Coursework 3

36071280

02/11/2021

Question 1

Q1. Read in the data and print the dimensions of the Storm Events data frame

```
storm_events <- read.csv("Australia_severe_storms_1975-2015.csv")
print(dim(storm_events))
```

```
## [1] 14457    14
```

Question 2

Q2. Clean the data by removing the variable ID and also Waterspout events from the database. Print the dimensions of the cleaned data frame. Also print the first few rows without the 6 columns of comments, without creating an intermediate data frame.

```
# Removing variable ID and Waterspout event
storm_events_clean <- storm_events %>% select(-(ID)) %>%
  filter(storm_events$Database != "Waterspout")

print(dim(storm_events_clean))
```

```
## [1] 14417    13
```

```
column_names <- colnames(storm_events_clean)
list_of_comment_columns <- column_names[8:13]
head(storm_events_clean %>% select(-all_of(list_of_comment_columns)))
```

```
##   Event.ID Database        Date.Time   Nearest.town State Latitude Longitude
## 1    20812     Wind 23/11/1975 07:00         SYDNEY   NSW -33.8834  151.2167
## 2    20813  Tornado 02/12/1975 14:00         BARHAM   NSW -35.6333  144.1333
## 3    20814     Wind 09/01/1976 08:50 COFF'S HARBOUR   NSW -30.3167  153.1167
## 4    20815     Hail 16/02/1976 14:00      BANKSTOWN   NSW -33.8834  151.2167
## 5    20816     Rain 25/10/1976 14:00          BOOMI   NSW -28.4333  152.6167
## 6    20817     Hail 08/11/1976 14:00          YOUNG   NSW -34.3167  148.3000
```

Question 3

Q3. Add a column to your data frame containing the time zone of each event using the following OlsonNames() classifications.

```
#declaring variables for timezone mapping

list_of_relevant_australian_tz <- c("QLD" = "Australia/Queensland",
                                    "NSW" = "Australia/NSW",
```

```
                            "NSW_BrokenHill" = "Australia/Broken_Hill",
                            "VIC" = "Australia/Victoria",
                            "SA"="Australia/South",
                            "WA" = "Australia/West",
                            "TAS"= "Australia/Tasmania",
                            "NT" = "Australia/North",
                            "ACT" = "Australia/ACT")

new_south_wales <- "NSW"
aus_central_time <- "ACT"
broken_hill <- "NSW_BrokenHill"
broken_hill_expr <- "broken hill"
```

```
allocate_tz <-function(state, nearest_town) {

  if (length(state) && !is.na(state))
  {
    if (state != new_south_wales)
      return (list_of_relevant_australian_tz[state])
    else if (state == new_south_wales)
    {
      if (length(nearest_town) && !is.na(nearest_town) &&
        str_detect(tolower(nearest_town), broken_hill_expr))
        return (list_of_relevant_australian_tz[broken_hill])
      else
        return (list_of_relevant_australian_tz[new_south_wales])
    }
  }
  return (list_of_relevant_australian_tz[aus_central_time])
}
```

```
storm_events_tz <- storm_events_clean %>% mutate(AustralianTimeZone = NA)

for ( i in 1:nrow(storm_events_tz))
{
  storm_events_tz$AustralianTimeZone[i]<-allocate_tz(storm_events_clean$State[i],
                                                     storm_events_clean$Nearest.town[i])
}
```

Question 4

Q4. Parse the date, time and time zones from the necessary columns to create a new variable in the data frame which converts the time into UTC. You may need the function lubridate::as_datetime() and/or the use of loops. Print the first few rows of the resultant data frame, without the 6 columns of comments, again without creating an intermediate data frame.

```
library(lubridate)
```

