**Probability and Hypothesis Testing in R**

**Probability in R**

**for the distributions**

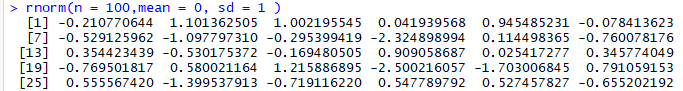
d stands for density

p stands for cdf

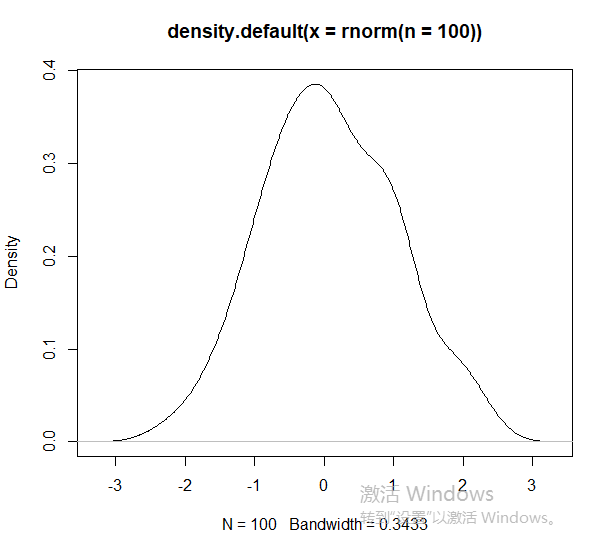
q(cdf) finds the value x with cdf,

r stands for **generating random variable** following distributions

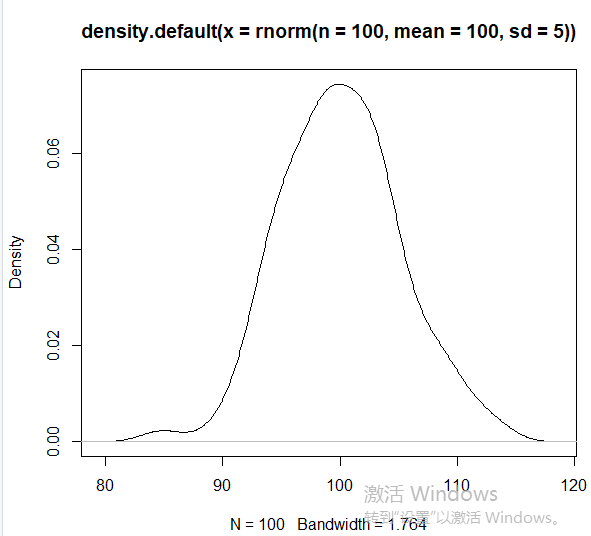
rnorm(n, mean = 0, sd = 1)——产生n个随机数，都服从正态分布，mean和sd都可以自己指定，不一定是normal(0,1)这个标准正态分布。什么都不写的话默认 标准正态分布



画出来：plot(density(rnorm(n=100)))



画个非标准的正态分布：plot(density(rnorm(n=100,mean = 100, sd =5)))



r-functions特别常见，做simulation首先要产生随机数.

如果想产生别的distribution的随机数，也是r+xxdist。比如产生binomial distribution的随机数，就是rbinom.

rbinom(n, size, prob)

n-number of observations. If length(n) > 1, the length is taken to be the number required.

size-number of trials (zero or more).

prob -probability of success on each trial.

我理解的是**做n次实验，每次实验扔硬币size次**，prob是扔一次硬币正面朝上的概率。n指“扔硬币size次”这个实验本身被重复n次。

rbinom给出的随机数有n个，每个随机数就是一次实验中正面朝上的次数。比如

> rbinom(n = 2, size = 100, prob = 0.5)

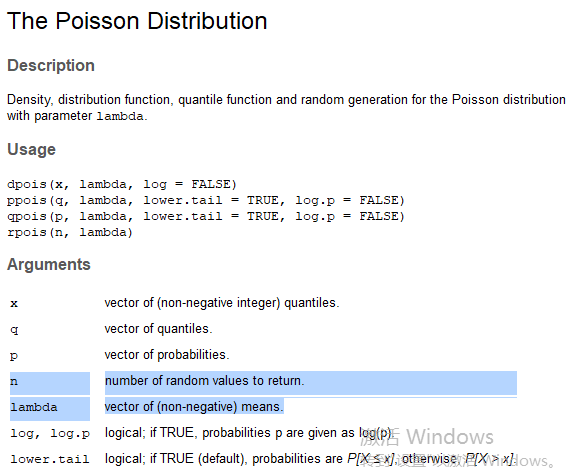
1. 48 51

**还有Bernoulli distribution, Poisson distribution**

**rpois(n, lambda)**

n-number of random values to return.

lambda-vector of (non-negative) means.



**在每种分布前面加上 r d p q就分别是符合该分布的随机数、density（pdf）、cdf和q(cdf) finds the value x with cdf(给定quantile求对应的z-score)。**

**d stands for density，dnorm(x = 0)相当于pdf，即x=0在该分布中的概率**

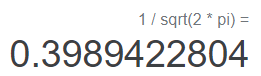
dnorm(x, mean = 0, sd = 1, log = FALSE)

The normal distribution has density f(x) = 1/(√(2 π) σ) e^-((x - μ)^2/(2 σ^2))

> dnorm(x = 0)

1. 0.3989423

代入公式手算验证：



**p stands for cdf - cumulative distribution function，即q<=0在该分布中的概率**

pnorm(q = 0)就是normal distribution中x<=0时的概率，应该是0.5

> pnorm(q = 0) # returns cdf

[1] 0.5

pnorm(q, mean = 0, sd = 1, lower.tail = TRUE, log.p = FALSE)

x, q-vector of quantiles.

**q(cdf) finds the value x with cdf. Quantile function。和p互为反函数，给定cdf/quantile求对应的x。**

qnorm(p, mean = 0, sd = 1, lower.tail = TRUE, log.p = FALSE)

p-vector of probabilities.

log, log.p-logical; if TRUE, probabilities p are given as log(p).

lower.tail-logical; if TRUE (default), probabilities are P[X ≤ x] otherwise, P[X > x].

