Lab 8

Jonathan Eng

2AM April 26, 2020

Data Wrangling / Munging / Carpentry

Throughout this assignment you can use either the `tidyverse` package suite or `data.table` to answer but not base R. You can mix `data.table` with `magrittr` piping if you wish but don't go back and forth between `tbl_df`'s and `data.table` objects.

```
pacman::p_load(tidyverse, magrittr, data.table)
```

Load the `storms` dataset from the `dplyr` package and investigate it using `str` and `summary` and `head`. Which two columns should be converted to type factor? Do so below.

```
data("storms")
str(storms)
```

```
## tibble [10,010 x 13] (S3: tbl_df/tbl/data.frame)
##  $ name       : chr [1:10010] "Amy" "Amy" "Amy" "Amy" ...
##  $ year       : num [1:10010] 1975 1975 1975 1975 1975 ...
##  $ month      : num [1:10010] 6 6 6 6 6 6 6 6 6 6 ...
##  $ day        : int [1:10010] 27 27 27 27 28 28 28 28 29 29 ...
##  $ hour       : num [1:10010] 0 6 12 18 0 6 12 18 0 6 ...
##  $ lat        : num [1:10010] 27.5 28.5 29.5 30.5 31.5 32.4 33.3 34 34.4 34 ...
##  $ long       : num [1:10010] -79 -79 -79 -79 -78.8 -78.7 -78 -77 -75.8 -74.8 ...
##  $ status     : chr [1:10010] "tropical depression" "tropical depression" "tropical depression" "trop
##  $ category   : Ord.factor w/ 7 levels "-1"<"0"<"1"<"2"<..: 1 1 1 1 1 1 1 1 2 2 ...
##  $ wind       : int [1:10010] 25 25 25 25 25 25 25 30 35 40 ...
##  $ pressure   : int [1:10010] 1013 1013 1013 1013 1012 1012 1011 1006 1004 1002 ...
##  $ ts_diameter: num [1:10010] NA NA NA NA NA NA NA NA NA NA ...
##  $ hu_diameter: num [1:10010] NA NA NA NA NA NA NA NA NA NA ...
```

```
summary(storms)
```

```
##      name                year          month             day       
##  Length:10010       Min.   :1975   Min.   : 1.000   Min.   : 1.00  
##  Class :character   1st Qu.:1990   1st Qu.: 8.000   1st Qu.: 8.00  
##  Mode  :character   Median :1999   Median : 9.000   Median :16.00  
##                     Mean   :1998   Mean   : 8.779   Mean   :15.86  
##                     3rd Qu.:2006   3rd Qu.: 9.000   3rd Qu.:24.00  
##                     Max.   :2015   Max.   :12.000   Max.   :31.00  
## 
```

```
##       hour             lat             long           status
##  Min.   : 0.000   Min.   : 7.20   Min.   :-109.30   Length:10010
##  1st Qu.: 6.000   1st Qu.:17.50   1st Qu.: -80.70   Class :character
##  Median :12.000   Median :24.40   Median : -64.50   Mode  :character
##  Mean   : 9.114   Mean   :24.76   Mean   : -64.23
##  3rd Qu.:18.000   3rd Qu.:31.30   3rd Qu.: -48.60
##  Max.   :23.000   Max.   :51.90   Max.   :  -6.00
##
##  category       wind          pressure       ts_diameter       hu_diameter
##  -1:2545   Min.   : 10.00   Min.   : 882.0   Min.   :   0.00   Min.   :  0.00
##  0 :4373   1st Qu.: 30.00   1st Qu.: 985.0   1st Qu.:  69.05   1st Qu.:  0.00
##  1 :1685   Median : 45.00   Median : 999.0   Median : 138.09   Median :  0.00
##  2 : 628   Mean   : 53.49   Mean   : 992.1   Mean   : 166.76   Mean   : 21.41
##  3 : 363   3rd Qu.: 65.00   3rd Qu.:1006.0   3rd Qu.: 241.66   3rd Qu.: 28.77
##  4 : 348   Max.   :160.00   Max.   :1022.0   Max.   :1001.18   Max.   :345.23
##  5 :  68                                     NA's   :6528      NA's   :6528
```

```
head(storms)
```

```
## # A tibble: 6 x 13
##   name   year month   day  hour   lat  long status category  wind pressure
##   <chr> <dbl> <dbl> <int> <dbl> <dbl> <dbl> <chr>  <ord>    <int>    <int>
## 1 Amy    1975     6    27     0  27.5 -79   tropi~ -1          25     1013
## 2 Amy    1975     6    27     6  28.5 -79   tropi~ -1          25     1013
## 3 Amy    1975     6    27    12  29.5 -79   tropi~ -1          25     1013
## 4 Amy    1975     6    27    18  30.5 -79   tropi~ -1          25     1013
## 5 Amy    1975     6    28     0  31.5 -78.8 tropi~ -1          25     1012
## 6 Amy    1975     6    28     6  32.4 -78.7 tropi~ -1          25     1012
## # ... with 2 more variables: ts_diameter <dbl>, hu_diameter <dbl>
```

```
storms %<>%
  mutate(name = factor(name), status = factor(status))
storms
```

```
## # A tibble: 10,010 x 13
##    name   year month   day  hour   lat  long status category  wind pressure
##    <fct> <dbl> <dbl> <int> <dbl> <dbl> <dbl> <fct>  <ord>    <int>    <int>
##  1 Amy    1975     6    27     0  27.5 -79   tropi~ -1          25     1013
##  2 Amy    1975     6    27     6  28.5 -79   tropi~ -1          25     1013
##  3 Amy    1975     6    27    12  29.5 -79   tropi~ -1          25     1013
##  4 Amy    1975     6    27    18  30.5 -79   tropi~ -1          25     1013
##  5 Amy    1975     6    28     0  31.5 -78.8 tropi~ -1          25     1012
##  6 Amy    1975     6    28     6  32.4 -78.7 tropi~ -1          25     1012
##  7 Amy    1975     6    28    12  33.3 -78   tropi~ -1          25     1011
##  8 Amy    1975     6    28    18  34   -77   tropi~ -1          30     1006
##  9 Amy    1975     6    29     0  34.4 -75.8 tropi~ 0           35     1004
## 10 Amy    1975     6    29     6  34   -74.8 tropi~ 0           40     1002
## # ... with 10,000 more rows, and 2 more variables: ts_diameter <dbl>,
## #   hu_diameter <dbl>
```

