Big Data Assignment 1

### Due Date: Wednesday, Feb. 20, 2019, before class

#### A note from your TAs:

Hi! We recognize that this file is a large file and may be a bit overwhelming at first – don’t worry! We’ll be here to help you with any and all questions you may have. With that being said, there are a couple of house keeping notes:

1. For some of the questions below, you’ll see that we’ve included code chunks underneath the question. This is where you’ll type in the code that will be grade. **Please do not** modify the chunk’s properties (aka the results = FALSE) that you’ll see at the top of each chunk. Even with these modifications, you can still run your code and view your specific results.
2. You will also see <br> pieces throughout the document. **Please do not** delete these tags, as they are for formatting purposes. If you want to add text to your responses, please ensure that there is an empty line between your last line and any of the <br> tags.
3. When you submit your assignment, please just submit this file and rename it bigdata\_asst1\_lastname.Rmd

Thank you so much for reading this, and good luck with the assignment!

### Question 1: Using R built-in datasets.

1. Use the R help function to identify 2 built-in datasets. Provide a 1-2 sentence description of one of them. Write down the code to load a built-in dataset for R.

#######Q1a########  
  
#data() # 0utputs all the inbuilt- datasets  
  
AirPassengers

## Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec  
## 1949 112 118 132 129 121 135 148 148 136 119 104 118  
## 1950 115 126 141 135 125 149 170 170 158 133 114 140  
## 1951 145 150 178 163 172 178 199 199 184 162 146 166  
## 1952 171 180 193 181 183 218 230 242 209 191 172 194  
## 1953 196 196 236 235 229 243 264 272 237 211 180 201  
## 1954 204 188 235 227 234 264 302 293 259 229 203 229  
## 1955 242 233 267 269 270 315 364 347 312 274 237 278  
## 1956 284 277 317 313 318 374 413 405 355 306 271 306  
## 1957 315 301 356 348 355 422 465 467 404 347 305 336  
## 1958 340 318 362 348 363 435 491 505 404 359 310 337  
## 1959 360 342 406 396 420 472 548 559 463 407 362 405  
## 1960 417 391 419 461 472 535 622 606 508 461 390 432

# Monthly Airline Passenger Numbers 1949-1960  
  
USJudgeRatings

## CONT INTG DMNR DILG CFMG DECI PREP FAMI ORAL WRIT PHYS  
## AARONSON,L.H. 5.7 7.9 7.7 7.3 7.1 7.4 7.1 7.1 7.1 7.0 8.3  
## ALEXANDER,J.M. 6.8 8.9 8.8 8.5 7.8 8.1 8.0 8.0 7.8 7.9 8.5  
## ARMENTANO,A.J. 7.2 8.1 7.8 7.8 7.5 7.6 7.5 7.5 7.3 7.4 7.9  
## BERDON,R.I. 6.8 8.8 8.5 8.8 8.3 8.5 8.7 8.7 8.4 8.5 8.8  
## BRACKEN,J.J. 7.3 6.4 4.3 6.5 6.0 6.2 5.7 5.7 5.1 5.3 5.5  
## BURNS,E.B. 6.2 8.8 8.7 8.5 7.9 8.0 8.1 8.0 8.0 8.0 8.6  
## CALLAHAN,R.J. 10.6 9.0 8.9 8.7 8.5 8.5 8.5 8.5 8.6 8.4 9.1  
## COHEN,S.S. 7.0 5.9 4.9 5.1 5.4 5.9 4.8 5.1 4.7 4.9 6.8  
## DALY,J.J. 7.3 8.9 8.9 8.7 8.6 8.5 8.4 8.4 8.4 8.5 8.8  
## DANNEHY,J.F. 8.2 7.9 6.7 8.1 7.9 8.0 7.9 8.1 7.7 7.8 8.5  
## DEAN,H.H. 7.0 8.0 7.6 7.4 7.3 7.5 7.1 7.2 7.1 7.2 8.4  
## DEVITA,H.J. 6.5 8.0 7.6 7.2 7.0 7.1 6.9 7.0 7.0 7.1 6.9  
## DRISCOLL,P.J. 6.7 8.6 8.2 6.8 6.9 6.6 7.1 7.3 7.2 7.2 8.1  
## GRILLO,A.E. 7.0 7.5 6.4 6.8 6.5 7.0 6.6 6.8 6.3 6.6 6.2  
## HADDEN,W.L.JR. 6.5 8.1 8.0 8.0 7.9 8.0 7.9 7.8 7.8 7.8 8.4  
## HAMILL,E.C. 7.3 8.0 7.4 7.7 7.3 7.3 7.3 7.2 7.1 7.2 8.0  
## HEALEY.A.H. 8.0 7.6 6.6 7.2 6.5 6.5 6.8 6.7 6.4 6.5 6.9  
## HULL,T.C. 7.7 7.7 6.7 7.5 7.4 7.5 7.1 7.3 7.1 7.3 8.1  
## LEVINE,I. 8.3 8.2 7.4 7.8 7.7 7.7 7.7 7.8 7.5 7.6 8.0  
## LEVISTER,R.L. 9.6 6.9 5.7 6.6 6.9 6.6 6.2 6.0 5.8 5.8 7.2  
## MARTIN,L.F. 7.1 8.2 7.7 7.1 6.6 6.6 6.7 6.7 6.8 6.8 7.5  
## MCGRATH,J.F. 7.6 7.3 6.9 6.8 6.7 6.8 6.4 6.3 6.3 6.3 7.4  
## MIGNONE,A.F. 6.6 7.4 6.2 6.2 5.4 5.7 5.8 5.9 5.2 5.8 4.7  
## MISSAL,H.M. 6.2 8.3 8.1 7.7 7.4 7.3 7.3 7.3 7.2 7.3 7.8  
## MULVEY,H.M. 7.5 8.7 8.5 8.6 8.5 8.4 8.5 8.5 8.4 8.4 8.7  
## NARUK,H.J. 7.8 8.9 8.7 8.9 8.7 8.8 8.9 9.0 8.8 8.9 9.0  
## O'BRIEN,F.J. 7.1 8.5 8.3 8.0 7.9 7.9 7.8 7.8 7.8 7.7 8.3  
## O'SULLIVAN,T.J. 7.5 9.0 8.9 8.7 8.4 8.5 8.4 8.3 8.3 8.3 8.8  
## PASKEY,L. 7.5 8.1 7.7 8.2 8.0 8.1 8.2 8.4 8.0 8.1 8.4  
## RUBINOW,J.E. 7.1 9.2 9.0 9.0 8.4 8.6 9.1 9.1 8.9 9.0 8.9  
## SADEN.G.A. 6.6 7.4 6.9 8.4 8.0 7.9 8.2 8.4 7.7 7.9 8.4  
## SATANIELLO,A.G. 8.4 8.0 7.9 7.9 7.8 7.8 7.6 7.4 7.4 7.4 8.1  
## SHEA,D.M. 6.9 8.5 7.8 8.5 8.1 8.2 8.4 8.5 8.1 8.3 8.7  
## SHEA,J.F.JR. 7.3 8.9 8.8 8.7 8.4 8.5 8.5 8.5 8.4 8.4 8.8  
## SIDOR,W.J. 7.7 6.2 5.1 5.6 5.6 5.9 5.6 5.6 5.3 5.5 6.3  
## SPEZIALE,J.A. 8.5 8.3 8.1 8.3 8.4 8.2 8.2 8.1 7.9 8.0 8.0  
## SPONZO,M.J. 6.9 8.3 8.0 8.1 7.9 7.9 7.9 7.7 7.6 7.7 8.1  
## STAPLETON,J.F. 6.5 8.2 7.7 7.8 7.6 7.7 7.7 7.7 7.5 7.6 8.5  
## TESTO,R.J. 8.3 7.3 7.0 6.8 7.0 7.1 6.7 6.7 6.7 6.7 8.0  
## TIERNEY,W.L.JR. 8.3 8.2 7.8 8.3 8.4 8.3 7.7 7.6 7.5 7.7 8.1  
## WALL,R.A. 9.0 7.0 5.9 7.0 7.0 7.2 6.9 6.9 6.5 6.6 7.6  
## WRIGHT,D.B. 7.1 8.4 8.4 7.7 7.5 7.7 7.8 8.2 8.0 8.1 8.3  
## ZARRILLI,K.J. 8.6 7.4 7.0 7.5 7.5 7.7 7.4 7.2 6.9 7.0 7.8  
## RTEN  
## AARONSON,L.H. 7.8  
## ALEXANDER,J.M. 8.7  
## ARMENTANO,A.J. 7.8  
## BERDON,R.I. 8.7  
## BRACKEN,J.J. 4.8  
## BURNS,E.B. 8.6  
## CALLAHAN,R.J. 9.0  
## COHEN,S.S. 5.0  
## DALY,J.J. 8.8  
## DANNEHY,J.F. 7.9  
## DEAN,H.H. 7.7  
## DEVITA,H.J. 7.2  
## DRISCOLL,P.J. 7.7  
## GRILLO,A.E. 6.5  
## HADDEN,W.L.JR. 8.0  
## HAMILL,E.C. 7.6  
## HEALEY.A.H. 6.7  
## HULL,T.C. 7.4  
## LEVINE,I. 8.0  
## LEVISTER,R.L. 6.0  
## MARTIN,L.F. 7.3  
## MCGRATH,J.F. 6.6  
## MIGNONE,A.F. 5.2  
## MISSAL,H.M. 7.6  
## MULVEY,H.M. 8.7  
## NARUK,H.J. 9.0  
## O'BRIEN,F.J. 8.2  
## O'SULLIVAN,T.J. 8.7  
## PASKEY,L. 8.1  
## RUBINOW,J.E. 9.2  
## SADEN.G.A. 7.5  
## SATANIELLO,A.G. 7.9  
## SHEA,D.M. 8.3  
## SHEA,J.F.JR. 8.8  
## SIDOR,W.J. 5.3  
## SPEZIALE,J.A. 8.2  
## SPONZO,M.J. 8.0  
## STAPLETON,J.F. 7.7  
## TESTO,R.J. 7.0  
## TIERNEY,W.L.JR. 7.9  
## WALL,R.A. 6.6  
## WRIGHT,D.B. 8.1  
## ZARRILLI,K.J. 7.1

