Storm Effects on Communities, Analysis

Anandu R

8/6/2020

Storms and other severe weather events can cause both public health and economic problems for communities and municipalities. Many severe events can result in fatalities, injuries, and property damage, and preventing such outcomes to the extent possible is a key concern.

This project involves exploring the U.S. National Oceanic and Atmospheric Administration's (NOAA) storm database. This database tracks characteristics of major storms and weather events in the United States, including when and where they occur, as well as estimates of any fatalities, injuries, and property damage.

1. Data Processing

There is also some documentation of the database available. Details on how some of the variables are constructed/defined is available on this website by National Weather Service : Storm Data Documentation

1.1 Getting the data

```
fileUrl = "https://d396qusza40orc.cloudfront.net/repdata%2Fdata%2FStormData.csv.bz2"
if(!file.exists("./data/data.csv.bz2")){
  download.file(fileUrl,"./data/data.csv.bz2")
}
## importing libraries
suppressMessages(
  {
    library(dplyr)
    library(ggplot2)
    library(reshape2)
  }
)
```

1.2 Reading the data

```
suppressMessages(library(dplyr))
data_raw <- read.csv("./data/data.csv.bz2", sep =",", header = T)
```

1.3 Preliminary analysis of data

```r
head(data_raw)
```

```
##   STATE__           BGN_DATE BGN_TIME TIME_ZONE COUNTY COUNTYNAME STATE  EVTYPE
## 1       1  4/18/1950 0:00:00     0130       CST     97     MOBILE    AL TORNADO
## 2       1  4/18/1950 0:00:00     0145       CST      3    BALDWIN    AL TORNADO
## 3       1  2/20/1951 0:00:00     1600       CST     57    FAYETTE    AL TORNADO
## 4       1   6/8/1951 0:00:00     0900       CST     89    MADISON    AL TORNADO
## 5       1 11/15/1951 0:00:00     1500       CST     43    CULLMAN    AL TORNADO
## 6       1 11/15/1951 0:00:00     2000       CST     77 LAUDERDALE    AL TORNADO
##   BGN_RANGE BGN_AZI BGN_LOCATI END_DATE END_TIME COUNTY_END COUNTYENDN
## 1         0                                               0         NA
## 2         0                                               0         NA
## 3         0                                               0         NA
## 4         0                                               0         NA
## 5         0                                               0         NA
## 6         0                                               0         NA
##   END_RANGE END_AZI END_LOCATI LENGTH WIDTH F MAG FATALITIES INJURIES PROPDMG
## 1         0                      14.0   100 3   0          0       15    25.0
## 2         0                       2.0   150 2   0          0        0     2.5
## 3         0                       0.1   123 2   0          0        2    25.0
## 4         0                       0.0   100 2   0          0        2     2.5
## 5         0                       0.0   150 2   0          0        2     2.5
## 6         0                       1.5   177 2   0          0        6     2.5
##   PROPDMGEXP CROPDMG CROPDMGEXP WFO STATEOFFIC ZONENAMES LATITUDE LONGITUDE
## 1          K       0                                         3040      8812
## 2          K       0                                         3042      8755
## 3          K       0                                         3340      8742
## 4          K       0                                         3458      8626
## 5          K       0                                         3412      8642
## 6          K       0                                         3450      8748
##   LATITUDE_E LONGITUDE_ REMARKS REFNUM
## 1       3051       8806              1
## 2          0          0              2
## 3          0          0              3
## 4          0          0              4
## 5          0          0              5
## 6          0          0              6
```

1.3.1 Reading column names

```r
names(data_raw)
```

```
##  [1] "STATE__"    "BGN_DATE"   "BGN_TIME"   "TIME_ZONE"  "COUNTY"
##  [6] "COUNTYNAME" "STATE"      "EVTYPE"     "BGN_RANGE"  "BGN_AZI"
## [11] "BGN_LOCATI" "END_DATE"   "END_TIME"   "COUNTY_END" "COUNTYENDN"
## [16] "END_RANGE"  "END_AZI"    "END_LOCATI" "LENGTH"     "WIDTH"
## [21] "F"          "MAG"        "FATALITIES" "INJURIES"   "PROPDMG"
## [26] "PROPDMGEXP" "CROPDMG"    "CROPDMGEXP" "WFO"        "STATEOFFIC"
## [31] "ZONENAMES"  "LATITUDE"   "LONGITUDE"  "LATITUDE_E" "LONGITUDE_"
## [36] "REMARKS"    "REFNUM"
```

1.4 Data Cleaning

1.4.1 Removing unnecessary variables/Subsetting the data

Since the END_DATE and END_TIME fields are same as the BGN_DATA and BGN_TIME, we also remove those columns from the data.