Reorder the columns so name is first, status is second, category is third and the rest are the same.

```
storms %<>%
  select(name, status, category, everything())

storms
```

```
## # A tibble: 10,010 x 13
##    name  status category  year month   day  hour   lat  long  wind pressure
##    <fct> <fct>  <ord>    <dbl> <dbl> <int> <dbl> <dbl> <dbl> <int>    <int>
##  1 Amy   tropi~ -1        1975     6    27     0  27.5 -79      25     1013
##  2 Amy   tropi~ -1        1975     6    27     6  28.5 -79      25     1013
##  3 Amy   tropi~ -1        1975     6    27    12  29.5 -79      25     1013
##  4 Amy   tropi~ -1        1975     6    27    18  30.5 -79      25     1013
##  5 Amy   tropi~ -1        1975     6    28     0  31.5 -78.8    25     1012
##  6 Amy   tropi~ -1        1975     6    28     6  32.4 -78.7    25     1012
##  7 Amy   tropi~ -1        1975     6    28    12  33.3 -78      25     1011
##  8 Amy   tropi~ -1        1975     6    28    18  34   -77      30     1006
##  9 Amy   tropi~ 0         1975     6    29     0  34.4 -75.8    35     1004
## 10 Amy   tropi~ 0         1975     6    29     6  34   -74.8    40     1002
## # ... with 10,000 more rows, and 2 more variables: ts_diameter <dbl>,
## #   hu_diameter <dbl>
```

Find a subset of the data of storms only in the 1970’s.

```
storm1970s =
  storms %>%
    filter(year>=1970 & year<1980)
storm1970s
```

```
## # A tibble: 546 x 13
##    name  status category  year month   day  hour   lat  long  wind pressure
##    <fct> <fct>  <ord>    <dbl> <dbl> <int> <dbl> <dbl> <dbl> <int>    <int>
##  1 Amy   tropi~ -1        1975     6    27     0  27.5 -79      25     1013
##  2 Amy   tropi~ -1        1975     6    27     6  28.5 -79      25     1013
##  3 Amy   tropi~ -1        1975     6    27    12  29.5 -79      25     1013
##  4 Amy   tropi~ -1        1975     6    27    18  30.5 -79      25     1013
##  5 Amy   tropi~ -1        1975     6    28     0  31.5 -78.8    25     1012
##  6 Amy   tropi~ -1        1975     6    28     6  32.4 -78.7    25     1012
##  7 Amy   tropi~ -1        1975     6    28    12  33.3 -78      25     1011
##  8 Amy   tropi~ -1        1975     6    28    18  34   -77      30     1006
##  9 Amy   tropi~ 0         1975     6    29     0  34.4 -75.8    35     1004
## 10 Amy   tropi~ 0         1975     6    29     6  34   -74.8    40     1002
## # ... with 536 more rows, and 2 more variables: ts_diameter <dbl>,
## #   hu_diameter <dbl>
```

Find a subset of the data of storm observations only with category 4 and above and wind speed 100MPH and above.

```
storms_cat4_100mph =
  storms %>%
    filter(category == 4 & wind >= 100)
storms_cat4_100mph
```

```
## # A tibble: 348 x 13
##    name  status category  year month   day  hour   lat  long  wind pressure
##    <fct> <fct>  <ord>    <dbl> <dbl> <int> <dbl> <dbl> <dbl> <int>    <int>
##  1 Anita hurri~ 4         1977     9     2    12  23.7 -98     120      940
##  2 David hurri~ 4         1979     8    28     0  12.2 -52.9   115      947
##  3 David hurri~ 4         1979     8    28     6  12.5 -54.4   125      941
##  4 David hurri~ 4         1979     8    28    12  12.8 -55.7   130      938
##  5 David hurri~ 4         1979     8    28    18  13.2 -56.9   125      941
##  6 David hurri~ 4         1979     8    29     0  13.7 -58     120      944
##  7 David hurri~ 4         1979     8    29     6  14.2 -59.2   120      942
##  8 David hurri~ 4         1979     8    29    12  14.8 -60.3   125      938
##  9 David hurri~ 4         1979     8    29    18  15.3 -61.6   125      933
## 10 David hurri~ 4         1979     8    30     0  15.6 -62.8   130      929
## # ... with 338 more rows, and 2 more variables: ts_diameter <dbl>,
## #   hu_diameter <dbl>
```

