NYC Construction Data using R

Vaishnavi S(1NT20IS182) ,J Naga Tulasi(1NT20IS066)

2023-05-22

Contents

Introduction and Report Overview 1

The Basics 3

Importing the dataset into R........ 3

Visualising Data: An Overview 5

Gathering Basic Information about the dataset 5

Converting the dataset into numerical values 6

Line Graphs 22

Bar Graphs 25

Introduction and Report Overview

This report presents a comprehensive analysis of the dataset **survey_results_schema.csv** and **survey_results_public.csv** using the R programming language, focusing on exploratory data analysis (EDA) techniques from the “R Graphics Cookbook” by Winston Chang. The goal of this analysis is to demonstrate the application of various visualization functions from specific chapters of the cookbook and provide a detailed interpretation of the output.

The analysis begins by importing the dataset into R and ensuring that all required libraries and dependencies are installed. The dataset is then read into R, enabling further exploration and manipulation of the data. To meet the requirements, this analysis will utilize visualization functions from the following chapters of the “R Graphics Cookbook”:

- **Quickly Exploring Data** (Chapter 2): This chapter provides techniques for quickly exploring and summarizing the dataset. The visualization functions presented in this chapter will help in gaining a preliminary understanding of the data’s characteristics.

- **Bar Graphs** (Chapter 3): Bar graphs are powerful tools for visualizing categorical variables. This chapter covers various types of bar graphs, including stacked and grouped bar graphs, which can effectively showcase the frequency or proportion of different categories in the dataset.

- **Line Graphs** (Chapter 4): Line graphs are ideal for visualizing trends and patterns over time or across continuous variables. This chapter explores techniques for creating line graphs that can reveal temporal variations or relationships in the dataset.

- **Scatter Plots** (Chapter 5): Scatter plots are used to visualize the relationship between two continuous variables. This chapter provides insights into creating scatter plots to identify patterns, clusters, outliers, and correlations within the dataset.

- **Summarized Data Distributions** (Chapter 6): Summarized data distributions offer a condensed view of the dataset by aggregating and summarizing values. This chapter explores techniques for visualizing summarized data distributions, such as box plots and violin plots, to gain insights into the overall distribution of numerical variables.

To document the analysis and provide a detailed interpretation of the output, R Markdown in RStudio will be used. R Markdown allows for the integration of code, visualizations, and text in a single document, facilitating the creation of reproducible reports. The output generated from the R Markdown document will be exported to PDF format, fulfilling the submission requirements.

To ensure reproducibility, the complete R script and executed commands will be shared via a GitHub repository. The repository link will be included in the report, allowing readers to review the code and reproduce the analysis.

Please refer to the subsequent sections for a detailed examination of the dataset, including the applied visualization functions, interpretations of the output, and the corresponding R code used for the analysis.

The Basics

Importing the dataset into R

```
>library(readr)

> dataset <- read_csv("Downloads/dataset.csv")
```

```
Rows: 426516 Columns: 12

── Column specification ──────────────────────────────────────────────────────────────

Delimiter: ","

chr (10): BORO, MANAGING_AGCY_CD, MANAGING_AGCY, PROJECT_ID, PROJECT_DESCR, T...

dbl  (2): PUB_DATE, SEQ_NUMBER
```

```
-  ℹ Use `spec()` to retrieve the full column specification for this data.
ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
# install.packages("ggplot2")
```

As the package is already installed , we load it into our current session using the **library()** function as shown below

```
library(ggplot2)
```

We will also be needing the **dplyr** package to manipulate data using the pipeline operator ***%>%***

```
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
```

Now we will verify if both the packages **ggplot2** and **dplyr** are loaded into the current session by performing the **search()** command again.

```
search()
```

```
## [1] ".GlobalEnv"          "package:dplyr"       "package:ggplot2"
## [4] "package:stats"       "package:graphics" "package:grDevices"
## [7] "package:utils"       "package:datasets" "package:methods"
## [10] "Autoloads"                    "package:base"
```

As seen above , **package:dplyr** and **package:ggplot2** are in the current sesssion.

Visualising Data: An Overview

Gathering Basic Information about the dataset

```
head(dataset)

# A tibble: 6 × 12

  PUB_DATE BORO     MANAGING_AGCY_CD MANAGING_AGCY   PROJECT_ID PROJECT_DESCR

     <dbl> <chr>    <chr>            <chr>           <chr>      <chr>

1 20220517 CITYWIDE 042              CITY UNIVERSITY CA202-006  ADA Compliance

2 20220517 CITYWIDE 042              CITY UNIVERSITY CA202-006  ADA Compliance

3 20220517 CITYWIDE 042              CITY UNIVERSITY CA202-006  ADA Compliance

4 20220517 CITYWIDE 042              CITY UNIVERSITY CA202-006  ADA Compliance

5 20220517 CITYWIDE 042              CITY UNIVERSITY CA202-006  ADA Compliance

6 20220517 CITYWIDE 042              CITY UNIVERSITY CA202-006  ADA Compliance

# ℹ 6 more variables: SEQ_NUMBER <dbl>, TASK_DESCRIPTION <chr>,

#   ORIG_START_DATE <chr>, ORIG_END_DATE <chr>, TASK_START_DATE <chr>,

#   TASK_END_DATE <chr>
```

```
  -
```

The above commmands **str()** gives information about the dataset we are currently using , the number of columns and a brief overview of each column in the dataset.

