Shark Attack Analysis

Ian Conners, Emma Gillen, Jared Mosley

6/12/2020

Contents

Importing & Initial Cleaning of the Data 2

Cleaning 2
- Fatality Status . . . 2
- Date . . . 2
- Time . . . 3
- Species . . . 5
- Injury Severity . . . 6
- Injury Placemnt . . . 7
- Activity . . . 8

Analysis 9
- Fatalities by Year . . . 9
- Attacks, Fatalities by Species . . . 10
- Month/Time Heatmap . . . 11
- Heatmap by time breaks . . . 12
- USA Heatmap . . . 13
- South Africa and Austrlia heatmaps . . . 14
- Attacks World Map . . . 16
- Attacks US State Map . . . 18
- Number of Attacks Worldwide by Species Bar Chart . . . 19
- US Species Bar Chart . . . 20
- Top Species in the Top US States Bar Chart . . . 20
- Top Species in the Top US States Plot . . . 21

Analysis: SVM 24
- Preparation . . . 24
- Can we predict Fatality based off of all the variables in the j__svm df? . . . 25
- Can we predict Fatality based off of Species, if the shark was Provoked, Activity, and Injury Location? . . . 26
- Can we predict Fatality based off of only the Activity, Species & Injury Location? . . . 26
- Can we predict Fatality based off of only the Activity & Injury Location? . . . 27

Loading Packages

```
library(tidyr)
library(stringr)
library(ggplot2)
library(dplyr)
library(kernlab)
library(gridExtra)
library(reshape2)
```

Importing & Initial Cleaning of the Data

```r
# Read shark attack csv file
sharkAttacks <- read.csv("attacks.csv")

# Starting with 25,723 rows and 24 columns

# Remove unneeded columns at the end of the file
sharkAttacks <- sharkAttacks[,1:15]

# Remove victim names, not necessary for analysis
sharkAttacks <- sharkAttacks[,-9]

# Removing NA years
# There were around 19,000 rows that ONLY had an NA year
# Now have 6300 rows of actual data
sharkAttacks <- sharkAttacks[ ! is.na(sharkAttacks$Year), ]

# Drop any rows that are from before 1900.
# We only want to look at data from that year on
# Now have 5562 rows
sharkAttacks <- sharkAttacks[sharkAttacks$Year >=1900,]
```

Cleaning

Fatality Status

The attacks have a column indicating if the attack invovled a fatality or not. For the most part, it is only Y and N. Drop the columns that don't have a status. Then, convert the Y and N to 0 and 1 for easier analysis.

```r
# Drop any rows that don't have a fatality status.
# Now have 5557 rows
sharkAttacks <- sharkAttacks[sharkAttacks$Fatal..Y.N. == "Y" | sharkAttacks$Fatal..Y.N. == "N",  ]

# Create Fatal column that is numeric
sharkAttacks$Fatal <- as.numeric(recode(sharkAttacks$Fatal..Y.N.,"Y"=1, "N"=0))

# Replace 5's as 0, binary no value
sharkAttacks$Fatal <- replace(sharkAttacks$Fatal, sharkAttacks$Fatal == 5, 0)
# Replace 9's as 1, binary yes value
sharkAttacks$Fatal <- replace(sharkAttacks$Fatal, sharkAttacks$Fatal == 9, 1)
```

Date

Using the case number to break out the month and day of each incident. The case number is formated as YYYY.MM.DD(.a). If multiple attacks occured on a single day they case number ends in a letter.

```r
# Using Case.Number, remove the letters and the periods
sharkAttacks$Dates <- str_replace_all(sharkAttacks$Case.Number,"[A-Z,a-z,.]","")

# Substring to create months
sharkAttacks$Month <- substring(sharkAttacks$Dates, 5,6)

# Remove months that are non existent
sharkAttacks <- sharkAttacks[!grepl("00", sharkAttacks$Month),]
```

```r
# Substring to create day
sharkAttacks$Day <- substring(sharkAttacks$Dates, 7,8)

# Remove days that are non existent
sharkAttacks <- sharkAttacks[!grepl("00", sharkAttacks$Day),]

# Paste the dates together
sharkAttacks$Date <- paste(sharkAttacks$Month,sharkAttacks$Day,sharkAttacks$Year)

# Reformat date as date value
sharkAttacks$Date <- as.Date(sharkAttacks$Date, format="%m %d %Y", tryFormats=c("%m%d%Y"), optional=FALSE)

# There are a couple remaining Dates that are NA. Remove them.
sharkAttacks <- sharkAttacks[ ! is.na(sharkAttacks$Date), ]
```

Time

Some of the times were described by a phrase. Subbing the phrase for an approximate hour for the given phrase. Once formated times are converted into four buckets to account for the lack of accuracy in the estimates. (e.g. Morning -> 06 -> Morning bucket)

```r
# Fix times
# Replace some string times with their representative numeric values
sharkAttacks$Time <- gsub("Morning", 06, sharkAttacks$Time)
sharkAttacks$Time <- gsub("Afternoon", 12, sharkAttacks$Time)
sharkAttacks$Time <- gsub("Sunset", 18, sharkAttacks$Time)
sharkAttacks$Time <- gsub("Noon", 12, sharkAttacks$Time)
sharkAttacks$Time <- gsub("Night", 22, sharkAttacks$Time)
sharkAttacks$Time <- gsub("Nightfall", 22, sharkAttacks$Time)
sharkAttacks$Time <- gsub("Midnight", 00, sharkAttacks$Time)
sharkAttacks$Time <- gsub("Midday", 12, sharkAttacks$Time)
sharkAttacks$Time <- gsub("Evening", 18, sharkAttacks$Time)
sharkAttacks$Time <- gsub("Sunset", 18, sharkAttacks$Time)
sharkAttacks$Time <- gsub("night", 22, sharkAttacks$Time)
sharkAttacks$Time <- gsub("Mid morning", 06, sharkAttacks$Time)
sharkAttacks$Time <- gsub("Mid afternoon", 12, sharkAttacks$Time)
sharkAttacks$Time <- gsub("Mid-morning", 06, sharkAttacks$Time)
sharkAttacks$Time <- gsub("Lunchtime", 12, sharkAttacks$Time)
sharkAttacks$Time <- gsub("Late night", 00, sharkAttacks$Time)
sharkAttacks$Time <- gsub("Late morning", 08, sharkAttacks$Time)
sharkAttacks$Time <- gsub("Late afternoon", 16, sharkAttacks$Time)
sharkAttacks$Time <- gsub("Late afternon", 16, sharkAttacks$Time)
sharkAttacks$Time <- gsub("Just before sundown", 17, sharkAttacks$Time)
sharkAttacks$Time <- gsub("Just before noon", 11, sharkAttacks$Time)
sharkAttacks$Time <- gsub("Just before dawn", 05, sharkAttacks$Time)
sharkAttacks$Time <- gsub("Just after 12h00", 13, sharkAttacks$Time)
sharkAttacks$Time <- gsub("Early morning", 04, sharkAttacks$Time)
sharkAttacks$Time <- gsub("Early afternoon", 12, sharkAttacks$Time)
sharkAttacks$Time <- gsub("Early 6", 06, sharkAttacks$Time)
sharkAttacks$Time <- gsub("Dusk", 18, sharkAttacks$Time)
sharkAttacks$Time <- gsub("Daytime", 02, sharkAttacks$Time)
sharkAttacks$Time <- gsub("Daybreak", 06, sharkAttacks$Time)
sharkAttacks$Time <- gsub("Dawn", 18, sharkAttacks$Time)
sharkAttacks$Time <- gsub("Dark", 22, sharkAttacks$Time)
sharkAttacks$Time <- gsub("Sometime between 06h00 & 08hoo", 06, sharkAttacks$Time)
sharkAttacks$Time <- gsub("Dark", 22, sharkAttacks$Time)
sharkAttacks$Time <- gsub("Shortly before 13h00", 13, sharkAttacks$Time)
```

```r
sharkAttacks$Time <- gsub("Shortly before 12h00", 12, sharkAttacks$Time)
sharkAttacks$Time <- gsub("Shortly after mid22", 23, sharkAttacks$Time)
sharkAttacks$Time <- gsub("Prior to 10h37", 10, sharkAttacks$Time)
sharkAttacks$Time <- gsub("Ship aban-doned at 03h10", 03, sharkAttacks$Time)
sharkAttacks$Time <- gsub("Late 22", 23, sharkAttacks$Time)
sharkAttacks$Time <- gsub("Between 06h00 & 07h20", 06, sharkAttacks$Time)
sharkAttacks$Time <- gsub("Before 07h00", 00, sharkAttacks$Time)
sharkAttacks$Time <- gsub("After noon", 16, sharkAttacks$Time)
sharkAttacks$Time <- gsub("After mid22", 23, sharkAttacks$Time)
sharkAttacks$Time <- gsub("After 18", 18, sharkAttacks$Time)
sharkAttacks$Time <- gsub("After 04h00", 04, sharkAttacks$Time)
sharkAttacks$Time <- gsub(6, 06, sharkAttacks$Time)

# Pull out the first 2 numbers, as some times have various lenghts
sharkAttacks$Time <- substr(sharkAttacks$Time, 1, 2)

# Chain ifelse statements, ensuring all times are numeric from 00-23
sharkAttacks$Time <- ifelse(grepl("00", sharkAttacks$Time, ignore.case = T), "00",
  ifelse(grepl("01", sharkAttacks$Time, ignore.case = T), "01",
  ifelse(grepl("02", sharkAttacks$Time, ignore.case = T), "02",
  ifelse(grepl("03", sharkAttacks$Time, ignore.case = T), "03",
  ifelse(grepl("04", sharkAttacks$Time, ignore.case = T), "04",
  ifelse(grepl("05", sharkAttacks$Time, ignore.case = T), "05",
  ifelse(grepl("06", sharkAttacks$Time, ignore.case = T), "06",
  ifelse(grepl("07", sharkAttacks$Time, ignore.case = T), "07",
  ifelse(grepl("08", sharkAttacks$Time, ignore.case = T), "08",
  ifelse(grepl("09", sharkAttacks$Time, ignore.case = T), "09",
  ifelse(grepl("10", sharkAttacks$Time, ignore.case = T), "10",
  ifelse(grepl("11", sharkAttacks$Time, ignore.case = T), "11",
  ifelse(grepl("12", sharkAttacks$Time, ignore.case = T), "12",
  ifelse(grepl("13", sharkAttacks$Time, ignore.case = T), "13",
  ifelse(grepl("14", sharkAttacks$Time, ignore.case = T), "14",
  ifelse(grepl("15", sharkAttacks$Time, ignore.case = T), "15",
  ifelse(grepl("16", sharkAttacks$Time, ignore.case = T), "16",
  ifelse(grepl("17", sharkAttacks$Time, ignore.case = T), "17",
  ifelse(grepl("18", sharkAttacks$Time, ignore.case = T), "18",
  ifelse(grepl("19", sharkAttacks$Time, ignore.case = T), "19",
  ifelse(grepl("20", sharkAttacks$Time, ignore.case = T), "20",
  ifelse(grepl("21", sharkAttacks$Time, ignore.case = T), "21",
  ifelse(grepl("22", sharkAttacks$Time, ignore.case = T), "22",
  ifelse(grepl("23", sharkAttacks$Time, ignore.case = T),"23",
  ifelse(grepl("24", sharkAttacks$Time, ignore.case = T), "00",
  ifelse(grepl("1", sharkAttacks$Time, ignore.case = T), "01",
  ifelse(grepl("2", sharkAttacks$Time, ignore.case = T), "02",
  ifelse(grepl("3", sharkAttacks$Time, ignore.case = T), "03",
  ifelse(grepl("4", sharkAttacks$Time, ignore.case = T), "04",
  ifelse(grepl("5", sharkAttacks$Time, ignore.case = T), "05",
  ifelse(grepl("6", sharkAttacks$Time, ignore.case = T), "06",
  ifelse(grepl("7", sharkAttacks$Time, ignore.case = T), "07",
  ifelse(grepl("8", sharkAttacks$Time, ignore.case = T), "08",
  ifelse(grepl("9", sharkAttacks$Time, ignore.case = T), "09",
  ifelse(grepl("0", sharkAttacks$Time, ignore.case = T), "00",
    ""))))))))))))))))))))))))))))))))))))


# Create Break in Day
# Begin replacing times with values
# 1 = Morning (>=6, <12)
# 2 = Afternoon (>=12, <17)
# 3 = Evening (>=17,<21)
# 4 = Night (>=21, <6)

# Chain ifelse statements, ensuring all times are in a time break
sharkAttacks$Break <- ifelse(grepl("00", sharkAttacks$Time, ignore.case = T), 4,
  ifelse(grepl("01", sharkAttacks$Time, ignore.case = T), 4,
  ifelse(grepl("02", sharkAttacks$Time, ignore.case = T), 4,
  ifelse(grepl("03", sharkAttacks$Time, ignore.case = T), 4,
  ifelse(grepl("04", sharkAttacks$Time, ignore.case = T), 4,
  ifelse(grepl("05", sharkAttacks$Time, ignore.case = T), 4,
  ifelse(grepl("06", sharkAttacks$Time, ignore.case = T), 1,
  ifelse(grepl("07", sharkAttacks$Time, ignore.case = T), 1,
  ifelse(grepl("08", sharkAttacks$Time, ignore.case = T), 1,
  ifelse(grepl("09", sharkAttacks$Time, ignore.case = T), 1,
  ifelse(grepl("10", sharkAttacks$Time, ignore.case = T), 1,
  ifelse(grepl("11", sharkAttacks$Time, ignore.case = T), 1,
  ifelse(grepl("12", sharkAttacks$Time, ignore.case = T), 2,
  ifelse(grepl("13", sharkAttacks$Time, ignore.case = T), 2,
  ifelse(grepl("14", sharkAttacks$Time, ignore.case = T), 2,
  ifelse(grepl("15", sharkAttacks$Time, ignore.case = T), 2,
  ifelse(grepl("16", sharkAttacks$Time, ignore.case = T), 2,
  ifelse(grepl("17", sharkAttacks$Time, ignore.case = T), 3,
  ifelse(grepl("18", sharkAttacks$Time, ignore.case = T), 3,
  ifelse(grepl("19", sharkAttacks$Time, ignore.case = T), 3,
  ifelse(grepl("20", sharkAttacks$Time, ignore.case = T), 3,
  ifelse(grepl("21", sharkAttacks$Time, ignore.case = T), 4,
  ifelse(grepl("22", sharkAttacks$Time, ignore.case = T),4,
  ifelse(grepl("23", sharkAttacks$Time, ignore.case = T),4,
  ifelse(grepl("24", sharkAttacks$Time, ignore.case = T), 4,
  ifelse(grepl("1", sharkAttacks$Time, ignore.case = T), 4,
  ifelse(grepl("2", sharkAttacks$Time, ignore.case = T), 4,
  ifelse(grepl("3", sharkAttacks$Time, ignore.case = T), 4,
  ifelse(grepl("4", sharkAttacks$Time, ignore.case = T), 4,
  ifelse(grepl("5", sharkAttacks$Time, ignore.case = T), 4,
  ifelse(grepl("6", sharkAttacks$Time, ignore.case = T), 1,
  ifelse(grepl("7", sharkAttacks$Time, ignore.case = T), 1,
  ifelse(grepl("8", sharkAttacks$Time, ignore.case = T), 1,
  ifelse(grepl("9", sharkAttacks$Time, ignore.case = T), 1,
  ifelse(grepl("0", sharkAttacks$Time, ignore.case = T), 4,
    ""))))))))))))))))))))))))))))))))))))
```