Create a new feature `wind_speed_per_unit_pressure`.

```
storms %<>%
  mutate(wind_speed_per_unit_pressure = wind / pressure)
storms
```

```
## # A tibble: 10,010 x 14
##    name  status category  year month   day  hour   lat  long  wind pressure
##    <fct> <fct>  <ord>    <dbl> <dbl> <int> <dbl> <dbl> <dbl> <int>    <int>
##  1 Amy   tropi~ -1        1975     6    27     0  27.5 -79      25     1013
##  2 Amy   tropi~ -1        1975     6    27     6  28.5 -79      25     1013
##  3 Amy   tropi~ -1        1975     6    27    12  29.5 -79      25     1013
##  4 Amy   tropi~ -1        1975     6    27    18  30.5 -79      25     1013
##  5 Amy   tropi~ -1        1975     6    28     0  31.5 -78.8    25     1012
##  6 Amy   tropi~ -1        1975     6    28     6  32.4 -78.7    25     1012
##  7 Amy   tropi~ -1        1975     6    28    12  33.3 -78      25     1011
##  8 Amy   tropi~ -1        1975     6    28    18  34   -77      30     1006
##  9 Amy   tropi~ 0         1975     6    29     0  34.4 -75.8    35     1004
## 10 Amy   tropi~ 0         1975     6    29     6  34   -74.8    40     1002
## # ... with 10,000 more rows, and 3 more variables: ts_diameter <dbl>,
## #   hu_diameter <dbl>, wind_speed_per_unit_pressure <dbl>
```

Create a new feature: `average_diameter` which averages the two diameter metrics. If one is missing, then use the value of the one that is present. If both are missing, leave missing.

```
storms %>%
  mutate(average_diameter = if_else(is.na(ts_diameter+hu_diameter),
                                    if_else(is.na(ts_diameter) & !is.na(hu_diameter), hu_diameter, ts_di
                                    (ts_diameter + hu_diameter) / 2 ))
```

```
## # A tibble: 10,010 x 15
##    name  status category  year month   day  hour   lat  long  wind pressure
##    <fct> <fct>  <ord>    <dbl> <dbl> <int> <dbl> <dbl> <dbl> <int>    <int>
##  1 Amy   tropi~ -1        1975     6    27     0  27.5 -79      25     1013
##  2 Amy   tropi~ -1        1975     6    27     6  28.5 -79      25     1013
##  3 Amy   tropi~ -1        1975     6    27    12  29.5 -79      25     1013
##  4 Amy   tropi~ -1        1975     6    27    18  30.5 -79      25     1013
##  5 Amy   tropi~ -1        1975     6    28     0  31.5 -78.8    25     1012
##  6 Amy   tropi~ -1        1975     6    28     6  32.4 -78.7    25     1012
##  7 Amy   tropi~ -1        1975     6    28    12  33.3 -78      25     1011
##  8 Amy   tropi~ -1        1975     6    28    18  34   -77      30     1006
##  9 Amy   tropi~ 0         1975     6    29     0  34.4 -75.8    35     1004
## 10 Amy   tropi~ 0         1975     6    29     6  34   -74.8    40     1002
## # ... with 10,000 more rows, and 4 more variables: ts_diameter <dbl>,
## #   hu_diameter <dbl>, wind_speed_per_unit_pressure <dbl>,
## #   average_diameter <dbl>
```

For each storm, summarize the maximum wind speed. “Summarize” means create a new dataframe with only the summary metrics you care about.

```
storms %>%
  group_by(name) %>%
  slice(which.max(wind)) %>%
  summarize(max_wind = wind)
```

```
## # A tibble: 198 x 2
##    name     max_wind
##    <fct>       <int>
##  1 AL011993       30
##  2 AL012000       25
##  3 AL021992       30
##  4 AL021994       30
##  5 AL021999       30
##  6 AL022000       30
##  7 AL022001       25
##  8 AL022003       30
##  9 AL022006       45
## 10 AL031987       40
## # ... with 188 more rows
```

Order your dataset by maximum wind speed storm but within the rows of storm show the observations in time order from early to late.

```
storms %<>%
  select(wind, year, month, day, hour, everything()) %<>%
  arrange(wind, year, month, day, hour)
storms
```

```
## # A tibble: 10,010 x 14
##     wind  year month   day  hour name  status category   lat  long pressure
##    <int> <dbl> <dbl> <int> <dbl> <fct> <fct>  <ord>    <dbl> <dbl>    <int>
##  1    10  1986     6    28     6 Bonn~ tropi~ -1        36.5 -91.3     1013
##  2    10  1986     6    28    12 Bonn~ tropi~ -1        37.2 -90       1012
##  3    10  1987     8    16    18 AL03~ tropi~ -1        30.9 -83.2     1014
##  4    10  1987     8    17     0 AL03~ tropi~ -1        31.4 -82.9     1015
##  5    10  1987     8    17     6 AL03~ tropi~ -1        31.8 -82.3     1015
##  6    10  1994     7     7     0 Albe~ tropi~ -1        32.7 -86.3     1012
##  7    10  1994     7     7     6 Albe~ tropi~ -1        32.7 -86.6     1012
##  8    10  1994     7     7    12 Albe~ tropi~ -1        32.8 -86.8     1012
##  9    10  1994     7     7    18 Albe~ tropi~ -1        33   -87       1013
## 10    15  1979     7    27    12 Clau~ tropi~ -1        34   -95.9     1007
## # ... with 10,000 more rows, and 3 more variables: ts_diameter <dbl>,
## #   hu_diameter <dbl>, wind_speed_per_unit_pressure <dbl>
```

