Date of run: 10 May 2023

Data set: 311 Service Requests from 1 Jan – 31 Mar 2023

Number of rows in the 311 SR data set: 718,801

> areAllDates <- function ( dateField ) {

+ allDates <- suppressWarnings( !is.na( as.Date( dateField[dateField != ""],format = "%m/%d/%Y %I:%M:%S %p" ) ) )

+ return ( all( allDates ) )

+ }

> cat( "Are all the values in the 'created\_date' field dates?", areAllDates( d311$created\_date ) )

Are all the values in the 'created\_date' field dates? TRUE

Are all the values in the 'closed\_date' field dates? TRUE

Are all the values in the 'due\_date' field dates? TRUE

Are all the values in the 'resolution\_action\_updated\_date' field dates? TRU

> areAllNumbers <- function ( numberField ){

+ allNumbers <- suppressWarnings( !is.na( as.numeric( numberField[numberField != ""] ) ) )

+ return ( all( allNumbers ) )

+ }

> cat( "Are all the values in the 'incident\_zip' field numbers?", areAllNumbers( d311$incident\_zip ) )

Are all the values in the 'incident\_zip' field numbers? FALSE

Are all the values in the 'x\_coordinate\_state\_plane' field numbers? TRUE

Are all the values in the 'y\_coordinate\_state\_plane' field numbers? TRUE

Are all the values in the 'latitude' field numbers? TRUE

Are all the values in the 'longitude' field numbers? TRUE

Are all the values in the 'zip\_codes' field numbers? TRUE

Are all the values in the 'community\_districts' field numbers? TRUE

Are all the values in the 'borough\_boundaries' field numbers? TRUE

Are all the values in the 'city\_council\_district' field numbers? TRUE

Are all the values in the 'police\_precincts' field numbers?

> #Validate that all fields are in the list of allowable values

>

> areInList <- function ( dataset, listValidValues ){

+ dataset <- dataset[dataset != ""]

+ inList <- ( dataset %in% listValidValues )

+ return ( all( inList ) )

+ }

> cat( "Are all the values in the 'borough' field valid?", areInList( d311$borough, c("BRONX", "BROOKLYN", "MANHATTAN", "QUEENS", "STATEN ISLAND", "Unspecified") ) )

Are all the values in the 'borough' field valid? TRUE

Are all the values in the 'borough\_boundaries' field valid? TRUE

Are all the values in the 'park\_borough' field valid? TRUE

Are all the values in the 'open\_data\_channel\_type'valid? TRUE

Are all the values in the 'police\_precinct valid? FALSE

Are all the values in the 'city\_council\_district valid? TRUE

The number of invalid zip codes in the 'incident\_zip' field is 503

representing 0.07% of non-blank data.

>print( head( badZipcodes1,15 ) )

agency invalid\_zip unique\_id

1 NYPD 10278 57192526

2 NYPD 10278 57192304

3 DOT 10000 57189036

4 DOB 10055 57189815

5 DCA 10179 57185954

6 NYPD 10000 57179059

7 EDC 10000 57171887

8 DOT 10000 57178337

9 TLC 10045 57170534

10 DCA 78758 57170282

11 NYPD 10278 57170613

12 TLC 10121 57163731

13 NYPD 10278 57169148

14 TLC 10121 57159760

15 DSNY 11359 57155904

sortedData <- as.data.frame(table(badZipcodes1$agency))

> sortedData <- sortedData[order(-sortedData$Freq),]

> print(sortedData)

Var1 Freq

11 NYPD 120

6 DOT 118

9 EDC 77

12 TLC 69

1 DCA 38

7 DPR 37

5 DOHMH 12

2 DEP 8

3 DHS 8

8 DSNY 8

4 DOB 7

10 HPD 1

# of bad zip codes in 'zip\_codes' field is 518,487 representing 72.13% of

overall data.

> print( head( badZipcodes2, 15 ) )

agency invalid\_zip unique\_id

1 NYPD 10934 57190759

2 NYPD 13828 57197361

3 NYPD 11608 57198624

4 NYPD 17614 57198077

5 TLC 24340 57194788

6 TLC 12081 57191820

7 NYPD 18180 57192379

8 NYPD 14193 57191684

9 NYPD 13512 57198090

10 NYPD 18181 57195926

11 NYPD 12081 57193808

12 NYPD 17616 57193031

13 NYPD 15310 57192951

14 NYPD 13829 57198271

15 NYPD 15311 57191636

> sortedData <- as.data.frame(table(badZipcodes2$agency))

> sortedData <- sortedData[order(-sortedData$Freq),]

> print(sortedData)

Var1 Freq

13 NYPD 210174

11 HPD 139579

9 DSNY 45436

7 DOT 36561

2 DEP 25049

4 DOB 16451

6 DOHMH 12964

8 DPR 12662

10 EDC 7657

15 TLC 4104

1 DCA 3639

3 DHS 3195

12 NYC311-PRD 771

5 DOE 217

14 OTI 28

The number of non-matching boroughs between 'borough' and

'borough\_boundaries' is 466 representing 0.07% of non-blank data.