Switching to clean shark data to match up two versions of the code.

```r
# save to clean_sharks
clean_shark_data <- sharkAttacks

# re-order data and remove old date columns
clean_shark_data <- clean_shark_data[c("Year","Date","Time","Break","Country","Area","Location","Type","Act
```

Species

Started by listing the 20 most common shark species using head(sort(table(clean_shark_data$Species), decreasing = TRUE), 20). Then, cleaning the column for the species pulling out keywords for these most common sharks, listing only the shark name for any row that had the shark within the string in the original species column. Reference: https://www.sharks.org/species

```r
clean_shark_data$Species <-
  ifelse(grepl("Nurse", clean_shark_data$Species, ignore.case = T), "Nurse",
  ifelse(grepl("Reef", clean_shark_data$Species, ignore.case = T), "Reef",
  ifelse(grepl("Wobbegong", clean_shark_data$Species, ignore.case = T), "Wobbegong",
  ifelse(grepl("Whitetip", clean_shark_data$Species, ignore.case = T), "Whitetip",
       # whitetip needs to be before white
  ifelse(grepl("Bull", clean_shark_data$Species, ignore.case = T), "Bull",
  ifelse(grepl("Lemon", clean_shark_data$Species, ignore.case = T), "Lemon",
  ifelse(grepl("Mako", clean_shark_data$Species, ignore.case = T), "Mako",
  ifelse(grepl("Blacktip", clean_shark_data$Species, ignore.case = T), "Blacktip",
  ifelse(grepl("White", clean_shark_data$Species, ignore.case = T), "White",
  ifelse(grepl("Galapagos", clean_shark_data$Species, ignore.case = T), "Galapagos",
  ifelse(grepl("Cookie", clean_shark_data$Species, ignore.case = T), "Cookie cutter",
  ifelse(grepl("Spinner", clean_shark_data$Species, ignore.case = T), "Spinner",
  ifelse(grepl("Porbeagle", clean_shark_data$Species, ignore.case = T), "Porbeagle",
  ifelse(grepl("Blue", clean_shark_data$Species, ignore.case = T), "Blue",
  ifelse(grepl("Hammerhead", clean_shark_data$Species, ignore.case = T), "Hammerhead",
  ifelse(grepl("Thresher", clean_shark_data$Species, ignore.case = T), "Thresher",
  ifelse(grepl("Dog", clean_shark_data$Species, ignore.case = T), "Dog Shark",
  ifelse(grepl("7-gi", clean_shark_data$Species, ignore.case = T), "Seven Gill",
  ifelse(grepl("seven-gi", clean_shark_data$Species, ignore.case = T), "Seven Gill",
  ifelse(grepl("sevengi", clean_shark_data$Species, ignore.case = T), "Seven Gill",
  ifelse(grepl("Raggedtooth", clean_shark_data$Species, ignore.case = T), "Raggedtooth",
  ifelse(grepl("Tiger", clean_shark_data$Species, ignore.case = T), "Tiger",
  ifelse(grepl("Whaler", clean_shark_data$Species, ignore.case = T), "Whaler",
  ifelse(grepl("Sand", clean_shark_data$Species, ignore.case = T), "Sand",
  ifelse(grepl("Zambesi", clean_shark_data$Species, ignore.case = T), "Zambesi",
  ifelse(grepl("Carpet", clean_shark_data$Species, ignore.case = T), "Carpet",
  ifelse(grepl("Goblin", clean_shark_data$Species, ignore.case = T), "Goblin",
       # everything after this point was helpful for cleaning
  #but only muddies things for analysis
  #ifelse(grepl("'", clean_shark_data$Species, ignore.case = T), "Size Only",
  #ifelse(grepl("m ", clean_shark_data$Species, ignore.case = T), "Size Only",
  #ifelse(grepl("small", clean_shark_data$Species, ignore.case = T), "Size Only",
  #ifelse(grepl("kg", clean_shark_data$Species, ignore.case = T), "Size Only",
  #ifelse(clean_shark_data$Species== "", "Unspecified",
       "Unknown"))))))))))))))))))))))))))))#)))))

# Top species of sharks involved in attacks
head(sort(table(clean_shark_data$Species), decreasing = TRUE), 10)
```

```
##
##  Unknown    White    Tiger     Bull    Nurse Blacktip   Whaler     Reef
##     2857      571      235      162       85       84       68       61
##     Mako     Blue
##       46       45
```

Injury Severity

Looking into the severity of non fatal attacks, giving insight into the reporting of the attacks. A new column is made, with all fatal attacks listed first. Then, major and minor injuries are sepeated. Some attacks had no reported injuries.

```r
clean_shark_data$Attack_Severity <- clean_shark_data$Fatal

clean_shark_data$Attack_Severity <-
```

```
  ifelse(grepl("1", clean_shark_data$Attack_Severity, ignore.case =  T), "Fatal",
  ifelse(grepl("major", clean_shark_data$Injury, ignore.case =  T), "Major Injury",
  ifelse(grepl("sever", clean_shark_data$Injury, ignore.case =  T), "Major Injury",
  ifelse(grepl("serious", clean_shark_data$Injury, ignore.case =  T), "Major Injury",
  ifelse(grepl("minor", clean_shark_data$Injury, ignore.case =  T), "Minor Injury",
  ifelse(grepl("bruise", clean_shark_data$Injury, ignore.case =  T), "Minor Injury",
  ifelse(grepl("no in", clean_shark_data$Injury, ignore.case =  T), "No Injury",
    "")))))))


sort(table(clean_shark_data$Attack_Severity), decreasing =  TRUE)
```

```
##
##                     Fatal    No Injury Major Injury Minor Injury
##         2560          868          587          249          243
```

Injury Placemnt

Pulling out keywords about the location on the body of the victim of the shark attack. Going to be used in the SVM model to predict fatality. Dividing the body into five regions: head, torso, arm, leg, foot, and hand.

```
clean_shark_data$Injury_Placement <-
  ifelse(grepl("head", clean_shark_data$Injury, ignore.case =  T), "Head",
  ifelse(grepl("face", clean_shark_data$Injury, ignore.case =  T), "Head",
  ifelse(grepl("arm", clean_shark_data$Injury, ignore.case =  T), "Arm",
  ifelse(grepl("elbow", clean_shark_data$Injury, ignore.case =  T), "Arm",
  ifelse(grepl("bicep", clean_shark_data$Injury, ignore.case =  T), "Arm",
  ifelse(grepl("leg", clean_shark_data$Injury, ignore.case =  T), "Leg",
  ifelse(grepl("thigh", clean_shark_data$Injury, ignore.case =  T), "Leg",
  ifelse(grepl("calf", clean_shark_data$Injury, ignore.case =  T), "Leg",
  ifelse(grepl("knee", clean_shark_data$Injury, ignore.case =  T), "Leg",
  ifelse(grepl("foot", clean_shark_data$Injury, ignore.case =  T), "Foot",
  ifelse(grepl("feet", clean_shark_data$Injury, ignore.case =  T), "Foot",
  ifelse(grepl("heel", clean_shark_data$Injury, ignore.case =  T), "Foot",
  ifelse(grepl("ankle", clean_shark_data$Injury, ignore.case =  T), "Foot",
  ifelse(grepl("toe", clean_shark_data$Injury, ignore.case =  T), "Foot",
  ifelse(grepl("hand", clean_shark_data$Injury, ignore.case =  T), "Hand",
  ifelse(grepl("wrist", clean_shark_data$Injury, ignore.case =  T), "Hand",
  ifelse(grepl("finger", clean_shark_data$Injury, ignore.case =  T), "Hand",
  ifelse(grepl("torso", clean_shark_data$Injury, ignore.case =  T), "Torso",
  ifelse(grepl("shoulder", clean_shark_data$Injury, ignore.case =  T), "Torso",
  ifelse(grepl("hip", clean_shark_data$Injury, ignore.case =  T), "Torso",
  ifelse(grepl("back", clean_shark_data$Injury, ignore.case =  T), "Torso",
  ifelse(grepl("chest", clean_shark_data$Injury, ignore.case =  T), "Torso",
  ifelse(grepl("abdomen", clean_shark_data$Injury, ignore.case =  T), "Torso",
  ifelse(grepl("no in", clean_shark_data$Injury, ignore.case =  T), "No Injury",
    "")))))))))))))))))))))))))

sort(table(clean_shark_data$Injury_Placement), decreasing =  TRUE)
```

```
##
##       Leg                Foot No Injury       Arm      Hand     Torso      Head
##      1286       918       735       571       497       314       123        63
```