Furthermore, since the COUNTY_END field has only the value 0 and would serve no purpose to the analysis, it too is removed

The "REFNUM" and "REMARKS" fields don't serve any purpose to our analysis

```
data_clean = select(data_raw,
                    STATE,
                    COUNTY,
                    BGN_DATE,
                    BGN_TIME,
                    EVTYPE,
                    FATALITIES,
                    INJURIES,
                    PROPDMG,
                    PROPDMGEXP,
                    CROPDMG,
                    CROPDMGEXP)
```

1.4.2 Missing data treatment

```
as.numeric(colMeans(is.na(data_clean)))
```

1.4.2.1 Checking distribution of Missing data and NAs in the dataset

```
##  [1] 0 0 0 0 0 0 0 0 0 0 0
```

```
as.numeric(colMeans(data_clean==""))
```

```
##  [1] 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
##  [8] 0.0000000 0.5163865 0.0000000 0.6853763
```

Columns 9, and 11 represent the "PROPDMGEXP", "CROPDMGEXP" fields which are required for the analysis therefore we will keep them.

Therefore all in all, there arent any records to be removed or are their any columns that can be removed.

NOTE: During analysis there may still be some fields with no value aka missing values in certain columns, but their percentages are in range 10-50% so the next suitable step would be to impute the values in the dataset, but since it is the weather data, imputing values would only create noise in the data(?)

Looking at cleaned data

```
head(data_clean)
```

```
##   STATE COUNTY           BGN_DATE BGN_TIME  EVTYPE FATALITIES INJURIES PROPDMG
## 1    AL     97  4/18/1950 0:00:00     0130 TORNADO          0       15    25.0
## 2    AL      3  4/18/1950 0:00:00     0145 TORNADO          0        0     2.5
## 3    AL     57  2/20/1951 0:00:00     1600 TORNADO          0        2    25.0
## 4    AL     89   6/8/1951 0:00:00     0900 TORNADO          0        2     2.5
## 5    AL     43 11/15/1951 0:00:00     1500 TORNADO          0        2     2.5
## 6    AL     77 11/15/1951 0:00:00     2000 TORNADO          0        6     2.5
##   PROPDMGEXP CROPDMG CROPDMGEXP
## 1          K       0
## 2          K       0
## 3          K       0
## 4          K       0
## 5          K       0
## 6          K       0
```

1.4.3 Fixing the datatypes and datafields

```
data_clean$BGN_DATE =
  as.POSIXct(data_clean$BGN_DATE, format = "%m/%d/%Y %H:%M:%S")

data_clean$BGN_TIME =
  format(strptime(data_clean$BGN_TIME,"%H%M"),'%H:%M')

data_clean$BGN_DATETIME =
  as.POSIXct(paste(data_clean$BGN_DATE,
                   data_clean$BGN_TIME
                   ), format="%Y-%m-%d %H:%M")
data_clean =
  select(data_clean,
         STATE, COUNTY,
         BGN_DATETIME,
         EVTYPE, FATALITIES,
         INJURIES,
         PROPDMG,
         PROPDMGEXP,
         CROPDMG,
         CROPDMGEXP)
```

1.4.3.1 Creating a datatime field

1.4.3.2 Imputing proper values in the "PROPDMGEXP", "CROPDMGEXP" fields

Current values in "CROPDMGEXP"

```
unique(data_clean$CROPDMGEXP)
```

```
## [1] ""  "M" "K" "m" "B" "?" "0" "k" "2"
```

Current values in "PROPDMGEXP"

```r
unique(data_clean$PROPDMGEXP)
```

```
##  [1] "K" "M" ""  "B" "m" "+" "0" "5" "6" "?" "4" "2" "3" "h" "7" "H" "-" "1" "8"
```

