Machine Learning

tanshuqiu

Email: tsq@cqut.edu.cn

From Data to Knowledge Discovery

1. Distinguishing the types of variables
2. Data preparation
3. Exploratory statistics - numerical measures
4. Exploratory visualization

1. Distinguishing the types of variables

Modern computer technology, coupled with the availability of more and more powerful sensors, has led to impressive-sized collections of information. Having a lot of data, on one hand, undoubtedly represents an advantage; on the other hand, it is a problem. This is because it imposes obvious management problems, in the sense that more sophisticated tools will be needed to extract knowledge from it.

1. Distinguishing the types of variables

It is therefore necessary to follow a path to transform data into an element of knowledge.

The two important steps in this path are the analysis, which extracts information from the raw data, and the model, which allows the information to be included in an interpretative context. This context defines its meaning and establishes correlation with other pieces of information, contributing in this way to the knowledge of the phenomenon.

1. Distinguishing the types of variables

Quantitative variables

Quantitative variables (also called continuous variables) are an expression of a measure and are presented in the form of numerical data. Some examples of quantitative variables are temperature, pressure, and humidity values of a precise location. Quantitative variables can be further categorized as either interval or ratio variables.

1. Distinguishing the types of variables

Quantitative variables

Interval variables are variables that assume numeric values that allow comparisons only by difference.

New York 10 °C

Miami 20 °C

Mexico City 30 °C

1. Distinguishing the types of variables

Qualitative variables

Qualitative variables, also called categorical variables, are variables that are not numerical. Categorical variables can be further grouped as nominal, dichotomous, or ordinal.

Nominal variables are variables that have two or more categories, but do not have an intrinsic order. For example, the blood group variable, limited to the ABO system, can assume values A, B, AB, O.

1. Distinguishing the types of variables

Qualitative variables

The dichotomous variables are a special case of nominal variables that have only two categories or levels, for example, gender.

Ordinal variables are variables that have two or more categories, just like nominal variables, but compared to the former, they can also be ordered or ranked. For example, the presence of blood in urine may take the following values: absent, traces, +, ++, +++.

2. Data preparation

We must first proceed to the preparation of data (data wrangling). This is a laborious process that can take a long time, in some cases about 80 percent of the entire data analysis process. However, it is a fundamental prerequisite for the rest of the data analysis workflow, so it is essential to acquire the best practices in such techniques.

2. Data preparation

A first look at data

Before passing our data to machine learning algorithms, we need to give a first look at what we've imported into MATLAB to see if there are any issues. Often, raw data is messy and poorly formatted. In other cases, it may not have the appropriate details for our study. Correcting the data in progress can be destructive because it can be overwritten without the ability to restore the original data.

To get started, it's good practice to keep your original data. To do this, every change will be performed on a copy of the dataset.

2. Data preparation

A first look at data

Let‘s move on to a practical example, now we need navigate to the folder where we saved the file.

```
>> cd('C:\Users\lavoro\Documents\MATLAB\MatlabForML')
```

It's possible to load the data with the help of the following command:

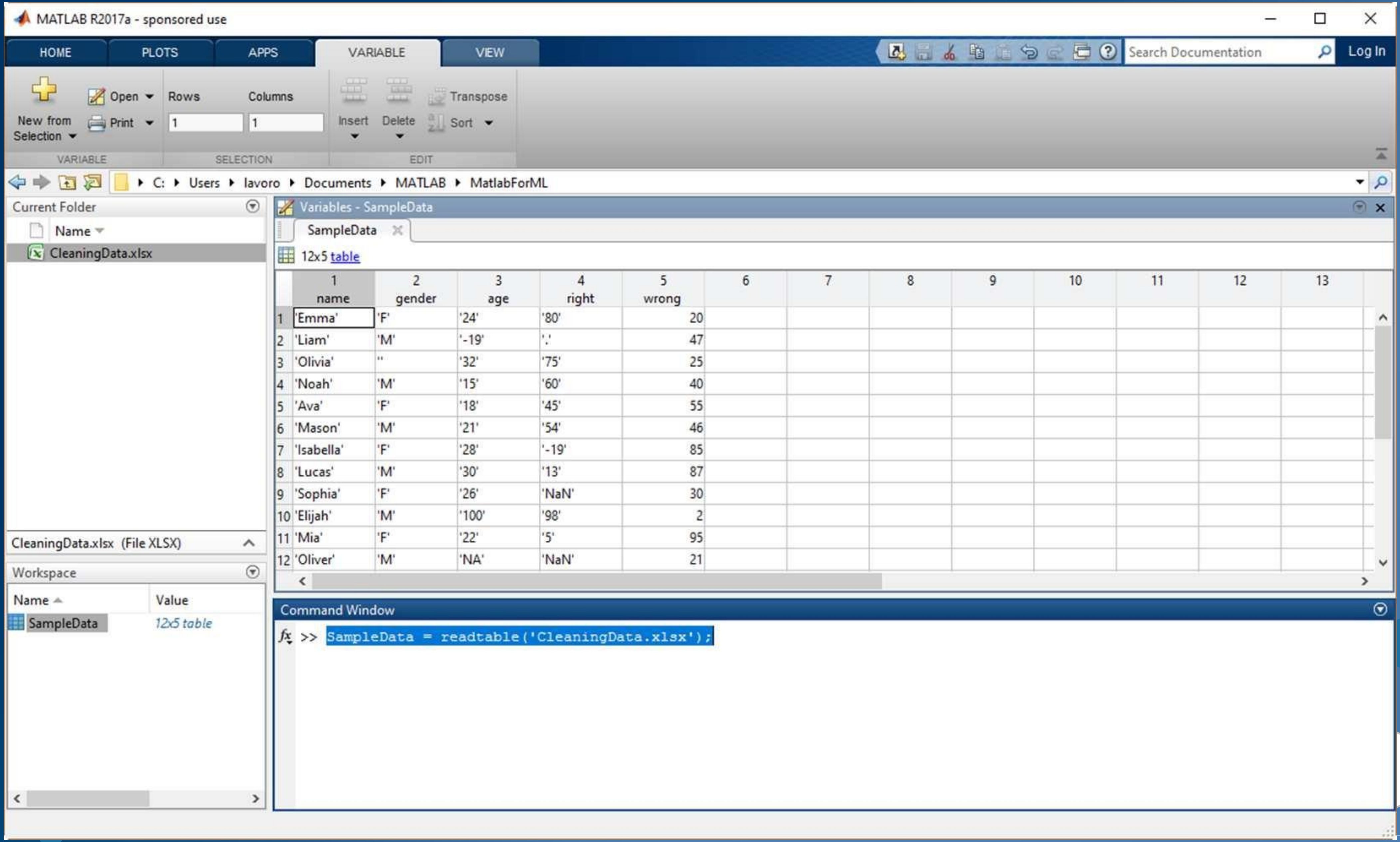
```
>> SampleData = readtable('CleaningData.xlsx');
```

2. Data preparation

A first look at data

we print a summary of the main features with this command:

```
>> summary(SampleData)
Variables:
    name: 12×1 cell array of character vectors
    gender: 12×1 cell array of character vectors
    age: 12×1 cell array of character vectors
    right: 12×1 cell array of character vectors
    wrong: 12×1 double
        Values:
            Min          2
            Median      43
            Max         95
```


MATLAB R2017a - sponsored use
HOME
PLOTS
APPS
VARIABLE
VIEW
Search Documentation
Log In
New from Selection
Open
Print
Rows
Columns
Insert
Delete
Transpose
Sort
VARIABLE
SELECTION
EDIT
C: ▸ Users ▸ lavoro ▸ Documents ▸ MATLAB ▸ MatlabForML
Current Folder
Name
CleaningData.xlsx
CleaningData.xlsx (File XLSX)
Workspace
Name
Value
SampleData
12x5 table
Variables - SampleData
SampleData
12x5 table