> if ( nrow( nonMatchingBouroughs ) > 0 ) {

+ cat("\n Sample of non-mathcing boroughs\n")

+ print(head( nonMatchingBouroughs,15 ) )

+ }

Sample of non-mathcing boroughs

key borough borough\_boundaries agency

254 56798520 BRONX MANHATTAN DCA

214 56857750 BRONX MANHATTAN DCA

131 56974520 BRONX MANHATTAN DCA

402 56527366 BRONX MANHATTAN DEP

372 56576474 BRONX MANHATTAN DEP

350 56599180 BRONX MANHATTAN DEP

292 56726423 BRONX MANHATTAN DEP

291 56729966 BRONX MANHATTAN DEP

277 56754096 BRONX MANHATTAN DEP

220 56855293 BRONX MANHATTAN DEP

172 56932804 BRONX MANHATTAN DEP

128 56968389 BRONX MANHATTAN DEP

134 56971592 BRONX MANHATTAN DEP

133 56971835 BRONX MANHATTAN DEP

124 56976124 BRONX MANHATTAN DEP

> sortedData <- as.data.frame(table( nonMatchingBouroughs$agency))

> sortedData <- sortedData[order(-sortedData$Freq),]

> print(sortedData)

Var1 Freq

10 NYPD 362

7 DOT 37

2 DEP 23

4 DOB 13

9 DSNY 13

8 DPR 9

6 DOHMH 4

1 DCA 3

3 DHS 1

5 DOE 1

> print( missingDataPerColumn[ order( -missingDataPerColumn[ , 2] ), ])

