STAT243 PS1

Wenrui Wang

9/6/2021

```
# 2(a)
## Using function rnorm() to generate random number from normal distribution
# I firstly generated a file with 2000000 numbers and take a look at the size of the file
# and then increment the numbers to approach a file of 100 mb
RandomNum<-matrix(rnorm(5476000),ncol=10)
# Write the random numbers we generate into the file called Random_Num.csv
write.csv(RandomNum,"Random_Num.csv")
```

```
# 2(b)
# Start the clock
ptm<-proc.time()
df=read.csv(file="Random_Num.csv",nrows = 100000)
# Stop the clock
length_of_time<-proc.time()-ptm
# Take a look at the time elapsed
length_of_time
```

```
##    user  system elapsed
##    1.85    0.10    1.97
```

```
# Test whether we can load the data more quickly if we use colClasses
# Start the clock
ptm1<-proc.time()
# Stop the clock
length_of_time1<-proc.time()-ptm1
length_of_time1
```

```
##    user  system elapsed
##       0       0       0
```

```
## Using colClasses enables us to read the data much faster!
```

```
# 2(c)
# Read the data from the middle of the file
nrow(RandomNum)
```

```
## [1] 547600
```

```r
# Start the clock
ptm2=proc.time()
test1<-read.csv(file="Random_Num.csv",skip=273800,nrows=100)
time1=proc.time()-ptm2
time1
```

```
##    user  system elapsed
##    0.84    0.22    1.19
```

```r
# Start the clock
ptm3=proc.time()
test2<-read.csv(file = "Random_Num.csv",nrows = 100)
time2=proc.time()-ptm3
time2
```

```
##    user  system elapsed
##       0       0       0
```

```r
## It shows that we didn't really skip 273800 lines since reading 1000 rows from the
# middle of the file is much slower than reading the first 1000 rows of data
```

```r
# 2(d)
diff<-read.csv(file="Random_Num.csv",nrows=100000,colClasses = c("NULL",rep('double',10)))
# Open a connection to file
con = file("Random_Num.csv","r")
# Read the first half of the file
#rep(read.csv(con,nrows=100),2730) I commented this part of code because it will make pdf extremely lar
# Start the clock
ptm4=proc.time()
read.csv(con,nrows=100)
```

```
##       X           V1          V2           V3          V4          V5
## 1     1 -0.982697431  1.17369875 -1.404964449 -0.22716701 -0.76516869
## 2     2  0.477895205 -0.04380547 -0.355238003 -0.56991547 -0.74984293
## 3     3 -3.055727171 -0.31092603  0.437167885 -0.20604601 -0.11085942
## 4     4 -1.018404432 -0.71427847 -1.801988189  0.91666959  1.34185427
## 5     5  0.183734766 -0.66415641  0.414099351  1.11921501 -0.14641656
## 6     6  0.087590865  0.40981162  0.458391960  1.72765300  0.18000225
## 7     7  0.737753003 -0.19570936  0.318725025  1.27573138 -1.81204281
## 8     8 -1.277570219 -0.90486171  0.111920334 -0.62390230 -0.24388503
## 9     9  0.888146659  0.59319515 -0.786686334  0.12688547 -0.60397599
## 10   10 -0.905456571  0.05425440 -0.581542236  0.56926865  0.22046063
## 11   11  1.082748397 -1.79795788 -1.670373499  0.04854722  0.53457350
## 12   12 -0.901733919 -0.57646348 -0.528677413  0.90348732  1.20025887
## 13   13  0.670099880  0.08415676 -1.124204724 -1.51382789  0.83818707
## 14   14 -0.006744739  0.96616635 -0.554894638  0.67567535  