	1 name	2 gender	3 age	4 right	5 wrong
1	'Emma'	'F'	'24'	'80'	20
2	'Liam'	'M'	'-19'	'.'	47
3	'Olivia'	''	'32'	'75'	25
4	'Noah'	'M'	'15'	'60'	40
5	'Ava'	'F'	'18'	'45'	55
6	'Mason'	'M'	'21'	'54'	46
7	'Isabella'	'F'	'28'	'-19'	85
8	'Lucas'	'M'	'30'	'13'	87
9	'Sophia'	'F'	'26'	'NaN'	30
10	'Elijah'	'M'	'100'	'98'	2
11	'Mia'	'F'	'22'	'5'	95
12	'Oliver'	'M'	'NA'	'NaN'	21

Command Window
>> SampleData = readtable('CleaningData.xlsx');

2. Data preparation

Finding missing values

To find observations with missing values just as fast, we can use the ismissing() function. This function displays the subset of observations that have at least one missing value:

```
>> id = {'NA' '' '-19' -19 NaN '.'};
>> WrongPos = ismissing(SampleData,id);
>> SampleData(any(WrongPos,2),:)
ans =
  4×5 table
       name        gender     age      right     wrong
    __________     ______    _____     _____     _____
    'Liam'         'M'       '-19'     '.'       47
    'Olivia'       ''        '32'      '75'      25
    'Isabella'     'F'       '28'      '-19'     85
    'Oliver'       'M'       'NA'      'NaN'     21
```

2. Data preparation

Changing the datatype

We can correct the problem by converting the char variables that should be numeric using the str2double() function:

```
>> SampleData.age = str2double(SampleData.age);
>> summary(SampleData(:,3))
Variables:
 age: 12×1 double
 Values:
 Min -19
 Median 24
 Max 100
 NumMissing 1
```

```
>> SampleData.right = str2double(SampleData.right);
>> summary(SampleData(:,4))
Variables:
    right: 12×1 double
        Values:
            Min             -19
            Median          54
            Max             98
            NumMissing      3
```

2. Data preparation

Replacing the missing value

we can use the standardizeMissing() function, which replaces the values specified in parentheses with standard missing values in an array or table:

```
>> SampleData = standardizeMissing(SampleData,-19);
>> summary(SampleData(:,3))
Variables:
    age: 12×1 double
        Values:
            Min            15
            Median         25
            Max            100
            NumMissing     2
```

2. Data preparation

Replacing the missing value

In this case, we fill missing values with values from previous rows of the table:

12x5 table

	1 name	2 gender	3 age	4 right	5 wrong	6	7
1	'Emma'	'F'	'24'	'80'	20		
2	'Liam'	'M'	'-19'	'.'	47		
3	'Olivia'	''	'32'	'75'	25		
4	'Noah'	'M'	'15'	'60'	40		
5	'Ava'	'F'	'18'	'45'	55		
6	'Mason'	'M'	'21'	'54'	46		
7	'Isabella'	'F'	'28'	'-19'	85		
8	'Lucas'	'M'	'30'	'13'	87		
9	'Sophia'	'F'	'26'	'NaN'	30		
10	'Elijah'	'M'	'100'	'98'	2		
11	'Mia'	'F'	'22'	'5'	95		
12	'Oliver'	'M'	'NA'	'NaN'	21		

```
>> SampleDataNew = fillmissing(SampleData,'previous')
SampleDataNew =
  12×5 table
        name        gender    age    right    wrong
    __________    ______    ___    _____    _____
    'Emma'        'F'        24     80       20
    'Liam'        'M'        24     80       47
    'Olivia'      'M'        32     75       25
    'Noah'        'M'        15     60       40
    'Ava'         'F'        18     45       55
    'Mason'       'M'        21     54       46
    'Isabella'    'F'        28     54       85
    'Lucas'       'M'        30     13       87
    'Sophia'      'F'        26     13       30
    'Elijah'      'M'       100     98        2
    'Mia'         'F'        22      5       95
    'Oliver'      'M'        22      5       21
```

