Luke Larter EDA Assignment 2

library(tidyverse)

## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.3.3 v purrr 0.3.4  
## v tibble 3.0.5 v dplyr 1.0.3  
## v tidyr 1.1.2 v stringr 1.4.0  
## v readr 1.4.0 v forcats 0.5.0

## -- Conflicts ------------------------------------------ tidyverse\_conflicts() --  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

library(mosaic)

## Warning: package 'mosaic' was built under R version 4.0.4

## Registered S3 method overwritten by 'mosaic':  
## method from   
## fortify.SpatialPolygonsDataFrame ggplot2

##   
## The 'mosaic' package masks several functions from core packages in order to add   
## additional features. The original behavior of these functions should not be affected by this.

##   
## Attaching package: 'mosaic'

## The following object is masked from 'package:Matrix':  
##   
## mean

## The following objects are masked from 'package:dplyr':  
##   
## count, do, tally

## The following object is masked from 'package:purrr':  
##   
## cross

## The following object is masked from 'package:ggplot2':  
##   
## stat

## The following objects are masked from 'package:stats':  
##   
## binom.test, cor, cor.test, cov, fivenum, IQR, median, prop.test,  
## quantile, sd, t.test, var

## The following objects are masked from 'package:base':  
##   
## max, mean, min, prod, range, sample, sum

#Challenge 1

movies=as\_tibble(read.csv("IMDB-movies.csv"))  
head(movies)

## # A tibble: 6 x 10  
## tconst titleType primaryTitle startYear runtimeMinutes genres averageRating  
## <chr> <chr> <chr> <int> <int> <chr> <dbl>  
## 1 tt000~ movie Dante's Inf~ 1911 68 Adven~ 7   
## 2 tt000~ movie FantÃ´mas: ~ 1913 54 Crime~ 7   
## 3 tt000~ movie Fantomas: T~ 1913 61 Crime~ 7   
## 4 tt000~ movie FantÃ´mas: ~ 1913 90 Crime~ 7   
## 5 tt000~ movie The Student~ 1913 85 Drama~ 6.5  
## 6 tt000~ movie The Avengin~ 1914 78 Crime~ 6.5  
## # ... with 3 more variables: numVotes <int>, nconst <chr>, director <chr>

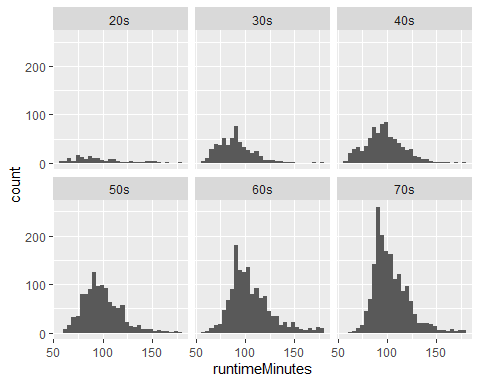
#get required subset  
sub=filter(movies, (startYear %in% (1920:1979)) & (runtimeMinutes %in% (60:180)))

#get decade column  
sub=sub %>% mutate("decade"=case\_when(  
 sub$startYear %in% (1920:1929) ~ "20s",  
 sub$startYear %in% (1930:1939) ~ "30s",  
 sub$startYear %in% (1940:1949) ~ "40s",  
 sub$startYear %in% (1950:1959) ~ "50s",  
 sub$startYear %in% (1960:1969) ~ "60s",  
 sub$startYear %in% (1970:1979) ~ "70s"))  
  
head(sub)

## # A tibble: 6 x 11  
## tconst titleType primaryTitle startYear runtimeMinutes genres averageRating  
## <chr> <chr> <chr> <int> <int> <chr> <dbl>  
## 1 tt001~ movie The Cabinet~ 1920 76 Fanta~ 8.1  
## 2 tt001~ movie Leaves From~ 1920 167 Drama 6.7  
## 3 tt001~ movie Dr. Jekyll ~ 1920 82 Drama~ 7   
## 4 tt001~ movie The Golem 1920 76 Fanta~ 7.2  
## 5 tt001~ movie The Last of~ 1920 73 Actio~ 6.7  
## 6 tt001~ movie The Mark of~ 1920 107 Adven~ 7.1  
## # ... with 4 more variables: numVotes <int>, nconst <chr>, director <chr>,  
## # decade <chr>

#plot runtime histograms  
p=ggplot(sub, aes(x=runtimeMinutes)) + geom\_histogram() + facet\_wrap(~decade)  
p

## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.



#get pop means and sds by decade  
results <- summarize(  
 group\_by(sub, decade),  
 pop\_mean = mean(runtimeMinutes, na.rm = TRUE),  
 pop\_sd = sqrt(sum((runtimeMinutes - mean(runtimeMinutes))^2)/(n())),  
 pop\_n=n()   
)  
  
results

## # A tibble: 6 x 4  
## decade pop\_mean pop\_sd pop\_n  
## \* <chr> <dbl> <dbl> <int>  
## 1 20s 96.3 26.1 152  
## 2 30s 90.3 17.3 530  
## 3 40s 97.2 19.1 782  
## 4 50s 98.9 19.2 1081  
## 5 60s 106. 21.2 1386  
## 6 70s 104. 18.0 1720

#Get SE for sample of 100 movies from each decade  
  
n=100  
  
samp\_dist <- summarize(  
 group\_by(sub, decade),  
 samp=sample(runtimeMinutes, n, replace= FALSE),  
 samp\_mean = mean(samp, na.rm = TRUE),  
 samp\_sd = sd(samp),  
 samp\_se = sd(samp)/sqrt(n)  
)