Activity

Cleaning the column for the species pulling out keywords for most common activity using a stepwise approach (similar to the species cleaning). Looked at the most frequent activities that occur, then created a new column that pulled out those words using head(sort(table(clean_shark_data$Activity), decreasing = TRUE), 50). Then, columns that said "Swimming in the ocean" were included in the "swimming" column Next, combined activities that were essentially the same (wading vs walking in the water). Sorted column again by activity, looked for patterns in the missing rows with empty clean_activity. Repeat.

```r
clean_shark_data$Activity <-
ifelse(grepl("skiing", clean_shark_data$Activity, ignore.case = T), "Small Boat",
ifelse(grepl("Body Surf", clean_shark_data$Activity, ignore.case = T), "Small Board",
ifelse(grepl("Surfing", clean_shark_data$Activity, ignore.case = T), "Surfing",
ifelse(grepl("Swimming", clean_shark_data$Activity, ignore.case = T), "Swimming",
ifelse(grepl("Spear", clean_shark_data$Activity, ignore.case = T), "Spearfishing",
       # speak needs to be before fishing
ifelse(grepl("Fishing", clean_shark_data$Activity, ignore.case = T), "Fishing",
ifelse(grepl("Bath", clean_shark_data$Activity, ignore.case = T), "Bathing",
ifelse(grepl("Wading", clean_shark_data$Activity, ignore.case = T), "Wading",
ifelse(grepl("Standing", clean_shark_data$Activity, ignore.case = T), "Wading",
ifelse(grepl("Walking", clean_shark_data$Activity, ignore.case = T), "Wading",
ifelse(grepl("Scuba", clean_shark_data$Activity, ignore.case = T), "Scuba",
ifelse(grepl("Snorkel", clean_shark_data$Activity, ignore.case = T), "Snorkeling",
ifelse(grepl("Div", clean_shark_data$Activity, ignore.case = T), "Diving",
       # Div needs to be after scuba and spear
ifelse(grepl("Body board", clean_shark_data$Activity, ignore.case = T), "Small Board",
ifelse(grepl("Boogie", clean_shark_data$Activity, ignore.case = T), "Small Board",
ifelse(grepl("Kayak", clean_shark_data$Activity, ignore.case = T), "Small Boat",
ifelse(grepl("Canoe", clean_shark_data$Activity, ignore.case = T), "Small Boat",
ifelse(grepl("Treading water", clean_shark_data$Activity, ignore.case = T), "Swimming",
ifelse(grepl("Float", clean_shark_data$Activity, ignore.case = T), "Floating",
ifelse(grepl("swamp", clean_shark_data$Activity, ignore.case = T), "Ship Wreck",
ifelse(grepl("wreck", clean_shark_data$Activity, ignore.case = T), "Ship Wreck",
ifelse(grepl("sink", clean_shark_data$Activity, ignore.case = T), "Ship Wreck",
ifelse(grepl("sunk", clean_shark_data$Activity, ignore.case = T), "Ship Wreck",
ifelse(grepl("capsiz", clean_shark_data$Activity, ignore.case = T), "Ship Wreck",
ifelse(grepl("found", clean_shark_data$Activity, ignore.case = T), "Ship Wreck",
ifelse(grepl("Feed", clean_shark_data$Activity, ignore.case = T), "Feeding Fish/Shark.",
ifelse(grepl("Paddle", clean_shark_data$Activity, ignore.case = T), "Paddle Boarding",
ifelse(grepl("Overboard", clean_shark_data$Activity, ignore.case = T), "Fell into water",
ifelse(grepl("Wash", clean_shark_data$Activity, ignore.case = T), "Bathing",
ifelse(grepl("Fell", clean_shark_data$Activity, ignore.case = T), "Fell into water",
  ""))))))))))))))))))))))))))))))

# Cleaned up the data so "Boating", "Boat", and "Boatomng" are all under the same name
clean_shark_data$Type <- gsub("Boa.*", "Boating", clean_shark_data$Type)

# 271 activities are actually blank
head(sort(table(clean_shark_data$Activity), decreasing = TRUE), 10)
```

```
##
##      Surfing     Swimming                   Fishing Spearfishing       Diving
##          988          832          608          527          332          287
##       Wading  Small Board        Scuba   Snorkeling
##          267          161           87           86
```

Analysis

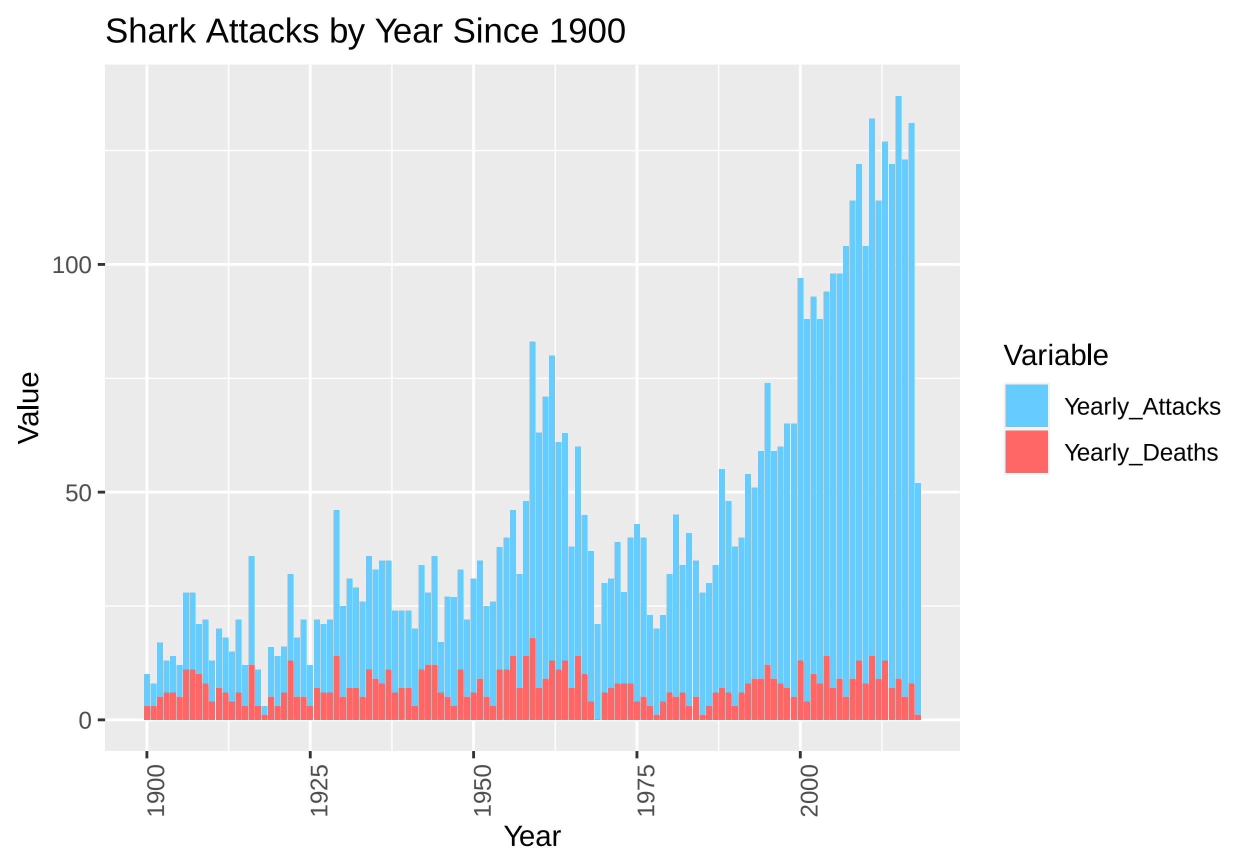
After this point, we will consider the data clean. Any other analysis that needs to be broken down further can be pulled from a copy of this data frame.

Creating a new dataframe

```r
# Create csd, a variant for clean_shark_data for further analysis
j_csd <- clean_shark_data


# Create a dummy variable to count each occurence
j_csd$dummy <- 1
```

Fatalities by Year

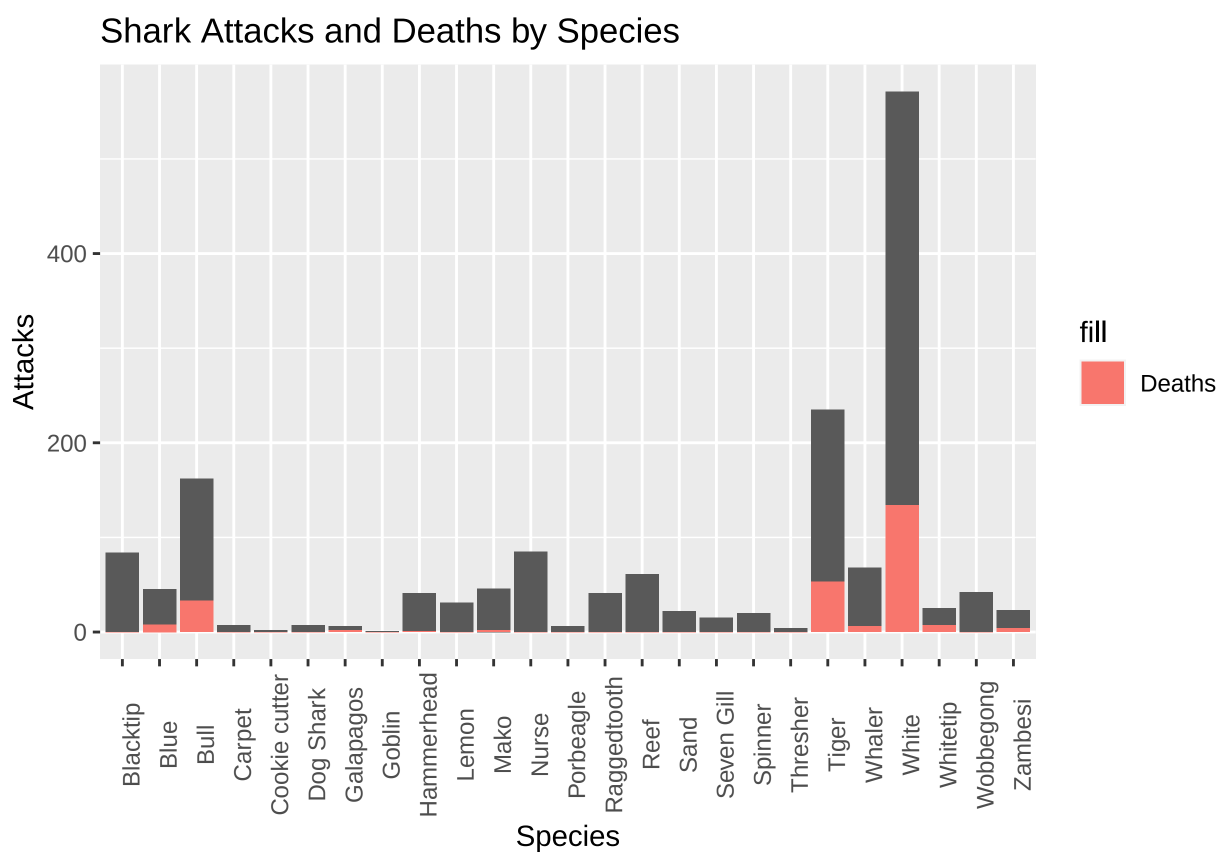
```r
# Find the sum of each attack by year
yearly_attacks <- tapply(j_csd$dummy,j_csd$Year, sum)
# Save to its own data frame
temp <- data.frame(Year = names(yearly_attacks), Yearly_Attacks = yearly_attacks)
# Merge back into csd
j_csd <- merge(j_csd, temp, by.x = "Year", by.y = "Year")

# Tapply to find fatalities/deaths per year
yearly_deaths <- tapply(j_csd$Fatal, j_csd$Year, sum)
# Save to its own data frame
temp <- data.frame(Year = names(yearly_deaths), Yearly_Deaths = yearly_deaths)
# Merge to csd
j_csd <- merge(j_csd, temp, by.x = "Year", by.y = "Year")

# Narrow down focus to allow easy analysis, and make a new df, focusing on yearly attacks and deaths
csd_yearly <- unique(j_csd[c("Year","Yearly_Attacks","Yearly_Deaths")], incomparables = FALSE)
# Melt to clean to better plot format
csd_yearly <- melt(csd_yearly, id.vars="Year")
# Rename Columns
colnames(csd_yearly) <- c("Year", "Variable", "Value")

# Create chart layering shark attacks and deaths by year
ggplot(csd_yearly) + geom_col(aes(x=Year, y=Value, fill=Variable)) + theme(axis.text.x=element_text(angle=9
```



Attacks, Fatalities by Species

```
# Tapply to find attacks per species
species <- tapply(j_csd$dummy, j_csd$Species, sum)
# Save to its own data frame
temp <- data.frame(Species = names(species), Attacks = species)
# Merge to csd
j_csd <- merge(j_csd, temp, by.x = "Species", by.y = "Species")

# Tapply to find fatalities/deaths per species
fatal <- tapply(j_csd$Fatal, j_csd$Species, sum)
# Save to its own data frame
temp <- data.frame(Species = names(fatal), Deaths_Caused = fatal)
# Merge to csd
j_csd <- merge(j_csd, temp, by.x = "Species", by.y = "Species")

# Narrow down focus to allow easy analysis, and make a new df, focusing on attacks and deaths by species
csd_species <- unique(j_csd[c("Species","Attacks","Deaths_Caused")], incomparables=FALSE)
# Remove various unidentifiably species
csd_species <- csd_species[csd_species$Species !="Unknown", ]

# Create percent fatal column
csd_species$Fatality_Percentage <- (csd_species$Deaths_Caused/csd_species$Attacks)*100

# Create chart showing attacks and deaths by species
ggplot(csd_species, aes(x=Species, y=Attacks)) + geom_col(aes(text="Attacks")) + geom_col(aes(y=Deaths_Caus
```