Number and fraction of blank values in each column, sorted descending

field #blanks %blank #unspecified %unspecified #unknowns %Unknown

32 vehicle\_type 718631 0.9998 0 0.0000 0 0.0000

33 taxi\_company\_borough 718360 0.9994 0 0.0000 0 0.0000

37 road\_ramp 717064 0.9976 0 0.0000 0 0.0000

21 due\_date 716721 0.9971 0 0.0000 0 0.0000

36 bridge\_highway\_direction 716532 0.9968 0 0.0000 0 0.0000

35 bridge\_highway\_name 714512 0.9940 0 0.0000 0 0.0000

38 bridge\_highway\_segment 714511 0.9940 0 0.0000 0 0.0000

34 taxi\_pick\_up\_location 711060 0.9892 0 0.0000 0 0.0000

19 facility\_type 663099 0.9225 0 0.0000 0 0.0000

18 landmark 330994 0.4605 0 0.0000 0 0.0000

14 intersection\_street\_1 277131 0.3855 0 0.0000 0 0.0000

15 intersection\_street\_2 276710 0.3850 0 0.0000 0 0.0000

12 cross\_street\_1 250338 0.3483 0 0.0000 0 0.0000

13 cross\_street\_2 250221 0.3481 0 0.0000 0 0.0000

8 location\_type 96434 0.1342 0 0.0000 0 0.0000

25 bbl 82551 0.1148 0 0.0000 0 0.0000

3 closed\_date 41406 0.0576 0 0.0000 0 0.0000

11 street\_name 34164 0.0475 0 0.0000 0 0.0000

10 incident\_address 34157 0.0475 0 0.0000 0 0.0000

17 city 30176 0.0420 0 0.0000 0 0.0000

42 zip\_codes 14875 0.0207 0 0.0000 0 0.0000

43 community\_districts 12774 0.0178 0 0.0000 0 0.0000

45 city\_council\_districts 12774 0.0178 0 0.0000 0 0.0000

46 police\_precincts 12774 0.0178 0 0.0000 0 0.0000

44 borough\_boundaries 12773 0.0178 0 0.0000 0 0.0000

39 latitude 12638 0.0176 0 0.0000 0 0.0000

40 longitude 12638 0.0176 0 0.0000 0 0.0000

41 location 12638 0.0176 0 0.0000 0 0.0000

27 x\_coordinate\_state\_plane 12622 0.0176 0 0.0000 0 0.0000

28 y\_coordinate\_state\_plane 12533 0.0174 0 0.0000 0 0.0000

22 resolution\_description 10356 0.0144 0 0.0000 0 0.0000

9 incident\_zip 7855 0.0109 0 0.0000 0 0.0000

23 resolution\_action\_updated\_date 6537 0.0091 0 0.0000 0 0.0000

7 descriptor 4710 0.0066 0 0.0000 0 0.0000

16 address\_type 3688 0.0051 0 0.0000 0 0.0000

1 unique\_key 0 0.0000 0 0.0000 0 0.0000

2 created\_date 0 0.0000 0 0.0000 0 0.0000

4 agency 0 0.0000 0 0.0000 0 0.0000

5 agency\_name 0 0.0000 0 0.0000 0 0.0000

6 complaint\_type 0 0.0000 0 0.0000 0 0.0000

20 status 0 0.0000 200 0.0003 0 0.0000

24 community\_board 0 0.0000 0 0.0000 0 0.0000

26 borough 0 0.0000 1122 0.0016 0 0.0000

29 open\_data\_channel\_type 0 0.0000 0 0.0000 66893 0.0931

30 park\_facility\_name 0 0.0000 717373 0.9980 0 0.0000

31 park\_borough 0 0.0000 1122 0.0016 0 0.0000

The number of SRs 'closed' before 'opened' is 2,039 which is 0.03% of all

non-blank data.

> if ( nrow( closedBeforOpened ) > 0 ) {

+ print( head( closedBeforeOpened, 13 ) )

Largest errors:

unique\_key created\_date closed\_date duration agency

1510 56636409 2023-01-27 14:40:00 2022-01-14 14:40:00 -378 DOT

1742 56559854 2023-01-18 10:06:00 2022-01-12 10:06:00 -371 DOT

1511 56636360 2023-01-27 14:36:00 2022-01-22 14:35:00 -370 DOT

1850 56505799 2023-01-11 11:10:00 2022-01-09 11:10:00 -367 DOT

1760 56555119 2023-01-17 11:58:00 2022-06-13 11:58:00 -218 DOT

1761 56549513 2023-01-17 11:57:00 2022-06-13 11:57:00 -218 DOT

1762 56554327 2023-01-17 11:56:00 2022-06-13 11:55:00 -218 DOT

1764 56547641 2023-01-17 11:54:00 2022-06-13 11:54:00 -218 DOT

1409 56694533 2023-02-02 14:36:00 2022-08-03 14:36:00 -183 DOT

1408 56693941 2023-02-02 14:39:00 2022-08-04 14:39:00 -182 DOT

1548 56619660 2023-01-25 14:58:00 2022-08-01 14:58:00 -177 DOT

262 57040556 2023-03-14 17:42:00 2022-11-30 17:42:00 -104 DOT

1072 56813092 2023-02-15 14:17:00 2022-12-28 14:16:00 -49 DOT

> tail(closedBeforeOpened, 10)

Smallest errors:

unique\_key created\_date closed\_date duration agency

1980 56431351 2023-01-03 11:44:00 2023-01-02 11:43:00 -1.00 DOT

970 56865903 2023-02-17 08:33:40 2023-02-17 07:46:00 -0.03 DOT

1262 56760914 2023-02-08 08:57:06 2023-02-08 08:15:00 -0.03 DOT

659 56932742 2023-02-28 07:24:08 2023-02-28 07:00:00 -0.02 DOT

1261 56764795 2023-02-08 08:58:13 2023-02-08 08:32:00 -0.02 DOT

1785 56547570 2023-01-13 11:12:55 2023-01-13 10:40:00 -0.02 DOT

909 56873136 2023-02-21 08:36:50 2023-02-21 08:20:00 -0.01 DOT

968 56859069 2023-02-17 08:37:54 2023-02-17 08:26:00 -0.01 DOT

969 56860177 2023-02-17 08:34:23 2023-02-17 08:21:00 -0.01 DOT

1268 56759382 2023-02-08 08:28:38 2023-02-08 08:12:00 -0.01 DOT

print( as.data.frame( table( closedBeforeOpened$agency ) ) )

Var1 Freq

1 DOT 2038

2 DSNY 1

The number of bad latitudes is: 0

The number of bad longitudes is 0

> is\_unique <- length(unique(d311$unique\_key)) == nrow(d311)

> is\_unique

[1] TRUE

> table(d311$facility\_type)

DSNY Garage N/A

663099 4211 51491

> table(d311$vehicle\_type)

