Data Preparation Homework I

Allen Rahrooh

Problem 1

Take the data sets Append_1.csv, and Append_2.csv and append the two sets together. Name the new data set Append.

```
library(data.table)

Append_1 <- fread("Append_1.csv")
Append_1 <- as.data.frame(Append_1)

Append_2 <- fread("Append_2.csv")
Append_2 <- as.data.frame(Append_2)

Append <- rbind(Append_1, Append_2)
Append
```

```
##   V1       Id Score
## 1  1 78917851    13
## 2  2 34554367    77
## 3  3 22173883    10
## 4  1 56993289    72
## 5  2 26856261    51
## 6  3 33921834    99
## 7  4 97613637    63
## 8  5 78816868    28
## 9  6 67731229    17
```

Problem 2

Take the data sets Merge_1.csv and Merge_2.csv and perform an

```
library(dplyr)
library(data.table)

Merge_1 <- fread("Merge_1.csv")
Merge_1 <- as.data.frame(Merge_1)

Merge_2 <- fread("Merge_2.csv")
Merge_2 <- as.data.frame(Merge_2)
```

Inner Join

```
Inner <- inner_join(Merge_1, Merge_2, by = c('Id'))
Inner
```

```
##   V1.x       Id Score.x V1.y Score.y
## 1    7 22113381      69    7      69
## 2    9 31937926      98    9      98
## 3    2 17245265      41    2      41
## 4   10 92922546      67   10      67
## 5    1 31674694      96    1      96
```

Left Join

```
Left <- left_join(Merge_1, Merge_2, by= c('Id'))
Left
```

```
##   V1.x       Id Score.x V1.y Score.y
## 1    4 68134933      71   NA      NA
## 2    7 22113381      69    7      69
## 3    9 31937926      98    9      98
## 4    2 17245265      41    2      41
## 5    3 42428425       9   NA      NA
## 6   10 92922546      67   10      67
## 7    1 31674694      96    1      96
```

Right Join

```
Right <- right_join(Merge_1, Merge_2, by = ('Id'))
Right
```

```
##   V1.x       Id Score.x V1.y Score.y
## 1   NA 23525437      NA    8      54
## 2    7 22113381      69    7      69
## 3    9 31937926      98    9      98
## 4    2 17245265      41    2      41
## 5   10 92922546      67   10      67
## 6   NA 38672872      NA    6      76
## 7    1 31674694      96    1      96
```

Full Join

```
Full <- full_join(Merge_1, Merge_2, by = c('Id'))
Full
```

```
##   V1.x       Id Score.x V1.y Score.y
## 1    4 68134933      71   NA      NA
## 2    7 22113381      69    7      69
## 3    9 31937926      98    9      98
## 4    2 17245265      41    2      41
## 5    3 42428425       9   NA      NA
## 6   10 92922546      67   10      67
## 7    1 31674694      96    1      96
## 8   NA 23525437      NA    8      54
## 9   NA 38672872      NA    6      76
```

Problem 3

Take the Filter.csv data set, and filter the data so that the new data set has

```
library(readr)
Filter <- read_csv("Filter.csv")
Filter <- as.data.frame(Filter)
```