## `summarise()` has grouped output by 'decade'. You can override using the `.groups` argument.

#need this to remove duplicate rows due to using sample() in summarize:  
samp\_dist=samp\_dist[!duplicated(samp\_dist$decade),][,-2]   
  
samp\_dist

## # A tibble: 6 x 4  
## # Groups: decade [6]  
## decade samp\_mean samp\_sd samp\_se  
## <chr> <dbl> <dbl> <dbl>  
## 1 20s 95.7 25.9 2.59  
## 2 30s 90.6 16.1 1.61  
## 3 40s 99.4 20.1 2.01  
## 4 50s 98.6 19.2 1.92  
## 5 60s 107. 23.6 2.36  
## 6 70s 104 16.7 1.67

compare=merge(results, samp\_dist, by="decade")  
compare

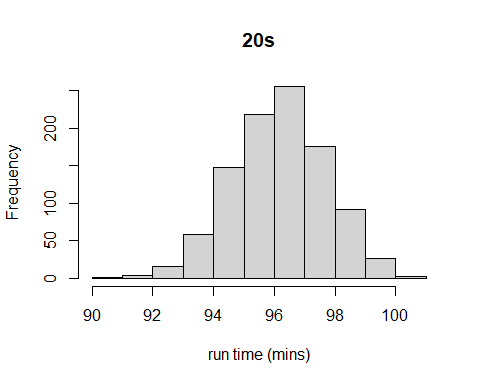
## decade pop\_mean pop\_sd pop\_n samp\_mean samp\_sd samp\_se  
## 1 20s 96.25658 26.11500 152 95.73 25.86674 2.586674  
## 2 30s 90.30000 17.27247 530 90.64 16.13799 1.613799  
## 3 40s 97.20332 19.11149 782 99.39 20.09970 2.009970  
## 4 50s 98.94820 19.19757 1081 98.60 19.24641 1.924641  
## 5 60s 105.58586 21.22436 1386 106.63 23.58636 2.358636  
## 6 70s 103.75000 17.95412 1720 104.00 16.65575 1.665575

#Pretty darn close!

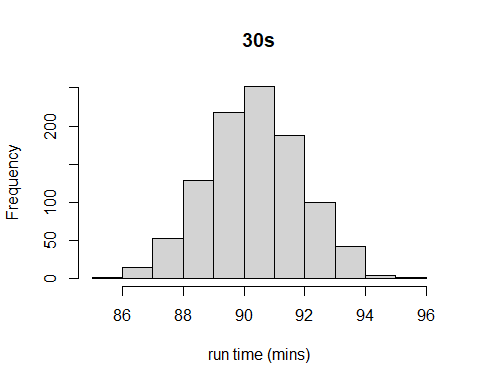
#creat function that generates sampling distributions  
  
sampling\_distribution=function(df, g, n){ #df = a dataframe, g = number reps, n = sample size  
 means=numeric(g)  
 sds=numeric(g)  
   
 for (i in 1:g){  
 sample=sample(df, n, replace=FALSE)  
 means[i]=mean(sample)  
 sds[i]=sd(sample)  
 }  
 samp\_mean=mean(means)  
 samp\_sd=sd(means)  
 res=list(means, sds, samp\_mean, samp\_sd)  
  
 return(res)  
}

#use function on all decades  
decades=vector("list", 6)  
  
for (i in 1:length(compare$decade)){  
 decades[[i]]=sampling\_distribution(filter(sub, decade==compare$decade[i])$runtimeMinutes, 1000, 100) #using above function  
}

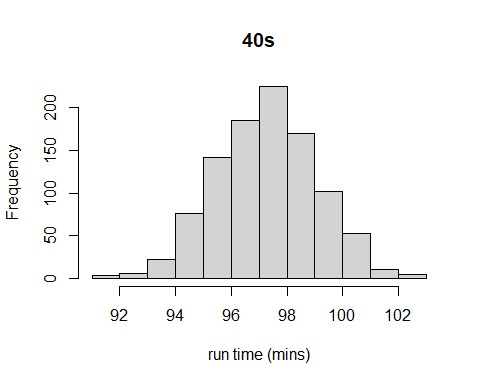
hist(decades[[1]][[1]], xlab="run time (mins)", main="20s")



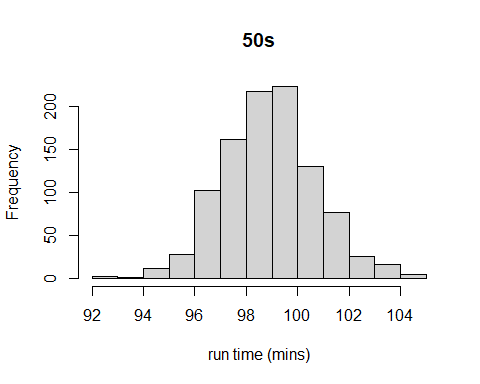
hist(decades[[2]][[1]], xlab="run time (mins)", main="30s")



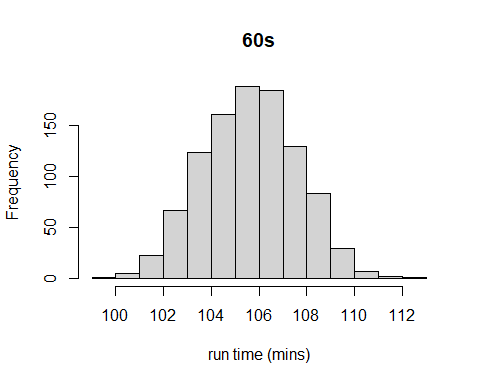
hist(decades[[3]][[1]], xlab="run time (mins)", main="40s")



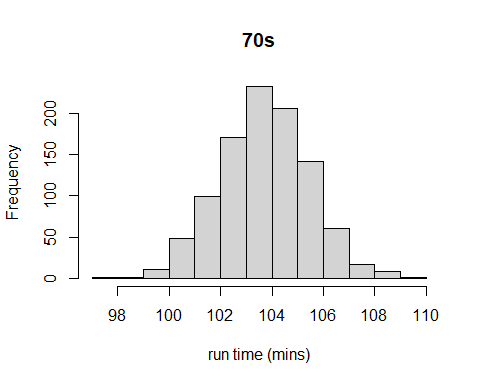
hist(decades[[4]][[1]], xlab="run time (mins)", main="50s")



hist(decades[[5]][[1]], xlab="run time (mins)", main="60s")



hist(decades[[6]][[1]], xlab="run time (mins)", main="70s")



#all are approximately normal

#get sd of all the sd sampling distributions (I hope this is correct, I was a little unsure as to which sampling distribution to summarize )  
compare$SE\_samp\_dist=c(sd(decades[[1]][[2]]),   
 sd(decades[[2]][[2]]),   
 sd(decades[[3]][[2]]),  
 sd(decades[[4]][[2]]),  
 sd(decades[[5]][[2]]),  
 sd(decades[[6]][[2]]))  
  
compare$SE\_pop\_var=results$pop\_sd/sqrt(results$pop\_n) #use values generated above to get pop SE  
compare[,7:9]

## samp\_se SE\_samp\_dist SE\_pop\_var  
## 1 2.586674 1.163831 2.1182064  
## 2 1.613799 1.484017 0.7502683  
## 3 2.009970 1.441917 0.6834254  
## 4 1.924641 1.647329 0.5838933  
## 5 2.358636 1.854607 0.5701026  
## 6 1.665575 1.776769 0.4329122

#sample SE and the SE calculated from the sampling distribution are pretty similar, though for the 20s there is a decent disparity. The SE from population variance is is much lower than the other 2 for all decades except 20s.