比如qnorm(p = 0.975) # returns z score for type 1 error 0.05

**右端的critical value对应的quantile是**1-5%/2 = 0.975，则该z-score也就是critical value就是qnorm(p = 0.975) = 1.96

> qnorm(p = 0.975) # returns z score for type 1 error 0.05

1. 1.959964

> qnorm(p = 0.5)

[1] 0

**similar functions——产生T-distribution的pdf,cdf,quantile对应的x和random number generator。**

# <http://www.cyclismo.org/tutorial/R/probability.html>

dt()

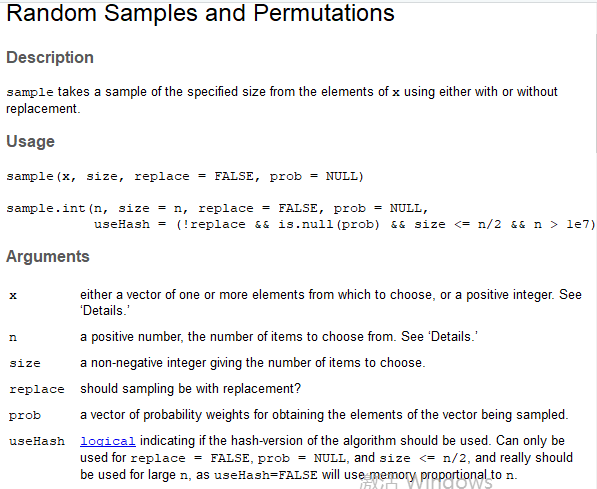
pt()

qt()

rt()

In probability and statistics, **Student's t-distribution** (or simply the t-distribution) is any member of a family of **continuous probability distributions** that arises when **estimating the mean of a normally distributed population** in situations **where the sample size is small and population standard deviation is unknown**.

再加上一个**可以产生随机数的函数-sample( )**



If x has length 1, is numeric (in the sense of is.numeric) and x >= 1, sampling via sample takes place from 1:x.

> sample(x = 5:12, size = 8, replace = F)

[1] 6 11 9 5 8 12 7 10

> sample(x = 5:12, size = 8, replace = T)

1. 5 8 11 12 7 7 10 6

Replace就是产生的随机数之间能不能重复（有无放回）。

**Hypothesis Testing in R**

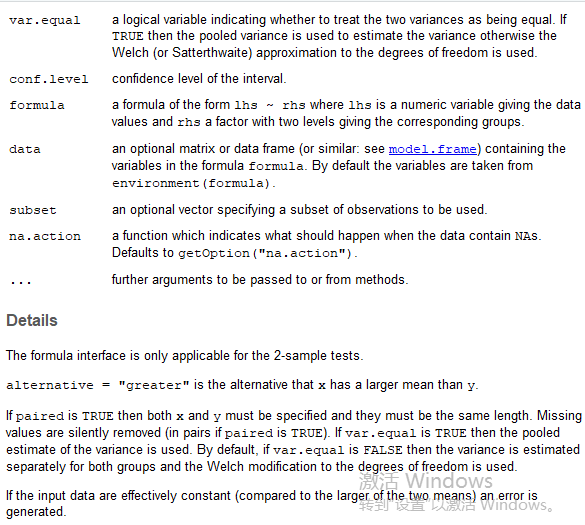
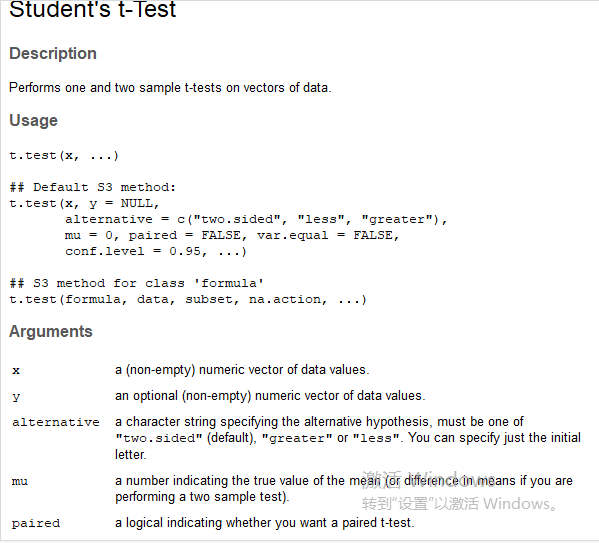
什么时候会用到t test？T test用的最多的就是在A/B test中，公司要做产品迭代（product iteration），要比较新旧版本，都是用A/B test做实验，A/B test的结果拿出来之后，要用t test来算这个结果是否统计显著，是否对metric有明显的提高或下降。

Z test vs. T test

Z test: sample size need to be large enough & variance should be known

T test: more conservative, 当你的sample size没有那么大的时候一般用t test。

When sample size is very large, there is almost no difference between z test and t test. Sample size很大的时候这两个test基本可以互换。

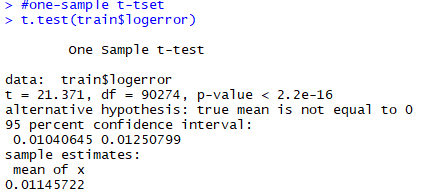


如果只放x一个参数，跑的就是One-sample t-test

如果放一个x一个y，跑的是two-sample t-test

* 比如我们可以先run一个**One-sample t-test**, Null Hypothesis: mean(logerror) = 0

One-sample t-test中x可以是任意column，任何数。这里举例用logerror。



**One sample t.test中的options：**

* alternative = c("two.sided", "less", "greater")——选的是alternative hypothesis是什么，有三种选择对应三种H1：

two.sided: mu != 0;

Less: mu < 0;

Greater: mu > 0.

默认是 two.sided.

例如：

> t.test(train$logerror,**alternative = "two.sided"**)

One Sample t-test

data: train$logerror

t = 21.371, df = 90274, **p-value < 2.2e-16**

**alternative hypothesis: true mean is not equal to 0**

95 percent confidence interval:

0.01040645 0.01250799

sample estimates:

mean of x

0.01145722

结果是**拒绝了H0（mu = 0）**，因为estimate的结果mean of x>0.

如果是less：

> t.test(train$logerror,**alternative = "less"**)

One Sample t-test

data: train$logerror

t = 21.371, df = 90274, **p-value = 1**

**alternative hypothesis: true mean is less than 0**

95 percent confidence interval:

-Inf 0.01233905

sample estimates:

mean of x

0.01145722

p-value变得很大，**不能拒绝H0（mu > = 0）**,因为测出来的结果 mean of x > 0.

如果换成greater：

> t.test(train$logerror,**alternative = "greater"**)

One Sample t-test

data: train$logerror

t = 21.371, df = 90274, **p-value < 2.2e-16**

**alternative hypothesis: true mean is greater than 0**

95 percent confidence interval:

0.01057539 Inf

sample estimates:

mean of x

0.01145722

p-value非常小，**拒绝了H0（mu < = 0）**，因为测出来的结果mean of x > 0.

一般t.test只看p-value就好了，它表明了这个测试结果是否显著。通过看p-value来决定接受还是reject这个Null hypothesis.但你要注意这个p-value所对应的alternative hypothesis是不是给的对，比如上面的例子中我们知道 mean of x 应该是>0的，如果你把alternative这个option设置成了“less”，那就会产生不对的结果——p-value = 1.就不能refuse H0了。选择alternative option的方法就是提前大概知道mean of

x的estimate结果，看它支持alternative hypothesis的”two.sided”,”less”,”greater”哪一种。一般就用默认的two.sided就可以了。这种只要结果不为0都能检测出来significant.