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

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

```
storm_events_utc <- data.frame()
for ( s in  list_of_relevant_australian_tz)
{
  storm_events_utc <- rbind(storm_events_utc, storm_events_tz %>%
                              filter(storm_events_tz$AustralianTimeZone == s) %>%
                              mutate(UTCDateTime = (Date.Time %>% dmy_hm(tz = s)
                                                    %>%  as_datetime())))
}

storm_events_utc <- storm_events_utc %>% arrange(storm_events_utc$Event.ID)
head(storm_events_utc %>% select(-all_of(list_of_comment_columns)))
```

```
##   Event.ID Database        Date.Time   Nearest.town State Latitude Longitude
## 1    20812     Wind 23/11/1975 07:00         SYDNEY   NSW -33.8834  151.2167
## 2    20813  Tornado 02/12/1975 14:00         BARHAM   NSW -35.6333  144.1333
## 3    20814     Wind 09/01/1976 08:50 COFF'S HARBOUR   NSW -30.3167  153.1167
## 4    20815     Hail 16/02/1976 14:00      BANKSTOWN   NSW -33.8834  151.2167
## 5    20816     Rain 25/10/1976 14:00          BOOMI   NSW -28.4333  152.6167
## 6    20817     Hail 08/11/1976 14:00          YOUNG   NSW -34.3167  148.3000
##   AustralianTimeZone         UTCDateTime
## 1      Australia/NSW 1975-11-22 20:00:00
## 2      Australia/NSW 1975-12-02 03:00:00
## 3      Australia/NSW 1976-01-08 21:50:00
## 4      Australia/NSW 1976-02-16 03:00:00
## 5      Australia/NSW 1976-10-25 04:00:00
## 6      Australia/NSW 1976-11-08 03:00:00
```

Question 5

Q5. Create new variables for the month and year of each event. Print the first few rows of the resultant data frame, without the 6 columns of comments and without creating an intermediate data frame.

```
#Adding month and year columns
storm_events_with_cols <- storm_events_utc %>%
  mutate(month_storm = month(storm_events_utc$UTCDateTime),
         year_storm = year(storm_events_utc$UTCDateTime))
print(head(storm_events_with_cols %>% select(-list_of_comment_columns)))
```

```
## Note: Using an external vector in selections is ambiguous.
## i Use `all_of(list_of_comment_columns)` instead of `list_of_comment_columns` to silence this message
## i See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
## This message is displayed once per session.
```

```
##   Event.ID Database        Date.Time   Nearest.town State Latitude Longitude
## 1    20812     Wind 23/11/1975 07:00         SYDNEY   NSW -33.8834  151.2167
## 2    20813  Tornado 02/12/1975 14:00         BARHAM   NSW -35.6333  144.1333
## 3    20814     Wind 09/01/1976 08:50 COFF'S HARBOUR   NSW -30.3167  153.1167
## 4    20815     Hail 16/02/1976 14:00      BANKSTOWN   NSW -33.8834  151.2167
## 5    20816     Rain 25/10/1976 14:00          BOOMI   NSW -28.4333  152.6167
## 6    20817     Hail 08/11/1976 14:00          YOUNG   NSW -34.3167  148.3000
##   AustralianTimeZone         UTCDateTime month_storm year_storm
## 1      Australia/NSW 1975-11-22 20:00:00          11       1975
## 2      Australia/NSW 1975-12-02 03:00:00          12       1975
## 3      Australia/NSW 1976-01-08 21:50:00           1       1976
```

```
## 4      Australia/NSW 1976-02-16 03:00:00           2       1976
## 5      Australia/NSW 1976-10-25 04:00:00          10       1976
## 6      Australia/NSW 1976-11-08 03:00:00          11       1976
```