The **dim()** command lists the number of rows and columns of the dataset in the form of an array.
As shown in the output pane above , the dataset has 79 rows and 6 columns.

Now we will inspect each column of the dataset , and convert them into numerical data to plot graphs and perform visualization techniques.

Converting the dataset into numerical values

```
names(dataset)

 [1] "PUB_DATE"        "BORO"            "MANAGING_AGCY_CD" "MANAGING_AGCY"

 [5] "PROJECT_ID"      "PROJECT_DESCR"    "SEQ_NUMBER"      "TASK_DESCRIPTION"

 [9] "ORIG_START_DATE"  "ORIG_END_DATE"    "TASK_START_DATE"  "TASK_END_DATE"

str(dataset)

spc_tbl_ [426,516 × 12] (S3: spec_tbl_df/tbl_df/tbl/data.frame)

 $ PUB_DATE       : num [1:426516] 20220517 20220517 20220517 20220517 20220517 ...

 $ BORO           : chr [1:426516] "CITYWIDE" "CITYWIDE" "CITYWIDE" "CITYWIDE" ...

 $ MANAGING_AGCY_CD: chr [1:426516] "042" "042" "042" "042" ...
 $ MANAGING_AGCY   : chr [1:426516] "CITY UNIVERSITY" "CITY UNIVERSITY" "CITY UNIVERSITY" "CITY UNIVERSITY" ...
 $ PROJECT_ID      : chr [1:426516] "CA202-006" "CA202-006" "CA202-006" "CA202-006" ...
 $ PROJECT_DESCR   : chr [1:426516] "ADA Compliance" "ADA Compliance" "ADA Compliance" "ADA Compliance" ...
 $ SEQ_NUMBER      : num [1:426516] 4 5 6 7 8 9 10 1 2 3 ...
 $ TASK_DESCRIPTION: chr [1:426516] "BID AWARD AND REGISTER CONTRCT" "CONSTRUCTION TO 25%" "CONSTRUCTION TO 50%" "CONSTRUCTION TO 75%" ...
 $ ORIG_START_DATE : chr [1:426516] "Jun 2006" "Sep 2006" "Jan 2007" "May 2007" ...
 $ ORIG_END_DATE   : chr [1:426516] "Sep 2006" "Jan 2007" "May 2007" "Sep 2007" ...
 $ TASK_START_DATE : chr [1:426516] "Jun 2006" "Sep 2006" "Jan 2007" "May 2007" ...
 $ TASK_END_DATE   : chr [1:426516] "Sep 2006" "Jan 2007" "May 2007" "Sep 2007" ...
 - attr(*, "spec")=
  .. cols(
  ..   PUB_DATE = col_double(),
  ..   BORO = col_character(),
  ..   MANAGING_AGCY_CD = col_character(),
  ..   MANAGING_AGCY = col_character(),
  ..   PROJECT_ID = col_character(),
  ..   PROJECT_DESCR = col_character(),
  ..   SEQ_NUMBER = col_double(),
  ..   TASK_DESCRIPTION = col_character(),
  ..   ORIG_START_DATE = col_character(),
  ..   ORIG_END_DATE = col_character(),
  ..   TASK_START_DATE = col_character(),
  ..   TASK_END_DATE = col_character()
  .. )
 - attr(*, "problems")=<externalptr>
```

```
print(as.factor(dataset$BORO))
```

```
[1] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
   [9] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
  [17] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
  [25] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
  [33] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
  [41] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
  [49] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
  [57] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
  [65] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
```

```
 [73] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
 [81] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
 [89] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
 [97] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[105] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[113] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[121] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[129] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[137] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[145] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[153] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[161] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[169] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[177] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[185] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[193] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[201] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[209] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[217] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[225] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[233] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[241] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[249] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[257] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[265] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[273] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[281] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[289] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
```

[297] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[305] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[313] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[321] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[329] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[337] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[345] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[353] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[361] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[369] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[377] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[385] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[393] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[401] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[409] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[417] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[425] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[433] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[441] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[449] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[457] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[465] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[473] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[481] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[489] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[497] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[505] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[513] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE

```
[521] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[529] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[537] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[545] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[553] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[561] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[569] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[577] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[585] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[593] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[601] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[609] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[617] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[625] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[633] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[641] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[649] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[657] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[665] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[673] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[681] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[689] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[697] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[705] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[713] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[721] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[729] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[737] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
```