Find the strongest storm by wind speed per year.

```
storms %>%
  group_by(year) %>%
  slice(which.max(wind)) %>%
  summarize(name, max_wind = wind)
```

```
## # A tibble: 41 x 3
##     year name     max_wind
##    <dbl> <fct>       <int>
##  1  1975 Caroline      100
##  2  1976 Belle         105
##  3  1977 Anita         150
##  4  1978 Cora           80
##  5  1979 David         150
##  6  1980 Ivan           90
##  7  1981 Harvey        115
##  8  1982 Debby         115
##  9  1983 Alicia        100
## 10  1984 Diana         115
## # ... with 31 more rows
```

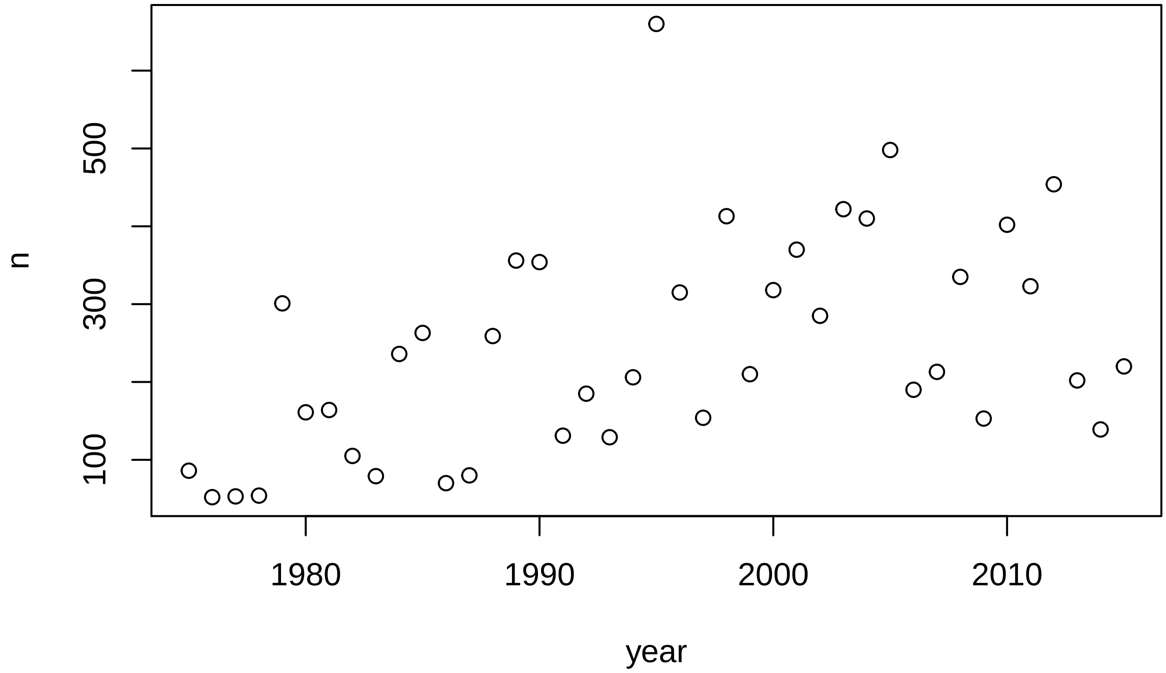
For each named storm, find its maximum category, wind speed, pressure and diameters. Do not allow the max to be NA (unless all the measurements for that storm were NA).

```
storms %>%
  group_by(name) %>%
  summarize(max_category = max(category),
            max_wind = max(wind),
            max_pressure = max(pressure),
            max_hu_diameter = ifelse( is.infinite(max(hu_diameter, na.rm = TRUE)), NA, max(hu_diameter,
            max_ts_diameter = ifelse( is.infinite(max(ts_diameter, na.rm = TRUE)), NA, max(ts_diameter,
```

```
## # A tibble: 198 x 6
##    name     max_category max_wind max_pressure max_hu_diameter max_ts_diameter
##    <fct>    <ord>           <int>        <int>           <dbl>           <dbl>
##  1 AL011993 -1                 30         1003              NA              NA
##  2 AL012000 -1                 25         1010              NA              NA
##  3 AL021992 -1                 30         1009              NA              NA
##  4 AL021994 -1                 30         1017              NA              NA
##  5 AL021999 -1                 30         1006              NA              NA
##  6 AL022000 -1                 30         1010              NA              NA
##  7 AL022001 -1                 25         1012              NA              NA
##  8 AL022003 -1                 30         1010              NA              NA
##  9 AL022006 0                  45         1008               0            69.0
## 10 AL031987 0                  40         1015              NA              NA
## # ... with 188 more rows
```

For each year in the dataset, tally the number of storms. “Tally” is a fancy word for “count the number of”. Plot the number of storms by year. Any pattern?

```
storms_per_year =
storms %>%
  count(year, sort = TRUE)
plot(storms_per_year)
```



```
#More storms occur as years increase
```

For each year in the dataset, tally the storms by category.

```
storms_per_year_per_category =
 storms %>%
  group_by(year, category) %>%
  count(category, sort = TRUE)
storms_per_year_per_category
```

```
## # A tibble: 233 x 3
## # Groups:   year, category [233]
##     year category     n
##    <dbl> <ord>    <int>
##  1  2012 0          276
##  2  1995 0          247
##  3  2005 0          221
##  4   2011 0          203
##  5   2010 0          193
##  6   2003 0          186
##  7   2008 0          183
##  8   2004 0          166
##  9   1995 1          164
## 10   1995 -1         158
## # ... with 223 more rows
```

For each year in the dataset, find the maximum wind speed per status level.

```
storms %>%
  group_by(status) %>%
  summarize(max_wind_speed = max(wind))
```

```
## # A tibble: 3 x 2
##   status              max_wind_speed
##   <fct>                        <int>
## 1 hurricane                      160
## 2 tropical depression             30
## 3 tropical storm                  70
```