reparation

function; it removes missing entries

12x5 table

	1 name	2 gender	3 age	4 right	5 wrong	6	7
1	'Emma'	'F'	'24'	'80'	20		
2	'Liam'	'M'	'-19'	'.'	47		
3	'Olivia'	''	'32'	'75'	25		
4	'Noah'	'M'	'15'	'60'	40		
5	'Ava'	'F'	'18'	'45'	55		
6	'Mason'	'M'	'21'	'54'	46		
7	'Isabella'	'F'	'28'	'-19'	85		
8	'Lucas'	'M'	'30'	'13'	87		
9	'Sophia'	'F'	'26'	'NaN'	30		
10	'Elijah'	'M'	'100'	'98'	2		
11	'Mia'	'F'	'22'	'5'	95		
12	'Oliver'	'M'	'NA'	'NaN'	21		

```
>> SampleDataMinor = rmmissing(SampleData)
SampleDataMinor =
  7×5 table
       name       gender    age    right    wrong
     ________    ______    ___    _____    _____
     'Emma'      'F'        24     80       20
     'Noah'      'M'        15     60       40
     'Ava'       'F'        18     45       55
     'Mason'     'M'        21     54       46
     'Lucas'     'M'        30     13       87
     'Elijah'    'M'       100     98        2
     'Mia'       'F'        22      5       95
```

2. Data preparation

Ordering the table

we will order the rows of the newly created table, SampleDataMinor, in descending order with the age variable. In this case, we will use the sortrows() function, which sorts the rows of a matrix in the order specified in parentheses based on the elements in the column, also specified in parentheses:

```
>> SampleDataOrdered = sortrows(SampleDataMinor,{'age'},{'descend'})
SampleDataOrdered =
  7×5 table
        name        gender    age    right    wrong
     _________      ______    ___    _____    _____
     'Elijah'       'M'       100    98        2
     'Lucas'        'M'        30    13       87
     'Emma'         'F'        24    80       20
     'Mia'          'F'        22     5       95
     'Mason'        'M'        21    54       46
     'Ava'          'F'        18    45       55
     'Noah'         'M'        15    60       40
```

2. Data preparation

Finding outliers in data

Outliers are the values that, compared to others, are particularly extreme. Outliers are a problem because they tend to distort data analysis results, in particular in descriptive statistics and correlations. These should be identified in the data cleaning phase, but can also be dealt in the next step of data analysis. Outliers can be univariate when they have an extreme value for a single variable or multivariate when they have an unusual combination of values on a number of variables.

```
>> SampleDataOutlier = isoutlier(SampleDataNew(2:end,3:5))
SampleDataOutlier =
  11×3 logical array
   0    0    0
   0
   0
   0
   0
   0
   0
   0
   1
   0
   0
```

aration

```
SampleDataNew =

  12×5 table
```

name	gender	age	right	wrong
'Emma'	'F'	24	80	20
'Liam'	'M'	24	80	47
'Olivia'	'M'	32	75	25
'Noah'	'M'	15	60	40
'Ava'	'F'	18	45	55
'Mason'	'M'	21	54	46
'Isabella'	'F'	28	54	85
'Lucas'	'M'	30	13	87
'Sophia'	'F'	26	13	30
'Elijah'	'M'	$\boxed{100}$	98	2
'Mia'	'F'	22	5	95
'Oliver'	'M'	22	5	21

```
SampleDataOutlier =

  11×3 logical array

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

ified easily and effectively.

lly finds outliers in data.

This function returns a logical array whose elements are true when an outlier is detected in the corresponding item in the table. By default, MATLAB identifies an outlier if it is more than three average escalating MADs (median absolute deviation) far from the median.

2. Data preparation

Organizing multiple sources of data into one

```
>> SampleData1 = SampleDataNew(1:6,:)
SampleData1 =
 6×5 table
 name gender age right wrong
 ________ ______ ___ _____ _____
 'Emma' 'F' 24 80 20
 'Liam' 'M' 24 80 47
 'Olivia' 'M' 32 75 25
 'Noah' 'M' 15 60 40
 'Ava' 'F' 18 45 55
 'Mason' 'M' 21 54 46

>> SampleData2 = SampleDataNew(7:end,:)
SampleData2 =
 6×5 table
 name gender age right wrong
 __________ ______ ___ _____ _____
 'Isabella' 'F' 28 54 85
 'Lucas' 'M' 30 13 87
 'Sophia' 'F' 26 13 30
 'Elijah' 'M' 100 98 2
 'Mia' 'F' 22 5 95
 'Oliver' 'M' 22 5 21
```