# Lawyers' Ratings of State Judges in the US Superior Court

Use the R dataset âSeatbeltsâ to answer the following:

1. What does the dataset contain? There are several ways to figure this out. Use two different ways. One way is just to type âSeatbeltsâ at the R prompt. Other commands to explore are str( ), summary( ), dim( ), nrow(), and ncol( ) where you put the name of the database within the parenthesis. Apply each of these functions to Seatbelts. Furthermore, apply is.na() and is.null() to check to see if there are any missing data from our datasets.

#######Q1b########  
  
Seatbelts

## DriversKilled drivers front rear kms PetrolPrice VanKilled law  
## Jan 1969 107 1687 867 269 9059 0.10297181 12 0  
## Feb 1969 97 1508 825 265 7685 0.10236300 6 0  
## Mar 1969 102 1507 806 319 9963 0.10206249 12 0  
## Apr 1969 87 1385 814 407 10955 0.10087330 8 0  
## May 1969 119 1632 991 454 11823 0.10101967 10 0  
## Jun 1969 106 1511 945 427 12391 0.10058119 13 0  
## Jul 1969 110 1559 1004 522 13460 0.10377398 11 0  
## Aug 1969 106 1630 1091 536 14055 0.10407640 6 0  
## Sep 1969 107 1579 958 405 12106 0.10377398 10 0  
## Oct 1969 134 1653 850 437 11372 0.10302640 16 0  
## Nov 1969 147 2152 1109 434 9834 0.10273011 13 0  
## Dec 1969 180 2148 1113 437 9267 0.10199719 14 0  
## Jan 1970 125 1752 925 316 9130 0.10127456 14 0  
## Feb 1970 134 1765 903 311 8933 0.10070398 6 0  
## Mar 1970 110 1717 1006 351 11000 0.10013961 8 0  
## Apr 1970 102 1558 892 362 10733 0.09862110 11 0  
## May 1970 103 1575 990 486 12912 0.09834929 7 0  
## Jun 1970 111 1520 866 429 12926 0.09808018 13 0  
## Jul 1970 120 1805 1095 551 13990 0.09727921 13 0  
## Aug 1970 129 1800 1204 646 14926 0.09741062 11 0  
## Sep 1970 122 1719 1029 456 12900 0.09742524 11 0  
## Oct 1970 183 2008 1147 475 12034 0.09638063 14 0  
## Nov 1970 169 2242 1171 456 10643 0.09573896 16 0  
## Dec 1970 190 2478 1299 468 10742 0.09510631 14 0  
## Jan 1971 134 2030 944 356 10266 0.09673597 17 0  
## Feb 1971 108 1655 874 271 10281 0.09610922 16 0  
## Mar 1971 104 1693 840 354 11527 0.09536725 15 0  
## Apr 1971 117 1623 893 427 12281 0.09470959 13 0  
## May 1971 157 1805 1007 465 13587 0.09411762 13 0  
## Jun 1971 148 1746 973 440 13049 0.09353215 15 0  
## Jul 1971 130 1795 1097 539 16055 0.09295405 12 0  
## Aug 1971 140 1926 1194 646 15220 0.09283979 6 0  
## Sep 1971 136 1619 988 457 13824 0.09272474 9 0  
## Oct 1971 140 1992 1077 446 12729 0.09226965 13 0  
## Nov 1971 187 2233 1045 402 11467 0.09170669 14 0  
## Dec 1971 150 2192 1115 441 11351 0.09126207 15 0  
## Jan 1972 159 2080 1005 359 10803 0.09071160 14 0  
## Feb 1972 143 1768 857 334 10548 0.09027633 3 0  
## Mar 1972 114 1835 879 312 12368 0.08995192 12 0  
## Apr 1972 127 1569 887 427 13311 0.08909964 13 0  
## May 1972 159 1976 1075 434 13885 0.08867919 12 0  
## Jun 1972 156 1853 1121 486 14088 0.08815929 8 0  
## Jul 1972 138 1965 1190 569 16932 0.08890206 8 0  
## Aug 1972 120 1689 1058 523 16164 0.08818133 15 0  
## Sep 1972 117 1778 939 418 14883 0.08894029 8 0  
## Oct 1972 170 1976 1074 452 13532 0.08772661 5 0  
## Nov 1972 168 2397 1089 462 12220 0.08742885 17 0  
## Dec 1972 198 2654 1208 497 12025 0.08703543 14 0  
## Jan 1973 144 2097 903 354 11692 0.08644992 13 0  
## Feb 1973 146 1963 916 347 11081 0.08587264 5 0  
## Mar 1973 109 1677 787 276 13745 0.08539822 8 0  
## Apr 1973 131 1941 1114 472 14382 0.08382198 5 0  
## May 1973 151 2003 1014 487 14391 0.08459078 12 0  
## Jun 1973 140 1813 1022 505 15597 0.08413690 11 0  
## Jul 1973 153 2012 1114 619 16834 0.08377841 13 0  
## Aug 1973 140 1912 1132 640 17282 0.08351074 15 0  
## Sep 1973 161 2084 1111 559 15779 0.08280639 11 0  
## Oct 1973 168 2080 1008 453 13946 0.08117889 11 0  
## Nov 1973 152 2118 916 418 12701 0.08285361 10 0  
## Dec 1973 136 2150 992 419 10431 0.09419012 13 0  
## Jan 1974 113 1608 731 262 11616 0.09239984 8 0  
## Feb 1974 100 1503 665 299 10808 0.10816148 6 0  
## Mar 1974 103 1548 724 303 12421 0.10721169 8 0  
## Apr 1974 103 1382 744 401 13605 0.11404297 14 0  
## May 1974 121 1731 910 413 14455 0.11245412 12 0  
## Jun 1974 134 1798 883 426 15019 0.11131625 14 0  
## Jul 1974 133 1779 900 516 15662 0.11030125 13 0  
## Aug 1974 129 1887 1057 600 16745 0.10819718 9 0  
## Sep 1974 144 2004 1076 459 14717 0.10702744 4 0  
## Oct 1974 154 2077 919 443 13756 0.10494698 13 0  
## Nov 1974 156 2092 920 412 12531 0.11935775 6 0  
## Dec 1974 163 2051 953 400 12568 0.11762190 15 0  
## Jan 1975 122 1577 664 278 11249 0.13302742 12 0  
## Feb 1975 92 1356 607 302 11096 0.13084524 16 0  
## Mar 1975 117 1652 777 381 12637 0.12831848 7 0  
## Apr 1975 95 1382 633 279 13018 0.12354745 12 0  
## May 1975 96 1519 791 442 15005 0.11858681 10 0  
## Jun 1975 108 1421 790 409 15235 0.11633748 9 0  
## Jul 1975 108 1442 803 416 15552 0.11516148 9 0  
## Aug 1975 106 1543 884 511 16905 0.11450120 6 0  
## Sep 1975 140 1656 769 393 14776 0.11352298 7 0  
## Oct 1975 114 1561 732 345 14104 0.11193018 13 0  
## Nov 1975 158 1905 859 391 12854 0.11061053 14 0  
## Dec 1975 161 2199 994 470 12956 0.11527439 13 0  
## Jan 1976 102 1473 704 266 12177 0.11379349 14 0  
## Feb 1976 127 1655 684 312 11918 0.11234958 11 0  
## Mar 1976 125 1407 671 300 13517 0.11175347 11 0  
## Apr 1976 101 1395 643 373 14417 0.10964252 10 0  
## May 1976 97 1530 771 412 15911 0.10844090 4 0  
## Jun 