Lab 1

Vincent Miceli

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`.

```
pi
```

```
## [1] 3.141593
```

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

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

```
sum(2^(0:-99))
```

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

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

```
prod(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? It is not exactly correct, the exact answer is 10^-33579.

```
prod(2^(0:-499))
```

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

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

```
10^(-33579)
```

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

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

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

```
## [1] 333333.83333
```

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

```
mean(sample(c(0,1), 100, replace = TRUE))
```

```
## [1] 0.52
```

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

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

```
## [1] 0.9
```

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

```
mean(rbinom(1000, size = 1, prob = .9))
```

```
## [1] 0.898
```

- 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
sample(strsplit(lorem, " ")[[1]], 68)
```

```
##  [1] "Cras"        "dolor"       "Donec"       "non"         "erat."
##  [6] "quam"        "consectetur" "nibh"        "elit."       "arcu"
## [11] "augue,"      "Mauris"      "in,"         "ultricies"   "ultricies"
## [16] "magna."      "at"          "sodales"     "volutpat."   "nisi"
## [21] "ante,"       "viverra."    "sit"         "lacinia"     "faucibus"
## [26] "id"          "in."         "vehicula"    "dapibus"     "finibus"
## [31] "semper"      "tempor"      "id"          "posuere"     "massa"
## [36] "suscipit"    "Morbi"       "posuere"     "Morbi"       "ac,"
## [41] "nulla"       "amet,"       "at"          "elementum."  "varius"
## [46] "ipsum"       "Lorem"       "Donec"       "arcu."       "Integer"
## [51] "Curabitur"   "sed"         "augue."      "eget"        "semper."
## [56] "vehicula"    "congue"      "Aenean"      "sagittis"    "eu"
## [61] "vel"         "adipiscing"  "mi"          "ligula"      "est"
## [66] "scelerisque" "lectus,"     "iaculis"
```

- 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.

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

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

This is not the proper binary threshold because there is no differentiation between the levels of crimes.

```
y = (x_2 != "none")
y = as.numeric(y)
y
```

```
##   [1] 0 0 1 1 0 1 1 1 1 1 1 0 1 1 1 1 1 1 1 0 1 1 1 1 1 0 1 1 0 1 1 1 0 1 0
##  [36] 1 1 1 0 1 1 0 1 0 0 1 0 1 1 1 1 1 1 0 0 1 1 1 1 1 1 1 0 0 1 0 1 1 1 1
##  [71] 1 0 1 1 1 1 1 0 1 0 1 1 0 1 0 0 1 1 1 1 1 1 1 1 0 0 0 1 0 0
```

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

```
x = factor(c("none", "infraction", "misdimeanor", "felony"))
x
```

```
## [1] none        infraction  misdimeanor felony
## Levels: felony infraction misdimeanor none
```

```
z = x_2
```