[745] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[753] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[761] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[769] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[777] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[785] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[793] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[801] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[809] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[817] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[825] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[833] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[841] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[849] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[857] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[865] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[873] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[881] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[889] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[897] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[905] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[913] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[921] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[929] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[937] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[945] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[953] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
[961] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE

```
 [969] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
 [977] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
 [985] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
 [993] CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE CITYWIDE
 [ reached getOption("max.print") -- omitted 425516 entries ]
Levels: BRONX BROOKLYN CITYWIDE MANHATTAN QUEENS RICHMOND
```

```
> class(dataset$TASK_START_DATE)
[1] "character"
> class(dataset$SEQ_NUMBER)
[1] "numeric"
```

```
questionfreq$questionfreq = as.numeric(questionfreq$questionfreq) questionfreq$questionfreq
```

```
## [1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 ## [26] 26 27 28 29 30 31 32 33 34
35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50
## [51] 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66
```

```
> table(dataset$SEQ_NUMBER)


    1     2     3     4     5     6     7     8     9    10    11    12    13 
62581 63534 62471 55394 52287 32974 23356 23350 23280 22148  1102  1011   973 
   14    15    16    17    18    19    20    21    22    23    24    25    26 
  823   815   124    77    58    57    57     6     6     6     6     6     3 
   27    28    29    30 
    3     3     3     2 
```

-

```
table(dataset$TASK_START_DATE)

Apr 1931 Apr 1932 Apr 1933 Apr 1995 Apr 1997 Apr 1999 Apr 2000 Apr 2001 Apr 2002
       9        4        1        6       15        1        9       16       12
Apr 2003 Apr 2004 Apr 2005 Apr 2006 Apr 2007 Apr 2008 Apr 2009 Apr 2010 Apr 2011
      18       21       16       11       37       36       40       28       57
Apr 2012 Apr 2013 Apr 2014 Apr 2015 Apr 2016 Apr 2017 Apr 2018 Apr 2019 Apr 2020
      50      134      139      172      270      306      212      402      479
Apr 2021 Apr 2022 Apr 2023 Apr 2024 Apr 2025 Apr 2026 Apr 2027 Apr 2028 Apr 2029
     447    10991      606      351      330      247      202       87       79
Apr 2030 Apr 2031 Apr 2032 Apr 2033 Apr 2039 Apr 2041 Aug 1932 Aug 1933 Aug 1935
      27        5        9        1        1        1       12        1        1
Aug 1940 Aug 1983 Aug 1992 Aug 1996 Aug 1997 Aug 1998 Aug 1999 Aug 2000 Aug 2001
       1        1        4        9       21       23       18       19        7
Aug 2002 Aug 2003 Aug 2004 Aug 2005 Aug 2006 Aug 2007 Aug 2008 Aug 2009 Aug 2010
       6       15       36       21       19       51       52      101       68
Aug 2011 Aug 2012 Aug 2013 Aug 2014 Aug 2015 Aug 2016 Aug 2017 Aug 2018 Aug 2019
      65       92      172      166      231      277      285      496      748
Aug 2020 Aug 2021 Aug 2022 Aug 2023 Aug 2024 Aug 2025 Aug 2026 Aug 2027 Aug 2028
     786      761    15896      713      572      294      322      195       78
```

```
Aug 2029 Aug 2030 Aug 2031 Aug 2032 Aug 2034 Aug 2035 Aug 2036 Aug 2039 Dec 1899 
      65       49       13       18        1        2        1        1      791 
Dec 1932 Dec 1934 Dec 1935 Dec 1992 Dec 1994 Dec 1995 Dec 1997 Dec 1998 Dec 1999 
       1        1        1        3        5        6       15        7        3 
Dec 2000 Dec 2001 Dec 2002 Dec 2003 Dec 2004 Dec 2005 Dec 2006 Dec 2007 Dec 2008 
       9        3       18       21       49       29      130       66       70 
Dec 2009 Dec 2010 Dec 2011 Dec 2012 Dec 2013 Dec 2014 Dec 2015 Dec 2016 Dec 2017 
     118      165      245      268      343      428      495      716     1315 
Dec 2018 Dec 2019 Dec 2020 Dec 2021 Dec 2022 Dec 2023 Dec 2024 Dec 2025 Dec 2026 
    1674     1725     1841     1848    33242     1158      663      399      323 
Dec 2027 Dec 2028 Dec 2029 Dec 2030 Dec 2031 Dec 2032 Dec 2035 Feb 1930 Feb 1931 
     214      130       61       73       22        7        4       48       11 
Feb 1932 Feb 1935 Feb 1936 Feb 1989 Feb 1991 Feb 1994 Feb 1998 Feb 1999 Feb 2000 
      13        1        2        4        7        7        6        8       29 
Feb 2001 Feb 2002 Feb 2003 Feb 2004 Feb 2005 Feb 2006 Feb 2007 Feb 2008 Feb 2009 
       1        1        9       10       14       18       45       46       37 
Feb 2010 Feb 2011 Feb 2012 Feb 2013 Feb 2014 Feb 2015 Feb 2016 Feb 2017 Feb 2018 
      46       45      121       99      161      193      333      344      419 
Feb 2019 Feb 2020 Feb 2021 Feb 2022 Feb 2023 Feb 2024 Feb 2025 Feb 2026 Feb 2027 
     676      715      892    18178     1196      735      594      428      344 
Feb 2028 Feb 2029 Feb 2030 Feb 2031 Feb 2032 Feb 2035 Feb 2036 Feb 2038 Feb 2040 
```