Ambulette / Paratransit Car Service Commuter Van

718631 5 158 7

|  |
| --- |
| Number of rows in the 311 SR data set: 718,801  >  > #########################################################################  >  > # Look for invalid zipcodes in the'incident\_zip'field, which is densely populated  > badZipcodes1 <- findInvalidZipcodes( USPSzipcodesOnly, d311,  + which( colnames( d311 ) == "unique\_key" ),  + which( colnames( d311 ) == "incident\_zip" ),  + which( colnames( d311 ) == "agency") )  Time spent in lookup of ALL dataset zipcodes %in% invalidZipcodes: 31.30 seconds.>  > cat("\nThe number of invalid zip codes in the 'incident\_zip' field is", format( nrow( badZipcodes1 ), big.mark = ",", scientific = FALSE ), "representing",  + percent( nrow( badZipcodes1 )/numRows, accuracy = 0.01 ),  + "of overall data.")  The number of invalid zip codes in the 'incident\_zip' field is 503 representing 0.07% of overall data.> print( head( badZipcodes1,15 ) )  agency invalid\_zip unique\_id  1 NYPD 10278 57192526  2 NYPD 10278 57192304  3 DOT 10000 57189036  4 DOB 10055 57189815  5 DCA 10179 57185954  6 NYPD 10000 57179059  7 EDC 10000 57171887  8 DOT 10000 57178337  9 TLC 10045 57170534  10 DCA 78758 57170282  11 NYPD 10278 57170613  12 TLC 10121 57163731  13 NYPD 10278 57169148  14 TLC 10121 57159760  15 DSNY 11359 57155904  > #print( as.data.frame( table( badZipcodes1$agency ) ) )  > sortedData <- as.data.frame(table(badZipcodes1$agency))  > sortedData <- sortedData[order(-sortedData$Freq),]  > print(sortedData)  Var1 Freq  11 NYPD 120  6 DOT 118  9 EDC 77  12 TLC 69  1 DCA 38  7 DPR 37  5 DOHMH 12  2 DEP 8  3 DHS 8  8 DSNY 8  4 DOB 7  10 HPD 1  >  > # Look for invalid zipcodes in the 'zip\_codes' field, Expect a large #  > badZipcodes2 <- findInvalidZipcodes( USPSzipcodesOnly, d311,  + which( colnames( d311 ) == "unique\_key" ),  + which( colnames( d311 ) == "zip\_codes" ),  + which( colnames( d311 ) == "agency" ) )  Time spent in lookup of ALL dataset zipcodes %in% invalidZipcodes: 17320.07 seconds.>  > #stopTime <- as.POSIXct( Sys.time() )  > #stopTimeFormatted <- format( stopTime, "%H:%M:%S" )  > #cat( "\nEnds at", stopTimeFormatted )  > #cat( "\nTime spent in the findBadZipcodes function is: ", sprintf( "%.2f", difftime( stopTime, startTime, units = "mins")), "minutes." )  >  > cat( "# of bad zip codes in 'zip\_codes' field is", format( nrow( badZipcodes2 ), big.mark = ",", scientific = FALSE ), "representing",  + percent( nrow( badZipcodes2 )/numRows, accuracy = 0.01 ),  + "of overall data." )  # of bad zip codes in 'zip\_codes' field is 518,487 representing 72.13% of overall data.>  > print( head( badZipcodes2, 15 ) )  agency invalid\_zip unique\_id  1 NYPD 10934 57190759  2 NYPD 13828 57197361  3 NYPD 11608 57198624  4 NYPD 17614 57198077  5 TLC 24340 57194788  6 TLC 12081 57191820  7 NYPD 18180 57192379  8 NYPD 14193 57191684  9 NYPD 13512 57198090  10 NYPD 18181 57195926  11 NYPD 12081 57193808  12 NYPD 17616 57193031  13 NYPD 15310 57192951  14 NYPD 13829 57198271  15 NYPD 15311 57191636  > #print( as.data.frame( table( badZipcodes2$agency ) ) )  > sortedData <- as.data.frame(table(badZipcodes2$agency))  > sortedData <- sortedData[order(-sortedData$Freq),]  > print(sortedData)  Var1 Freq  13 NYPD 210174  11 HPD 139579  9 DSNY 45436  7 DOT 36561  2 DEP 25049  4 DOB 16451  6 DOHMH 12964  8 DPR 12662  10 EDC 7657  15 TLC 4104  1 DCA 3639  3 DHS 3195  12 NYC311-PRD 771  5 DOE 217  14 OTI 28  >  > #########################################################################  >  > nonMatchingBouroughs <- mismatchedBoroughs( d311 )  > cat( "\nThe number