Question 6

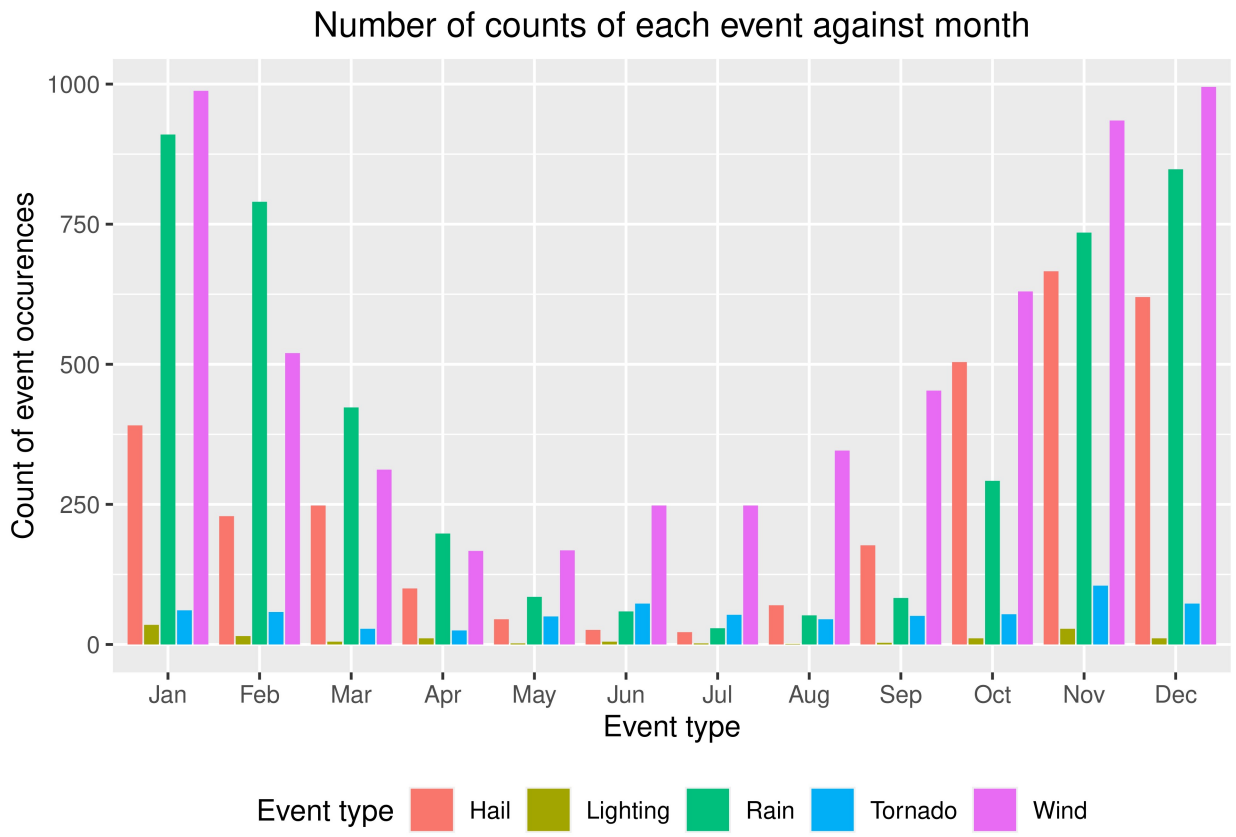
Q6. After discarding Waterspout events there are five types of events left in the data; Rain, Hail, Lighting, Wind, and Tornado.

i) Create a new data frame which contains the total number of counts for each of the above type of events for each of the twelve months over the forty year period.

```
storm_cols_grouped <-storm_events_with_cols %>% group_by(Database, month_storm)
storm_counts_df <- data.frame(count(storm_cols_grouped))

# 'storm_counts_df' dataframe contains the count of each event against each month
```

ii)On a single plot, plot the total number of counts of each event against month. Use the R object month.abb for the labels of the months in the plot.

```
#Labeling months
storm_counts_df$month_storm <- factor(storm_counts_df$month_storm, labels=month.abb)

# Bar plot
ggplot(storm_counts_df,aes(month_storm, n, fill=factor(Database))) +
  geom_bar(position="dodge2", stat="identity") +
  labs(x = "Event type", y="Count of event occurences", fill="Event type",
       title = "Number of counts of each event against month") +
  theme(legend.position = "bottom", plot.title = element_text(hjust = 0.5))
```




```
# Using points
#ggplot(storm_counts_df, aes(month_storm, n, color= Database))+geom_point()
```

