R Notebook

Bibek Sapkota

PART 1: Data Wrangling

Loading Packages and Viewing Data

Task 1: Installing and Loading Packages

```
packages <- c("tidyverse", "dplyr", "lubridate")

for (pkg in packages) {
  if (!requireNamespace(pkg, quietly = TRUE)) {
    install.packages(pkg)
  }
}

library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.0     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(dplyr)
library(lubridate)
```

Task 2: Reading the accidents.csv dataset and printing top 6 data using head() function.

```
df <- read_csv("accidents.csv")
```

```
## Rows: 2069 Columns: 14
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (5): Accident Date, 1st Road Class, Road Surface, Daylight/Dark, Local A...
## dbl (9): Number of Vehicles, Time (24hr), Lighting Conditions, Weather Condi...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
head(df)
```

```
## # A tibble: 6 x 14
##   `Number of Vehicles` `Accident Date` `Time (24hr)` `1st Road Class`
##                  <dbl> <chr>                   <dbl> <chr>
## 1                    2 01/01/2017               2120 U
## 2                    2 04/01/2017               1500 U
## 3                    2 05/01/2017                732 A58
## 4                    2 05/01/2017                930 A646
## 5                    2 14/01/2017                909 U
## 6                    1 15/01/2017               1659 U
## # i 10 more variables: `Road Surface` <chr>, `Lighting Conditions` <dbl>,
## #   `Daylight/Dark` <chr>, `Weather Conditions` <dbl>, `Local Authority` <chr>,
## #   `Type of Vehicle` <dbl>, `Casualty Class` <dbl>, `Casualty Severity` <dbl>,
## #   `Sex of Casualty` <dbl>, `Age of Casualty` <dbl>
```

Task 3: Displaying bottom 6 data of accidents dataset using tail() function.

```
tail(df)
```

```
## # A tibble: 6 x 14
##   `Number of Vehicles` `Accident Date` `Time (24hr)` `1st Road Class`
##                  <dbl> <chr>                   <dbl> <chr>
## 1                    1 29/12/2014                735 6
## 2                    4 30/12/2014               1341 3
## 3                    4 30/12/2014               1341 3
## 4                    1 30/12/2014               1535 6
## 5                    2 31/12/2014               1800 6
## 6                    2 31/12/2014               1800 6
## # i 10 more variables: `Road Surface` <chr>, `Lighting Conditions` <dbl>,
## #   `Daylight/Dark` <chr>, `Weather Conditions` <dbl>, `Local Authority` <chr>,
## #   `Type of Vehicle` <dbl>, `Casualty Class` <dbl>, `Casualty Severity` <dbl>,
## #   `Sex of Casualty` <dbl>, `Age of Casualty` <dbl>
```

Task 4: Displaying dimension of dataset.

```
dim(df)
```

```
## [1] 2069   14
```

Task 5: Using View() function to open the dataframe in RStudio's data viewer.

```
view(df)
```

Task 6: Using str() function to display the structure of the dataset.

```
str(df)
```

```
## spc_tbl_ [2,069 x 14] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ Number of Vehicles : num [1:2069] 2 2 2 2 2 1 1 3 1 1 ...
```

```
##  $ Accident Date      : chr [1:2069] "01/01/2017" "04/01/2017" "05/01/2017" "05/01/2017" ...
##  $ Time (24hr)        : num [1:2069] 2120 1500 732 930 909 ...
##  $ 1st Road Class     : chr [1:2069] "U" "U" "A58" "A646" ...
##  $ Road Surface       : chr [1:2069] "Wet/Damp" "Dry" "Wet/Damp" "Wet/Damp" ...
##  $ Lighting Conditions: num [1:2069] 4 1 4 1 1 4 1 4 1 1 ...
##  $ Daylight/Dark      : chr [1:2069] "Dark" "Daylight" "Dark" "Daylight" ...
##  $ Weather Conditions : num [1:2069] 2 1 1 1 1 1 1 1 1 1 ...
##  $ Local Authority    : chr [1:2069] "Calderdale" "Calderdale" "Calderdale" "Calderdale" ...
##  $ Type of Vehicle    : num [1:2069] 9 9 9 4 9 9 9 9 9 9 ...
##  $ Casualty Class     : num [1:2069] 2 3 1 1 1 3 3 1 3 1 ...
##  $ Casualty Severity  : num [1:2069] 2 2 3 1 3 3 3 3 3 3 ...
##  $ Sex of Casualty    : num [1:2069] 2 2 1 1 1 1 2 2 2 2 ...
##  $ Age of Casualty    : num [1:2069] 16 67 56 20 46 NA 25 50 64 22 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   `Number of Vehicles` = col_double(),
##   ..   `Accident Date` = col_character(),
##   ..   `Time (24hr)` = col_double(),
##   ..   `1st Road Class` = col_character(),
##   ..   `Road Surface` = col_character(),
##   ..   `Lighting Conditions` = col_double(),
##   ..   `Daylight/Dark` = col_character(),
##   ..   `Weather Conditions` = col_double(),
##   ..   `Local Authority` = col_character(),
##   ..   `Type of Vehicle` = col_double(),
##   ..   `Casualty Class` = col_double(),
##   ..   `Casualty Severity` = col_double(),
##   ..   `Sex of Casualty` = col_double(),
##   ..   `Age of Casualty` = col_double()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

Task 7: Using summary() function to display summary statistics of the set.

```
summary(df)
```

```
##  Number of Vehicles Accident Date       Time (24hr)   1st Road Class
##  Min.   :1.000      Length:2069        Min.   :   0   Length:2069
##  1st Qu.:1.000      Class :character   1st Qu.:1045   Class :character
##  Median :2.000      Mode  :character   Median :1500   Mode  :character
##  Mean   :1.906                         Mean   :1405
##  3rd Qu.:2.000                         3rd Qu.:1755
##  Max.   :7.000                         Max.   :2350
##
##  Road Surface       Lighting Conditions Daylight/Dark      Weather Conditions
##  Length:2069        Min.   :1.000       Length:2069        Min.   :1.000
##  Class :character   1st Qu.:1.000       Class :character   1st Qu.:1.000
##  Mode  :character   Median :1.000       Mode  :character   Median :1.000
##                     Mean   :2.015                          Mean   :1.464
##                     3rd Qu.:4.000                          3rd Qu.:1.000
##                     Max.   :7.000                          Max.   :9.000
##
##  Local Authority    Type of Vehicle  Casualty Class  Casualty Severity
```

```
##  Length:2069        Min.   : 1.000   Min.   :1.000   Min.   :1.000
##  Class :character   1st Qu.: 9.000   1st Qu.:1.000   1st Qu.:3.000
##  Mode  :character   Median : 9.000   Median :1.000   Median :3.000
##                     Mean   : 8.917   Mean   :1.591   Mean   :2.831
##                     3rd Qu.: 9.000   3rd Qu.:2.000   3rd Qu.:3.000
##                     Max.   :97.000   Max.   :3.000   Max.   :3.000
##
##  Sex of Casualty Age of Casualty
##  Min.   :1.000   Min.   :  1.00
##  1st Qu.:1.000   1st Qu.: 21.00
##  Median :1.000   Median : 33.00
##  Mean   :1.395   Mean   : 36.21
##  3rd Qu.:2.000   3rd Qu.: 49.00
##  Max.   :2.000   Max.   :115.00
##                  NA's   :19
```

Creating a new dataset df_clean as a copy of df

```
df_clean <- df
```

Treating date and time columns

Task 1: Convert 'Accident Date' column to Date format using lubridate::dmy() and Displaing the structure of df_clean after converting 'Accident Date' column

```
df_clean$'Accident Date' <- dmy(df_clean$'Accident Date')
str(df_clean)
```

```
## spc_tbl_ [2,069 x 14] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ Number of Vehicles : num [1:2069] 2 2 2 2 2 1 1 3 1 1 ...
##  $ Accident Date      : Date[1:2069], format: "2017-01-01" "2017-01-04" ...
##  $ Time (24hr)        : num [1:2069] 2120 1500 732 930 909 ...
##  $ 1st Road Class     : chr [1:2069] "U" "U" "A58" "A646" ...
##  $ Road Surface       : chr [1:2069] "Wet/Damp" "Dry" "Wet/Damp" "Wet/Damp" ...
##  $ Lighting Conditions: num [1:2069] 4 1 4 1 1 4 1 4 1 1 ...
##  $ Daylight/Dark      : chr [1:2069] "Dark" "Daylight" "Dark" "Daylight" ...
##  $ Weather Conditions : num [1:2069] 2 1 1 1 1 1 1 1 1 1 ...
##  $ Local Authority    : chr [1:2069] "Calderdale" "Calderdale" "Calderdale" "Calderdale" ...
##  $ Type of Vehicle    : num [1:2069] 9 9 9 4 9 9 9 9 9 9 ...
##  $ Casualty Class     : num [1:2069] 2 3 1 1 1 3 3 1 3 1 ...
##  $ Casualty Severity  : num [1:2069] 2 2 3 1 3 3 3 3 3 3 ...
##  $ Sex of Casualty    : num [1:2069] 2 2 1 1 1 1 2 2 2 2 ...
##  $ Age of Casualty    : num [1:2069] 16 67 56 20 46 NA 25 50 64 22 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   `Number of Vehicles` = col_double(),
##   ..   `Accident Date` = col_character(),
##   ..   `Time (24hr)` = col_double(),
##   ..   `1st Road Class` = col_character(),
##   ..   `Road Surface` = col_character(),
##   ..   `Lighting Conditions` = col_double(),
```

