Z 分数和标准分布

ch04 Z标准分和标准分布

- Z 分数的计算
  - z分数的直观表示
  - z 分数转换为实际测量值

ch04 Z标准分和标准分布

Info

在 NBA 比赛里,解说员总是或说某位球员是矮个子球员. 比如很多人喜欢的斯蒂夫.库里

或许很多人会觉得这很荒谬. 对于统计学和一个故事一样, 需要背景的上下文材料, 统计信息才算完整. 对于解说的这个看法, 的确是有一定理由. 这里我们借用NBA 的身高资料来看看解说员所说的是否有依据.

.

.

1 rows × 5 columns

	name	year_start	year_end	position	hc
	String31	Int64	Int64	String3	Float64
1	Stephen Curry	2010	2018	G	190.5

对于普通人身高, 是比较高的. 但是在 NBA的数据集可能是另外一种情况. 同一个统计数字出现在不同的上下文 里意义不同. 数据的核心是不是数值, 而是位置(排名)关系.

在针对不同试卷和不同科目, 同一个分数的意义不同. 同一个分数 当考试内容简单时会考试者排名可能会靠后, 当 考试内容较难, 会排名靠前.

Z 标准分就是提取位置信息的一种方法.

有了z 分数就可以比较不同数据集下的变异问题.

Z 分数的计算

由于会使用到均数, 所以必须要考虑到符号问题. 因为不同批次抽样获取的样本中个体不同, 所以提取的位置信息也不同. 这一点要注意, 我认为这个地方是理解抽样统计的一个很好的切入点.

在使用总体计算 z 分数, 位置信息是参数, 一旦总体确定, z分数就不会改变.

在分段和分类统计中, 抽取的个体分属于不同的区间或者类型, 在不同类型的集合中一个观察值的位置会有所不同

在随机抽样中, 抽取的个体不同,同一个分数在不同的样本中是和不同的个体在一起, 位置信息就会不同. 这里的位置也可以考虑为一个平均位置, 当样本抽样次数增加, 位置信息会趋近于一个值.

例如以 NBA的所有选手作为一个总体, 那么库里会得到一个 z分数 作为参数

抽取 2000 年以后的选手作为一个样本, 库里会得到在该样本中的 z 分数,这是分段统计

抽取所有后卫选手作为一个样本, 库里会得到在后卫样本中的 z分数. 这是分类统计

如果我们随机抽取 100 名选手的数据, 重复多次, 由于随机性, 每次抽取的队员都有所不同, 同一个身高在不同次的抽样中的 z 分数会有所不同

Z 分数计算公式:

$$z = \frac{\text{原始取值} - \text{均值}}{\text{标准差}}$$

针对总体z分数:

$$z = \frac{X - \mu}{\sigma}$$

针对样本z 分数:

$$z = \frac{X - \bar{X}}{s}$$

其中符号表示:

$$X - \text{原始数据}$$

$$\mu - \text{总体均值}$$

$$\sigma - \text{总体标准差}$$

$$\bar{X} - \text{样本均值}$$

$$s - \text{样本标准差}$$

Example

example1

计算下面样本集合的 z 分数

```
sampla_data1 =
1×112 Matrix{Int64}:
 78  73  72  66  63  71  67  42  62  73  …  12  82  32  92  73  46  68  19  11  64
```

```
sampla_data1=[78 73 72 66 63 71 67 42 62 73 45 22 62 99 73 91 52 37 55 97 91 51 44
23 46 64 97 62 97 31 21 49 93 91 89 46 73 82 55 98 12 56 73 82 37 55 89 83 73 27 83
82 73 46 97 62 57 96 46 55 46 19 13 67 73 26 58 64 32 73 23 94 66 55 91 73 67 73 82
55 64 62 46 39 87 11 99 73 56 73 63 73 91 82 63 33 16 88 19 42 62 91 12 82 32 92 73
46 68 19 11 64]
```

```
z_score (generic function with 1 method)
```

```
begin
# z_zscore 函数接收一个数据集, 返回一个新的函数等待输入单个测量值
    function z_score(data)
            return (X,digits=1)->round((X-mean(data))/std(data),digits=digits)
    end
end
```

```
get_zscore = #1 (generic function with 2 methods)
```

```
get_zscore= z_score(sampla_data1)
```

```
1.1
```

```
get_zscore(87)
```

```
quad =  [11.0, 46.0, 64.0, 82.0, 99.0]
```

```
quad=quantile(sampla_data1[1,:],[0,0.25,0.5, 0.75,1.0]) #求sampla_data1的四分位数
```

```
[-2.1, -0.6, 0.1, 0.9, 1.6]
```

```
get_zscore.(quad)    #四分位数的 z 标准分
```

如果是标准的正态分布, 集合的四分位数转换的*z* 分数的绝对值会呈现对称. 中位数的 z 分数会是 0 .