Correct representations:
- """" = 10^0,
- "-" = 10^0,
- "?" = 10^0,
- "+" = 10^0,
- "0" = 10^0,
- "1" = 10^1,
- "2" = 10^2,
- "3" = 10^3,
- "4" = 10^4,
- "5" = 10^5,
- "6" = 10^6,
- "7" = 10^7,
- "8" = 10^8,
- "9" = 10^9,
- "H" = 10^2,
- "K" = 10^3,
- "M" = 10^6,
- "B" = 10^9

Imputing the correct values

```r
data_clean = transform(data_clean,
                       PROPDMGEXP = toupper(PROPDMGEXP),
                       CROPDMGEXP = toupper(CROPDMGEXP))
DmgExP =  c("\"\"" = 10^0,
            "-" = 10^0,
            "+" = 10^0,
            "?" = 10^0,
            "0" = 10^0,
            "1" = 10^1,
            "2" = 10^2,
            "3" = 10^3,
            "4" = 10^4,
            "5" = 10^5,
            "6" = 10^6,
            "7" = 10^7,
            "8" = 10^8,
            "9" = 10^9,
            "H" = 10^2,
            "K" = 10^3,
            "M" = 10^6,
            "B" = 10^9)
data_clean = transform(
  data_clean,
  PROPDMGEXP = as.numeric(DmgExP[as.character(data_clean[,"PROPDMGEXP"])]),
  CROPDMGEXP = as.numeric(DmgExP[as.character(data_clean[,"CROPDMGEXP"])])
)
data_clean = transform(
```

```r
  data_clean,
  PROPDMGEXP = ifelse(is.na(PROPDMGEXP),10^0,PROPDMGEXP),
  CROPDMGEXP = ifelse(is.na(CROPDMGEXP),10^0,CROPDMGEXP)
)
```

1.4.3.3 Subsetting the data, removing EVTYPEs that have 0 impact of any sort

```r
data_clean = subset(data_clean,
                    EVTYPE != "?" &
                      (INJURIES > 0 |
                         FATALITIES > 0 |
                         PROPDMG > 0 |
                         CROPDMG > 0)
                    )
```

Looking at cleaned data

```r
head(data_clean)
```

```
##   STATE COUNTY         BGN_DATETIME  EVTYPE FATALITIES INJURIES PROPDMG
## 1    AL     97 1950-04-18 01:30:00 TORNADO          0       15    25.0
## 2    AL      3 1950-04-18 01:45:00 TORNADO          0        0     2.5
## 3    AL     57 1951-02-20 16:00:00 TORNADO          0        2    25.0
## 4    AL     89 1951-06-08 09:00:00 TORNADO          0        2     2.5
## 5    AL     43 1951-11-15 15:00:00 TORNADO          0        2     2.5
## 6    AL     77 1951-11-15 20:00:00 TORNADO          0        6     2.5
##   PROPDMGEXP CROPDMG CROPDMGEXP
## 1       1000       0          1
## 2       1000       0          1
## 3       1000       0          1
## 4       1000       0          1
## 5       1000       0          1
## 6       1000       0          1
```

1.4.4 Standardising data in the '"EVTYPE" field

The various fields in EVTYPES have been misspelled or two names that represents the same event have been used therefore all of the event types have been standardized

Since the code for this is very long it has been hidden from view, if you wish to take a look at the cook please look into the Analysis.Rmd file in the repo

```r
unique(data_clean$EVTYPE)
```

```
##  [1] "TORNADO"                "THUNDERSTORM"           "HAIL"
##  [4] "FLASH FLOOD"            "BLIZZARD"               "HURRICANE"
##  [7] "RAINFALL"               "LIGHTNING"              "DENSE FOG"
## [10] "RIP CURRENT"            "HEAT+DROUGHT"           "WIND"
## [13] "FROST+SNOW"             "FLOOD"                  "WATERSPOUT+TORNADO"
## [16] "RURAL FLOOD"            "AVALANCHE"              "MARINE ACCIDENT"
```

```
## [19] "TIDE"                 "TIDE/ROUGH SEAS"       "COASTAL FLOOD+EROSION"
## [22] "SEVERE TURBULENCE"    "DUST"                  "SURF"
## [25] "WILDFIRE"             "MUD+LAND SLIDES"       "URBAN FLOOD"
## [28] "STORM SURGE"          "TROPICAL CYCLONE"      "WETNESS"
## [31] "FOG"                  "ICY ROADS"             "HEAVY MIX"
## [34] "HIGH WAVES"           "HYPOTHERMIA"           "HEAVY SEAS"
## [37] "OTHER"                "COASTAL STORM"         "DAM BREAK"
## [40] "TYPHOON"              "HIGH SWELLS"           "HYPERTHERMIA"
## [43] "ROUGH SEAS"           "ROGUE WAVE"            "DROWNING"
## [46] "TSUNAMI"
```