Question 7

Q7. From the answer to Question 5, the 6 columns titled Comments, X, X.1, X.2, X.3, X.4 consist of comments.

i) Combine the comments from these columns into a single column, named All.comments.

```
#storm_events_with_cols is the data frame from Q5.

storm_events_with_cols[is.na(storm_events_with_cols)] <- ''

column_names_comments <- colnames(storm_events_with_cols)
list_of_comment_columns_names <- column_names[8:13]

storm_counts_all_comments <- storm_events_with_cols %>% mutate(All.Comments = str_trim(paste(storm_even
                                                                                             storm_events_with_co
                                                                                             storm_events_with_co
```

ii) Select the following columns to keep for further analysis, Event.ID, Database, State, All.comments, and the year variable you created.

```
storm_analysis <-  storm_counts_all_comments %>% select(c(Event.ID, Database, State, All.Comments, year
```

iii) After which you should add the following command to your script: print(sapply(DF, class)) where DF is the name of the data frame.

```
print(sapply(storm_analysis, class))
```

```
##     Event.ID     Database        State All.Comments   year_storm
##    "integer"  "character"  "character"  "character"    "numeric"
```

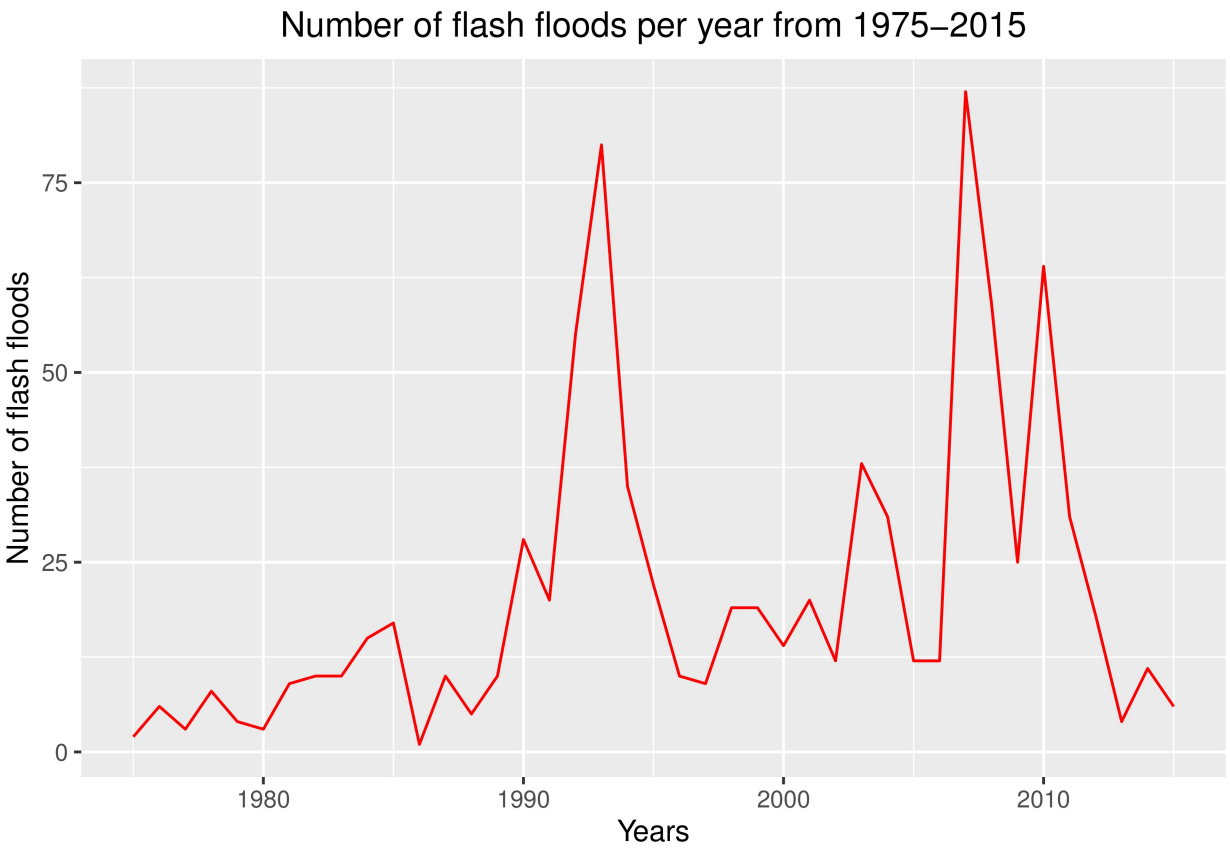
Q8. Now we use the answer to Question 7(ii) for further analysis

