Lab 1

Moshe Weiss

This lab is due 11:59 PM Satuday 2/9/19.

You should have RStudio installed to edit this file. You will write code in places marked "TO-DO" to complete the problems. Some of this will be a pure programming assignment. The tools for the solutions to these problems can be found in the class practice lectures. I want you to use the methods I taught you, not for you to google and come up with whatever works. You won't learn that way.

To "hand in" the homework, you should compile or publish this file into a PDF that includes output of your code. Once it's done, push by the deadline to your repository in a directory called "labs".

- Print out the numerical constant pi with ten digits after the decimal point using the internal constant `pi`.

```
options(digits=11)
pi
```

```
## [1] 3.1415926536
```

```
options(digits=7)
```

- Sum up the first 100 terms of the series 1 + 1/2 + 1/4 + 1/8 + …

```
sum((1/2)^(0:99))
```

```
## [1] 2
```

- Find the product of the first 100 terms of `1 * 1/2 * 1/4 * 1/8 *` …

```
prod((1/2)^(0:99))
```

```
## [1] 0
```

- Find the product of the first 500 terms of `1 * 1/2 * 1/4 * 1/8 *` … Answer in English: is this answer correct?

```
prod((1/2)^(0:499))
```

```
## [1] 0
```

```
"Nope, numerical underflow"
```

```
## [1] "Nope, numerical underflow"
```

- Figure out a means to express the answer more exactly. Not compute exactly, but express more exactly.

```
paste("10 e", -log10(2)*sum(0:499))
```

```
## [1] "10 e -37553.4919590817"
```

- Use the left rectangle method to numerically integrate x^2 from 0 to 1 with rectangle size 1e-6.

```
sum(seq(0,1,by=(1e-6))^2)*(1e-6)
```

```
## [1] 0.3333338
```

- Calculate the average of 100 realizations of standard Bernoullis in one line using the `sample` function.

```
sum(sample((0:1),100,replace = TRUE))/100
```

```
## [1] 0.43
```

- Calculate the average of 500 realizations of Bernoullis with p = 0.9 in one line using the `sample` function.

```
sum(sample(c(0,1,1,1,1,1,1,1,1,1),500,replace=TRUE))/500
```

```
## [1] 0.91
```

- Calculate the average of 1000 realizations of Bernoullis with p = 0.9 in one line using `rbinom`.

```
sum(rbinom(1000,1,0.9))/1000
```

```
## [1] 0.879
```

- Use the `strsplit` function and `sample` to put the sentences below in random order.

```
lorem = "Lorem ipsum dolor sit amet, consectetur adipiscing elit. Morbi posuere varius volutpat. Morbi
paste(paste(sample(unlist(strsplit(lorem,"[.] "))), collapse = ". "), ".", sep = "")
```

```
## [1] "Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec at tempor erat. Aenean nulla ant
```

- In class we generated the variable criminality with levels "none", "infraction", "misdimeanor" and "felony". Create a variable `x_2` here with 100 random elements (equally probable) and ensure the proper ordinal ordering.

```
y=c("none", "infraction", "misdimeanor", "felony")
x_2 = factor(sample(y,100,replace = TRUE),levels = c("none", "infraction", "misdimeanor", "felony"), or
x_2
```

```
##   [1] felony      infraction  misdimeanor misdimeanor infraction
##   [6] infraction  felony      infraction  none        none
##  [11] none        infraction  misdimeanor none        misdimeanor
##  [16] none        none        none        misdimeanor infraction
##  [21] none        misdimeanor felony      none        infraction
##  [26] infraction  infraction  misdimeanor none        misdimeanor
##  [31] misdimeanor none        none        felony      felony
##  [36] misdimeanor misdimeanor infraction  misdimeanor felony
##  [41] none        none        misdimeanor none        felony
##  [46] none        infraction  infraction  none        infraction
##  [51] infraction  none        felony      misdimeanor none
##  [56] none        infraction  none        felony      none
##  [61] felony      none        misdimeanor infraction  infraction
##  [66] none        felony      infraction  felony      felony
##  [71] felony      none        none        infraction  misdimeanor
##  [76] none        misdimeanor felony      misdimeanor misdimeanor
##  [81] misdimeanor infraction  misdimeanor misdimeanor felony
##  [86] none        none        infraction  infraction  misdimeanor
##  [91] none        none        infraction  misdimeanor none
##  [96] infraction  felony      misdimeanor misdimeanor infraction
## Levels: none < infraction < misdimeanor < felony
```