```
>> SampleDataComplete = [SampleData1;SampleData2]
SampleDataComplete =
  12×5 table
        name          gender     age     right     wrong
    ___________      _______     ___     _____     _____
    'Emma'           'F'          24     80        20
    'Liam'           'M'          24     80        47
```

```
    'Olivia'        'M'        32     75       25
    'Noah'          'M'        15     60       40
    'Ava'           'F'        18     45       55
    'Mason'         'M'        21     54       46
    'Isabella'      'F'        28     54       85
    'Lucas'         'M'        30     13       87
    'Sophia'        'F'        26     13       30
    'Elijah'        'M'       100     98        2
    'Mia'           'F'        22      5       95
    'Oliver'        'M'        22      5       21
```

2. Data preparation

In addition, we have some data that allows us to add useful information about gender features.

```
>> LifeExpectancy = readtable('LifeExpectancy.xlsx')
LifeExpectancy =
  2×3 table
       state        Le       gender

    ___________    _____    ______

    'Hong Kong'    80.91    'M'
    'Hong Kong'    86.58    'F'
```

```
>> SampleDataLE = join(SampleDataComplete,LifeExpectancy,'Keys','gender')
SampleDataLE =
  12×7 table
name          gender    age    right    wrong       state         Le

_______       ______    ___    _____    _____    ___________    _____

'Emma'        'F'        24     80       20      'Hong Kong'    86.58
'Liam'        'M'        24     80       47      'Hong Kong'    80.91
'Olivia'      'F'        32     75       25      'Hong Kong'    86.58
'Noah'        'M'        15     60       40      'Hong Kong'    80.91
'Ava'         'F'        18     45       55      'Hong Kong'    86.58
'Mason'       'M'        21     54       46      'Hong Kong'    80.91
'Isabella'    'F'        28     54       85      'Hong Kong'    86.58
'Lucas'       'M'        30     13       87      'Hong Kong'    80.91
'Sophia'      'F'        26     13       30      'Hong Kong'    86.58
'Elijah'      'M'       100     98        2      'Hong Kong'    80.91
'Mia'         'F'        22      5       95      'Hong Kong'    86.58
'Oliver'      'M'        22      5       21      'Hong Kong'    80.91
```

3. Exploratory statistics – numerical measures

The purpose of exploratory analysis is to use statistical indicator and visualizations to better understand the data, find clues about data trends and its quality, and formulate hypotheses from our analysis. We do not want to make imaginative or aesthetically pleasing views to surprise the interlocutor; our goal is to try to answer specific questions through data analysis.

3. Exploratory statistics – numerical measures

Measures of location

```
>> GlassIdentificationDataSet = readtable('GlassIdentificationDataSet.xlsx');
```

This is a dataset with 214 records and 11 variables (id, refractive index, Na, Mg, Al, Si, K, Ca, Ba, Fe, and type of glass).

3. Exploratory statistics – numerical measures

Measures of location

We begin our explorative analysis by calculating the maximum, mean, and minimum of the newly imported table. For this purpose, we use three useful functions: max(), mean(),and min().

```
>> Max = max(GlassIdentificationDataSet{:,3:8})
Max =
17.3800    4.4900    3.5000   75.4100    6.2100   16.1900
```

```
>> Mean = mean(GlassIdentificationDataSet{:,3:8})
Mean =
13.4079    2.6845    1.4449   72.6509    0.4971    8.9570
```

```
>> Min = min(GlassIdentificationDataSet{:,3:8})
Min =
10.7300         0    0.2900   69.8100         0    5.4300
```

3. Exploratory statistics – numerical measures

It may be useful to identify the records in which minimum and maximum are found.

```
>> [Max,IndRowMax] = max(GlassIdentificationDataSet{:,3:8})
Max =
17.3800    4.4900    3.5000   75.4100    6.2100   16.1900
IndRowMax =
185      1    164    185    172    108

>> [Min,IndRowMin] = min(GlassIdentificationDataSet{:,3:8})
Min =
10.7300         0    0.2900   69.8100         0    5.4300
IndRowMin =
107    106     22    107     64    186
```

The mode is the value that appears most often in a set of data.