1.98758855
## 15   15 -0.558533566 -0.71073877 -0.505151885  0.43402883  0.41906541
## 16   16  0.299663228  0.98012521  0.166969313 -1.23179110 -0.16587559
## 17   17  0.312964350 -0.32578111  0.449886306 -0.05051079 -0.82600985
## 18   18  0.769329372  2.23901690 -0.982450004  0.77240840  0.07775818
## 19   19 -0.720710043 -0.62526823 -0.005749462  0.72577038 -0.13757561
## 20   20  0.058955134 -1.58818194  0.832390614 -0.83799472 -0.74021745
## 21   21  0.820391443  1.71790756  0.576725196  1.14911353  3.43719021
## 22   22  0.578927489 -0.91781994 -0.762929314 -0.49463049  0.27641217
## 23   23  3.699953505 -0.52280884  0.276147970  1.78294153 -0.18450194
## 24   24  1.279295026 -1.24090118  0.484541526  2.81846128 -0.67500409
## 25   25 -0.032895467  0.74385728  1.090166818  0.37551218  0.77209693
## 26   26 -1.105231791  0.22653564 -0.137044843  0.70930554  0.27630033
## 27   27 -0.723380058  1.26534012 -0.735110498 -0.47183165 -1.85890990
## 28   28  0.395659440 -0.57099079  1.602912185 -1.13818224  0.73803752
## 29   29  1.052823656 -0.20972557 -1.463143776 -1.14692416 -0.12260394
## 30   30  0.362617333  1.45611212  0.286106627 -0.36640659 -0.30229627
## 31   31  0.290694234  1.30597605  0.322641347  0.79524334 -0.66859949
## 32   32  0.403479387 -0.67833639 -1.023726455  1.33644076  1.71323002
## 33   33  0.362092807 -0.72987620  0.275472516 -0.12414081 -0.17155061
## 34   34  0.942065346  0.09098140  1.099556475  1.04688989 -0.46017700
## 35   35  0.552610461  0.75908253  1.178652586 -0.68021831  0.76198189
## 36   36 -0.916498316 -0.11413032  0.045182673 -0.03852634 -2.25545777
## 37   37  0.223629768  1.11790014  0.176607904  0.26210711 -2.89643935
## 38   38  0.093450467 -0.52811377 -0.660416896 -0.47133182  0.32011721
## 39   39  0.808028077 -0.21532375 -1.346853344 -0.03248779  1.43153630
## 40   40  0.349566029 -1.84415604  0.018172238 -0.09614959 -1.11431525
## 41   41 -1.993695601  0.69704656 -0.022965649  0.94713167 -0.63834414
## 42   42 -0.420577459  2.12532949 -1.208072295 -0.50947939  1.32793205
## 43   43  0.163023676  1.18033132 -0.763665518  1.07785135 -1.97377435
## 44   44  0.473528792  0.04021948  0.166183598  1.10794243 -0.51866316
## 45   45  0.486295295 -1.61724070 -1.527532460  2.21306731  0.22609664
## 46   46 -1.015086044  0.76160087 -1.616327045  0.69731414  1.28907327
## 47   47  0.189809214  0.19410241  0.620786353 -0.77101384  0.30653147
## 48   48  1.235891083 -0.85763520 -0.590121925  0.49187020  0.73949664
## 49   49 -0.753742282 -1.19858429  1.366175210  1.17537290  0.14776222
## 50   50  0.614802265 -0.71854172 -2.340649972 -1.62614653  0.62513584
## 51   51 -2.726622660  0.63142037 -0.104811665  0.24311480 -0.14610621
## 52   52 -0.920660856 -0.18169302 -0.274390392 -0.19746166  0.35706456
## 53   53  0.638964768 -1.66899544  0.498843116 -0.25440728 -1.57234613
## 54   54  1.531698446  1.99522895  0.566610933  0.99656120 -1.62935640
## 55   55 -0.627642845  0.05209660  0.485765126  0.11814176 -0.27176097
## 56   56 -0.694136065 -1.47029904 -0.035953859 -0.06871489 -0.74576100
## 57   57 -0.119533611 -3.11526957  0.381560426  