- Convert this variable to binary where 0 is no crime and 1 is any crime. Answer in English: is this the proper binary threshold?

```
x_3=as.numeric(x_2!="none")
"No, it doesn't appropriately inform the proclivity to pay back."
```

```
## [1] "No, it doesn't appropriately inform the proclivity to pay back."
```

```
#I'm a bit concerned I have the wrong idea about the binary threshold.
```

- Convert this variable to an unordered, nominal factor variable.

```
#if converting binary to nominal
#for(index in 1:length(x_3)){
#  if(x_3[index]==1){
#    x_3[index] = "crime"
#  }else{
#    x_3[index] = "no crime"
#  }
#}
#x_3
#if converting from ordinal to nominal
factor(x_2,ordered = FALSE)
```

```
##   [1] felony      infraction  misdimeanor misdimeanor infraction 
##   [6] infraction  felony      infraction  none        none       
##  [11] none        infraction  misdimeanor none        misdimeanor
##  [16] none        none        none        misdimeanor infraction 
##  [21] none        misdimeanor felony      none        infraction 
##  [26] infraction  infraction  misdimeanor none        misdimeanor
##  [31] misdimeanor none        none        felony      felony     
##  [36] misdimeanor misdimeanor infraction  misdimeanor felony     
##  [41] none        none        misdimeanor none        felony     
##  [46] none        infraction  infraction  none        infraction 
##  [51] infraction  none        felony      misdimeanor none       
##  [56] none        infraction  none        felony      none       
##  [61] felony      none        misdimeanor infraction  infraction 
##  [66] none        felony      infraction  felony      felony     
##  [71] felony      none        none        infraction  misdimeanor
##  [76] none        misdimeanor felony      misdimeanor misdimeanor
##  [81] misdimeanor infraction  misdimeanor misdimeanor felony     
##  [86] none        none        infraction  infraction  misdimeanor
##  [91] none        none        infraction  misdimeanor none       
##  [96] infraction  felony      misdimeanor misdimeanor infraction 
## Levels: none infraction misdimeanor felony
```

