Assessment of economic and public health consequences of adverse weather events in the United States

J. Varberg

2/26/2022

Synopsis

This report uses data from the National Oceanic and Atmospheric Administration (NOAA) Storm Database to examine the economic and publich health impacts that different types of weather events have in the United States.

Data Processing

The raw data were obtained here and read into R using the `read_csv` function from the `readr` package, which can directly handle reading/import of zipped files.

```
rawData <- read_csv("./data/repdata-data-StormData.csv.bz2")
glimpse(rawData)
```

```
## Rows: 902,297
## Columns: 37
## $ STATE__    <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,~
## $ BGN_DATE   <chr> "4/18/1950 0:00:00", "4/18/1950 0:00:00", "2/20/1951 0:00:0~
## $ BGN_TIME   <chr> "0130", "0145", "1600", "0900", "1500", "2000", "0100", "09~
## $ TIME_ZONE  <chr> "CST", "CST", "CST", "CST", "CST", "CST", "CST", "CST", "CS~
## $ COUNTY     <dbl> 97, 3, 57, 89, 43, 77, 9, 123, 125, 57, 43, 9, 73, 49, 107,~
## $ COUNTYNAME <chr> "MOBILE", "BALDWIN", "FAYETTE", "MADISON", "CULLMAN", "LAUD~
## $ STATE      <chr> "AL", "AL", "AL", "AL", "AL", "AL", "AL", "AL", "AL", "AL",~
## $ EVTYPE     <chr> "TORNADO", "TORNADO", "TORNADO", "TORNADO", "TORNADO", "TOR~
## $ BGN_RANGE  <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,~
## $ BGN_AZI    <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ BGN_LOCATI <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ END_DATE   <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ END_TIME   <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ COUNTY_END <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,~
## $ COUNTYENDN <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ END_RANGE  <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,~
## $ END_AZI    <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ END_LOCATI <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ LENGTH     <dbl> 14.0, 2.0, 0.1, 0.0, 0.0, 1.5, 1.5, 0.0, 3.3, 2.3, 1.3, 4.7~
## $ WIDTH      <dbl> 100, 150, 123, 100, 150, 177, 33, 33, 100, 100, 400, 400, 2~
## $ F          <dbl> 3, 2, 2, 2, 2, 2, 2, 1, 3, 3, 1, 1, 3, 3, 3, 4, 1, 1, 1, 1,~
## $ MAG        <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,~
```

```
## $ FATALITIES <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 4, 0, 0, 0, 0,~
## $ INJURIES   <dbl> 15, 0, 2, 2, 2, 6, 1, 0, 14, 0, 3, 3, 26, 12, 6, 50, 2, 0, ~
## $ PROPDMG    <dbl> 25.0, 2.5, 25.0, 2.5, 2.5, 2.5, 2.5, 2.5, 25.0, 25.0, 2.5, ~
## $ PROPDMGEXP <chr> "K", "K", "K", "K", "K", "K", "K", "K", "K", "K", "M", "M",~
## $ CROPDMG    <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,~
## $ CROPDMGEXP <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ WFO        <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ STATEOFFIC <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ ZONENAMES  <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ LATITUDE   <dbl> 3040, 3042, 3340, 3458, 3412, 3450, 3405, 3255, 3334, 3336,~
## $ LONGITUDE  <dbl> 8812, 8755, 8742, 8626, 8642, 8748, 8631, 8558, 8740, 8738,~
## $ LATITUDE_E <dbl> 3051, 0, 0, 0, 0, 0, 0, 0, 3336, 3337, 3402, 3404, 0, 3432,~
## $ LONGITUDE_ <dbl> 8806, 0, 0, 0, 0, 0, 0, 0, 8738, 8737, 8644, 8640, 0, 8540,~
## $ REMARKS    <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ REFNUM     <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, ~
```

Our task for analysis is to answer the following two questions:

1. Across the United States, which types of events (as indicated in the `EVTYPE` variable) are most harmful with respect to population health?
2. Across the United States, which types of events have the greatest economic consequences?

To answer these questions, we will be most interested in examining all of the event types for their values in the columns for fatalities, injuries, property damage, and crop damage. For the property and crop damage, we will also need the values stored in `PROPDMGEXP` and `CROPDMGEXP`, which encode information about the multiplier for values in the `PROPDMG` and `CROPDMG` columns.