0.31527159 -2.68650078
## 58   58 -2.704574386  1.11919478  0.541288166 -1.14475011  0.46846001
## 59   59  0.731934138 -0.19698669 -0.774916388 -2.03195909  2.73821310
## 60   60 -2.136033869  0.48804992  1.665625377  2.59052216  0.23031465
## 61   61 -0.986407279  1.23034633 -0.111153610 -1.02618511 -0.57324175
## 62   62  0.431185922  0.01812814  0.056387799  0.12155576 -1.06095126
## 63   63 -0.576580779  2.00245742 -1.308624007 -1.72476965  0.70029589
## 64   64  1.033742633 -0.69119723  0.638376397  0.04669473  0.81125213
## 65   65 -1.287292543 -1.17664356  0.080365426 -1.00011871  1.84639297
## 66   66 -0.069938858 -1.68424959 -1.571693145 -0.68371341 -0.78532100
## 67   67  1.521989338 -0.31537642  0.195760082 -1.12728602 -0.37010707
## 68   68  0.479134008  0.10524150  1.073167365  0.06557754 -1.69772377
## 69   69  1.172650123 -0.01004709  2.563302243 -0.19170151 -0.26647631
## 70   70 -0.391077155  0.57733170 -0.741315470 -0.66988607  0.10335680
## 71   71 -1.696958669 -0.93935346 -1.249622083 -0.49123538 -1.09154038
## 72   72  0.307150118 -1.09125549 -2.571393557  0.07741106  1.27354150
## 73   73  0.287792688  0.23568540 -0.261698868  0.09841319  1.03850740
## 74   74  0.784061160 -0.89397423  0.203885929  0.67061813  0.32701036
## 75   75 -1.376712511 -0.47659246  0.487030819  2.07800551  0.58750913
## 76   76  0.402360616  1.52632501  0.040564745 -1.39549496 -0.11026798
## 77   77  0.339286052  1.05648341  1.691483747  0.34969775  0.64876489
## 78   78 -0.550871543 -0.23407921  1.055195192  1.48502881  0.26961215
## 79   79 -1.047350841 -0.01352909  2.002702760  0.61513263 -1.66465534
## 80   80 -0.128156433  2.17202757  0.869404232 -0.48813973 -0.02573702
## 81   81  1.885534758  0.88880045 -0.409596625  1.17687947 -0.17783541
## 82   82 -0.722459837 -0.94961249 -0.704355145  1.08892979 -0.44084518
## 83   83  0.934352851  0.01950793  0.468460405  0.21739061 -0.32005741
## 84   84  0.370477703  0.01390993  0.782453711  1.07797479 -0.56027815
## 85   85 -0.587894667 -1.12899016  2.008042596 -2.44822403 -1.03314340
## 86   86  0.433044847  0.20361202 -1.220933308 -0.54610066 -1.21830558
## 87   87 -1.180701259  0.90132391 -0.120171944  0.55307105  0.74971427
## 88   88  0.663600489  0.68588612 -0.353644026 -0.13993150  1.04122736
## 89   89  0.548117127  0.99322121 -0.245543096 -0.54075068  0.18123028
## 90   90 -0.005049628 -0.63215698  0.579185711  0.79833032 -0.54539642
## 91   91  0.772561149 -2.14818520 -0.371973731  0.04623124  0.36758761
## 92   92 -1.521547766  0.77458639  1.294688888 -1.81353781  2.28934185
## 93   93 -0.032857692 -0.47089425 -0.723978779  0.37669753 -0.43178301
## 94   94 -0.810511785 -1.01998165  1.145254033  0.33127397  1.06001182
## 95   95  0.856602599 -0.68906448 -0.274088472  2.25592563  1.38501270
## 96   96 -0.991232611  1.06014712  0.604875148  1.16935342 -0.74337427
## 97   97  0.145386709 -0.49936590 -0.093003535  0.39995637  0.30147040
## 98   98 -0.685316598  1.10617198 -2.033662422 -0.61227202 -2.06869115
## 99   99 -1.274007510 -0.02617196  1.641065356  0.30465202 -1.81813698
## 100 100 -0.813702633 -0.70770222 -0.459415472  0.24550564  1.49818431
##              V6          V7          V8           V9         V10
## 1    1.94164956  0.33543776  0.01246608  0.144476034 -0.76720358
## 2   -0.22145159  0.34423489 -0.75325493 -0.588982518  0.48047376
## 3    0.50258971  0.23846849  0.78271294 -2.004248886 -0.68321577
## 4   -0.22242291 -0.95306215  0.85037842  0.298449471 -0.82331950
## 5   -0.48239677  0.49514552 -0.22656015  0.714724430 -0.77259369
## 6   -1.65058807 -0.27677221 -0.25570083  0.314506497  2.67308655
## 7   -0.06588861 -0.89959104 -0.81515533  1.430959642  0.22390554
## 8   -0.52984040  0.11089055  0.98404715  0.688900249 -1.45271237
## 9    0.56426295 -0.92741396  0.27808429 -0.510132870  0.06889569
## 10   0.59227805 -0.17377534 -0.02951135  1.358189615 -1.01359791
## 11  -1.63280543  0.15665666  0.13523867  1.073758192 -1.87903103
## 12  -1.41782172 -0.38102354  2.49552541  1.607909596 -1.08867255
## 13  -2.21125500 -0.58189816  0.31288534 -2.727152086  0.30020244
## 14  -1.33227078 -0.77551190  0.12806687 -0.009464864 -1.86939226
## 15   0.25188819  0.57248369 -1.36087080  0.051948777 -0.84572255
## 16   0.06729447 -0.29285732  1.22341964 -0.227685483 -1.22355322
## 17  -1.12082271 -0.10123843  0.73576120  0.214078862 -0.25612016
## 18   1.38156795  1.08015828 -0.12431466 -0.900354705  0.07366177
## 19  -0.10146595 -1.33505964 -0.74535604 -0.577165205 -1.33358752
## 20  -0.62581157 -0.12848242  0.25394973  0.386908409 -0.81873758
## 21  -0.11865815 -0.96381967  0.65102873  0.631814324  0.08249409
## 22  -1.54802569 -0.02833912 -0.40022475 -0.776333109 -1.98900913
## 23  -0.21855249 -0.43548581  0.22934387 -0.099414942  0.18643591
## 24   0.41447616  0.57288712  0.08854941 -0.425942885  1.10934441
## 25  -1.02461433 -0.12502457 -0.22112908 -0.251222546  0.79798701
## 26   0.07243982  0.20366918 -0.49001895  0.846230401 -0.49808466
## 27  -1.19777890  1.68036048 -2.33831590  1.855013976 -0.34118264
## 28   3.14911436 -1.75057976  0.98774250  1.960284900 -2.37173584
## 29   0.51999939 -0.97099094 -0.25690741 -0.254697334  0.17659479
## 30   0.05292286 -0.32528281  0.02458769 -0.308875427  0.47860211
## 31   1.06780430  0.01025342 -0.18638765 -0.753345018  0.99687297
## 32   1.38276456 -2.84040185 -2.08875143  0.160343281  0.99428936
## 33   2.33854832 -2.04549411  0.84651192  1.174166650  0.84771374
## 34  -0.59587252 -0.23500125  0.02788954  0.203927567  1.11306906
## 35   1.95146446  0.50807859  0.38712677  0.436668170 -1.36149606
## 36  -1.38382477  1.15081076  0.36187557  0.559197622 -1.37978887
## 37   1.50239729 -0.04232523  0.45707743 -0.138952397  0.16702699
## 38   0.15896524  0.40728468 -2.45911392 -1.162495393 -1.20546150
## 39  -0.22541908  1.16007658  1.21526011 -1.175669439  1.72231397
## 40   1.11997792 -1.23516047 -1.79784803  0.070663730 -0.89872526
## 41  -1.67891914  0.45033165 -0.94064501  0.473641927 -0.90859054
## 42  -0.82282070 -0.23208592  0.89885489 -0.471451315  0.02167156
## 43  -1.38580835 -1.02993542  0.85366707 -1.005826712 -1.82428391
## 44   0.66182292 -1.04315909  0.63177931 -1.537097437  0.05193760
## 45  -1.06363448  