- Convert this variable into three binary variables without any information loss and put them into a data matrix.

```
#create level vectors
msdmnr = c()
infrctn = c()
flny = c()
# assign appropriate binary value
for(instance in x_2){
  flny = c(flny, as.numeric(instance =="felony"))

  msdmnr = c(msdmnr, as.numeric(instance == "misdimeanor"))

  infrctn = c(infrctn, as.numeric(instance =="infraction"))
}
#columns correspond to order listed in the following function. If row is composed of zeroes, no crim hi
x_4 = matrix(c(infrctn,msdmnr,flny),100,3)
x_4
```

```
##        [,1] [,2] [,3]
##   [1,]    0    0    1
##   [2,]    1    0    0
##   [3,]    0    1    0
##   [4,]    0    1    0
##   [5,]    1    0    0
##   [6,]    1    0    0
##   [7,]    0    0    1
##   [8,]    1    0    0
##   [9,]    0    0    0
##  [10,]    0    0    0
##  [11,]    0    0    0
##  [12,]    1    0    0
##  [13,]    0    1    0
##  [14,]    0    0    0
##  [15,]    0    1    0
##  [16,]    0    0    0
##  [17,]    0    0    0
##  [18,]    0    0    0
##  [19,]    0    1    0
##  [20,]    1    0    0
##  [21,]    0    0    0
##  [22,]    0    1    0
##  [23,]    0    0    1
##  [24,]    0    0    0
##  [25,]    1    0    0
##  [26,]    1    0    0
##  [27,]    1    0    0
##  [28,]    0    1    0
##  [29,]    0    0    0
##  [30,]    0    1    0
##  [31,]    0    1    0
##  [32,]    0    0    0
##  [33,]    0    0    0
##  [34,]    0    0    1
##  [35,]    0    0    1
##  [36,]    0    1    0
##  [37,]    0    1    0
##  [38,]    1    0    0
##  [39,]    0    1    0
##  [40,]    0    0    1
##  [41,]    0    0    0
##  [42,]    0    0    0
##  [43,]    0    1    0
##  [44,]    0    0    0
##  [45,]    0    0    1
##  [46,]    0    0    0
##  [47,]    1    0    0
##  [48,]    1    0    0
##  [49,]    0    0    0
##  [50,]    1    0    0
##  [51,]    1    0    0
##  [52,]    0    0    0
##  [53,]    0    0    1
##  [54,]    0    1    0
##  [55,]    0    0    0
##  [56,]    0    0    0
##  [57,]    1    0    0
##  [58,]    0    0    0
##  [59,]    0    0    1
##  [60,]    0    0    0
##  [61,]    0    0    1
##  [62,]    0    0    0
##  [63,]    0    1    0
##  [64,]    1    0    0
##  [65,]    1    0    0
##  [66,]    0    0    0
##  [67,]    0    0    1
##  [68,]    1    0    0
##  [69,]    0    0    1
##  [70,]    0    0    1
##  [71,]    0    0    1
##  [72,]    0    0    0
##  [73,]    0    0    0
##  [74,]    1    0    0
##  [75,]    0    1    0
##  [76,]    0    0    0
##  [77,]    0    1    0
##  [78,]    0    0    1
##  [79,]    0    1    0
##  [80,]    0    1    0
##  [81,]    0    1    0
##  [82,]    1    0    0
##  [83,]    0    1    0
##  [84,]    0    1    0
##  [85,]    0    0    1
##  [86,]    0    0    0
##  [87,]    0    0    0
##  [88,]    1    0    0
##  [89,]    1    0    0
##  [90,]    0    1    0
##  [91,]    0    0    0
##  [92,]    0    0    0
##  [93,]    1    0    0
##  [94,]    0    1    0
##  [95,]    0    0    0
##  [96,]    1    0    0
##  [97,]    0    0    1
##  [98,]    0    1    0
##  [99,]    0    1    0
## [100,]    1    0    0
```

- What should the sum of each row be (in English)? Verify that.

```r
"The sum of each row should be equal to 0 or 1, depending on the existence of criminal history"
```

```
## [1] "The sum of each row should be equal to 0 or 1, depending on the existence of criminal history"
```

```r
  for(i in (1:100)){
    if(!(sum(x_4[i,]) == 1 | sum(x_4[i,]) == 0)){
        stop("Sum of row is not 0 or 1")
    }
  }
  print(TRUE)
```

```
## [1] TRUE
```

- How should the column sum look (in English)? Verify that.

```
"The column sum should be between 0 and 100, though it will likely be around 25 since all 4 outcomes are
```

```
## [1] "The column sum should be between 0 and 100, though it will likely be around 25 since all 4 outco
```

```
sumCol1 = sum(x_4[,1])
sumCol2 = sum(x_4[,2])
sumCol3 = sum(x_4[,3])
nones=100-sum(sumCol1,sumCol2,sumCol3)
print(paste("Sum of column 1 is less than 100 and greater than 0: ",(sumCol1 > 0 && sumCol1 < 100)))
```

```
## [1] "Sum of column 1 is less than 100 and greater than 0:  TRUE"
```

```
print(paste("Sum of column 2 is less than 100 and greater than 0: ",(sumCol2 > 0 && sumCol2 < 100)))
```

```
## [1] "Sum of column 2 is less than 100 and greater than 0:  TRUE"
```

```
print(paste("Sum of column 3 is less than 100 and greater than 0: ",(sumCol3 > 0 && sumCol3 < 100)))
```

```
## [1] "Sum of column 3 is less than 100 and greater than 0:  TRUE"
```