1976 112 1309 644 322 15589 0.10788494 8 0  
## Jul 1976 112 1526 828 458 16543 0.10908477 9 0  
## Aug 1976 113 1327 748 427 17925 0.10757145 10 0  
## Sep 1976 108 1627 767 346 15406 0.10616402 10 0  
## Oct 1976 128 1748 825 421 14601 0.10630000 5 0  
## Nov 1976 154 1958 810 344 13107 0.10482531 13 0  
## Dec 1976 162 2274 986 370 12268 0.10345175 12 0  
## Jan 1977 112 1648 714 291 11972 0.10144992 10 0  
## Feb 1977 79 1401 567 224 12028 0.10040232 9 0  
## Mar 1977 82 1411 616 266 14033 0.09886203 7 0  
## Apr 1977 127 1403 678 338 14244 0.10249615 5 0  
## May 1977 108 1394 742 298 15287 0.10302743 10 0  
## Jun 1977 110 1520 840 386 16954 0.10217891 5 0  
## Jul 1977 123 1528 888 479 17361 0.09983664 6 0  
## Aug 1977 103 1643 852 473 17694 0.09263669 8 0  
## Sep 1977 97 1515 774 332 16222 0.09181496 6 0  
## Oct 1977 140 1685 831 391 14969 0.09072430 12 0  
## Nov 1977 165 2000 889 370 13624 0.09002121 15 0  
## Dec 1977 183 2215 1046 431 13842 0.08933071 7 0  
## Jan 1978 148 1956 889 366 12387 0.08844273 14 0  
## Feb 1978 111 1462 626 250 11608 0.08835257 4 0  
## Mar 1978 116 1563 808 355 15021 0.08675736 10 0  
## Apr 1978 115 1459 746 304 14834 0.08499524 8 0  
## May 1978 100 1446 754 379 16565 0.08456794 7 0  
## Jun 1978 106 1622 865 440 16882 0.08443190 11 0  
## Jul 1978 134 1657 980 500 18012 0.08435088 3 0  
## Aug 1978 125 1638 959 511 18855 0.08360098 5 0  
## Sep 1978 117 1643 856 384 17243 0.08341726 11 0  
## Oct 1978 122 1683 798 366 16045 0.08274514 10 0  
## Nov 1978 153 2050 942 432 14745 0.08523527 10 0  
## Dec 1978 178 2262 1010 390 13726 0.08477030 7 0  
## Jan 1979 114 1813 796 306 11196 0.08445892 10 0  
## Feb 1979 94 1445 643 232 12105 0.08535212 11 0  
## Mar 1979 128 1762 794 342 14723 0.08755921 9 0  
## Apr 1979 119 1461 750 329 15582 0.09038292 7 0  
## May 1979 111 1556 809 394 16863 0.09078329 8 0  
## Jun 1979 110 1431 716 355 16758 0.10874278 13 0  
## Jul 1979 114 1427 851 385 17434 0.11414223 8 0  
## Aug 1979 118 1554 931 463 18359 0.11299293 5 0  
## Sep 1979 115 1645 834 453 17189 0.11132071 8 0  
## Oct 1979 132 1653 762 373 16909 0.10912623 7 0  
## Nov 1979 153 2016 880 401 15380 0.10769846 12 0  
## Dec 1979 171 2207 1077 466 15161 0.10760157 10 0  
## Jan 1980 115 1665 748 306 14027 0.10377502 7 0  
## Feb 1980 95 1361 593 263 14478 0.10711417 4 0  
## Mar 1980 92 1506 720 323 16155 0.10737477 10 0  
## Apr 1980 100 1360 646 310 16585 0.11169537 4 0  
## May 1980 95 1453 765 424 18117 0.11063818 8 0  
## Jun 1980 114 1522 820 403 17552 0.11185521 8 0  
## Jul 1980 102 1460 807 406 18299 0.10974234 7 0  
## Aug 1980 104 1552 885 466 19361 0.10819393 10 0  
## Sep 1980 132 1548 803 381 17924 0.10625536 8 0  
## Oct 1980 136 1827 860 369 17872 0.10419303 14 0  
## Nov 1980 117 1737 825 378 16058 0.10193397 8 0  
## Dec 1980 137 1941 911 392 15746 0.10279382 9 0  
## Jan 1981 111 1474 704 284 15226 0.10476034 8 0  
## Feb 1981 106 1458 691 316 14932 0.10400254 6 0  
## Mar 1981 98 1542 688 321 16846 0.11665552 7 0  
## Apr 1981 84 1404 714 358 16854 0.11516148 6 0  
## May 1981 94 1522 814 378 18146 0.11298954 5 0  
## Jun 1981 105 1385 736 382 17559 0.11386064 4 0  
## Jul 1981 123 1641 876 433 18655 0.11911808 5 0  
## Aug 1981 109 1510 829 506 19453 0.12448999 10 0  
## Sep 1981 130 1681 818 428 17923 0.12322295 7 0  
## Oct 1981 153 1938 942 479 17915 0.12067793 10 0  
## Nov 1981 134 1868 782 370 16496 0.12104898 12 0  
## Dec 1981 99 1726 823 349 13544 0.11696857 7 0  
## Jan 1982 115 1456 595 238 13601 0.11275026 4 0  
## Feb 1982 104 1445 673 285 15667 0.10807931 5 0  
## Mar 1982 131 1456 660 324 17358 0.10883852 6 0  
## Apr 1982 108 1365 676 346 18112 0.11129177 4 0  
## May 1982 103 1487 755 410 18581 0.11130401 4 0  
## Jun 1982 115 1558 815 411 18759 0.11545436 8 0  
## Jul 1982 122 1488 867 496 20668 0.11476830 8 0  
## Aug 1982 122 1684 933 534 21040 0.11720743 3 0  
## Sep 1982 125 1594 798 396 18993 0.11907640 7 0  
## Oct 1982 137 1850 950 470 18668 0.11796586 12 0  
## Nov 1982 138 1998 825 385 16768 0.11744913 2 0  
## Dec 1982 152 2079 911 411 16551 0.11698846 7 0  
## Jan 1983 120 1494 619 281 16231 0.11261054 8 0  
## Feb 1983 95 1057 426 300 15511 0.11365702 3 1  
## Mar 1983 100 1218 475 318 18308 0.11314445 2 1  
## Apr 1983 89 1168 556 391 17793 0.11849553 6 1  
## May 1983 82 1236 559 398 19205 0.11796940 3 1  
## Jun 1983 89 1076 483 337 19162 0.11768661 7 1  
## Jul 1983 60 1174 587 477 20997 0.12005924 6 1  
## Aug 1983 84 1139 615 422 20705 0.11943775 8 1  
## Sep 1983 113 1427 618 495 18759 0.11888127 8 1  
## Oct 1983 126 1487 662 471 19240 0.11846236 4 1  
## Nov 1983 122 1483 519 368 17504 0.11801660 3 1  
## Dec 1983 118 1513 585 345 16591 0.11770662 5 1  
## Jan 1984 92 1357 483 296 16224 0.11777609 5 1  
## Feb 1984 86 1165 434 319 16670 0.11479699 3 1  
## Mar 1984 81 1282 513 349 18539 0.11573525 4 1  
## Apr 1984 84 1110 548 375 19759 0.11535626 3 1  
## May 1984 87 1297 586 441 19584 0.11481536 6 1  
## Jun 1984 90 1185 522 465 19976 0.11477748 6 1  
## Jul 1984 79 1222 601 472 21486 0.11493598 7 1  
## Aug 1984 96 1284 644 521 21626 0.11479699 5 1  
## Sep 1984 122 1444 643 429 20195 0.11409316 7 1  
## Oct 1984 120 1575 641 408 19928 0.11646552 7 1  
## Nov 1984 137 1737 711 490 18564 0.11602611 4 1  
## Dec 1984 154 1763 721 491 18149 0.11606673 7 1

print('Variables of Seatbelts ')