* **Two-sample T-test:** 这里举例看一下 房间计数正确的组和错误的组的logerror是否有很大差异？

两种写法：

1. 首先命名一个逻辑变量，来为room\_cnt是否正确进行分组。这个逻辑变量是roomcnt < bathroomcnt + bedroomcnt，我们把这个逻辑变量命名为group\_room\_wrong. group\_room\_wrong为T的group代表房间计数错误的组（roomcnt < bathroomcnt + bedroomcnt），group\_room\_wrong为F的group代表房间计数正确的组（roomcnt >= bathroomcnt + bedroomcnt）.

train$group\_room\_wrong = train$roomcnt < train$bedroomcnt+train$bathroomcnt

summary(train$group\_room\_wrong)



通过看summary of this logical variable我们知道了房间计数正确的（对应False）有20240个，房间计数错误的（对应T）有70035个。

如何利用这个逻辑变量分组？用[ ]来提取符合条件的target variable

x = train$logerror[train$group\_room\_wrong **==** TRUE]

y = train$logerror[train$group\_room\_wrong **==** FALSE](注意判断语句中的等于要用双等号！！！)

t.test(x,y)

> x = train$logerror[train$group\_room\_wrong == TRUE]

> y = train$logerror[train$group\_room\_wrong == FALSE]

> t.test(x,y)

Welch Two Sample t-test

data: x and y

t = -0.13244, df = 32911, **p-value = 0.8946**

alternative hypothesis: **true difference in means is not equal to 0**

95 percent confidence interval:

-0.002685520 0.002345569

sample estimates:

mean of x mean of y

0.01141911 0.01158909

得出的**p-value非常大**，**无法拒绝null hypothesis H0**(difference between two groups = 0 i.e. **no difference in two gps**)。

说明**两组在logerror上没啥区别**。

1. 用一个formula的形式，也就是用 ~

~前的是我们要看的target variable

~是指我们要针对哪两个组做test

~后的是分组依据，两个组是什么

仍然要用到刚刚命名的逻辑变量train$group\_room\_wrong = train$roomcnt < train$bedroomcnt+train$bathroomcnt

代表对该逻辑判断结果分别为T和F的两组。

t.test(train$logerror ~ train$group\_room\_wrong)

可以得到和method（1）一样的结果

> t.test(train$logerror ~ train$group\_room\_wrong)

Welch Two Sample t-test

data: train$logerror by train$group\_room\_wrong

t = 0.13244, df = 32911, **p-value = 0.8946**

**alternative hypothesis: true difference in means is not equal to 0**

95 percent confidence interval:

-0.002345569 0.002685520

sample estimates:

**mean in group FALSE mean in group TRUE**

0.01158909 0.01141911

**Two sample t.test 中的options**：

* **paired = FALSE**

什么是**paired t-test**？ Same user,给他吃一个药，吃药之前有一个pre的value，吃药之后有一个post的value。即我有两组数据，一组数据是pre的一组数据是post的；也不一定是pre&post,也有可能是同一个人，赛车，连续跑两圈，两圈的数值。比如同一组用户，先给他们看一个旧版本，过几天再给他们看个新版本。但都是**来自于同一个subject的观测值**。对这两个观测值作比较。比较的是group2 - group1 **for the same user** , or post - pre **for the same user.**这时就可以把它变成一个one -sample t.test，因为求的是difference。但是在R code中，还是可以把它写成一个x(pre)和一个y(post),但**要加上option-“paired = TRUE”，这样R就会自动帮你计算每个user的group 2-group1的difference。**

Code looks like this:

t.test(x = user$pre, y = user$post, **paired = TRUE**) 选了paired option为TRUE，R就会自动帮我们配对，第一行的post和pre相减，第二行的post和pre相减，确保是same user。但是注意要用paired = TRUE两个组的数据必须have the same length不然R会报错。

什么是**unpaired t-test**？ Different subjects-随机地将一群人分成两组，给一组看旧版本，给另一组看新版本-two independent populations，或一组给他们吃药，一组没给他们吃药，观察no -medicine value和with-medicine value。也是比较吃没吃药的区别，但是是**比较不同的subjects**。看的是整体，不再是individual。因为观测值并不是来源于同个用户,所以无法一一配对。

t.test(x = group1, y = group2, **paired = FALSE**)

什么时候用paired t-test什么时候用unpaired t-test？取决于数据。在同一个user上采集了两次产生实验组和对照组就用paired，在不同的user上产生实验组和对照组就用unpaired。

用这两种方法做t-test得出的结果会有什么不同？——最大的不同是计算得到的variance会很不一样。具体要使用公式计算。Paired t test最后给出的结果是difference of the means，unpaired t test最后给出的是两个组各自的t test。

* **var.equal =**

Welch t-test (var.equal = FALSE) and student t-test(var.equal = TRUE)

如果两个sample的variance是相等的，就把这个**option设成TRUE，R就会使用pooled variance**（把这两个variance合在一起）（笔记上写道因为t test相比于z test就是population variance unknown，so we may use the pooled variance of the two samples to replace the population variance）

如果var.equal = FALSE, 用的就是各自的variance，分开算。就会做Welch t test.

student t-test assume two samples’ variances are equal, 而 Welch t-test 可以用于两个sample的variance不同的情况。具体来说它们在计算variance的公式方法上有不同。（笔记上有具体描述）

Simplified version：

**With函数**：表示”,”后续的都在“train”这个dataset中操作，不用每个变量名前都加$和所属dataset

with(train, t.test(logerror ~ (roomcnt < bathroomcnt + bedroomcnt)))

即with(train, t.test(logerror ~ group\_room\_wrong))

#下面这段就是根据**公式**手算，手算结果和直接用t.test的结果应该是一样的

train$logerror.abs <- abs(train$logerror)

correct.rmcnt <- subset(train, roomcnt >= bathroomcnt + bedroomcnt)

wrong.rmcnt <- subset(train, roomcnt < bathroomcnt + bedroomcnt)

**stderr** <- sqrt(var(correct.rmcnt$logerror.abs) / dim(correct.rmcnt)[1] +

var(wrong.rmcnt$logerror.abs) / dim(wrong.rmcnt)[1])

**t.score** <- (mean(correct.rmcnt$logerror.abs) - mean(wrong.rmcnt$logerror.abs)) / stderr

**p.val** <- 2 \* (1 - pt(t.score, df = 32425))

2 \* (1 - pnorm(t.score)) # compares with normal distribtion

**T test也可以用在EDA中**

比如想检查 target variable在某个categorical variable上有无区别，（检查某个categorical variable对target variable是否有很大的影响），就可以做一个t test，看不同group之间的target variable是否有显著差异。

**T Test For category variables**

1. if level is not too many, we can use bar chart to check the difference first and then use t test to compare. Take fip for example.

According to data dictionary, fips means Federal Information Processing Standard code

see https://en.wikipedia.org/wiki/FIPS\_county\_code for more details

可以用table( )函数来看这个variable中有几种level，每个level分别对应多少行

**table** uses the cross-classifying factors to build a contingency table of the **counts at each combination of factor levels**.