- Generate a matrix with 100 rows where the first column is realization from a normal with mean 17 and variance 38, the second column is uniform between -10 and 10, the third column is poisson with mean 6, the fourth column in exponential with lambda of 9, the fifth column is binomial with n = 20 and p = 0.12 and the sixth column is a binary variable with 24% 1's.

```
y = matrix(c(rnorm(100,17,38),runif(100,-10,10),rpois(100,6),rexp(100,6),rbinom(100,20,.12),sample(c(c(
y
```

```
##              [,1]       [,2] [,3]        [,4] [,5] [,6]
##   [1,]  48.1430292  3.1698593    3 0.212683025    0    0
##   [2,]  83.5049455 -7.6844535    5 0.206588366    2    0
##   [3,]  11.3714642  2.0583412    7 0.225542315    4    0
##   [4,]  82.2744792 -3.2612535    9 0.197403248    1    0
##   [5,] -21.1017262 -7.0628280    5 0.087423046    2    1
##   [6,]  -3.3684629  4.9960593    0 0.586893787    1    1
##   [7,]  45.8265133  7.5636526    4 0.099993632    0    0
##   [8,] -29.3986845  1.7893580    3 0.098861584    2    0
##   [9,]  15.9569871 -4.1495855    7 0.468270345    1    1
##  [10,]  48.7339727  3.1591708    5 0.342233479    0    0
##  [11,]  63.0436185 -8.7232064    3 0.236525753    2    1
##  [12,] -19.4225013 -1.0244633    7 0.109144047    2    1
##  [13,] -24.2345300  1.9490713    7 0.053467992    3    1
##  [14,]  -9.0942130  5.1475284    8 0.214685222    2    0
##  [15,] -12.1726942 -4.9928132    4 0.158168468    2    0
##  [16,]   3.2278016  6.1440970    9 0.017971820    6    0
##  [17,] -37.4015800 -2.3615438    9 0.099573719    2    0
##  [18,]  17.7022401 -4.0550228    7 0.249941372    5    0
##  [19,]  33.9733891  7.9406918    9 0.361386696    0    0
##  [20,]  52.4052033  8.9505159    6 0.059507507    3    0
##  [21,] 119.3070884  1.3162027    8 0.055698742    4    1
## [22,] -11.4015365  9.5503022    7 0.088927229    5    0
## [23,]  45.9205155 -9.3749536    8 0.001905976    1    1
## [24,]  44.9232525 -2.4052832    6 0.281959965    1    0
## [25,]  -9.4514858  3.1883660    6 0.090045903    1    0
## [26,]  60.3947649 -2.3932643    7 0.031530863    1    1
## [27,]  25.8524287  1.7663256    5 0.248584054    3    0
## [28,]  59.9410370 -9.6416085    2 0.075504153    3    1
## [29,]  19.4064053  7.4404462    7 0.010987456    2    0
## [30,]  19.1471225  4.8782985    7 0.412132324    1    0
## [31,]  19.9046562 -1.7968123    5 0.314405829    3    0
## [32,]  24.1018526  0.1834998    5 0.164021527    3    0
## [33,]  21.8446439  0.4294380   10 0.229544390    1    1
## [34,]  22.1317453 -3.6786202   10 0.083557108    2    0
## [35,] 101.0223912  6.3076633    5 0.599970828    2    0
## [36,]  50.5747712 -3.7540650    4 0.109292756    3    0
## [37,] -25.7304999 -1.3778161    7 0.014947046    2    0
## [38,]   9.2599091 -0.5576643    7 0.254059994    1    1
## [39,]  39.3466147  0.4600024    4 0.216980529    4    0
## [40,]   5.9422123 -2.2149208    5 0.090788996    3    0
## [41,] 132.8102400  1.6492225    7 0.211844496    3    0
## [42,]  -2.3826510  2.7216480    5 0.079368532    3    0
## [43,]   3.1812527  7.4483995    6 0.063330268    1    0
## [44,]  75.1315612  7.7204373    6 0.305679578    5    0
## [45,]  -3.6976954 -0.1214477    8 0.804033233    1    1
## [46,]  41.4554162  