## [1] "Variables of Seatbelts "

str(Seatbelts)

## Time-Series [1:192, 1:8] from 1969 to 1985: 107 97 102 87 119 106 110 106 107 134 ...  
## - attr(\*, "dimnames")=List of 2  
## ..$ : NULL  
## ..$ : chr [1:8] "DriversKilled" "drivers" "front" "rear" ...

print('Summary of Seatbelts ')

## [1] "Summary of Seatbelts "

summary(Seatbelts)

## DriversKilled drivers front rear   
## Min. : 60.0 Min. :1057 Min. : 426.0 Min. :224.0   
## 1st Qu.:104.8 1st Qu.:1462 1st Qu.: 715.5 1st Qu.:344.8   
## Median :118.5 Median :1631 Median : 828.5 Median :401.5   
## Mean :122.8 Mean :1670 Mean : 837.2 Mean :401.2   
## 3rd Qu.:138.0 3rd Qu.:1851 3rd Qu.: 950.8 3rd Qu.:456.2   
## Max. :198.0 Max. :2654 Max. :1299.0 Max. :646.0   
## kms PetrolPrice VanKilled law   
## Min. : 7685 Min. :0.08118 Min. : 2.000 Min. :0.0000   
## 1st Qu.:12685 1st Qu.:0.09258 1st Qu.: 6.000 1st Qu.:0.0000   
## Median :14987 Median :0.10448 Median : 8.000 Median :0.0000   
## Mean :14994 Mean :0.10362 Mean : 9.057 Mean :0.1198   
## 3rd Qu.:17203 3rd Qu.:0.11406 3rd Qu.:12.000 3rd Qu.:0.0000   
## Max. :21626 Max. :0.13303 Max. :17.000 Max. :1.0000

print('Dimensions of Seatbelts ')

## [1] "Dimensions of Seatbelts "

dim(Seatbelts)

## [1] 192 8

print('Rows in Seatbelts ')

## [1] "Rows in Seatbelts "

nrow(Seatbelts)

## [1] 192

print('Cols in Seatbelts ')

## [1] "Cols in Seatbelts "

ncol(Seatbelts)

## [1] 8

Seatbelts Dataset conatins multiple time series with columns like Drivers killed , killometers , petrol proice, van killed. It contains 192 rows and 8 columns. Time series is from 1969 - 1985. It is processed dataset, there are no NaN valus and also no null vaues in the dataset.

sum(is.na(Seatbelts))

## [1] 0

sum(is.null(Seatbelts))

## [1] 0

1. How is the built-in dataset âUKDriversDeathâ different from âSeatbeltsâ?

#######Q1c########  
  
  
UKDriverDeaths

## Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec  
## 1969 1687 1508 1507 1385 1632 1511 1559 1630 1579 1653 2152 2148  
## 1970 1752 1765 1717 1558 1575 1520 1805 1800 1719 2008 2242 2478  
## 1971 2030 1655 1693 1623 1805 1746 1795 1926 1619 1992 2233 2192  
## 1972 2080 1768 1835 1569 1976 1853 1965 1689 1778 1976 2397 2654  
## 1973 2097 1963 1677 1941 2003 1813 2012 1912 2084 2080 2118 2150  
## 1974 1608 1503 1548 1382 1731 1798 1779 1887 2004 2077 2092 2051  
## 1975 1577 1356 1652 1382 1519 1421 1442 1543 1656 1561 1905 2199  
## 1976 1473 1655 1407 1395 1530 1309 1526 1327 1627 1748 1958 2274  
## 1977 1648 1401 1411 1403 1394 1520 1528 1643 1515 1685 2000 2215  
## 1978 1956 1462 1563 1459 1446 1622 1657 1638 1643 1683 2050 2262  
## 1979 1813 1445 1762 1461 1556 1431 1427 1554 1645 1653 2016 2207  
## 1980 1665 1361 1506 1360 1453 1522 1460 1552 1548 1827 1737 1941  
## 1981 1474 1458 1542 1404 1522 1385 1641 1510 1681 1938 1868 1726  
## 1982 1456 1445 1456 1365 1487 1558 1488 1684 1594 1850 1998 2079  
## 1983 1494 1057 1218 1168 1236 1076 1174 1139 1427 1487 1483 1513  
## 1984 1357 1165 1282 1110 1297 1185 1222 1284 1444 1575 1737 1763

str(UKDriverDeaths)

## Time-Series [1:192] from 1969 to 1985: 1687 1508 1507 1385 1632 ...

print('Summary of UKDriverDeaths ')

## [1] "Summary of UKDriverDeaths "

summary(UKDriverDeaths)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 1057 1462 1631 1670 1851 2654

Datasets UKDriverDeaths & Seatbelts both are Timeseries data but UKDriverDeaths just have the count of deaths of driver with no other variables whereas in Seatbelts we have more variables in addition to the year and month data.UKDriverDeaths is a time series giving the monthly totals of car drivers in Great Britain killed or seriously injured Jan 1969 to Dec 1984. Compulsory wearing of seat belts was introduced on 31 Jan 1983. Seatbelts is more information on the same problem.

1. What does Seatbelts[1,1] return? What does Seatbelts[29, 5] return? Describe in 1 sentence what is going on.

#######Q1d########  
  
Seatbelts[1,1]

## DriversKilled   
## 107

Seatbelts[29, 5]

## kms   
## 13587

Seatbelts[1,1] returns the value(107) in cell[1,1] i.e. 1st row and 1st column(DriversKilled) of the dataframe & Seatbelts[29, 5] returns the value(13587) in 29th row and 5th column(kms) of the dataframe.

1. If you were interested in analyzing deaths due to car accidents in the UK, describe how you could combine Seatbelts and UKDriversDeath to do so. (You do not need actually do this.)

What I can do is combine datasets and grab the deaths from UKdriverDeaths and combine it with the Front , vankilled over the years of 1969-1985. this will give the deaths in UK with van killed.

1. Create a variable Bob and set it to True. Type Bob and what does R return. (Note that R is case sensitive so âBobâ does not equal âbobâ.) Type Bob+Bob. What is the result and what is going on?

#######Q1f########  
  
Bob = TRUE  
  
Bob+Bob

## [1] 2

Bob is assigned as ‘TRUE’ which is treated as bolean which is ‘1’ for ‘TRUE’ & ‘0’ for ‘FALSE’. So, when we perform Bob+Bob it returns “2” which is 1+1 in boolean.

1. What is vector recycling in R? (It applies to all vectors not just logical ones.) Create a vector of two logical values and another of 5 logical values. Ask R if those two vectors are equal. What happens and what is going on?

#######Q1g########  
  
a = c(1,0)  
b= c(1,1,1,0,1)  
  
a == b

## Warning in a == b: longer object length is not a multiple of shorter object  
## length

## [1] TRUE FALSE TRUE TRUE TRUE

Vector Recycling in r: If two vectors are of unequal length, the shorter one will be recycled in order to match the longer vector. For example, the following vectors u and v have different lengths, and their sum is computed by recycling values of the shorter vector u.

if the value of “a” matches with Value of ‘b’ then R returns ‘TRUE’ otherwise ‘FALSE’. Each vale of vector ‘a’ is been compared with vector ‘b’. since ‘a’ has only 2 elements so for comparing the rest of the elements from ‘b , ’a’ is been repeated . which means that R compares a(1,0,1,0,1) & b(1,1,1,0,1) which returns “TRUE FALSE TRUE TRUE TRUE”

### Question 2: Basic Data Manipulation

1. Download the PUMS dataset from Canvas, file name: psam\_p34.csv. This is the Public Use Micro Dataset, a subset from the ACS survey. You can also find on Canvas the definition of variables for this dataset (PUMS\_Data\_Dictionary\_2017). Follow the directions below to import a dataset into RStudio
   * Go to âFileâ tab at top of Computer Screen
   * Under âImport Datasetâ, choose âFrom Text(base)â
   * Navigate to the folder in which dataset is downloaded
   * Click Import to continue through with the dataset
   * **Note**: R Studio actually provides you with the code to import datasets. Type that code below

#######Q2a########  
  
psam <- read.csv("D:/Downloads/psam\_p34.csv", stringsAsFactors=FALSE)

1. Add a new column to the data frame and fill it with 10 in all rows.

#######Q2b########  
  
psam$newcol = 10  
  
colnames(psam)  
  
psam['newcol']

1. Add a new column to the data frame and copy the data from an existing column in the dataset, PWGTP80 into this column.