i) Create an indicator variable which states whether or not a storm event has resulted in a flash flood. Make sure you sort out all terms relating to flash floods.

```
flash_flood_expr_mul <- "flash(.|\\s*)?flood*" #final

storm_analysis <- storm_analysis %>% mutate(flash_flood_present = str_detect(storm_analysis$All.Comment
```

ii) Print a plot of the number of flash floods per year from 1975-2015. You may need to first create a vector or data frame to contain the number of flash floods per year.

```
storm_analysis_with_floods <- storm_analysis %>% filter(flash_flood_present==TRUE)
storm_analysis_grouped <- storm_analysis_with_floods %>% group_by(year_storm)

flash_flood_counts <- count(storm_analysis_grouped)

ggplot(flash_flood_counts) + geom_line(aes(year_storm, n), color="red") +
  labs(x = "Years", y="Number of flash floods",
       title = "Number of flash floods per year from 1975-2015") +
  theme(plot.title = element_text(hjust = 0.5))
```



Q9. For severe wind events often the wind speed is given. The wind speed is given in knots or km/h.

i) Extract all wind speeds both those in knots and km/h. Hint: Knots can be abbreviated by kts or kt. Also note that wind speed can be a single, double or triple digit number

```
wind_speed_expr <- "(\\d{1,3})\\s?(knot|kt(s)?|km\\s?\\/\\s?h(r)?)"

storm_analysis_speeds <- str_extract_all(storm_analysis,regex(wind_speed_expr))
```

```
## Warning in stri_extract_all_regex(string, pattern, simplify = simplify, :
## argument is not an atomic vector; coercing
```

```
storm_analysis_speeds <- storm_analysis %>% mutate(wind_speed = str_extract(
  storm_analysis$All.Comments, wind_speed_expr))

storm_analysis_speeds <- storm_analysis_speeds %>% filter(!is.na(wind_speed))
```

ii) Convert km/h wind speeds to knots (1 knot = 1.852 km/h) rounding the wind/speed to the nearest knot. Hint: It is helpful to work with a reduced data frame which includes only those observations with a wind speed recorded.

```
knot_to_kmh = 1.852
wind_speed_expr_km <- "(\\d{1,3})\\s?(km\\s?\\/\\s?h(r)?)"
extract_number_expr <- "\\d+"
```

```
get_wind_value <- function (wind_speed) {
  wind_numeric_value = as.numeric(str_extract(wind_speed, extract_number_expr))
  if (str_detect(wind_speed, regex(wind_speed_expr_km, ignore_case = TRUE)))
  {
    return (as.integer(wind_numeric_value/knot_to_kmh))
  }
  else
  {
    return (wind_numeric_value)
  }
}
```

```
storm_analysis_speed_numbers <- storm_analysis_speeds %>% mutate(wind_values = NA)
storm_analysis_speed_numbers$wind_values <- sapply(storm_analysis_speed_numbers$wind_speed, FUN = get_w
```

iii) Print a boxplot of the wind speeds recorded per state.

```
ggplot(storm_analysis_speed_numbers, aes(State, wind_values)) +
  geom_boxplot(position="dodge2", color ="black") +
  labs(x="State", y="Wind speeds", title="Plot of Wind speeds by State") +
  theme(plot.title = element_text(hjust = 0.5))
```