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

```
q = as.numeric(z == "infraction")
r = as.numeric(z == "misdimeanor")
s = as.numeric(z == "felony")
m = cbind(q,r,s)
m
```

```
##        q r s
##   [1,] 0 0 0
##   [2,] 0 0 0
##   [3,] 0 0 1
##   [4,] 0 1 0
##   [5,] 0 0 0
##   [6,] 0 1 0
##   [7,] 1 0 0
##   [8,] 0 1 0
##   [9,] 1 0 0
##  [10,] 0 0 1
##  [11,] 0 0 1
##  [12,] 0 0 0
##  [13,] 1 0 0
##  [14,] 1 0 0
##  [15,] 1 0 0
##  [16,] 1 0 0
##  [17,] 1 0 0
##  [18,] 0 1 0
##  [19,] 0 0 1
##  [20,] 0 0 0
##  [21,] 0 1 0
##  [22,] 0 1 0
##  [23,] 1 0 0
##  [24,] 0 1 0
##  [25,] 0 0 1
```

```
##  [26,] 0 0 0
##  [27,] 0 1 0
##  [28,] 0 0 1
##  [29,] 0 0 0
##  [30,] 0 0 1
##  [31,] 0 1 0
##  [32,] 1 0 0
##  [33,] 0 0 0
##  [34,] 0 0 1
##  [35,] 0 0 0
##  [36,] 0 0 1
##  [37,] 0 1 0
##  [38,] 1 0 0
##  [39,] 0 0 0
##  [40,] 0 1 0
##  [41,] 0 0 1
##  [42,] 0 0 0
##  [43,] 0 0 1
##  [44,] 0 0 0
##  [45,] 0 0 0
##  [46,] 0 0 1
##  [47,] 0 0 0
##  [48,] 0 1 0
##  [49,] 1 0 0
##  [50,] 0 1 0
##  [51,] 1 0 0
##  [52,] 1 0 0
##  [53,] 0 0 1
##  [54,] 0 0 0
##  [55,] 0 0 0
##  [56,] 0 1 0
##  [57,] 0 1 0
##  [58,] 1 0 0
##  [59,] 0 1 0
##  [60,] 0 1 0
##  [61,] 1 0 0
##  [62,] 0 0 1
##  [63,] 0 0 0
##  [64,] 0 0 0
##  [65,] 0 0 1
##  [66,] 0 0 0
##  [67,] 0 0 1
##  [68,] 0 1 0
##  [69,] 0 1 0
##  [70,] 1 0 0
##  [71,] 0 1 0
##  [72,] 0 0 0
##  [73,] 1 0 0
##  [74,] 1 0 0
##  [75,] 0 1 0
##  [76,] 0 0 1
##  [77,] 0 0 1
##  [78,] 0 0 0
##  [79,] 1 0 0
```

```
##  [80,] 0 0 0
##  [81,] 1 0 0
##  [82,] 0 0 1
##  [83,] 0 0 0
##  [84,] 0 0 1
##  [85,] 0 0 0
##  [86,] 0 0 0
##  [87,] 0 1 0
##  [88,] 0 1 0
##  [89,] 1 0 0
##  [90,] 0 1 0
##  [91,] 0 0 1
##  [92,] 1 0 0
##  [93,] 0 1 0
##  [94,] 0 1 0
##  [95,] 0 0 0
##  [96,] 0 0 0
##  [97,] 0 0 0
##  [98,] 0 1 0
##  [99,] 0 0 0
## [100,] 0 0 0
```

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

The sum of each row should be 1 or 0, depending if there was a crime or none. The sum of all the rows should be 100 minus the number of "none"'s.

```
sum(m) == (100 - sum(z == "none"))
```

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

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

The column sum should be the number of each type of crime, roughly 25.

```
print(sum(m[,1]))
```

```
## [1] 22
```

```
print(sum(m[,2]))
```

```
## [1] 27
```

```
print(sum(m[,3]))
```

```
## [1] 21
```

