Inferential Statistics

Jan 2020

Simple probability rules

Table 1: Probabilities

Definition	Formula
events with opposite outcomes	$P(A) + P(B) = 1$
both independent events occuring	$P(A\&B) = P(A) * P(B)$
at least 1 occur of mutually exclusive	$P(A \cup B) = P(A) + P(B)$
at least 1 occur non-mutually exclusive	$P(A \cup B) = P(A) + P(B) - P(A\&B)$
conditional probs: prob A occurs if B occured	$P(A \mid B) = \frac{P(A\&B)}{P(B)}$

Bayes' Rule application in medical diagnostics

Accuracy Rates			Bayer's Formula
sensitivity	$P(+ \mid D)$	$+predVal$	$P(D \mid +) = \frac{P(+\mid D)P(D)}{P(+\mid D)P(D)+P(+\mid \sim D)P(\sim D)}$
specificity	$P(- \mid \sim D)$	$-predVal$	$P(\sim D \mid -) = \frac{P(-\mid \sim D)P(\sim D)}{P(-\mid \sim D)P(\sim D)+P(-\mid D)P(D)}$

[1] prevalence: (D) event patient has disease; (~D) event patient doesnt have disease
[2] ($+$) positive test result; ($-$) negative test result

Diagnostic Likelihood Ratios	Pre/Post test odds w/DLR (Post=DLR*Pre)	
$DLR_+ = \frac{P(+\mid D)}{P(+\mid \sim D)}$	data w/disease present	$\frac{P(D\mid +)}{P(\sim D\mid +)} = (\frac{P(+\mid D)}{P(+\mid \sim D)})(\frac{P(D)}{P(\sim D)})$
$DLR_- = \frac{P(-\mid D)}{P(-\mid \sim D)}$	data w/disease not present	$\frac{P(D\mid -)}{P(\sim D\mid -)} = (\frac{P(-\mid D)}{P(-\mid \sim D)}(\frac{P(D)}{P(\sim D)})$

- $+predVal$:prob patients has disease given positive test result
- $-predVal$:prob patient doesnt have disease given negative test value
- DLR_+=large usually
- DLR_-=small usually
- data w/ disease: post > pre
- data wo disease: post < pre

Elements characterizing a distribution

Abbreviations	Meaning
E(X)	expected value of iid (population mean)
p(x)	probability of discrete variable (PMF)
n	sample size
μ	population mean
$\overline{X}$	sample mean
σ^2	population variance
s^2	sample variance
σ	population standard deviation
s	sample standard deviation
λ	mean/variance poisson distribution
t	total monitoriing time poisson
df	degrees of freedom (n-1)

Definition	Formula
population mean	$E(X) = \sum x * p(x)$
variance	$Var(x) = E(x^2) - E(x)^2$
standard deviation	$SD = \sqrt{(variance)}$
variance of sample mean	$Var(\overline{X}) = \frac{\sigma^2}{n}$
standard error of mean	$SEM = \frac{\sigma}{\sqrt{n}}$
standard error of sample mean	$SE(\overline{X}) = \frac{s}{\sqrt{n}}$

[1] increase in variance/SD= increase in spread, vice versa

Useful Rcommands:

- pbinom(x, size=n, prob=0.0, lower.tail=T/F): calcs prob of getting **greater (F)** or **less (T)** than x
- qnorm(percent, mean=mu, sd=sigma, lower.tail=T/F): calc the xVAl on the quantile/percent from the **left (T)** or the **right (F)** end of the distribution curve, also calcs quantile associated with test (th%) when all other parameters left as default
- pnorm(x, mean=mu, sd=sigma, lower.tail=T/F): calcs prob that choose a random variable **less than (T)** or **more than (F)** x or with quantile value gives pVal when alternative mean is **less than (T)** or **greater than (F)** the hypo mean
- rnorm(n, mean=mu, sd=sigma): generate desired number of random normal samples with specific μ and σ
- ppois(x, lamda=rate x t, lower.tail=T/F): calcs prob of getting a value **less than (T)** or **greater than (F)** than x
- binom.test(probability, n)$conf.int: calcs the 95% CI without relying on CLT, better to use when have small n
- poisson.test(x, t)$conf: calcs 95% CI for rate related poisson
- qt(percentage, df): calc t-qunatile when given percent as probability and degrees of freedom