```
>> Mode = mode(GlassIdentificationDataSet{:,3:8})
Mode =
13.0000         0    1.5400   72.8600         0    8.0300
```

3. Exploratory statistics – numerical measures

Measures of dispersion

The range() function returns the difference between the maximum and the minimum of a sample:

```
>> Range = range(GlassIdentificationDataSet{:,3:8})
Range =
    6.6500    4.4900    3.2100    5.6000    6.2100   10.7600
```

MATLAB calculates the variance for each column with the var() function:

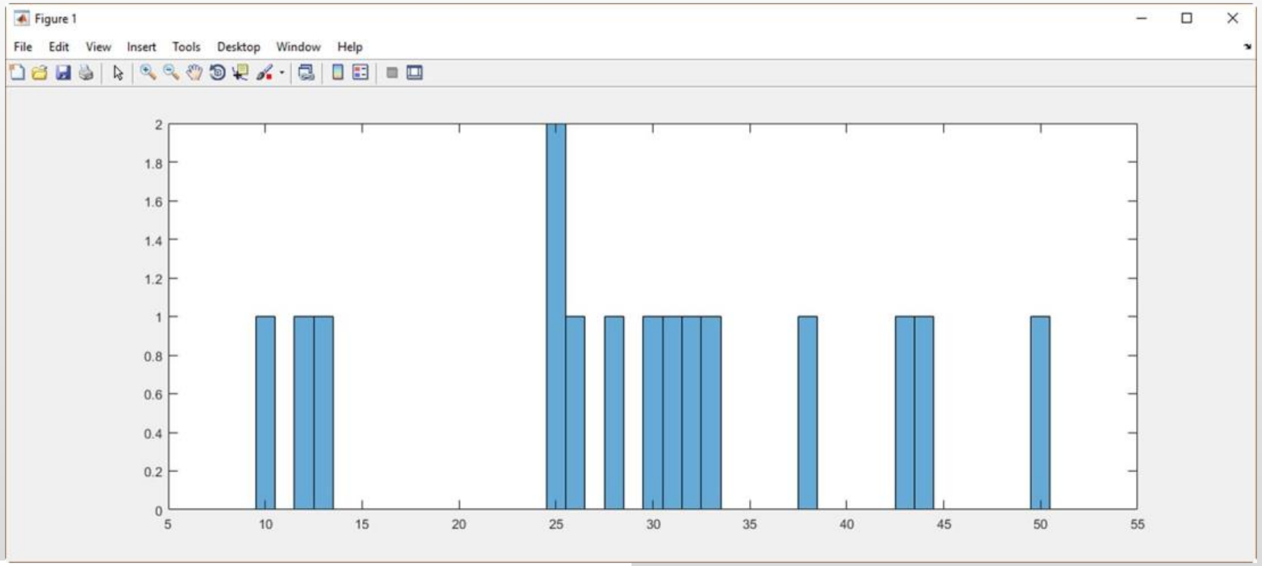
```
>> Variance = var(GlassIdentificationDataSet{:,3:8})
Variance =
    0.6668    2.0805    0.2493    0.5999    0.4254    2.0254
```

4. Exploratory visualization

Histogram

In the MATLAB environment, it is possible to create histograms with the histogram() function.