> table(train$fips)#看每个county中有多少property

6037 6059 6111

58574 24505 7196

# 6037 Los Angeles

# 6059 Orange County

# 6111 Ventura County

str( )函数是给你展示一些该列名下的value的例子：Compactly **display the internal structure of an R object**, a diagnostic function and an alternative to summary (and to some extent, dput).

> str(train$fips)

int [1:90275] 6037 6037 6037 6037 6037 6037 6037 6037 6037 6037 ...

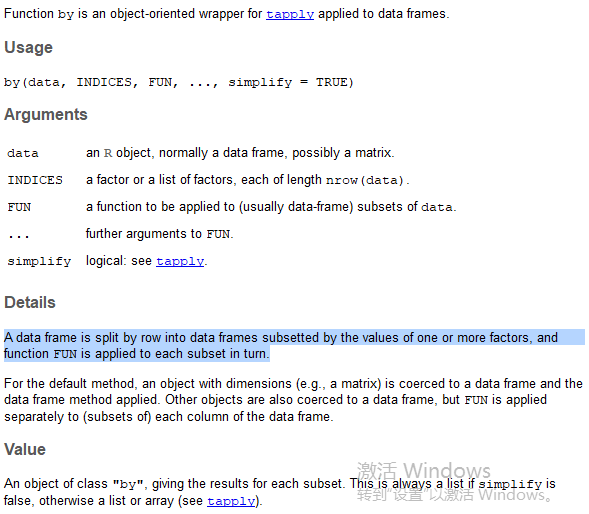
画一下各个county的图：按county（即fip）分类，对每一类都画图

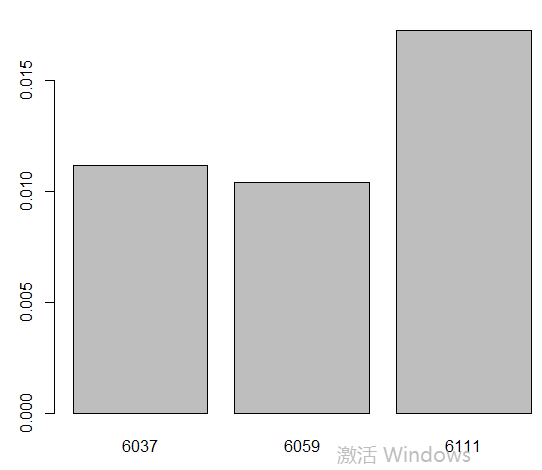
error.fip = by(train, train$fip, function(x) {

return(mean(x$logerror))})

barplot(error.fip)

by函数：





y轴是 mean of logerror。

从图中可以看出，**6111这个county的logerror明显高于其他组**，我们想进一步探究**是否真的在county间的logerror有很大的差异**——做t test。

with(train, t.test(logerror ~ (**fips == '6111'**)))

这里就是通过**判断“fips是否为6111”**来分组，达到“比较**6111和非6111这两组**的目的”

> with(train, t.test(logerror ~ (fips == '6111')))

Welch Two Sample t-test

data: logerror by fips == "6111"

t = -3.251, df = 8559.3, p-value = 0.001154

alternative hypothesis: true difference in means is not equal to 0

95 percent confidence interval:

-0.010098659 -0.002501363

sample estimates:

mean in group FALSE mean in group TRUE

0.01095503 0.01725504

结果发现p-value非常小，reject了null hypothesis，说明fips是6111的county的logerror的确明显大于非6111的组的logerror。差异显著。可能需要对这个county单独（separately）分析，或者需要把它作为predictor放在model中。\

用公式手算：

#this part of the code just conduct the t.test by hand

## calculate the t.score using formula

hs.not.6111 <- subset(train, fips != '6111')

hs.6111 <- subset(train, fips == '6111')

**stderr <- sqrt(var(hs.6111$logerror) / dim(hs.6111)[1] +**

**var(hs.not.6111$logerror) / dim(hs.not.6111)[1])**

t.score <- (mean(hs.not.6111$logerror) - mean(hs.6111$logerror)) / stderr

# we find that the caculated t.score is equal to the result from t.test() function

> t.score

1. -3.251036

如果选择**var.equal = TRUE**:

> # if we use pooled variance

> with(train, t.test(logerror ~ (fips == '6111')), var.equal = TRUE)

Welch Two Sample t-test

data: logerror by fips == "6111"

t = -3.251, df = 8559.3, p-value = 0.001154

alternative hypothesis: true difference in means is not equal to 0

95 percent confidence interval:

-0.010098659 -0.002501363

sample estimates:

mean in group FALSE mean in group TRUE

0.01095503 0.01725504

结果差不多，几乎一样

手算pooled variance: 注意公式,笔记本折页

n1 <- dim(hs.6111)[1]

n2 <- dim(hs.not.6111)[1]

**pooled.stderr <- sqrt((var(hs.6111$logerror) \* (n1 - 1) +**

**var(hs.not.6111$logerror) \* (n2 - 1))**

**/ (n1 + n2 - 2)) \* sqrt(1/n1 + 1/n2)**

# a bit larger than stderr.

# pooled.stderr^2 - stderr^2 = (s2^2 - s1^2) (n2 - n1) / (n1 + n2 - 2)

> pooled.stderr

[1] 0.001979288

> stderr

1. 0.001937847

为什么6111这个组的logerror格外大？可能是这个组内的数据点太少了——看一下fip各组内分别有多少条parcel数据

# Now think about why 6111 have larger logerror, maybe we have too few data points

> num.fip <- by(train, train$fips, function(x){

+ return(length(unique(x$parcelid)))})

> num.fip

train$fips: 6037

[1] 58493

---------------------------------------------------------------

train$fips: 6059

[1] 24467

---------------------------------------------------------------

train$fips: 6111

[1] **7190**

#发现6111对应的数据点相比其他组真的特别少。

train$num.fip <- num.fip[train$fips]

#这一行就是在train这个dataset中创造了新的一列叫做num.fip，把每个fip对应有多少条数据记录在该行的这个列。

# conclusion, we should include boolean feature indicating whether fips = 6111.

# Also if **other feature having level with sparse data**, we should **expect large log.error**.

1. if category variables have too many levels, Take regionidzip for example. Disadvantage of using such variable as it is.

Find similar levels and collapse them, **how to find similar levels** though?

According to data dictionary, regionidcity means City in which the property is located (if any).

