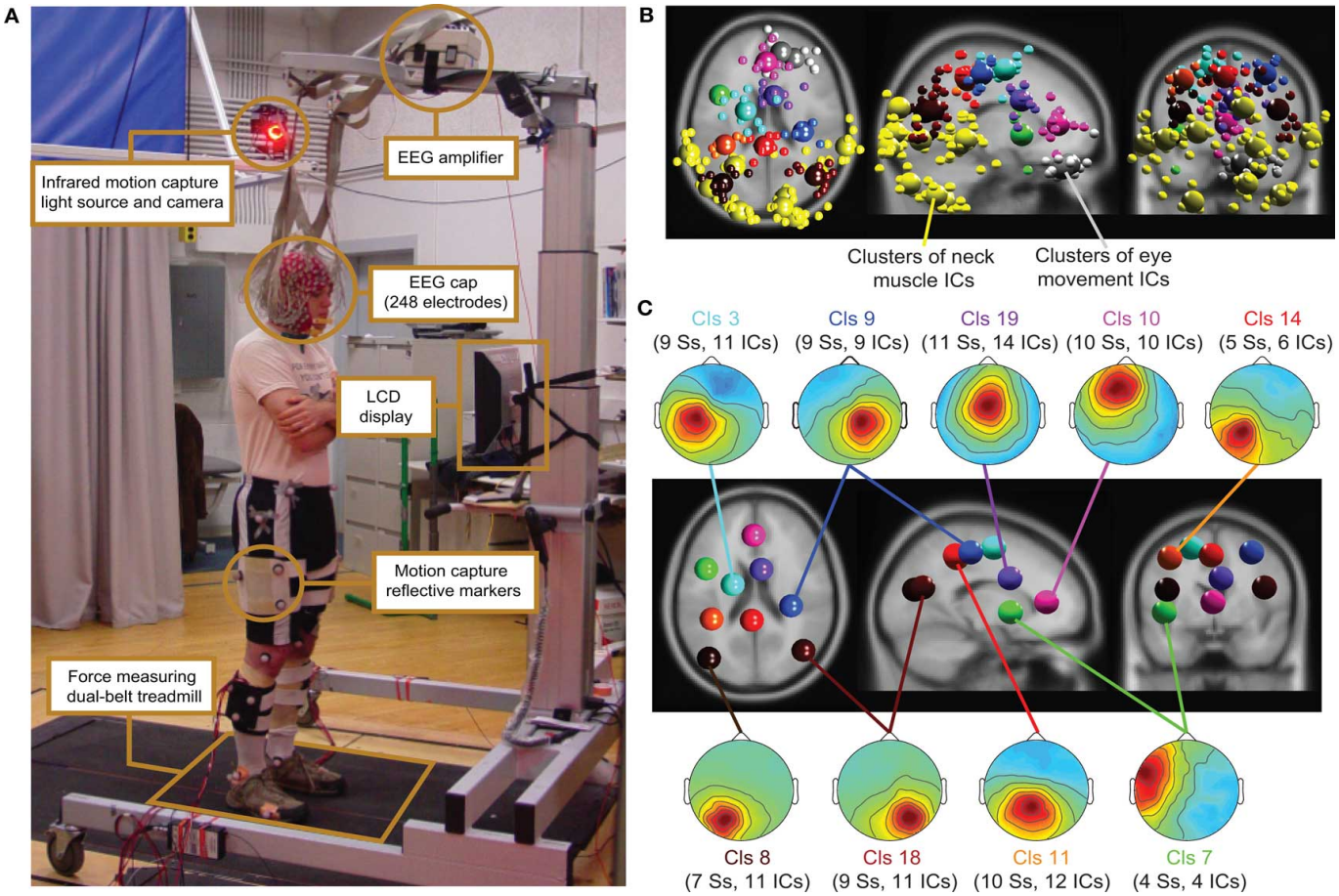
-

Decision making

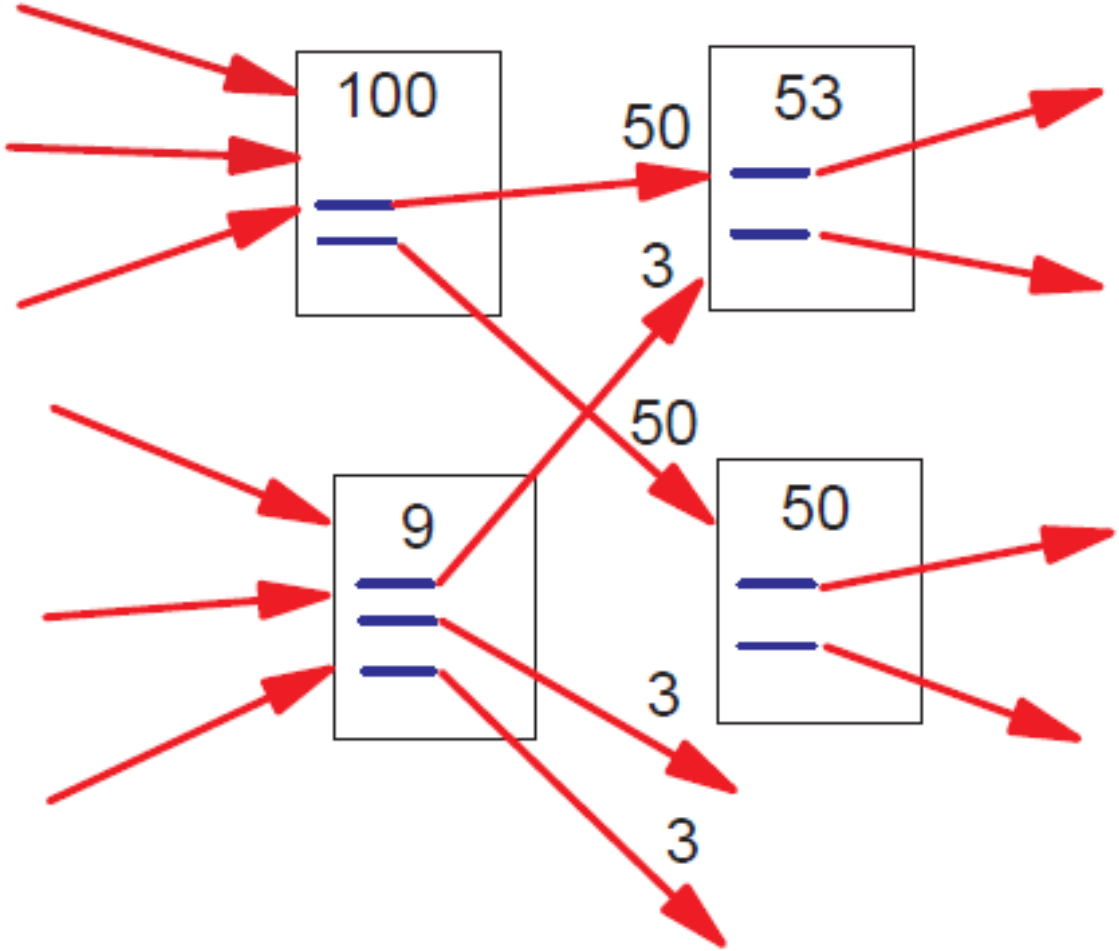
•

Brain computer interface

•

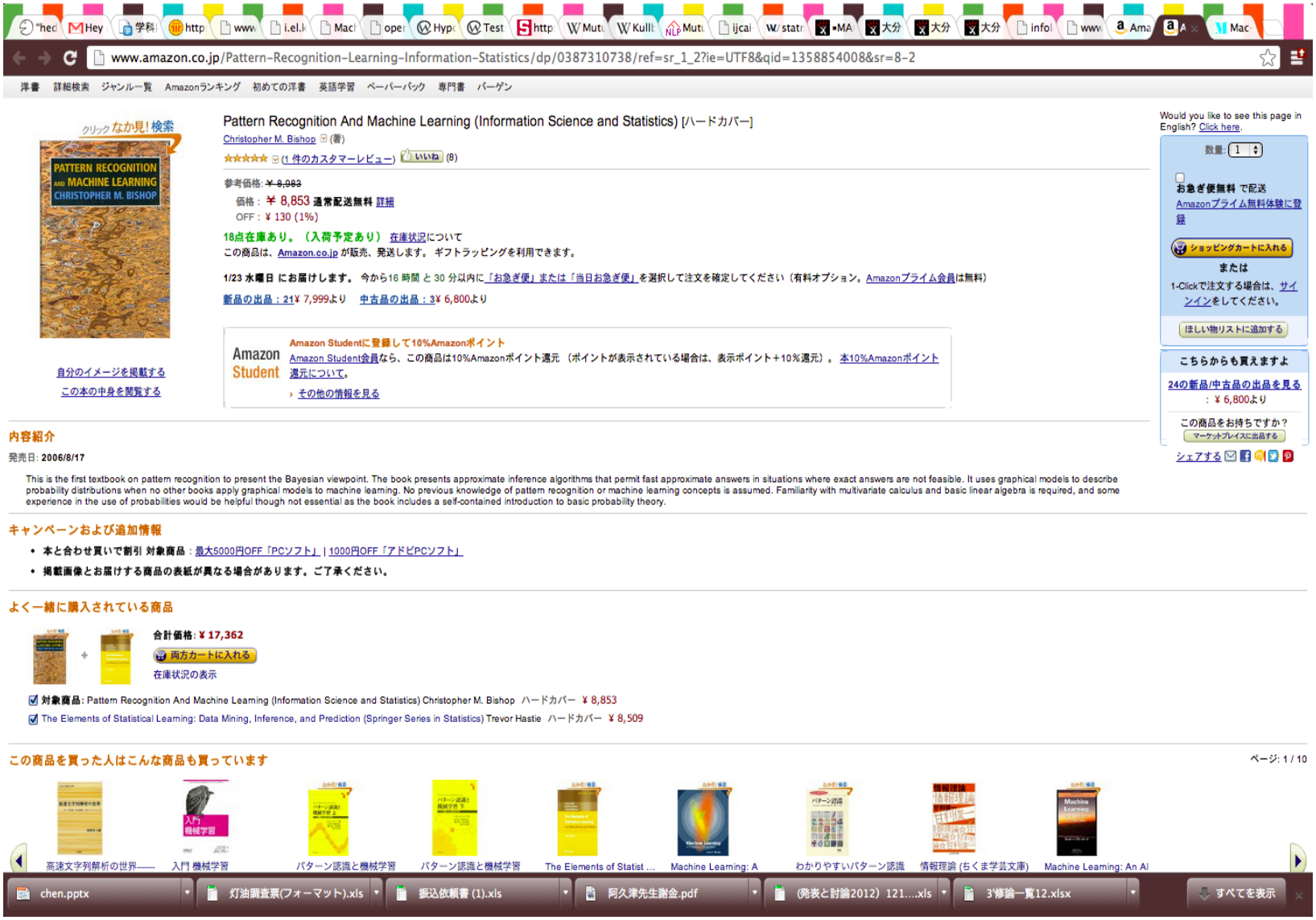


Google PageRank



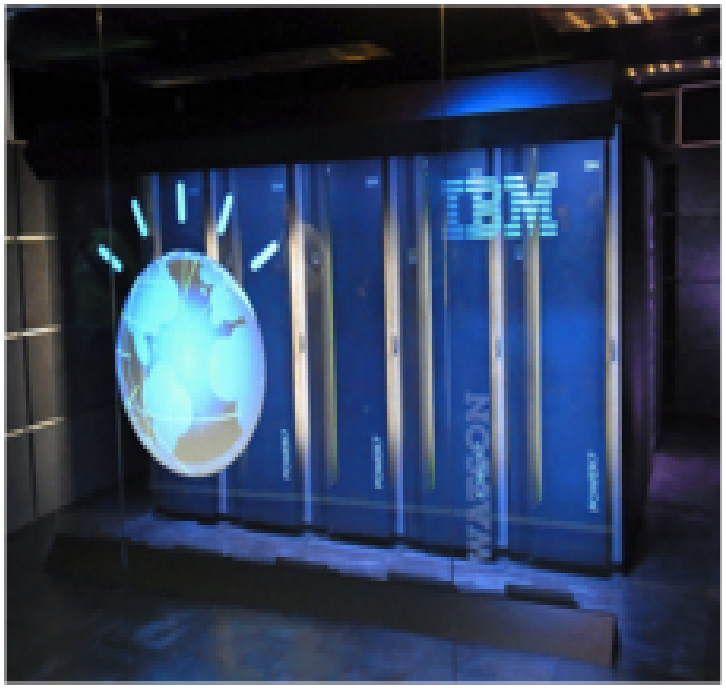
$$r(u) = c \sum_{v \in b(u)} r(v) / |f_v|$$

Web technologies



IBM's computer; Watson wins against quiz champion.

•



ニュース

2016年8月6日　ニュース・解説

東大の人工知能「ワトソン」、１０分で遺伝子解析…白血病患者を救う

ツイート　いいね！ 65　G+1 1　B! 3　チェック