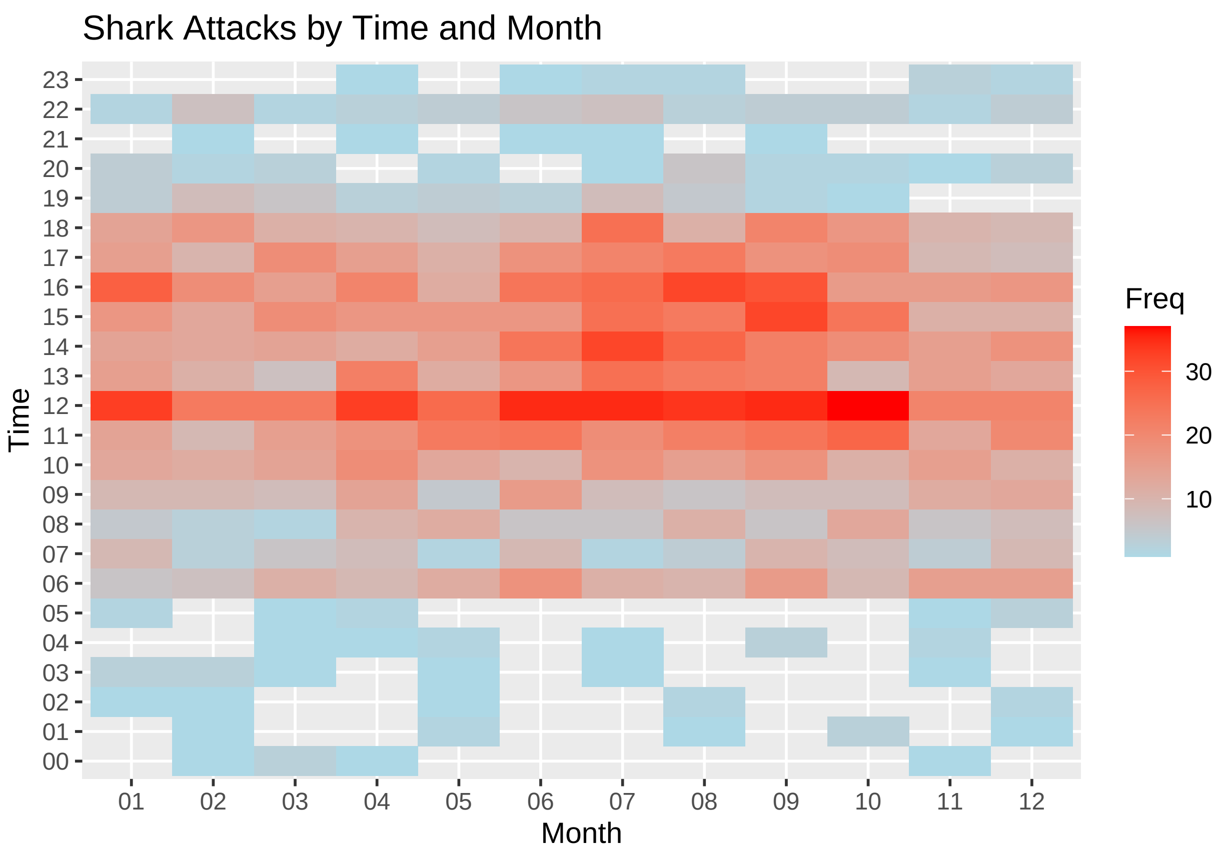
```
## Warning: Ignoring unknown aesthetics: text
```



Month/Time Heatmap

```
# Create data frame for a heatmap of month by hour
j_csd$Month <- format(as.Date(j_csd$Date, format="%d %m %Y"),"%m")
# Pull out only necessary columns
csd_time <- j_csd[c("Month","Time","dummy")]
# Remove time data that does not exist
csd_time <- csd_time[csd_time$Time >= 0, ]
# Omit NA's
csd_time <- csd_time[ ! is.na(csd_time$Time), ]

# Combine Month and Time
csd_time$paste  <- as.numeric(paste0(csd_time$Month,csd_time$Time))
# Create count of all occurences in that timeframe
csd_time <- as.data.frame(table(unlist(csd_time$paste)))

# Turn into format we can manipulate
csd_time$Var1 <- paste0("",csd_time$Var1)
csd_time$Var1 <- as.numeric(csd_time$Var1)
csd_time$Var1 <- ifelse(csd_time$Var1<1000, paste0(0,csd_time$Var1),csd_time$Var1)

# Pull out time data
csd_time$Time <- substring(csd_time$Var1,3,4)
# Pull out month data
csd_time$Month <- substring(csd_time$Var1,1,2)


# Create heat map of attacks by time and month
```

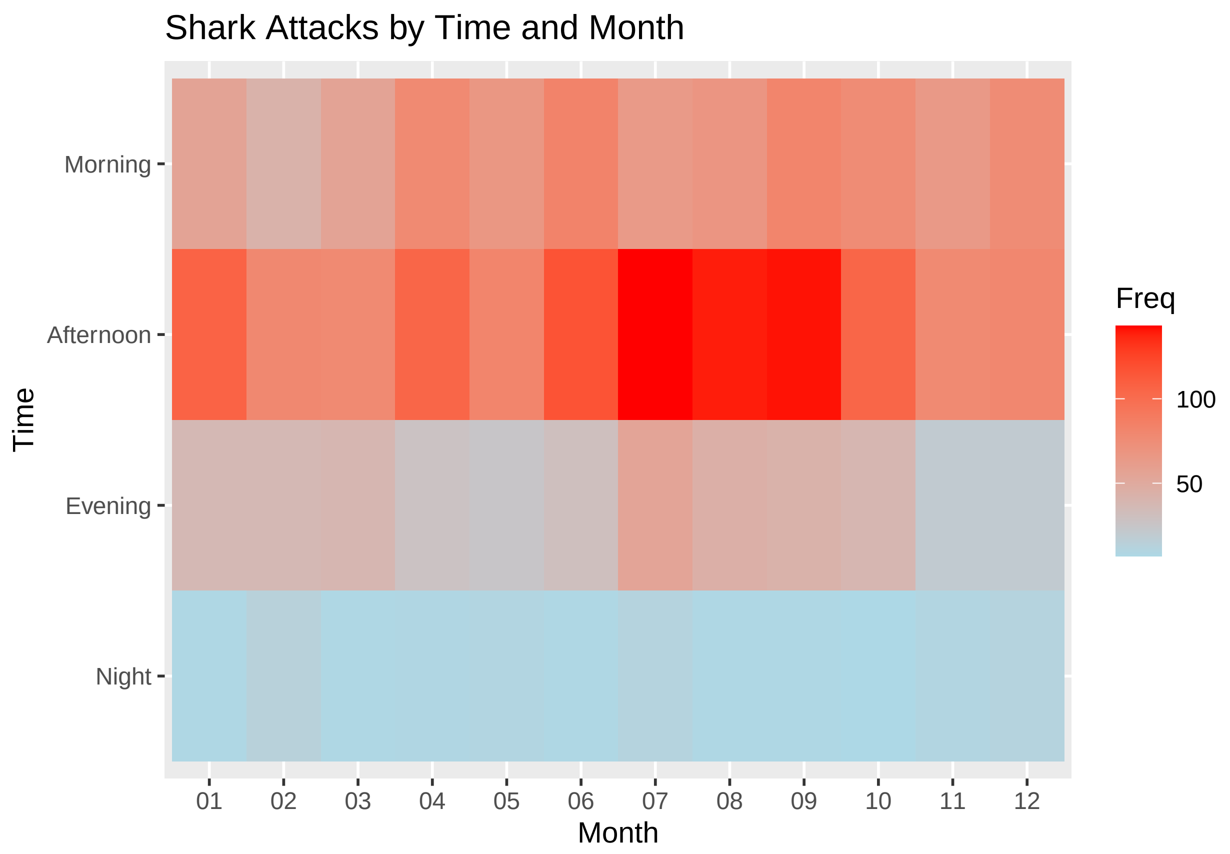
```
ggplot(csd_time, aes(x=Month, y=Time)) + geom_tile(aes(fill=Freq)) + scale_fill_gradient(low="lightblue",hi
```



Heatmap by time breaks

```
# Looking at everything, it may be easier to display times in buckets, as there isn't significant Time data
# Pull out only necessary columns
csd_heat <- j_csd[c("Month","Break","dummy")]
# Remove time data that does not exist
csd_heat <- csd_heat[csd_heat$Break >= 0, ]
# Omit NA's
csd_heat <- csd_heat[ ! is.na(csd_heat$Break), ]

# Combine Month and break
csd_heat$paste  <- as.numeric(paste0(csd_heat$Break,csd_heat$Month))
# Create count of all occurences in that timeframe
csd_heat <- as.data.frame(table(unlist(csd_heat$paste)))


# Turn into format we can manipulate
csd_heat$Var1 <- paste0("",csd_heat$Var1)
csd_heat$Var1 <- as.numeric(csd_heat$Var1)

# Pull out time data
csd_heat$Time <- substring(csd_heat$Var1,0,1)
# Pull out month data
csd_heat$Month <- substring(csd_heat$Var1,2,3)
# Make Time numeric
csd_heat$Time <- as.numeric(csd_heat$Time)
```

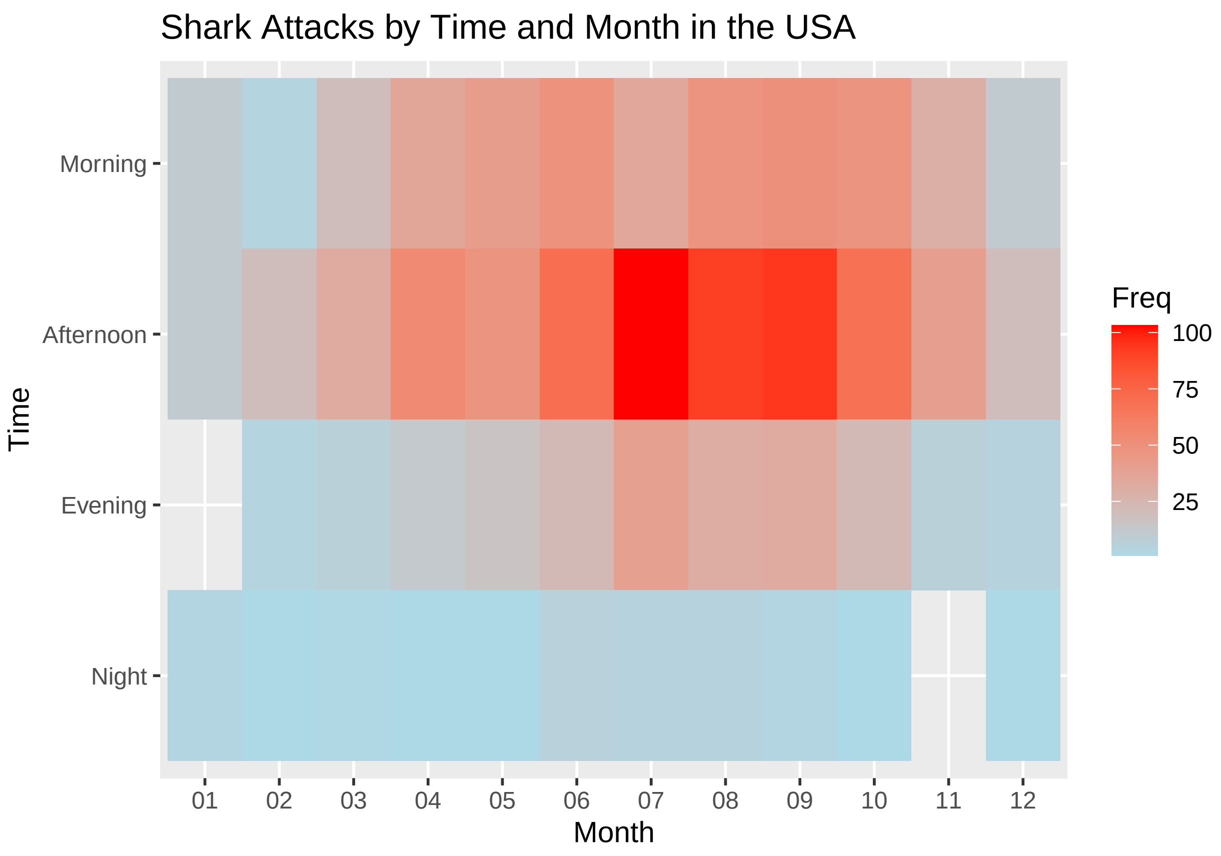
```r
# Rename Breaks
csd_heat$Time <- ifelse(grepl("1", csd_heat$Time, ignore.case = T),"Morning",ifelse(grepl("2", csd_heat$Tim
# Reorder Break for better plot
csd_heat$Time <- ordered(csd_heat$Time, levels=c("Night", "Evening", "Afternoon", "Morning"))

# Create heat map of attacks by time and month
ggplot(csd_heat, aes(x=Month, y=Time)) + geom_tile(aes(fill=Freq)) + scale_fill_gradient(low="lightblue",hi
```