Regionidzip——Zip code in which the property is located

table(train$regionidzip)

> table(train$regionidzip)

95982 95983 95984 95985 95986 95987 95988 95989 95991 95992 95993 95994

156 222 258 194 113 125 41 149 18 191 90 47

95995 95996 95997 95998 95999 96000 96001 96002 96003 96004 96005 96006

12 105 226 21 189 272 108 4 116 76 343 323

96007 96008 96009 96010 96012 96013 96014 96015 96016 96017 96018 96019

249 204 53 43 97 207 60 198 187 126 147 74

96020 96021 96022 96023 96024 96025 96026 96027 96028 96029 96030 96034

189 29 154 361 247 287 278 366 287 199 436 1

96037 96038 96039 96040 96042 96043 96044 96045 96046 96047 96048 96049

53 37 3 188 88 145 159 206 248 316 82 273

96050 96058 96072 96083 96086 96087 96088 96090 96091 96092 96095 96097

410 110 89 101 239 50 67 248 187 149 310 82

96100 96101 96102 96103 96104 96105 96106 96107 96109 96110 96111 96113

137 173 174 140 181 50 113 349 187 122 126 141

96116 96117 96119 96120 96121 96122 96123 96124 96125 96126 96127 96128

253 353 30 314 298 434 375 387 261 73 211 239

96129 96133 96134 96135 96136 96137 96148 96149 96150 96151 96152 96159

160 52 115 35 38 65 11 108 139 121 155 250

96160 96161 96162 96163 96169 96170 96171 96172 96173 96174 96180 96181

140 261 198 274 234 116 181 235 226 172 284 163

96183 96185 96186 96190 96192 96193 96197 96201 96203 96204 96206 96207

87 385 494 395 233 593 273 89 172 76 295 32

96208 96210 96212 96213 96215 96216 96217 96218 96220 96221 96222 96225

286 145 265 293 101 50 153 151 161 285 268 130

96226 96228 96229 96230 96234 96236 96237 96238 96239 96240 96241 96242

1 137 250 99 145 449 324 220 362 153 287 394

96244 96245 96246 96247 96265 96267 96268 96270 96271 96273 96275 96278

148 152 181 392 315 278 251 201 224 327 53 102

96280 96282 96284 96289 96291 96292 96293 96294 96295 96296 96320 96321

159 197 244 157 141 260 195 232 337 131 95 152

96322 96323 96324 96325 96326 96327 96329 96330 96336 96337 96338 96339

53 33 101 193 83 132 5 309 311 311 91 278

96341 96342 96346 96349 96351 96352 96354 96355 96356 96361 96364 96366

299 265 338 353 456 300 164 217 313 352 476 129

96368 96369 96370 96371 96373 96374 96375 96377 96378 96379 96383 96384

560 340 415 91 500 315 195 493 440 373 420 311

96385 96387 96389 96393 96395 96398 96401 96403 96410 96411 96412 96414

521 256 474 172 245 264 385 255 229 233 250 188

96415 96420 96424 96426 96433 96434 96436 96437 96438 96446 96447 96449

309 122 341 196 114 30 190 231 155 172 189 381

96450 96451 96452 96464 96465 96467 96469 96473 96474 96475 96478 96479

196 191 210 378 313 1 311 252 108 211 70 173

96480 96485 96486 96488 96489 96490 96492 96494 96496 96497 96500 96505

101 183 279 324 342 126 260 285 186 146 2 512

96506 96507 96508 96510 96513 96514 96515 96517 96522 96523 96524 96525

330 277 150 219 291 133 137 238 279 223 250 187

96531 96533 96939 96940 96941 96943 96946 96947 96948 96951 96952 96954

221 129 213 289 238 122 152 275 187 26 274 511

96956 96957 96958 96959 96961 96962 96963 96964 96965 96966 96967 96969

63 198 296 329 375 624 398 516 323 413 347 299

96971 96973 96974 96975 96978 96979 96980 96981 96982 96983 96985 96986

351 27 824 178 429 46 33 218 374 437 368 35

96987 96989 96990 96993 96995 96996 96998 97001 97003 97004 97005 97006

902 504 388 687 570 640 513 138 159 292 351 177

97007 97008 97016 97018 97020 97021 97023 97024 97025 97026 97027 97035

241 368 322 306 218 141 366 298 175 390 261 306

97037 97039 97040 97041 97043 97047 97048 97050 97051 97052 97059 97063

32 257 110 418 272 286 171 177 105 147 55 210

97064 97065 97066 97067 97068 97078 97079 97081 97083 97084 97089 97091

93 286 111 351 396 474 242 252 464 245 436 498

97094 97097 97098 97099 97101 97104 97106 97107 97108 97109 97111 97113

90 395 59 204 238 226 336 321 6 229 1 134

97116 97118 97119 97298 97316 97317 97318 97319 97323 97324 97328 97329

546 719 17 90 26 403 726 912 106 16 704 760

97330 97331 97344 399675

399 20 85 13

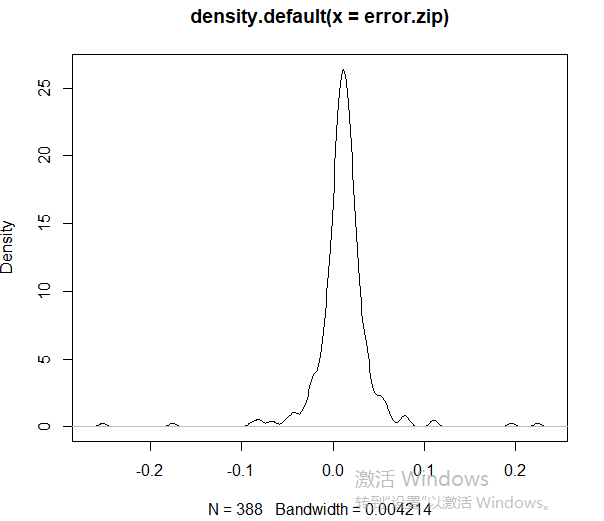
train$regionidzip <- as.character(train$regionidzip)

error.zip <- by(train, train$regionidzip, function(x) {

return(mean(x$logerror))

})

plot(density(error.zip))



train$error.zip <- error.zip[train$regionidzip]

summary(train$error.zip) # seeing NA due to regionidzip is NA.

> summary(train$error.zip) # seeing NA due to regionidzip is NA.

Min. 1st Qu. Median Mean 3rd Qu. Max. NA's

-0.25230 0.00451 0.01163 0.01145 0.01958 0.22395 35

> summary(error.zip)

Min. 1st Qu. Median Mean 3rd Qu. Max.