2. Exploratory Analysis

2.1 Creating new fields CROPDMGPRICE and PROPDMGPRICE

```
data_clean = transform(data_clean,
                       CROPDMGPRICE = CROPDMG*CROPDMGEXP,
                       PROPDMGPRICE = PROPDMG*PROPDMGEXP)
```

2.2 Aggregating the data based on event type

```
## Creating a 'wide' aggregation of data
suppressMessages(
  {
  data_aggr_w = data_clean %>%
    group_by(EVTYPE) %>%
    summarise(
      FATALITIES = sum(FATALITIES, na.rm = T),
      INJURIES = sum(INJURIES,na.rm = T),
      CROPDMGPRICE = sum(CROPDMGPRICE, na.rm = T),
      PROPDMGPRICE = sum(PROPDMGPRICE, na.rm = T)
    )
  }
)
head(data_aggr_w[order(-data_aggr_w[,"FATALITIES"],
                   -data_aggr_w[,"INJURIES"],
                   -data_aggr_w[,"CROPDMGPRICE"],
                   -data_aggr_w[,"PROPDMGPRICE"]),])
```

```
## # A tibble: 6 x 5
##   EVTYPE       FATALITIES INJURIES CROPDMGPRICE PROPDMGPRICE
##   <chr>             <dbl>    <dbl>        <dbl>        <dbl>
## 1 TORNADO            5633    91367    414961520 56952347026.
## 2 HEAT+DROUGHT       3178     9247  14877045280   1066431750
## 3 FLASH FLOOD        1035     1802   1532197150 17589261096.
## 4 LIGHTNING           817     5231     12092090   930419430.
## 5 THUNDERSTORM        755     9543   1274213988 12785456700.
## 6 FROST+SNOW          659     1986   3565490400   1315567650
```

```r
data_aggr_w = transform(data_aggr_w,
                        TOTPUBDMG = FATALITIES + INJURIES,
                        TOTECODMG = CROPDMGPRICE + PROPDMGPRICE)

## Splitting the public damage and economy damage data
data_aggr_wp = data_aggr_w[order(-data_aggr_w$TOTPUBDMG),c(1,2,3,6)]
data_aggr_we = data_aggr_w[order(-data_aggr_w$TOTECODMG),c(1,4,5,7)]

## Selecting only the top 10 most devastating events for each category
data_aggr_wp = data_aggr_wp[1:10,]
data_aggr_we = data_aggr_we[1:10,]

## Creating a 'narrow' aggregation of data
suppressMessages(
  {
  data_aggr_np = melt(
    data_aggr_wp,
    id.vars = c("EVTYPE"),
    measure.vars = c("FATALITIES","INJURIES","TOTPUBDMG"),
    variable.name = "ATTRIBUTE",
    value.name = "MEASURE")
  data_aggr_ne = melt(
    data_aggr_we,
    id.vars = c("EVTYPE"),
    measure.vars = c("CROPDMGPRICE","PROPDMGPRICE","TOTECODMG"),
    variable.name = "ATTRIBUTE",
    value.name = "MEASURE")
  }
)
```

```r
nrow(data_aggr_w)
```

```
## [1] 46
```

There are 46 rows of data available on various events, which we'll use to create various plots to show which types of events across the United States are most harmful with respect to population health and have the greatest economic consequences.