- 148 192 109 31 34 2 5 2 1

Jan 1932 Jan 1934 Jan 1935 Jan 1997 Jan 1998 Jan 1999 Jan 2000 Jan 2001 Jan 2002

7 1 1 11 10 1 52 8 13

Jan 2003 Jan 2004 Jan 2005 Jan 2006 Jan 2007 Jan 2008 Jan 2009 Jan 2010 Jan 2011

23 34 58 30 55 34 80 33 118

Jan 2012 Jan 2013 Jan 2014 Jan 2015 Jan 2016 Jan 2017 Jan 2018 Jan 2019 Jan 2020

129 136 226 292 496 430 537 760 998

Jan 2021 Jan 2022 Jan 2023 Jan 2024 Jan 2025 Jan 2026 Jan 2027 Jan 2028 Jan 2029

850 24172 1187 1252 919 647 413 299 191

Jan 2030 Jan 2031 Jan 2032 Jan 2033 Jan 2035 Jan 2037 Jan 2039 Jan 2041 Jul 1932

79 96 5 4 4 1 2 1 9

Jul 1933 Jul 1934 Jul 1937 Jul 1938 Jul 1981 Jul 1986 Jul 1992 Jul 1993 Jul 1995

20 4 3 3 1 2 3 8 6

Jul 1997 Jul 1998 Jul 1999 Jul 2000 Jul 2001 Jul 2002 Jul 2003 Jul 2004 Jul 2005

23 21 42 70 21 11 35 28 29

Jul 2006 Jul 2007 Jul 2008 Jul 2009 Jul 2010 Jul 2011 Jul 2012 Jul 2013 Jul 2014

179 134 104 97 170 217 148 392 395

Jul 2015 Jul 2016 Jul 2017 Jul 2018 Jul 2019 Jul 2020 Jul 2021 Jul 2022 Jul 2023

547 1070 1489 1459 1648 1647 1879 37301 1250

Jul 2024 Jul 2025 Jul 2026 Jul 2027 Jul 2028 Jul 2029 Jul 2030 Jul 2031 Jul 2032

900 674 403 278 196 98 56 51 10

```
Jul 2033 Jul 2034 Jul 2035 Jul 2036 Jul 2038 Jun 1931 Jun 1933 Jun 1934 Jun 1935
      27        4        2        2        1        3        1        2        2
Jun 1938 Jun 1990 Jun 1994 Jun 1997 Jun 1998 Jun 1999 Jun 2000 Jun 2001 Jun 2002
       1        7        4        1       34        3       19       22       11
Jun 2003 Jun 2004 Jun 2005 Jun 2006 Jun 2007 Jun 2008 Jun 2009 Jun 2010 Jun 2011
      11       27       33       42       56       66      103      163      193
Jun 2012 Jun 2013 Jun 2014 Jun 2015 Jun 2016 Jun 2017 Jun 2018 Jun 2019 Jun 2020
     228      346      388      422      650      766     1194     1586     1587
Jun 2021 Jun 2022 Jun 2023 Jun 2024 Jun 2025 Jun 2026 Jun 2027 Jun 2028 Jun 2029
    2285    39319     1896     1574      983      687      546      278      142
Jun 2030 Jun 2031 Jun 2032 Jun 2033 Jun 2034 Jun 2038 Jun 2039 Jun 2040 Jun 2041
      59       10        2        1        4        1        1        1        1
Mar 1932 Mar 1933 Mar 1934 Mar 1939 Mar 1993 Mar 1994 Mar 1996 Mar 1997 Mar 1998
       3        2        2        1        7       12        2        5        8
Mar 1999 Mar 2000 Mar 2001 Mar 2002 Mar 2003 Mar 2004 Mar 2005 Mar 2006 Mar 2007
       1       16        9        5       18       29       29       39      154
Mar 2008 Mar 2009 Mar 2010 Mar 2011 Mar 2012 Mar 2013 Mar 2014 Mar 2015 Mar 2016
     171      103      103      147      212      274      354      543      769
Mar 2017 Mar 2018 Mar 2019 Mar 2020 Mar 2021 Mar 2022 Mar 2023 Mar 2024 Mar 2025
     878      899      896      848      807    28045      918      931      658
Mar 2026 Mar 2027 Mar 2028 Mar 2029 Mar 2030 Mar 2031 Mar 2032 Mar 2033 Mar 2034
```

```
   -     354      233      226      110      109      41       13       7       4
May 1933 May 1935 May 1994 May 1996 May 1997 May 1998 May 1999 May 2000 May 2001
    2        1       12       13       26       10        1       25        7
May 2002 May 2003 May 2004 May 2005 May 2006 May 2007 May 2008 May 2009 May 2010
    6       10       46       20       30       74       53       72       77
May 2011 May 2012 May 2013 May 2014 May 2015 May 2016 May 2017 May 2018 May 2019
  111       96      190      221      326      389      510      397      519
May 2020 May 2021 May 2022 May 2023 May 2024 May 2025 May 2026 May 2027 May 2028
  618      732    17844     1070      728      531      352      224      170
