MINING DATA WITH R

R is just a tool

GIVE ME 6 HOURS TO CUT DOWN A TREE AND I WILL SPEND THE FIRST FOUR HOURS SHARPENING MY AXE
....ANONYMOUS

TO BE CONTINUED

Continuous Learning

You have taken the first step toward sharpening your axe!

Table 1.1 Example Analytics Applications

Marketing	Risk Management	Government	Web	Logistics	Other
Response modeling	Credit risk modeling	Tax avoidance	Web analytics	Demand forecasting	Text analytics
Net lift modeling	Market risk modeling	Social security fraud	Social media analytics	Supply chain analytics	Business process analytics
Retention modeling	Operational risk modeling	Money laundering	Multivariate testing		
Market basket analysis	Fraud detection	Terrorism detection			
Recommender systems					
Customer segmentation					

WHY: INDUSTRY –APPLICATIONS

Mining large amounts of structured and unstructured data to identify patterns that can help an organization rein in costs, increase efficiencies, recognize new market opportunities, understand and predict customer behavior and increase an organization's competitive advantage.

WHAT: DATA → DATA SCIENCE

- python
- Apache Spark
- Julia
- SAS, SPSS
- Weka, H2O etc…
- R

THERE ARE SO MANY TOOLS

OUR CHOICE IS R!

WHY, WHAT AND HOW?

Why

What

How

Why should we learn R?

What are we seeking to learn in R?

How to learn R?

R is a data centric language
Written for data manipulation
Written by those whose only occupation was data
It is the most statistically grounded language
Results are robust and statistically valid

R is COMPELLING for
Exploratory Data Analysis
EDA aids in understanding data
EDA = quick analysis with effortless visualization
EDA is most important step in data analysis

Our standard process for data analysis → CRISP-DM
Understand Business, Understand Data, prepare data,model,evaluate, deploy
https://en.wikipedia.org/wiki/Cross-industry_standard_process_for_data_mining

RULE#1: ASK QUESTIONS

- A language designed for vector processing
- Functional – repository based
- Memory driven
- Free, actively maintained by serious statisticians
- Learn by doing – from example solutions
  - Make it a habit -- R-bloggers.com, stackoverflow.com/
  - Download R from https://www.r-project.org/
  - Rstudio and rmarkdown can wait.
- http://www.r-tutor.com/content/r-tutorial-ebook

WHAT IS R?

DATA MINING AND ANALYSIS

Fundamental Concepts and Algorithms

MOHAMMED J. ZAKI
Rensselaer Polytechnic Institute, Troy, New York

WAGNER MEIRA JR.
Universidade Federal de Minas Gerais, Brazil

- http://dataminingbook.info
- http://www.cs.rpi.edu/~zaki/dataminingbook
- http://www.dcc.ufmg.br/dataminingbook

WE WILL WORK THROUGH SOME EXAMPLES FROM THIS BOOK

R system supports many types of objects
Scalars, vectors,lists, array, matrix, data.frame,struct

```
aScalar<-3
aVector<-seq(1:7)
anotherVector<-1:7
alist<-list(aScalar,aVector,anotherVector)
o2<-alist[2]
class(o2)
o2[1]
o2l<-o2[[1]]
o2l[3]
```

```
named.alist<-list(s=aScalar,v=aVector,av=anotherVector)
named.alist$s
named.alist$v
```

```
named.alist<-list(s=aScalar,v=aVector,av=anotherVector)
named.alist$s
named.alist$v
names(named.alist)
lapply(named.alist,length)
```

Switch gear up a notch

```
mx<-cbind(sample(c('m','F'),10,replace=T))
df2<-data.frame(gender=sample(c('M','F'),10,replace=T),age=sample(1:300,10,replace=T))
```

HOW? BY DOING, LET US START

MATH

```
exp(named.alist$v)
log(exp(named.alist$v))


R like many other PL can do any kind of math
The difference R understands a vector and does vector math
```

```
a = c(1, 3, 5, 7)
b = c(1, 2, 4, 8)
```

Data mining is the process of discovering insightful, interesting, and novel patterns, as well as descriptive, understandable, and predictive models from large-scale data. We begin this chapter by looking at basic properties of data modeled as a data matrix. We emphasize the geometric and algebraic views, as well as the probabilistic interpretation of data. We then discuss the main data mining tasks, which span exploratory data analysis, frequent pattern mining, clustering, and classification, laying out the roadmap for the book.

1.1 DATA MATRIX

Data can often be represented or abstracted as an $n \times d$ *data matrix*, with n rows and d columns, where rows correspond to entities in the dataset, and columns represent attributes or properties of interest. Each row in the data matrix records the observed attribute values for a given entity. The $n \times d$ data matrix is given as

$$\mathbf{D} = \left(\begin{array}{c|cccc} & X_1 & X_2 & \cdots & X_d \\ \hline \mathbf{x}_1 & x_{11} & x_{12} & \cdots & x_{1d} \\ \mathbf{x}_2 & x_{21} & x_{22} & \cdots & x_{2d} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{x}_n & x_{n1} & x_{n2} & \cdots & x_{nd} \end{array}\right)$$

where $\mathbf{x}_i$ denotes the ith row, which is a d-tuple given as

$$\mathbf{x}_i = (x_{i1}, x_{i2}, \ldots, x_{id})$$

and X_j denotes the jth column, which is an n-tuple given as

$$X_j = (x_{1j}, x_{2j}, \ldots, x_{nj})$$

Depending on the application domain, rows may also be referred to as *entities*, *instances*, *examples*, *records*, *transactions*, *objects*, *points*, *feature-vectors*, *tuples*, and so on. Likewise, columns may also be called *attributes*, *properties*, *features*, *dimensions*, *variables*, *fields*, and so on. The number of instances n is referred to as the *size* of

Not all datasets are in the form of a data matrix. For instance, more complex datasets can be in the form of sequences (e.g., DNA and protein sequences), text, time-series, images, audio, video, and so on, which may need special techniques for analysis. However, in many cases even if the raw data is not a data matrix it can usually be transformed into that form via feature extraction. For example, given a

Each row is an observation. Each column
Is an attribute. Also known as Wide Format.