For each storm, summarize its average location in latitude / longitude coordinates.

```
storms %>%
  group_by(status) %>%
  summarize(max_wind_speed = max(wind))
```

```
## # A tibble: 3 x 2
##   status              max_wind_speed
##   <fct>                        <int>
## 1 hurricane                      160
## 2 tropical depression             30
## 3 tropical storm                  70
```

For each storm, summarize its duration in number of hours (to the nearest 6hr increment).

```
#TO-DO
```

Convert year, month, day, hour into the variable `timestamp` using the `lubridate` package.

```
#pacman::p_load(lubridate)
library(lubridate)
```

```
##
## Attaching package: 'lubridate'

## The following objects are masked from 'package:data.table':
##
##     hour, isoweek, mday, minute, month, quarter, second, wday, week,
##     yday, year
## The following objects are masked from 'package:dplyr':
##
##     intersect, setdiff, union

## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
storms %<>%
  mutate(timestamp = ymd_h(paste(year, month, day, hour, sep = "-"))) %<>%
  select(-year, -month, -day, -hour)

storms
```

```
## # A tibble: 10,010 x 11
##     wind name  status category   lat  long pressure ts_diameter hu_diameter
##    <int> <fct> <fct>  <ord>    <dbl> <dbl>    <int>       <dbl>       <dbl>
##  1    10 Bonn~ tropi~ -1        36.5 -91.3     1013          NA          NA
##  2    10 Bonn~ tropi~ -1        37.2 -90       1012          NA          NA
##  3    10 AL03~ tropi~ -1        30.9 -83.2     1014          NA          NA
##  4    10 AL03~ tropi~ -1        31.4 -82.9     1015          NA          NA
##  5    10 AL03~ tropi~ -1        31.8 -82.3     1015          NA          NA
##  6    10 Albe~ tropi~ -1        32.7 -86.3     1012          NA          NA
##  7    10 Albe~ tropi~ -1        32.7 -86.6     1012          NA          NA
##  8    10 Albe~ tropi~ -1        32.8 -86.8     1012          NA          NA
##  9    10 Albe~ tropi~ -1        33   -87       1013          NA          NA
## 10    15 Clau~ tropi~ -1        34   -95.9     1007          NA          NA
## # ... with 10,000 more rows, and 2 more variables:
## #   wind_speed_per_unit_pressure <dbl>, timestamp <dttm>
```

Using the lubridate package, create new variables day_of_week which is a factor with levels “Sunday”, “Monday”, … “Saturday” and week_of_year which is integer 1, 2, …, 52.

```
storms %<>%
  mutate(day_of_week = weekdays(timestamp), week_of_year = week(timestamp))
storms
```

```
## # A tibble: 10,010 x 13
##     wind name  status category   lat  long pressure ts_diameter hu_diameter
##    <int> <fct> <fct>  <ord>    <dbl> <dbl>    <int>       <dbl>       <dbl>
##  1    10 Bonn~ tropi~ -1        36.5 -91.3     1013          NA          NA
##  2    10 Bonn~ tropi~ -1        37.2 -90       1012          NA          NA
##  3    10 AL03~ tropi~ -1        30.9 -83.2     1014          NA          NA
##  4    10 AL03~ tropi~ -1        31.4 -82.9     1015          NA          NA
##  5    10 AL03~ tropi~ -1        31.8 -82.3     1015          NA          NA
##  6    10 Albe~ tropi~ -1        32.7 -86.3     1012          NA          NA
##  7    10 Albe~ tropi~ -1        32.7 -86.6     1012          NA          NA
##  8    10 Albe~ tropi~ -1        32.8 -86.8     1012          NA          NA
##  9    10 Albe~ tropi~ -1        33   -87       1013          NA          NA
## 10    15 Clau~ tropi~ -1        34   -95.9     1007          NA          NA
## # ... with 10,000 more rows, and 4 more variables:
## #   wind_speed_per_unit_pressure <dbl>, timestamp <dttm>, day_of_week <chr>,
## #   week_of_year <dbl>
```

For each storm, summarize the day in which is started in the following format “Friday, June 27, 1975”.

```
#TO-DO
```

Create a new factor variable `decile_windspeed` by binning wind speed into 10 bins.

```
#TO-DO
```

Create a new data frame `serious_storms` which are category 3 and above hurricanes.

```
serious_storms =
  storms %>%
    filter(category >= 3)
serious_storms
```

```
## # A tibble: 779 x 13
##     wind name  status category   lat  long pressure ts_diameter hu_diameter
##    <int> <fct> <fct>  <ord>    <dbl> <dbl>    <int>       <dbl>       <dbl>
##  1   100 Caro~ hurri~ 3         24   -97        973          NA          NA
##  2   100 Caro~ hurri~ 3         24.1 -97.5      963          NA          NA
##  3   100 Belle hurri~ 3         29.5 -75.3      958          NA          NA
##  4   100 David hurri~ 3         19.3 -72        978          NA          NA
##  5   100 Fred~ hurri~ 3         25.7 -85.8      960          NA          NA
##  6   100 Floyd hurri~ 3         26.4 -69.1      981          NA          NA
##  7   100 Floyd hurri~ 3         27.5 -68.9      978          NA          NA
##  8   100 Floyd hurri~ 3         28.4 -68.5      975          NA          NA
##  9   100 Floyd hurri~ 3         29.3 -67.8      975          NA          NA
## 10   100 Harv~ hurri~ 3         32.1 -60.3      963          NA          NA
## # ... with 769 more rows, and 4 more variables:
## #   wind_speed_per_unit_pressure <dbl>, timestamp <dttm>, day_of_week <chr>,
## #   week_of_year <dbl>
```