2.3 Analyis to find events most harmful with respect to population health

Looking at data in relevant columns "FATALITIES" and "INJURIES" sorted by descending order of the field values

```r
head(data_aggr_w[order(-data_aggr_w[,"FATALITIES"],
                       -data_aggr_w[,"INJURIES"]),c(1,2,3)])
```

```
##          EVTYPE FATALITIES INJURIES
## 38      TORNADO       5633    91367
## 14 HEAT+DROUGHT       3178     9247
## 9   FLASH FLOOD       1035     1802
## 23    LIGHTNING        817     5231
## 35 THUNDERSTORM        755     9543
## 12   FROST+SNOW        659     1986
```

2.4 Analyis to find events most harmful with respect to economic damage

Looking at data in relevant columns "CROPDMGPRICE" and "PROPDMGPRICE" sorted by descending order of the field values

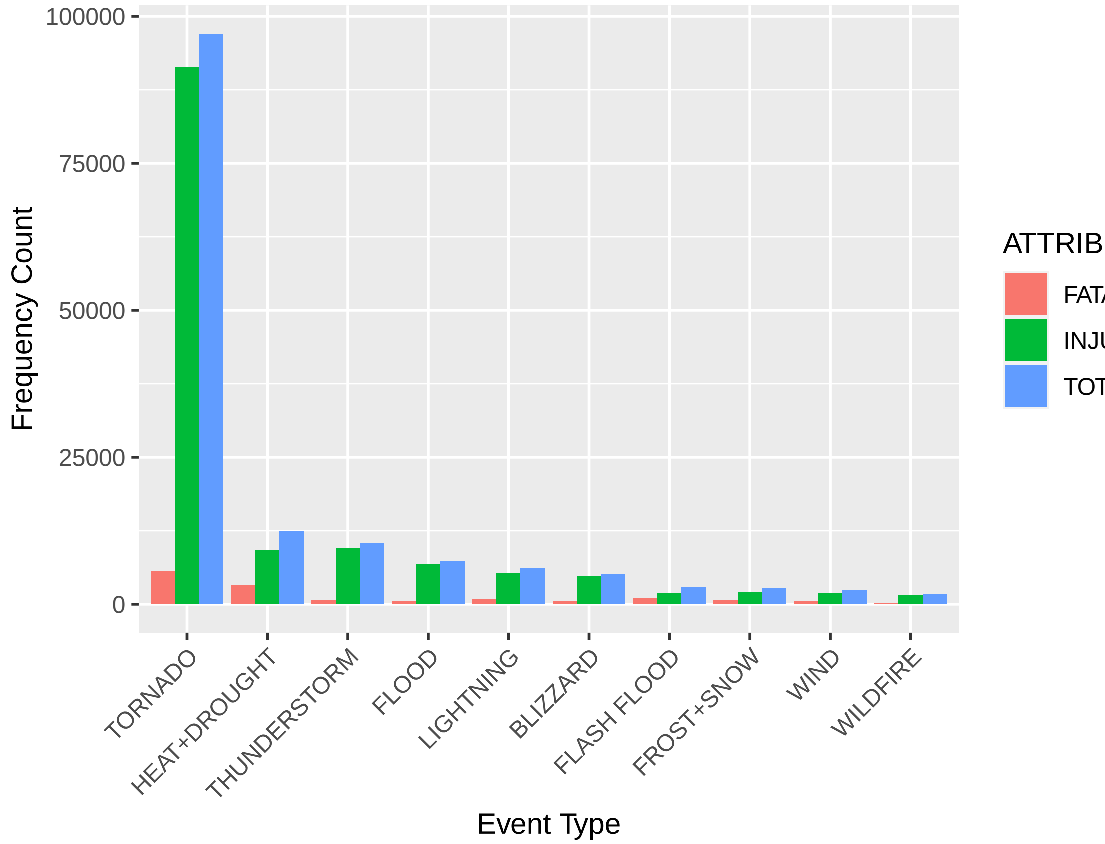
```
head(data_aggr_w[order(-data_aggr_w[,"CROPDMGPRICE"],
                       -data_aggr_w[,"PROPDMGPRICE"]),c(1,4,5)])
```

```
##          EVTYPE CROPDMGPRICE PROPDMGPRICE
## 14 HEAT+DROUGHT  14877045280   1066431750
## 10        FLOOD   5817438450 144922006929
## 19    HURRICANE   5515292800  84756180010
## 2      BLIZZARD   5181617500  11381587061
## 31  RURAL FLOOD   5029464000   5128216700
## 12   FROST+SNOW   3565490400   1315567650
```

