# Machine Learning - Octave/Matlab Tutorial

赵燕

## 目录

**1 Octave/Matlab Tutorial** **2**
1.1 Basic Operations . . . . . 2
1.2 Moving Data Around . . . . . 6
1.3 Computing on Data . . . . . 10
1.4 Plotting Data . . . . . 10
1.5 Control Statements:for,while,if statement . . . . . 10
1.6 Vectorization . . . . . 10

# 1 Octave/Matlab Tutorial

## 1.1 Basic Operations

1.赋值语句:

```
>> a=3
a =  3
>> a=3;     #不会打印出a=3
>> a=3
a =  3
>> b='hi';
>> b
b = hi
>> c=(3>=1);
>> c
c = 1       #输出为真
```

2.打印和显示变量

```
>> a=pi;
>> a
a =  3.1416
>> disp(a);
 3.1416            #disp命令输出
>> disp(sprintf('2 decimals:%0.2f',a))
2 decimals:3.14    #打印字符串，保留两位小数
>> disp(sprintf('6 decimals:%0.6f',a))
6 decimals:3.141593  #sprintf是打印生成字符串
>> a
a =  3.1416
>> format long
>> a
a =  3.14159265358979
>> format short
>> a
a =  3.1416
>>
```

3.向量，矩阵和集合

```
>> A=[1 3;3,4;5,6]
A =         #矩阵

   1   3
   3   4
   5   6

>> v=[1,2,3]
v =       #行向量

   1   2   3

>> v=[1;2;3]
v =       #列向量

   1
   2
   3
```

```
>> v=1:0.1:2
v =      #集合，从1开始，增量（步长）为0.1，直到2

 Columns 1 through 4:

    1.0000    1.1000    1.2000    1.3000

 Columns 5 through 8:

    1.4000    1.5000    1.6000    1.7000

 Columns 9 through 11:

    1.8000    1.9000    2.0000
>> v=1:6
v =

   1   2   3   4   5   6

>>
```

4.生成矩阵的方法

```
>> ones(2,3)
ans =    #元素都为1矩阵

   1   1   1
   1   1   1

>> C=2*ones(2,3)
C =      #元素都为2的矩阵

   2   2   2
   2   2   2

>> w=ones(1,3)
w =      #1行3列

   1   1   1

>> w=zeros(1,3)
w =      #0矩阵

   0   0   0

>> w=rand(1,3)
#随机矩阵，元素随机，数值在0到1之间
w =

   0.056270   0.270442   0.232801

>> rand(3,3)
#随机矩阵，元素随机，数值在0到1之间
ans =

   0.42812   0.94129   0.32911
```

```
   0.37266   0.52775   0.89005
   0.43005   0.61385   0.76779

>> w=randn(1,3)
 #高斯随机矩阵（正态分布），元素随机，平均值为0的高斯分布
w =

  -1.11347   0.73961  -0.43813

>> w=randn(1,3)
#高斯随机矩阵（正态分布），元素随机，平均值为0的高斯分布
w =

  -0.20530   1.09960  -1.53719

>>

>> w=-6+sqrt(10)*(randn(1,10000))
w =Columns 1 through 3:

  -5.0452e+00  -3.6748e+00  -9.9375e+00

 Columns 4 through 6:

  -2.4220e+00  -9.0436e+00  -5.9153e+00

 Columns 7 through 9:

  -8.9856e+00  -7.3453e+00  -7.7757e+00

 Columns 10 through 12:

  -7.7120e+00  -4.2215e+00  -9.6187e+00

 Columns 13 through 15:

  -4.5269e+00  -3.2191e+00  -2.3526e+00

 Columns 16 through 18:

  -4.7875e+00  -6.7731e+00  -6.5302e+00

 Columns 19 through 21:

  -6.9177e+00  -5.0446e+00  -8.6510e+00

 Columns 22 through 24:

  -2.6468e+00  -4.2173e+00  -9.5689e+00
warning: broken pipe
>> hist(w)   #绘制直方图
>> hist(w,50)
>>>
```

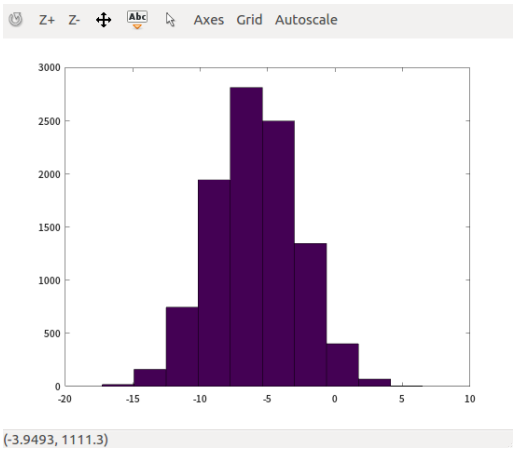


图 1: 集合w的直方图

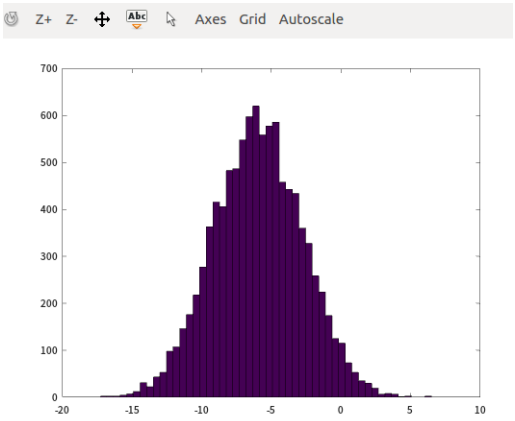


图 2: 集合w的直方图(50条)