```
##   ..   `Daylight/Dark` = col_character(),
##   ..   `Weather Conditions` = col_double(),
##   ..   `Local Authority` = col_character(),
##   ..   `Type of Vehicle` = col_double(),
##   ..   `Casualty Class` = col_double(),
##   ..   `Casualty Severity` = col_double(),
##   ..   `Sex of Casualty` = col_double(),
##   ..   `Age of Casualty` = col_double()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

Task 2: Converting numeric time in 24-hour format to a character string in HH:MM format and Displaying the structure of df_clean after converting 'Time (24hr)' column.

```
df_clean$`Time (24hr)` <- format(as.POSIXct(sprintf("%04d", df_clean$`Time (24hr)`),
                                            format="%H%M", tz = "UTC"), format="%H:%M", usetz = FALSE)
str(df_clean)
```

```
## spc_tbl_ [2,069 x 14] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ Number of Vehicles : num [1:2069] 2 2 2 2 2 1 1 3 1 1 ...
##  $ Accident Date      : Date[1:2069], format: "2017-01-01" "2017-01-04" ...
##  $ Time (24hr)        : chr [1:2069] "21:20" "15:00" "07:32" "09:30" ...
##  $ 1st Road Class     : chr [1:2069] "U" "U" "A58" "A646" ...
##  $ Road Surface       : chr [1:2069] "Wet/Damp" "Dry" "Wet/Damp" "Wet/Damp" ...
##  $ Lighting Conditions: num [1:2069] 4 1 4 1 1 4 1 4 1 1 ...
##  $ Daylight/Dark      : chr [1:2069] "Dark" "Daylight" "Dark" "Daylight" ...
##  $ Weather Conditions : num [1:2069] 2 1 1 1 1 1 1 1 1 1 ...
##  $ Local Authority    : chr [1:2069] "Calderdale" "Calderdale" "Calderdale" "Calderdale" ...
##  $ Type of Vehicle    : num [1:2069] 9 9 9 4 9 9 9 9 9 9 ...
##  $ Casualty Class     : num [1:2069] 2 3 1 1 1 3 3 1 3 1 ...
##  $ Casualty Severity  : num [1:2069] 2 2 3 1 3 3 3 3 3 3 ...
##  $ Sex of Casualty    : num [1:2069] 2 2 1 1 1 1 2 2 2 2 ...
##  $ Age of Casualty    : num [1:2069] 16 67 56 20 46 NA 25 50 64 22 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   `Number of Vehicles` = col_double(),
##   ..   `Accident Date` = col_character(),
##   ..   `Time (24hr)` = col_double(),
##   ..   `1st Road Class` = col_character(),
##   ..   `Road Surface` = col_character(),
##   ..   `Lighting Conditions` = col_double(),
##   ..   `Daylight/Dark` = col_character(),
##   ..   `Weather Conditions` = col_double(),
##   ..   `Local Authority` = col_character(),
##   ..   `Type of Vehicle` = col_double(),
##   ..   `Casualty Class` = col_double(),
##   ..   `Casualty Severity` = col_double(),
##   ..   `Sex of Casualty` = col_double(),
##   ..   `Age of Casualty` = col_double()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

Examine data for any missing values

Task 1: Checking for missing values in df_clean and printing total missing values

```
missing_values <- colSums(is.na(df_clean))
print("Columns with Missing Values:",)
```

```
## [1] "Columns with Missing Values:"
```

```
missing_values <- missing_values[missing_values > 0]
print(missing_values)
```

```
##   Daylight/Dark Age of Casualty
##              18              19
```

Task 2: Finding and Printing rows with Missing 'Age of Casualty'

```
missing_age <- df %>% filter(is.na(`Age of Casualty`))
print(missing_age)
```

```
## # A tibble: 19 x 14
##    `Number of Vehicles` `Accident Date` `Time (24hr)` `1st Road Class`
##                   <dbl> <chr>                   <dbl> <chr>
##  1                    1 15/01/2017               1659 U
##  2                    2 27/01/2017               1835 U
##  3                    1 04/02/2017               1730 A646
##  4                    2 26/03/2017               1353 U
##  5                    1 01/05/2017               1454 U
##  6                    2 20/06/2017                752 A58
##  7                    1 24/07/2017               2328 U
##  8                    1 03/10/2017                850 A58
##  9                    1 18/10/2017               1007 U
## 10                    2 20/11/2017               1930 U
## 11                    2 26/02/2016               1340 6
## 12                    1 21/05/2016                210 3
## 13                    4 27/10/2016               1735 1
## 14                    2 23/01/2015               1825 6
## 15                    2 03/08/2015               1403 3
## 16                    1 07/11/2015               2230 6
## 17                    2 30/11/2015               1700 3
## 18                    2 14/01/2014               1350 3
## 19                    1 09/05/2014                824 3
## # i 10 more variables: `Road Surface` <chr>, `Lighting Conditions` <dbl>,
## #   `Daylight/Dark` <chr>, `Weather Conditions` <dbl>, `Local Authority` <chr>,
## #   `Type of Vehicle` <dbl>, `Casualty Class` <dbl>, `Casualty Severity` <dbl>,
## #   `Sex of Casualty` <dbl>, `Age of Casualty` <dbl>
```

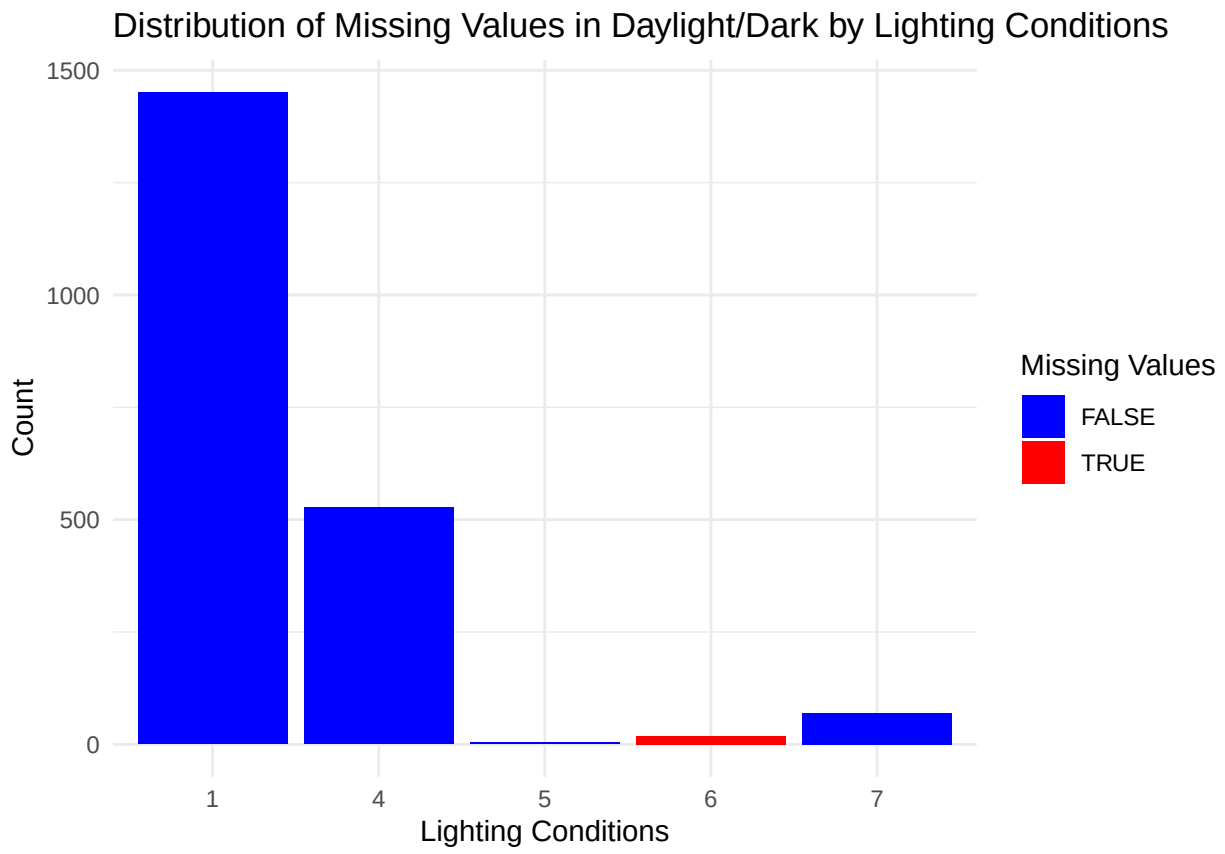
Task 3: Finding and Printing rows with Missing 'Daylight/Dark'