In `serious_storms`, merge the variables lat and long together into `lat_long` with values `lat / long` as a string.

```
serious_storms %<>%
  unite(lat_long, lat, long, sep = " / ")
serious_storms
```

```
## # A tibble: 779 x 12
##     wind name  status category lat_long pressure ts_diameter hu_diameter
##    <int> <fct> <fct>  <ord>    <chr>       <int>       <dbl>       <dbl>
##  1   100 Caro~ hurri~ 3        24 / -97      973          NA          NA
##  2   100 Caro~ hurri~ 3        24.1 / ~      963          NA          NA
##  3   100 Belle hurri~ 3        29.5 / ~      958          NA          NA
##  4   100 David hurri~ 3        19.3 / ~      978          NA          NA
##  5   100 Fred~ hurri~ 3        25.7 / ~      960          NA          NA
##  6   100 Floyd hurri~ 3        26.4 / ~      981          NA          NA
##  7   100 Floyd hurri~ 3        27.5 / ~      978          NA          NA
##  8   100 Floyd hurri~ 3        28.4 / ~      975          NA          NA
##  9   100 Floyd hurri~ 3        29.3 / ~      975          NA          NA
## 10   100 Harv~ hurri~ 3        32.1 / ~      963          NA          NA
## # ... with 769 more rows, and 4 more variables:
## #   wind_speed_per_unit_pressure <dbl>, timestamp <dttm>, day_of_week <chr>,
## #   week_of_year <dbl>
```

Let's return now to the original storms data frame. For each category, find the average wind speed, pressure and diameters (do not count the NA's in your averaging).

```
storms %>%
  group_by(category) %>%
  summarise(avg_wind_speed = mean(wind),
            avg_pressure = mean(pressure),
            avg_ts_diameter = mean(ts_diameter, na.rm = TRUE),
            avg_hu_diameter = mean(hu_diameter, na.rm = TRUE))
```

```
## # A tibble: 7 x 5
##   category avg_wind_speed avg_pressure avg_ts_diameter avg_hu_diameter
##   <ord>             <dbl>        <dbl>           <dbl>           <dbl>
## 1 -1                 27.3        1008.               0               0
## 2 0                  45.8         999.            160.               0
## 3 1                  70.9         982.            278.            57.3
## 4 2                  89.4         967.            282.            78.8
## 5 3                 105.          954.            307.            91.4
## 6 4                 122.          940.            315.           102.
## 7 5                 145.          916.            317.           120.
```

For each named storm, find its maximum category, wind speed, pressure and diameters (do not allow the max to be NA) and the number of readings (i.e. observations).

```
storms %>%
  group_by(name) %>%
  summarize(max_category = max(category),
            max_wind_speed = max(wind),
            max_pressure = max(pressure),
            max_hu_diameter = max(hu_diameter, na.rm = TRUE),
            max_ts_diameter = max(ts_diameter, na.rm = TRUE),
            readings = n() )
```

```
## # A tibble: 198 x 7
##    name  max_category max_wind_speed max_pressure max_hu_diameter
##    <fct> <ord>                 <int>        <int>           <dbl>
##  1 AL01~ -1                       30         1003            -Inf
##  2 AL01~ -1                       25         1010            -Inf
##  3 AL02~ -1                       30         1009            -Inf
##  4 AL02~ -1                       30         1017            -Inf
##  5 AL02~ -1                       30         1006            -Inf
##  6 AL02~ -1                       30         1010            -Inf
##  7 AL02~ -1                       25         1012            -Inf
##  8 AL02~ -1                       30         1010            -Inf
##  9 AL02~ 0                        45         1008               0
## 10 AL03~ 0                        40         1015            -Inf
## # ... with 188 more rows, and 2 more variables: max_ts_diameter <dbl>,
## #   readings <int>
```

Calculate the distance from each storm observation to Miami in a new variable `distance_to_miami`. This is very challenging. You will need a function that computes distances from two sets of latitude / longitude coordinates.

```
MIAMI_COORDS = c(25.7617, -80.1918)

get_distance = function(end, start){
  earth = 3958.8 #Miles

  d_longitude = (end[2] - start[2]) * 180 / pi
  d_latitude  = (end[1] - start[1]) * 180 / pi

  a = (sin(d_latitude/2))**2 + cos(start[1]) * cos(end[1]) * (sin(d_longitude/2))**2
  c = 2 * atan2( sqrt(a), sqrt(1-a) )
  distance = earth * c

  distance
}

storms %>%
  mutate(distance_to_miami = get_distance(MIAMI_COORDS, c(lat, long))) %>%
  select(lat, long, distance_to_miami, everything())
```

```
## # A tibble: 10,010 x 14
##      lat  long distance_to_mia~  wind name  status category pressure ts_diameter
##    <dbl> <dbl>            <dbl> <int> <fct> <fct>  <ord>       <int>       <dbl>
##  1  36.5 -91.3            5033.    10 Bonn~ tropi~ -1           1013          NA
##  2  37.2 -90              5033.    10 Bonn~ tropi~ -1           1012          NA
##  3  30.9 -83.2            5033.    10 AL03~ tropi~ -1           1014          NA
##  4  31.4 -82.9            5033.    10 AL03~ tropi~ -1           1015          NA
##  5  31.8 -82.3            5033.    10 AL03~ tropi~ -1           1015          NA
##  6  32.7 -86.3            5033.    10 Albe~ tropi~ -1           1012          NA
##  7  32.7 -86.6            5033.    10 Albe~ tropi~ -1           1012          NA
##  8  32.8 -86.8            5033.    10 Albe~ tropi~ -1           1012          NA
##  9  33   -87              5033.    10 Albe~ tropi~ -1           1013          NA
## 10  34   -95.9            5033.    15 Clau~ tropi~ -1           1007          NA
## # ... with 10,000 more rows, and 5 more variables: hu_diameter <dbl>,
## #   wind_speed_per_unit_pressure <dbl>, timestamp <dttm>, day_of_week <chr>,
## #   week_of_year <dbl>
```