東京大学医科学研究所が導入した人工知能（ＡＩ）が、白血病患者の特殊なタイプの遺伝子を１０分で見つけ、治療に役立てていたことが分かった。

AlphaGO's victory against professional go player(s).

How do we realize them ?

- Computer cannot “think” like human beings.
- Let see how it is realized in face recognition example.

Representation of a face in black and white

- An image with 50 x 50 pixels. Color density is represented by integers in range from 0 to 255
- 50x50x256=640000 possible different images. It is impractical to remember all the patterns.

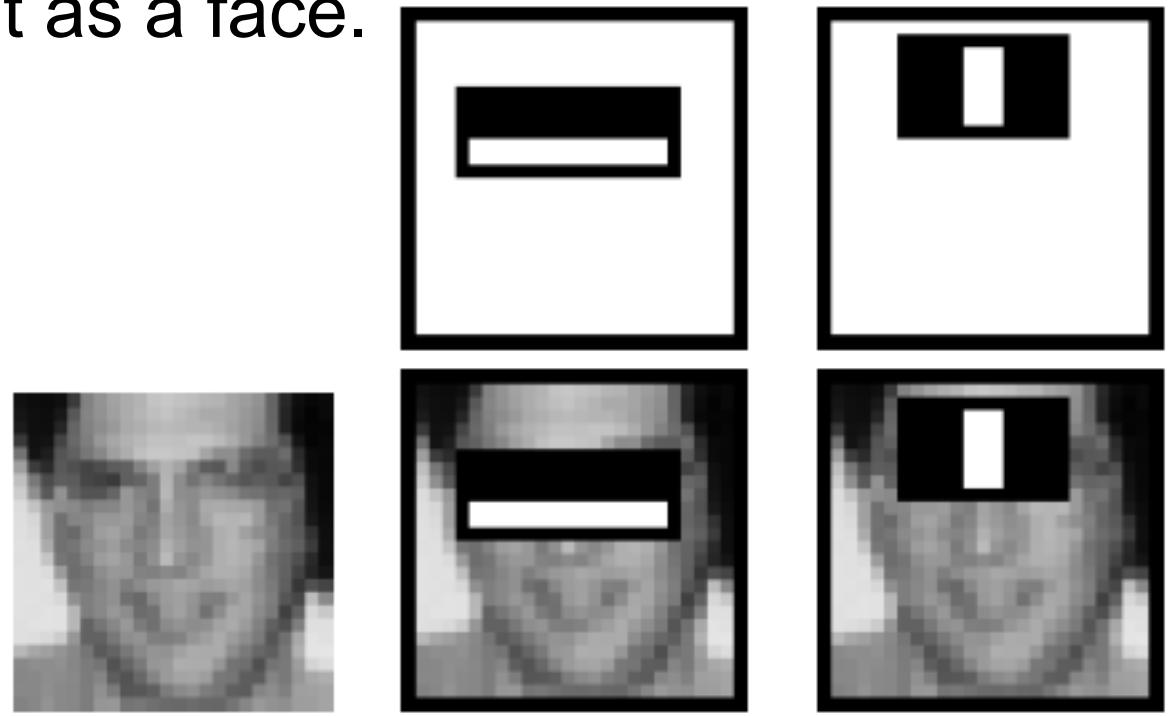
```
> M=imread('face.jpg');
> size(M)
ans =

   50   50
> M(1:10,1:10)
ans =

  254  255  252  251  255  252  255  255  255  254
  255  253  251  255  253  252  254  253  255  251
  253  255  255  252  255  255  253  254  255  255
  249  255  255  253  255  246  255  255  255  255
  255  255  254  255  253  255  255  255  255  254
  255  254  255  255  251    0    0    0    2    0
  253  251  254  255  253    1    0    2    0    1
```