#######Q2c########  
  
psam$newcol = psam$PWGTP80  
  
psam['newcol']

1. Delete the column PWGTP74

#######Q2d########  
  
psam$PWGTP74 <- NULL  
  
#colnames(psam)

1. Rename the column CIT as CHARCT

#######Q2e########  
  
  
names(psam)[names(psam) == 'CIT'] <- 'CHARCT'  
  
#colnames(psam)  
  
psam["CHARCT"]

### Question 3: Subsetting & Sorting Data

Subsetting data is the process of retrieving just the parts of larger datasets that are of specific interest for the project at hand. It is a very important component of data management and there are several ways that one can subset data in R.

1. Complete the data subsetting tutorial at this [website](https://stats.idre.ucla.edu/r/faq/frequently-asked-questions-about-rhow-can-i-subset-a-data-setthe-r-program-as-a-text-file-for-all-the-code-on-this-page-subsetting-is-a-very-important-component/)
2. Sort the data according to the variable MAR in ascending order

#######Q3b########  
  
head(psam)  
  
psam1 = psam[order(psam$MAR),]  
  
head(psam1)

1. Sort the data in ascending order by PWGTP3 and descending order by PWGTP7 together.

#######Q3c########  
  
  
psam2 = psam[order(psam$PWGTP3, -psam$PWGTP7),]  
  
head(psam2)

1. Create a subset of the data by âkeepingâ the first 10 variables in the PUMS dataset (RT to AGEP) or âdroppingâ the other variables.

#######Q3d########  
  
  
psam\_subset = psam[,1:10]  
  
head(psam\_subset)  
  
dim(psam\_subset)

1. Create a subset of the data by âkeepingâ the first 10 observations.

#######Q3e########  
  
psam\_sub = psam[1:10,]  
  
head(psam\_sub)  
dim(psam\_sub)

1. Take a random sample of the dataset of size 50:
   * with replacement

#######Q3f########  
  
#(i) with replacement  
set.seed(100)  
sam\_rep = psam[sample(nrow(psam),50, replace = TRUE),]  
  
head(sam\_rep)

\* without replacement

#(ii) without replacement  
sam\_wrep = psam[sample(nrow(psam),50, replace = FALSE),]  
  
head(sam\_wrep)

### Question 4: Descriptive Stats

For this question, we will use one of the most common R built-in datasets, mtcars. The easiest way to get descriptive statistics in R is using the summary() command.

1. Find the summary statistics of the mtcars dataset using the summary() command.

#######Q4a########  
  
summary(mtcars)

## mpg cyl disp hp   
## Min. :10.40 Min. :4.000 Min. : 71.1 Min. : 52.0   
## 1st Qu.:15.43 1st Qu.:4.000 1st Qu.:120.8 1st Qu.: 96.5   
## Median :19.20 Median :6.000 Median :196.3 Median :123.0   
## Mean :20.09 Mean :6.188 Mean :230.7 Mean :146.7   
## 3rd Qu.:22.80 3rd Qu.:8.000 3rd Qu.:326.0 3rd Qu.:180.0   
## Max. :33.90 Max. :8.000 Max. :472.0 Max. :335.0   
## drat wt qsec vs   
## Min. :2.760 Min. :1.513 Min. :14.50 Min. :0.0000   
## 1st Qu.:3.080 1st Qu.:2.581 1st Qu.:16.89 1st Qu.:0.0000   
## Median :3.695 Median :3.325 Median :17.71 Median :0.0000   
## Mean :3.597 Mean :3.217 Mean :17.85 Mean :0.4375   
## 3rd Qu.:3.920 3rd Qu.:3.610 3rd Qu.:18.90 3rd Qu.:1.0000   
## Max. :4.930 Max. :5.424 Max. :22.90 Max. :1.0000   
## am gear carb   
## Min. :0.0000 Min. :3.000 Min. :1.000   
## 1st Qu.:0.0000 1st Qu.:3.000 1st Qu.:2.000   
## Median :0.0000 Median :4.000 Median :2.000   
## Mean :0.4062 Mean :3.688 Mean :2.812   
## 3rd Qu.:1.0000 3rd Qu.:4.000 3rd Qu.:4.000   
## Max. :1.0000 Max. :5.000 Max. :8.000

1. Another way to get more detailed descriptive statistics is to use the pastecs package.
   * Install the pastecs package:
     + Type install.packages("pastecs") and load it from the library by typing library(pastecs)

#######Q4b########  
  
#install.packages("pastecs")  
library(pastecs)

## Warning: package 'pastecs' was built under R version 3.5.2

\* Find the command to get the descriptive statistics using this package. (Hint: your output should give you a minimum, maximum, range, SE. mean, C.I Mean, standard deviation and coefficient of variance etc)

stat.desc(mtcars)

## mpg cyl disp hp  
## nbr.val 32.0000000 32.0000000 3.200000e+01 32.0000000  
## nbr.null 0.0000000 0.0000000 0.000000e+00 0.0000000  
## nbr.na 0.0000000 0.0000000 0.000000e+00 0.0000000  
## min 10.4000000 4.0000000 7.110000e+01 52.0000000  
## max 33.9000000 8.0000000 4.720000e+02 335.0000000  
## range 23.5000000 4.0000000 4.009000e+02 283.0000000  
## sum 642.9000000 198.0000000 7.383100e+03 4694.0000000  
## median 19.2000000 6.0000000 1.963000e+02 123.0000000  
## mean 20.0906250 6.1875000 2.307219e+02 146.6875000  
## SE.mean 1.0654240 0.3157093 2.190947e+01 12.1203173  
## CI.mean.0.95 2.1729465 0.6438934 4.468466e+01 24.7195501  
## var 36.3241028 3.1895161 1.536080e+04 4700.8669355  
## std.dev 6.0269481 1.7859216 1.239387e+02 68.5628685  
## coef.var 0.2999881 0.2886338 5.371779e-01 0.4674077  
## drat wt qsec vs am  
## nbr.val 32.00000000 32.0000000 32.0000000 32.00000000 32.00000000  
## nbr.null 0.00000000 0.0000000 0.0000000 18.00000000 19.00000000  
## nbr.na 0.00000000 0.0000000 0.0000000 0.00000000 0.00000000  
## min 2.76000000 1.5130000 14.5000000 0.00000000 0.00000000  
## max 4.93000000 5.4240000 22.9000000 1.00000000 1.00000000  
## range 2.17000000 3.9110000 8.4000000 1.00000000 1.00000000  
## sum 115.09000000 102.9520000 571.1600000 14.00000000 13.00000000  
## median 3.69500000 3.3250000 17.7100000 0.00000000 0.00000000  
## mean 3.59656250 3.2172500 17.8487500 0.43750000 0.40625000  
## SE.mean 0.09451874 0.1729685 0.3158899 0.08909831 0.08820997  
## CI.mean.0.95 0.19277224 0.3527715 0.6442617 0.18171719 0.17990541  
## var 0.28588135 0.9573790 3.1931661 0.25403226 0.24899194  
## std.dev 0.53467874 0.9784574 1.7869432 0.50401613 0.49899092  
## coef.var 0.14866382 0.3041285 0.1001159 1.15203687 1.22828533  
## gear carb  
## nbr.val 32.0000000 32.0000000  
## nbr.null 0.0000000 0.0000000  
## nbr.na 0.0000000 0.0000000  
## min 3.0000000 1.0000000  
## max 5.0000000 8.0000000  
## range 2.0000000 7.0000000  
## sum 118.0000000 90.0000000  
## median 4.0000000 2.0000000  
## mean 3.6875000 2.8125000  
## SE.mean 0.1304266 0.2855297  
## CI.mean.0.95 0.2660067 0.5823417  
## var 0.5443548 2.6088710  
## std.dev 0.7378041 1.6152000  
## coef.var 0.2000825 0.5742933

1. There are also separate commands to get the mean, median and mean statistics. Find the mean, median and mode of the variable mpg by separate commands.