of non-matching boroughs between the 'borough' and the 'borough\_boundaries' fields is", nrow( nonMatchingBouroughs ) )  The number of non-matching boroughs between the 'borough' and the 'borough\_boundaries' fields is 466> print(head( nonMatchingBouroughs,15 ) )  key borough borough\_boundaries agency  254 56798520 BRONX MANHATTAN DCA  214 56857750 BRONX MANHATTAN DCA  131 56974520 BRONX MANHATTAN DCA  402 56527366 BRONX MANHATTAN DEP  372 56576474 BRONX MANHATTAN DEP  350 56599180 BRONX MANHATTAN DEP  292 56726423 BRONX MANHATTAN DEP  291 56729966 BRONX MANHATTAN DEP  277 56754096 BRONX MANHATTAN DEP  220 56855293 BRONX MANHATTAN DEP  172 56932804 BRONX MANHATTAN DEP  128 56968389 BRONX MANHATTAN DEP  134 56971592 BRONX MANHATTAN DEP  133 56971835 BRONX MANHATTAN DEP  124 56976124 BRONX MANHATTAN DEP  > #print( as.data.frame( table( nonMatchingBouroughs$agency ) ) )  >  > sortedData <- as.data.frame(table( nonMatchingBouroughs$agency))  > sortedData <- sortedData[order(-sortedData$Freq),]  > print(sortedData)  Var1 Freq  10 NYPD 362  7 DOT 37  2 DEP 23  4 DOB 13  9 DSNY 13  8 DPR 9  6 DOHMH 4  1 DCA 3  3 DHS 1  5 DOE 1  >  > #########################################################################  >  > missingDataPerColumn <- countColumnsMissingData( d311 )  > missingDataPerColumn[, sapply( missingDataPerColumn, is.numeric )] <- round( missingDataPerColumn[, sapply(missingDataPerColumn, is.numeric )], 4)  > cat( "\n Number and fraction of blank values in each column, sorted descending" )  Number and fraction of blank values in each column, sorted descending> print( missingDataPerColumn[ order( -missingDataPerColumn[ , 2] ), ])  field #blanks %blank #unspecified %unspecified #unknowns %Unknown  32 vehicle\_type 718631 0.9998 0 0.0000 0 0.0000  33 taxi\_company\_borough 718360 0.9994 0 0.0000 0 0.0000  37 road\_ramp 717064 0.9976 0 0.0000 0 0.0000  21 due\_date 716721 0.9971 0 0.0000 0 0.0000  36 bridge\_highway\_direction 716532 0.9968 0 0.0000 0 0.0000  35 bridge\_highway\_name 714512 0.9940 0 0.0000 0 0.0000  38 bridge\_highway\_segment 714511 0.9940 0 0.0000 0 0.0000  34 taxi\_pick\_up\_location 711060 0.9892 0 0.0000 0 0.0000  19 facility\_type 663099 0.9225 0 0.0000 0 0.0000  18 landmark 330994 0.4605 0 0.0000 0 0.0000  14 intersection\_street\_1 277131 0.3855 0 0.0000 0 0.0000  15 intersection\_street\_2 276710 0.3850 0 0.0000 0 0.0000  12 cross\_street\_1 250338 0.3483 0 0.0000 0 0.0000  13 cross\_street\_2 250221 0.3481 0 0.0000 0 0.0000  8 location\_type 96434 0.1342 0 0.0000 0 0.0000  25 bbl 82551 0.1148 0 0.0000 0 0.0000  3 closed\_date 41406 0.0576 0 0.0000 0 0.0000  11 street\_name 34164 0.0475 0 0.0000 0 0.0000  10 incident\_address 34157 0.0475 0 0.0000 0 0.0000  17 city 30176 0.0420 0 0.0000 0 0.0000  42 zip\_codes 14875 0.0207 0 0.0000 0 0.0000  43 community\_districts 12774 0.0178 0 0.0000 0 0.0000  45 city\_council\_districts 12774 0.0178 0 0.0000 0 0.0000  46 police\_precincts 12774 0.0178 0 0.0000 0 0.0000  44 borough\_boundaries 12773 0.0178 0 0.0000 0 0.0000  39 latitude 12638 0.0176 0 0.0000 0 0.0000  40 longitude 12638 0.0176 0 0.0000 0 0.0000  41 location 12638 0.0176 0 0.0000 0 0.0000  27 x\_coordinate\_state\_plane 12622 0.0176 0 0.0000 0 0.0000  28 y\_coordinate\_state\_plane 12533 0.0174 0 0.0000 0 0.0000  22 resolution\_description 10356 0.0144 0 0.0000 0 0.0000  9 incident\_zip 7855 0.0109 0 0.0000 0 0.0000  23 resolution\_action\_updated\_date 6537 0.0091 0 0.0000 0 0.0000  7 descriptor 4710 0.0066 0 0.0000 0 0.0000  16 address\_type 3688 0.0051 0 0.0000 0 0.0000  1 unique\_key 0 0.0000 0 0.0000 0 0.0000  2 