USA Heatmap

```r
# Pull out only necessary columns
csd_heat_USA <- j_csd[c("Month","Break","dummy", "Country")]
# Remove time data that does not exist
csd_heat_USA <- csd_heat_USA[csd_heat_USA$Break >= 0, ]
# Omit NA's
csd_heat_USA <- csd_heat_USA[ ! is.na(csd_heat_USA$Break), ]
# Focus on USA
csd_heat_USA <- csd_heat_USA[csd_heat_USA$Country=="USA",]

# Combine Month and break
csd_heat_USA$paste  <- as.numeric(paste0(csd_heat_USA$Break,csd_heat_USA$Month))
# Create count of all occurences in that timeframe
csd_heat_USA <- as.data.frame(table(unlist(csd_heat_USA$paste)))


# Turn into format we can manipulate
csd_heat_USA$Var1 <- paste0("",csd_heat_USA$Var1)
```

```
csd_heat_USA$Var1 <- as.numeric(csd_heat_USA$Var1)

# Pull out time data
csd_heat_USA$Time <- substring(csd_heat_USA$Var1,0,1)
# Pull out month data
csd_heat_USA$Month <- substring(csd_heat_USA$Var1,2,3)
# Make Time numeric
csd_heat_USA$Time <- as.numeric(csd_heat_USA$Time)

# Rename Breaks
csd_heat_USA$Time <- ifelse(grepl("1", csd_heat_USA$Time, ignore.case = T),"Morning",ifelse(grepl("2", csd_
# Reorder Break for better plot
csd_heat_USA$Time <- ordered(csd_heat_USA$Time, levels=c("Night", "Evening", "Afternoon", "Morning"))

# Create heat map of attacks by time and month
ggplot(csd_heat_USA, aes(x=Month, y=Time)) + geom_tile(aes(fill=Freq)) + scale_fill_gradient(low="lightblue
```

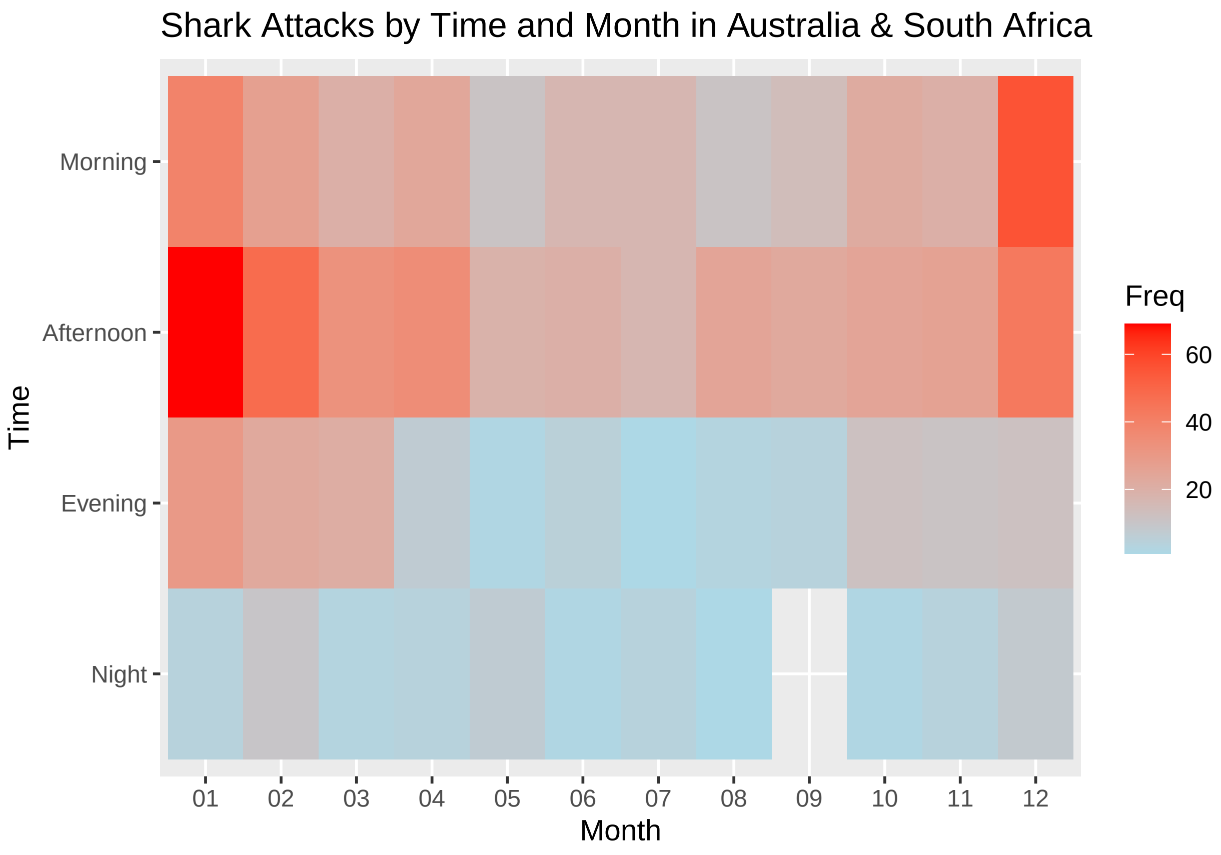


South Africa and Austrlia heatmaps

How does the seasonality of the USA compare to the two other most dangerous locations? South Africa and Australia are both located in the southern hemisphere, so we should expect an inverse on the frequency of attacks by Month.

```
# Pull out only necessary columns
csd_heat_SH <- j_csd[c("Month","Break","dummy", "Country")]
# Remove time data that does not exist
csd_heat_SH <- csd_heat_SH[csd_heat_SH$Break >= 0, ]
# Omit NA's
csd_heat_SH <- csd_heat_SH[ ! is.na(csd_heat_SH$Break), ]
```

```
# Focus on Australia and South Africa
csd_heat_SH <- csd_heat_SH %>% filter(csd_heat_SH$Country=="AUSTRALIA"| csd_heat_SH$Country=="SOUTH AFRICA"

# Combine Month and break
csd_heat_SH$paste  <- as.numeric(paste0(csd_heat_SH$Break,csd_heat_SH$Month))
# Create count of all occurences in that timeframe
csd_heat_SH <- as.data.frame(table(unlist(csd_heat_SH$paste)))


# Turn into format we can manipulate
csd_heat_SH$Var1 <- paste0("",csd_heat_SH$Var1)
csd_heat_SH$Var1 <- as.numeric(csd_heat_SH$Var1)

# Pull out time data
csd_heat_SH$Time <- substring(csd_heat_SH$Var1,0,1)
# Pull out month data
csd_heat_SH$Month <- substring(csd_heat_SH$Var1,2,3)
# Make Time numeric
csd_heat_SH$Time <- as.numeric(csd_heat_SH$Time)

# Rename Breaks
csd_heat_SH$Time <- ifelse(grepl("1", csd_heat_SH$Time, ignore.case = T),"Morning",ifelse(grepl("2", csd_he
# Reorder Break for better plot
csd_heat_SH$Time <- ordered(csd_heat_SH$Time, levels=c("Night", "Evening", "Afternoon", "Morning"))

# Create heat map of attacks by time and month
ggplot(csd_heat_SH, aes(x=Month, y=Time)) + geom_tile(aes(fill=Freq)) + scale_fill_gradient(low="lightblue"
```



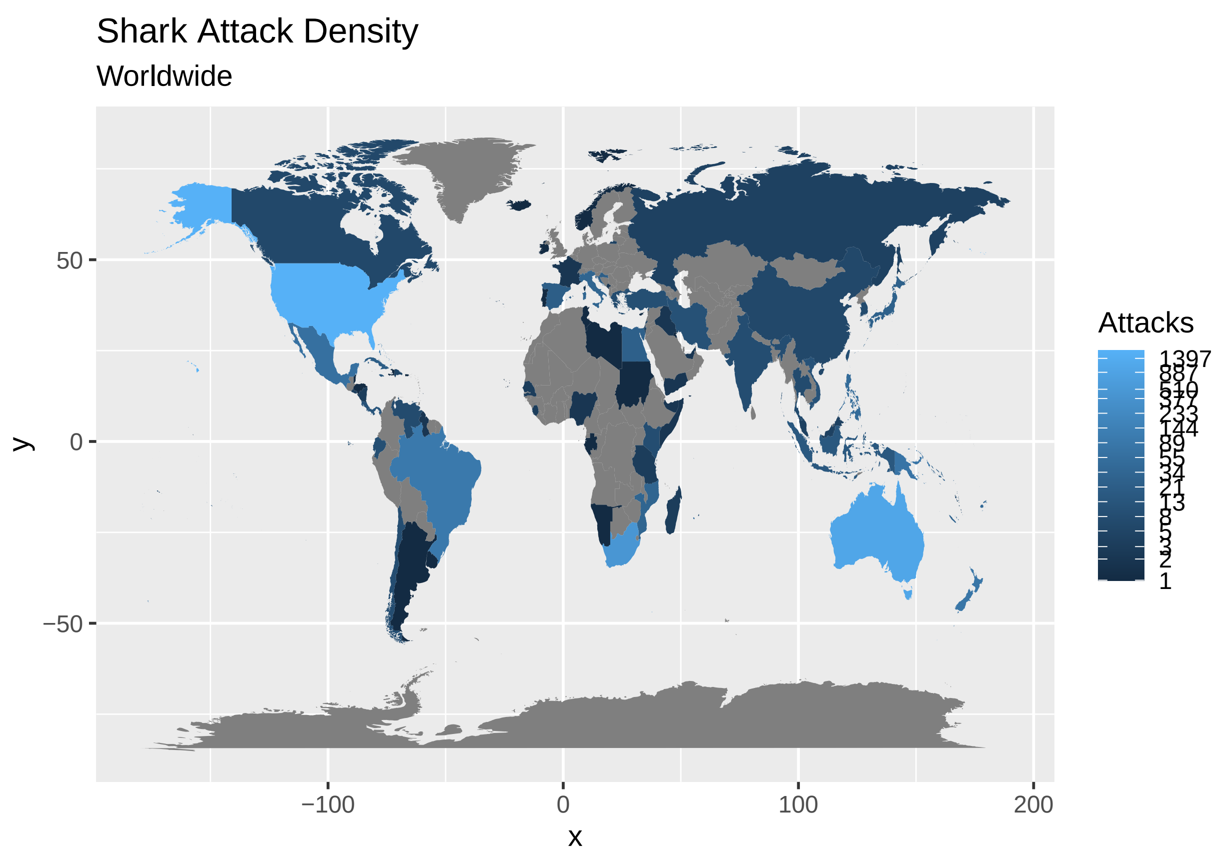
Worldwide Attacks

```r
# making a new version of the data frame to use in this section
e_csd <- clean_shark_data

# Create a dummy variable to count each occurence
e_csd$dummy <- 1

# making all of the countries lower case
# combines rows with mixed case naming
e_csd$Country <- tolower(e_csd$Country)

# use tapply to find the total attacks per country
# attack per country abbreviated to ac
attacks_per_country <- tapply(e_csd$dummy, e_csd$Country, sum)

# create a data frame with the name of the country and the number of attacks
attacks_per_country <- data.frame(region = names(attacks_per_country), attacks = attacks_per_country)

#dropping the NA rows
attacks_per_country <- attacks_per_country[!is.na( attacks_per_country$attacks),]

# getting a world map
world <- map_data("world")
# making the countries all lowercase to match our data
world$region <- tolower(world$region)

# getting a list of all of the counries and putting them in a dataframe
countries <- unique(world$region)
countries <- data.frame(region=countries)

# merging the dataframe so all of the countries we use are "real"
# and ensuring all countries are on the list, even if they have no attacks
attacks_per_country <- merge(attacks_per_country, countries, by="region", all.y=TRUE)

#replace all of the NA attacks with 0
attacks_per_country$attacks <- replace_na(attacks_per_country$attacks, 0)
```