#######Q4c########  
  
mean(mtcars$mpg)

## [1] 20.09062

median(mtcars$mpg)

## [1] 19.2

mode1 = names(table(mtcars$mpg))[table(mtcars$mpg)==max(table(mtcars$mpg))]  
  
mode1

## [1] "10.4" "15.2" "19.2" "21" "21.4" "22.8" "30.4"

1. Find the length of the variable qtsec. Why are the lengths of all the variables the same?

#######Q4d########  
  
length(mtcars$qsec)

## [1] 32

# Length of 'qsec' is 32  
# Variables are part of dataframe , which is a table and each varaible needs to be equal lentyh to form a tabular structure. even if variables are of uneqaul lenth they are filled with NULL to make up the dimensions of table.

1. Find the maximum and minimum value of the mpg variable.

#######Q4e########  
  
mpg\_max = max(mtcars$mpg)  
mpg\_max

## [1] 33.9

mpg\_min= min(mtcars$mpg)  
mpg\_min

## [1] 10.4

1. Determine the location i.e index of the maximum and minimum value you found in part e. (Hint: Try the which.max command).

#######Q4f########  
  
which.max(mtcars$mpg)

## [1] 20

which.min(mtcars$mpg)

## [1] 15

### Question 5: Putting it all together

#### Downloading a dataset:

1. Go to [this link](https://www.kaggle.com/aparnashastry/building-permit-applications-data#Building_Permits.csv) on Kaggle. This should take you to a page for the “San Francisco Building Permits” dataset. (**note**: you will have to create an account in roder to download this dataset. Kaggle is a PHENOMENAL resource for datasets and data-related explorations, so making this account now will help you for future assignments.)
2. Once youâve downloaded this dataset, time to **import** the dataset into RStudio (Refer to Question 2 for tips on importing datasets). Type in the code that imports this dataset below: (After a while, you will see that the dataset âBuilding\_Permitsâ is available in your âGlobal Variableâ explorer in RStudio)

#######Q5->2########  
  
Building\_Permits <- read.csv("C:/Users/akhil/Downloads/Building\_Permits.csv", stringsAsFactors=FALSE)

#### Preparing for data manipulation:

1. If youâve successfully imported the dataset, you should have a Building\_Permits variable in your global explorer – **congrats!** As per convention, itâs always a great idea to create a **copy** of your dataset, so that whatever manipulations you make donât affect the original dataset. With that being said, make a copy of the dataset! \* **hint:** Name the copy whatever you would like and literally use the <- operator to assign your newly named variable to the existing Building\_Permits dataset

#######Q5->2->1########  
  
buldgp = Building\_Permits

#### Exploring the dataset:

1. Thus far, weâve downloaded the dataset and made copies to prevent against any future accidents. Now, letâs explore our dataset a little further and really understand what weâre dealing with here:
2. Reproduce the following printed statement **with code** and **replace X** with the number of rows and **replace Y** with the number of columns of your dataset : Dimensions: X rows, Y columns

* **hints**:
  1. you’ll find the [R cat()](https://www.rdocumentation.org/packages/base/versions/3.5.2/topics/cat) function really helpful
  2. The dim() function from question 1 will be really useful! (Also, there are two components that are returned by calling the dim() function, and you can access each portion with a proper index call (example: dim(â<dataset\_nameâ>)[index])
  3. You can use the cat() function as follows: cat(â<String: â, data, â<another string>â, more data)

#######Q5->3->2########  
  
dim(buldgp)  
  
cat("Dimensions: ",dim(buldgp)[1], " rows, ",dim(buldgp)[2], " columns" )

1. Generally, if weâre dealing with data that is numeric, it might be helpful to look for the averages in a dataset. Take a quick look at the different columns of this dataset – do you think itâs appropriate to analyze stats like the mean, median, mode for the numeric columns? Why or why not?

#######Q5->3->3########  
str(buldgp)

## 'data.frame': 198900 obs. of 43 variables:  
## $ Permit.Number : chr "201505065519" "201604195146" "201605278609" "201611072166" ...  
## $ Permit.Type : int 4 4 3 8 6 8 8 8 8 8 ...  
## $ Permit.Type.Definition : chr "sign - erect" "sign - erect" "additions alterations or repairs" "otc alterations permit" ...  
## $ Permit.Creation.Date : chr "05/06/2015" "04/19/2016" "05/27/2016" "11/07/2016" ...  
## $ Block : chr "0326" "0306" "0595" "0156" ...  
## $ Lot : chr "023" "007" "203" "011" ...  
## $ Street.Number : int 140 440 1647 1230 950 800 1291 1465 2094 89 ...  
## $ Street.Number.Suffix : chr "" "" "" "" ...  
## $ Street.Name : chr "Ellis" "Geary" "Pacific" "Pacific" ...  
## $ Street.Suffix : chr "St" "St" "Av" "Av" ...  
## $ Unit : int NA 0 NA 0 NA NA 0 NA NA NA ...  
## $ Unit.Suffix : chr "" "" "" "" ...  
## $ Description : chr "ground fl facade: to erect illuminated, electric, wall, single faced sign. n/a for maher ordinance 155-13." "remove (e) awning and associated signs." "installation of separating wall" "repair dryrot & stucco at front of bldg." ...  
## $ Current.Status : chr "expired" "issued" "withdrawn" "complete" ...  
## $ Current.Status.Date : chr "12/21/2017" "08/03/2017" "09/26/2017" "07/24/2017" ...  
## $ Filed.Date : chr "05/06/2015" "04/19/2016" "05/27/2016" "11/07/2016" ...  
## $ Issued.Date : chr "11/09/2015" "08/03/2017" "" "07/18/2017" ...  
## $ Completed.Date : chr "" "" "" "07/24/2017" ...  
## $ First.Construction.Document.Date : chr "11/09/2015" "08/03/2017" "" "07/18/2017" ...  
## $ Structural.Notification : chr "" "" "" "" ...  
## $ Number.of.Existing.Stories : num 6 7 6 2 3 5 3 NA NA NA ...  
## $ Number.of.Proposed.Stories : num NA NA 6 2 NA 5 3 NA NA NA ...  
## $ Voluntary.Soft.Story.Retrofit : chr "" "" "" "" ...  
## $ Fire.Only.Permit : chr "" "" "" "" ...  
## $ Permit.Expiration.Date : chr "11/03/2016" "12/03/2017" "" "07/13/2018" ...  
## $ Estimated.Cost : num 4000 1 20000 2000 100000 4000 12000 NA NA NA ...  
## $ Revised.Cost : num 4000 500 NA 2000 100000 4000 12000 0 1 0 ...  
## $ Existing.Use : chr "tourist hotel/motel" "tourist hotel/motel" "retail sales" "1 family dwelling" ...  
## $ Existing.Units : num 143 NA 39 1 NA 326 5 NA NA NA ...  
## $ Proposed.Use : chr "" "" "retail sales" "1 family dwelling" ...  
## $ Proposed.Units : int NA NA 39 1 NA 326 5 NA NA NA ...  
## $ Plansets : int 2 2 2 2 2 2 0 NA NA NA ...  
## $ TIDF.Compliance : chr "" "" "" "" ...  
## $ Existing.Construction.Type : int 3 3 1 5 3 1 5 NA NA NA ...  
## $ Existing.Construction.Type.Description: chr "constr type 3" "constr type 3" "constr type 1" "wood frame (5)" ...  
## $ Proposed.Construction.Type : int NA NA 1 5 NA 1 5 NA NA NA ...  
## $ Proposed.Construction.Type.Description: chr "" "" "constr type 1" "wood frame (5)" ...  
## $ Site.Permit : chr "" "" "" "" ...  
## $ Supervisor.District : int 3 3 3 3 6 10 5 10 5 8 ...  
## $ Neighborhoods...Analysis.Boundaries : chr "Tenderloin" "Tenderloin" "Russian Hill" "Nob Hill" ...  
## $ Zipcode : int 94102 94102 94109 94109 94102 94107 94122 94124 94117 94117 ...  
## $ Location : chr "(37.785719256680785, -122.40852313194863)" "(37.78733980600732, -122.41063199757738)" "(37.7946573324287, -122.42232562979227)" "(37.79595867909168, -122.41557405519474)" ...  
## $ Record.ID : num 1.38e+12 1.42e+12 1.42e+12 1.44e+12 1.45e+11 ...