May 2029 May 2030 May 2031 May 2033 May 2034 May 2035 May 2036 May 2037 May 2038
  108       50       27        9        2        6        1        2        1
May 2039 Nov 1931 Nov 1932 Nov 1988 Nov 1992 Nov 1997 Nov 1998 Nov 1999 Nov 2000
    1        2        2        1        4        8        9       30       18
Nov 2001 Nov 2002 Nov 2003 Nov 2004 Nov 2005 Nov 2006 Nov 2007 Nov 2008 Nov 2009
    4        1       10       12       10        7       56       39       50
Nov 2010 Nov 2011 Nov 2012 Nov 2013 Nov 2014 Nov 2015 Nov 2016 Nov 2017 Nov 2018
   37       31       89      135      114      170      220      340      340
Nov 2019 Nov 2020 Nov 2021 Nov 2022 Nov 2023 Nov 2024 Nov 2025 Nov 2026 Nov 2027
  491      436      344     9877      282      233      128       59       27
Nov 2028 Nov 2029 Nov 2030 Nov 2031 Nov 2032 Nov 2037 Nov 2039 Oct 1992 Oct 1993
   62       28       27        5        7        1        1        1        8
```

```
Oct 1994 Oct 1995 Oct 1996 Oct 1997 Oct 1998 Oct 1999 Oct 2000 Oct 2001 Oct 2002
       4        1       12       14        5        5       23        9        9
Oct 2003 Oct 2004 Oct 2005 Oct 2006 Oct 2007 Oct 2008 Oct 2009 Oct 2010 Oct 2011
      38       11       34       52       51       48       61       58       58
Oct 2012 Oct 2013 Oct 2014 Oct 2015 Oct 2016 Oct 2017 Oct 2018 Oct 2019 Oct 2020
      53      110      218      397      371      390      711      652      756
Oct 2021 Oct 2022 Oct 2023 Oct 2024 Oct 2025 Oct 2026 Oct 2027 Oct 2028 Oct 2029
    1267    21085     1223      825      586      441      287      131       90
Oct 2030 Oct 2031 Oct 2033 Oct 2035 Oct 2036 Oct 2038 Oct 2040 Sep 1931 Sep 1934
      61        5        2        1        1        2        1        7        1
Sep 1935 Sep 1986 Sep 1992 Sep 1998 Sep 1999 Sep 2000 Sep 2001 Sep 2002 Sep 2003
       1        1        3        9        7       18        6        3       28
Sep 2004 Sep 2005 Sep 2006 Sep 2007 Sep 2008 Sep 2009 Sep 2010 Sep 2011 Sep 2012
      23       37      139       92      122      105      220      262      210
Sep 2013 Sep 2014 Sep 2015 Sep 2016 Sep 2017 Sep 2018 Sep 2019 Sep 2020 Sep 2021
     365      486      619      665      851      965     2058     1956     2506
Sep 2022 Sep 2023 Sep 2024 Sep 2025 Sep 2026 Sep 2027 Sep 2028 Sep 2029 Sep 2030
   41860     1748     1728     1089      557      389      237       95       36
Sep 2031 Sep 2032 Sep 2034 Sep 2035
      10        3        4        4
```

```
questionfreq = as.data.frame(questionfreq) head(questionfreq)
```

```
##   -   questionfreq Freq
        ## 1        1          1
## 2     2          1 ## 3    3
         1 ## 4     4          1
## 5     5          1 ## 6    6
         1
```

In the above code , we converted the *qid* column to a numeric column since graphs can only be plotted when the data is not categorical. In other words, we have to convert the categorical data into non-categorical data to visualize the plots without any error.

To make the data frame simpler to understand , we have renamed the *questionfreq* column to *qid* in the **questionfreq** dataframe that we extracted from the dataset.

```
questionfreq = questionfreq %>% rename(qid = questionfreq)
head(questionfreq)
```

```
##       qid Freq
## 1      1          1
## 2      2          1
## 3      3          1
## 4      4          1
## 5      5          1
## 6      6          1
```

We have finished converting the qid column and generated the frequency of each question-id in the survey. We will next examine the structure of the dataset again and convert the remaining columns into numeric data.