Example

example 2

使用NBA 统计数据 计算 斯蒂夫库里 和姚明的z分数

NBA 身高数据

https://raw.githubusercontent.com/kuriousrajib/SportsAnalytics/main/player*data*v1.csv

注: 由于姚明的数据里 大学有缺失, 可以补充一个大学名称,否则使用 dropmissing函数之后会查不到数据 具体方法: 编辑器打开 csv, 查找 yao ming 然后添加一个大学名字

	name	year_start	year_end	position	hc
1	"Alex Acker"	2006	2009	"G"	195.58
2	"Quincy Acy"	2013	2018	"F"	200.66
3	"Hassan Adams"	2007	2009	"G"	193.04
4	"Jordan Adams"	2015	2016	"G"	195.58
5	"Steven Adams"	2014	2018	"C"	215.9
6	"Bam Adebayo"	2018	2018	"C-F"	208.28
7	"Jeff Adrien"	2011	2015	"F"	200.66
8	"Arron Afflalo"	2008	2018	"G"	195.58
9	"Maurice Ager"	2007	2011	"G"	195.58
10	"Blake Ahearn"	2008	2012	"G"	187.96
	more				
885	"Luke Ridnour"	2004	2015	"G"	187.96

```
begin
    # preparing processing
    local_server_url="http://127.0.0.1:8080/nba-player-data.csv"
    #remote_server_url="https://raw.githubusercontent.com/kuriousrajib/Sports-Analytics/main/player_data_v1.csv"
    getdata(url)=HTTP.get(url).body
    csv(data)=CSV.read(data,DataFrame)
    change_height_name(df)=rename!(df,[8 => :hc])
    processingdata(url)=getdata(url)|>csv|>dropmissing!|>change_height_name

    sort_by_height(df)=sort(df,[order(:hc, rev=true)])
    topten(df)=first(df,10)
    getheight(data)=data[1,"hc"]

    # processing data
    df=processingdata(local_server_url)
    player=select(df,["name","year_start","year_end","position","hc"])
    top10=player|>sort_by_height|>topten
    curry_data=filter(row -> row.name=="Stephen Curry", player)
    yaoming_data=filter(row -> row.name=="Yao Ming", player)
    select_player_since(year)=filter(row -> row.year_start >=year,player)
    later2000_player=select_player_since(2000)

end
```

```
thread = 2 warning: only found 1 / 11 columns around data row: 3416. Filling remaining columns with `missing`
```

Info

NBA身高数据有三个问题要注意

1. 时间跨度

NBA的身高数据时间跨度较大, 如果绘制时间序列, 会发现平均身高一直在上升. 所以在计算的均值时,从数据表中选取最近一段时间的数据, 比如 2000 年以后的数据

2. 位置

球员的在不同位置身高要求不同, 比如传统意义上后卫(G),需要灵活跑动身高一般都不会太高, 中锋(C),在球场上需要越过对方的高度方向, 同时要为对手制造高度墙,所以身高都比较高. NBA 身高最高的球员达到 **2.31**米. 不同位置的球员比较身高就没有意义. 最好是按照位置来比较身高.

3. 单纯的比较身高没有意义

体育运动始终都是力量和智慧的结合. 总结NBA 身高最高的前 10 位,除了姚明得分较高以外,其他选手的成绩都不算太好

10 rows × 5 columns

	name String31	year_start Int64	year_end Int64	position String3	hc Float64
1	Manute Bol	1986	1995	C	231.14
2	Shawn Bradley	1994	2005	C	228.6
3	Yao Ming	2003	2011	C	228.6
4	Sim Bhullar	2015	2015	C	226.06
5	Chuck Nevitt	1983	1994	C	226.06
6	Mark Eaton	1983	1993	C	223.52

	name String31	year_start Int64	year_end Int64	position String3	hc Float64
7	Priest Lauderdale	1997	1998	C	223.52
8	Randy Breuer	1984	1994	C	220.98
9	Keith Closs	1998	2000	C	220.98
10	Swede Halbrook	1961	1962	C	220.98

```
get_player_zscore = #1 (generic function with 2 methods)
```

```
• get_player_zscore= z_score(later2000_player[:,:hc])
```

```
curry_zscore = -1.1
```

```
• curry_zscore=curry_data|>getheight|>get_player_zscore  #库里z分数
```

```
yaoming_zscore = 3.3
```

```
• yaoming_zscore=yaoming_data|>getheight|>get_player_zscore #姚明 z分数
```

z分数的直观表示

比较上面的数据, 可以看到库里的身高比均值低了 1.1个标准差, 而姚明的 z分数比均值高了 3.3个标准差.

所以说库里是矮个子球员也是可以的, 因为他的身高比均值低. 姚明的身高比均值高出了三个标准差, 属于很少出现的测量值.