First, let's look at the entered values for `EVTYPE`.

```
sample(unique(rawData$EVTYPE), 50)
```

```
##  [1] "THUNDERSTORM DAMAGE TO"         "Summary: Sept. 18"
##  [3] "THUNDERSTORM WIND 65 MPH"       "Monthly Snowfall"
##  [5] "DRY MICROBURST 84"              "TORNDAO"
##  [7] "DAM FAILURE"                    "COASTALSTORM"
##  [9] "Snow"                           "THUNDERSTORM WIND."
## [11] "ROUGH SURF"                     "LIGHT SNOW/FREEZING PRECIP"
## [13] "LIGHT SNOW"                     "HAIL 100"
## [15] "NONE"                           "WATERSPOUT/ TORNADO"
## [17] "HIGH WIND/WIND CHILL/BLIZZARD" "ASTRONOMICAL LOW TIDE"
## [19] "HIGH WIND"                      "APACHE COUNTY"
## [21] "EXTREME/RECORD COLD"            "Summary August 21"
## [23] "UNUSUALLY LATE SNOW"            "Temperature record"
## [25] "VOG"                            "HURRICANE EMILY"
## [27] "ICE STORM"                      "MONTHLY RAINFALL"
## [29] "SNOW AND WIND"                  "HEAVY SNOW AND"
## [31] "SAHARAN DUST"                   "THUNDERSTORM WIND/ TREES"
## [33] "HURRICANE GORDON"               "SNOW AND ICE STORM"
## [35] "Urban Flooding"                 "THUNDERSTORM WIND 59"
## [37] "THUNDERSTORM WIND/AWNING"       "HURRICANE OPAL"
## [39] "THUNDERSTORM"                   "THUNDERSTORM WINDS HAIL"
## [41] "WINTERY MIX"                    "HEAVY SNOW/FREEZING RAIN"
## [43] "FOG"                            "HIGH WAVES"
```

```
## [45] "DUST DEVEL"                    "MODERATE SNOW"
## [47] "TYPHOON"                       "Tidal Flooding"
## [49] "URBAN SMALL"                   "Summary of March 14"
```

The entries for EVTYPE are messy - there is a mix of upper and lower class characters used, typos, event types that are combined, etc. We can do a first pass clean up by converting to all upper case to allow combining of types that are similar but coded in different case.

```
length(unique(rawData$EVTYPE))
```

```
## [1] 977
```

```
length(unique(toupper(rawData$EVTYPE)))
```

```
## [1] 890
```

This quick fix resolves 87 coding errors. According to the NOAA documentation, there are only 55 specific event types that should be entered into the database. We will try to match the entered event type with the most relevant allowed event type. First, let's create a vector of the allowed event types, and visually inspect the EVTYPE entries.

```
allowed_events <- toupper(c("Astronomical Low Tide",
    "Avalanche", "Blizzard", "Coastal Flood", "Cold/Wind Chill",
    "Debris Flow", "Dense Fog", "Dense Smoke", "Drought",
    "Dust Devil", "Dust Storm", "Excessive Heat", "Extreme Cold/Wind Chill",
    "Flash Flood", "Flood", "Frost/Freeze", "Funnel Cloud",
    "Freezing Fog", "Hail", "Heat", "Heavy Rain", "Heavy Snow",
    "High Surf", "High Wind", "Hurricane (Typhoon)",
    "Ice Storm", "Lake-Effect Snow", "Lakeshore Flood",
    "Lightning", "Marine Dense Fog", "Marine Hail",
    "Marine Heavy Freezing Spray", "Marine High Wind",
    "Marine Hurricane/Typhoon", "Marine Lightning",
    "Marine Strong Wind", "Marine Thunderstorm Wind",
    "Marine Tropical Depression", "Marine Tropical Storm",
    "Rip Current", "Seiche", "Sleet", "Sneaker Wave",
    "Storm Surge/Tide", "Strong Wind", "Thunderstorm Wind",
    "Tornado", "Tropical Depression", "Tropical Storm",
    "Tsunami", "Volcanic Ash", "Waterspout", "Wildfire",
    "Winter Storm", "Winter Weather"))

head(unique(toupper(rawData$EVTYPE)), n = 25)
```

```
##  [1] "TORNADO"                    "TSTM WIND"
##  [3] "HAIL"                       "FREEZING RAIN"
##  [5] "SNOW"                       "ICE STORM/FLASH FLOOD"
##  [7] "SNOW/ICE"                   "WINTER STORM"
##  [9] "HURRICANE OPAL/HIGH WINDS"  "THUNDERSTORM WINDS"
## [11] "RECORD COLD"                "HURRICANE ERIN"
## [13] "HURRICANE OPAL"             "HEAVY RAIN"
## [15] "LIGHTNING"                  "THUNDERSTORM WIND"
## [17] "DENSE FOG"                  "RIP CURRENT"
```

```
## [19] "THUNDERSTORM WINS"            "FLASH FLOOD"
## [21] "FLASH FLOODING"               "HIGH WINDS"
## [23] "FUNNEL CLOUD"                 "TORNADO F0"
## [25] "THUNDERSTORM WINDS LIGHTNING"
```