1. Only rows where Id is a vowel
2. Only columns where the column means of the original data are positive

```
library(dplyr)
library(magrittr)

#calculating column means and storing as a data frame
Filter_Means <- Filter[,3:12]
Filter_Means <- colMeans(Filter_Means)
Filter_Means <- as.data.frame(Filter_Means)
Filter_Means
```

```
##     Filter_Means
## V1    -1.7546089
## V2    -1.0486212
## V3    -0.6699354
## V4    -0.3875917
## V5    -0.1561851
## V6     0.1135856
## V7     0.3730993
## V8     0.6572796
## V9     1.0049938
## V10    1.6893417
```

```
#extracting the vowels
Filter_letter <- Filter %>% filter(Id %in% c("a", "e", "o", "i", "u"))
Filter_letter
```

```
##    X1 Id        V1         V2         V3         V4          V5          V6
## 1   1  a -3.131767 -1.2961544 -0.8407142 -0.5113045 -0.27692720 -0.03008066
## 2   2  a -2.753743 -1.2934195 -0.8300568 -0.5102201 -0.27176500 -0.02936955
## 3   3  a -2.717051 -1.2886454 -0.8286109 -0.5091529 -0.27112787 -0.02787884
## 4   4  a -2.475701 -1.2799619 -0.8242495 -0.4996810 -0.26609603 -0.02742792
## 5  21  e -2.071032 -1.1678474 -0.7600930 -0.4617289 -0.22786285  0.03344387
## 6  22  e -2.063066 -1.1586161 -0.7572115 -0.4572410 -0.22568412  0.03551959
## 7  35  i -1.846590 -1.1048029 -0.7270832 -0.4107124 -0.20191588  0.07267170
## 8  36  i -1.816983 -1.1025278 -0.7226210 -0.4102414 -0.19949265  0.07301385
## 9  55  o -1.625952 -1.0201384 -0.6551350 -0.3717132 -0.14393427  0.14060135
## 10 56  o -1.624472 -1.0112439 -0.6528754 -0.3714763 -0.14312152  0.14614447
## 11 57  o -1.623765 -1.0079263 -0.6507698 -0.3712410 -0.14201603  0.14924360
## 12 58  o -1.613101 -1.0060281 -0.6431687 -0.3703086 -0.14030381  0.15120047
## 13 83  u -1.402569 -0.9081516 -0.5612001 -0.3131085 -0.06864142  0.19386525
## 14 84  u -1.399668 -0.9069317 -0.5600069 -0.3106948 -0.06417877  0.20373507
```

```
## 15 85  u -1.398094 -0.9043283 -0.5537997 -0.3079776 -0.06169715  0.20418094
##           V7        V8        V9      V10
## 1  0.2285643 0.5091338 0.8164077 1.209791
## 2  0.2285739 0.5104243 0.8174819 1.215493
## 3  0.2305320 0.5116415 0.8184036 1.224739
## 4  0.2353110 0.5152766 0.8190109 1.225300
## 5  0.2843961 0.5598955 0.8715923 1.354335
## 6  0.2893904 0.5605851 0.8736125 1.359896
## 7  0.3268028 0.6034912 0.9369523 1.461178
## 8  0.3309897 0.6133755 0.9377772 1.469812
## 9  0.3952787 0.6623323 1.0254306 1.634369
## 10 0.3959874 0.6640933 1.0325190 1.635442
## 11 0.4004178 0.6645465 1.0370777 1.637068
## 12 0.4039819 0.6654216 1.0393211 1.645503
## 13 0.4652579 0.7692250 1.1353363 2.136592
## 14 0.4663985 0.7713248 1.1381021 2.204738
## 15 0.4675145 0.7826525 1.1398751 2.205843
```

```
#dropping columns with negative means
Vowels <- Filter_letter[-c(3:7)]
Vowels
```

```
##    X1 Id          V6        V7        V8        V9      V10
## 1   1  a -0.03008066 0.2285643 0.5091338 0.8164077 1.209791
## 2   2  a -0.02936955 0.2285739 0.5104243 0.8174819 1.215493
## 3   3  a -0.02787884 0.2305320 0.5116415 0.8184036 1.224739
## 4   4  a -0.02742792 0.2353110 0.5152766 0.8190109 1.225300
## 5  21  e  0.03344387 0.2843961 0.5598955 0.8715923 1.354335
## 6  22  e  0.03551959 0.2893904 0.5605851 0.8736125 1.359896
## 7  35  i  0.07267170 0.3268028 0.6034912 0.9369523 1.461178
## 8  36  i  0.07301385 0.3309897 0.6133755 0.9377772 1.469812
## 9  55  o  0.14060135 0.3952787 0.6623323 1.0254306 1.634369
## 10 56  o  0.14614447 0.3959874 0.6640933 1.0325190 1.635442
## 11 57  o  0.14924360 0.4004178 0.6645465 1.0370777 1.637068
## 12 58  o  0.15120047 0.4039819 0.6654216 1.0393211 1.645503
## 13 83  u  0.19386525 0.4652579 0.7692250 1.1353363 2.136592
## 14 84  u  0.20373507 0.4663985 0.7713248 1.1381021 2.204738
## 15 85  u  0.20418094 0.4675145 0.7826525 1.1398751 2.205843
```