- t.test(V2, V1, diffV, paired=T/F, var.equal=T/F)$conf: calc CI for t test, for paired and independent variables, variations on use specified below. without varConf, then print t,df,pValue,95%CI,sample mean estimate
- qt(q=quantile, df=df, lower,tail=T/F): distribution function of t distribution, quantile would be the t-stat, calc p-value according to the alternative mean being **greater than (F)** or **less than (T)** the mean hypothesis
- power.t.test(power, delta, sd, type, alt): calculates power even with missing values, specify specific need with $power,delta,n etc. more details below
- quantile(vector, c(0.025, 0.975“,”sample(vector, vectorLength * BootstrapNum, replace=TRUE): caclulates the variable values at the indicated th% values, another way to calculate CI
- sample(vector, vLength * bootstrapNum, replace=T): resamples data points from vector by a specified number of time (vector length x bootstrapNum) and saves its in a single longer vector. Used in bootstraping and permutations

Types of distributions

a. Bernoulli's distrbution:

- PMF of Binomial random varialbles: $P(X = x) = \binom{n}{x} p^x (1-p)^{n-x}$
- calculation in R:
  1. choose(n, x) * pˆx * (1-p)ˆ(n-x) + choose(n, x)…..
     - n=total number of trials, x=target/desired number of successes
     - at least ⇒ repeat choose() calc from x 'til n
  2. **pbinom(x-1, size=n, prob=p, lower.tail=TRUE/FALSE)**
     - at least ⇒ lower.tail=FALSE

b. Normal/Gaussian distribution:

- normally distributed random variable characterized as: $X \sim N(\mu, \sigma^2)$ **== sampleSize(mean,variance)**
- standard normal distribution (Z): when $\mu = 0$ and $\sigma = 1$ –> $Z \sim N(0, 1)$
- $Z = \frac{X-\mu}{\sigma}$; $X = \mu + \sigma Z$

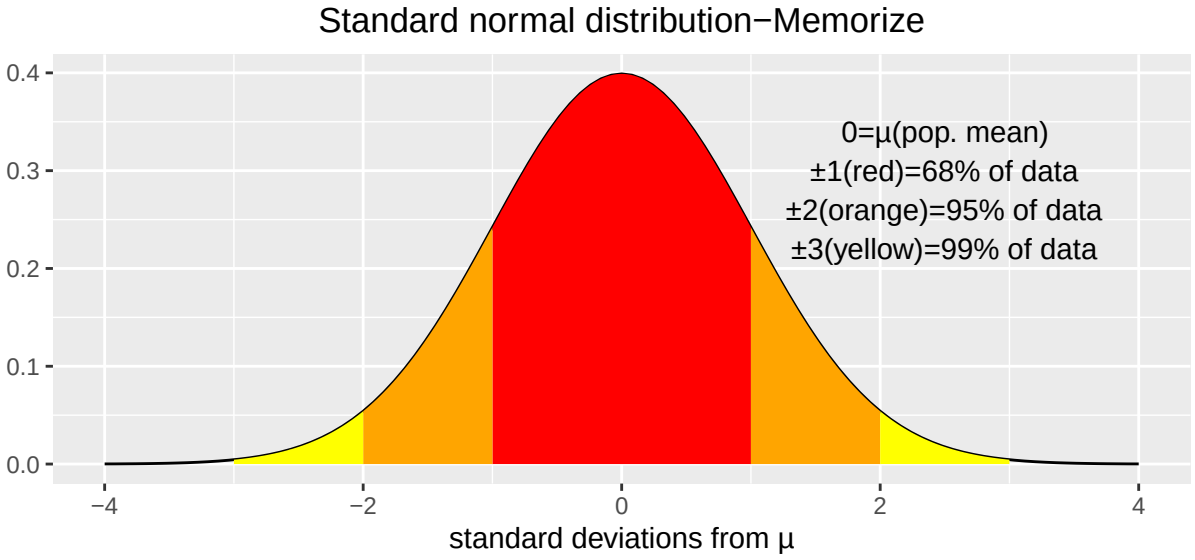


Table 2: SD and percentile of normal distribution

Z(SD)	th %	Z(SD)	th %
-2.330	1.0	1.280	90.0
-1.960	2.5	1.645	95.0
-1.645	5.0	1.960	97.5
-1.280	10.0	2.330	99.0