5.单位阵

```
>> eye(4)
ans =

Diagonal Matrix

   1   0   0   0
   0   1   0   0
   0   0   1   0
   0   0   0   1

>> I=eye(4)
I =

Diagonal Matrix

   1   0   0   0
   0   1   0   0
   0   0   1   0
   0   0   0   1
```

```
>> I=eye(6)
I =

Diagonal Matrix

   1   0   0   0   0   0
   0   1   0   0   0   0
   0   0   1   0   0   0
   0   0   0   1   0   0
   0   0   0   0   1   0
   0   0   0   0   0   1

>> eye(3)
ans =

Diagonal Matrix

   1   0   0
   0   1   0
   0   0   1

>>
```

6.help命令

```
>> help eye
'eye' is a built-in function from the file libinterp/corefcn/data.cc

 -- eye (N)
 -- eye (M, N)
 -- eye ([M N])
 -- eye (..., CLASS)
     Return an identity matrix.

     If invoked with a single scalar argument N,return a square NxN
     identity matrix.

     If supplied two scalar arguments (M, N), 'eye' takes them to be the
     number of rows and columns.  If given a vector with two elements,
     'eye' uses the values of the elements as the number of rows and
     columns, respectively.  For example:

          eye (3)
           =>  1  0  0
               0  1  0
               0  0  1

     The following expressions all produce the same result:
     #q退出该命令
>> help rand
>> help help
```

## 1.2 Moving Data Around

1.size()命令，返回矩阵的尺寸

```
>> A=[1,2;3,4;5,6]
A =
```

```
   1   2
   3   4
   5   6

>> size(A)
#size()命令返回一个1*2的矩阵，返回矩阵的尺寸
ans =

   3   2

>> sz=size(A)
#这个矩阵用sz来存放，所以sz就是一个1*2的矩阵
sz =

   3   2

>> size(sz)
#计算矩阵的维度
ans =

   1   2

>> size(A,1)
#返回A矩阵的第一个元素3，行数
ans =  3
>> size(A,2)
#返回A矩阵的第2个元素2，列数
ans =  2
```

2.length命令

```
>> v=[1 2 3 4]
#向量v
v =

   1   2   3   4

>> length(v)
#返回最大维度的大小
ans =  4
>> length(A)
#矩阵A的最大维度是3
ans =  3
>> length([1;2;3;4;5])
#一般只是给向量用length命令
ans =  5
>>
```

3.在系统中加载和寻找数据

```
>> pwd
#显示出Octave当前所处路径
ans = /home/zhaozhao
>> cd #改变路径 'C:\Users\ang\Desktop
>> ls #列出所有的路径
courses-learning   Notes Octave
>> load features.dat  #加载了features文件
```

```
>> load priceY.dat
>> load ('featuresX.dat')
>> who
#显示出当前Octave所存储的变量
Variables in the current scope:

A    I    ans  c    v
C    a    b    sz   w

>> size(featuresX)
>> size(PriceY)
>> whos
#同时会列出维度
Variables in the current scope:

   Attr Name        Size                     Bytes  Class
   ==== ====        ====                     =====  =====
        A           3x2                         48  double
        C           2x3                         48  double
        I           6x6                         48  double
        a           1x18  double
        ans         1x14                        14  char
        b           1x22  char
        c           1x11  logical
        sz          1x2                         16  double
        v           1x4                         32  double
        w           1x10000                  80000  double

Total is 10072 elements using 80217 bytes
>> v=priceY(1:10)
#存储数据
>> save hello.mat v;
#将v存储为hello.mat
>> save hello.txt v -ascii
#save as text(ASCII)
```

4.在系统中操作数据

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

   1   2
   3   4
   5   6

>> A(3,2)
ans =  6
>> A(2,:)
ans =

   3   4

# ":"means every element along that row/column
>> A(:,2)
ans =

   2
```

```
   4
   6
>> A([1,3],:)
ans =

   1   2
   5   6

>> A
A =

   1   2
   3   4
   5   6

>> A(:,2)
ans =

   2
   4
   6

>> A(:,2)=[10;11;12]
A =

    1   10
    3   11
    5   12
#第2列被替换为[10;11;12]
>>>> A=[A,[100;101;102]];
>> A
A =

     1     2   100
     3     4   101
     5     6   102
#在原来的矩阵A右边附上一个新的列矩阵
>> A=[A,[100;101;102]]
A =

     1     2   100   100
     3     4   101   101
     5     6   102   102

>> size(A)
ans =

   3   4

>> A(:)
ans =

     1
     3
     5
     2
```

```
     4
     6
   100
   101
   102
   100
   101
   102
#把A中的所有元素放入一个单独的列向量，得到一个12*1的向量，这些元素都是A中元素排列起来的
>>
>>>> A=[1 2;3 4;5 6]
A =

   1   2
   3   4
   5   6

>> B=[11 12;13 14;15 16]
B =

   11   12
   13   14
   15   16

>> C=[A B]#与[A,B]一样
C =

    1    2   11   12
    3    4   13   14
    5    6   15   16
#把两个矩阵直接连接起来，A在左边，B在右边，组成了矩阵C
>> C=[A;B]
C =

    1    2
    3    4
    5    6
   11   12
   13   14
   15   16
#用\：''隔开，A在B的上面
>>
```

## 1.3 Computing on Data

## 1.4 Plotting Data

## 1.5 Control Statements:for,while,if statement

## 1.6 Vectorization