Course,title,cr,faculty,dept
CS6513, Big Data,3,RK,CSE
CS6923,ML,RK,CSE

There is a long format in which an observation spans multiple rows, as many rows as there are attributes. Each row has one attribute and the value associated with that attribute.

Course,id,cs6923
Course,title,ML
Course,faculty,RK
Course,dept,CSE

DATA MATRIX

1.2 ATTRIBUTES

Attributes may be classified into two main types depending on their domain, that is, depending on the types of values they take on.

Numeric Attributes

A *numeric* attribute is one that has a real-valued or integer-valued domain. For example, `Age` with $domain(\texttt{Age}) = \mathbb{N}$, where $\mathbb{N}$ denotes the set of natural numbers (non-negative integers), is numeric, and so is `petal length` in Table 1.1, with $domain(\texttt{petal length}) = \mathbb{R}^+$ (the set of all positive real numbers). Numeric attributes that take on a finite or countably infinite set of values are called *discrete*, whereas those that can take on any real value are called *continuous*. As a special case of discrete, if an attribute has as its domain the set $\{0, 1\}$, it is called a *binary* attribute. Numeric attributes can be classified further into two types:

- *Interval-scaled*: For these kinds of attributes only differences (addition or subtraction) make sense. For example, attribute `temperature` measured in °C or °F is interval-scaled. If it is 20 °C on one day and 10 °C on the following day, it is meaningful to talk about a temperature drop of 10 °C, but it is not meaningful to say that it is twice as cold as the previous day.
- *Ratio-scaled*: Here one can compute both differences as well as ratios between values. For example, for attribute `Age`, we can say that someone who is 20 years old is twice as old as someone who is 10 years old.

ATTRIBUTES: WITHOUT ATTRIBUTES SILVER=COPPER

4 **Data Mining and Analysis**

1.3 DATA: ALGEBRAIC AND GEOMETRIC VIEW

If the d attributes or dimensions in the data matrix $\mathbf{D}$ are all numeric, then each row can be considered as a d-dimensional point:

$$\mathbf{x}_i = (x_{i1}, x_{i2}, \ldots, x_{id}) \in \mathbb{R}^d$$

or equivalently, each row may be considered as a d-dimensional column vector (all vectors are assumed to be column vectors by default):

$$\mathbf{x}_i = \begin{pmatrix} x_{i1} \\ x_{i2} \\ \vdots \\ x_{id} \end{pmatrix} = \begin{pmatrix} x_{i1} & x_{i2} & \cdots & x_{id} \end{pmatrix}^T \in \mathbb{R}^d$$

where T is the *matrix transpose* operator.

The d-dimensional Cartesian coordinate space is specified via the d unit vectors, called the standard basis vectors, along each of the axes. The jth *standard basis vector* $\mathbf{e}_j$ is the d-dimensional unit vector whose jth component is 1 and the rest of the components are 0

$$\mathbf{e}_j = (0, \ldots, 1_j, \ldots, 0)^T$$

Any other vector in $\mathbb{R}^d$ can be written as a *linear combination* of the standard basis vectors. For example, each of the points $\mathbf{x}_i$ can be written as the linear combination

$$\mathbf{x}_i = x_{i1}\mathbf{e}_1 + x_{i2}\mathbf{e}_2 + \cdots + x_{id}\mathbf{e}_d = \sum_{j=1}^{d} x_{ij}\mathbf{e}_j$$

where the scalar value x_{ij} is the coordinate value along the jth axis or attribute.

Each numeric column or attribute can also be treated as a vector in an n-dimensional space $\mathbb{R}^n$:

$$X_j = \begin{pmatrix} x_{1j} \\ x_{2j} \\ \vdots \\ x_{nj} \end{pmatrix}$$

If all attributes are numeric, then the data matrix $\mathbf{D}$ is in fact an $n \times d$ matrix, also written as $\mathbf{D} \in \mathbb{R}^{n \times d}$, given as

$$\mathbf{D} = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1d} \\ x_{21} & x_{22} & \cdots & x_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nd} \end{pmatrix} = \begin{pmatrix} - \mathbf{x}_1^T - \\ - \mathbf{x}_2^T - \\ \vdots \\ - \mathbf{x}_n^T - \end{pmatrix} = \begin{pmatrix} | & | & & | \\ X_1 & X_2 & \cdots & X_d \\ | & | & & | \end{pmatrix}$$

As we can see, we can consider the entire dataset as an $n \times d$ matrix, or equivalently as a set of n row vectors $\mathbf{x}_i^T \in \mathbb{R}^d$ or as a set of d column vectors $X_j \in \mathbb{R}^n$.

LINEAR ALGEBRA: FOUNDATIONS

1.3.1 Distance and Angle

Treating data instances and attributes as vectors, and the entire dataset as a matrix, enables one to apply both geometric and algebraic methods to aid in the data mining and analysis tasks.