2.81264227 -0.83857095  2.053509313 -0.07863522
## 46  -0.38470782 -1.80266363 -1.43683465 -0.910637371  0.02503078
## 47   1.30572985  0.50116983  0.19159008  1.091054379  0.36274530
## 48  -1.76792137  0.85931064  0.42901810  0.564605702  0.24557276
## 49   1.23011764 -0.65852565 -0.19421965  0.575049483  0.70664962
## 50   0.48837447 -1.61212612 -0.45990749 -0.184986190 -0.44279045
## 51  -1.41593436 -1.28724298 -0.35657235  0.765490046 -0.25109248
## 52   0.24089626  0.23076180 -0.50350599  1.155243218 -0.40034520
## 53  -1.39947912  1.15818250 -0.43772153  0.148120324  2.41483858
## 54   0.19969021 -0.22058135 -0.59493424 -0.964837226 -1.95317442
## 55  -0.14444194  0.55297751  0.26182045  0.676833557 -0.64008322
## 56  -0.34110336 -1.23775125 -0.75710950  0.191606214 -0.01193976
## 57  -0.37159978 -2.44387443 -0.15523493  0.911208673 -0.45516704
## 58   1.02109625  0.82465940 -1.06426929 -0.536111835  1.32195529
## 59  -0.70404503  1.29488732  0.71796930  0.450218714 -1.38638481
## 60  -0.14627891  0.05383627 -0.41006206 -0.685847211 -0.16110413
## 61   1.13964352 -0.03872904  0.26885762  0.343820940 -1.71811375
## 62   2.30373144 -1.35361192 -0.53267823  1.239297279 -0.14766466
## 63   0.04556446 -2.10037773  2.67474651  0.082127711 -1.47861387
## 64   0.58732851 -0.16634287 -0.03736114 -0.577914093 -0.42359696
## 65  -0.24768877 -0.68486352 -1.75880734 -0.840898002  0.69693086
## 66   0.54172785  0.39429597  0.60513059 -1.550115606 -0.81304981
## 67   1.57887760 -0.47942811  0.20717028 -0.727848194  0.18811857
## 68   0.07346535 -0.60476323  0.11790411  0.043950224  1.03428540
## 69   1.62028023  0.49375995 -2.07980056  1.372648102 -1.97387668
## 70  -0.10338676  1.21501967 -0.36879115  0.354580957 -0.55093967
## 71   0.01141342 -1.67498593 -0.38367108 -0.848804749 -1.76410401
## 72  -1.60386778 -0.34600060  0.72584673  0.662850534 -0.62823929
## 73   0.72828580  0.54131431 -1.15029932 -0.320919298 -0.32552864
## 74  -0.43494537 -0.05346555  0.34143450  0.882651988 -0.20989568
## 75   1.15451318 -0.90413117 -0.26784248 -1.458800026  1.61656400
## 76   0.40756268 -0.73268960 -0.76681041  1.152479492  0.10808940
## 77  -0.62251432  1.72252113  0.29482337 -1.146445248  0.59537459
## 78   1.73601351  0.98089235 -0.93288330 -1.523712525  0.52282602
## 79  -0.86620570 -0.19706122  0.30167975 -1.677501147  0.04758765
## 80  -1.29506240 -0.73927418 -1.10548385 -1.526289738  0.89921250
## 81   0.15276635 -0.31177278  0.98386026  0.205865762 -1.39378144
## 82  -1.53312862  0.48016677 -1.19887533  2.022594765 -0.26823519
## 83   1.24838972  0.30543657  0.24745107  1.224847131 -0.73298620
## 84   0.71949472  1.48242111  0.90479823  0.245983109  0.27238269
## 85  -2.25897570  0.48291868  0.12362240 -0.302127453  1.44512059
## 86  -2.24560188  0.29826066  0.72878604 -1.280812006  1.14419396
## 87  -0.09361250 -0.63986965 -0.65344002  1.192714160 -0.08584738
## 88  -0.81876112 -0.49110309  0.62996369 -1.591695856  0.34107622
## 89  -0.83903235 -0.08388972  0.33655412 -0.416902086  1.58125949
## 90   0.29240507  1.46567646  0.46426784  0.129403791  0.50088573
## 91   0.64866385  