Math operations are same as C++

Logic operations are same as C++

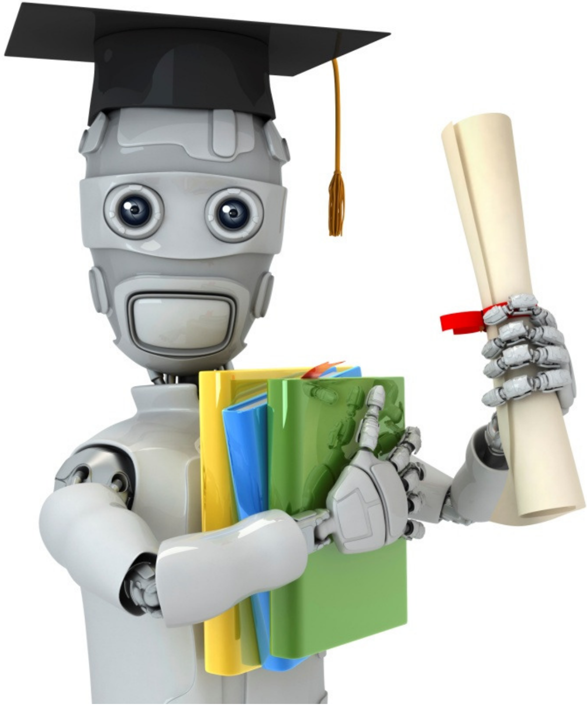
Semicolon supresses the print output

disp() for printing

sprintf() for string printing like printf in C

Matrices: A = [x1,x2,x3 ; y1,y2,y3] ==> 3*2 Matrix & semicolon for row separation

Vector: A = [x1;x2;x3] ==> 3d vector



Machine Learning

Octave Tutorial

Basic operations

v = 1:0.1:2 ==> starting from 1 & incrementing by 0.1 till reaching 2

v = 1:6 ==> starting from 1 to 6

ones(n,m) ==> generates matrix of ones(n:rows *m:cols)

also zeros(n,m)

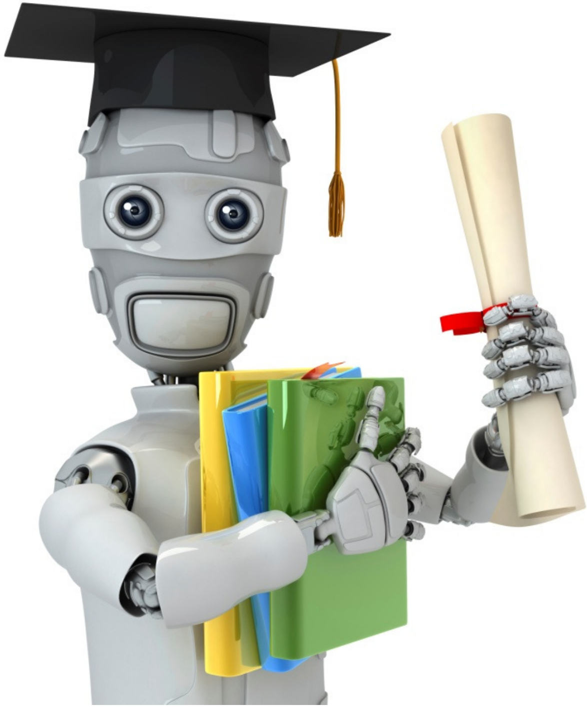
rand(n,m) ==> generates matrix of random no.s between 0 & 1

randn(n,m) ==> generates matrix of gaussian distribution(Normal) with u=0 & stdDeviation = 1