-0.252300 0.001525 0.011749 0.010973 0.022195 0.223946

# How to impute

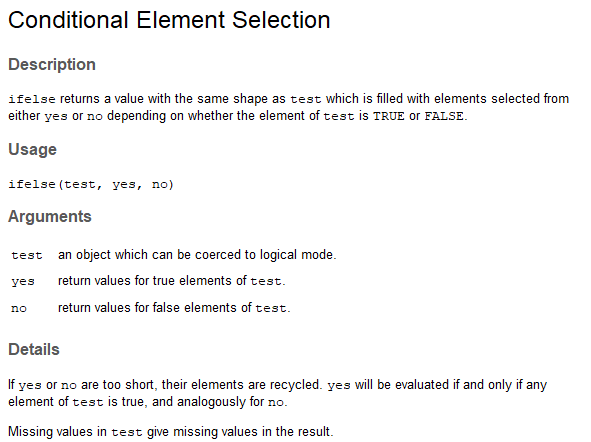
# Set up new level, find per region city what's most likely zip code and more advanced imputation, like libarry(mice)

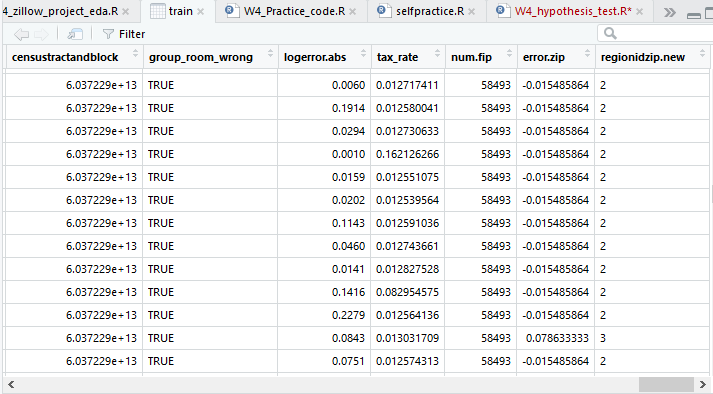
Quantile函数：The generic function quantile produces **sample quantiles corresponding to the given probabilities**. The smallest observation corresponds to a probability of 0 and the largest to a probability of 1.

quantile(x, probs = seq(0, 1, 0.25), na.rm = FALSE,

names = TRUE, type = 7, ...)

Ifelse( )函数：



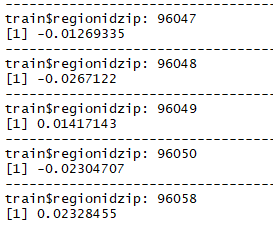
现在的dataset多的几列：  


Which函数：Give the **TRUE indices** of a logical object, allowing for array indices.

代码error.zip <- by(train, train$regionidzip, function(x) {

return(mean(x$logerror))

})中的变量error.zip（不同的regionidzip对应的logerror的平均值）中的几行长这样：



> length(error.zip)

1. 388

> which(error.zip < -0.1)

96226 96329

146 188

其中146和188是满足条件的indices，上面的96226和96329是这两个满足条件的indices对应的regionidzip：

> error.zip[146]

96226

-0.2523

> error.zip[188]

96329

-0.17572

**check these extreme cases** and find out **they also have relative sparse data**.

> error.zip[which(error.zip < -0.1)] # 96226

train$regionidzip

96226 96329

-0.25230 -0.17572

> error.zip[which(error.zip > 0.1)]

train$regionidzip

96207 96323 96951 97331

0.1949625 0.1103636 0.2239462 0.1100900

> dim(subset(train, regionidzip == **96226**))

[1] **1** 67

> dim(subset(train, regionidzip == 96226))[1]

1. 1

> dim(subset(train, regionidzip == **96329**))

[1] **5** 67

> dim(subset(train, regionidzip == **96207**))

[1] **32** 67

> dim(subset(train, regionidzip == **96323**))

1. **33** 67

> dim(subset(train, regionidzip == **96951**))

[1] **26** 67

> dim(subset(train, regionidzip == **97331**))

[1] **20** 67

> dim(train)

1. 90275 67

这些Logerror取极端情况的regionidzip对应的datapoints特别少。

**Correlation**

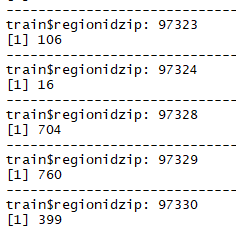
# Assumption: few number of houses for certain region id zip caused logerror large.

# How to verify

num.zip <- by(train, train$regionidzip, function(x) {

return(dim(x)[1])})#计算每个regionidzip对应有多少个property，命名为num.zip

部分结果：



> length(num.zip)

[1] 388

train$num.zip = num.zip[train$regionidzip]#在train dataset中创建这个column

with(train, cor(abs(logerror), num.zip, use = 'pairwise.complete.obs'))

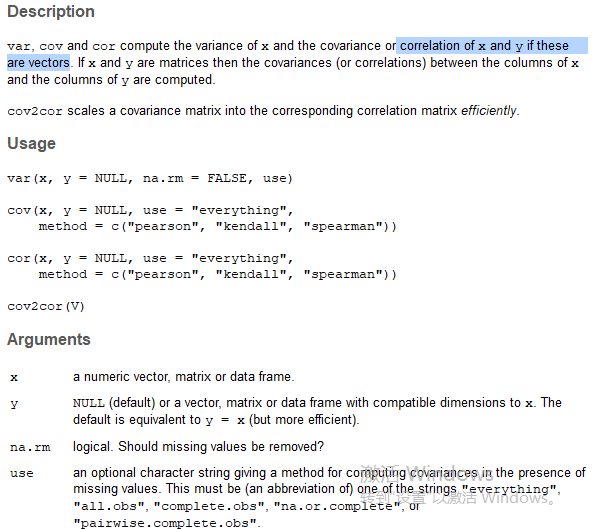
Use Option选择了complete.obs就是指 只用complete的，没有missing value的data

> with(train, **cor**(abs(logerror), num.zip, use = 'pairwise.complete.obs'))

[1] **-0.06620762**

返回的值负相关：说明num.zip（这个zip中对应的房子数量）越多，|logerror|越小。证明了assumption正确。

计算Correlation——cor( ) function：



**Correlation matirx**

在R中一个一个求correlation非常慢，这里有一个求correlation matrix的方法：

安装corrplot这个liabrary

package ‘corrplot’ successfully unpacked and MD5 sums checked

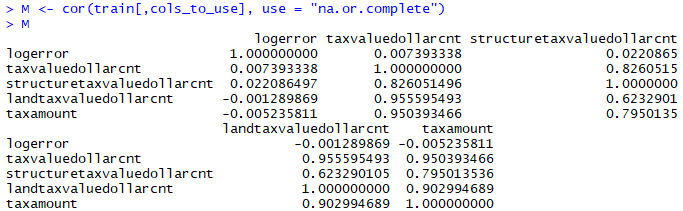
The downloaded binary packages are in

C:\Users\hp\AppData\Local\Temp\RtmpYhpjzc\downloaded\_packages

cols\_to\_use = c('logerror','taxvaluedollarcnt','structuretaxvaluedollarcnt',

'landtaxvaluedollarcnt','taxamount')

M <- cor(train[,cols\_to\_use], use = "na.or.complete")#选取这些列，看它们之间的correlation，na.or.complete指的是If use is "complete.obs" then **missing values are handled by casewise deletion** (and if there are no complete cases, that gives an error). "na.or.complete" is the same unless there are no complete cases, that gives NA——即选取其中不是missing的data。