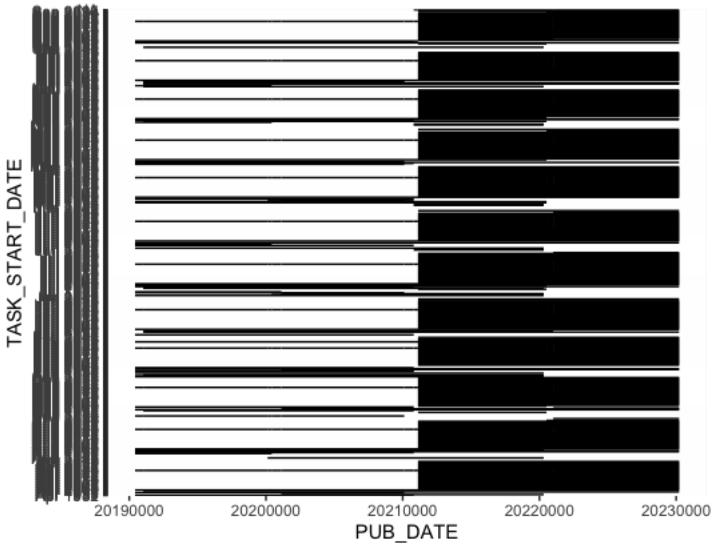
```
str(dataset)

spc_tbl_ [426,516 × 12] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
```

```
 $ PUB_DATE        : num [1:426516] 20220517 20220517 20220517 20220517 20220517 ...
 $ BORO            : chr [1:426516] "CITYWIDE" "CITYWIDE" "CITYWIDE" "CITYWIDE" ...
 $ MANAGING_AGCY_CD: chr [1:426516] "042" "042" "042" "042" ...
 $ MANAGING_AGCY   : chr [1:426516] "CITY UNIVERSITY" "CITY UNIVERSITY" "CITY UNIVERSITY" "CITY UNIVERSITY"
...
 $ PROJECT_ID      : chr [1:426516] "CA202-006" "CA202-006" "CA202-006" "CA202-006" ...
 $ PROJECT_DESCR   : chr [1:426516] "ADA Compliance" "ADA Compliance" "ADA Compliance" "ADA Compliance" ...
 $ SEQ_NUMBER      : num [1:426516] 4 5 6 7 8 9 10 1 2 3 ...
 $ TASK_DESCRIPTION: chr [1:426516] "BID AWARD AND REGISTER CONTRCT" "CONSTRUCTION TO 25%"
"CONSTRUCTION TO 50%" "CONSTRUCTION TO 75%" ...
 $ ORIG_START_DATE : chr [1:426516] "Jun 2006" "Sep 2006" "Jan 2007" "May 2007" ...
 $ ORIG_END_DATE   : chr [1:426516] "Sep 2006" "Jan 2007" "May 2007" "Sep 2007" ...
 $ TASK_START_DATE : chr [1:426516] "Jun 2006" "Sep 2006" "Jan 2007" "May 2007" ...
 $ TASK_END_DATE   : chr [1:426516] "Sep 2006" "Jan 2007" "May 2007" "Sep 2007" ...
 - attr(*, "spec")=
  .. cols(
  ..   PUB_DATE = col_double(),
  ..   BORO = col_character(),
  ..   MANAGING_AGCY_CD = col_character(),

  ..   MANAGING_AGCY = col_character(),

  ..   PROJECT_ID = col_character(),

  ..   PROJECT_DESCR = col_character(),

  ..   SEQ_NUMBER = col_double(),

  ..   TASK_DESCRIPTION = col_character(),

  ..   ORIG_START_DATE = col_character(),

  ..   ORIG_END_DATE = col_character(),

  ..   TASK_START_DATE = col_character(),

  ..   TASK_END_DATE = col_character()

  .. )

 - attr(*, "problems")=<externalptr>
```

Line Graphs

Since both the columns in the dataframe are converted into numeric values , we can now plot a line graph to visualize the data.

```
ggplot(dataset , aes(x= PUB_DATE , y = TASK_START_DATE)) + geom_line()
```



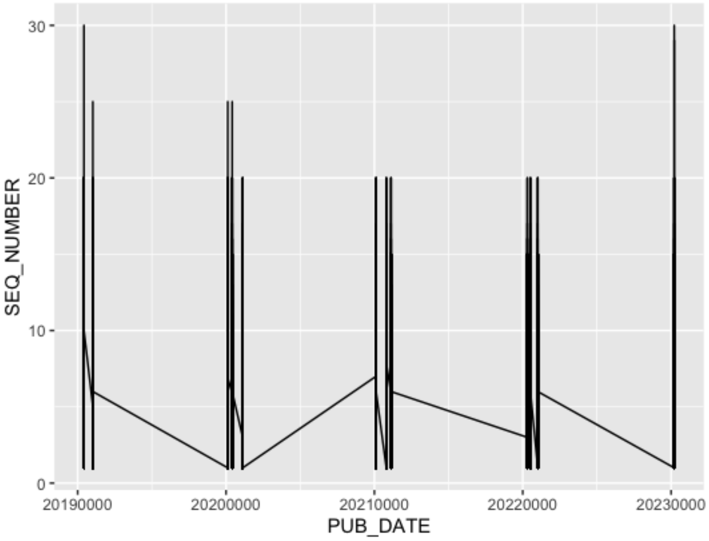
The above output depicts a line graph plotted between *qid* and *Freq* in the **questionfreq** dataframe.

```
typefreq = typefreq %>% rename( type = typefreq ) typefreq$type =
as.numeric(typefreq$type) print(typefreq)
```

```
##        type Freq
## 1 1 8 ## 2 2 14
## 3 3 54 ## 4 4 1
##   5   5   1   ##   6
            6
            1
```