- 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.

```
norm = rnorm(100, mean = 17, sd = sqrt(38))
unif = runif(100, min = -10, max = 10)
pois = rpois(100, lambda = 6)
exp = rexp(100, rate = 9)
binom = rbinom(100, size = 20, prob = 0.12)
bernoul = rbinom(100, size = 1, prob = 0.24)
m = cbind(norm, unif, pois, exp, binom, bernoul)
m
```

```
##                 norm             unif pois              exp binom bernoul
##   [1,]  3.0496834188  4.0678531629965   13 0.00902682216663     2       0
##   [2,] 12.7008273648  1.2793237762526    3 0.12519242686477     2       0
##   [3,] 17.2537919670  2.2291738353670    6 0.09717641196579     1       1
##   [4,] 10.8194168533 -3.9425821043551    3 0.14077433487968     4       1
##   [5,]  6.0185529637 -9.1169884381816    5 0.06740950974118     4       0
##   [6,] 11.5659936232 -7.2469268273562    5 0.02462432926728     3       0
##   [7,] 16.1915039091  2.3430039733648    8 0.03703177125297     3       0
##   [8,]  8.8216216585 -9.9561504088342    5 0.08716607662751     3       0
##   [9,]  5.7020781089 -8.7830963497981    5 0.06185635965731     2       0
##  [10,] 22.2694149360  7.1266610547900    3 0.05882097547874     0       0
##  [11,] 22.4963119535 -8.0392885021865    8 0.11726352586464     3       1
##  [12,] 22.3190425659  7.1974046062678    4 0.37623182185611     2       0
##  [13,]  3.1447643367  2.4679639237002    5 0.06712952530425     4       0
##  [14,] 19.2294038147 -9.8997832462192    7 0.45257695609313     1       0
##  [15,] 23.1256348538 -0.3420170163736    7 0.05300722452294     0       1
##  [16,] 17.5393099625  0.1474394835532    6 0.00633260571501     0       0
##  [17,] 14.3089777374  6.7466875584796   11 0.02106145484787     2       0
##  [18,]  8.4686137643  2.5904814526439    2 0.32994325374352     2       0
##  [19,] 28.3515480665  3.0747832311317    6 0.06324839167711     3       0
##  [20,] 17.2449279051  7.6607124460861    2 0.26796020545321     1       0
##  [21,] 11.7574594500  6.6382713802159    6 0.18582254812844     1       1
##  [22,] 16.4282067789  3.9823368098587    5 0.28766421330727     1       0
##  [23,] 19.6313926357  6.7055486328900    3 0.00467219953518     5       1
##  [24,] 12.1546940267 -9.3071009172127    5 0.06384409317333     2       0
##  [25,] 18.7118982454 -1.3807900669053    8 0.13366980198304     2       0
##  [26,] 26.7290887957 -2.2123725945130    7 0.16782231379609     4       0
##  [27,] 16.0577920725  5.4616357665509    5 0.02502047803460     2       0
##  [28,] 27.4161615337  4.2901014303789    6 0.08236380421372     3       0
##  [29,] 22.1637507831  4.5087050786242    6 0.04540725665534     1       0
##  [30,]  3.3783417725  6.9125611428171    6 0.07387397991907     1       1
##  [31,]  5.8973620433  7.6219922630116    7 0.08501104302200     6       0
##  [32,] 13.8830539072 -1.3457824476063    5 0.12055775412809     1       1
##  [33,] 11.2154982703 -5.5868708109483    8 0.04155117112936     3       0
##  [34,] 21.2559469927 -8.6375112878159    8 0.08546616093706     1       0
##  [35,] 16.1284315292 -9.6785698365420    6 0.18057267403781     4       0
##  [36,] 15.1843252035 -6.2509147776291    3 0.15813437333563     3       0
##  [37,] 21.7216322803 -7.9473146144301    3 0.10670119414135     2       0
##  [38,] 16.5333942682 -1.8882898287848    8 0.04272750071767     2       1
##  [39,] 20.0535266531  0.9870556602255    4 0.12563240856043     0       0
##  [40,] 11.4584718090  7.3367327731103    7 0.12006897540917     1       0
##  [41,] 20.4647979925 -5.5076890625060    5 0.05973401106894     2       1
##  [42,] 21.4721169560 -8.6360300658271    6 0.00965120097247     4       1
##  [43,] 24.7240493566 -0.3425573324785    5 0.10294134252134     2       0
##  [44,]  8.0433532483 -1.3583637587726    3 0.34583076336106     1       0
##  [45,] 21.7638106411 -5.9318406321108    5 0.05283203197297     2       0
##  [46,] 16.5008383514 -0.1265132566914    9 0.00855216924101     4       0
##  [47,] 15.0806571558 -2.6856896840036    1 0.07281274260539     1       1
##  [48,] 10.6532580062  5.6574820680544    4 0.00932767107669     1       0
##  [49,] 21.7218424971 -0.6850036745891    6 0.25602187980636     2       0
##  [50,] 21.7779664467 -0.3323932597414    7 0.07195079890597     0       0
##  [51,] 21.1918749029 -0.8029590640217    8 0.04001023683811     1       0
##  [52,] 14.1333352900  9.7626013355330    6 0.00624444910015     1       0
##  [53,] 10.3098347296 -9.9105740478262    5 0.02151518075392     4       0
```