```
missing_day <- df %>% filter(is.na(`Daylight/Dark`))
print(missing_day)
```

```
## # A tibble: 18 x 14
##    `Number of Vehicles` `Accident Date` `Time (24hr)` `1st Road Class`
##                   <dbl> <chr>                   <dbl> <chr>
##  1                    1 18/02/2017               2330 U
##  2                    1 22/02/2017               1939 A681
##  3                    1 01/06/2017                325 U
##  4                    2 18/08/2017               2100 U
##  5                    2 08/11/2017               1651 U
##  6                    1 17/01/2016               2040 6
##  7                    2 10/03/2016                657 6
##  8                    1 07/05/2016               2207 3
##  9                    1 30/03/2015               1955 6
## 10                    1 31/05/2015                150 6
## 11                    1 28/08/2015               2345 6
## 12                    1 28/08/2015               2345 6
## 13                    1 22/11/2015               1136 6
## 14                    1 22/11/2015               1136 6
## 15                    1 09/10/2014               2100 3
## 16                    1 23/11/2014                640 3
## 17                    1 23/11/2014                640 3
## 18                    1 29/12/2014                735 6
## # i 10 more variables: `Road Surface` <chr>, `Lighting Conditions` <dbl>,
## #   `Daylight/Dark` <chr>, `Weather Conditions` <dbl>, `Local Authority` <chr>,
## #   `Type of Vehicle` <dbl>, `Casualty Class` <dbl>, `Casualty Severity` <dbl>,
## #   `Sex of Casualty` <dbl>, `Age of Casualty` <dbl>
```

Task 4: Creating bar plot showing distribution of missing values in 'Daylight/Dark' by 'Lighting Conditions'

```
ggplot(df_clean, aes(x = factor(`Lighting Conditions`), fill = factor(is.na(`Daylight/Dark`)))) +
  geom_bar(position = "stack") +
  labs(title = "Distribution of Missing Values in Daylight/Dark by Lighting Conditions",
       x = "Lighting Conditions",
       y = "Count") +
  scale_fill_manual(values = c("FALSE" = "blue", "TRUE" = "red"), name = "Missing Values") +
  theme_minimal()
```



1. MAR(Missing at Random)

Ans- Missing at Random occurs when the missing of data is not related to the missing values themselves, but it may be related to other observed variables. For Example, In an employee survey, salary information might be missing more frequently for employees with higher job satisfaction levels. The probability of missing salary data depends on job satisfaction but not directly on the actual salary.

2. NMAR(Not Missing at Random)

Ans- Not Missing at Random occurs when the missing of data is not related to the missing values themselves and not related to other observed variables. For Example, In a financial survey, individuals with very high incomes might intentionally choose not to report their income because they are uncomfortable sharing it. The missingness is directly related to the actual income levels.

Checking for inconsistency and Applying values for consistency

Task 1:Print unique values of '1st Road Class'.

```
cat("Unique values of 1st Road Class:\n")
```

```
## Unique values of 1st Road Class:
```

```r
unique(df_clean$`1st Road Class`)
```

```
##  [1] "U"         "A58"       "A646"      "B6138"     "A629"      "A641"
##  [7] "A672"      "A6033"     "A6139"     "A644"      "A62"       "B6114"
## [13] "A6319"     "B6112"     "M62"       "A681"      "B6113"     "A629(M)"
## [19] "A643"      "A6036"     "A6025"     "A647"      "A6026(M)"  "A649"
## [25] "A6026"     "3"         "6"         "1"         "4"         "2"
```

Task 2: Defining and Applying the mapping for standardizing '1st Road Class'.

```r
road_class_mapping <- c("1" = "Motorway","2" = "A(M)","3" = "A","4" = "B","5" = "C","6" = "Unclassified

df_clean <- df_clean %>%
  mutate(`1st Road Class` = map_chr(`1st Road Class`, function(x) {
    ifelse(as.character(x) %in% names(road_class_mapping), road_class_mapping[as.character(x)], as.chara
  }))
```

Task 3:Printing unique values of '1st Road Class'.

```r
cat("Unique values of 1st Road Class:\n")
```

```
## Unique values of 1st Road Class:
```

```r
unique(df_clean$`1st Road Class`)
```

```
##  [1] "U"            "A58"          "A646"         "B6138"        "A629"
##  [6] "A641"         "A672"         "A6033"        "A6139"        "A644"
## [11] "A62"          "B6114"        "A6319"        "B6112"        "M62"
## [16] "A681"         "B6113"        "A629(M)"      "A643"         "A6036"
## [21] "A6025"        "A647"         "A6026(M)"     "A649"         "A6026"
## [26] "A"            "Unclassified" "Motorway"     "B"            "A(M)"
```

Task 4:Standardize '1st Road Class' values (i.e. changing A58 to A, B6138 to B, A6026(M) to A (M)).

```r
df_clean <- df_clean %>%
  mutate(
    `1st Road Class` = case_when(
      `1st Road Class` == "U" ~ "Unclassified",
      grepl("^A\\d+", `1st Road Class`) ~ "A",
      grepl("^A\\(M\\)$", `1st Road Class`) ~ "A(M)",
      grepl("^A\\d+\\(M\\)$", `1st Road Class`) ~ "A(M)",
      grepl("^B\\d+", `1st Road Class`) ~ "B",
      `1st Road Class` == "M62" ~ "Motorway",
      TRUE ~ `1st Road Class`
    )
  )
```

Task 5:Printing unique values of '1st Road Class'.

```r
cat("Unique values of 1st Road Class:\n")
```

```
## Unique values of 1st Road Class:
```

```r
unique(df_clean$`1st Road Class`)
```

```
## [1] "Unclassified" "A"            "B"            "Motorway"     "A(M)"
```

Task 6:Printing unique values of 'Road Surface'.

```r
cat("Unique values of Road Surface:\n")
```

```
## Unique values of Road Surface:
```

```r
unique(df_clean$`Road Surface`)
```

```
##  [1] "Wet/Damp"      "Dry"           "Frost/Ice"     "Ice"
##  [5] "Snow"          "Wet"           "Wet \xa8 Damp" "2"
##  [9] "1"             "3"             "4"             "5"
```

Task 7:Defining a mapping to standardize 'Road Surface' values and Applying the mapping to 'Road Surface' column in df_clean.

```r
road_surface_mapping <- c(
  "Wet/Damp" = "Wet / Damp",
  "Wet" = "Wet / Damp",
  "Frost/Ice" = "Frost / Ice",
  "Ice" = "Frost / Ice",
  "1" = "Dry",
  "2" = "Wet / Damp",
  "3" = "Snow",
  "4" = "Frost / Ice",
  "5" = "Flood (surface water over 3cm deep)",
  "Wet \xa8 Damp" = "Wet / Damp"
)

df_clean <- df_clean %>%
  mutate(`Road Surface` = map_chr(`Road Surface`, function(x) {
    ifelse(as.character(x) %in% names(road_surface_mapping), road_surface_mapping[as.character(x)], as.
  }))
```

Task 8:Printing unique values of 'Road Surface' after applying mapping in Road Surface.

```r
cat("Unique values of Road Surface:\n")
```

```
## Unique values of Road Surface:
```

```r
unique(df_clean$`Road Surface`)
```

```
## [1] "Wet / Damp"                          "Dry"
## [3] "Frost / Ice"                         "Snow"
## [5] "Flood (surface water over 3cm deep)"
```

Task 9:Printing unique values of 'Lighting Conditions'.

```r
cat("Unique values of Lighting Conditions:\n")
```

```
## Unique values of Lighting Conditions:
```

```r
unique(df_clean$`Lighting Conditions`)
```

```
## [1] 4 1 5 7 6
```

Task 10:Defining a mapping to standardize 'Lighting Conditions' values and Applying the mapping to Lighting condition in df_clean.

```r
lighting_conditions_mapping <- c("1" = "Daylight: street lights present",
                                 "2" = "Daylight: no street lighting",
                                 "3" = "Daylight: street lighting unknown",
                                 "4" = "Darkness: street lights present and lit",
                                 "5" = "Darkness: street lights present but unlit",
                                 "6" = "Darkness: no street lighting",
                                 "7" = "Darkness: street lighting unknown")

df_clean <- df_clean %>%
  mutate(`Lighting Conditions` = map_chr(`Lighting Conditions`, function(x) {
    ifelse(as.character(x) %in% names(lighting_conditions_mapping), lighting_conditions_mapping[as.chara
  }))
```

Task 11:Printing unique values of 'Lighting Conditions' after applying mapping .

```r
cat("Unique values of Lighting Conditions:\n")
```

```
## Unique values of Lighting Conditions:
```

```r
unique(df_clean$`Lighting Conditions`)
```

```
## [1] "Darkness: street lights present and lit"
## [2] "Daylight: street lights present"
## [3] "Darkness: street lights present but unlit"
## [4] "Darkness: street lighting unknown"
## [5] "Darkness: no street lighting"
```

Task 12:Printing unique values of 'Weather Conditions'.

```r
cat("Unique values of Weather Conditions:\n")
```

```
## Unique values of Weather Conditions:
```

```r
unique(df_clean$`Weather Conditions`)
```

```
## [1] 2 1 5 7 3 6 4 8 9
```

Task 13:Defining a mapping to standardize 'Weather Conditions' values and Applying the mapping to Weather condition in df_clean.

```r
weather_conditions_mapping <- c("1" = "Fine without high winds",
                                "2" = "Raining without high winds",
                                "3" = "Snowing without high winds",
                                "4" = "Fine with high winds",
                                "5" = "Raining with high winds",
                                "6" = "Snowing with high winds",
                                "7" = "Fog or mist ? if hazard",
                                "8" = "Other",
                                "9" = "Unknown")

df_clean <- df_clean %>%
  mutate(`Weather Conditions` = map_chr(`Weather Conditions`, function(x) {
    ifelse(as.character(x) %in% names(weather_conditions_mapping), weather_conditions_mapping[as.charac
  }))
```

Task 14:Printing unique values of 'Weather Conditions' after applying mapping .

```r
cat("Unique values of Weather Conditions:\n")
```

```
## Unique values of Weather Conditions:
```

```r
unique(df_clean$`Weather Conditions`)
```

```
## [1] "Raining without high winds" "Fine without high winds"
## [3] "Raining with high winds"    "Fog or mist ? if hazard"
## [5] "Snowing without high winds" "Snowing with high winds"
## [7] "Fine with high winds"       "Other"
## [9] "Unknown"
```

Task 15:Printing unique values of 'Casualty Class'.