Extracting patterns

- Sliding a rectangle window from left to right, top to bottom. Then add points if the window matches facial patterns such as eyes or mouth.
- If the final score obtained is above some threshold, then recognize it as a face.



© Viola and Jones

Learning rules

- In the previous example, it is unrealistic to collect more than 640k data for learning.
- We instead consider extracting rules effective for recognizing faces (such as eyes, nose etc.)
- The data we have can be represented in the following way.

$$\{\{\boldsymbol{x}_{\mathbf{1},} y_1\}, \{\boldsymbol{x}_{\mathbf{2},} y_2\} \ldots, \{\boldsymbol{x}_{\boldsymbol{n}}, y_n\}\}$$

  - Where X is a length 640k vector, y is a binary variable corresponding face(1) or not(0).
- X is called as explanatory variable or feature vector. Y is called as target response variable or class.

Representation of vector, matrix etc.

$$\{\{\boldsymbol{x}_{\mathbf{1},} y_1\}, \{\boldsymbol{x}_{\mathbf{2},} y_2\} \ldots, \{\boldsymbol{x}_{\boldsymbol{n}}, y_n\}\}$$

- Italic font represents a scalar.

  $y \in \boldsymbol{R}^{\mathbf{1}}$

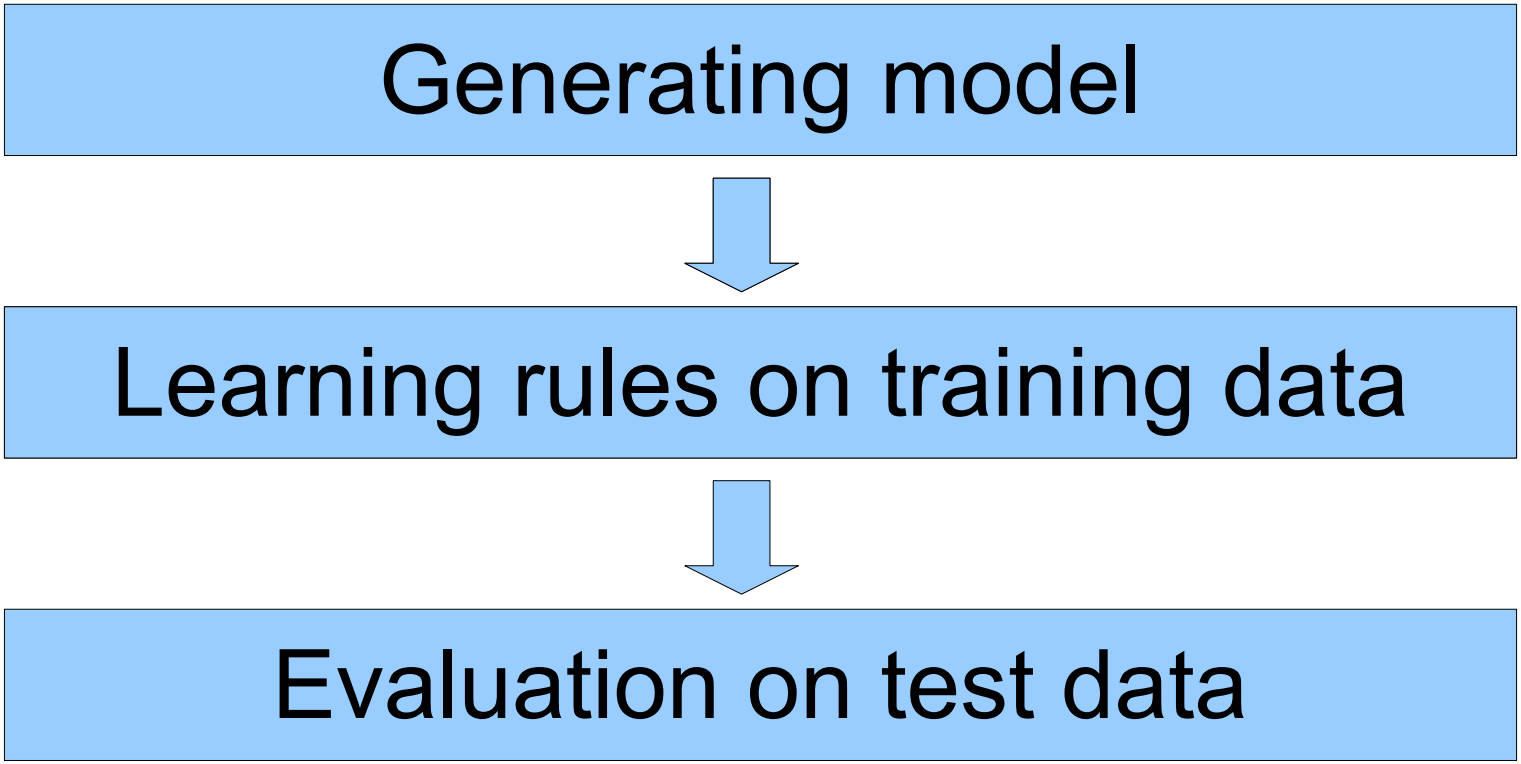
- Bold font represents a vector. (p by 1)