For each storm observation, use the function from the previous question to calculate the distance it moved since the previous observation.

```
#TO-DO
```

For each storm, find the total distance it moved over its observations and its total displacement. "Distance" is a scalar quantity that refers to "how much ground an object has covered" during its motion. "Displacement" is a vector quantity that refers to "how far out of place an object is"; it is the object's overall change in position.

```
#TO-DO
```

For each storm observation, calculate the average speed the storm moved in location.

```
#TO-DO
```

For each storm, calculate its average ground speed (how fast its eye is moving which is different from windspeed around the eye).

```
#TO-DO
```

Is there a relationship between average ground speed and maximum category attained? Use a dataframe summary (not a regression).

```
#TO-DO
```

Now we want to transition to building real design matrices for prediction. This is more in tune with what happens in the real world. Large data dump and you convert it into X and y how you see fit.

Suppose we wish to predict the following: given the first three readings of a storm, can you predict its maximum wind speed? Identify the `y` and identify which features you need $x_1, ... x_p$ and build that matrix with `dplyr` functions. This is not easy, but it is what it's all about. Feel free to "featurize" as creatively as you would like. You aren't going to overfit if you only build a few features relative to the total 198 storms.

```
#TO-DO
```

Fit your model. Validate it. Assess your level of success at this endeavor.

Interactions in linear models

Load the Boston Housing Data from package `MASS` and use `str` and `summary` to remind yourself of the features and their types and then use `?MASS::Boston` to read an English description of the features.

```
data(Boston, package = "MASS")
str(Boston)
```

```
## 'data.frame':    506 obs. of  14 variables:
##  $ crim   : num  0.00632 0.02731 0.02729 0.03237 0.06905 ...
##  $ zn     : num  18 0 0 0 0 0 12.5 12.5 12.5 12.5 ...
##  $ indus  : num  2.31 7.07 7.07 2.18 2.18 2.18 7.87 7.87 7.87 7.87 ...
##  $ chas   : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ nox    : num  0.538 0.469 0.469 0.458 0.458 0.458 0.524 0.524 0.524 0.524 ...
##  $ rm     : num  6.58 6.42 7.18 7 7.15 ...
##  $ age    : num  65.2 78.9 61.1 45.8 54.2 58.7 66.6 96.1 100 85.9 ...
##  $ dis    : num  4.09 4.97 4.97 6.06 6.06 ...
##  $ rad    : int  1 2 2 3 3 3 5 5 5 5 ...
##  $ tax    : num  296 242 242 222 222 222 311 311 311 311 ...
##  $ ptratio: num  15.3 17.8 17.8 18.7 18.7 18.7 15.2 15.2 15.2 15.2 ...
##  $ black  : num  397 397 393 395 397 ...
##  $ lstat  : num  4.98 9.14 4.03 2.94 5.33 ...
##  $ medv   : num  24 21.6 34.7 33.4 36.2 28.7 22.9 27.1 16.5 18.9 ...
```

```
summary(Boston)
```

```
##       crim                zn             indus            chas
##  Min.   : 0.00632   Min.   :  0.00   Min.   : 0.46   Min.   :0.00000
##  1st Qu.: 0.08205   1st Qu.:  0.00   1st Qu.: 5.19   1st Qu.:0.00000
##  Median : 0.25651   Median :  0.00   Median : 9.69   Median :0.00000
##  Mean   : 3.61352   Mean   : 11.36   Mean   :11.14   Mean   :0.06917
##  3rd Qu.: 3.67708   3rd Qu.: 12.50   3rd Qu.:18.10   3rd Qu.:0.00000
##  Max.   :88.97620   Max.   :100.00   Max.   :27.74   Max.   :1.00000
##       nox               rm             age              dis
##  Min.   :0.3850   Min.   :3.561   Min.   :  2.90   Min.   : 1.130
##  1st Qu.:0.4490   1st Qu.:5.886   1st Qu.: 45.02   1st Qu.: 2.100
##  Median :0.5380   Median :6.208   Median : 77.50   Median : 3.207
##  Mean   :0.5547   Mean   :6.285   Mean   : 68.57   Mean   : 3.795
##  3rd Qu.:0.6240   3rd Qu.:6.623   3rd Qu.: 94.08   3rd Qu.: 5.188
##  Max.   :0.8710   Max.   :8.780   Max.   :100.00   Max.   :12.127
##       rad              tax           ptratio          black
##  Min.   : 1.000   Min.   :187.0   Min.   :12.60   Min.   :  0.32
##  1st Qu.: 4.000   1st Qu.:279.0   1st Qu.:17.40   1st Qu.:375.38
##  Median : 5.000   Median :330.0   Median :19.05   Median :391.44
##  Mean   : 9.549   Mean   :408.2   Mean   :18.46   Mean   :356.67
##  3rd Qu.:24.000   3rd Qu.:666.0   3rd Qu.:20.20   3rd Qu.:396.23
##  Max.   :24.000   Max.   :711.0   Max.   :22.00   Max.   :396.90
##      lstat            medv
##  Min.   : 1.73   Min.   : 5.00
##  1st Qu.: 6.95   1st Qu.:17.02
##  Median :11.36   Median :21.20
##  Mean   :12.65   Mean   :22.53
##  3rd Qu.:16.95   3rd Qu.:25.00
##  Max.   :37.97   Max.   :50.00
```