hist(var,no. of pins) ==> draws histogram of var

eye(n) ==> generates n*n identity matrix

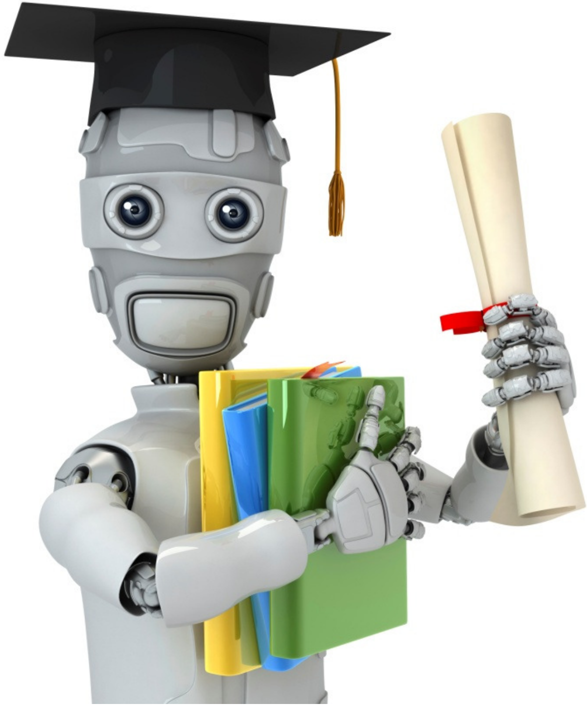
help command



Machine Learning

Octave Tutorial

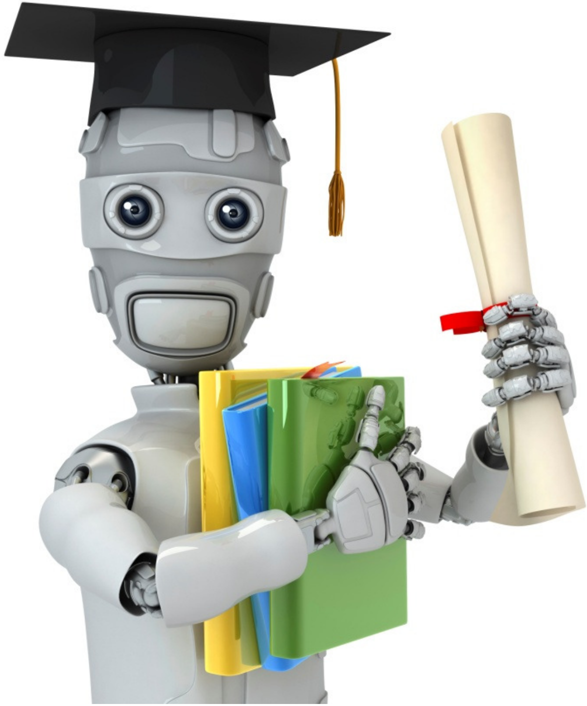
Moving data around



Machine Learning

Octave Tutorial

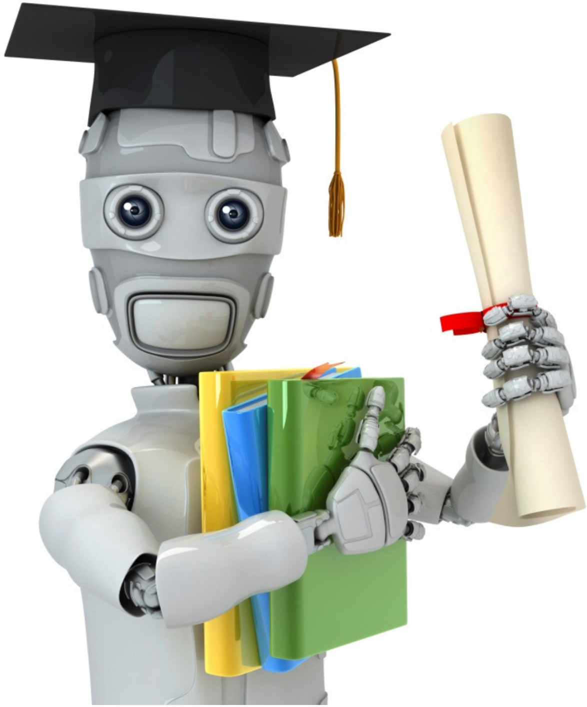
Computing on data



Machine Learning

Octave Tutorial

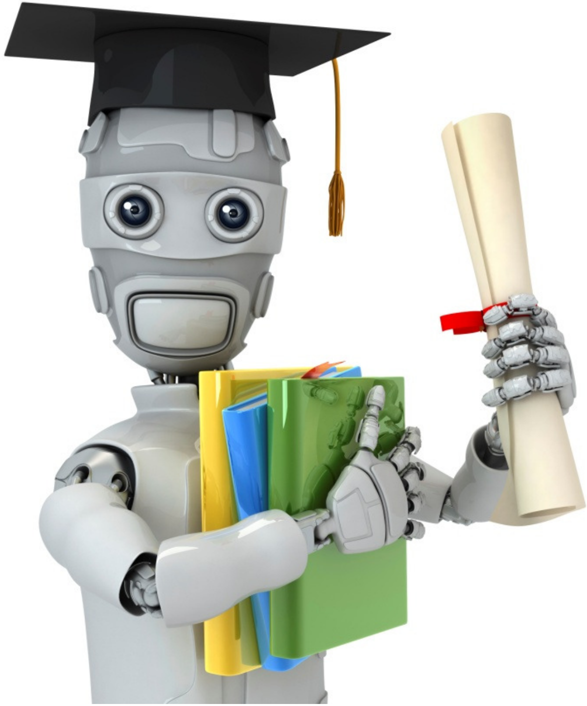
Plotting data



Machine Learning

Octave Tutorial

Control statements: for, while, if statements



Machine Learning

Octave Tutorial

Vectorial implementation

Vectorization example.

$$h_\theta(x) = \sum_{j=\theta}^{n} \theta_j x_j$$

$$= \theta^T x$$

Unvectorized implementation

```
prediction = 0.0;
for j = 1:n+1,
  prediction = prediction +
               theta(j) * x(y)
end;
```

Vectorized implementation

```
prediction = theta′ * x;
```

Vectorization example.

$$h_\theta(x) = \sum_{j=\theta}^{n} \theta_j x_j$$

$$= \theta^T x$$

Unvectorized implementation

```
double prediction = 0.0;
for (int j = 0; j < n; j++)
  prediction += theta[j] * x[y];
```

Vectorized implementation

```
double prediction
  = theta.transpose() * x;
```

Gradient descent

$$\theta_j := \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)}$$

(for all j)

$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_0^{(i)}$$

$$\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_1^{(i)}$$

$$\theta_2 := \theta_2 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_2^{(i)}$$

$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_0^{(i)}$$

$$\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_1^{(i)}$$

$$\theta_2 := \theta_2 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_2^{(i)}$$

$(n = 2)$

$u(j) = 2v(j) + 5w(j)$ (for all j)

$u = 2v + 5w$