  $\boldsymbol{x} \in \boldsymbol{R}^{p \times \mathbf{1}}$

- Bold capitalized font represents a matrix.
  - ' on the shoulder means transpose.

$$\boldsymbol{X} = \begin{pmatrix} \boldsymbol{x}_{\mathbf{1}}^{t} \\ \boldsymbol{x}_{\mathbf{2}}^{t} \\ \ldots \\ \boldsymbol{x}_{\boldsymbol{n}}^{t} \end{pmatrix} \qquad \boldsymbol{X} \in \boldsymbol{R}^{n \times p}$$

A (simple) process of learning rules



Categories of machine learning

- 教師あり学習(Supervised Learning)
  - 分類（Classification）
  - 回帰（Regression）
  - 強化学習（Reinforcement learning）
  - Ranking
- 教師なし学習(Unsupervised Learning)
  - Clustering
  - Outlier detection

Keywords

-
  - Neural network, support vector machines (SVM), Maximum Likelihood, Model selection, information criteria, generalization performance, big data, Bayesian, posterior probability, etc..

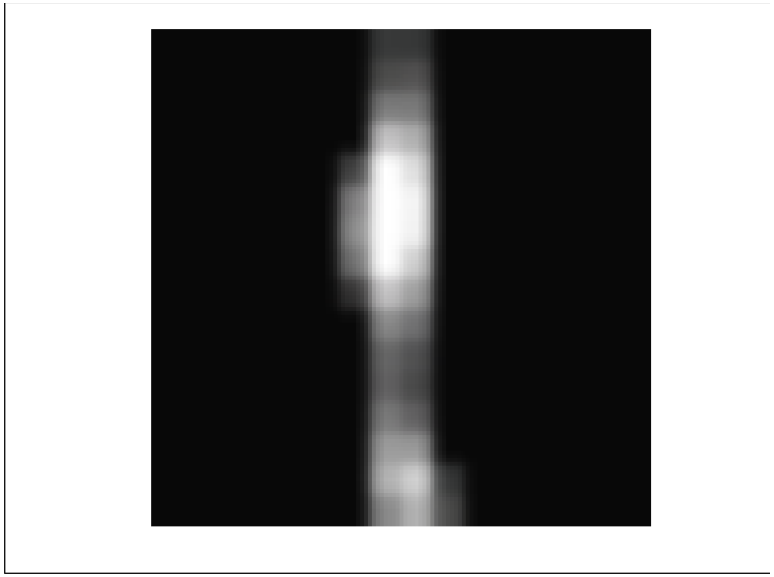
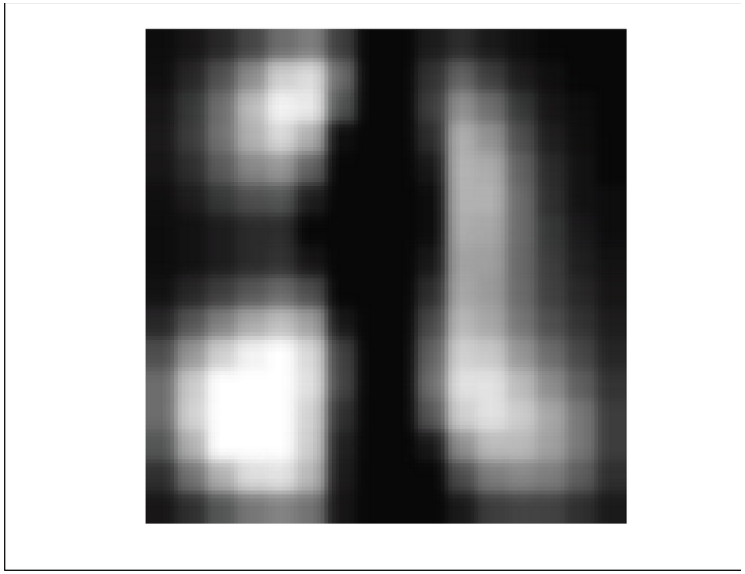
Lecture plan