Problem 4

Take the Filter.csv data set, and take a simple random sample of the data with ten rows. Name the new data set SRS_Filter

```
library(readr)
set.seed(831)
Filter <- read_csv("Filter.csv")
Filter <- as.data.frame(Filter)
SRS_Filter <-  Filter[sample(nrow(Filter), 10), ]
SRS_Filter
```

```
##    X1 Id        V1         V2         V3         V4          V5          V6
```

```
## 52 52  m -1.641384 -1.0320225 -0.6709809 -0.3755183 -0.16222181  0.12650838
## 71 71  r -1.512308 -0.9615876 -0.5905187 -0.3389771 -0.10746624  0.17044025
## 69 69  q -1.519675 -0.9630305 -0.6002471 -0.3405831 -0.10836918  0.16892613
## 1   1  a -3.131767 -1.2961544 -0.8407142 -0.5113045 -0.27692720 -0.03008066
## 58 58  o -1.613101 -1.0060281 -0.6431687 -0.3703086 -0.14030381  0.15120047
## 47 47  l -1.674005 -1.0578920 -0.6805481 -0.3863153 -0.16796454  0.11393470
## 29 29  g -1.907103 -1.1273020 -0.7467367 -0.4335874 -0.21200497  0.05142802
## 75 75  s -1.463181 -0.9473371 -0.5822273 -0.3251911 -0.10393951  0.17472742
## 86 86  v -1.379663 -0.8998828 -0.5536274 -0.3040222 -0.06093693  0.20479320
## 39 39  j -1.775011 -1.0919994 -0.7198380 -0.4069794 -0.18951151  0.08762975
##           V7        V8        V9      V10
## 52 0.3840244 0.6529226 1.0218645 1.625707
## 71 0.4392067 0.7333447 1.0993459 1.814628
## 69 0.4348871 0.7235151 1.0908150 1.804995
## 1  0.2285643 0.5091338 0.8164077 1.209791
## 58 0.4039819 0.6654216 1.0393211 1.645503
## 47 0.3568159 0.6458548 0.9872944 1.571017
## 29 0.3078841 0.5854739 0.9104246 1.421806
## 75 0.4563045 0.7451636 1.1233107 1.879640
## 86 0.4736966 0.7849710 1.1419960 2.208003
## 39 0.3382315 0.6166692 0.9495051 1.488104
```

Problem 5

Randomly partition the Filter.csv data set into three subsets.

For each of the three subsets, print the first three rows and the dimensions of the data set.

```
set.seed(821)
library(readr)
Filter <- read_csv("Filter.csv")
Filter <- as.data.frame(Filter)
index <- sample(seq(1,3), size = nrow(Filter), replace = TRUE, prob = c(0.7, 0.15, 0.15))
```

Train (70% of your data)

```
train <- Filter[index == 1,]
train[1:3,]
```

```
##   X1 Id        V1        V2         V3         V4         V5           V6
## 1  1  a -3.131767 -1.296154 -0.8407142 -0.5113045 -0.2769272 -0.030080660
## 5  5  b -2.450370 -1.267123 -0.8221215 -0.4981733 -0.2649640 -0.019236785
## 6  6  b -2.382935 -1.267034 -0.8202799 -0.4976726 -0.2636937 -0.007223675
##          V7        V8        V9      V10
## 1 0.2285643 0.5091338 0.8164077 1.209791
## 5 0.2379641 0.5158717 0.8215711 1.236532
## 6 0.2383893 0.5178508 0.8235384 1.250910
```

Dimension of Train

```
dim(train)
```

```
## [1] 63 12
```

Validation (15% of your data)

```
validation <- Filter[index == 2,]
validation[1:3,]
```

```
##    X1 Id        V1        V2         V3         V4         V5           V6
## 4   4  a -2.475701 -1.279962 -0.8242495 -0.4996810 -0.2660960 -0.027427916
## 8   8  b -2.334174 -1.242407 -0.8138314 -0.4930041 -0.2561289 -0.001366260
## 12 12  c -2.229638 -1.226936 -0.7912583 -0.4861654 -0.2512839  0.006022347
##           V7        V8        V9      V10
## 4  0.2353110 0.5152766 0.8190109 1.225300
## 8  0.2492938 0.5251589 0.8328762 1.260697
## 12 0.2569421 0.5315225 0.8479204 1.283832
```

Dimension of Validation

```
dim(validation)
```

```
## [1] 22 12
```

Test (15% of your data)

```
test <- Filter[index == 3,]
test[1:3,]
```

```
##    X1 Id        V1        V2         V3         V4         V5           V6
## 2   2  a -2.753743 -1.293420 -0.8300568 -0.5102201 -0.2717650 -0.029369554
## 3   3  a -2.717051 -1.288645 -0.8286109 -0.5091529 -0.2711279 -0.027878837
## 10 10  c -2.288709 -1.232114 -0.8117702 -0.4886991 -0.2542612  0.004448164
##           V7        V8        V9      V10
## 2  0.2285739 0.5104243 0.8174819 1.215493
## 3  0.2305320 0.5116415 0.8184036 1.224739
## 10 0.2548153 0.5265355 0.8390865 1.278628
```

Dimension of Test

```
dim(test)
```

```
## [1] 15 12
```