[1] in R: qnorm(th %) –> Z

- when given μ, σ and q(th%quantile), can calculate the variable of that quantile using:
  - $\mu + \sigma Z$
  - qnorm(q, μ, σ, lower.tail=TRUE)
- to calculate probability of picking a random variable....
  - $Z = \frac{X-\mu}{\sigma} \Rightarrow \pm 1/2/3$ SD(Z) $\Rightarrow \frac{(68/95/99)}{2} = x \Rightarrow 50 \pm$x(less or greater)=probability
  - pnorm(indicateVal or q, μ, σ, lower.tail=TRUE/FALSE)
    - lower.tail=TRUE(less than) or FALSE (greater than)

c. Poisson distribution

- useful for rates: $X \sim Poisson(\lambda t)$ and $\lambda = E[\frac{X}{t}]$
  - λ = rate, t = total time
  - ppois(indicateVal, lamda=rate*t, lower.tail=TRUE/FALSE)
- example: num of ppl at bus stop is Poisson with a mean of 2.5/hr. Survey for 4 hours, prob that 3 or **fewer** people show up for the whole time?
  - Rcode: ppois(3, lambda=2.5*4, lower.tail=TRUE) –> 0.01033605
- Poisson approximates binomial when n is large and p is small
  - $\lambda = np$ rather than rate*t
  - ppois() and pbinom() results in very close probabilities

Central Limit Theorem (CLT)

a. when variables are iid and n is large, they become a standard normal distibution where **N(0,1)**, then: $\bar{X} \sim N(\mu, \frac{\sigma^2}{\sqrt{n}})$ –> convergence to normality

$$\frac{\bar{X} - \mu}{\frac{\sigma}{\sqrt{n}}}$$

b. convergence to normality poor when sample proportion is close to 0 or 1 (in formula replace means with proportions) or when n is small

c. Z-stat under standard normal distribution (N(0,1)): $Z = \frac{\bar{X}-\mu}{\frac{\sigma}{\sqrt{n}}}$

Confidence intervals

- default: 95% for μ is $\bar{X} \pm 2(\frac{\sigma}{\sqrt{n}})$
- CI for normal distribution:
  - calc quantile(Z) for CI (example 90%): $\frac{100-90}{2} + 90 \Rightarrow$ qnorm(0.95)=1.644854
  - calc CI with Z-stat: $\bar{X} \pm ZxSE(\bar{X})$
    - Rcode: $\bar{X}$ + c(-1,1) x qnorm(0.975) x sd(X)/sqrt(length(X))
- CI for binomial: $\bar{p} \pm Z(\sqrt{\frac{p(1-p)}{n}})$
  - Wald: maximize to get largest CI ($p = \frac{1}{2}$): $\bar{p} \pm \frac{1}{\sqrt{n}}$ (formula to use without R)
  - Rcode: $\bar{p} + c(-1,1)xqnorm(0.975)xsqrt(\bar{p}x(1-\bar{p})/n)$
  - Rcode: binom.test(numSuccesses, totalTrials)$conf.int
- CI for poisson: $\hat{\lambda} \pm Z\sqrt{\frac{\hat{\lambda}}{t}}$ when $\hat{\lambda} = \frac{X}{t}$
  - Rcode: numSuccesses/totalTime + c(-1,1) x qnorm(0.975) x sqrt((numSuccess/totalTime)/totalTime)
  - Rcode: poisson.test(numSuccesses, totalTime)$conf

t Confidence Intervals

a. Student t-test (depends on small n): $t = \frac{\overline{x}-\mu}{\frac{s}{\sqrt{n}}}$

- t-quantile in R: qt(percent, df)
- t distribution (larger tails, lower mean peak) –> increase n –> t-intervals become Z-intervals (higher mean peak and smaller tails)

b. CI with t-stat: $\overline{X} \pm t_{(n-1)} * SE(\overline{X})$

c. CI for paired observations (paired t-Test):