- Basic probabilities and Baye's theorem
- Estimating parameters of normal distribution via maximum likelihood (ML)
- Classification of hand-written characters by linear discriminant analysis (LDA)
- Least squares and ridge regression
- Regularization and model selection
- Logistic regression by gradient descent and newton's method
- Feature selection

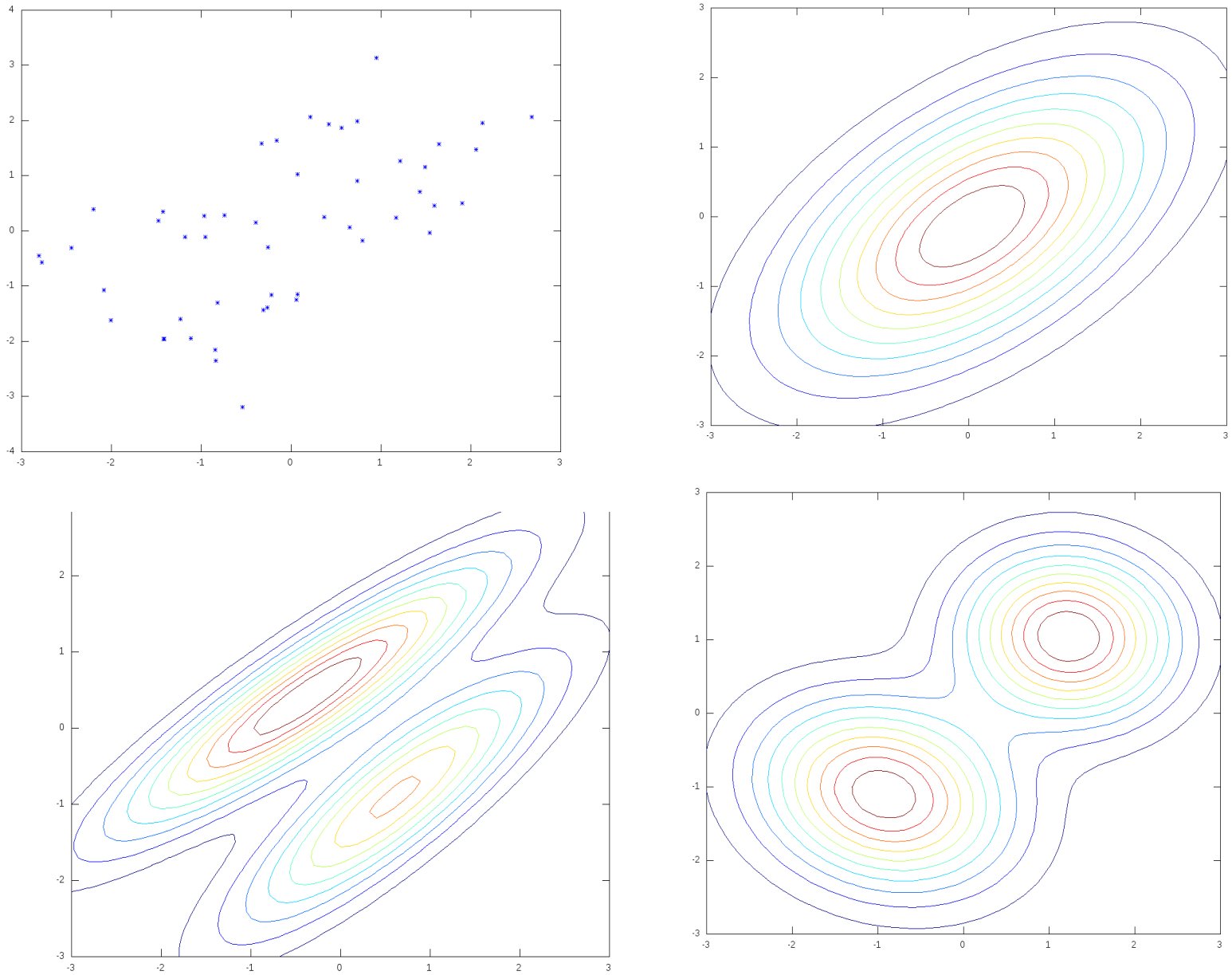
Lecture plan contd.

- Clustering by k-means and expectation maximization (EM)
- Dimension reduction techniques: PCA, PLS, CCA
- Support vector machines (SVM) and kernels
- Recommendation systems
- Ranking ?
- Structured output ?
- Deep learning ?
- Sparse learning ?