summary(buldgp)

## Permit.Number Permit.Type Permit.Type.Definition  
## Length:198900 Min. :1.000 Length:198900   
## Class :character 1st Qu.:8.000 Class :character   
## Mode :character Median :8.000 Mode :character   
## Mean :7.522   
## 3rd Qu.:8.000   
## Max. :8.000   
##   
## Permit.Creation.Date Block Lot Street.Number   
## Length:198900 Length:198900 Length:198900 Min. : 0   
## Class :character Class :character Class :character 1st Qu.: 235   
## Mode :character Mode :character Mode :character Median : 710   
## Mean :1122   
## 3rd Qu.:1700   
## Max. :8400   
##   
## Street.Number.Suffix Street.Name Street.Suffix   
## Length:198900 Length:198900 Length:198900   
## Class :character Class :character Class :character   
## Mode :character Mode :character Mode :character   
##   
##   
##   
##   
## Unit Unit.Suffix Description   
## Min. : 0.00 Length:198900 Length:198900   
## 1st Qu.: 0.00 Class :character Class :character   
## Median : 0.00 Mode :character Mode :character   
## Mean : 78.52   
## 3rd Qu.: 1.00   
## Max. :6004.00   
## NA's :169421   
## Current.Status Current.Status.Date Filed.Date   
## Length:198900 Length:198900 Length:198900   
## Class :character Class :character Class :character   
## Mode :character Mode :character Mode :character   
##   
##   
##   
##   
## Issued.Date Completed.Date First.Construction.Document.Date  
## Length:198900 Length:198900 Length:198900   
## Class :character Class :character Class :character   
## Mode :character Mode :character Mode :character   
##   
##   
##   
##   
## Structural.Notification Number.of.Existing.Stories  
## Length:198900 Min. : 0.00   
## Class :character 1st Qu.: 2.00   
## Mode :character Median : 3.00   
## Mean : 5.71   
## 3rd Qu.: 4.00   
## Max. :78.00   
## NA's :42784   
## Number.of.Proposed.Stories Voluntary.Soft.Story.Retrofit  
## Min. : 0.00 Length:198900   
## 1st Qu.: 2.00 Class :character   
## Median : 3.00 Mode :character   
## Mean : 5.75   
## 3rd Qu.: 4.00   
## Max. :78.00   
## NA's :42868   
## Fire.Only.Permit Permit.Expiration.Date Estimated.Cost   
## Length:198900 Length:198900 Min. :1.00e+00   
## Class :character Class :character 1st Qu.:3.30e+03   
## Mode :character Mode :character Median :1.10e+04   
## Mean :1.69e+05   
## 3rd Qu.:3.50e+04   
## Max. :5.38e+08   
## NA's :38066   
## Revised.Cost Existing.Use Existing.Units   
## Min. : 0 Length:198900 Min. : 0.00   
## 1st Qu.: 1 Class :character 1st Qu.: 1.00   
## Median : 7000 Mode :character Median : 1.00   
## Mean : 132856 Mean : 15.67   
## 3rd Qu.: 28708 3rd Qu.: 4.00   
## Max. :780500000 Max. :1907.00   
## NA's :6066 NA's :51538   
## Proposed.Use Proposed.Units Plansets TIDF.Compliance   
## Length:198900 Min. : 0.00 Min. : 0.00 Length:198900   
## Class :character 1st Qu.: 1.00 1st Qu.: 0.00 Class :character   
## Mode :character Median : 2.00 Median : 2.00 Mode :character   
## Mean : 16.51 Mean : 1.27   
## 3rd Qu.: 4.00 3rd Qu.: 2.00   
## Max. :1911.00 Max. :9000.00   
## NA's :50911 NA's :37309   
## Existing.Construction.Type Existing.Construction.Type.Description  
## Min. :1.00 Length:198900   
## 1st Qu.:3.00 Class :character   
## Median :5.00 Mode :character   
## Mean :4.07   
## 3rd Qu.:5.00   
## Max. :5.00   
## NA's :43366   
## Proposed.Construction.Type Proposed.Construction.Type.Description  
## Min. :1.00 Length:198900   
## 1st Qu.:3.00 Class :character   
## Median :5.00 Mode :character   
## Mean :4.09   
## 3rd Qu.:5.00   
## Max. :5.00   
## NA's :43162   
## Site.Permit Supervisor.District  
## Length:198900 Min. : 1.000   
## Class :character 1st Qu.: 3.000   
## Mode :character Median : 6.000   
## Mean : 5.538   
## 3rd Qu.: 8.000   
## Max. :11.000   
## NA's :1717   
## Neighborhoods...Analysis.Boundaries Zipcode Location   
## Length:198900 Min. :94102 Length:198900   
## Class :character 1st Qu.:94109 Class :character   
## Mode :character Median :94114 Mode :character   
## Mean :94116   
## 3rd Qu.:94122   
## Max. :94158   
## NA's :1716   
## Record.ID   
## Min. :1.294e+10   
## 1st Qu.:1.309e+12   
## Median :1.372e+12   
## Mean :1.162e+12   
## 3rd Qu.:1.435e+12   
## Max. :1.498e+12   
##

# No, Because by running the above commands we can see that most of the columns in the dataset are character strings.  
# but there are some columns like units , Estimated cost , revised cost , etc. for which we can have need for mean , median.

1. We know that weâre dealing with an incredible number of rows in our dataset (if you discovered the dimensions properly, weâre looking at ~200k rows). However, for some columns, we donât have ~200k unique values. Letâs discover some unique values. Find the number of unique values that are in the Existing.Use and the Neighborhoods...Analysis.Boundaries columns and print your results in the following format, replacing X and Y with their appropriate values (Please donât just *write* in the numbers, we want to see you **use** the functions in R to figure this out!) The Existing.Use column has X unique values and the Neighborhoods...Analysis.Boundaries has Y unique values

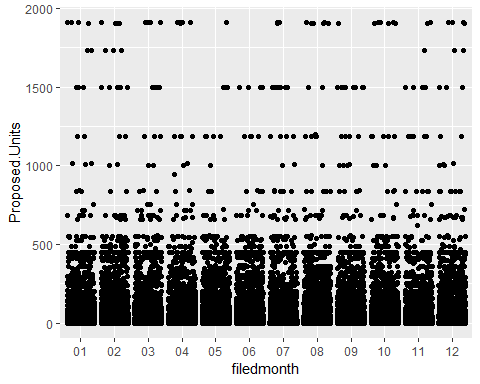
* **hints:**
  1. cat() will be your best friend!
  2. There is literally a function called [unique()](https://www.rdocumentation.org/packages/base/versions/3.5.2/topics/unique)– figure out how to manipulate this!

#######Q5->3->4########  
#colnames(buldgp)  
  
cat("The Existing.Use column has ", length(unique(buldgp$Existing.Use))," unique values and the Neighborhoods...Analysis.Boundaries has ",length(unique(buldgp$Neighborhoods...Analysis.Boundaries)) ," unique values")

1. This is the DIY part of your data exploration – find something interesting about the data using R code, and tell us why you think itâs interesting!

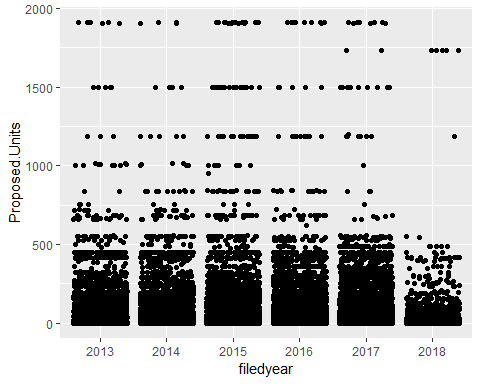
colnames(buldgp)  
library(ggplot2)  
  
# making a copy of dataset to perent changes.  
temp = buldgp  
str(temp)  
  
#parsing the FiledDate column to recognize as Date in R  
buldgp$Filed.Date <- as.Date(buldgp$Filed.Date, "%m/%d/%Y")  
  
#creating new column with filedmonth  
temp$filedmonth = format(buldgp$Filed.Date,'%m')  
  
#creating new column with filedyear  
temp$filedyear = format(buldgp$Filed.Date,'%Y')  
  