One of the things we can see is that `Thunderstorm` is often encoded in shorthand, as `TSTM`. This can easily be replaced with `mutate` and the `str_replace` function. While tidying up the data, let's also select just the columns of interest that have health or economic impact values. Then, we'll convert the `EVTYPE` column values to upper case and date columns from character to date types. We'll also add a column coding whether or not the value is one of the allowed event types.

```
tidyData <- rawData %>%
    select(BGN_DATE, COUNTY, COUNTYNAME, STATE, EVTYPE,
        END_DATE, FATALITIES, INJURIES, PROPDMG, PROPDMGEXP,
        CROPDMG, CROPDMGEXP, REMARKS) %>%
    mutate(BGN_DATE = mdy_hms(BGN_DATE), END_DATE = mdy_hms(END_DATE),
        EVTYPE = toupper(EVTYPE), EVTYPE = str_replace(EVTYPE,
            "TSTM", "THUNDERSTORM"), ALLOWED = if_else(EVTYPE %in%
            allowed_events, true = "ALLOWED", false = "NOT_ALLOWED"))
```

Now, let's see how many events are allowed vs. not allowed event types:

```
with(tidyData, table(ALLOWED))
```

```
## ALLOWED
##     ALLOWED NOT_ALLOWED
##      861476       40821
```

There are still quite a few entries (~6%) are not properly classified for event type. We only care for ones that have a public health or economic impact, so let's filter for those and then see how many need to be fixed.

```
# add columns coding if there was health or
# economic damages, filter to keep only rows with
# at least one type of damages

tidyDataDamages <- tidyData %>%
    mutate(HealthImpact = if_else(condition = FATALITIES >
        0 | INJURIES > 0, true = TRUE, false = FALSE),
        EconImpact = if_else(condition = PROPDMG >
            0 | CROPDMG > 0, true = TRUE, false = FALSE)) %>%
    filter(HealthImpact == TRUE | EconImpact == TRUE)
with(tidyDataDamages, table(ALLOWED))
```

```
## ALLOWED
##     ALLOWED NOT_ALLOWED
##      236219       18414
```

Removing to only keep events with health or economic damages did not resolve the problem: still have ~8% of events that are not properly classified. We will first remove all of the correctly classified entries, then focus on the improperly classified entries to try to match them to the appropriate allowed event type.

```
# filter to only keep allowed event types.
tidyDataDamagesAllowed <- tidyDataDamages %>%
    filter(ALLOWED == "ALLOWED")

# get data that we need to fix event type i.e.
# NON-ALLOWED
tidyDataDamagesNonAllowed <- tidyDataDamages %>%
    filter(ALLOWED == "NOT_ALLOWED")

# look at which event types still need to be
# corrected
head(unique(tidyDataDamagesNonAllowed$EVTYPE), n = 25)
```

```
##  [1] "ICE STORM/FLASH FLOOD"          "HURRICANE OPAL/HIGH WINDS"
##  [3] "THUNDERSTORM WINDS"             "HURRICANE ERIN"
##  [5] "HURRICANE OPAL"                 "THUNDERSTORM WINS"
##  [7] "FLASH FLOODING"                 "TORNADO F0"
##  [9] "THUNDERSTORM WINDS LIGHTNING"   "THUNDERSTORM WINDS/HAIL"
## [11] "HIGH WINDS"                     "WIND"
## [13] "HEAVY RAINS"                    "LIGHTNING AND HEAVY RAIN"
## [15] "THUNDERSTORM WINDS HAIL"        "COLD"
## [17] "HEAVY RAIN/LIGHTNING"           "FLASH FLOODING/THUNDERSTORM WI"
## [19] "FLOODING"                       "EXTREME COLD"
## [21] "LIGHTNING/HEAVY RAIN"           "BREAKUP FLOODING"
## [23] "FREEZE"                         "RIVER FLOOD"
## [25] "HIGH WINDS HEAVY RAINS"
```