```
>> histogram(x)
```




```
>> Vect1=[10,25,12,13,33,25,44,50,43,26,38,32,31,28,30];
>> histogram(Vect1)
```

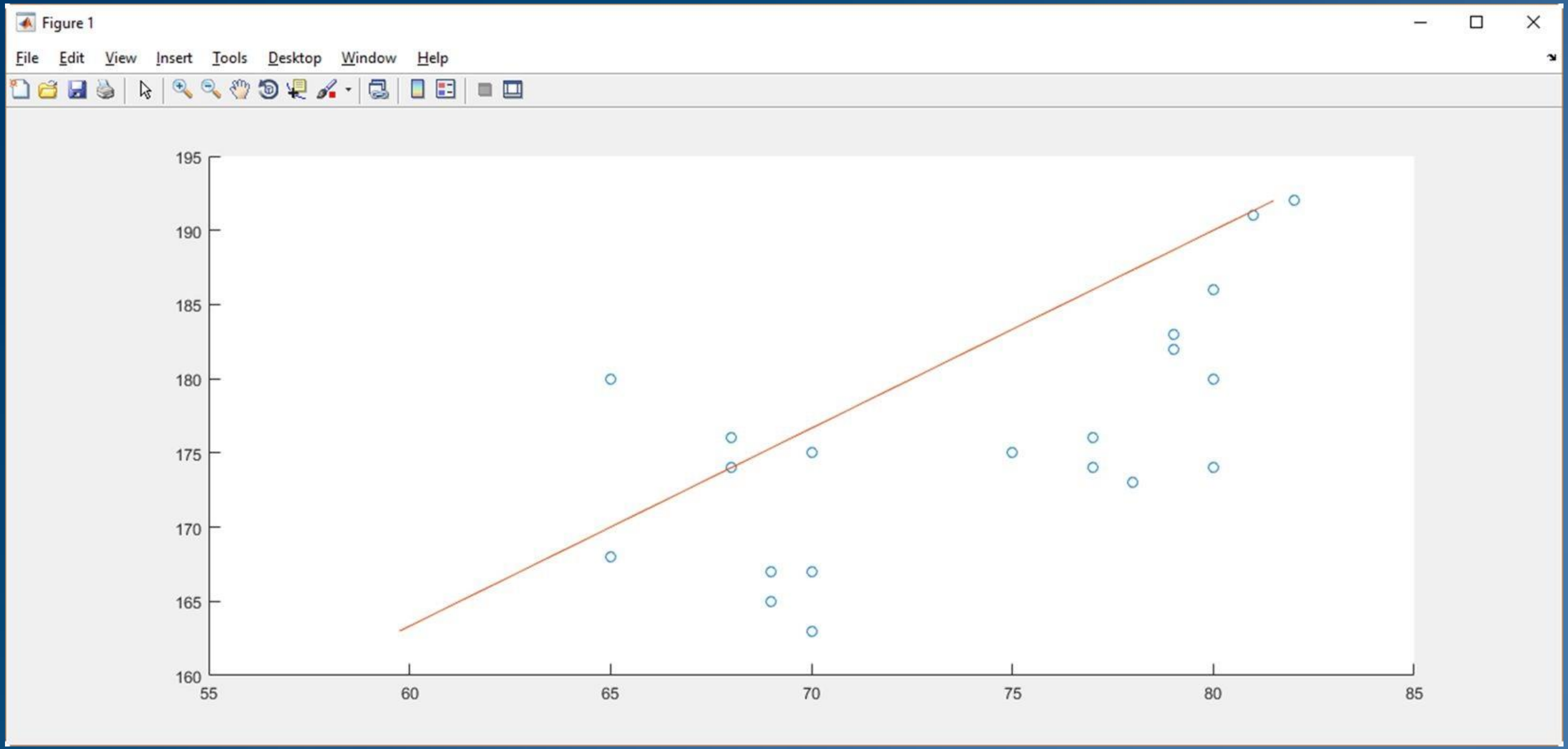
4. Exploratory visualization

Scatter plots

In MATLAB, to create a scatter plot, we can use scatter() function

```
>> scatter(a,b)
```

```
>> Height = [168  168     168    173    163    174    174    174    175    175    176    165    1
>> Weight = [65       65  65     78     70     68     68     80     70     75     77     69     8
>> scatter(Weight,Height)
>> IdealWeight=Height-100-[(Height-150)/4];
>> hold on
>> plot(IdealWeight,Height)
```




```
>> Height = [168  168     168   173   163   174   174   174   175   175   176   165   1
>> Weight = [65        65   65    78    70    68    68    80    70    75    77    69    8
>> scatter(Weight,Height)
>> IdealWeight=Height-100-[(Height-150)/4];
>> hold on
>> plot(IdealWeight,Height)
```