#Challenge 2

l <- 12  
  
print(paste('prob of 9 or fewer bees: ',ppois(9,l), sep=""))

## [1] "prob of 9 or fewer bees: 0.242392161670512"

print(paste('prob of 0 bees: ', 1-ppois(0,l, lower.tail = FALSE), sep=""))

## [1] "prob of 0 bees: 6.14421235334284e-06"

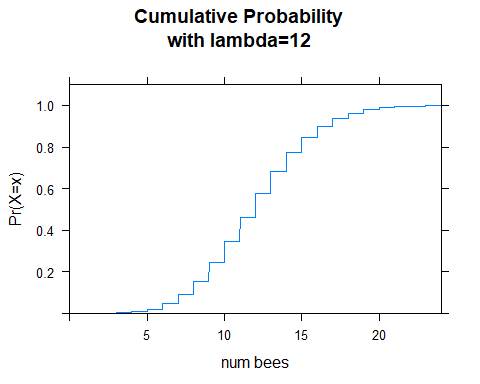
print(paste('prob of 5 bees: ', 1-(ppois(4,l)+ppois(6,l, lower.tail=F)), sep=""))

## [1] "prob of 5 bees: 0.0382219162075841"

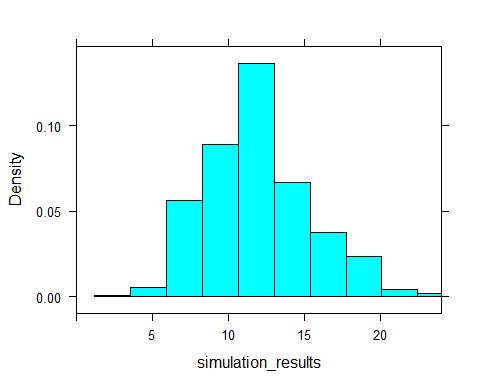
print(paste('prob of more than 18 bees: ', ppois(18,l, lower.tail=F), sep=""))

## [1] "prob of more than 18 bees: 0.0374164896633918"

p1 <-  
 plotDist(  
 "pois",  
 lambda = l,  
 kind = "cdf",  
 main = paste0("Cumulative Probability\nwith lambda=", l),  
 xlab = "num bees",  
 ylab = "Pr(X≤x)",  
 type = "l",  
 xlim=c(0,24)  
 )  
  
p1



simulation\_results=rpois(1460, 12)  
  
histogram(simulation\_results, xlim=c(0,24))



#Challenge 3

d <- as\_tibble(read\_csv('https://raw.githubusercontent.com/difiore/ada-2021-datasets/master/zombies.csv', col\_names = TRUE))

##   
## -- Column specification --------------------------------------------------------  
## cols(  
## id = col\_double(),  
## first\_name = col\_character(),  
## last\_name = col\_character(),  
## gender = col\_character(),  
## height = col\_double(),  
## weight = col\_double(),  
## zombies\_killed = col\_double(),  
## years\_of\_education = col\_double(),  
## major = col\_character(),  
## age = col\_double()  
## )

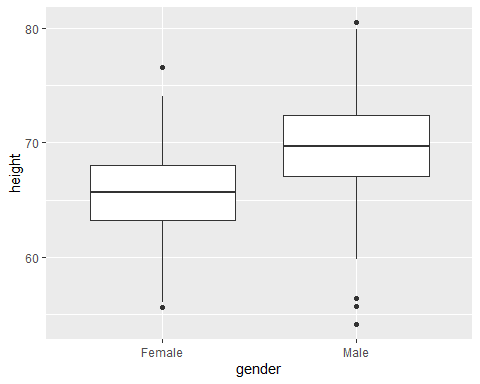
head(d)

## # A tibble: 6 x 10  
## id first\_name last\_name gender height weight zombies\_killed  
## <dbl> <chr> <chr> <chr> <dbl> <dbl> <dbl>  
## 1 1 Sarah Little Female 62.9 132. 2  
## 2 2 Mark Duncan Male 67.8 146. 5  
## 3 3 Brandon Perez Male 72.1 153. 1  
## 4 4 Roger Coleman Male 66.8 130. 5  
## 5 5 Tammy Powell Female 64.7 132. 4  
## 6 6 Anthony Green Male 71.2 153. 1  
## # ... with 3 more variables: years\_of\_education <dbl>, major <chr>, age <dbl>

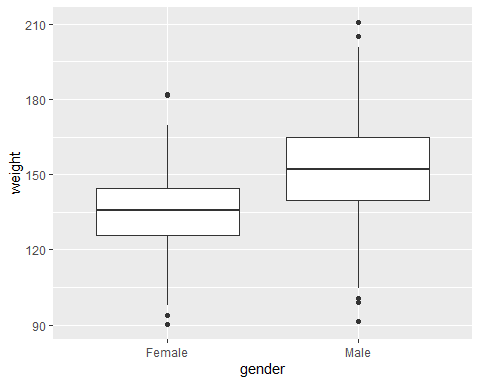
#pop mean and sd for each quant var  
pops=summarize(d[c(4:7,9),],  
pop\_mean\_height=mean(height, na.rm = TRUE),  
pop\_mean\_weight=mean(weight, na.rm = TRUE),  
pop\_mean\_zombies\_killed=mean(zombies\_killed, na.rm = TRUE),  
pop\_mean\_age=mean(age, na.rm = TRUE),  
pop\_mean\_education=mean(years\_of\_education, na.rm = TRUE),  
pop\_sd\_height=sqrt(sum((height - mean(height))^2)/(n())),  
pop\_sd\_weight=sqrt(sum((weight - mean(weight))^2)/(n())),  
pop\_sd\_zombies\_killed=sqrt(sum((zombies\_killed - mean(zombies\_killed))^2)/(n())),  
pop\_sd\_age=sqrt(sum((age - mean(age))^2)/(n())),  
pop\_sd\_education=sqrt(sum((years\_of\_education - mean(years\_of\_education))^2)/(n()))  
)  
  
pops

## # A tibble: 1 x 10  
## pop\_mean\_height pop\_mean\_weight pop\_mean\_zombie~ pop\_mean\_age pop\_mean\_educat~  
## <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 68.3 146. 3.8 19.7 4  
## # ... with 5 more variables: pop\_sd\_height <dbl>, pop\_sd\_weight <dbl>,  
## # pop\_sd\_zombies\_killed <dbl>, pop\_sd\_age <dbl>, pop\_sd\_education <dbl>