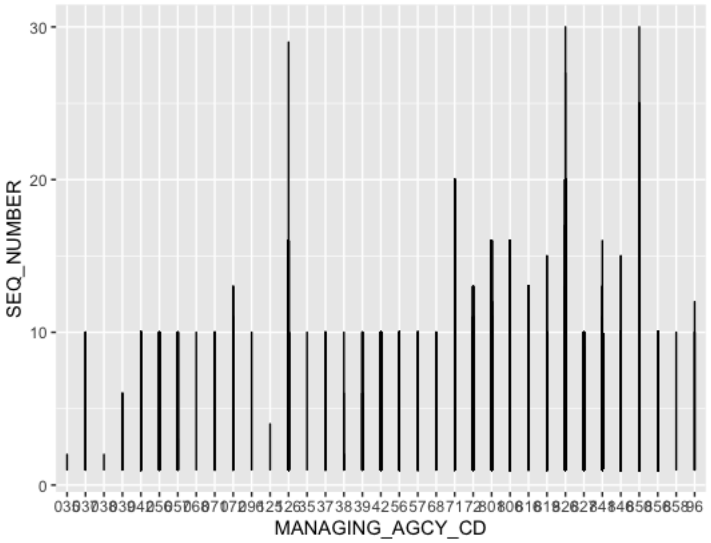
- We have converted the **typefreq** dataframe to numeric data . Below is the plot between *type* and *freq*

ggplot(dataset , aes(x= PUB_DATE , y = SEQ_NUMBER)) + geom_line()



Below is the code for *freq* as X axis and *type* as Y axis .

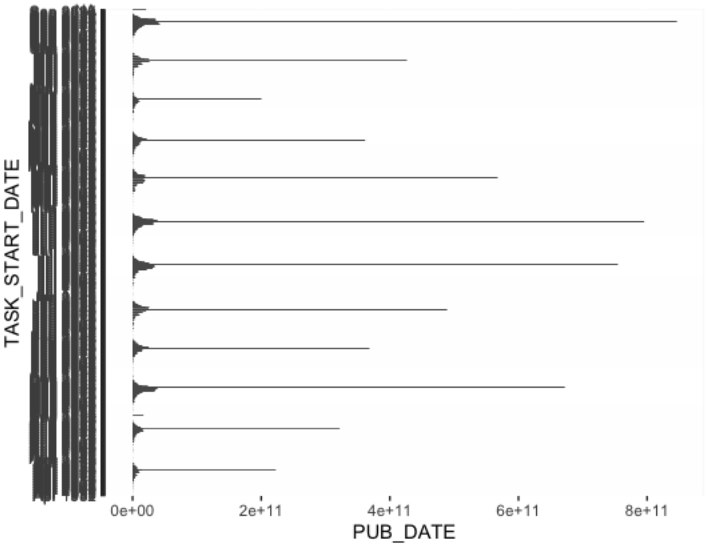
ggplot(dataset , aes(x= MANAGING_AGCY_CD, y = SEQ_NUMBER)) + geom_line()



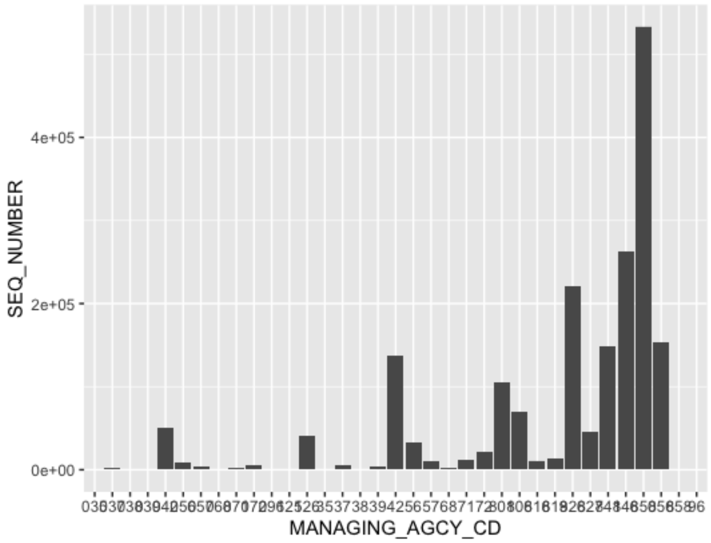
Bar Graphs

```
ggplot(dataset , aes(x= PUB_DATE , y =
TASK_START_DATE)) + geom_col()
```