Let $\mathbf{a}, \mathbf{b} \in \mathbb{R}^m$ be two m-dimensional vectors given as

$$\mathbf{a} = \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_m \end{pmatrix} \qquad \mathbf{b} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{pmatrix}$$

Dot Product

The *dot product* between $\mathbf{a}$ and $\mathbf{b}$ is defined as the scalar value

$$\mathbf{a}^T\mathbf{b} = \begin{pmatrix} a_1 & a_2 & \cdots & a_m \end{pmatrix} \times \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{pmatrix} = a_1b_1 + a_2b_2 + \cdots + a_mb_m = \sum_{i=1}^{m} a_ib_i$$

Length

The *Euclidean norm* or *length* of a vector $\mathbf{a} \in \mathbb{R}^m$ is defined as

$$\|\mathbf{a}\| = \sqrt{\mathbf{a}^T\mathbf{a}} = \sqrt{a_1^2 + a_2^2 + \cdots + a_m^2} = \sqrt{\sum_{i=1}^{m} a_i^2}$$

The *unit vector* in the direction of $\mathbf{a}$ is given as

$$\mathbf{u} = \frac{\mathbf{a}}{\|\mathbf{a}\|} = \left(\frac{1}{\|\mathbf{a}\|}\right)\mathbf{a}$$

By definition $\mathbf{u}$ has length $\|\mathbf{u}\| = 1$, and it is also called a *normalized* vector, which can be used in lieu of $\mathbf{a}$ in some analysis tasks.

Distance

From the Euclidean norm we can define the *Euclidean distance* between $\mathbf{a}$ and $\mathbf{b}$, as follows

$$\delta(\mathbf{a}, \mathbf{b}) = \|\mathbf{a} - \mathbf{b}\| = \sqrt{(\mathbf{a} - \mathbf{b})^T(\mathbf{a} - \mathbf{b})} = \sqrt{\sum_{i=1}^{m} (a_i - b_i)^2} \qquad (1.1)$$

Thus, the length of a vector is simply its distance from the zero vector $\mathbf{0}$, all of whose elements are 0, that is, $\|\mathbf{a}\| = \|\mathbf{a} - \mathbf{0}\| = \delta(\mathbf{a}, \mathbf{0})$.

From the general L_p-norm we can define the corresponding L_p-distance function, given as follows

$$\delta_p(\mathbf{a}, \mathbf{b}) = \|\mathbf{a} - \mathbf{b}\|_p \qquad (1.2)$$

If p is unspecified, as in Eq. (1.1), it is assumed to be $p = 2$ by default.

LINEAR ALGEBRA: DISTANCE

```
t(a)*b/(sqrt(sum(a*a))*sqrt(sum(b*b)))
```

Angle
The cosine of the smallest angle between vectors $\mathbf{a}$ and $\mathbf{b}$, also called the *cosine similarity*, is given as

$$\cos\theta = \frac{\mathbf{a}^T\mathbf{b}}{\|\mathbf{a}\|\,\|\mathbf{b}\|} = \left(\frac{\mathbf{a}}{\|\mathbf{a}\|}\right)^T \left(\frac{\mathbf{b}}{\|\mathbf{b}\|}\right) \tag{1.3}$$

Thus, the cosine of the angle between $\mathbf{a}$ and $\mathbf{b}$ is given as the dot product of the unit vectors $\frac{\mathbf{a}}{\|\mathbf{a}\|}$ and $\frac{\mathbf{b}}{\|\mathbf{b}\|}$.

The *Cauchy–Schwartz* inequality states that for any vectors $\mathbf{a}$ and $\mathbf{b}$ in $\mathbb{R}^m$

$$|\mathbf{a}^T\mathbf{b}| \le \|\mathbf{a}\| \cdot \|\mathbf{b}\|$$

It follows immediately from the Cauchy–Schwartz inequality that

$$-1 \le \cos\theta \le 1$$

```
> a = c(1, 3, 5, 7)
> b = c(1, 2, 4, 8)
> t(a)*b/(sqrt(sum(a*a))*sqrt(sum(b*b)))
           [,1]       [,2]      [,3]      [,4]
[1,] 0.01183453 0.07100716 0.2366905 0.6627335
> sum(t(a)*b/(sqrt(sum(a*a))*sqrt(sum(b*b))))
[1] 0.9822657
> sum(a*b/(sqrt(sum(a*a))*sqrt(sum(b*b))))
[1] 0.9822657
```

```
a = c(1, 3, 5, 7)
b = c(1, 2, 4, 8)
5 * a                                 4D
a + b
a*b
sum(a*b/(sqrt(sum(a*a))*sqrt(sum(b*b))))
```

```
> a<-c(5,3)
> b<-c(1,4)
> c<-a-b                              2D
> c
> c*c
> sum(c*c)
> absa<-sum(a*a)
> ua<-a/sqrt(absa)
> ua
[1] 0.8574929 0.5144958
```

NORM, ANGLE, COSINE

Orthogonality
Two vectors $\mathbf{a}$ and $\mathbf{b}$ are said to be *orthogonal* if and only if $\mathbf{a}^T\mathbf{b} = 0$, which in turn implies that $\cos\theta = 0$, that is, the angle between them is 90° or $\frac{\pi}{2}$ radians. In this case, we say that they have no similarity.

Example 1.3 (Distance and Angle). Figure 1.3 shows the two vectors

$$\mathbf{a} = \begin{pmatrix}5\\3\end{pmatrix} \text{ and } \mathbf{b} = \begin{pmatrix}1\\4\end{pmatrix}$$

Using Eq. (1.1), the Euclidean distance between them is given as

$$\delta(\mathbf{a},\mathbf{b}) = \sqrt{(5-1)^2 + (3-4)^2} = \sqrt{16+1} = \sqrt{17} = 4.12$$

The distance can also be computed as the magnitude of the vector:

$$\mathbf{a} - \mathbf{b} = \begin{pmatrix}5\\3\end{pmatrix} - \begin{pmatrix}1\\4\end{pmatrix} = \begin{pmatrix}4\\-1\end{pmatrix}$$

because $\|\mathbf{a}-\mathbf{b}\| = \sqrt{4^2 + (-1)^2} = \sqrt{17} = 4.12$.