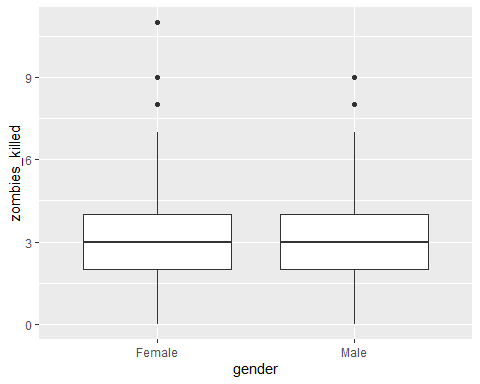
#box plots of quant vars by gender  
  
ggplot(d, aes(x=gender, y=height))+   
 geom\_boxplot()



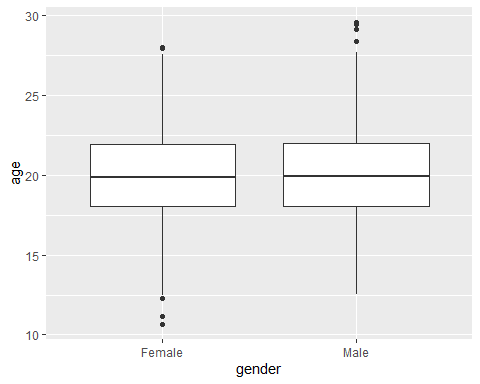
ggplot(d, aes(x=gender, y=weight))+   
 geom\_boxplot()



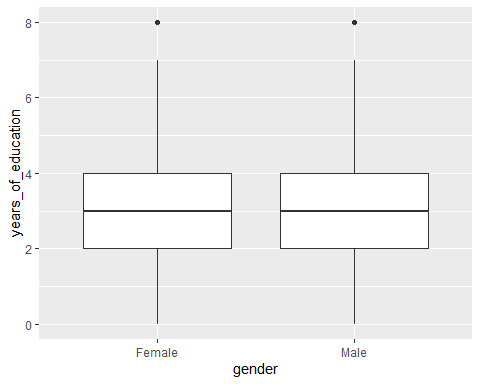
ggplot(d, aes(x=gender, y=zombies\_killed))+   
 geom\_boxplot()



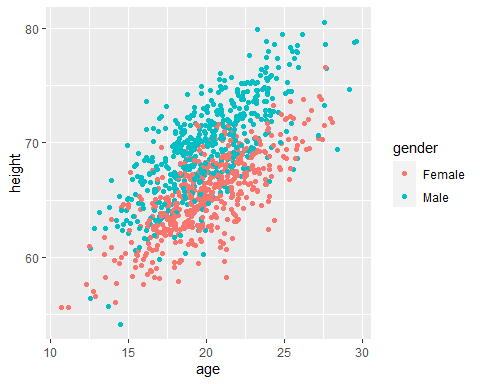
ggplot(d, aes(x=gender, y=age))+   
 geom\_boxplot()



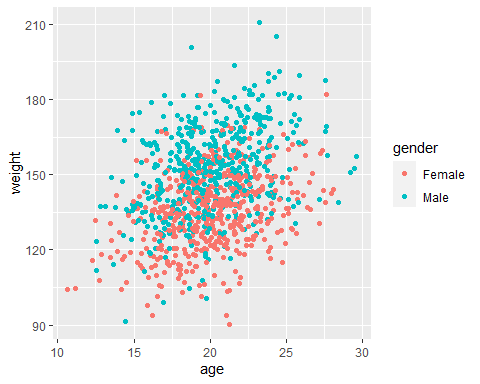
ggplot(d, aes(x=gender, y=years\_of\_education))+   
 geom\_boxplot()



#scatterplots of height and weight   
  
ggplot(d, aes(x=age, y=height, color=gender))+   
 geom\_point()

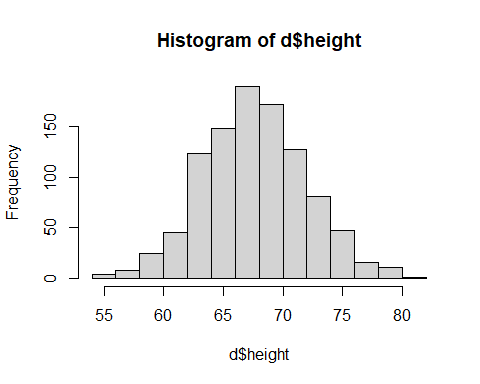


ggplot(d, aes(x=age, y=weight, color=gender))+   
 geom\_point()

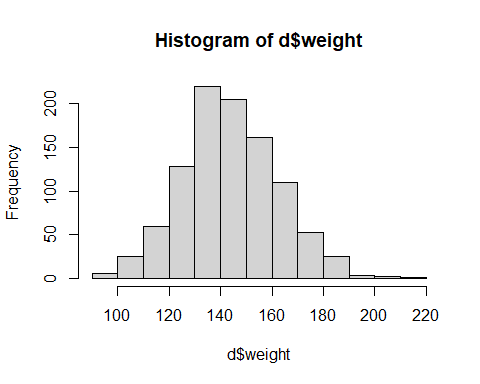


#height and weight seem to be pretty positively correlated, and the slope seems similar for men and women, but the intercept for women is a bit lower