Attacks World Map

```r
# A few outliers skew the fill
# creating breaks to better show the attacks
attack_breaks <- c(0,1, 2, 3, 5,8, 13, 21, 34, 55, 89, 144, 233, 377, 510, 887, 1397)

# generating a map based on the analysis in the above section
attack_map <- ggplot(attacks_per_country, aes(map_id=region))
attack_map <- attack_map + geom_map(map=world, aes(fill=attacks))
attack_map <- attack_map + scale_fill_gradient(name="Attacks", trans="log", breaks = attack_breaks)
attack_map <- attack_map + expand_limits(x=world$long, y=world$lat)
attack_map <- attack_map + labs(title="Shark Attack Density", subtitle="Worldwide")

attack_map
```

```
## Warning: Transformation introduced infinite values in discrete y-axis
```



US Attacks

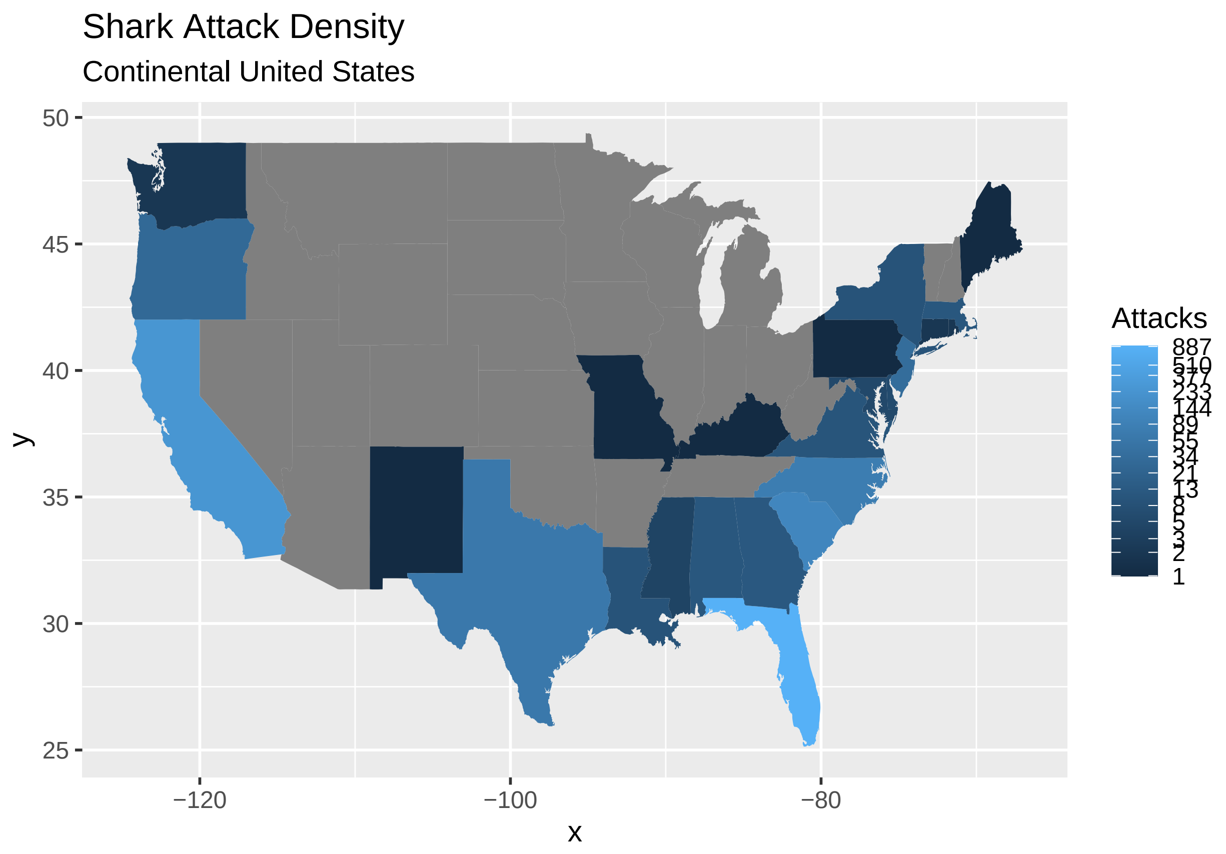
```
# make a data frame of only US attacks
attacks_per_us <- e_csd[e_csd$Country=="usa",]

# making the states all lowercase
attacks_per_us$Area <- tolower(attacks_per_us$Area)

# use tapply to find the total attacks per staate
# attack per country abbreviated to ac
attacks_per_state <- tapply(attacks_per_us$dummy, attacks_per_us$Area, sum)

# create a data frame with the name of the state and the number of attacks
# using this in later plots
attacks_per_us <- data.frame(region = names(attacks_per_state), attacks = attacks_per_state)

#dropping the NA rows
attacks_per_us <- attacks_per_us[!is.na(attacks_per_us$attacks),]

# getting a us map
us <- map_data("state")
# making them all lowercase to match our data
us$region <- tolower(us$region)

# getting a list of all of the states and putting them in a dataframe
states <- unique(us$region)
states <- data.frame(region=states)
```

```r
# merging the dataframe so all of the states we use are "real"
# and ensuring all states are on the list, even if they have no attacks
attacks_per_us <- merge(attacks_per_us, states, by="region", all.y=TRUE)

#replace all of the NA attacks with 0
attacks_per_us$attacks <- replace_na(attacks_per_us$attacks, 0)
```

Attacks US State Map

```r
# generating a map based on the analysis in the above section
us_attack_map <- ggplot(attacks_per_us, aes(map_id=region))
us_attack_map <- us_attack_map + geom_map(map=us, aes(fill=attacks))
# using the same breaks as we used in the world map
us_attack_map <- us_attack_map + scale_fill_gradient(name="Attacks", trans="log", breaks = attack_breaks)
us_attack_map <- us_attack_map + expand_limits(x=us$long, y=us$lat)
us_attack_map <- us_attack_map + labs(title="Shark Attack Density", subtitle="Continental United States")
us_attack_map
```

```
## Warning: Transformation introduced infinite values in discrete y-axis
```



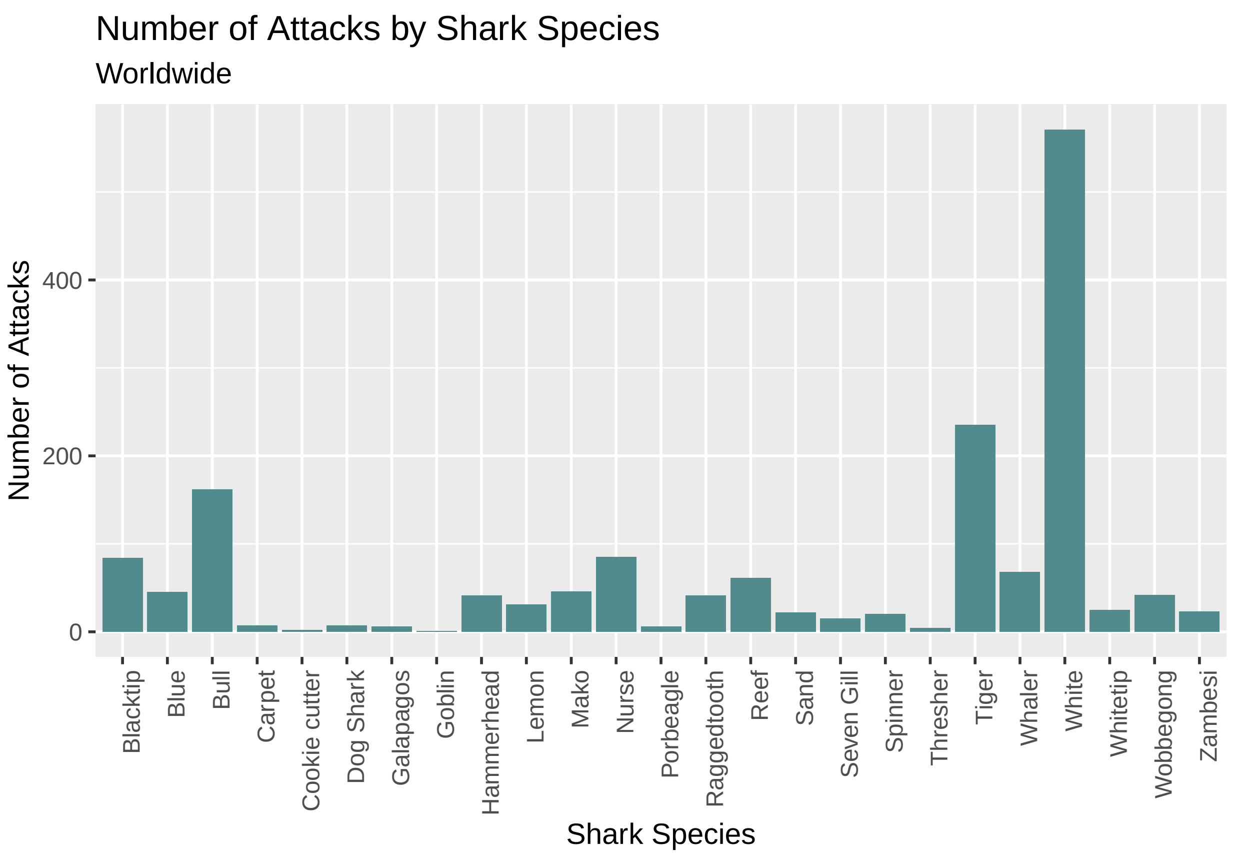
Worldwide Species

```r
# creating a new dataframe requring a species
species_df <- e_csd[e_csd$Species!="Unknown",]

# creating a count of how many attacks per species
attacks_per_species <- tapply(species_df$dummy, species_df$Species, sum)
```

```
# creating a data frame of those counts and the associated species
ww_species_df <- data.frame(species = names(attacks_per_species), attacks = attacks_per_species)
```

Number of Attacks Worldwide by Species Bar Chart

```
# generating a barchart of the above analysis
Species_Bar <- ggplot(ww_species_df, aes(x=species, y=attacks)) + geom_col(fill="darkslategray4")
Species_Bar <- Species_Bar + theme(axis.text.x = element_text(angle = 90, hjust = 1))
Species_Bar <- Species_Bar + labs(x="Shark Species",y="Number of Attacks" ,title="Number of Attacks by Shar

Species_Bar
```