画出correlation plot，看不同variable之间的correlation

# Continuous variables, take tax amount for example

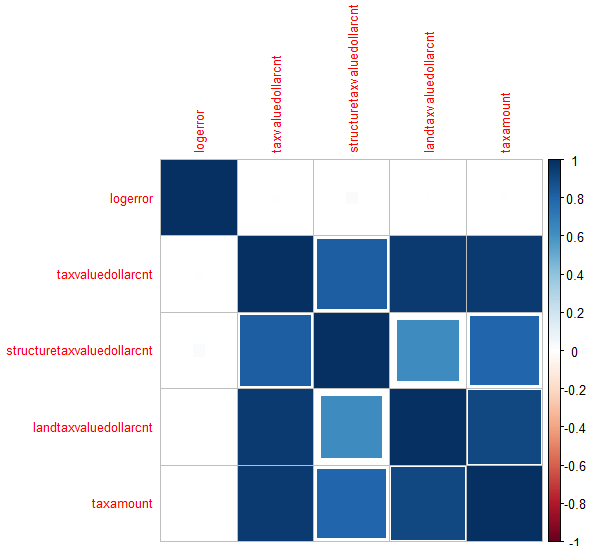
# taxvaluedollarcnt: value to be taxed

# structuretaxvaluedollarcnt: value to be taxed from structure

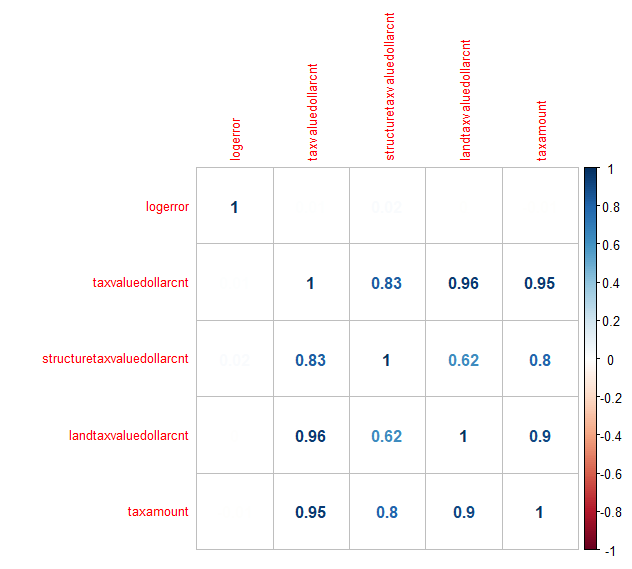
# landtaxvaluedollarcnt: value to be taxed from land

# taxamount: actual paid tax

corrplot(M, method = "square", tl.cex=0.8)



corrplot(M, method = "number", tl.cex=0.8)

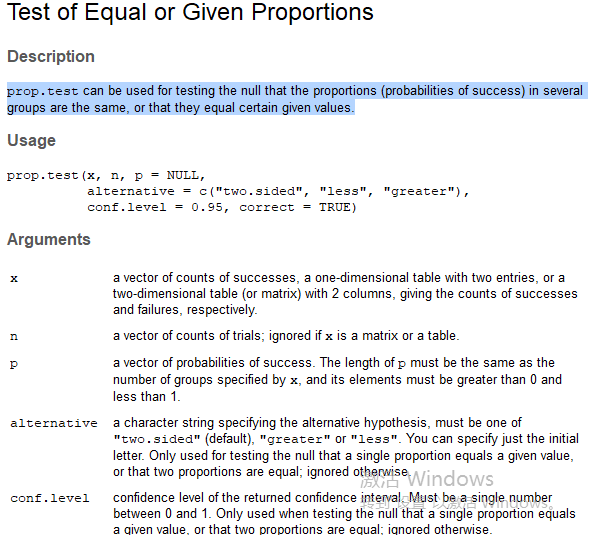


看correlation plot很有用，做proj时如果有很多continuous variables不知道从何下手，可以看一下它们和target variable之间的corrplot，看哪个变量和target之间相关性比较强，就可以重点研究这个variable了。

**Proportion test：函数为 prop.test( )**

这个和t test基本差不多，like z-test，是测group间的比例（null的成功率）是否相同。

**一般p-value < 0.05我们才说有显著差异。**



比如

> #example to learn prop.test:

> x1 = c(50,100)# 100次扔硬币的trial，50次正面/100个人看到，50个人点击。 counts of successses, number of trials

> x2 = c(80,200)#逗号前后的数就是分子和分母。 proportion = x/n

> prop.test(x1,x2)

2-sample test for equality of proportions with continuity correction

data: x1 out of x2

X-squared = 3.1048, df = 1, **p-value = 0.07806**

alternative hypothesis: two.sided

95 percent confidence interval:

-0.01046266 0.26046266

**sample estimates:**

**prop 1 prop 2**

**0.625 0.500**

我们看到p-value 还没有小于0.05，不太有显著差异。

看customer turn? Rate(用户停止使用)等凡是关于比率的，比如点击率等等都可以用proportion test。

在project中的例子：看前两行两个property的**税率**是否有显著差异

summary(with(train, taxamount/taxvaluedollarcnt))

train$tax\_rate = train$taxamount/train$taxvaluedollarcnt

with(train[1:2,], prop.test(taxamount, taxvaluedollarcnt))

Taxamount指的是 交了多少税——The total property tax assessed for that assessment year

taxvaluedollarcnt指的是 要交税的资产价值——The total tax assessed value of the parcel

交的税=资产价值\*税率

> with(train[1:2,], prop.test(taxamount, taxvaluedollarcnt))

**2-sample test for equality of proportions** with continuity correction

data: taxamount out of taxvaluedollarcnt

X-squared = 96.985, df = 1, p-value < 2.2e-16

alternative hypothesis: two.sided

95 percent confidence interval:

-0.002736450 -0.001819017

sample estimates:

prop 1 prop 2

0.01272973 0.01500746

不光可以做两个组的**proportion test**，**可以跑n个组的**，但是每个组中要符合x<=n，不然会报错。

根据

> summary(with(train, taxamount/taxvaluedollarcnt))

Min. 1st Qu. Median Mean 3rd Qu. **Max.** NA's

0.00020 0.01188 0.01261 0.01779 0.01424 **109.54182** 7

中的max值可以知道有些数据有错误，taxamount/taxvaluedollarcnt>1，所以有些sample（row）的数据喂到prop.test函数中会产生如下error：

> with(train, prop.test(taxamount, taxvaluedollarcnt))

**Error** in prop.test(taxamount, taxvaluedollarcnt) :

**elements of 'x' must not be greater than those of 'n'**

> with(train[1:100,], prop.test(taxamount, taxvaluedollarcnt))