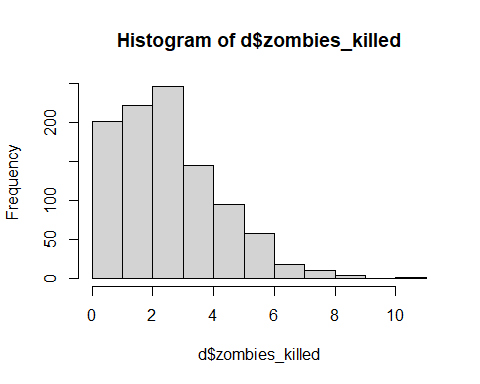
hist(d$height)



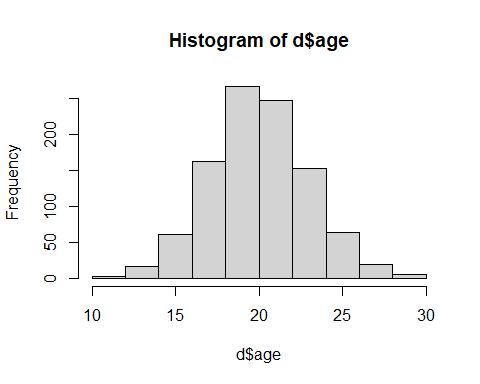
hist(d$weight)



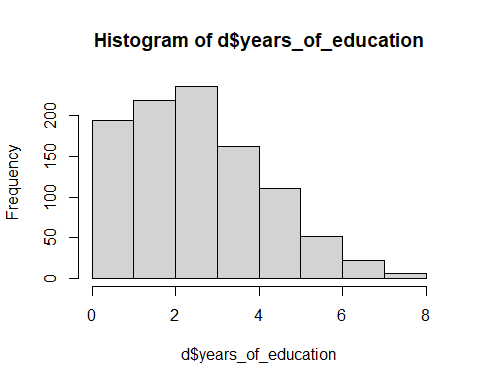
hist(d$zombies\_killed)



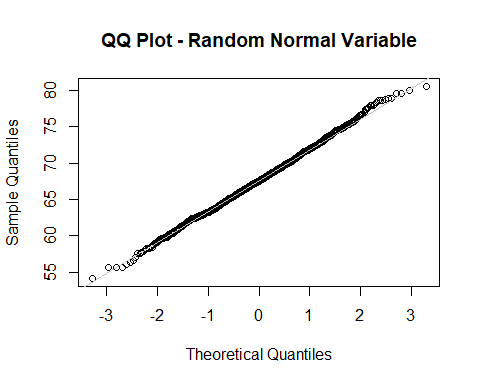
hist(d$age)



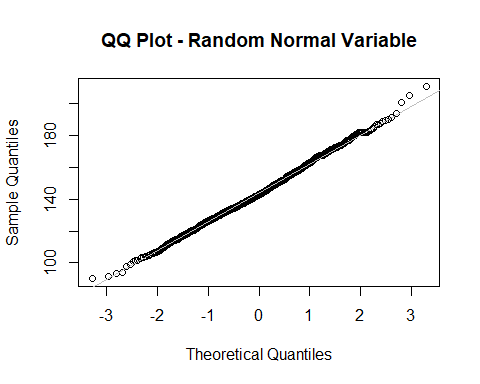
hist(d$years\_of\_education)



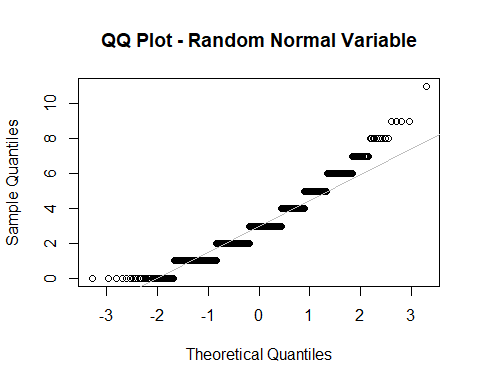
qqnorm(d$height, main = "QQ Plot - Random Normal Variable")  
qqline(d$height, col = "gray")



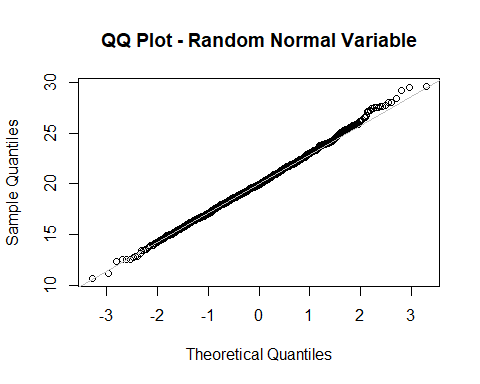
qqnorm(d$weight, main = "QQ Plot - Random Normal Variable")  
qqline(d$weight, col = "gray")



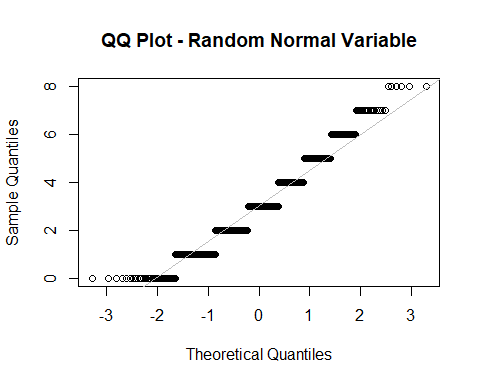
qqnorm(d$zombies\_killed, main = "QQ Plot - Random Normal Variable")  
qqline(d$zombies\_killed, col = "gray")



qqnorm(d$age, main = "QQ Plot - Random Normal Variable")  
qqline(d$age, col = "gray")



qqnorm(d$years\_of\_education, main = "QQ Plot - Random Normal Variable")  
qqline(d$years\_of\_education, col = "gray")



#Data for most variables seem normally distributed but, not unexpectedly, the 2 variables which represent counts (num zombies killed, years of education) have a poisson-ish distribution and are not normally distributed.