created\_date 0 0.0000 0 0.0000 0 0.0000  4 agency 0 0.0000 0 0.0000 0 0.0000  5 agency\_name 0 0.0000 0 0.0000 0 0.0000  6 complaint\_type 0 0.0000 0 0.0000 0 0.0000  20 status 0 0.0000 200 0.0003 0 0.0000  24 community\_board 0 0.0000 0 0.0000 0 0.0000  26 borough 0 0.0000 1122 0.0016 0 0.0000  29 open\_data\_channel\_type 0 0.0000 0 0.0000 66893 0.0931  30 park\_facility\_name 0 0.0000 717373 0.9980 0 0.0000  31 park\_borough 0 0.0000 1122 0.0016 0 0.0000  >  > #########################################################################  > ## Change the various date fields to date-time objects and reformat dates.There are four date fields in the 311 data.  > d311$created\_date <- convertToDateObject( d311$created\_date )  > d311$closed\_date <- convertToDateObject( d311$closed\_date )  > d311$due\_date <- convertToDateObject( d311$due\_date )  > d311$resolution\_action\_updated\_date <- convertToDateObject( d311$resolution\_action\_updated\_date )  >  > ## Compute and store "duration" in a new additional column for the dataframe "d311".  > ## Duration is the time between created\_date and closed\_date, although due to data errors the value may be negative.  > ## Identify the Service Records that were closed before they were create, i.e. bad dates.  > d311$duration <- round( as.numeric( difftime( d311$closed\_date, d311$created\_date, units = "days") ), 2 )  > closedBeforeOpened <- findBadDates( d311,  + which( colnames(d311) == "unique\_key" ),  + which( colnames(d311) == "created\_date" ),  + which( colnames(d311) == "closed\_date" ),  + which( colnames(d311) == "duration" ),  + which( colnames(d311) == "agency") )  Results>  > cat( "Number of SRs 'closed' before 'opened': ", format( nrow( closedBeforeOpened ), big.mark = ",", scientific = FALSE ), "\n" )  Number of SRs 'closed' before 'opened': 2,039  > if ( nrow( closedBeforOpened ) > 0 ) {  + print( head( closedBeforeOpened, 13 ) )  + print( as.data.frame( table( closedBeforeOpened$agency ) ) )  + }  unique\_key created\_date closed\_date duration agency  1510 56636409 2023-01-27 14:40:00 2022-01-14 14:40:00 -378 DOT  1742 56559854 2023-01-18 10:06:00 2022-01-12 10:06:00 -371 DOT  1511 56636360 2023-01-27 14:36:00 2022-01-22 14:35:00 -370 DOT  1850 56505799 2023-01-11 11:10:00 2022-01-09 11:10:00 -367 DOT  1760 56555119 2023-01-17 11:58:00 2022-06-13 11:58:00 -218 DOT  1761 56549513 2023-01-17 11:57:00 2022-06-13 11:57:00 -218 DOT  1762 56554327 2023-01-17 11:56:00 2022-06-13 11:55:00 -218 DOT  1764 56547641 2023-01-17 11:54:00 2022-06-13 11:54:00 -218 DOT  1409 56694533 2023-02-02 14:36:00 2022-08-03 14:36:00 -183 DOT  1408 56693941 2023-02-02 14:39:00 2022-08-04 14:39:00 -182 DOT  1548 56619660 2023-01-25 14:58:00 2022-08-01 14:58:00 -177 DOT  262 57040556 2023-03-14 17:42:00 2022-11-30 17:42:00 -104 DOT  1072 56813092 2023-02-15 14:17:00 2022-12-28 14:16:00 -49 DOT  Var1 Freq  1 DOT 2038  2 DSNY 1  >  > #########################################################################  >  > ## Change the lat/long and state\_plane fields into type "numeric".  > d311$x\_coordinate\_state\_plane <- as.numeric( d311$x\_coordinate\_state\_plane )  > d311$y\_coordinate\_state\_plane <- as.numeric (d311$y\_coordinate\_state\_plane )  >  > # Check to see if any of the latitudes or longitudes fall outside the extreme points of New York City.  > # Extreme points of the boundaries of New York City as provide by chatGPT and confirmed elsewhere.  > # Note that Longitudes (west of prime meridian) are expressed as negative values  > southernMostLatitude <- 40.477399  > northernMostLatitude <- 40.917576  > easternMostLongitude <- -73.700181  > westernMostLongitude <- -74.259090  >  > # Convert lat/long to numeric conversions for comparisons  > d311$latitude <- as.numeric( d311$latitude )  > d311$longitude <- as.numeric( d311$longitude )  >  > # Check latitudes & longitudes in 311 data to determine any outliers  > badLatitudes <- subset(d311, ( latitude < southernMostLatitude | latitude > northernMostLatitude ) & latitude != "" )  > badLongitudes <- subset(d311, ( longitude > easternMostLongitude | longitude < westernMostLongitude ) & latitude != "")  > cat("\nThe number of bad latitudes is:", nrow(badLatitudes))  The number of bad latitudes is: 0> if ( nrow( badLatitudes ) >0 ) { print(head ( badLatitudes$latitude,5 ))}  > cat("\nThe number of bad longitudes is", nrow(badLongitudes))  The number of bad longitudes is 0> if ( nrow( badLongitudes ) >0 ) { print(head ( badLongitudes$longitude,5 ))}  >  > cat("\n\n\n END OF PROGRAM")  END OF PROGRAM> ##  > ##################################################################################  > ##################################################################################  > ##################################################################################  > ##################################################################################  > ##################################################################################  > ##################################################################################  >  > ##  > # use colSums() function to check which columns have no values  > #blank\_cols <- colSums(is.na(d311) | d311 == "") == nrow(d311)  >  > # print the column names that have no values  > #names(df)[blank\_cols]  >  > # display the unique values and their count for each column  > #for (col in names(d311)) {  > # cat("Column:", col, "\n")  > # cat("Unique values and their count:\n")  > # print(table(d311[[col]]))  > # cat("\n")  > #}  >  > # display the unique values, their count, and the count of blank or missing fields for each column  > #for (col in names(d311)) {  > # cat("Column:", col, "\n")  > # cat("Unique values and their count:\n")  > # print(table(d311[[col]]))  > # cat("Blank/missing fields:", sum(is.na(d311[[col]]) | !nzchar(d311[[col]])), "\n")  > # cat("\n")  > #}  >  > # count the unique values of col2 and sort them in descending order  > #cat("Counts of unique values in borough:\n")  > #print(sort(table(d311$borough, useNA = "ifany"), decreasing = TRUE))  >  > # count the unique values of col2 and sort them in descending order  > #cat("Counts of unique values in city:\n")  > #print(sort(table(d311$city, useNA = "ifany"), decreasing = TRUE))  >  >  >  >  > ## get the duration from creation time to closing time  > #t1 <- strptime(d311$closed\_date, format = '%m/%d/%Y %I:%M:%S %p')  > #t0 <- strptime(d311$created\_date, format = '%m/%d/%Y %I:%M:%S %p')  > #tt <- as.numeric(difftime(t1, t0, units = "secs"))  >  >  > ###########################################################  > ## Exploratory cleaning  > ###########################################################  >  > ## any creation time later than closing time?  > #table(tt < 0, useNA = "ifany")  >  >  >  > ## to be organized later  > ## mean(tt[d311$agency == "NYPD"]/ 3600 > 3, na.rm = TRUE)  >  > ## table(d311$intersection\_street\_1 == d311$cross\_street\_1)  >  > ## head(subset(d311, d311$intersection\_street\_1 != d311$cross\_street\_1, select = c("intersection\_street\_1", "cross\_street\_1")))  >  > ## d311$cross\_street\_1 <- ifelse(is.na(d311$cross\_street\_1), d311$intersection\_street\_1, d311$cross\_street\_1)  >  >  > ## library(lubridate)  > ## library(ggplot2)  > ## wkday <- ifelse(wday(t0, week\_start = 1) > 5, "weekend", "weekday")  > ## str\_df <- na.omit(subset(data.frame(time = tt / 3600, day = wkday, borough = d311$borough),  > ## d311$agency == "NYPD"))  > ## str\_df <- subset(str\_df, borough != "Unspecified")  > ## pdf("nypdtime.pdf", height = 10, width = 15)  > ## ggplot(str\_df, aes(x = borough, y = time, fill = day)) +  > ## geom\_violin() +  > ## coord\_flip() +  > ## ylim(0, 24) +  > ## ylab("time to close requests to NYPD (hours)") +  > ## theme(legend.position = "top",  > ## strip.background = element\_rect(fill = "grey77", color = "grey77"))  > ## dev.off()  > tail(closedBeforeOpened)  unique\_key created\_date closed\_date duration agency  1261 56764795 2023-02-08 08:58:13 2023-02-08 08:32:00 -0.02 DOT  1785 56547570 2023-01-13 11:12:55 2023-01-13 10:40:00 -0.02 DOT  909 56873136 2023-02-21 08:36:50 2023-02-21 08:20:00 -0.01 DOT  968 56859069 2023-02-17 08:37:54 2023-02-17 08:26:00 -0.01 DOT  969 56860177 2023-02-17 08:34:23 2023-02-17 08:21:00 -0.01 DOT  1268 56759382 2023-02-08 08:28:38 2023-02-08 08:12:00 -0.01 DOT  > tail(closedBeforeOpened, 20)  unique\_key created\_date closed\_date duration agency  1962 56444336 2023-01-04 11:04:00 2023-01-03 11:03:00 -1.00 DOT  1963 56440732 2023-01-04 10:40:00 2023-01-03 10:39:00 -1.00 DOT  1966 56440411 2023-01-04 09:31:00 2023-01-03 09:30:00 -1.00 DOT  1967 56436415 2023-01-04 09:08:00 2023-01-03 09:07:00 -1.00 DOT  1974 56434549 2023-01-03 13:45:00 2023-01-02 13:44:00 -1.00 DOT  1975 56433345 2023-01-03 13:40:00 2023-01-02 13:39:00 -1.00 DOT  1976 56434550 2023-01-03 13:38:00 2023-01-02 13:36:00 -1.00 DOT  1977 56432542 2023-01-03 13:35:00 2023-01-02 13:34:00 -1.00 DOT  1978 56428644 2023-01-03 13:34:00 2023-01-02 13:33:00 -1.00 DOT  1979 56432565 2023-01-03 13:30:00 2023-01-02 13:29:00 -1.00 DOT  1980 56431351 2023-01-03 11:44:00 2023-01-02 11:43:00 -1.00 DOT  970 56865903 2023-02-17 08:33:40 2023-02-17 07:46:00 -0.03 DOT  1262 56760914 2023-02-08 08:57:06 2023-02-08 08:15:00 -0.03 DOT  659 56932742 2023-02-28 07:24:08 2023-02-28 07:00:00 -0.02 DOT  1261 56764795 2023-02-08 08:58:13 2023-02-08 08:32:00 -0.02 DOT  1785 56547570 2023-01-13 11:12:55 2023-01-13 10:40:00 -0.02 DOT  909 56873136 2023-02-21 08:36:50 2023-02-21 08:20:00 -0.01 DOT  968 56859069 2023-02-17 08:37:54 2023-02-17 08:26:00 -0.01 DOT  969 56860177 2023-02-17 08:34:23 2023-02-17 08:21:00 -0.01 DOT  1268 56759382 2023-02-08 08:28:38 2023-02-08 08:12:00 -0.01 DOT  > tail(closedBeforeOpened, 15)  unique\_key created\_date closed\_date duration agency  1975 56433345 2023-01-03 13:40:00 2023-01-02 13:39:00 -1.00 DOT  1976 56434550 2023-01-03 13:38:00 2023-01-02 13:36:00 -1.00 DOT  1977 56432542 2023-01-03 13:35:00 2023-01-02 13:34:00 -1.00 DOT  1978 56428644 2023-01-03 13:34:00 2023-01-02 13:33:00 -1.00 DOT  1979 56432565 2023-01-03 13:30:00 2023-01-02 13:29:00 -1.00 DOT  1980 56431351 2023-01-03 11:44:00 2023-01-02 11:43:00 -1.00 DOT  970 56865903 2023-02-17 08:33:40 2023-02-17 07:46:00 -0.03 DOT  1262 56760914 2023-02-08 08:57:06 2023-02-08 08:15:00 -0.03 DOT  659 56932742 2023-02-28 07:24:08 2023-02-28 07:00:00 -0.02 DOT  1261 56764795 2023-02-08 08:58:13 2023-02-08 08:32:00 -0.02 DOT  1785 56547570 2023-01-13 11:12:55 2023-01-13 10:40:00 -0.02 DOT  909 56873136 2023-02-21 08:36:50 2023-02-21 08:20:00 -0.01 DOT  968 56859069 2023-02-17 08:37:54 2023-02-17 08:26:00 -0.01 DOT  969 56860177 2023-02-17 08:34:23 2023-02-17 08:21:00 -0.01 DOT  1268 56759382 2023-02-08 08:28:38 2023-02-08 08:12:00 -0.01 DOT |
|  |
| |  | | --- | | > | |