1.2415721    6 0.180999642    2    0
## [47,] -32.2677138  8.0918031   10 0.015197762    6    0
## [48,]   8.6186154 -9.7487193    8 0.022745948    3    0
## [49,]  16.4584408  1.1113568    5 0.040318638    2    0
## [50,]  13.8856585 -4.0748206    5 0.065166328    0    1
## [51,]  -1.0857766  0.9676041    9 0.541694040    4    0
## [52,]  32.9840461 -1.8004526    5 0.634044964    0    0
## [53,]  53.7940701 -8.3443292    5 0.013882163    3    0
## [54,]  26.4520786 -1.8942290    5 0.333848502    4    0
## [55,]  -0.1481111  8.4598749    2 0.199615145    0    1
## [56,] -24.7662111 -3.0088421    6 0.114023050    0    0
## [57,]  12.4387246 -5.3000560    6 0.428846551    4    0
## [58,]   0.7672397 -4.5653742    5 0.103988953    0    1
## [59,]  62.6780110  4.7866920    6 0.204684893    3    0
## [60,]  90.0647514  9.2279339    6 0.055142135    3    1
## [61,] -23.2729208 -7.1021284    6 0.108902142    1    1
## [62,]  -9.5652954 -6.6763619    6 0.247014378    2    0
## [63,]  53.5046229 -5.4596357    5 0.199275442    2    1
## [64,]   3.7936526  3.3277803    2 0.632693220    6    0
## [65,]  12.3141368  3.2244027    5 0.112429765    2    0
## [66,] -44.8680596 -0.7981700    5 0.061340523    4    0
## [67,]  18.8909771 -5.7127703    2 0.028405868    2    0
## [68,]  36.7810066  8.1851450    5 0.029967511    2    0
## [69,]  46.7413118  5.7027082    5 0.020484788    4    0
## [70,] -12.0523036 -8.0822428    2 0.168758113    4    0
## [71,]  29.6002659  9.3383025    7 0.083647604    2    0
## [72,]  26.3170366  0.5993754    8 0.261363487    0    1
## [73,]  59.3033010  2.9837055    6 0.176530490    4    0
## [74,]  40.5123747 -0.5950302    6 0.013768228    2    0
## [75,] -71.0514039 -0.9135701    2 0.451630064    3    0
##  [76,]  34.4831316  0.4136837    6 0.480954580    2    0
##  [77,]  -2.1353821 -2.4973729    5 0.019363562    1    0
##  [78,]  -2.8761503 -0.8920657    5 0.141183877    2    0
##  [79,]  23.8331186  9.1034300    7 0.195406940    2    0
##  [80,]  76.6398865 -3.5140923    4 0.191840376    2    0
##  [81,]  26.8226961  8.4413865   13 0.242139814    3    0
##  [82,] -10.3137036  4.4958528    7 0.061839041    3    0
##  [83,]  51.8497048 -4.5286362    9 0.163180926    2    0
##  [84,] -28.0026358 -7.6434645   10 0.002979791    2    1
##  [85,] -39.9750790  7.7261454    7 0.018758265    4    0
##  [86,] -24.9995396 -8.0420404    8 0.475761023    4    0
##  [87,]   4.7863197  0.5904083    6 0.070132082    2    0
##  [88,] -10.0008550 -6.3034864    2 0.021677439    2    1
##  [89,]  34.0845156 -1.6093452    5 0.154304647    1    0
##  [90,]  13.8809186 -5.1395897    7 0.016318114    1    0
##  [91,]  38.0287397  9.3739368    4 0.091785723    3    0
##  [92,] -18.3645631  7.5507351    7 0.485419569    1    0
##  [93,]  -2.1267003  7.0683146    6 0.107104988    4    0
##  [94,] 163.2281037 -4.5900027    5 0.051787121    2    1
##  [95,]  24.2271371 -1.3436805   10 0.246174308    1    0
##  [96,]  48.6082167  6.6464817   10 0.193005788    2    0
##  [97,] -43.7688812 -6.5349047    8 0.033122367    5    0
##  [98,] -34.3021104  5.6834702    6 0.053131755    4    0
##  [99,] -25.9333772  5.4396327    5 0.060809941    1    1
## [100,] -31.8315675 -1.6918193    6 0.194953517    2    0
```