The unit vector in the direction of $\mathbf{a}$ is given as

$$\mathbf{u}_a = \frac{\mathbf{a}}{\|\mathbf{a}\|} = \frac{1}{\sqrt{5^2+3^2}}\begin{pmatrix}5\\3\end{pmatrix} = \frac{1}{\sqrt{34}}\begin{pmatrix}5\\3\end{pmatrix} = \begin{pmatrix}0.86\\0.51\end{pmatrix}$$

The unit vector in the direction of $\mathbf{b}$ can be computed similarly:

$$\mathbf{u}_b = \begin{pmatrix}0.24\\0.97\end{pmatrix}$$

These unit vectors are also shown in gray in Figure 1.3.

By Eq. (1.3) the cosine of the angle between $\mathbf{a}$ and $\mathbf{b}$ is given as

$$\cos\theta = \frac{\begin{pmatrix}5\\3\end{pmatrix}^T \begin{pmatrix}1\\4\end{pmatrix}}{\sqrt{5^2+3^2}\sqrt{1^2+4^2}} = \frac{17}{\sqrt{34\times 17}} = \frac{1}{\sqrt{2}}$$

We can get the angle by computing the inverse of the cosine:

$$\theta = \cos^{-1}(1/\sqrt{2}) = 45^\circ$$

Let us consider the L_p-norm for $\mathbf{a}$ with $p = 3$; we get

$$\|\mathbf{a}\|_3 = (5^3 + 3^3)^{1/3} = (152)^{1/3} = 5.34$$

The distance between $\mathbf{a}$ and $\mathbf{b}$ using Eq. (1.2) for the L_p-norm with $p = 3$ is given as

$$\|\mathbf{a}-\mathbf{b}\|_3 = \|(4,-1)^T\|_3 = (4^3 + |-1|^3)^{1/3} = (65)^{1/3} = 4.02$$

ORTHOGONALITY

1.3.2 Mean and Total Variance

Mean

The *mean* of the data matrix $\mathbf{D}$ is the vector obtained as the average of all the points:

$$mean(\mathbf{D}) = \boldsymbol{\mu} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i$$

Total Variance

The *total variance* of the data matrix $\mathbf{D}$ is the average squared distance of each point from the mean:

$$var(\mathbf{D}) = \frac{1}{n}\sum_{i=1}^{n}\delta(\mathbf{x}_i,\boldsymbol{\mu})^2 = \frac{1}{n}\sum_{i=1}^{n}\|\mathbf{x}_i - \boldsymbol{\mu}\|^2 \qquad (1.4)$$

Simplifying Eq. (1.4) we obtain

$$\begin{aligned} var(\mathbf{D}) &= \frac{1}{n}\sum_{i=1}^{n}\left(\|\mathbf{x}_i\|^2 - 2\mathbf{x}_i^T\boldsymbol{\mu} + \|\boldsymbol{\mu}\|^2\right) \\ &= \frac{1}{n}\left(\sum_{i=1}^{n}\|\mathbf{x}_i\|^2 - 2n\boldsymbol{\mu}^T\left(\frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i\right) + n\|\boldsymbol{\mu}\|^2\right) \\ &= \frac{1}{n}\left(\sum_{i=1}^{n}\|\mathbf{x}_i\|^2 - 2n\boldsymbol{\mu}^T\boldsymbol{\mu} + n\|\boldsymbol{\mu}\|^2\right) \\ &= \frac{1}{n}\left(\sum_{i=1}^{n}\|\mathbf{x}_i\|^2\right) - \|\boldsymbol{\mu}\|^2 \end{aligned}$$

The total variance is thus the difference between the average of the squared magnitude of the data points and the squared magnitude of the mean (average of the points).

Centered Data Matrix

Often we need to center the data matrix by making the mean coincide with the origin of the data space. The *centered data matrix* is obtained by subtracting the mean from all the points:

$$\mathbf{Z} = \mathbf{D} - \mathbf{1}\cdot\boldsymbol{\mu}^T = \begin{pmatrix}\mathbf{x}_1^T \\ \mathbf{x}_2^T \\ \vdots \\ \mathbf{x}_n^T\end{pmatrix} - \begin{pmatrix}\boldsymbol{\mu}^T \\ \boldsymbol{\mu}^T \\ \vdots \\ \boldsymbol{\mu}^T\end{pmatrix} = \begin{pmatrix}\mathbf{x}_1^T - \boldsymbol{\mu}^T \\ \mathbf{x}_2^T - \boldsymbol{\mu}^T \\ \vdots \\ \mathbf{x}_n^T - \boldsymbol{\mu}^T\end{pmatrix} = \begin{pmatrix}\mathbf{z}_1^T \\ \mathbf{z}_2^T \\ \vdots \\ \mathbf{z}_n^T\end{pmatrix} \qquad (1.5)$$

where $\mathbf{z}_i = \mathbf{x}_i - \boldsymbol{\mu}$ represents the centered point corresponding to $\mathbf{x}_i$, and $\mathbf{1} \in \mathbb{R}^n$ is the n-dimensional vector all of whose elements have value 1. The mean of the centered data matrix $\mathbf{Z}$ is $\mathbf{0} \in \mathbb{R}^d$, because we have subtracted the mean $\boldsymbol{\mu}$ from all the points $\mathbf{x}_i$.