m=sample\_n(d, 50, replace = FALSE)  
  
  
m\_summ= m %>% summarize(  
 samp\_mean\_height= mean(height),  
 samp\_sd\_height=sd(height),  
 samp\_se\_height=sd(height)/50,  
 ci\_height=mean(height) + c(-1, 1) \* qnorm(1 - 0.05 / 2) \* samp\_se\_height,  
 samp\_mean\_weight= mean(weight),  
 samp\_sd\_weight=sd(weight),  
 samp\_se\_weight=sd(weight)/50,  
 ci\_weight=mean(weight) + c(-1, 1) \* qnorm(1 - 0.05 / 2) \* samp\_se\_weight,  
 samp\_mean\_zombies\_killed= mean(zombies\_killed),  
 samp\_sd\_zombies\_killed=sd(zombies\_killed),  
 samp\_se\_zombies\_killed=sd(zombies\_killed)/50,  
 ci\_zombies\_killed=mean(zombies\_killed) + c(-1, 1) \* qnorm(1 - 0.05 / 2) \* samp\_se\_zombies\_killed,  
 samp\_mean\_years\_of\_education= mean(years\_of\_education),  
 samp\_sd\_years\_of\_education=sd(years\_of\_education),  
 samp\_se\_years\_of\_education=sd(years\_of\_education)/50,  
 ci\_years\_of\_education= mean(years\_of\_education) + c(-1, 1) \* qnorm(1 - 0.05 / 2) \* samp\_se\_years\_of\_education,  
 samp\_mean\_age= mean(age),  
 samp\_sd\_age=sd(age),  
 samp\_se\_age=sd(age)/50,  
 ci\_age=mean(age) + c(-1, 1) \* qnorm(1 - 0.05 / 2) \* samp\_se\_age,  
)  
  
m\_summ

## # A tibble: 2 x 20  
## samp\_mean\_height samp\_sd\_height samp\_se\_height ci\_height samp\_mean\_weight  
## <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 67.5 4.79 0.0958 67.4 145.  
## 2 67.5 4.79 0.0958 67.7 145.  
## # ... with 15 more variables: samp\_sd\_weight <dbl>, samp\_se\_weight <dbl>,  
## # ci\_weight <dbl>, samp\_mean\_zombies\_killed <dbl>,  
## # samp\_sd\_zombies\_killed <dbl>, samp\_se\_zombies\_killed <dbl>,  
## # ci\_zombies\_killed <dbl>, samp\_mean\_years\_of\_education <dbl>,  
## # samp\_sd\_years\_of\_education <dbl>, samp\_se\_years\_of\_education <dbl>,  
## # ci\_years\_of\_education <dbl>, samp\_mean\_age <dbl>, samp\_sd\_age <dbl>,  
## # samp\_se\_age <dbl>, ci\_age <dbl>

#note, for the ci column, the number sin the 2 rows represent the upper and lower CI bounds, and are redundant for all other metrics

#generate sampling distribution; loop populates empty vectors with sample statistics.  
means\_height=numeric(100)  
means\_weight=numeric(100)  
means\_zombies\_killed=numeric(100)  
means\_years\_of\_education=numeric(100)  
means\_age=numeric(100)  
sd\_height=numeric(100)  
sd\_weight=numeric(100)  
sd\_zombies\_killed=numeric(100)  
sd\_years\_of\_education=numeric(100)  
sd\_age=numeric(100)  
  
for (i in 1:100){  
 if (i != 100){  
 temp=sample\_n(d, 50, replace = FALSE)}  
 else{  
 temp=m #uses my previous sample for the last slot  
 }  
 means\_height[i]=mean(temp$height)  
 means\_weight[i]=mean(temp$weight)  
 means\_zombies\_killed[i]=mean(temp$zombies\_killed)  
 means\_years\_of\_education[i]=mean(temp$years\_of\_education)  
 means\_age[i]=mean(temp$age)  
 sd\_height[i]= sd(temp$height)  
 sd\_weight[i]= sd(temp$weight)  
 sd\_zombies\_killed[i]= sd(temp$zombies\_killed)  
 sd\_years\_of\_education[i]= sd(temp$years\_of\_education)  
 sd\_age[i]= sd(temp$age)  
   
   
   
}

#means of sampling distributions of means  
print(paste('mean of sample means, height is ', mean(means\_height), sep=""))

## [1] "mean of sample means, height is 67.626223125646"

print(paste('mean of sample means, weight is ', mean(means\_weight), sep=""))

## [1] "mean of sample means, weight is 144.117449200526"

print(paste('mean of sample means, zombies\_killed is ', mean(means\_zombies\_killed), sep=""))

## [1] "mean of sample means, zombies\_killed is 2.9632"

print(paste('mean of sample means, age is ', mean(means\_age), sep=""))

## [1] "mean of sample means, age is 19.99862625233"

print(paste('mean of sample means, years\_of\_education is ', mean(means\_years\_of\_education), sep=""))

## [1] "mean of sample means, years\_of\_education is 2.9906"

print(paste('mean of sample sd, height is ', mean(sd\_height), sep=""))

## [1] "mean of sample sd, height is 4.29633748664818"

print(paste('mean of sample sd, weight is ', mean(sd\_weight), sep=""))

## [1] "mean of sample sd, weight is 18.3631529927531"

print(paste('mean of sample sd, zombies\_killed is ', mean(sd\_zombies\_killed), sep=""))

## [1] "mean of sample sd, zombies\_killed is 1.69904362196408"

print(paste('mean of sample sd, years\_of\_education is ', mean(sd\_years\_of\_education), sep=""))

## [1] "mean of sample sd, years\_of\_education is 1.66991493422568"

print(paste('mean of sample sd, age is ', mean(sd\_age), sep=""))

## [1] "mean of sample sd, age is 2.92411119202113"

#sds of sampling distrinutions of means  
print(paste('sd of sample means, height is ', sd(means\_height), sep=""))

## [1] "sd of sample means, height is 0.577077624965377"

print(paste('sd of sample means, weight is ', sd(means\_weight), sep=""))

## [1] "sd of sample means, weight is 2.75451519757556"

print(paste('sd of sample means, zombies\_killed is ', sd(means\_zombies\_killed), sep=""))

## [1] "sd of sample means, zombies\_killed is 0.231267390828465"

print(paste('sd of sample means, age is ', sd(means\_age), sep=""))

## [1] "sd of sample means, age is 0.370324698040198"

print(paste('sd of sample means, years\_of\_education is ', sd(means\_years\_of\_education), sep=""))

## [1] "sd of sample means, years\_of\_education is 0.242136200089824"

print(paste('sd of sample sd, height is ', sd(sd\_height), sep=""))