当数据转换为 z 分数时, 数据集合 均值的 z 分数为 0, 作为一个分界线, 负值比均值小, 正值比均值大.

z 分数直观的显示了个体得分在集合中的位置

z 分数转换为实际测量值

以总体 z分数公式为例:

$$z = \frac{X - \mu}{\sigma}$$

经过代数变化后可以表示为:

$$X = \sigma * z + \mu$$

也就是知道了均值和标准差, 带入 z 分数就可以得到实际的观察值

```
(199.905, 8.82484)
```

```
• player_mean, player_std=mean_and_std(later2000_player[:,:hc])
```

```
curry_height = 190.19784873068184
```

```
• curry_height=player_std*curry_zscore+player_mean
```

```
yaoming_height = 229.02715437292665
```

```
yaoming_height=player_std*yaoming_zscore+player_mean    #由于 求 z分数时进行保留了位数，所以计算稍有差异
```

```
describe(later2000_player[:,:hc])
```

```
Summary Stats:
Length:         885
Missing Count:  0
Mean:           199.905175
Minimum:        175.260000
1st Quartile:   193.040000
Median:         200.660000
3rd Quartile:   205.740000
Maximum:        228.600000
Type:           Float64
```

```
(175.26, 228.6)
```

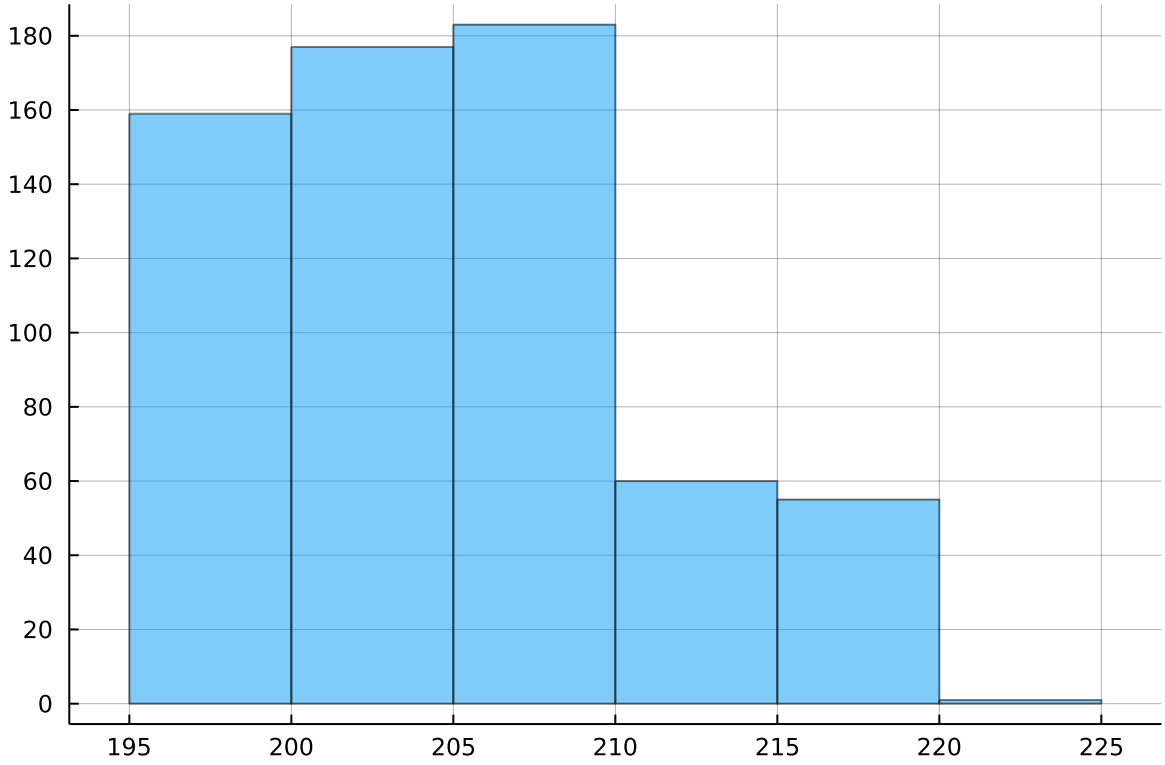
```
minheight,maxheight=extrema(later2000_player[:,:hc])
```

```
Histogram{Int64, 1, Tuple{StepRangeLen{Float64, Base.TwicePrecision{Float64}, Base.TwiceP
edges:
  195.0:5.0:225.0
weights: [159, 177, 183, 60, 55, 1]
closed: right
isdensity: false
```

```
begin
    #以 2为间隔装箱
    bins =195:5:maxheight;
    nbah = fit(Histogram, later2000_player[:,:hc], bins,closed=:right)
end
```

```
885
```

```
length(later2000_player[:,:hc])
```




```
begin
  bar(nbah,label=false,yticks=(0:20:200),xticks=bins,alpha=0.5,gridalpha=0.3)
end
```