STATISTICS

Often in data mining we need to project a point or vector onto another vector, for example, to obtain a new point after a change of the basis vectors. Let $\mathbf{a}, \mathbf{b} \in \mathbb{R}^m$ be two m-dimensional vectors. An *orthogonal decomposition* of the vector $\mathbf{b}$ in the direction of another vector $\mathbf{a}$, illustrated in Figure 1.4, is given as

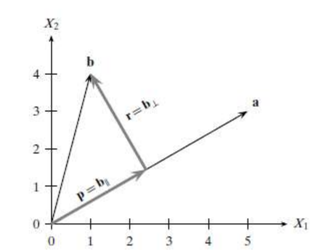


Figure 1.4. Orthogonal projection.

$$\mathbf{b} = \mathbf{b}_\parallel + \mathbf{b}_\perp = \mathbf{p} + \mathbf{r} \qquad (1.6)$$

where $\mathbf{p} = \mathbf{b}_\parallel$ is parallel to $\mathbf{a}$, and $\mathbf{r} = \mathbf{b}_\perp$ is perpendicular or orthogonal to $\mathbf{a}$. The vector $\mathbf{p}$ is called the *orthogonal projection* or simply projection of $\mathbf{b}$ on the vector $\mathbf{a}$. Note that the point $\mathbf{p} \in \mathbb{R}^m$ is the point closest to $\mathbf{b}$ on the line passing through $\mathbf{a}$. Thus, the magnitude of the vector $\mathbf{r} = \mathbf{b} - \mathbf{p}$ gives the *perpendicular distance* between $\mathbf{b}$ and $\mathbf{a}$, which is often interpreted as the residual or error vector between the points $\mathbf{b}$ and $\mathbf{p}$.

We can derive an expression for $\mathbf{p}$ by noting that $\mathbf{p} = c\mathbf{a}$ for some scalar c, as $\mathbf{p}$ is parallel to $\mathbf{a}$. Thus, $\mathbf{r} = \mathbf{b} - \mathbf{p} = \mathbf{b} - c\mathbf{a}$. Because $\mathbf{p}$ and $\mathbf{r}$ are orthogonal, we have

$$\mathbf{p}^T\mathbf{r} = (c\mathbf{a})^T(\mathbf{b} - c\mathbf{a}) = c\mathbf{a}^T\mathbf{b} - c^2\mathbf{a}^T\mathbf{a} = 0$$

which implies that

$$c = \frac{\mathbf{a}^T\mathbf{b}}{\mathbf{a}^T\mathbf{a}}$$

Therefore, the projection of $\mathbf{b}$ on $\mathbf{a}$ is given as

$$\mathbf{p} = \mathbf{b}_\parallel = c\mathbf{a} = \left(\frac{\mathbf{a}^T\mathbf{b}}{\mathbf{a}^T\mathbf{a}}\right)\mathbf{a} \qquad (1.7)$$

PROJECTION: HOW SIMILAR IS A DOG SIMILAR TO AN ELEPHANT?

1.3.4 Linear Independence and Dimensionality

Given the data matrix

$$\mathbf{D} = \begin{pmatrix} \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_n \end{pmatrix}^T = \begin{pmatrix} X_1 & X_2 & \cdots & X_d \end{pmatrix}$$

we are often interested in the linear combinations of the rows (points) or the columns (attributes). For instance, different linear combinations of the original d attributes yield new derived attributes, which play a key role in feature extraction and dimensionality reduction.

Given any set of vectors $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k$ in an m-dimensional vector space $\mathbb{R}^m$, their *linear combination* is given as

$$c_1\mathbf{v}_1 + c_2\mathbf{v}_2 + \cdots + c_k\mathbf{v}_k$$

where $c_i \in \mathbb{R}$ are scalar values. The set of all possible linear combinations of the k vectors is called the *span*, denoted as $span(\mathbf{v}_1, \ldots, \mathbf{v}_k)$, which is itself a vector space being a *subspace* of $\mathbb{R}^m$. If $span(\mathbf{v}_1, \ldots, \mathbf{v}_k) = \mathbb{R}^m$, then we say that $\mathbf{v}_1, \ldots, \mathbf{v}_k$ is a *spanning set* for $\mathbb{R}^m$.

Linear Independence

We say that the vectors $\mathbf{v}_1, \ldots, \mathbf{v}_k$ are *linearly dependent* if at least one vector can be written as a linear combination of the others. Alternatively, the k vectors are linearly dependent if there are scalars $c_1, c_2, \ldots, c_k$, at least one of which is not zero, such that

$$c_1\mathbf{v}_1 + c_2\mathbf{v}_2 + \cdots + c_k\mathbf{v}_k = \mathbf{0}$$

On the other hand, $\mathbf{v}_1, \cdots, \mathbf{v}_k$ are *linearly independent* if and only if

$$c_1\mathbf{v}_1 + c_2\mathbf{v}_2 + \cdots + c_k\mathbf{v}_k = \mathbf{0} \text{ implies } c_1 = c_2 = \cdots = c_k = 0$$

Simply put, a set of vectors is linearly independent if none of them can be written as a linear combination of the other vectors in the set.

LINEAR INDEPENDENCE

The probabilistic view of the data assumes that each numeric attribute X is a *random variable*, defined as a function that assigns a real number to each outcome of an experiment (i.e., some process of observation or measurement). Formally, X is a function $X: \mathcal{O} \to \mathbb{R}$, where $\mathcal{O}$, the domain of X, is the set of all possible outcomes of the experiment, also called the *sample space*, and $\mathbb{R}$, the *range* of X, is the set of real numbers. If the outcomes are numeric, and represent the observed values of the random variable, then $X: \mathcal{O} \to \mathcal{O}$ is simply the identity function: $X(v) = v$ for all $v \in \mathcal{O}$. The distinction between the outcomes and the value of the random variable is important, as we may want to treat the observed values differently depending on the context, as seen in Example 1.6.

A random variable X is called a *discrete random variable* if it takes on only a finite or countably infinite number of values in its range, whereas X is called a *continuous random variable* if it can take on any value in its range.