```
##   [54,] 15.8880664245  7.2437222907320    5 0.00229079261162     2       1
##   [55,] 28.1917681564  0.1258689351380    4 0.00042781321746     1       0
##   [56,]  7.8744845794 -7.9282409511507    2 0.09702766541339     5       0
##   [57,] 18.7078254294  6.8017819244415    7 0.08754862523688     3       1
##   [58,] 18.5387579098  5.4501551995054    4 0.00494325949833     1       0
##   [59,] 17.2452492848 -2.6955916546285    3 0.05068654008210     4       1
##   [60,] 16.7614059178  4.9850276578218    4 0.07634236234137     2       0
##   [61,] 18.7182551609 -9.5361331244931   10 0.03111766646099     1       0
##   [62,] 12.4032219711  0.6426338991150    6 0.07008804189455     2       0
##   [63,] 14.5183508448 -1.0413387930021    9 0.08673580425014     1       1
##   [64,] 24.8736476016  4.1424611071125    6 0.14920424835083     3       1
##   [65,] 18.9760880519 -8.4220936801285    2 0.03352172647615     2       1
##   [66,] 11.1640530670  0.7974479813129    7 0.02325518378846     2       0
##   [67,]  9.9395704749 -7.3499703034759    6 0.09334339022642     0       0
##   [68,] 18.1952350535 -0.0043672230095    8 0.46218941150951     3       0
##   [69,]  7.9090758893  9.8367141140625    1 0.06524484355910     1       1
##   [70,] 20.1533797987  5.1930083520710    9 0.09482713315628     6       1
##   [71,] 22.0995371412 -2.3605212103575    6 0.22586624137124     3       0
##   [72,] 22.7854583005 -1.8009986728430    4 0.02283498820745     2       0
##   [73,]  9.6565542409  4.8640768229961    8 0.12952962113698     5       1
##   [74,]  8.2360119852 -4.9848112370819    7 0.03755688915650     2       0
##   [75,]  7.1553609575  8.7684937380254    8 0.03077429527831     1       1
##   [76,] 14.6422620975  6.6292962990701    7 0.01495871769144     4       0
##   [77,] 11.2364874649  1.4763554139063    4 0.26675457112747     0       0
##   [78,] 17.2814736239  4.0125482296571    9 0.16790004404777     1       0
##   [79,] 19.7174507773 -0.4897531820461    6 0.40429923991512     4       0
##   [80,] 17.5676468521  2.5479708053172    3 0.18172245306168     4       0
##   [81,] 11.3363334557 -2.5432448415086    5 0.26237800785303     3       0
##   [82,] 14.8520616717  3.9712002314627    7 0.00780843151733     3       0
##   [83,] 21.3035507971  9.0280742058530    6 0.08652790169696     1       1
##   [84,] 18.1926126121 -5.3095643175766    5 0.02364510134992     1       0
##   [85,] 12.5901442095 -1.7216428369284    9 0.12217987131688     2       0
##   [86,] 21.0402447050  6.5550987469032    7 0.03867474420824     4       1
##   [87,] 14.3298088682 -5.3273121267557    8 0.03637234080169     3       0
##   [88,] 15.3233105659 -9.7160708764568    7 0.09391211262861     1       1
##   [89,] 17.9309088721  7.9768939642236    7 0.05342799186871     1       0
##   [90,] 10.5804287613 -2.3998189158738    3 0.13372750390716     1       1
##   [91,] 27.4952882042 -8.0242897383869    7 0.01269935464693     4       0
##   [92,] 20.3242691639 -3.5955234896392   10 0.23716659562178     2       0
##   [93,] 10.1186430438 -2.6667185919359    6 0.32532086242030     4       0
##   [94,] 21.8224259896  5.8906188188121    9 0.05316918617528     2       1
##   [95,] 16.7405216669 -1.0180986626074    7 0.00173395701648     1       1
##   [96,] 21.3011654305  5.6045898888260    8 0.06291264978548     2       0
##   [97,] 28.8143026246 -7.4079628754407    4 0.06368043056379     5       0
##   [98,] 19.2998901864 -2.3612919868901    6 0.07144946796406     2       0
##   [99,] 15.0042174611 -4.8457459080964    6 0.28953840561466     1       0
##  [100,] 17.8240079516  6.5008094906807    4 0.14449658099509     2       0
```