- $\overline{X}$= difference between the observations (stored as a vector)
- Rcode: mean($\overline{X}$) + c(-1, 1) x qt(0.975, df) x sd($\overline{X}$)/sqrt(n)
- Rcode: t.test(diffVector)$conf.int
- Rcode: t.test(Ob2Vector, Ob1Vector, paired=TRUE)$conf.int
- Rcode(paired observations in single vector): t.test(ObVector ~ I(relevel(categoryVector, 2ndOb-Category)), paired=TRUE, data=dataFrame)$conf.test

d. CI for 2 independent variables with pooled variance (sp)

- $sp = \sqrt{(\frac{(df_x s_x^2)+(df_y s_y^2)}{n_x+n_y-2})}$; $\mu_y - \mu_x \pm t_{n_x+n_y-2} * sp * \sqrt{(\frac{1}{n_x} + \frac{1}{n_y})}$
- Rcode:

1) sp <- nx(times)sxˆ2 + ny(times)syˆ2
2) ns <- nx+ny-2
3) sp <- sqrt(sp/ns) OR sp <- sqrt((dfx*var(VectorX)+dfy*var(VectorY))/18)

4) muy - mux + c(-1,1) * qt(0.975, ns) * sp*sqrt(1/nx+1/ny)

when interval ranges from neg. to pos. (aka contain 0), cant rule out that means of 2 groups are equal

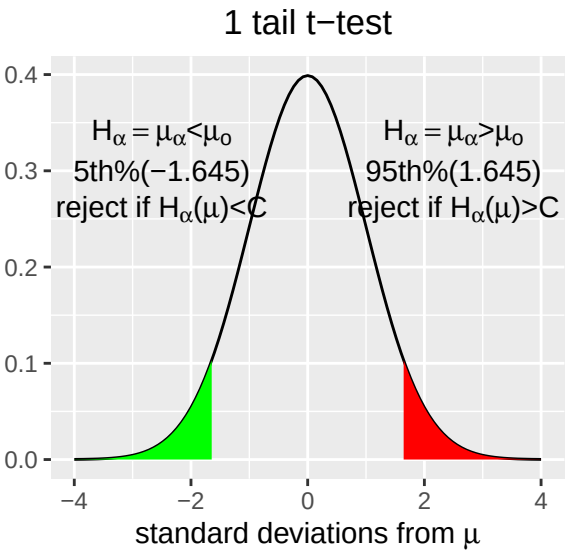
- Rcode: t.test(Vector2, Vector1, paired=FALSE, var.equal=TRUE)$conf

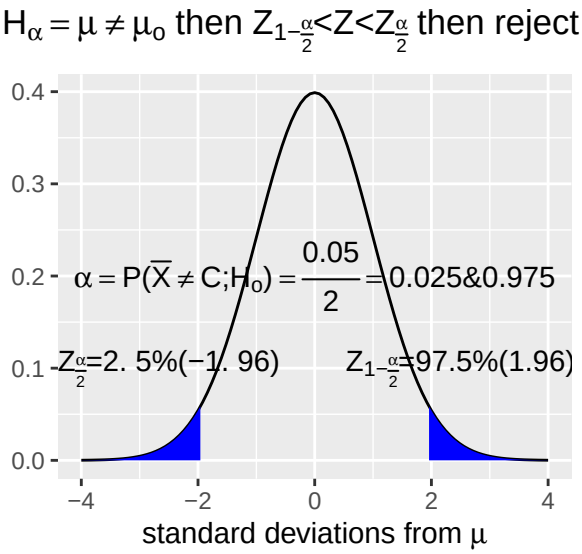
e. CI for 2 independent variables with unequal variance

- $df = \frac{(\frac{s_x^2}{n_x}+\frac{s_y^2}{n_y})^2}{\frac{(\frac{s_x^2}{n_x})^2}{n_x-1}+\frac{(\frac{s_y^2}{n_y})^2}{n_y-1}}$; $SE = \sqrt{(\frac{s_x^2}{n_x}+\frac{s_y^2}{n_y})}$; $\mu_y - \mu_x \pm t_{df} * SE$
- Rcode:
  1) num <- (sx^2/nx + sy$^{2/ny)}$2
  2) den <- sx$^{4/nx}$2/nx-1 + sy$^{4/ny}$2/ny-1
  3) df <- num/def
  4) muy-mux + c(-1,1) x qt(0.975, df) x sqrt(sx$^{2/nx+sy}$2/ny)
- Rcode: t.test(Vector2, Vector1, paired=FALSE, var.equal=FALSE)$conf