```r
cat("Unique values of Casualty Class:\n")
```

```
## Unique values of Casualty Class:
```

```r
unique(df_clean$`Casualty Class`)
```

```
## [1] 2 3 1
```

Task 16:Defining a mapping to standardize 'Casualty Class' values and Applying the mapping to Casualty Class in df_clean.

```
casualty_class_mapping <- c("1" = "Driver or rider",
                            "2" = "Vehicle or pillion passenger",
                            "3" = "Pedestrian")

df_clean <- df_clean %>%
  mutate(`Casualty Class` = map_chr(`Casualty Class`, function(x){
    ifelse(as.character(x) %in% names(casualty_class_mapping), casualty_class_mapping[as.character(x)],
  }))
```

Task 17:Printing unique values of 'Casualty Class' after applying mapping .

```
cat("Unique values of Casualty Class:\n")
```

```
## Unique values of Casualty Class:
```

```
unique(df_clean$`Casualty Class`)
```

```
## [1] "Vehicle or pillion passenger" "Pedestrian"
## [3] "Driver or rider"
```

Task 18:Printing unique values of 'Casualty Severity'.

```
cat("Unique values of Casualty Severity:\n")
```

```
## Unique values of Casualty Severity:
```

```
unique(df_clean$`Casualty Severity`)
```

```
## [1] 2 3 1
```

Task 19:Defining a mapping to standardize 'Casualty Severity' values and Applying the mapping to Casualty Severity in df_clean.

```
casualty_severity_mapping <- c("1" = "Fatal",
                               "2" = "Serious",
                               "3" = "Slight")

df_clean <- df_clean %>%
  mutate(`Casualty Severity` = map_chr(`Casualty Severity`, function(x) {
    ifelse(as.character(x) %in% names(casualty_severity_mapping), casualty_severity_mapping[as.characte
  }))
```

Task 20:Printing unique values of 'Casualty Severity' after applying mapping .

```
cat("Unique values of Casualty Severity:\n")
```

```
## Unique values of Casualty Severity:
```

```r
unique(df_clean$`Casualty Severity`)
```

```
## [1] "Serious" "Slight"  "Fatal"
```

Task 21:Printing unique values of 'Type of Vehicle'.

```r
cat("Unique values of Type of Vehicle:\n")
```

```
## Unique values of Type of Vehicle:
```

```r
unique(df_clean$`Type of Vehicle`)
```

```
##  [1]  9  4 19  8  3 21  2 10  5 11  1 97 90 20 23 22 17 18 16
```

Task 22:Defining a mapping to standardize 'Type of Vechicle' values and Applying the mapping in df_clean.

```r
vehicle_type_mapping <- c("1" = "Pedal cycle",
                          "2" = "M/cycle 50cc and under",
                          "3" = "Motorcycle over 50cc and up to 125cc",
                          "4" = "Motorcycle over 125cc and up to 500cc",
                          "5" = "Motorcycle over 500cc",
                          "8" = "Taxi/Private hire car",
                          "9" = "Car",
                          "10" = "Minibus (8 - 16 passenger seats)",
                          "11" = "Bus or coach (17 or more passenger seats)",
                          "14" = "Other motor vehicle",
                          "15" = "Other non-motor vehicle",
                          "16" = "Ridden horse",
                          "17" = "Agricultural vehicle (includes diggers etc.)",
                          "18" = "Tram / Light rail",
                          "19" = "Goods vehicle 3.5 tonnes mgw and under",
                          "20" = "Goods vehicle over 3.5 tonnes and under 7.5 tonnes mgw",
                          "21" = "Goods vehicle 7.5 tonnes mgw and over",
                          "22" = "Mobility Scooter",
                          "90" = "Other Vehicle",
                          "97" = "Motorcycle - Unknown CC")

df_clean <- df_clean %>%
  mutate(`Type of Vehicle` = map_chr(`Type of Vehicle`, function(x) {
    ifelse(as.character(x) %in% names(vehicle_type_mapping), vehicle_type_mapping[as.character(x)], as.
  }))
```

Task 23:Printing unique values of 'Type of Vehicle' after applying mapping .

```r
cat("Unique values of Type of Vehicle:\n")
```

```
## Unique values of Type of Vehicle:
```

```r
unique(df_clean$`Type of Vehicle`)
```

```
##  [1] "Car"
##  [2] "Motorcycle over 125cc and up to 500cc"
##  [3] "Goods vehicle 3.5 tonnes mgw and under"
##  [4] "Taxi/Private hire car"
##  [5] "Motorcycle over 50cc and up to 125cc"
##  [6] "Goods vehicle 7.5 tonnes mgw and over"
##  [7] "M/cycle 50cc and under"
##  [8] "Minibus (8 - 16 passenger seats)"
##  [9] "Motorcycle over 500cc"
## [10] "Bus or coach (17 or more passenger seats)"
## [11] "Pedal cycle"
## [12] "Motorcycle - Unknown CC"
## [13] "Other Vehicle"
## [14] "Goods vehicle over 3.5 tonnes and under 7.5 tonnes mgw"
## [15] "23"
## [16] "Mobility Scooter"
## [17] "Agricultural vehicle (includes diggers etc.)"
## [18] "Tram / Light rail"
## [19] "Ridden horse"
```

Task 24:Filtering and displaying rows with 'Type of Vehicle' value 23

```r
df_clean %>%
  filter(`Type of Vehicle` == "23") %>%
  print()
```

```
## # A tibble: 1 x 14
##   `Number of Vehicles` `Accident Date` `Time (24hr)` `1st Road Class`
##                  <dbl> <date>          <chr>         <chr>
## 1                    2 2016-05-31      17:50         Unclassified
## # i 10 more variables: `Road Surface` <chr>, `Lighting Conditions` <chr>,
## #   `Daylight/Dark` <chr>, `Weather Conditions` <chr>, `Local Authority` <chr>,
## #   `Type of Vehicle` <chr>, `Casualty Class` <chr>, `Casualty Severity` <chr>,
## #   `Sex of Casualty` <dbl>, `Age of Casualty` <dbl>
```

Task 25: Replacing the value "23" in the 'Type of Vehicle' column with "[Not used]" because categories 6 or 7 in this column indicate cases where the specific type of vehicle is not applicable or not used.

```r
df_clean <- df_clean %>%
  mutate(`Type of Vehicle` = recode(`Type of Vehicle`, "23" = "[Not used]"))
```

Task 26:Printing unique values of 'Type of Vehicle' after replacing "23" with "[Not used]" .

```r
cat("Unique values of Casualty Severity:\n")
```

```
## Unique values of Casualty Severity:
```

```r
unique(df_clean$`Type of Vehicle`)
```

```
##  [1] "Car"
##  [2] "Motorcycle over 125cc and up to 500cc"
##  [3] "Goods vehicle 3.5 tonnes mgw and under"
##  [4] "Taxi/Private hire car"
##  [5] "Motorcycle over 50cc and up to 125cc"
##  [6] "Goods vehicle 7.5 tonnes mgw and over"
##  [7] "M/cycle 50cc and under"
##  [8] "Minibus (8 - 16 passenger seats)"
##  [9] "Motorcycle over 500cc"
## [10] "Bus or coach (17 or more passenger seats)"
## [11] "Pedal cycle"
## [12] "Motorcycle - Unknown CC"
## [13] "Other Vehicle"
## [14] "Goods vehicle over 3.5 tonnes and under 7.5 tonnes mgw"
## [15] "[Not used]"
## [16] "Mobility Scooter"
## [17] "Agricultural vehicle (includes diggers etc.)"
## [18] "Tram / Light rail"
## [19] "Ridden horse"
```

Task 27:Printing unique values of 'Sex of Casualty'.

```r
cat("Unique values of Sex of Casualty:\n")
```

```
## Unique values of Sex of Casualty:
```

```r
unique(df_clean$'Sex of Casualty')
```

```
## [1] 2 1
```

Task 28:Defining a mapping to standardize 'Sex of Casualty' values and Applying the mapping in df_clean.

```r
sex_of_casualty_mapping <- c("1" = "Male",
                             "2" = "Female")

df_clean <- df_clean %>%
  mutate(`Sex of Casualty` = map_chr(`Sex of Casualty`, function(x) {
    ifelse(as.character(x) %in% names(sex_of_casualty_mapping), sex_of_casualty_mapping[as.character(x)]
  }))
```

Task 29:Printing unique values of 'Sex of Casualty' after applying mapping .

```r
cat("Unique values of Sex of Casualty:\n")
```

```
## Unique values of Sex of Casualty:
```

```r
unique(df_clean$'Sex of Casualty')
```

```
## [1] "Female" "Male"
```

Task 27:Printing unique values of 'Local Authority'.

```r
cat("Unique values of Local Authority:\n")
```

```
## Unique values of Local Authority:
```

```r
unique(df_clean$'Local Authority')
```

```
## [1] "Calderdale"
```

Task 28: Removing 'Local Authority' column beacause it has the same value for all rows and doesnot provide useful information. Removing 'Local Authority; column can save memory and computation time

```r
df_clean <- df_clean %>%
  select(-`Local Authority`)
```

Task 29: Viewing 'df_clean' dataset after applying all the mapping.