Probability Mass Function

If X is discrete, the *probability mass function* of X is defined as

$$f(x) = P(X = x) \qquad \text{for all } x \in \mathbb{R}$$

In other words, the function f gives the probability $P(X = x)$ that the random variable X has the exact value x. The name "probability mass function" intuitively conveys the fact that the probability is concentrated or massed at only discrete values in the range of X, and is zero for all other values. f must also obey the basic rules of probability. That is, f must be non-negative:

$$f(x) \ge 0$$

and the sum of all probabilities should add to 1:

$$\sum_x f(x) = 1$$

Cumulative Distribution Function

For any random variable X, whether discrete or continuous, we can define the *cumulative distribution function (CDF)* $F: \mathbb{R} \to [0,1]$, which gives the probability of observing a value at most some given value x:

$$F(x) = P(X \le x) \qquad \text{for all } -\infty < x < \infty$$

When X is discrete, F is given as

$$F(x) = P(X \le x) = \sum_{u \le x} f(u)$$

and when X is continuous, F is given as

$$F(x) = P(X \le x) = \int_{-\infty}^{x} f(u)\, du$$

Probability Density Function

If X is continuous, its range is the entire set of real numbers $\mathbb{R}$. The probability of any specific value x is only one out of the infinitely many possible values in the range of X, which means that $P(X = x) = 0$ for all $x \in \mathbb{R}$. However, this does not mean that the value x is impossible, because in that case we would conclude that all values are impossible! What it means is that the probability mass is spread so thinly over the range of values that it can be measured only over intervals $[a, b] \subset \mathbb{R}$, rather than at specific points. Thus, instead of the probability mass function, we define the *probability density function*, which specifies the probability that the variable X takes on values in any interval $[a, b] \subset \mathbb{R}$:

$$P\big(X \in [a, b]\big) = \int_a^b f(x)\, dx$$

As before, the density function f must satisfy the basic laws of probability:

$$f(x) \ge 0, \qquad \text{for all } x \in \mathbb{R}$$

and

$$\int_{-\infty}^{\infty} f(x)\, dx = 1$$

PROBABILITY

Read dataset

Get a Summary of data (EDA)

Visualize the data (EDA)

Therefore, there is no excuse for not exploring the data,
To understand the data
To cleanse
To prepare data
For further analysis

Data collection is hard.
Preparing data is even harder.

It does cost time and effort to prepare/cleanse data .
Try running models on unsuitable data!!!.

erroneous conclusion
Is sometime unavoidable because one chooses an incorrect model.
However, coming to an invalid conclusion because we did not explore, prepare and cleanse the data is ALWAYS avoidable.

ESSENTIAL FUNCTIONS WE NEED

– Missing data
— outlier data
  – not all outlier are equal
  -- anomaly detection
    www-users.cs.umn.edu/~banerjee/papers/09/anomaly.pdf

— some rare patterns occur due to randomness and size of sample – *Bonferroni's Principle* and *Bonferroni Correction* can help us reject such patterns which occur due to randomness not because of any underlying physical or other phenomena.
Our brain operates on patterns – the zodiac signs are a classic manifestation of that.

ERRONEOUS DATA

Let us start with a famous example anscombe with a simple linear model...where y= mx+c

M is the slope and c is the intercept, this is called linear because y varies linearly with x. It is easy to find m and c given y and x in R.

```
plot(anscombe$x1,anscombe$y1)
> plot(anscombe$x2,anscombe$y2)
> plot(anscombe$x3,anscombe$y3)
> plot(anscombe$x4,anscombe$y4)
```

```
> lm1<-lm(y1~x1,data=anscombe)
> lm2<-lm(y2~x2,data=anscombe)
> lm3<-lm(y3~x3,data=anscombe)
> lm4<-lm(y4~x4,data=anscombe)
```

LET US RUN A SIMPLE LINEAR MODEL

```
> summary(lm1)

Call:
lm(formula = y1 ~ x1, data = anscombe)

Residuals:
     Min       1Q   Median       3Q      Max
-1.92127 -0.45577 -0.04136  0.70941  1.83882

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   3.0001     1.1247   2.667  0.02573 *
x1            0.5001     0.1179   4.241  0.00217 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.237 on 9 degrees of freedom
Multiple R-squared:  0.6665,    Adjusted R-squared:  0.6295
F-statistic: 17.99 on 1 and 9 DF,  p-value: 0.00217
```

```
> summary(lm3)

Call:
lm(formula = y3 ~ x3, data = anscombe)

Residuals:
    Min      1Q  Median      3Q     Max
-1.1586 -0.6146 -0.2303  0.1540  3.2411

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   3.0025     1.1245   2.670  0.02562 *
x3            0.4997     0.1179   4.239  0.00218 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.236 on 9 degrees of freedom
Multiple R-squared:  0.6663,    Adjusted R-squared:  0.6292
F-statistic: 17.97 on 1 and 9 DF,  p-value: 0.002176
```

```
> summary(lm2)

Call:
lm(formula = y2 ~ x2, data = anscombe)

Residuals:
    Min      1Q  Median      3Q     Max
-1.9009 -0.7609  0.1291  0.9491  1.2691

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)    3.001      1.125   2.667  0.02576 *
x2             0.500      0.118   4.239  0.00218 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.237 on 9 degrees of freedom
Multiple R-squared:  0.6662,    Adjusted R-squared:  0.6292
F-statistic: 17.97 on 1 and 9 DF,  p-value: 0.002179
```