3. RESULTS

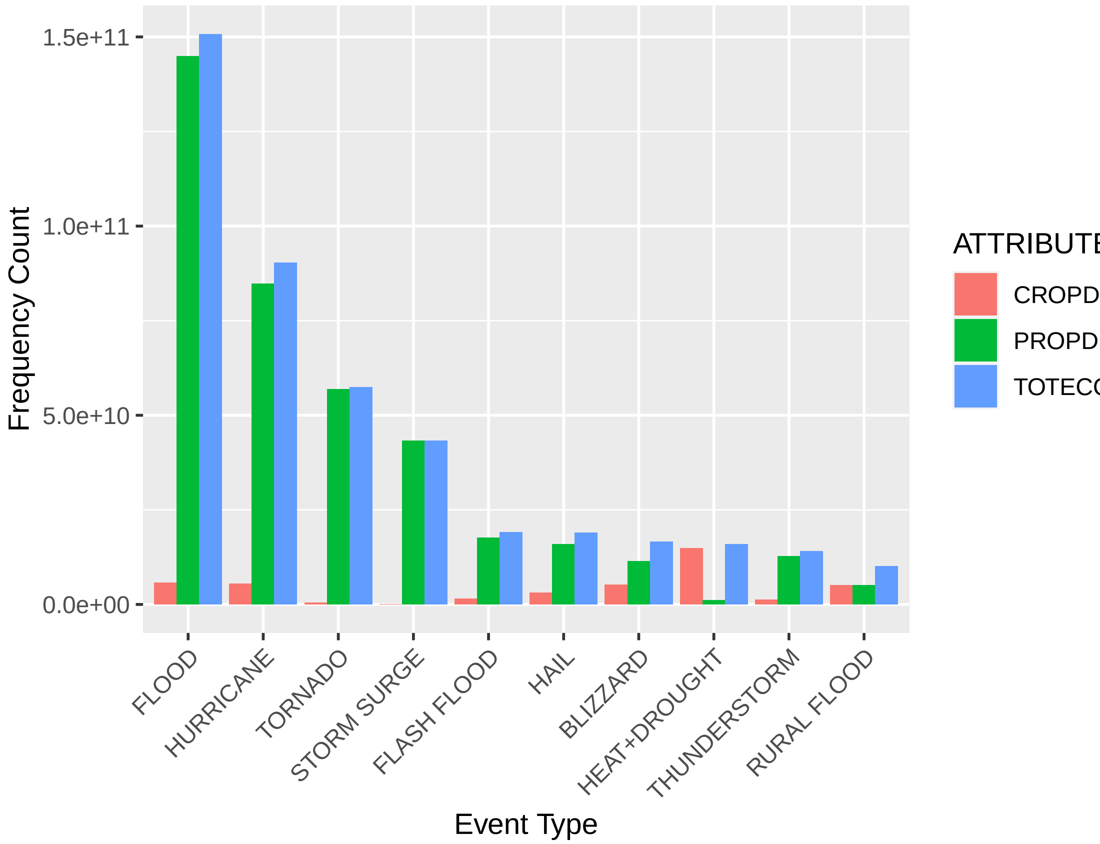
3.1 Visualization

```
ggplot(
  data_aggr_np,
  aes(
    x = reorder(EVTYPE, -MEASURE),
    y = MEASURE
  ),
) + geom_bar(stat="identity", aes(fill=ATTRIBUTE), position="dodge") +
  theme(axis.text.x = element_text(angle=45, hjust=1)) + guides() +
  xlab("Event Type") +
  ylab("Frequency Count")
```



3.1.1 Population health

```
ggplot(
  data_aggr_ne,
  aes(
    x = reorder(EVTYPE, -MEASURE),
    y = MEASURE
  ),
) + geom_bar(stat="identity", aes(fill=ATTRIBUTE), position="dodge") +
  theme(axis.text.x = element_text(angle=45, hjust=1)) + guides() +
  xlab("Event Type") +
  ylab("Frequency Count")
```



3.1.2 Economic damage

Removing data file after analysis

```
unlink("./data/data.csv.bz2",recursive = T)
#unlink("./analysis_cache", recursive = T)
```