Classification example

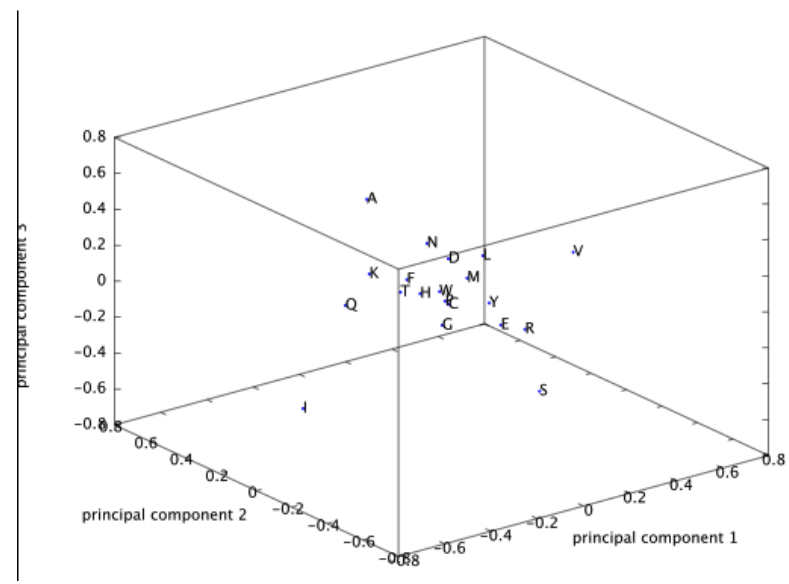
-  

Clustering example



Dimensionality reduction example

	A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V
A	4	-1	-2	-2	0	-1	-1	0	-2	-1	-1	-1	-1	-2	-1	1	0	-3	-2	0
R	-1	5	0	-2	-3	1	0	-2	0	-3	-2	2	-1	-3	-2	-1	-1	-3	-2	-3
N	-2	0	6	1	-3	0	0	0	1	-3	-3	0	-2	-3	-2	1	0	-4	-2	-3
D	-2	-2	1	6	-3	0	2	-1	-1	-3	-4	-1	-3	-3	-1	0	-1	-4	-3	-3
C	0	-3	-3	-3	9	-3	-4	-3	-3	-1	-1	-3	-1	-2	-3	-1	-1	-2	-2	-1
Q	-1	1	0	0	-3	5	2	-2	0	-3	-2	1	0	-3	-1	0	-1	-2	-1	-2
E	-1	0	0	2	-4	2	5	-2	0	-3	-3	1	-2	-3	-1	0	-1	-3	-2	-2
G	0	-2	0	-1	-3	-2	-2	6	-2	-4	-4	-2	-3	-3	-2	0	-2	-2	-3	-3
H	-2	0	1	-1	-3	0	0	-2	8	-3	-3	-1	-2	-1	-2	-1	-2	-2	2	-3
I	-1	-3	-3	-3	-1	-3	-3	-4	-3	4	2	-3	1	0	-3	-2	-1	-3	-1	3
L	-1	-2	-3	-4	-1	-2	-3	-4	-3	2	4	-2	2	0	-3	-2	-1	-2	-1	1
K	-1	2	0	-1	-3	1	1	-2	-1	-3	-2	5	-1	-3	-1	0	-1	-3	-2	-2
M	-1	-1	-2	-3	-1	0	-2	-3	-2	1	2	-1	5	0	-2	-1	-1	-1	-1	1
F	-2	-3	-3	-3	-2	-3	-3	-3	-1	0	0	-3	0	6	-4	-2	-2	1	3	-1
P	-1	-2	-2	-1	-3	-1	-1	-2	-2	-3	-3	-1	-2	-4	7	-1	-1	-4	-3	-2
S	1	-1	1	0	-1	0	0	0	-1	-2	-2	0	-1	-2	-1	4	1	-3	-2	-2
T	0	-1	0	-1	-1	-1	-1	-2	-2	-1	-1	-1	-1	-2	-1	1	5	-2	-2	0
W	-3	-3	-4	-4	-2	-2	-3	-2	-2	-3	-2	-3	-1	1	-4	-3	-2	11	2	-3
Y	-2	-2	-2	-3	-2	-1	-2	-3	2	-1	-1	-2	-1	3	-3	-2	-2	2	7	-1
V	0	-3	-3	-3	-1	-2	-2	-3	-3	3	1	-2	1	-1	-2	-2	0	-3	-1	4



octaveをインストールするには

- Linux
  http://www.gnu.org/software/octave/download.html
- Mac, Windows
  - http://octave.sourceforge.net/
- Mac版のインストールにはhomebrewの使用がオススメ
  - https://brew.sh/
  - 難しい場合はパッケージ版でも問題なし

Octave入門

OctaveはMatlab のクローン

- Matlab は、
  - 最適化やデータ解析の分野で標準的に使われている数値計算、プログラミング環境。
  - Mathworks社製
  - 高価
- Octave は、
  - Matlab のコマンドがほとんどそのまま使える Matlab のクローン。（但し、グラフィックス環境の互換性は高くない）
  - GNU project
  - 無料

プログラミング言語としてのOctave

- スクリプト言語、コンパイルは要らない
- 変数を定義する必要はない。必要に応じて初期化する必要はあり。
- 標準の変数はdouble型の配列
- 端末講義室ではWindowsとLinuxの両方で利用可能。

演習　Octave

- 四則演算

```
(1*2) + 5^2
```

- 算術関数

```
exp(10)*sin(pi/2)*log(100)
```

- ベクトル

```
v = [ 1 2 3 ]
```

- 長さ10の0ベクトル、1ベクトル

```
v0 = zeros(10,1), v1 = ones(10,1)
```

- ベクトルの定数倍

```
c = 10; c*v
```

- ベクトルの要素の和

```
sum(v)
```

演習 Octave

- 行列、ベクトルの作成 （'（クォーテーション）で転置を表す）

```
A = [1 2; 3 4]
b = [5 6]'
```

$$A=\begin{pmatrix}1 & 2\\ 3 & 4\end{pmatrix} \quad b=\begin{pmatrix}5\\ 6\end{pmatrix}$$