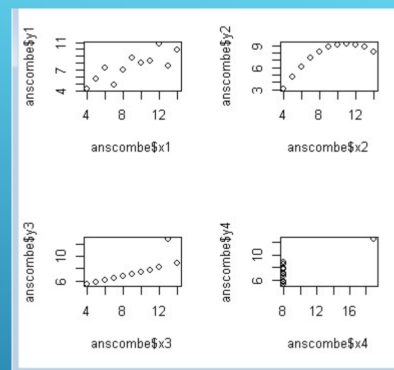
```
> summary(lm4)

Call:
lm(formula = y4 ~ x4, data = anscombe)

Residuals:
   Min     1Q Median     3Q    Max
-1.751 -0.831  0.000  0.809  1.839

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   3.0017     1.1239   2.671  0.02559 *
x4            0.4999     0.1178   4.243  0.00216 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.236 on 9 degrees of freedom
Multiple R-squared:  0.6667,    Adjusted R-squared:  0.6297
F-statistic:    18 on 1 and 9 DF,  p-value: 0.002165
```

STATISTICS

```
plot(anscombe$x1,anscombe$y1)
> plot(anscombe$x2,anscombe$y2)
> plot(anscombe$x3,anscombe$y3)
> plot(anscombe$x4,anscombe$y4)
```

If we rely exclusively on Statistics
We will come to the wrong conclusion.

Visualization sets us straight in very easy terms.



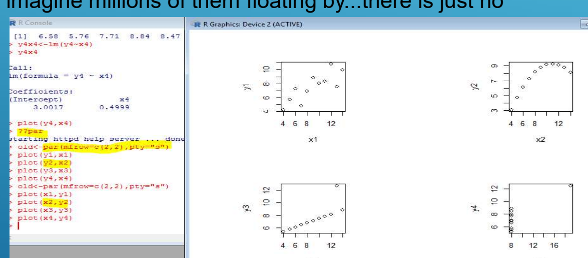
NOW LET US RUN A SIMPLE VISUALIZATION

OUTLIERS

```
> x3
 [1] 10  8 13  9 11 14  6  4 12  7  5
> y3
 [1]  7.46  6.77 12.74  7.11  7.81  8.84  6.08  5.39  8.15  6.42  5.73
> x4
 [1]  8  8  8  8  8  8  8 19  8  8  8
> y4
 [1]  6.58  5.76  7.71  8.84  8.47  7.04  5.25 12.50  5.56  7.91  6.89
> |
```

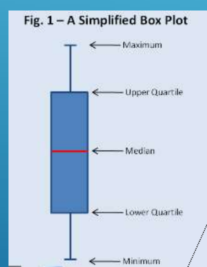
Even though this is a small dataset, it is still taxing to spot the outliers we cannot tell... imagine millions of them floating by...there is just no way to tell.

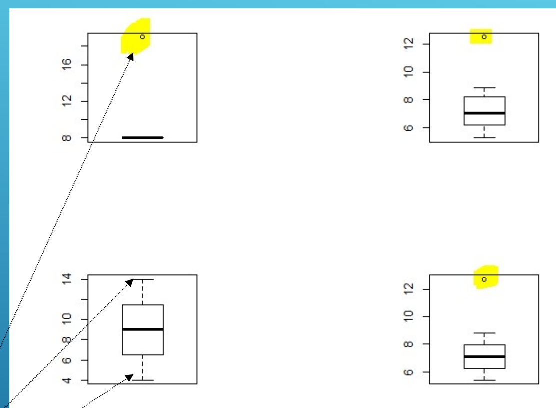
However
plotting gives
It away...
Effortless



OUTLIERS USING BOXPLOT

```
>
> boxplot(x4)
> boxplot(y4)
> boxplot(x3)
> boxplot(y3)
>
```





8 8 8 8 8 8 8 19 8 8 8

10 8 13 9 11 14 6 4 12 7 5

6.58 5.76 7.71 8.84 8.47 7.04 5.25 12.50 5.56 7.91 6.89

7.46 6.77 12.74 7.11 7.81 8.84 6.08 5.39 8.15 6.42 5.73

ANOMALY

Anomaly Detection: A Survey
VARUN CHANDOLA, ARINDAM BANERJEE, and VIPIN KUMAR University of Minnesota
www-users.cs.umn.edu/~banerjee/papers/09/anomaly.pdf

Adding to our vocabulary ...

Input is generally a collection of data instances
(also referred as object, record, point, vector, pattern, event, case, sample, observation, or entity).
Each data instance can be described using a set of attributes
(also referred to as variable, characteristic, feature, field, or dimension).
The attributes can be of different types suchas binary,categorical,or continuous.
Each data instance might consist of only one attribute (univariate) or multiple attributes
(multivariate).

```
> write.csv(anscombe,"ans.csv")
> ans<-read.csv("ans.csv",sep=",",head=T)
> ans
    X x1 x2 x3 x4    y1   y2    y3    y4
1   1 10 10 10  8  8.04 9.14  7.46  6.58
2   2  8  8  8  8  6.95 8.14  6.77  5.76
3   3 13 13 13  8  7.58 8.74 12.74  7.71
4   4  9  9  9  8  8.81 8.77  7.11  8.84
5   5 11 11 11  8  8.33 9.26  7.81  8.47
6   6 14 14 14  8  9.96 8.10  8.84  7.04
7   7  6  6  6  8  7.24 6.13  6.08  5.25
8   8  4  4  4 19  4.26 3.10  5.39 12.50
9   9 12 12 12  8 10.84 9.13  8.15  5.56
10 10  7  7  7  8  4.82 7.26  6.42  7.91
11 11  5  5  5  8  5.68 4.74  5.73  6.89
> 
```

HOW TO READ/WRITE DATA

ARE WE DONE WITH READ/WRITE?

ans$x1==anscombe$x1
table(ans$x1==anscombe$x1)