0.01462381 -1.12592368 -1.827079857  0.54631898
## 92  -0.30161277  0.68503344 -2.14262237 -0.113811435 -0.38468202
## 93   1.12510734  1.59272752 -1.52227775  1.845454540  0.52073456
## 94  -0.83121838  1.99925342  0.32592046 -0.876226009  0.92651078
## 95  -1.05682479 -0.17245504  1.26953963 -0.502064338 -0.25627355
## 96  -1.15118543  0.61779396  0.74169427  0.636945033 -0.54818985
## 97   0.11777818 -1.09449824  0.81541897  0.930531227  1.51558568
## 98  -1.35504544  0.22298475 -0.52920223  2.306418518 -0.25874595
## 99  -0.05446759 -1.63989339  0.72867525  0.906592640  1.70440205
## 100  0.13071028 -0.79259001 -0.04178753 -0.121503926  0.34628721
```

```
# Stop the clock
time3=proc.time()-ptm4
time3
```

```
##    user  system elapsed 
##       0       0       0
```

```
close(con)
## By using a connection, we read the same lines of data from the middle of the data file more quickly
```

```
# 2(e)
save(RandomNum,file="RandomNum.rData",compress = FALSE)
file.info("RandomNum.rData")$size
```

```
## [1] 43808105
```

```
# 2(f)
Same_value<-matrix(2,547600,10)
save(Same_value,file = "SameNum.rData",compress=TRUE)
file.info("SameNum.rData")$size
```

```
## [1] 63877
```

```
# The size of the file with a single repeated value is significantly smaller
# than the file with same number of random values.
#"Compression" allows me to keep data files on disk compressed saving space and time!
```

```
# 3
## Because it is more readable and more convenient for me to debug the functions.
# Besides that, I pay attention to assigning a meaningful name to each of the functions. If the functio
# if i am not sure about whether the function I wrote is correct, I will test the function and check
# whether the output is valid.
```

```r
# 4(a)
library(magrittr)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(rvest)
  url_webscrap<-function(artist,title){
  string1=strsplit(title,split=" ")[[1]]%>%paste(collapse="+")
  first_url<-paste("http://www.mldb.org/search?mq=",string1
                   ,"&si=2&mm=0&ob=1",sep = "")
  html=read_html(first_url)
  all_a_name<-html %>% html_elements("a")%>%html_text()
  all_a_value<-html %>% html_elements("a")%>%html_attr("href")
  song_url<-paste("http://mldb.org/",all_a_value[which(all_a_name==artist)+1],sep="")
  return (song_url)}

# Test the output of the function
m<-url_webscrap("Adele","Someone like you")
m
```

```
## [1] "http://mldb.org/song-248319-someone-like-you.html"
```

```r
n<-url_webscrap("Sarah Connor","Just One Last Dance")
n
```

```
## [1] "http://mldb.org/album-20721-key-to-my-soul.html"
```

```r
# 4(b) based on the code written in 4(a),we made some revision to get the required function
# Firstly, I just copied the function from 4(a)
  datascrap<-function(title, artist){
  string1=strsplit(title,split=" ")[[1]]%>%paste(collapse="+")
  first_url<-paste("http://www.mldb.org/search?mq=",string1,"&si=2&mm=0&ob=1",sep = "")
  html=read_html(first_url)
  all_a_name<-html %>% html_elements("a")%>%html_text()
  all_a_value<-html %>% html_elements("a")%>%html_attr("href")
  song_url<-paste("http://mldb.org/",all_a_value[which(all_a_name==artist)+1],sep="")