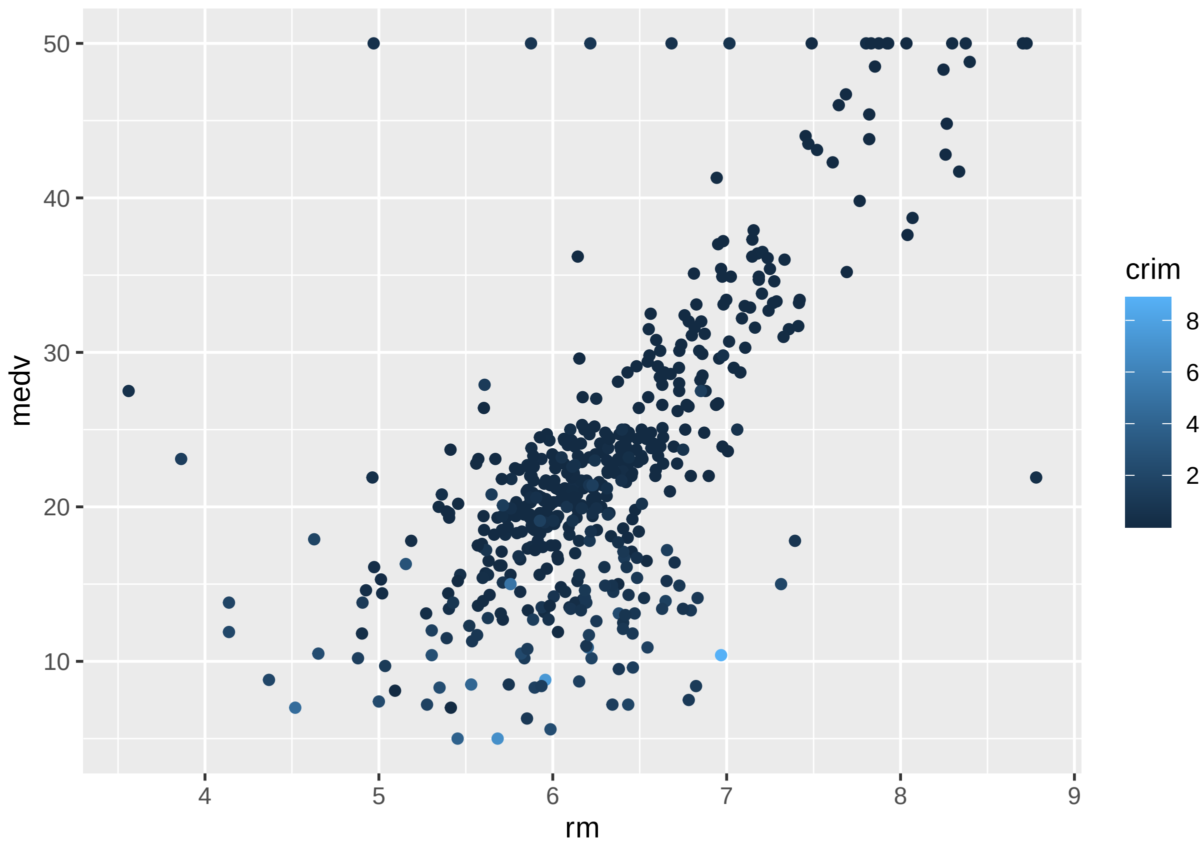
```
?MASS::Boston
```

```
## starting httpd help server ... done
```

Using what you learned about the Boston Housing Data in the previous question, try to guess which features are interacting. Confirm using plots in `ggplot` that illustrate three (or more) features.

```
#pacman::p_load(ggplot2)
library(ggplot2)

Boston %>%
  ggplot(aes(x = rm, y = medv)) + geom_point(aes(color = crim))
```



Once an interaction has been located, confirm the "non-linear linear" model with the interaction term does better than just the vanilla linear model by demonstrating a lower RMSE. In Econ 382 you would test this explicitly using a hypothesis test. We know in this class than increasing p yields alower RMSE. But the exercise is still a good one.

```
mod = lm(medv ~ ., Boston)
modv = lm(medv ~ + . + (rm*crim), Boston)

summary(mod)$sigma
```

```
## [1] 4.745298
```

```
summary(modv)$sigma
```

```
## [1] 4.555341
```

Repeat this procedure for another interaction with two different features (not used in the previous interaction you found) and verify.

```
modv = lm(medv ~ . + (indus * tax), Boston)

summary(mod)$sigma
```

```
## [1] 4.745298
```

```
summary(modv)$sigma
```

```
## [1] 4.71553
```

Fit a model using all possible first-order interactions. Verify it is "better" than the linear model. Do you think you overfit? Why or why not?

```
modv = lm(medv ~ .*. , Boston)

summary(mod)$sigma
```

```
## [1] 4.745298
```

```
summary(modv)$sigma
```

```
## [1] 2.851634
```

```
#The model is most likely overfit due to the large increase in complexity
```

CV

Use 5-fold CV to estimate the generalization error of the model with all interactions.

```
#TO-DO
```