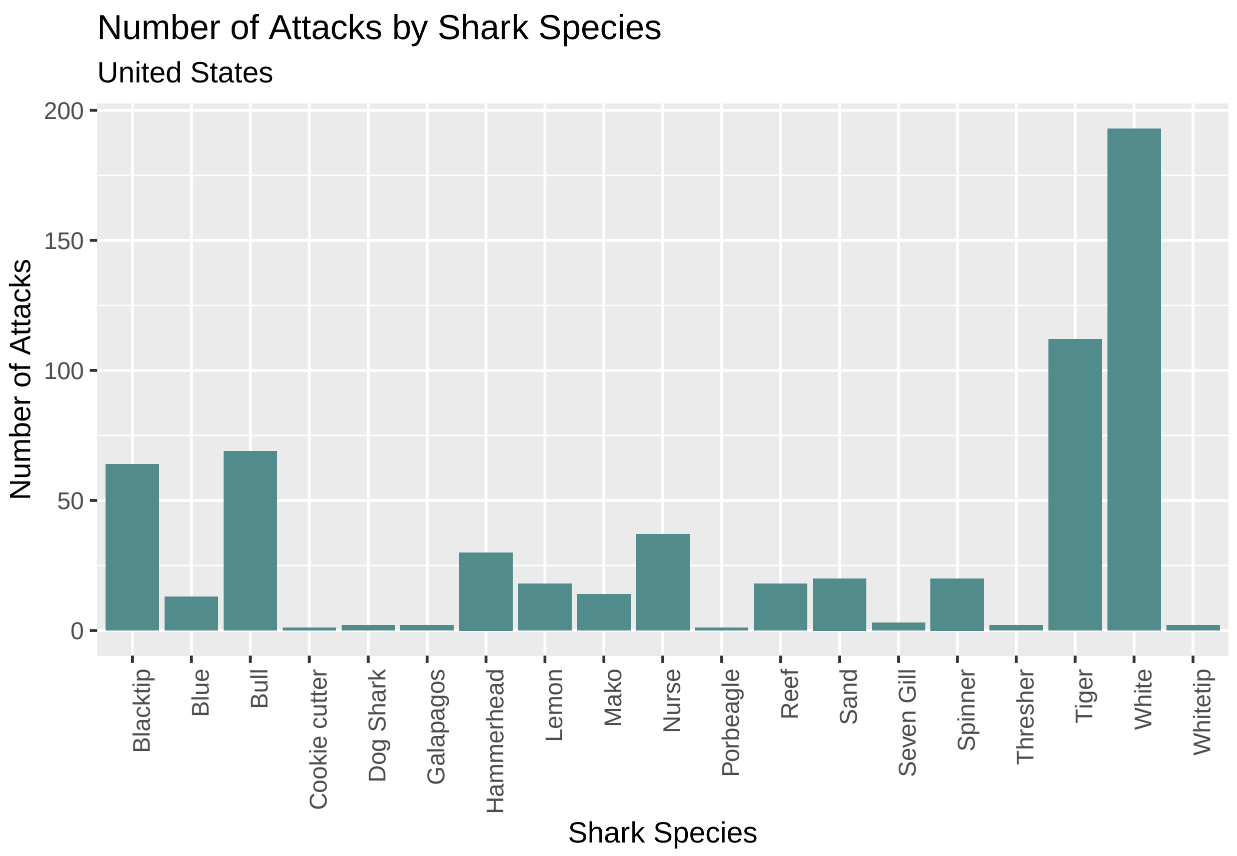


US Species

```
# taking only the attacks from the US out of the species dataframe
us_species_df <- species_df[species_df$Country == "usa",]

# making the states all lower case
us_species_df$Area <- tolower(us_species_df$Area)

# creating a count of how many attacks per species
us_attacks_per_species <- tapply(us_species_df$dummy, us_species_df$Species, sum)

# creating a data frame of those counts and the associated species
us_species_df2 <- data.frame(species = names(us_attacks_per_species), attacks = us_attacks_per_species)
```

US Species Bar Chart

```
US_Species_Bar <- ggplot(us_species_df2, aes(x=species, y=attacks)) + geom_col(fill="darkslategray4")
US_Species_Bar <- US_Species_Bar + theme(axis.text.x = element_text(angle = 90, hjust = 1))
US_Species_Bar <- US_Species_Bar + labs(x="Shark Species",y="Number of Attacks" ,title="Number of Attacks b

US_Species_Bar
```



Top Species in the Top US States Bar Chart

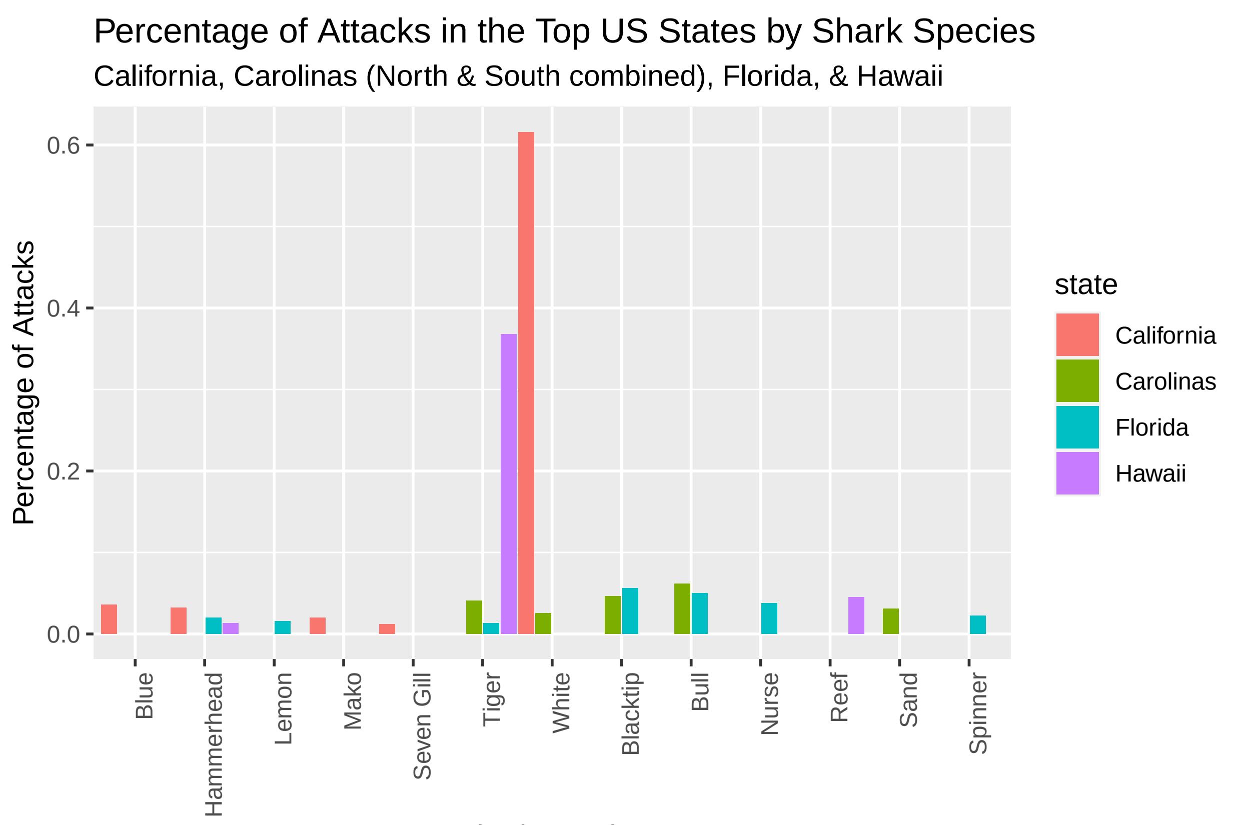
```
# Florida
# Commenting for Florida but the process is the same for the other dataframes below.
# make a new dataframe with only attacks in florida from the us_species_df previously created
fl_species_df <- us_species_df[us_species_df$Area == "florida",]
# create a count of the number of attacks in florida by each species
fl_attacks_per_species <- tapply(fl_species_df$dummy, fl_species_df$Species, sum)
# create a dataframe of the name of the species of shark and the percentage of attacks involving that spec
# using the attacks_per_state variable created earlier
fl_species_df <- data.frame(species = names(fl_attacks_per_species), Florida = (fl_attacks_per_species/atta
# dropping any species that was involved in less than 1% of attacks
fl_species_df <- fl_species_df[fl_species_df$Florida>.01,]

# California
ca_species_df <- us_species_df[us_species_df$Area == "california",]
ca_attacks_per_species <- tapply(ca_species_df$dummy, ca_species_df$Species, sum)
ca_species_df <- data.frame(species = names(ca_attacks_per_species), California = (ca_attacks_per_species/a
ca_species_df <- ca_species_df[ca_species_df$California>.01,]

# Carolinas
```

```r
# Combining North and South Carolina as their coast is continuous and very similar.
car_species_df <- us_species_df[us_species_df$Area == "south carolina" | us_species_df$Area == "north carol
car_attacks_per_species <- tapply(car_species_df$dummy, car_species_df$Species, sum)
car_species_df <- data.frame(species = names(car_attacks_per_species), Carolinas = (car_attacks_per_species
car_species_df <- car_species_df[car_species_df$Carolinas>.01,]

#Hawaii
hi_species_df <- us_species_df[us_species_df$Area == "hawaii",]
hi_attacks_per_species <- tapply(hi_species_df$dummy, hi_species_df$Species, sum)
hi_species_df <- data.frame(species = names(hi_attacks_per_species), Hawaii = (hi_attacks_per_species/attac
hi_species_df <- hi_species_df[hi_species_df$Hawaii>.01,]

# Combing the data frames for the 5 states
top_state_species_df <- merge(ca_species_df, fl_species_df, by="species", all.x = TRUE, all.y = TRUE)
top_state_species_df <- merge(top_state_species_df, car_species_df, by="species", all.x = TRUE, all.y = TRU
top_state_species_df <- merge(top_state_species_df, hi_species_df, by="species", all.x = TRUE, all.y = TRUE

# Combinging the 4 merged columns
# we now have one column named state and one for attacks
top_state_species_df <- tidyr::pivot_longer(top_state_species_df, cols=c("California", "Florida", "Carolina
```

Top Species in the Top US States Plot

```r
Top_State_Species_Plot <- ggplot(top_state_species_df, aes(x=species, y=attacks, fill=state)) + geom_bar(st
Top_State_Species_Plot <- Top_State_Species_Plot + theme(axis.text.x = element_text(angle = 90, hjust = 1))
Top_State_Species_Plot <- Top_State_Species_Plot + labs(x="Shark Species",y="Percentage of Attacks" ,title=

Top_State_Species_Plot
```

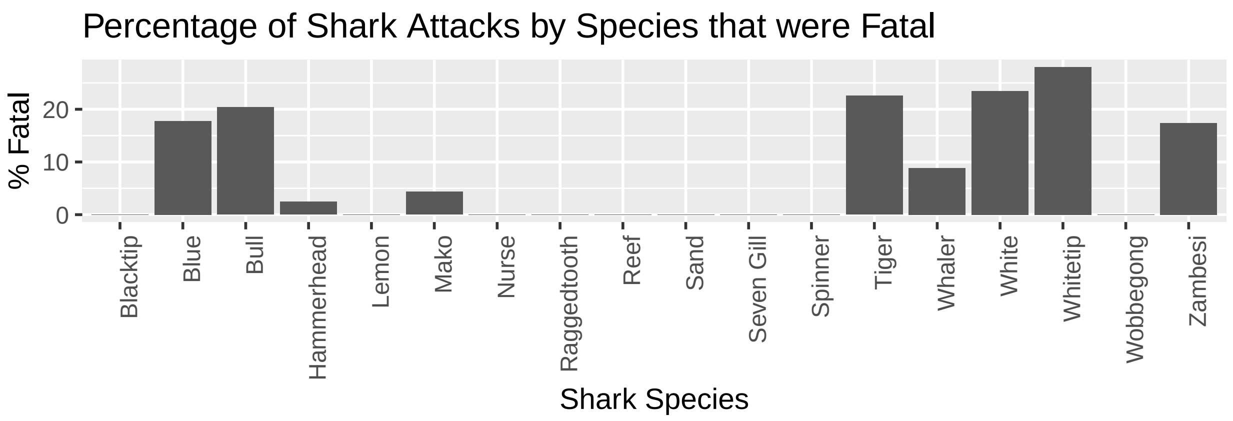
```
## Warning: Removed 32 rows containing missing values (geom_bar).
```



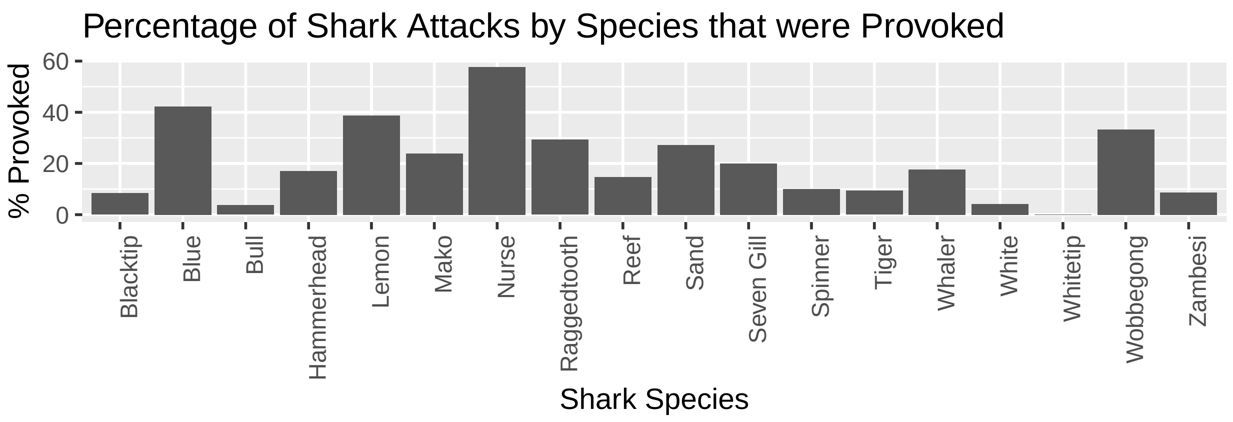
##

Species Provoked and Fatal Attacks Comparison Plots

```
species_df$Provoked <- ifelse(species_df$Type == "Provoked", 1, 0)

provoked_species <- tapply(species_df$Provoked, species_df$Species, sum)

fatal_species <- tapply(species_df$Fatal, species_df$Species, sum)

species_behavior_df <- data.frame(Species = names(fatal_species), Attacks= attacks_per_species, Provoked =

species_behavior_df <- species_behavior_df[species_behavior_df$Attacks>10,]


fatality_plot <- ggplot(species_behavior_df, aes(x=Species, y=Fatal_Percentage)) + geom_col()
fatality_plot <- fatality_plot + theme(axis.text.x = element_text(angle = 90, hjust = 1))
fatality_plot <- fatality_plot + labs(x="Shark Species",y="% Fatal" ,title="Percentage of Shark Attacks by

provoked_plot <- ggplot(species_behavior_df, aes(x=Species, y=Provoked_Percentage)) + geom_col()
provoked_plot <- provoked_plot + theme(axis.text.x = element_text(angle = 90, hjust = 1))
provoked_plot <- provoked_plot + labs(x="Shark Species",y="% Provoked" ,title="Percentage of Shark Attacks

grid.arrange(fatality_plot, provoked_plot)
```