  return (song_url)}

Advanced_search<-function(title,artist){
  string2=strsplit(title,split=" ")[[1]]%>%paste(collapse="+")
  second_url<-paste("http://www.mldb.org/search?mq=",string2,"&si=2&mm=0&ob=1",sep = "")
  song_link<-datascrap(title,artist)
  html2<-read_html(song_link)
  frame1<-html2%>%html_elements("#thelist")%>%html_table()
  song_lyrics<-html2%>%html_elements("p")%>%html_text()
  album_sol=frame1[[1]][2,2]
  artist_sol=frame1[[1]][1,2]

  return(c(artist_sol,album_sol,song_lyrics))}
```

```
# 4(C)
# Firstly, I got the code from part(b)
  datascrap<-function(title, artist){

  string1=strsplit(title,split=" ")[[1]]%>%paste(collapse="+")
  first_url<-paste("http://www.mldb.org/search?mq=",string1,"&si=2&mm=0&ob=1",sep = "")
  html=read_html(first_url)
  all_a_name<-html %>% html_elements("a")%>%html_text()
  all_a_value<-html %>% html_elements("a")%>%html_attr("href")
  song_url<-paste("http://mldb.org/",all_a_value[which(all_a_name==artist)+1],sep="")
  # Use the length of all_a_name to verify whether the input is valid
  if(length(all_a_name)<=51){
    stop("The input you provided is invalid")
  }
  # Verify whether the lyrics are returned directly from the initial search
  if(length(all_a_name)<=63){
  html99<-read_html("http://mldb.org/album-20721-key-to-my-soul.html")
  all_a_name<-html99 %>% html_elements("a")%>%html_text()
  all_a_value<-html99 %>% html_elements("a")%>%html_attr("href")
  song_url<-paste("http://mldb.org/",all_a_value[which(all_a_name==title)],sep="")
     return(song_url)}
  else{

  return (song_url)}
  }

  Advanced_search<-function(title,artist){
  string2=strsplit(title,split=" ")[[1]]%>%paste(collapse="+")
  second_url<-paste("http://www.mldb.org/search?mq=",string2,"&si=2&mm=0&ob=1",sep = "")
  song_link<-datascrap(title,artist)
  html2<-read_html(song_link)
  frame1<-html2%>%html_elements("#thelist")%>%html_table()
  song_lyrics<-html2%>%html_elements("p")%>%html_text()
  album_sol=frame1[[1]][2,2]
  artist_sol=frame1[[1]][1,2]

  return(c(artist_sol,album_sol,song_lyrics))}
  # Now test whether our function works when the lyrics are returned directly from
  # the initial search
  p<-datascrap("Just One Last Dance","Sarah Connor")
  p
```

```
## [1] "http://mldb.org/song-195467-just-one-last-dance.html"
```

```
  ## It works and returns the lyrics directly

  # Test function Advanced_search to see whether it returns the album, artist, and
  # the lyrics to the screen
  f<-Advanced_search("Someone like you","Adele")
  f
```

```
## $`Song Details`
## [1] "Adele"
##
## $`Song Details`
## [1] "21"
##
## [[3]]
## [1] "I heard that you're settled down\nThat you found a girl and you're married now.\nI heard that yo
```

```
# 5.
# It is ok to scrape the data from mldb.org since the search directory is not disallowed.
# However, it is not allowed to imiate the search to scrape the data from Google Scholar
# unless the data that you scrape is general data such as
# https://scholar.google.com/citations?view_op=metrics_intro.
```

```
#install.packages("tinytex")
tinytex::install_tinytex()
```