Hypothesis Testing

- type I error: reject true H_o; $\alpha = 0.05$
- $C = \mu_{Ho} + 1.645 * \frac{s}{\sqrt{n}}$; $Z = \frac{\overline{X}-\mu_{Ho}}{\frac{s}{\sqrt{n}}}$
- when $H_\alpha : \mu > \mu_o$ then.....
  - $H_o : \mu = C$ and $H_\alpha : \mu > C$
  - OR use Z (for large n): if test statistic $Z > Z_{1-\alpha}(1.645)$ then reject H_o
- when $H_\alpha : \mu < \mu_o$ then $Z < Z_\alpha(-1.645)$ then reject H_o if thats the case





if $\mu \geq C$ (or $\mu \leq C$), then only 5% chance a random draw from distribution is larger than C (or less than C)

if observed mean fell in shaded red/green area, then reject the true $H_o : \mu = \mu_o$ with probability of 5%

- general rule of rejection(H_o): $\frac{(\overline{X}-\mu_o)\sqrt{n}}{s} > Z_{1-\alpha}$
- with small n then calc t rather than Z
  - $t = \frac{\overline{X}-\mu_{Ho}}{\frac{s}{\sqrt{n}}}$
  - $\alpha = qt(0.95, df)$
  - $\overline{X} = \frac{t*sd(\mu_y-\mu_x)}{\sqrt{n}}$
  - t-statistic indicated the number of estimated std error btw H_α and H_o means
  - one sided t test: $t > \alpha$ reject H_o or $t < \alpha$ fail to reject H_o
  - 2 sided t test: $H_\alpha : \mu \neq \mu_o$, reject H_o if qt(0.975, df) < t < qt(0.025, df)
  - t-test in R:
    - Rcode: t.test(V2-V1) OR t.test(V2, V1, paired=TRUE), print: t/df/p-value/95% CI and sample estimate mean of $\overline{X}$
    - with CI, if not contain 0 (μ_o) then can safely reject hypothesis (or making Type I error), if contail 0, then fail to reject H_o

P values

p-value (under the null hypothesis) states how extreme t-values is towards the alternative hypothesis based on observed data. Its also the smallest alpha value at which to reject the H_o

- Rcode: qt(q=quantile, df=df, lower,tail=T/F) & pnorm(q=quantile, lower.tail=T/F)
  - **lower.tail=FALSE:** $H_\alpha : \mu > \mu_o$ **($\overline{X} > quantile$)**
  - **lower.tail=TRUE:** $H_\alpha : \mu < \mu_o$ **($\overline{X} \leq quantile$)**
  - when value less than 0.05 –> significant (if H_o true then see large t-stat at 1%)
- p-value in 2-sided test
  - when have 2 possible outcomes, must calculate p-values for $H_\alpha : \mu > \mu_o$ and $H_\alpha : \mu < \mu_o$ at $\alpha = 0.05$, then multiply the smaller of the two by 2 which then is the official p-value
  - pbinom(x, size=n, prob=p, lower.tail=TRUE)=a
  - pbinom(x, size=n, prob=p, lower.tail=FALSE)=b
  - 2*(a or b)=p-value
- p-value in poisson distribution
  - $H_o : \lambda = \alpha * t$ and $H_\alpha : \lambda > \alpha$
  - α = rate (benchmark), x= less/greater (TRUE/FALSE) than variable want to see for specified rate
  - Rcode: ppois(x, lambda=rate x t, lower.tail=T/F) –> p-value

Power

- type II error: accept false H_o
- β = prob of Type II error
- 1-β = prob of rejecting false H_o aka power
- power is the prob $\mu > Z_{1-\alpha}$
- when thinking of the 2 different hypothesis as distributions: $H_o : \overline{X} \sim N(\mu_o, \frac{\sigma^2}{n})$ and $H_\alpha : \overline{X} \sim N(\mu_\alpha, \frac{\sigma^2}{n})$
  - power determine how much of H_α distribution is the right of $Z_{1-\alpha}$ in H_o