```r
view(df_clean)
```

Handaling missing Values

Task 1:Finding and Displaying e rows from the df_clean dataframe where the 'Daylight/Dark' column contains missing values (NA).

```r
missing_daylight_dark <- df_clean[is.na(df_clean$`Daylight/Dark`), ]
print(missing_daylight_dark)
```

```
## # A tibble: 18 x 13
##    `Number of Vehicles` `Accident Date` `Time (24hr)` `1st Road Class`
##                   <dbl> <date>          <chr>         <chr>
##  1                    1 2017-02-18      23:30         Unclassified
##  2                    1 2017-02-22      19:39         A
##  3                    1 2017-06-01      03:25         Unclassified
##  4                    2 2017-08-18      21:00         Unclassified
##  5                    2 2017-11-08      16:51         Unclassified
##  6                    1 2016-01-17      20:40         Unclassified
##  7                    2 2016-03-10      06:57         Unclassified
##  8                    1 2016-05-07      22:07         A
##  9                    1 2015-03-30      19:55         Unclassified
## 10                    1 2015-05-31      01:50         Unclassified
## 11                    1 2015-08-28      23:45         Unclassified
## 12                    1 2015-08-28      23:45         Unclassified
## 13                    1 2015-11-22      11:36         Unclassified
## 14                    1 2015-11-22      11:36         Unclassified
```

```
## 15                    1 2014-10-09      21:00         A
## 16                    1 2014-11-23      06:40         A
## 17                    1 2014-11-23      06:40         A
## 18                    1 2014-12-29      07:35         Unclassified
## # i 9 more variables: 'Road Surface' <chr>, 'Lighting Conditions' <chr>,
## #   'Daylight/Dark' <chr>, 'Weather Conditions' <chr>, 'Type of Vehicle' <chr>,
## #   'Casualty Class' <chr>, 'Casualty Severity' <chr>, 'Sex of Casualty' <chr>,
## #   'Age of Casualty' <dbl>
```

```
str(df_clean)
```

```
## tibble [2,069 x 13] (S3: tbl_df/tbl/data.frame)
##  $ Number of Vehicles : num [1:2069] 2 2 2 2 2 1 1 3 1 1 ...
##  $ Accident Date      : Date[1:2069], format: "2017-01-01" "2017-01-04" ...
##  $ Time (24hr)        : chr [1:2069] "21:20" "15:00" "07:32" "09:30" ...
##  $ 1st Road Class     : chr [1:2069] "Unclassified" "Unclassified" "A" "A" ...
##  $ Road Surface       : chr [1:2069] "Wet / Damp" "Dry" "Wet / Damp" "Wet / Damp" ...
##  $ Lighting Conditions: chr [1:2069] "Darkness: street lights present and lit" "Daylight: street ligh
##  $ Daylight/Dark      : chr [1:2069] "Dark" "Daylight" "Dark" "Daylight" ...
##  $ Weather Conditions : chr [1:2069] "Raining without high winds" "Fine without high winds" "Fine wi
##  $ Type of Vehicle    : chr [1:2069] "Car" "Car" "Car" "Motorcycle over 125cc and up to 500cc" ...
##  $ Casualty Class     : chr [1:2069] "Vehicle or pillion passenger" "Pedestrian" "Driver or rider" "
##  $ Casualty Severity  : chr [1:2069] "Serious" "Serious" "Slight" "Fatal" ...
##  $ Sex of Casualty    : chr [1:2069] "Female" "Female" "Male" "Male" ...
##  $ Age of Casualty    : num [1:2069] 16 67 56 20 46 NA 25 50 64 22 ...
```

Task 2: Filling missing values on the Daylight/Dark column based on the Lighting Condition

```
lighting_mapping <- c(
  "Darkness: no street lighting" = "Dark"
)

df_clean <- df_clean %>%
  mutate(`Daylight/Dark` = ifelse(is.na(`Daylight/Dark`), lighting_mapping[`Lighting Conditions`], `Dayl
```

Task 3: Displaying total null values in Dayligh/Dark column after Filling it.

```
missing_daylight_dark <- df_clean[is.na(df_clean$`Daylight/Dark`), ]
print(missing_daylight_dark)
```

```
## # A tibble: 0 x 13
## # i 13 variables: Number of Vehicles <dbl>, Accident Date <date>,
## #   Time (24hr) <chr>, 1st Road Class <chr>, Road Surface <chr>,
## #   Lighting Conditions <chr>, Daylight/Dark <chr>, Weather Conditions <chr>,
## #   Type of Vehicle <chr>, Casualty Class <chr>, Casualty Severity <chr>,
## #   Sex of Casualty <chr>, Age of Casualty <dbl>
```

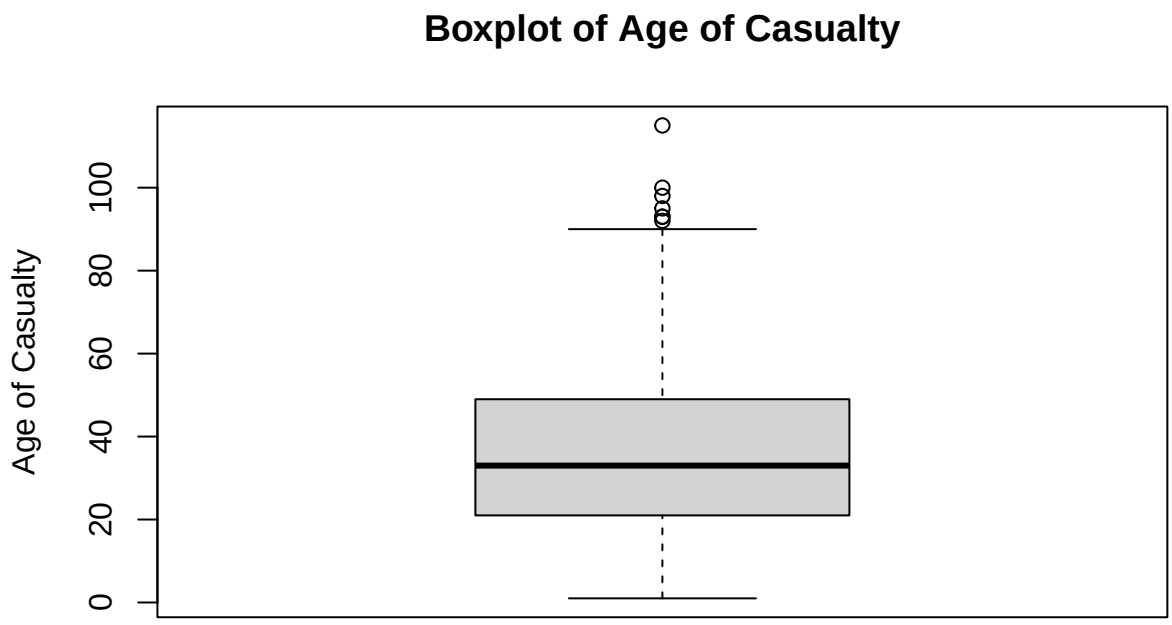
Task 4: Removing Daylight/Darkness from Lighting Conditions columns.

```
df_clean$`Lighting Conditions` <- sub("^(Daylight:|Darkness:)(.*)", "\\2", df_clean$`Lighting Condition
view(df_clean)
```

OUTLIERS in Age of Casualty Column

Task 1: Boxplot- Creating a boxplot to visualize the distribution of age of casualties in the dataframe, while identifying and storing outliers in outlier_box.

```
boxplot(df_clean$`Age of Casualty`, main = "Boxplot of Age of Casualty",
        ylab = "Age of Casualty")
```




```
outlier_box <- boxplot(df_clean$`Age of Casualty`, plot = FALSE)$out
```

Task 2: Printing the number of outliers identified using boxplot analysis from the 'Age of Casualty' column stored in outlier_box.

```
# Display the number of outliers
cat("Number of outliers using boxplots:", outlier_box, "\n")
```

```
## Number of outliers using boxplots: 115 93 93 100 92 95 98
```

Task 3: 3 Sigms Rule- Calculating mean and standard deviation then Calculating upper and lower 3 sigma bounds then Extracting and printing outliers

```
mean_age <- mean(df_clean$`Age of Casualty`, na.rm = TRUE)
sd_age <- sd(df_clean$`Age of Casualty`, na.rm = TRUE)
```

```r
upper_bound_3sigma <- mean_age + 3 * sd_age
lower_bound_3sigma <- mean_age - 3 * sd_age

outliers_3sigma <- df_clean %>% filter((`Age of Casualty` > upper_bound_3sigma) | (`Age of Casualty` < l

outliers_3sigma$`Age of Casualty`
```

```
## [1] 115 100  95  98
```

Task 4: Hample Identifyer- Calculate median and mad then calculating hample bounds and then identifying and Displaying outliers

```r
median_age <- median(df_clean$`Age of Casualty`, na.rm = TRUE)
mad_age <- mad(df_clean$`Age of Casualty`, na.rm = TRUE)

upper_bound <- median_age + 3 * mad_age
lower_bound <- max(0, median_age - 3 * mad_age)

outliers <- df_clean %>% filter(`Age of Casualty` > upper_bound | `Age of Casualty` < lower_bound)

outliers$`Age of Casualty`
```

```
## [1] 115  93  93 100  92  95  98
```

Best Method of outlier detection

Saving the clean Dataframe

```r
view(df_clean)
write.csv(df_clean, "clean_accident.csv", row.names = FALSE,  quote= FALSE, fileEncoding = "UTF-8")
```

PART 2: Data Exploration

Task 1:Loading hte 'clean_accident.csv' dataset