Percentage of Shark Attacks by Species that were Fatal
% Fatal
0
10
20
Blacktip
Blue
Bull
Hammerhead
Lemon
Mako
Nurse
Raggedtooth
Reef
Sand
Seven Gill
Spinner
Tiger
Whaler
White
Whitetip
Wobbegong
Zambesi
Shark Species



Percentage of Shark Attacks by Species that were Provoked
% Provoked
0
20
40
60
Blacktip
Blue
Bull
Hammerhead
Lemon
Mako
Nurse
Raggedtooth
Reef
Sand
Seven Gill
Spinner
Tiger
Whaler
White
Whitetip
Wobbegong
Zambesi
Shark Species

Analysis: SVM

Preparation

```
# Create a new df to begin building SVM
j_svm <- j_csd
# We need to isolate which variables we need to perform the analysis
# We are looking at categorical data

# Remove alternatives in Provoked, to keep Provoked vs Unprovoked
j_svm$Provoked <- ifelse(j_svm$Type == "Provoked", j_svm$Type, "Unprovoked")

# Fix Fatality back to Y/N
j_svm$Fatal <- ifelse(j_svm$Fatal =="1", "Yes", "No")

# Limit Activities to represent the top 6 activities and "other"
j_svm$Activity <- ifelse(j_svm$Activity == "Surfing", j_svm$Activity,ifelse(j_svm$Activity == "Swimming", j


# Do the same, but for country, focusing on top 3 + other
j_svm$Country <- ifelse(j_svm$Country == "USA","USA", ifelse(j_svm$Country == "AUSTRALIA", "AUSTRALIA", ife

# Do the same, but for time breaks
j_svm$Breaks <- ifelse(j_svm$Break == "1", "Morning",ifelse(j_svm$Break == "2", "Afternoon", ifelse(j_svm$B

# Do the same, but for Months
j_svm$Month <- ifelse(j_svm$Month == "01", "January", ifelse(j_svm$Month == "02", "February",ifelse(j_svm$M


# Keep only columns specified
j_svm <- j_svm[c("Fatal", "Month", "Breaks", "Provoked", "Country", "Activity", "Species", "Injury_Placemen

# Turn characters into factors
j_svm <- mutate_if(j_svm, is.character, as.factor)


# Examine the dataset
str(j_svm)
```

```
## 'data.frame':    4507 obs. of  8 variables:
##  $ Fatal           : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Month           : Factor w/ 12 levels "April","August",..: 6 2 10 8 9 9 11 1 2 7 ...
##  $ Breaks          : Factor w/ 5 levels "Afternoon","Evening",..: 5 1 3 1 2 5 3 3 1 1 ...
##  $ Provoked        : Factor w/ 2 levels "Provoked","Unprovoked": 2 2 2 2 1 2 2 2 2 1 ...
##  $ Country         : Factor w/ 4 levels "AUSTRALIA","Other",..: 4 4 4 3 4 1 4 4 4 4 ...
##  $ Activity        : Factor w/ 7 levels "Diving","Fishing",..: 6 5 5 6 2 4 5 5 5 2 ...
##  $ Species         : Factor w/ 26 levels "Blacktip","Blue",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Injury_Placement: Factor w/ 8 levels "","Arm","Foot",..: 8 4 2 3 6 6 3 3 3 5 ...
```

```
summary(j_svm)
```

```
##  Fatal          Month            Breaks           Provoked
##  No :3639   July     : 520   Afternoon:1254   Provoked  : 472
##  Yes: 868   August   : 448   Evening  : 419   Unprovoked:4035
##             September: 442   Morning  : 814
##             January  : 402   Night    : 114
##             June     : 392   Unknown  :1906
##             October  : 358
##             (Other)  :1945
##          Country             Activity        Species      Injury_Placement
##  AUSTRALIA   :1014   Diving      : 287   Unknown :2857   Leg      :1286
##  Other       :1240   Fishing     : 527   White   : 571            : 918
##  SOUTH AFRICA: 454   Other       :1274   Tiger   : 235   Foot     : 735
##  USA         :1799   Spearfishing: 332   Bull    : 162   No Injury: 571
##                      Surfing     : 988   Nurse   :  85   Arm      : 497
##                      Swimming    : 832   Blacktip:  84   Hand     : 314
##                      Wading      : 267   (Other) : 513   (Other)  : 186
```

```
# Create the training data
nrows <-nrow(j_svm) # Identify length
cutpoint <- floor(nrows/3*2) # Find cutpoint at 2/3
rand <- sample(1:nrows) # Create random indices
j_svm.train <- j_svm[rand[1:cutpoint],] # Traning dataset
j_svm.test <- j_svm[rand[(cutpoint+1):nrows],] # Testing dataset
str(j_svm.train) # Examine new train data to ensure it is correct
```

```
## 'data.frame':    3004 obs. of  8 variables:
##  $ Fatal           : Factor w/ 2 levels "No","Yes": 1 1 2 1 1 1 1 2 1 1 ...
##  $ Month           : Factor w/ 12 levels "April","August",..: 1 7 4 3 1 5 5 5 6 4 ...
##  $ Breaks          : Factor w/ 5 levels "Afternoon","Evening",..: 2 3 3 5 4 5 4 5 1 1 ...
##  $ Provoked        : Factor w/ 2 levels "Provoked","Unprovoked": 2 2 2 2 2 2 2 2 2 2 ...
##  $ Country         : Factor w/ 4 levels "AUSTRALIA","Other",..: 4 4 2 1 2 2 4 4 1 4 ...
##  $ Activity        : Factor w/ 7 levels "Diving","Fishing",..: 5 3 6 3 2 3 3 3 5 5 ...
##  $ Species         : Factor w/ 26 levels "Blacktip","Blue",..: 21 23 23 21 20 24 21 21 21 21 ...
##  $ Injury_Placement: Factor w/ 8 levels "","Arm","Foot",..: 2 7 1 6 4 7 7 5 6 3 ...
```

```
str(j_svm.test) #  Examine new test data to ensure it is correct
```

```
## 'data.frame':    1503 obs. of  8 variables:
##  $ Fatal           : Factor w/ 2 levels "No","Yes": 1 2 1 1 2 1 2 1 1 1 ...
##  $ Month           : Factor w/ 12 levels "April","August",..: 12 10 7 6 1 7 10 11 11 2 ...
##  $ Breaks          : Factor w/ 5 levels "Afternoon","Evening",..: 5 4 3 5 1 2 5 5 1 2 ...
##  $ Provoked        : Factor w/ 2 levels "Provoked","Unprovoked": 2 2 2 2 2 2 2 2 2 2 ...
##  $ Country         : Factor w/ 4 levels "AUSTRALIA","Other",..: 4 1 3 4 2 4 2 4 4 4 ...
##  $ Activity        : Factor w/ 7 levels "Diving","Fishing",..: 6 3 5 6 6 3 4 5 5 6 ...
##  $ Species         : Factor w/ 26 levels "Blacktip","Blue",..: 21 21 21 21 21 21 21 21 21 21 ...
##  $ Injury_Placement: Factor w/ 8 levels "","Arm","Foot",..: 4 1 6 6 1 3 2 3 6 5 ...
```

Can we predict Fatality based off of all the variables in the j_svm df?

```
# Run SVM model for Fatality with all other data
model <- ksvm(Fatal ~ ., data=j_svm.train)
model
```

```
## Support Vector Machine object of class "ksvm"
##
## SV type: C-svc  (classification)
##  parameter : cost C = 1
##
## Gaussian Radial Basis kernel function.
##  Hyperparameter : sigma =  0.145454545454545
##
## Number of Support Vectors : 1009
##
## Objective Function Value : -714.0538
## Training error : 0.092876
```

```r
# Predict the model from the training data onto the test data
pred <- predict(model, j_svm.test)
results <- table(pred, j_svm.test$Fatal) # Store results
results # View table of results
```

```
##
## pred    No  Yes
##   No  1148  108
##   Yes   60  187
```

```r
totalCorrect <- results[1,1] + results[2,2] # Store total correct
totalCorrect/nrow(j_svm.test) # Evaluate percentage correct
```

```
## [1] 0.8882236
```

Can we predict Fatality based off of Species, if the shark was Provoked, Activity, and Injury Location?

```r
model1 <- ksvm(Fatal ~ Species + Injury_Placement+ Provoked + Activity, data=j_svm.train)
model1
```

```
## Support Vector Machine object of class "ksvm"
##
## SV type: C-svc  (classification)
##  parameter : cost C = 1
##
## Gaussian Radial Basis kernel function.
##  Hyperparameter : sigma =  0.333333333333333
##
## Number of Support Vectors : 788
##
## Objective Function Value : -663.1433
## Training error : 0.098535
```

```r
# Predict the model from the training data onto the test data
pred1 <- predict(model1, j_svm.test)
results <- table(pred1, j_svm.test$Fatal) # Store results
results # View table of results
```

```
##
## pred1   No  Yes
##   No  1142  108
##   Yes   66  187
```

```r
totalCorrect <- results[1,1] + results[2,2] # Store total correct
totalCorrect/nrow(j_svm.test) # Evaluate percentage correct
```

```
## [1] 0.8842315
```

Can we predict Fatality based off of only the Activity, Species & Injury Location?

```r
model2 <- ksvm(Fatal ~ Injury_Placement + Species + Activity, data=j_svm.train)
model2
```

```
## Support Vector Machine object of class "ksvm"
##
## SV type: C-svc  (classification)
##  parameter : cost C = 1
##
```

```
## Gaussian Radial Basis kernel function.
##  Hyperparameter : sigma =  0.333333333333333
##
## Number of Support Vectors : 770
##
## Objective Function Value : -673.174
## Training error : 0.103862
```

```
# Predict the model from the training data onto the test data
pred2 <- predict(model2, j_svm.test)
results <- table(pred2, j_svm.test$Fatal) # Store results
results # View table of results
```

```
##
## pred2   No  Yes
##   No  1128  107
##   Yes   80  188
```

```
totalCorrect <- results[1,1] + results[2,2] # Store total correct
totalCorrect/nrow(j_svm.test) # Evaluate percentage correct
```

```
## [1] 0.8755822
```

Can we predict Fatality based off of only the Activity & Injury Location?

```
model3 <- ksvm(Fatal ~ Injury_Placement + Activity, data=j_svm.train)
model3
```

```
## Support Vector Machine object of class "ksvm"
##
## SV type: C-svc  (classification)
##  parameter : cost C = 1
##
## Gaussian Radial Basis kernel function.
##  Hyperparameter : sigma =  0.375
##
## Number of Support Vectors : 754
##
## Objective Function Value : -716.4564
## Training error : 0.11751
```

```
# Predict the model from the training data onto the test data
pred3 <- predict(model3, j_svm.test)
results <- table(pred3, j_svm.test$Fatal) # Store results
results # View table of results
```

```
##
## pred3   No  Yes
##   No  1129  122
##   Yes   79  173
```

```
totalCorrect <- results[1,1] + results[2,2] # Store total correct
totalCorrect/nrow(j_svm.test) # Evaluate percentage correct
```

```
## [1] 0.8662675
```