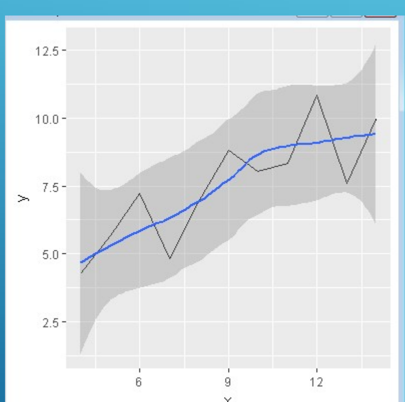
ans$x2==anscombe$x2
table(ans$x1==anscombe$x1)

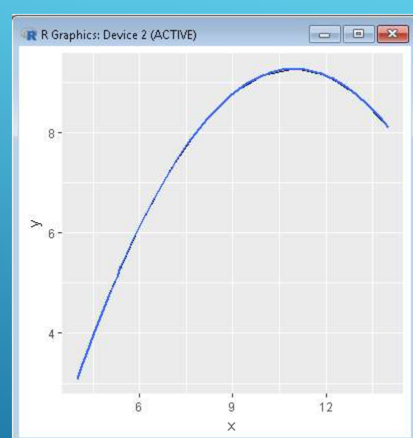
Now let us use R to do all the work for us….

unlist(lapply(1:8,FUN=function(x)table(ans[x+1]==anscombe[x])))

WE MUST ALWAYS VERIFY

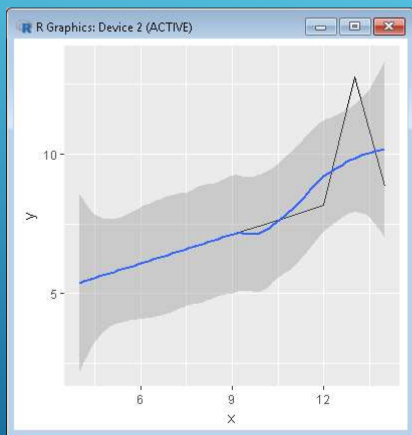
ggplot(data.frame(x=anscombe$x1,y=anscombe$y1),aes(x=x,y=y))+geom_line()+geom_smooth()
ggplot(data.frame(x=anscombe$x2,y=anscombe$y2),aes(x=x,y=y))+geom_line()+geom_smooth()

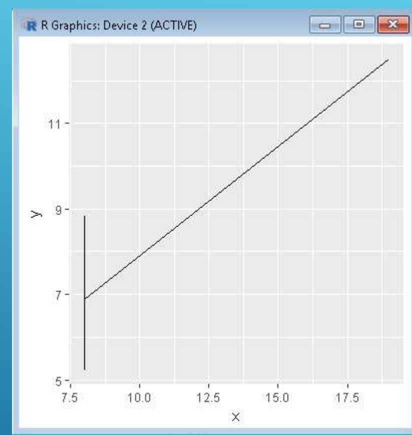




GGPLOT

```
ggplot(data.frame(x=anscombe$x3,y=anscombe$y3),aes(x=x,y=y))+geom_line()+geom_smooth()
ggplot(data.frame(x=anscombe$x4,y=anscombe$y4),aes(x=x,y=y))+geom_line()+geom_smooth()
```





GGPLOT GRAMMAR OF GRAPHICS

1.2 ATTRIBUTES

Attributes may be classified into two main types depending on their domain, that is, depending on the types of values they take on.

Numeric Attributes

A *numeric* attribute is one that has a real-valued or integer-valued domain. For example, `Age` with $domain(\texttt{Age}) = \mathbb{N}$, where $\mathbb{N}$ denotes the set of natural numbers (non-negative integers), is numeric, and so is `petal length` in Table 1.1, with $domain(\texttt{petal length}) = \mathbb{R}^+$ (the set of all positive real numbers). Numeric attributes that take on a finite or countably infinite set of values are called *discrete*, whereas those that can take on any real value are called *continuous*. As a special case of discrete, if an attribute has as its domain the set $\{0, 1\}$, it is called a *binary* attribute. Numeric attributes can be classified further into two types:

- *Interval-scaled*: For these kinds of attributes only differences (addition or subtraction) make sense. For example, attribute `temperature` measured in °C or °F is interval-scaled. If it is 20 °C on one day and 10 °C on the following day, it is meaningful to talk about a temperature drop of 10 °C, but it is not meaningful to say that it is twice as cold as the previous day.
- *Ratio-scaled*: Here one can compute both differences as well as ratios between values. For example, for attribute `Age`, we can say that someone who is 20 years old is twice as old as someone who is 10 years old.

Categorical Attributes

A *categorical* attribute is one that has a set-valued domain composed of a set of symbols. For example, `Sex` and `Education` could be categorical attributes with their domains given as

$$domain(\texttt{Sex}) = \{\texttt{M}, \texttt{F}\}$$
$$domain(\texttt{Education}) = \{\texttt{HighSchool}, \texttt{BS}, \texttt{MS}, \texttt{PhD}\}$$

Categorical attributes may be of two types:

- *Nominal*: The attribute values in the domain are unordered, and thus only equality comparisons are meaningful. That is, we can check only whether the value of the attribute for two given instances is the same or not. For example, `Sex` is a nominal attribute. Also `class` in Table 1.1 is a nominal attribute with $domain(\texttt{class}) = \{\texttt{iris-setosa}, \texttt{iris-versicolor}, \texttt{iris-virginica}\}$.
- *Ordinal*: The attribute values are ordered, and thus both equality comparisons (is one value equal to another?) and inequality comparisons (is one value less than or greater than another?) are allowed, though it may not be possible to quantify the difference between values. For example, `Education` is an ordinal attribute because its domain values are ordered by increasing educational qualification.

FLASHBACK REVIEW AND RECALL

lm – linear model

Summary

Table

Plot

Data.frame

Matrix

List/vector/lapply

R UTILITIES WE HAVE LOOKED AT TODAY