```r
data <- read_csv("clean_accident.csv")
```

```
## Rows: 2069 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (9): 1st Road Class, Road Surface, Lighting Conditions, Daylight/Dark, ...
## dbl  (2): Number of Vehicles, Age of Casualty
## date (1): Accident Date
## time (1): Time (24hr)
##
## i Use 'spec()' to retrieve the full column specification for this data.
## i Specify the column types or set 'show_col_types = FALSE' to quiet this message.
```

```r
head(data)
```

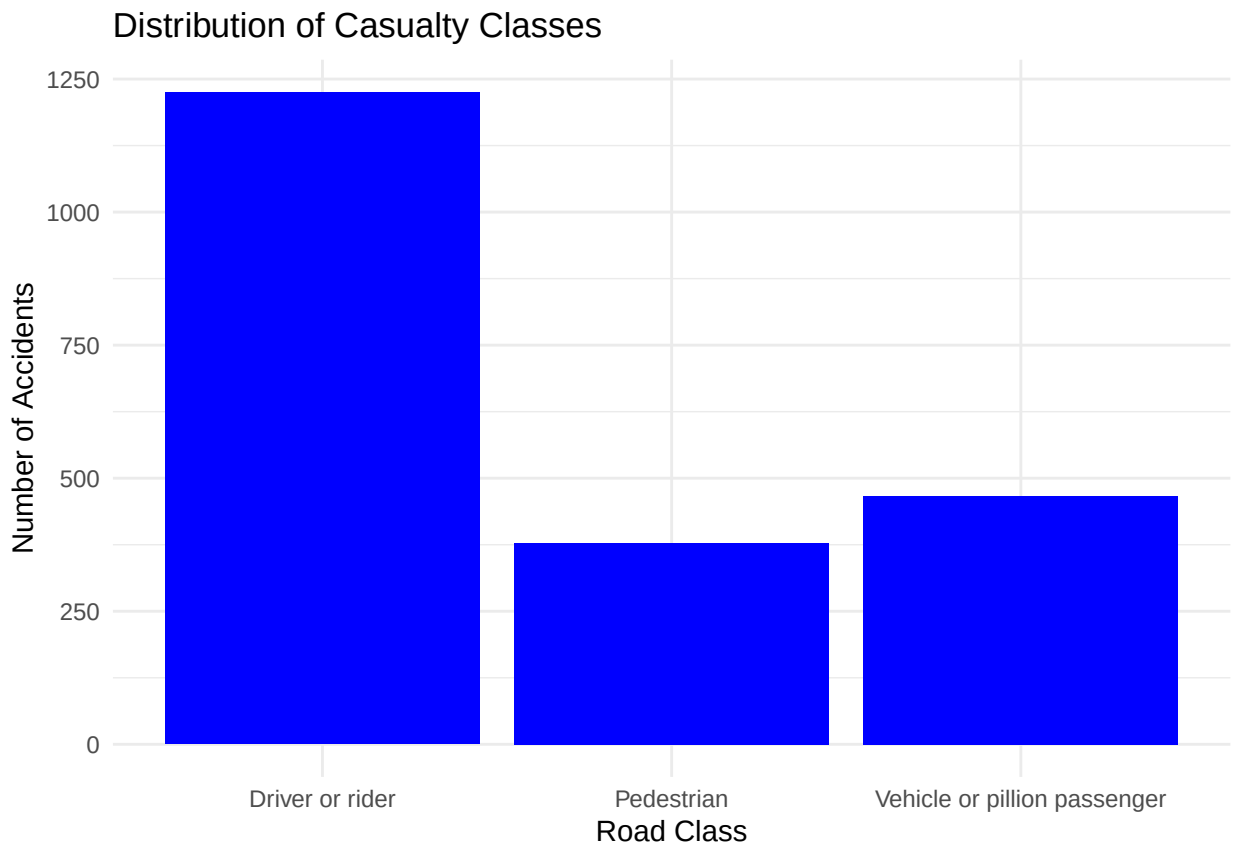
```
## # A tibble: 6 x 13
##   `Number of Vehicles` `Accident Date` `Time (24hr)` `1st Road Class`
##                  <dbl> <date>          <time>        <chr>
## 1                    2 2017-01-01      21:20         Unclassified
## 2                    2 2017-01-04      15:00         Unclassified
## 3                    2 2017-01-05      07:32         A
## 4                    2 2017-01-05      09:30         A
## 5                    2 2017-01-14      09:09         Unclassified
## 6                    1 2017-01-15      16:59         Unclassified
## # i 9 more variables: `Road Surface` <chr>, `Lighting Conditions` <chr>,
## #   `Daylight/Dark` <chr>, `Weather Conditions` <chr>, `Type of Vehicle` <chr>,
## #   `Casualty Class` <chr>, `Casualty Severity` <chr>, `Sex of Casualty` <chr>,
## #   `Age of Casualty` <dbl>
```

Task 2: Generate random colors for each unique level of `Casualty Severity`

```r
colors <- sample(colors(), length(unique(data$`Casualty Severity`)))
```

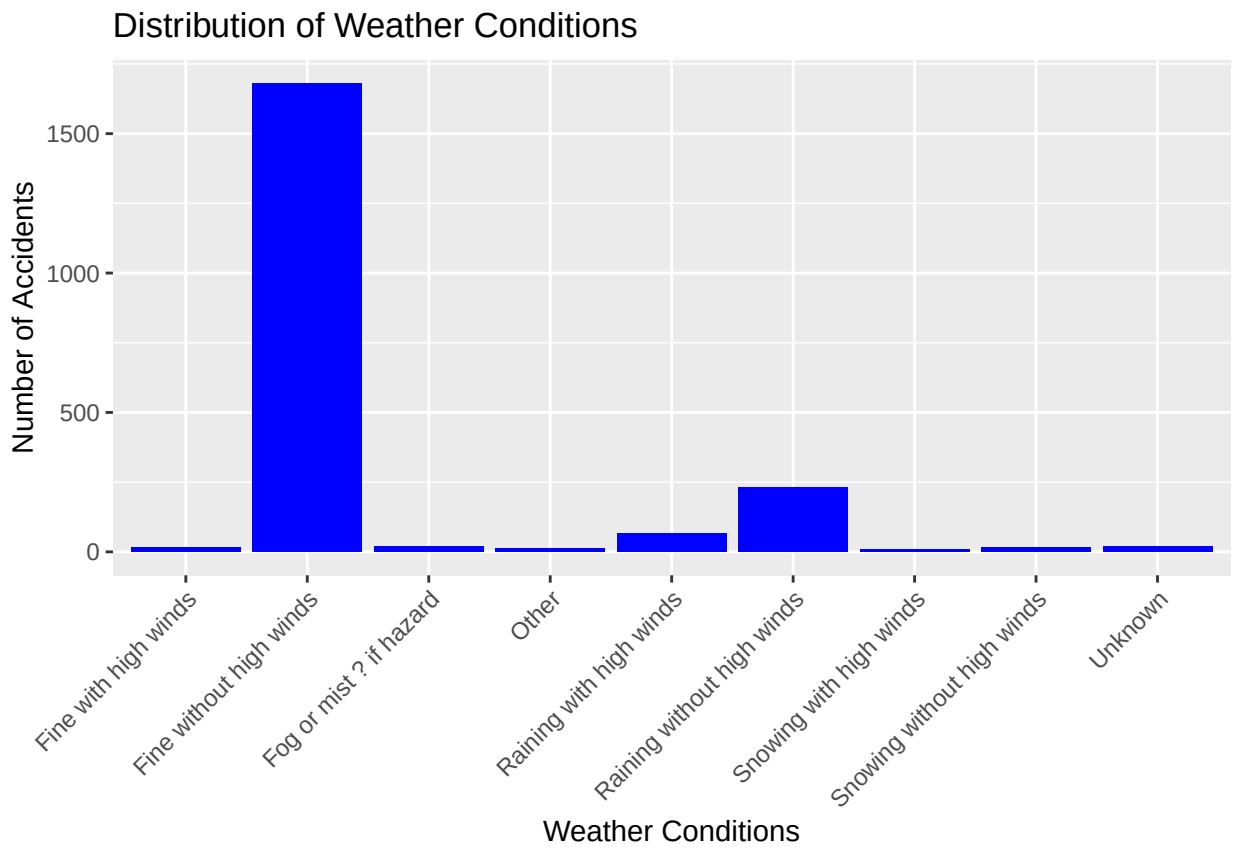
Task 3: Displaying distribution of road classes.

```r
ggplot(data, aes(x = `Casualty Class`)) +
  geom_bar(fill = "blue") +
  labs(title = "Distribution of Casualty Classes",
       x = "Road Class",
       y = "Number of Accidents") +
  theme_minimal()
```



Task 4: Displaying distribution of Weather Conditions.