# plotting filedmonth with the proposed.Units  
ggplot(temp,aes(x=filedmonth , y = Proposed.Units)) +  
 geom\_jitter()

## Warning: Removed 50911 rows containing missing values (geom\_point).



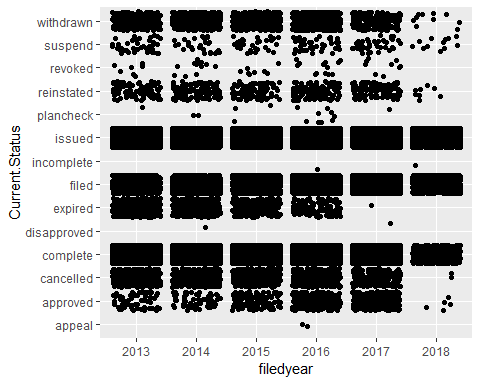
# plotting filedmonth with the proposed.Units  
ggplot(temp,aes(x=filedyear , y = Proposed.Units)) +  
 geom\_jitter()

## Warning: Removed 50911 rows containing missing values (geom\_point).

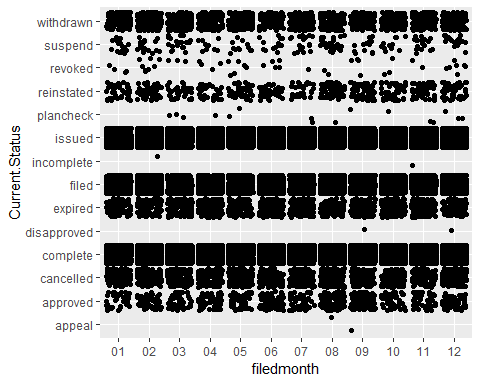


From the above graph we can conclude that most of the proposed units where in the months of January & May and the higgest number of proposed units is 1500 in months of january. Also, Looking at the year graph we can say that most of the units where proposed in 2013 and some of units where proposed in 2017.

temp$Current.Status = as.factor( temp$Current.Status)  
  
ggplot(temp,aes(x=filedyear , y = Current.Status)) +  
 geom\_jitter()



ggplot(temp,aes(x=filedmonth , y = Current.Status)) +  
 geom\_jitter()



From the above graph we can conclude that Current status of most of the Bulding permits were in 2013 Also in 2013, the Current Status of permits where mostly issued, filed, complete & somewat withdrawm. Some of the permits where issued in 2017. From the other graph current status of projects are mostly concentrated in first 5 months and then permits are took down in rest of the year. Most of the permits appears to the issued & completed till the month of May.

#### Data Manipulation

*Letâs draw upon Question 3: Subsetting datasets. Oftentimes, when weâre working with data, weâre not concerned about every single column in a dataset. Instead, there is only a handful of columns that are important to our needs. With this in mind, weâll subset our dataset so that we donât have to continually sift through relatively useless information in order to use our data. To this effect, weâre going to create 2 individual âdatasetsâ that are simply subsets of our main, overarching dataset.*s

1. Create a subset of your **copy** of the Building\_Permits dataset that only contains the following columns: Permit.Number, Description, Existing.Use

#######Q5->4->1a########  
  
#colnames(buldgp)  
  
#buldgp\_sub = cbind(buldgp$Existing.Use,buldgp$Permit.Number,buldgp$Description)  
  
buldg\_sub1 = buldgp[c("Existing.Use","Description","Permit.Number")]  
  
dim(buldg\_sub1)

1. Create a second subset of your **copy** of the Building\_Permits dataset that only contains the following columns: Permit.Number, Proposed.Use . However, we want this subset to only access entries from row 50,000 to 60,000.
   * **hints:**
   * when subsetting for specific entries in a dataset, we can actually do the following: dataset[index, index][<condition>]
   * To access rows in a column, we specify the index to be dataset[index,]. The lefthand side is for specifying rows, the righthand side is for specifying columns

#######Q5->4->b########  
buldg\_sub2 = buldgp[50000:60000, c("Permit.Number","Proposed.Use")]

1. Now that we have two separate components of our dataset, letâs merge them together! Realistically, youâd really just create a singular subset with this information together. However, we have a highly specific use case now: one of our subsets only refers to a portion of the entries in our dataset, while the other dataset refers to all of the entries in our dataset 1. Merging datasets requires a really, really longwinded and misleading complex function: [merge()](https://www.rdocumentation.org/packages/base/versions/3.5.2/topics/merge) (this was a miserable joke by one of your TAs, feel free to send hate mail to Sridhar). Read the documentation, understand the parameters, and merge the datasets based on the Permit.Number column into a new variable.

#######Q5->4->c########  
  
buldg\_sub = merge(buldg\_sub1,buldg\_sub2,by="Permit.Number")  
  
head(buldg\_sub,10)  
  
dim(buldg\_sub)

1. Thereâs now an interesting phenomenon regarding our dataset: even though the second subset dealt with rows 50,000 to 60,000 (~ 10k entries), our new dataset does not match the ~10k dimension! Why do you think this is? (**hint**: unique() might come in handy)

#######Q5->4->d########  
  
length(unique(buldg\_sub$Permit.Number))

## [1] 9221

#By default the data frames are merged on the columns with names they both have, but separate specifcations of the columns can be given by by.x and by.y. Columns can be specified by name, number or by a logical vector: the name "row.names" or the number 0 specifies the row names. The rows in the two data frames that match on the specified columns are extracted, and joined together. If there is more than one match, all possible matches contribute one row each.  
  
#So, in our case the 10,000 values of second dataframe matched with the ~200k values in 1st data frame and if there where more than one match it lead to their individul columns. which lead to a total of 11783 values means we have unique values as 9221 , so 2562 are the duplicates that matched more than one pair.

1. Letâs take this newly merged dataset, and alphabetize the data based on the Proposed.Use column. The [order()](https://www.rdocumentation.org/packages/base/versions/3.5.2/topics/order) function will help tremendously!

#######Q5->4->e########  
  
buldgp = buldgp[order(buldg\_sub$Proposed.Use),]  
  
head(buldgp)

1. Take a look at your new, alphabetized dataset. In the Proposed.Use column, weâre missing data for what seems to be a decent amount of columnâs entries. Normally, weâd use the is.na() or is.null() function like we did earlier to check for missing data. However, in this dataset, all empty data are actually considered to be *empty strings*. (Example: ââ). It sounds really counterintuitive but despite these entries being visibly empty, R considers them to be non-empty entries. With this in mind, letâs tackle the missing data:
   1. Find the number of missing data points in the Proposed.Use column. (Youâll need to check which entries are **empty strings**)

#######Q5->4->f########  
  
sum(buldg\_sub$Proposed.Use == "")  
  
which(buldg\_sub$Proposed.Use == "")

1. Through a stroke of luck, Dr. Felder recently stumbled on a bit of cash and has decided to quit his job as a professor and invest in real estate full time! (again, a miserable joke). To help him with this, we want to replace all of the missing entries that we found in the Proposed.Use column with âfelderâs penthouseâ
   * **warning:** this is not an easy task and requires a bit of thinking. [This post](https://stackoverflow.com/questions/5824173/replace-a-value-in-a-data-frame-based-on-a-conditional-if-statement) on StackOverflow is really insightful to approach this problem.
     + This post converts the existing column to a character datatype with as.character() because even though that we can see that the entries in a column are text, R sometimes encodes text-based columns as different data types. To guard against this, we use as.character().
   * **side note**: side note (optional): sometimes, we want to export datasets that we create so that others can use them! [write.csv()](http://rprogramming.net/write-csv-in-r/) is a really helpful way to write any dataframes to .csv files!

#######Q5->4->g########  
  
  
buldg\_sub$Proposed.Use[buldg\_sub$Proposed.Use == ""] <- "felder's penthouse"  
  
sum(buldg\_sub$Proposed.Use == "felder's penthouse")

### Feedback

1. How long did it take to complete this homework? -> Almost 2 days
2. How difficult was the homework? -> 6 (on scale of 1-10) it was not difficult just the thing is it had many questions. So, it took time .
3. Which parts did you find useful and which parts were less useful? -> Q5 was challenging and whole assignment is usefull.
4. What suggestions do you have regarding the lectures or homework assignments that would improve them? -> instead of putting questions in Rmd file you can just put chunks and write their respesctive question numbers so that it becomes easy to navigate and code looks much clean. whereas in this current scenario it becomes too crowded and much harder to navigate to any sub question. just put the questions in assignmnet pdf we can reffer question from their. example :

####### this is an example for Question number#####  
# this above line helps to navigate to respective chunks.