## [1] "sd of sample sd, height is 0.456815290142482"

print(paste('sd of sample sd, weight is ', sd(sd\_weight), sep=""))

## [1] "sd of sample sd, weight is 1.78347311521541"

print(paste('sd of sample sd, zombies\_killed is ', sd(sd\_zombies\_killed), sep=""))

## [1] "sd of sample sd, zombies\_killed is 0.190155995075931"

print(paste('sd of sample sd, years\_of\_education is ', sd(sd\_years\_of\_education), sep=""))

## [1] "sd of sample sd, years\_of\_education is 0.146506174491536"

print(paste('sd of sample sd, age is ', sd(sd\_age), sep=""))

## [1] "sd of sample sd, age is 0.31537809164765"

#CIs of sampling distribution of means. This printed weird... but I was running out of time! apologies  
print(paste('CI of sample means, height is ', mean(means\_height) + c(-1, 1) \* qnorm(1 - 0.05 / 2) \* (sd(means\_height)/sqrt(100)), sep=""))

## [1] "CI of sample means, height is 67.5131179895244"  
## [2] "CI of sample means, height is 67.7393282617676"

print(paste('CI of sample means, weight is ', mean(means\_weight) + c(-1, 1) \* qnorm(1 - 0.05 / 2) \* (sd(means\_weight)/sqrt(100)), sep=""))

## [1] "CI of sample means, weight is 143.577574142314"  
## [2] "CI of sample means, weight is 144.657324258738"

print(paste('CI of sample means, zombies\_killed is ', mean(means\_zombies\_killed) + c(-1, 1) \* qnorm(1 - 0.05 / 2) \* (sd(means\_zombies\_killed)/sqrt(100)), sep=""))

## [1] "CI of sample means, zombies\_killed is 2.91787242431777"  
## [2] "CI of sample means, zombies\_killed is 3.00852757568223"

print(paste('CI of sample means, age is ', mean(means\_age) + c(-1, 1) \* qnorm(1 - 0.05 / 2) \* (sd(means\_age)/sqrt(100)), sep=""))

## [1] "CI of sample means, age is 19.9260439452556"  
## [2] "CI of sample means, age is 20.0712085594044"

print(paste('CI of sample means, years\_of\_education is ', mean(means\_years\_of\_education) + c(-1, 1) \* qnorm(1 - 0.05 / 2) \* (sd(means\_years\_of\_education)/sqrt(100)), sep=""))

## [1] "CI of sample means, years\_of\_education is 2.94314217684706"  
## [2] "CI of sample means, years\_of\_education is 3.03805782315294"

print(paste('CI of sample sd, height is ', mean(sd\_height) + c(-1, 1) \* qnorm(1 - 0.05 / 2) \* (sd(sd\_height)/sqrt(100)), sep=""))

## [1] "CI of sample sd, height is 4.20680333502153"  
## [2] "CI of sample sd, height is 4.38587163827483"

print(paste('CI of sample sd, weight is ', mean(sd\_weight) + c(-1, 1) \* qnorm(1 - 0.05 / 2) \* (sd(sd\_weight)/sqrt(100)), sep=""))

## [1] "CI of sample sd, weight is 18.0135986854313"  
## [2] "CI of sample sd, weight is 18.7127073000748"

print(paste('CI of sample sd, zombies\_killed is ', mean(sd\_zombies\_killed) + c(-1, 1) \* qnorm(1 - 0.05 / 2) \* (sd(sd\_zombies\_killed)/sqrt(100)), sep=""))

## [1] "CI of sample sd, zombies\_killed is 1.66177373178476"  
## [2] "CI of sample sd, zombies\_killed is 1.7363135121434"

print(paste('CI of sample sd, years\_of\_education is ', mean(sd\_years\_of\_education) + c(-1, 1) \* qnorm(1 - 0.05 / 2) \* (sd(sd\_years\_of\_education)/sqrt(100)), sep=""))

## [1] "CI of sample sd, years\_of\_education is 1.64120025167406"  
## [2] "CI of sample sd, years\_of\_education is 1.69862961677729"

print(paste('CI of sample sd, age is ', mean(sd\_age) + c(-1, 1) \* qnorm(1 - 0.05 / 2) \* (sd(sd\_age)/sqrt(100)), sep=""))

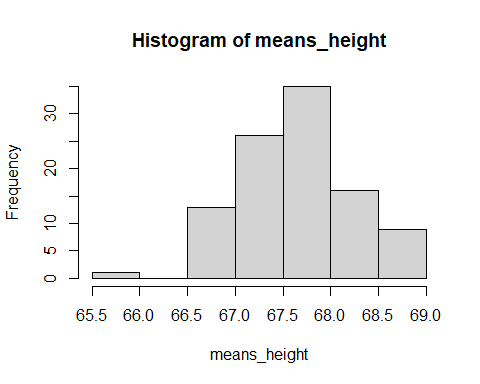
## [1] "CI of sample sd, age is 2.8622982219069"   
## [2] "CI of sample sd, age is 2.98592416213537"

m\_summ

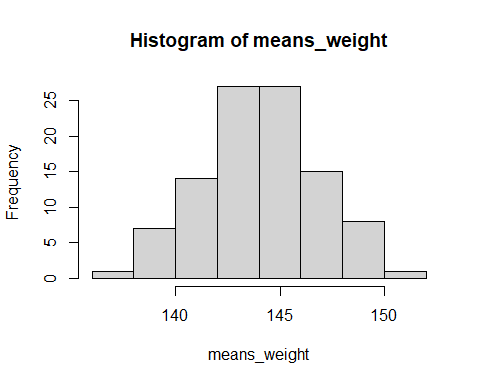
## # A tibble: 2 x 20  
## samp\_mean\_height samp\_sd\_height samp\_se\_height ci\_height samp\_mean\_weight  
## <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 67.5 4.79 0.0958 67.4 145.  
## 2 67.5 4.79 0.0958 67.7 145.  
## # ... with 15 more variables: samp\_sd\_weight <dbl>, samp\_se\_weight <dbl>,  
## # ci\_weight <dbl>, samp\_mean\_zombies\_killed <dbl>,  
## # samp\_sd\_zombies\_killed <dbl>, samp\_se\_zombies\_killed <dbl>,  
## # ci\_zombies\_killed <dbl>, samp\_mean\_years\_of\_education <dbl>,  
## # samp\_sd\_years\_of\_education <dbl>, samp\_se\_years\_of\_education <dbl>,  
## # ci\_years\_of\_education <dbl>, samp\_mean\_age <dbl>, samp\_sd\_age <dbl>,  
## # samp\_se\_age <dbl>, ci\_age <dbl>

#The original CIS I calculated from my sample are similar to the CIs calculated from the sampling distribution, though there are some discrepancies. However, the intervals are, for the most part, pretty similar in how 'tight' they are.