```
ggplot(data, aes(x = `Weather Conditions`)) +
  geom_bar(fill = "blue") +
  labs(title = "Distribution of Weather Conditions",
       x = "Weather Conditions",
       y = "Number of Accidents") +
 theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



Task 5:Summarizes the count of casualties grouped by 'Weather Conditions' and 'Casulty class' of Driver/rider only from the dataset

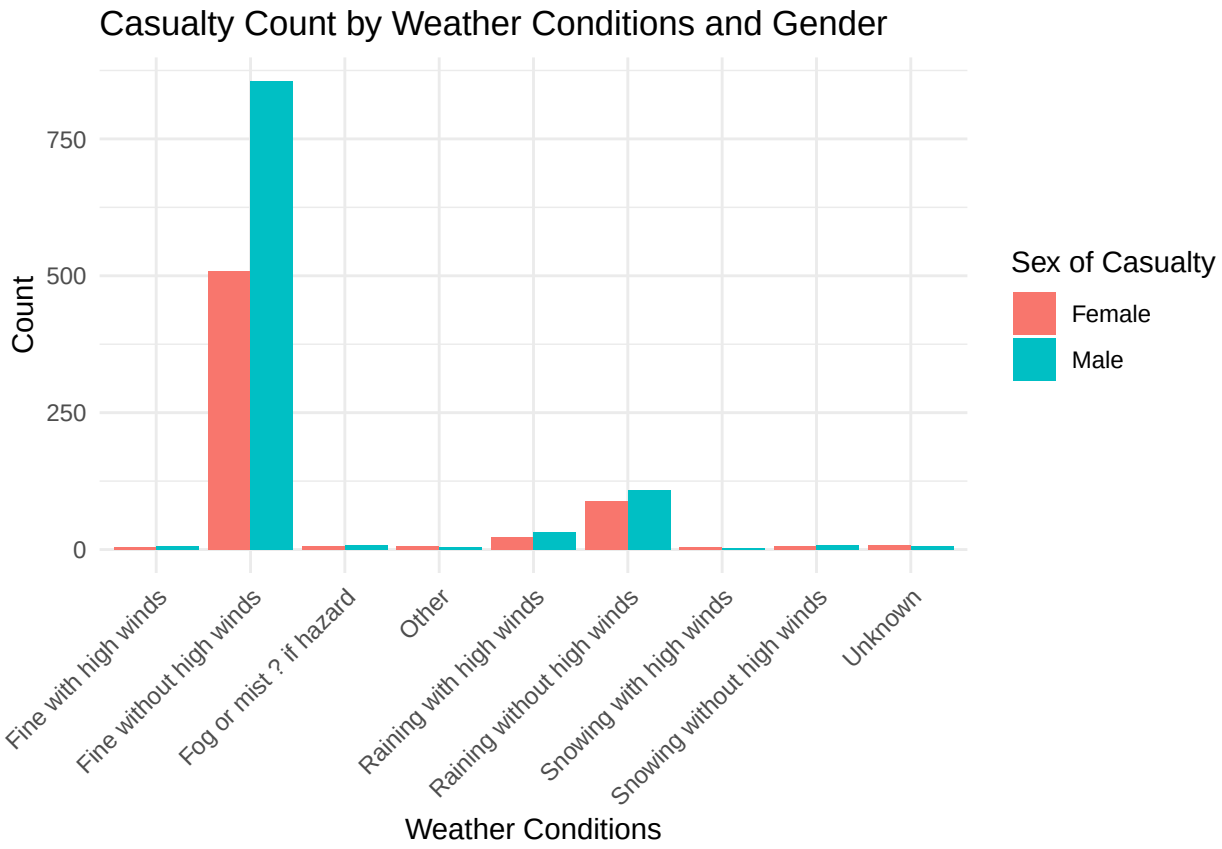
```
gender_data <- data %>%
  filter(`Sex of Casualty` %in% c("Male", "Female"), `Casualty Class` %in% c("Driver or rider", "Vehicl
weather_gender_table <- table(gender_data$`Weather Conditions`, gender_data$`Sex of Casualty`)

print(weather_gender_table)
```

```
##
##                             Female Male
##   Fine with high winds           5    7
##   Fine without high winds      508  856
##   Fog or mist ? if hazard        7    9
##   Other                          7    4
##   Raining with high winds       23   32
##   Raining without high winds    88  109
##   Snowing with high winds        4    3
##   Snowing without high winds     7    8
##   Unknown                        8    6
```

Task 6: Displaying Accidents by weather conditions and Gender in bar plot.

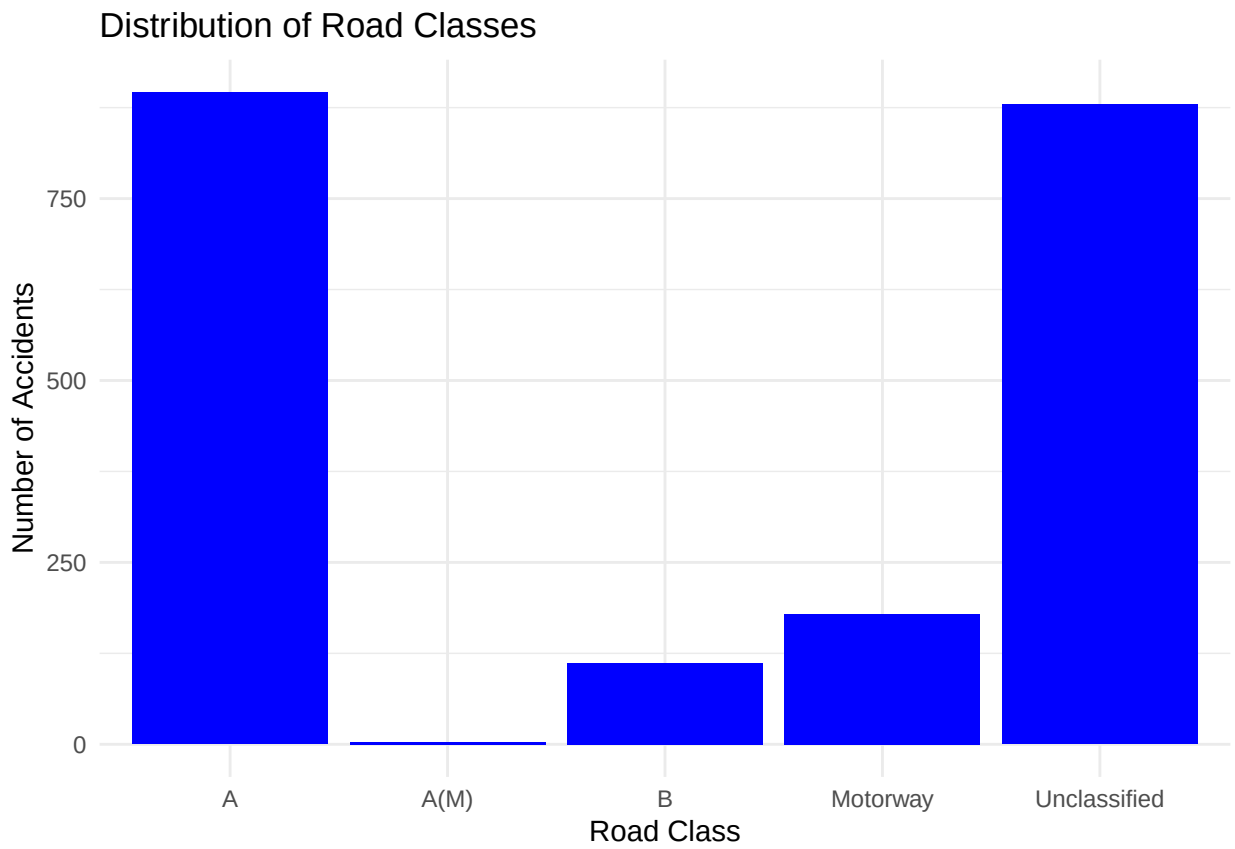
```
weather_gender_df <- as.data.frame(weather_gender_table)
```

```r
colnames(weather_gender_df) <- c("Weather Conditions", "Sex of Casualty", "Count")

ggplot(weather_gender_df, aes(x = `Weather Conditions`, y = Count, fill = `Sex of Casualty`)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(title = "Casualty Count by Weather Conditions and Gender",
       x = "Weather Conditions",
       y = "Count",
       fill = "Sex of Casualty") +
  theme_minimal()+
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



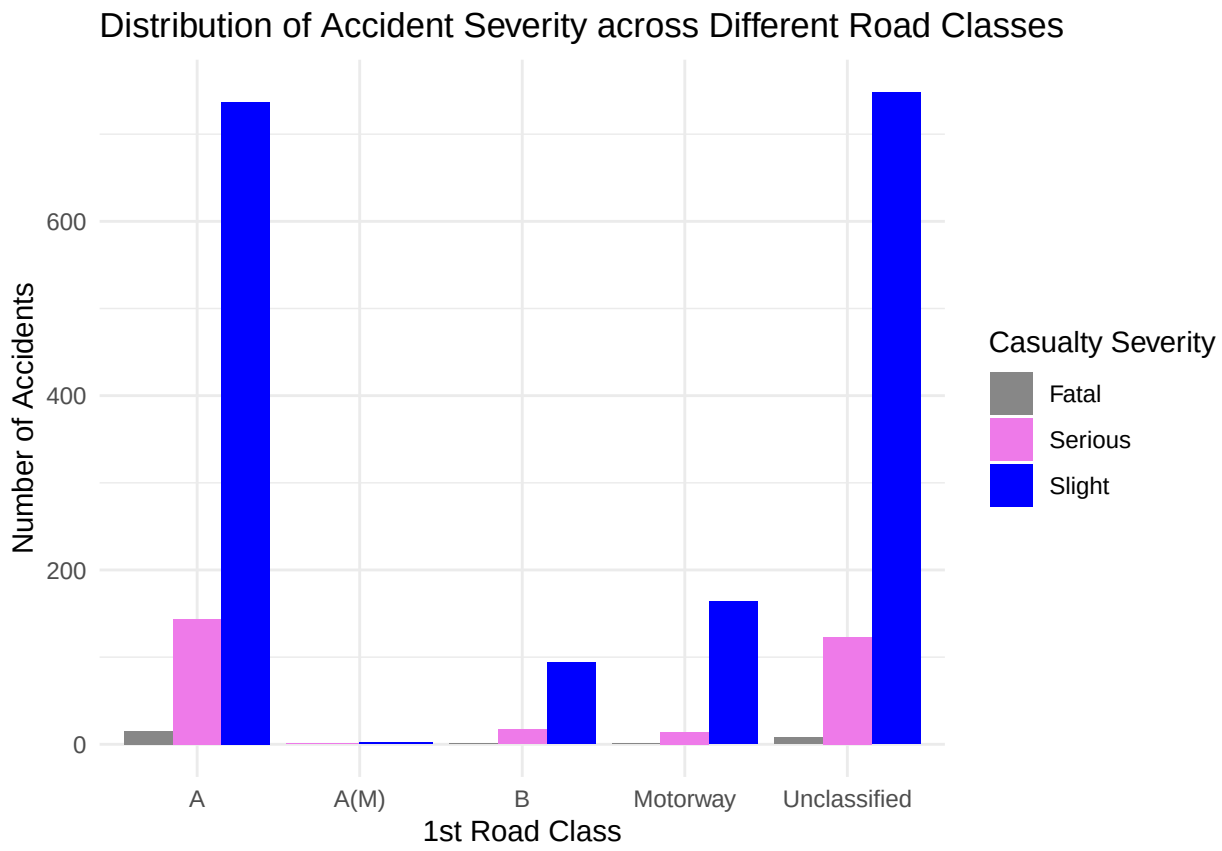
Task 7: Displaying Distribution of Road Classes.