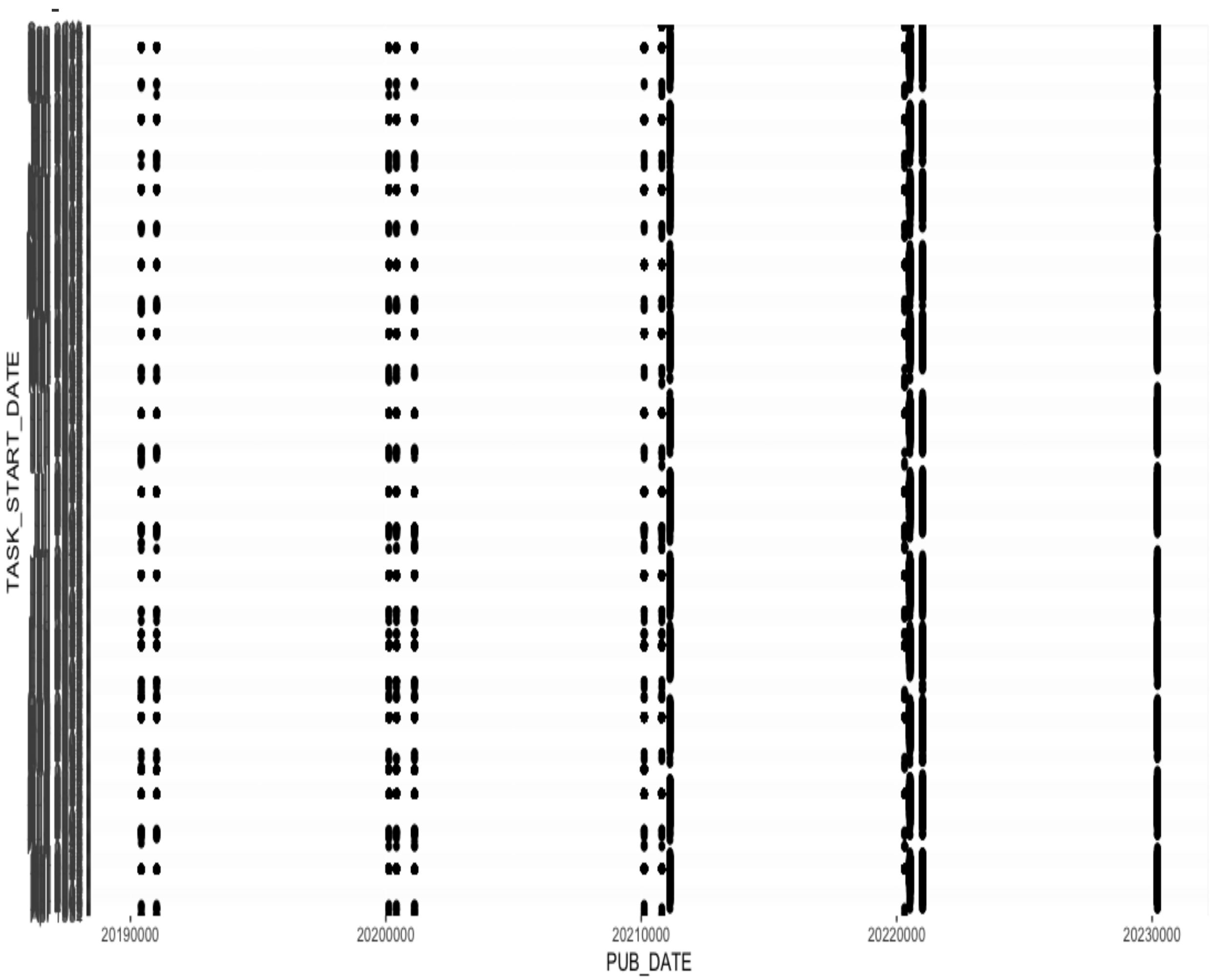
-



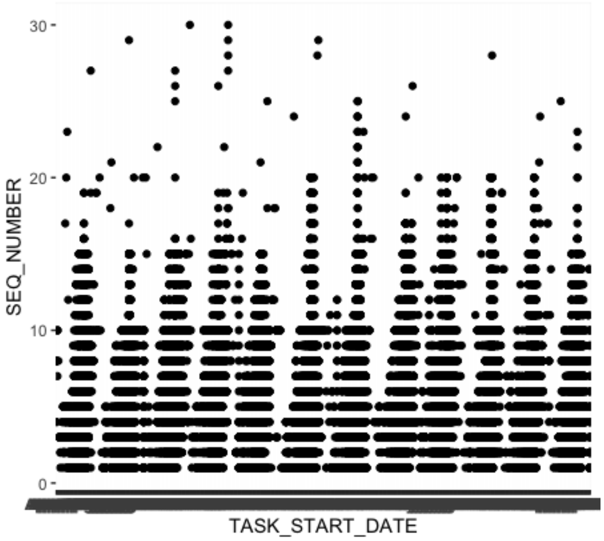
```
ggplot(dataset , aes(x= MANAGING_AGCY_CD , y =SEQ_NUMBER )) + geom_col()
```



```
ggplot(dataset , aes(x= PUB_DATE , y = TASK_START_DATE)) + geom_point()
```


TASK_START_DATE
20190000
20200000
20210000
20220000
20230000
PUB_DATE

```
ggplot(dataset , aes(x= TASK_START_DATE , y = SEQ_NUMBER)) + geom_point()
```


SEQ_NUMBER
TASK_START_DATE