We will use a "fuzzy join" approach to try to match the coded event type to the closest allowed event type. This essentially works by calculating a distance matrix between the coded string and each of the strings in the allowed events vector, then returns the value with the shortest distance. It is implemented with the `fuzzyjoin` package, for which more details can be found here.

```
allowedEvents <- as_tibble(allowed_events)
colnames(allowedEvents) <- c("EVTYPE")

not_allowed <- as_tibble(unique(tidyDataDamagesNonAllowed$EVTYPE))
colnames(not_allowed) <- c("EVTYPE")
```

There are multiple methods for fuzzy joining, let's see which one works best to accurately find matches for our non-allowed events. We'll create a custom function to loop through all of the available methods, and return a dataframe containing the method name, number of remaining unmatched event types, and then number of event types that a fuzzy join found a corresponding match for.

```
string_match_test <- function(x, y, method = "lv",
    ...) {

    match <- stringdist_join(x, y, mode = "left", ignore_case = FALSE,
        method = method)
    colnames(match)[1] <- c("Test")
    colnames(match)[2] <- c("Matched")

    matches <- match %>%
```

```
        mutate(Match = case_when(Test == Matched ~
            "Exact", is.na(Matched) ~ "Unmatched",
            TRUE ~ "Replaced")) %>%
        count(Match)

    data.frame(matches)
}

match_methods <- c("osa", "lv", "dl", "hamming", "lcs",
    "qgram", "cosine", "jaccard", "jw", "soundex")

test <- map(match_methods, string_match_test, x = not_allowed,
    y = allowedEvents)
names(test) <- match_methods

out <- bind_rows(test, .id = "Method")
pivot_wider(out, id_cols = Method, names_from = Match,
    values_from = n)
```

```
## # A tibble: 10 x 3
##    Method  Replaced Unmatched
##    <chr>      <int>     <int>
##  1 osa           39       353
##  2 lv            39       353
##  3 dl            39       353
##  4 hamming        7       385
##  5 lcs           38       354
##  6 qgram         40       352
##  7 cosine     21560        NA
##  8 jaccard    21560        NA
##  9 jw         21560        NA
## 10 soundex      219       186
```

From this output, it looks like the **soundex** method found the most matches. You can read more about how this method works here. Let's look at the matches from the soundex method to make sure that it is finding accurate matches.

```
soundex <- stringdist_join(not_allowed, allowedEvents,
    method = "soundex")
colnames(soundex) <- c("Not_Allowed_EVTYPE", "Matched_Allowed_EVTYPE")
kable(head(soundex, n = 30), booktabs = TRUE) %>%
    kable_styling(latex_options = "striped")
```

Not_Allowed_EVTYPE	Matched_Allowed_EVTYPE
ICE STORM/FLASH FLOOD	ICE STORM
HURRICANE OPAL/HIGH WINDS	HURRICANE (TYPHOON)
THUNDERSTORM WINDS	THUNDERSTORM WIND
HURRICANE ERIN	HURRICANE (TYPHOON)
HURRICANE OPAL	HURRICANE (TYPHOON)
THUNDERSTORM WINS	THUNDERSTORM WIND
FLASH FLOODING	FLASH FLOOD
TORNADO F0	TORNADO
THUNDERSTORM WINDS LIGHTNING	THUNDERSTORM WIND
THUNDERSTORM WINDS/HAIL	THUNDERSTORM WIND
HIGH WINDS	HIGH WIND
HEAVY RAINS	HEAVY RAIN
LIGHTNING AND HEAVY RAIN	LIGHTNING
THUNDERSTORM WINDS HAIL	THUNDERSTORM WIND
HEAVY RAIN/LIGHTNING	HEAVY RAIN
FLASH FLOODING/THUNDERSTORM WI	FLASH FLOOD
EXTREME COLD	EXTREME COLD/WIND CHILL
LIGHTNING/HEAVY RAIN	LIGHTNING
HIGH WINDS HEAVY RAINS	HIGH WIND
HIGH WIND/SEAS	HIGH WIND
HIGH WINDS/HEAVY RAIN	HIGH WIND
HEAVY SNOW/WIND	HEAVY SNOW
THUNDERSTORM WINDS/FUNNEL CLOU	THUNDERSTORM WIND
WILD FIRES	WILDFIRE
WINTER STORM HIGH WINDS	WINTER STORM
WINTER STORM HIGH WINDS	WINTER WEATHER
WINTER STORMS	WINTER STORM
WINTER STORMS	WINTER WEATHER
THUNDERSTORMS WINDS	THUNDERSTORM WIND
THUNDERSTORMS	THUNDERSTORM WIND