**100-sample test for equality of proportions** without continuity correction

data: taxamount out of taxvaluedollarcnt

X-squared = 48037, df = 99, p-value < 2.2e-16

alternative hypothesis: two.sided

sample estimates:

prop 1 prop 2 prop 3 prop 4 prop 5 prop 6 prop 7

0.01272973 0.01500746 0.01267353 0.01271815 0.01276845 0.01452589 0.01267258

prop 8 prop 9 prop 10 prop 11 prop 12 prop 13 prop 14

0.01409168 0.01261254 0.01278853 0.01262187 0.01419503 0.01291072 0.01377059

prop 15 prop 16 prop 17 prop 18 prop 19 prop 20 prop 21

0.01455085 0.01228534 0.01257620 0.01253095 0.01247343 0.01542845 0.01407629

prop 22 prop 23 prop 24 prop 25 prop 26 prop 27 prop 28

0.01256090 0.01266748 0.01336644 0.01286560 0.01265112 0.02072773 0.01386103

prop 29 prop 30 prop 31 prop 32 prop 33 prop 34 prop 35

0.01266083 0.01202964 0.01220709 0.01216014 0.01221280 0.01243269 0.01264047

prop 36 prop 37 prop 38 prop 39 prop 40 prop 41 prop 42

0.01243149 0.01240341 0.01259217 0.01257474 0.01237108 0.01349844 0.01457727

prop 43 prop 44 prop 45 prop 46 prop 47 prop 48 prop 49

0.01166570 0.01422247 0.01253317 0.01463247 0.01386334 0.01342581 0.01258404

prop 50 prop 51 prop 52 prop 53 prop 54 prop 55 prop 56

0.01253084 0.01622448 0.01297724 0.01345381 0.01499085 0.01268246 0.01289338

prop 57 prop 58 prop 59 prop 60 prop 61 prop 62 prop 63

0.01260574 0.01269722 0.01259409 0.01270637 0.01240959 0.01231044 0.01232773

prop 64 prop 65 prop 66 prop 67 prop 68 prop 69 prop 70

0.01278474 0.01524378 0.01271914 0.01259832 0.01277092 0.01373358 0.01356693

prop 71 prop 72 prop 73 prop 74 prop 75 prop 76 prop 77

0.01235089 0.01266138 0.01250827 0.01257496 0.01276482 0.01407657 0.01241599

prop 78 prop 79 prop 80 prop 81 prop 82 prop 83 prop 84

0.07491698 0.01208838 0.01248388 0.01246919 0.01258818 0.01242792 0.01329109

prop 85 prop 86 prop 87 prop 88 prop 89 prop 90 prop 91

0.01288105 0.01270060 0.01250382 0.01265376 0.01232503 0.01282232 0.01321770

prop 92 prop 93 prop 94 prop 95 prop 96 prop 97 prop 98

0.01321770 0.01226608 0.08774469 0.01403987 0.01281775 0.01282622 0.01233305

prop 99 prop 100

0.01268135 0.01278594

计算出的sample estimates和直接跑下面代码得到的结果一模一样

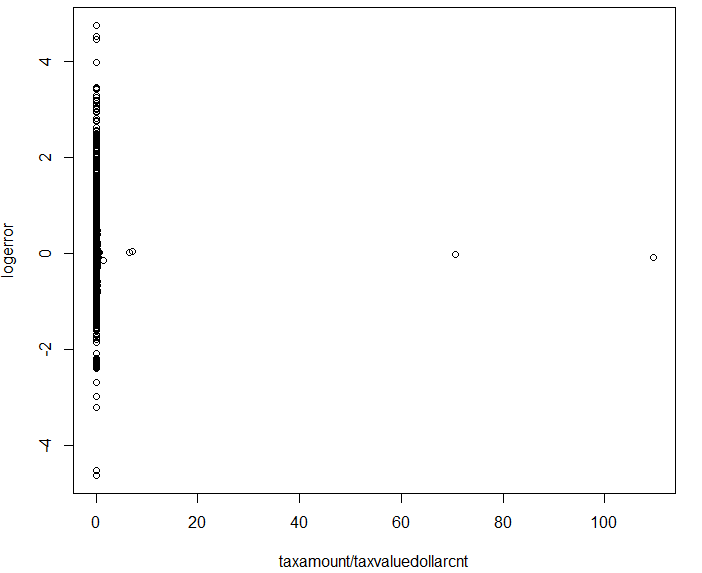
train$tax\_rate = train$taxamount/train$taxvaluedollarcnt

> train$tax\_rate

**More Exploration**

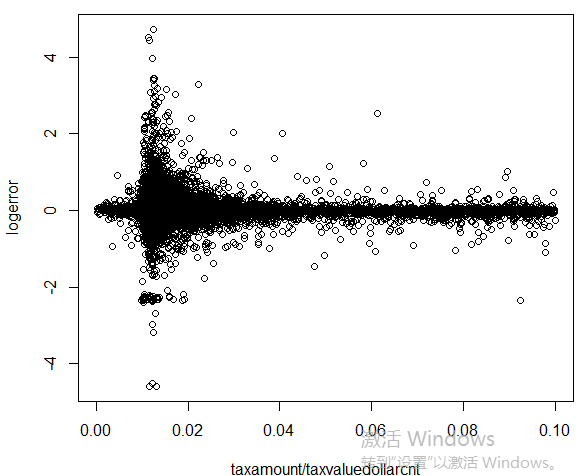
税率和logerror之间的关系的visualization

with(train, plot(taxamount/taxvaluedollarcnt, logerror))



with(subset(train,taxamount/taxvaluedollarcnt <= 0.1),

plot(taxamount/taxvaluedollarcnt, logerror))



看correlation：

> with(subset(train,taxamount/taxvaluedollarcnt <= 0.1),

+ cor(logerror, taxamount/taxvaluedollarcnt, use = 'pairwise.complete.obs'))

[1] -0.04117847

> with(subset(train,taxamount/taxvaluedollarcnt <= 0.1),

+ cor(logerror, taxamount, use = 'pairwise.complete.obs')) # -0.004

[1] -0.004363346

> with(subset(train,taxamount/taxvaluedollarcnt <= 0.1),

+ cor(logerror, taxvaluedollarcnt, use = 'pairwise.complete.obs')) # 0.004

1. 0.00407929

研究新的变量： 关于居住面积——

'calculatedfinishedsquarefeet'——Calculated total finished living area of the home

'lotsizesquarefeet'—— Area of the lot in square feet

> train$living.per <- with(train, calculatedfinishedsquarefeet/lotsizesquarefeet)

> summary(train$living.per)

Min. 1st Qu. Median Mean 3rd Qu. Max. NA's

0.000 0.146 0.218 0.257 0.319 11.497 10483

> with(train, cor(logerror, calculatedfinishedsquarefeet, use = 'pairwise.complete.obs'))

[1] 0.03878407

> with(train, cor(logerror, lotsizesquarefeet, use = 'pairwise.complete.obs'))

[1] 0.00483525

> with(train, cor(logerror, calculatedfinishedsquarefeet/lotsizesquarefeet,

+ use = 'pairwise.complete.obs'))

1. 0.00378775

Cor( )中的option：if **use** has the value "**pairwise.complete.obs**" then the correlation or covariance between each pair of variables is computed using all complete pairs of observations on those variables.