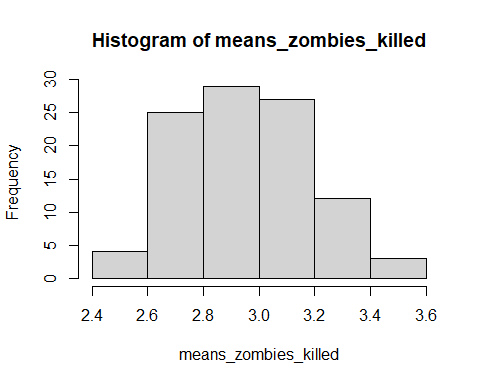
# the standard deviations of the sampling distributions are much higher than the standard errors calculated for our sample of 50, apart from weight.  
  
  
#plot sampling distributions:  
  
hist(means\_height)



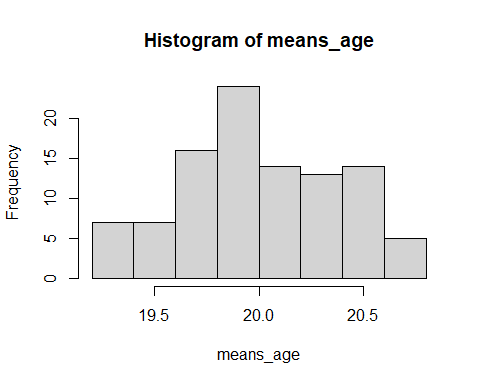
hist(means\_weight)



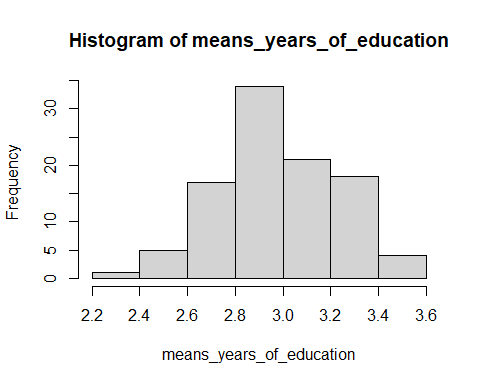
hist(means\_zombies\_killed)



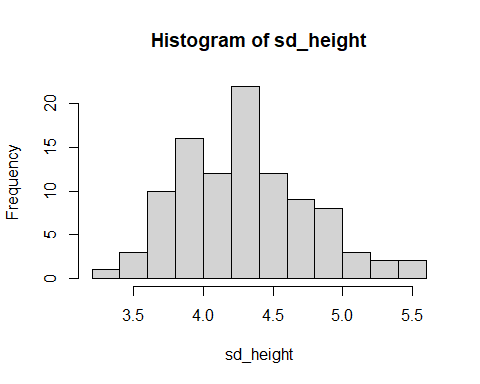
hist(means\_age)



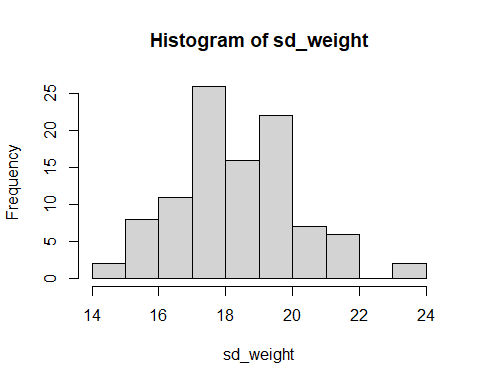
hist(means\_years\_of\_education)



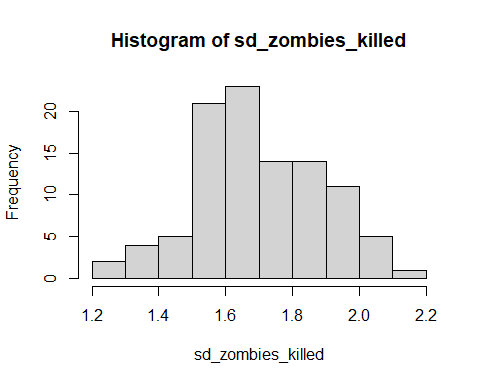
hist(sd\_height)



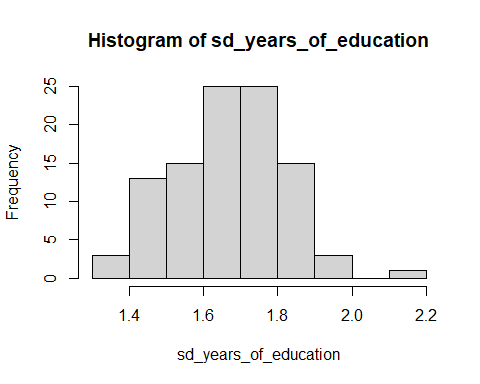
hist(sd\_weight)



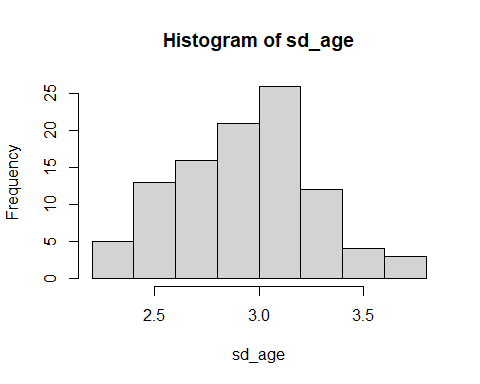
hist(sd\_zombies\_killed)



hist(sd\_years\_of\_education)



hist(sd\_age)



#all of the sampling distributions look pretty normal-ish, even for the count variables. However, number of zombies killed looks slightly 'block-y'.