  - スペースで行を区切る。セミコロンで列を区切る
  - '（クォーテーション）で行列やベクトルの転置を表す
  - 行の最後に何も付けない時は変数を表示、最後にセミコロンを付けることで変数を非表示

- 行列の要素にアクセス、代入

```
A(1,1), A(1,2), A(1,1)=0
```

$$A=\begin{pmatrix}0 & 2\\ 3 & 4\end{pmatrix}$$

- 行列のサイズを調べる

```
size(A)
```

- 行列の行、又は列の成分にまとめてアクセス

```
A(1,:), A(:,1)
```

演習　Octave

- 行列とベクトルの足し算、掛け算

```
A*b, A*A, A+A
```

- ベクトルの自動作成(:がfor loopに対応する)

```
c=1:5, d=1:2:9
```

$$c = 1\,2\,3\,4\,5, d = 1\,3\,5\,7\,9$$

- ベクトルの内積

```
c*d'
```

- ベクトルの外積

```
c'*d
```

- Ax = b を x について解く

```
x = A\b
```

演習　Octave

- 固有値E、固有ベクトルVの計算

```
[V E] = eig(A)
```

- 逆行列

```
inv(A), A^(-1)
```

- 行列の定数倍

```
c=10; A.*c, A./c
```

- 行列の累乗

```
A^2, A^3, A^4
```

- 行列の要素の累乗(. ドットに注意)

```
A.^2, A.^3, A.^4
```

- ここまでのコマンドの確認

```
history
```

演習 Octave

- ベクトルのソート

```
b=[10 -3 5 27 -9], [S I]=sort(b)
```

- 平均、分散、最大値

```
mean(b), var(b), max(b)
```

- 分からなかったらヘルプ

```
help sort
```

- 関数のプロット

```
f = @(x) exp(-x); ezplot(f, [0 100])
```

- 乱数の発生とプロット

```
x=rand(10,1); y=rand(10,1); plot(x,y,'x')
z=rand(10); imagesc(z)
```

- 画像の保存

```
print -dpng image.png
```

スクリプト言語としてのOctave

- 自分がタイプしたコマンドを忘れないように、一つのファイルに書いて保存しておくと便利
- ファイルの編集には emacs, vi 等のエディタを使う
- function と endfunction の間に次の内容を書く

test1.m

```
disp(“A=[1 2;3 4]”);
A=[1 2; 3 4]
disp(“A(1,1)=0”);
A(1,1)=0
```

- 保存したファイルを実行するにはoctaveのコマンドプロンプトより

```
test1
```

ディレクトリ、作業スペースの保存

- 現在のディレクトリを調べる。移動する。

```
pwd; cd ..;
```

- 現在の変数を表示

```
who
```

- 現在の変数（作業スペース）を保存

```
save octave0412.mat
```

- 終了

```
quit
```

- (Zドライブに移動後)前回保存した作業スペースをロード

```
load octave0412.mat
```

- **作業スペースや自作の関数は、毎回ネットワークドライブに移すこと。そのままログアウトすると初期化されます。**

関数の作り方(1)

- “edit mytfact.m”で、次のように階乗の計算をする関数を作ることが出来る。
  - 関数の内容はfunction と endfunctionの間に書くこと
    - サブルーチンsubfactは同じファイルに書いても、(同じディレクトリの)別のファイルに書いてもよい

myfact.m

```
function [y] = myfact (x)
y = subfact(x);
endfunction

function [y] = subfact(x)
if x==1
  y = 1;
else
  y = x*subfact(x-1)   % ← ; なし
end
 endfunction
```

関数の作り方(1)

- 例えば5の階乗を計算するには

```
> myfact(5)
y =  2
y =  6
y =  24
y =  120
ans =  120
```

セミコロンを行末に書かない文は計算結果が画面に出力されるので便利。

関数の作り方(2)

- フィボナッチの数列を計算するコード
- 制御文にはif, for の他にwhileやswitchもある。

fib.m

```
function [X] = fib (n)
X=zeros(n,1);
X(1)=1;
X(2)=2;

for k=3:n
    X(k) = X(k-1) + X(k-2);
end

endfunction
```

関数の作り方(2)

- 10番目までの数列を計算するにはfib(10)

```
fib(10)
```

- 入力を変えたり、出力を変数に代入することもできる。
  - 20番目までの数列を変数(ベクトル)varに代入

```
•var = fib(20)
```

デバッグ

- 行の最後に ;(セミコロン)をつけると、内容を標準出力に印刷しない。
  - A*A;
- 行の最後に何も書かないか、 ,(コンマ)をつけると、内容を標準出力に印刷する。(デバッグに使えます)
  - A*A
- プログラミングの怪しいところには “keyboard” と挿入することにより、プログラムの動作を止めてチェックすることが出来ます。プロンプトに戻るときは 'return'
- コメントアウトは “%”