** the further to the right, the greater the $effectsize = \frac{\Delta}{\sigma}$ when $\Delta = \mu_\alpha - \mu o$**

- good to know:
  - $\uparrow (\mu_\alpha - \mu o) \Rightarrow \uparrow (1-\beta)$ when $H_\alpha : \mu > \mu_o$
  - $\downarrow (\mu_\alpha - \mu_o) \Rightarrow (1-\beta) = \alpha$
  - $\uparrow \mu_\alpha$ & $n \Rightarrow \uparrow (1-\beta)$
  - $\uparrow \sigma \Rightarrow \downarrow (1-\beta)$
  - $\uparrow \alpha \Rightarrow \uparrow (1-\beta)$
- Rcode: pnorm(q=μ_o + qnorm(0.95), mean=μ_α, lower.tail=T/F) $\Rightarrow$ power(%)
- power (%) indicates the probability (%) of rejecting H_o when the true value of μ is μ_α
- if $H_\alpha : \mu \neq \mu_o$ then use qnorm(0.975)(FALSE) or qnorm(0.025)(FALSE), depending on the direction of μ_α
- when solving for power really only need: $\frac{\sqrt{n}(\mu_\alpha - \mu_o}{\sigma}$
- calc power with either t or Z stat: $1-\beta = P(\overline{X} > \mu_o + Z_{1-\alpha}(\frac{\sigma}{\sqrt{n}}))$; $1-\beta = P(\frac{\overline{X}-\mu_o}{\frac{s}{\sqrt{n}}} > t_{1-\alpha,df})$
- When delta(Δ) positive, $H_\alpha : \mu > \mu_o$
  - Rcode: power.t.test(n=n, delta=delta/sigma, sd=sigma, type="one.sample", alt="one.sided")$power
  - Rcode: power.t.test(power=1-beta, delta=delta/sigma, type="one.sample", alt="one.sided")$n
  - Rcode: power.t.test(power=power, n=n, sd=sigma, type="one.sample", alt="one.sided")$delta
    - power and n values remains the same so long as the effect size (delta/SD) remains constant

Multiple testing

Abbreviations	Meaning
m	number of test
V	number of falsely declared significants
R	number of true H_o
μ_o	number of test declared significant
$E(\frac{V}{R})$	false discoveries rate (FDR)
$E(\frac{V}{\mu_o})$	false positive rate, simi to Type I error
$P(V \geq 1) < \alpha$	prob of at least 1 false positive being detected, aka family wise error rate (FWER)

no p-value correction: $m * \alpha$ = number of anticipated false positives

Multi test correction (independent tests):

- Bonferroni Correction:
  - control FWER
  - calc: $\alpha_{FWER} = \frac{\alpha}{m}$
  - only accept: $p < \alpha_{FWER}$
- Benjamini-Hochberg (BH) Method:
  - control FDR
  - p-value compared to a value dependeing on its ranking
  - set FDR at α –> calc p-values –> order from smallest to largest (i) –> result significant when $p_i \leq \frac{\alpha * i}{m}$
- adjust p-value approach:
  - check amount of false positives in R: sum(pValueVector < 0.05)
  - adjusting pvalue in R: p.adjust(pValueVector, method="bonferroni/BH")

Resampling

1) Bootstraping (non-parametric):
   - use observed data to construct estimated population distribution using randm sampling with replacement, then use distribution to estimate stat distribution
   - applifying original observed data into a matrix:
     - n <- length(vector)
     - B <- bootstrap#
     - sam <- sample(vector, n*B, replace=TRUE)
     - resampled <- matrix(sam, B, n)
   - using matrix to create estimated stat of population:
     - meds <- apply(resampled, 1, median)
     - sd(meds)
     - quantile(meds, c(0.025, 0.975)) –> CI
     - Bootstrap package in R by broom
2) Permutation testing:
   - to determine if data stats are independent of its group labels via sampling and permutation
   - Rcode:
     - target <- vector w/data points of interest of multiple groups
     - group <- vector w/ group labels of all groups of interest
     - testStat <- function(w, g) mean(w[g=="X"]) - mean(g[=="Y"]
       - can use another stat rathern than mean
     - observedStat <- testStat(target, group)
     - permutations <- sapply(1:#maxTimesDesire, function(i) testStat(target, sample(group)))
     - mean(permutations > obseredStat)
       - examine number of times got larger simulated stat compared to observed stat
       - if "0" then p-value=0, so data was independent of group label