采用

一、判断缺失值

缺失值有两种表示方式：NA和NaN，两者之间的关系是：NaN属于NA，NA不属于NaN。原因是：NaN一般表示数值类型的缺失值，NA表示的数据类型可以多种，比如：数值的缺失值、字符的缺失值等。

1.先用c()函数创建一个向量

```
> x<-c(1,NA,NA,2,3)
> x
[1]  1 NA NA  2  3
```

2.用is.na()和is.nan()函数判断缺失值

```
> is.na(x)
[1] FALSE  TRUE  TRUE FALSE FALSE
> is.nan(x)
[1] FALSE FALSE FALSE FALSE FALSE
```

从结果可以看出：is.na()函数判断出了每个元素是否为缺失值，缺失值返回了真，数值返回了假；is.nan()函数没有判断出缺失值，说明在向量x中不存在NaN这种类型的缺失值，同时也印证了NA不属于NaN。若我们把向量x里的NA改成NaN，结果如下：

```
> x<-c(1,NaN,NaN,2,3)
> x
[1]   1 NaN NaN   2   3
> is.na(x)
[1] FALSE  TRUE  TRUE FALSE FALSE
> is.nan(x)
[1] FALSE  TRUE  TRUE FALSE FALSE
```

二、处理缺失值

```
#1.先用c()函数创建一个包含缺失值的向量
> x<-c(1,NA,NA,2,3)
> x
[1]  1 NA NA  2  3
#2.取向量中不是缺失值的元素，！的意思是取反
> x[!is.na(x)]
[1] 1 2 3

#3.用complete.case()函数取多个向量对应位置都不是缺失值的元素
> x<-c(1,NA,NA,2,3)
> y<-c(NA,"a","b","c","d")
> z<-c(1L,2L,3L,NA,4L)
> w<-complete.cases(x,y,z)
> w
[1] FALSE FALSE FALSE FALSE  TRUE
#返回结果是逻辑向量，其中只有x和y对应位置都不是缺失值的才会返回TRUE，否则返回FALSE。我们用x[
```

```
> x[w]
[1] 3
> y[w]
[1] "d"
> z[w]
[1] 4
```

三、实例

1.加载数据集所在的包

```
> library(datasets)
```

2.用head()函数查看数据集的前6行

```
head(airquality)
  Ozone Solar.R Wind Temp Month Day
1    41     190  7.4   67     5   1
```

```
2    36     118  8.0   72     5   2
3    12     149 12.6   74     5   3
4    18     313 11.5   62     5   4
5    NA      NA 14.3   56     5   5
6    28      NA 14.9   66     5   6
```

3.用complete.cases()函数选择在变量

```
g<-complete.cases(airquality)
> g
  [1]  TRUE  TRUE  TRUE  TRUE FALSE FALSE  TRUE  TRUE  TRUE FALSE
 [11] FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
 [21]  TRUE  TRUE  TRUE  TRUE FALSE FALSE FALSE  TRUE  TRUE  TRUE
 [31]  TRUE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE  TRUE
 [41]  TRUE FALSE FALSE  TRUE FALSE FALSE  TRUE  TRUE  TRUE  TRUE
 [51]  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
 [61] FALSE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE
 [71]  TRUE FALSE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE
 [81]  TRUE  TRUE FALSE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
 [91]  TRUE  TRUE  TRUE  TRUE  TRUE FALSE FALSE FALSE  TRUE  TRUE
[101]  TRUE FALSE FALSE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE
[111]  TRUE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE FALSE  TRUE
[121]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
[131]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
[141]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE
[151]  TRUE  TRUE  TRUE

```

4.选择该数据集不包含缺失值记录的前10行，且每个变量都要

```
> airquality[g,][1:10,]
   Ozone Solar.R Wind Temp Month Day
1     41     190  7.4   67     5   1
2     36     118  8.0   72     5   2
3     12     149 12.6   74     5   3
4     18     313 11.5   62     5   4
7     23     299  8.6   65     5   7
8     19      99 13.8   59     5   8
```

```
9      8      19 20.1   61     5   9
12    16     256  9.7   69     5  12
13    11     290  9.2   66     5  13
14    14     274 10.9   68     5  14
```

四、识别缺失值的模式

```
install.packages("mice")
library(mice)
```

```
#用md.pattern()函数查看数据缺失值的分布
md.pattern(data)
```

五、处理缺失值

1.行删除法

用na.omit()函数删除不完整观测

```
library(datasets)
> data<-head(airquality)
> data
  Ozone Solar.R Wind Temp Month Day
1    41     190  7.4   67     5   1
2    36     118  8.0   72     5   2
3    12     149 12.6   74     5   3
4    18     313 11.5   62     5   4
5    NA      NA 14.3   56     5   5
6    28      NA 14.9   66     5   6
> newdata<-na.omit(data)  #用na.omit()函数删除不完整观测
> newdata
  Ozone Solar.R Wind Temp Month Day
1    41     190  7.4   67     5   1
2    36     118  8.0   72     5   2
3    12     149 12.6   74     5   3
4    18     313 11.5   62     5   4
```

2.多重插补法

多重插补法（MI）是一种基于重复模拟的处理缺失值的方法，多重插补是从一个包含缺失值的数据集中生成一组完整的数据集。每个模拟数据集中，缺失数据将使用蒙特卡洛方法来填补。

```
library(datasets)
data<-head(airquality)
data

data1<-mice(data,m=6)
#用mice()函数从包含缺失数据的数据框开始，进行6重插补，即生成6个完整数据集
#mice()函数的第一个参数data为数据，第二个参数m为要返回的完整数据集的个数

fit<-with(data1,lm(Solar.R~Wind+Temp+Month+Day+Ozone))
fit
#用with()函数依次对6个完整数据集分别应用lm()模型
#结果分别返回6个完整数据集的回归结果

jieguo<-pool(fit)
jieguo
summary(jieguo)
#用pool()函数汇总回归结果

result<-complete(data1,action=3)
result
#选择第三个插补数据集作为结果
```