```r
ggplot(data, aes(x = `1st Road Class`)) +
  geom_bar(fill = "blue") +
  labs(title = "Distribution of Road Classes",
       x = "Road Class",
       y = "Number of Accidents") +
  theme_minimal()
```



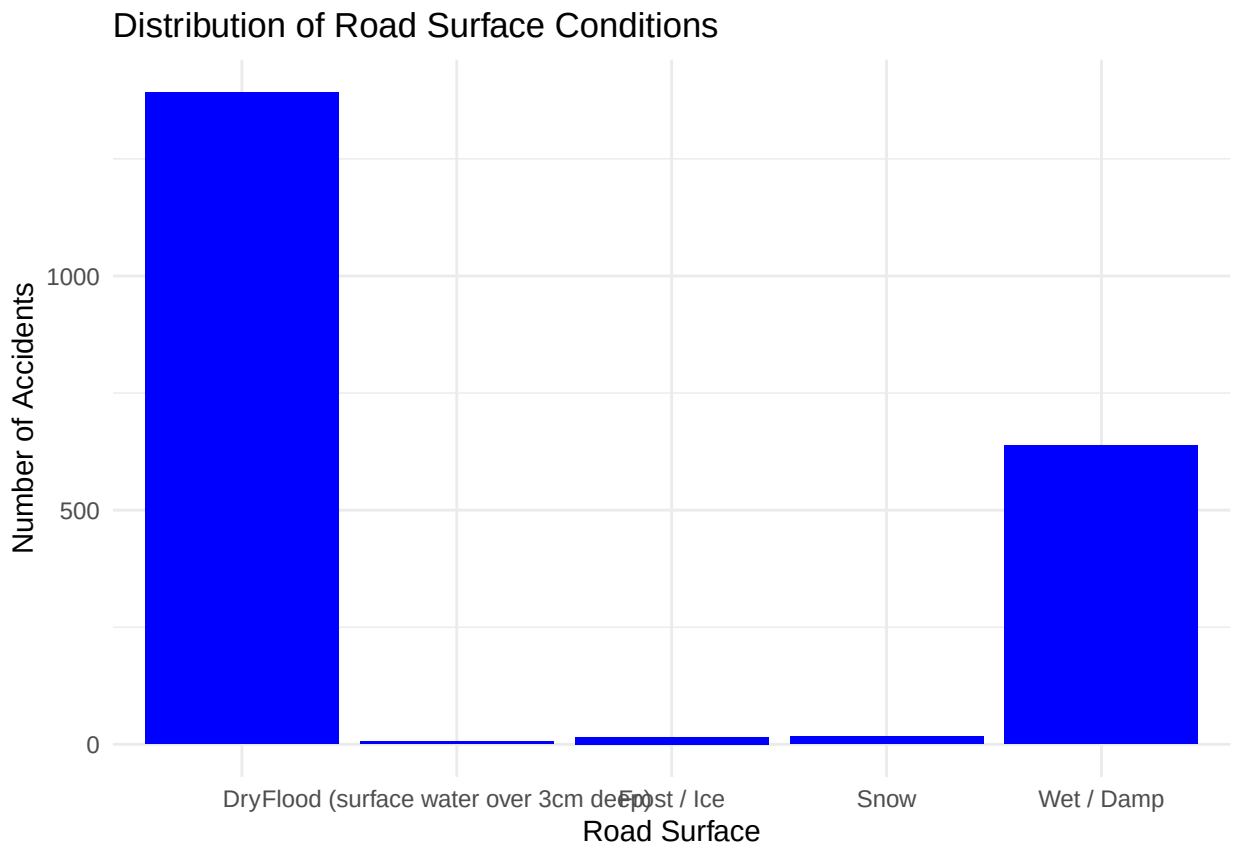
Task 8: Displaying Distribution of Casualty Severity across different Road Classes.

```
ggplot(data, aes(x = `1st Road Class`, fill = `Casualty Severity`)) +
  geom_bar(position = "dodge") +
  labs(title = "Distribution of Accident Severity across Different Road Classes",
       x = "1st Road Class",
       y = "Number of Accidents",
       fill = "Casualty Severity") +
  scale_fill_manual(values = colors) +  # Set random fill colors
  theme_minimal()
```



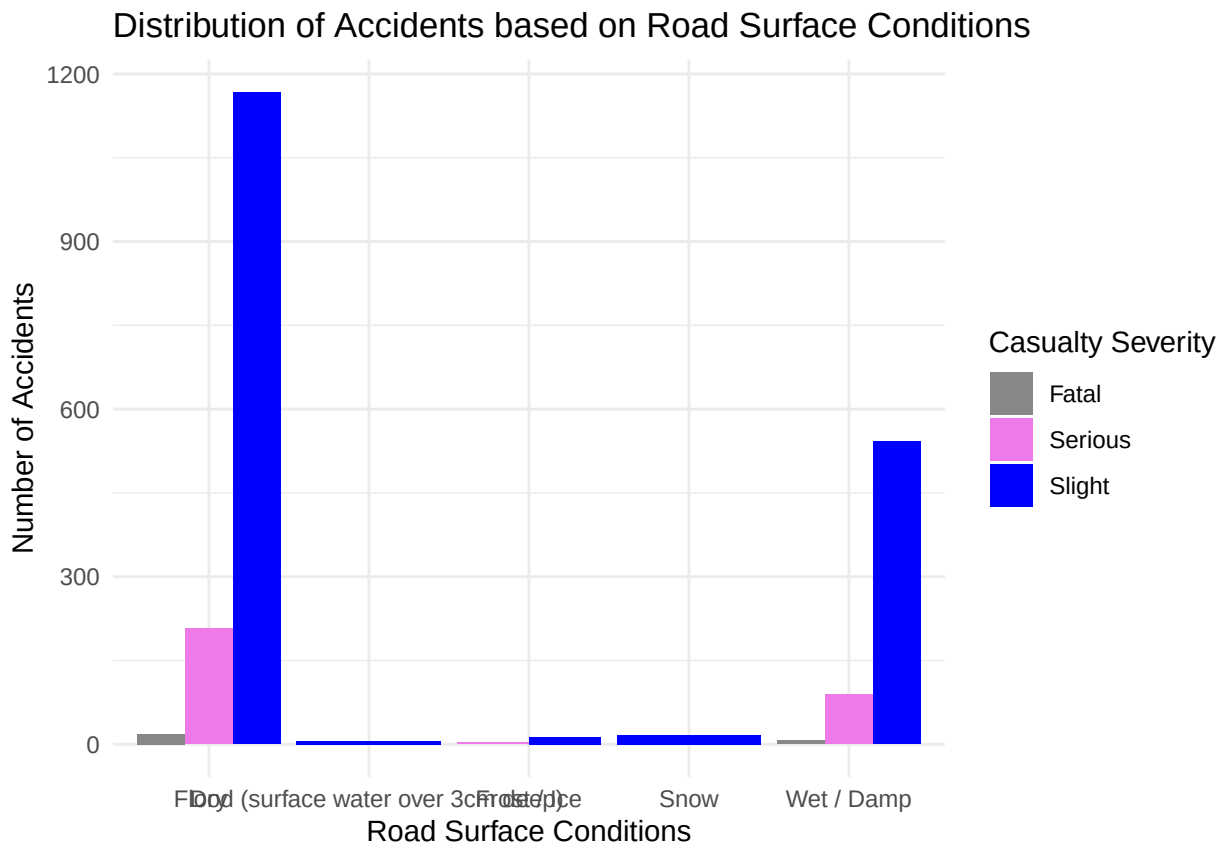
Task 9: Displaying Distribution of Road Surface Conditions.

```
ggplot(data, aes(x = `Road Surface`)) +
  geom_bar(fill = "blue") +
  labs(title = "Distribution of Road Surface Conditions",
       x = "Road Surface",
       y = "Number of Accidents") +
  theme_minimal()
```



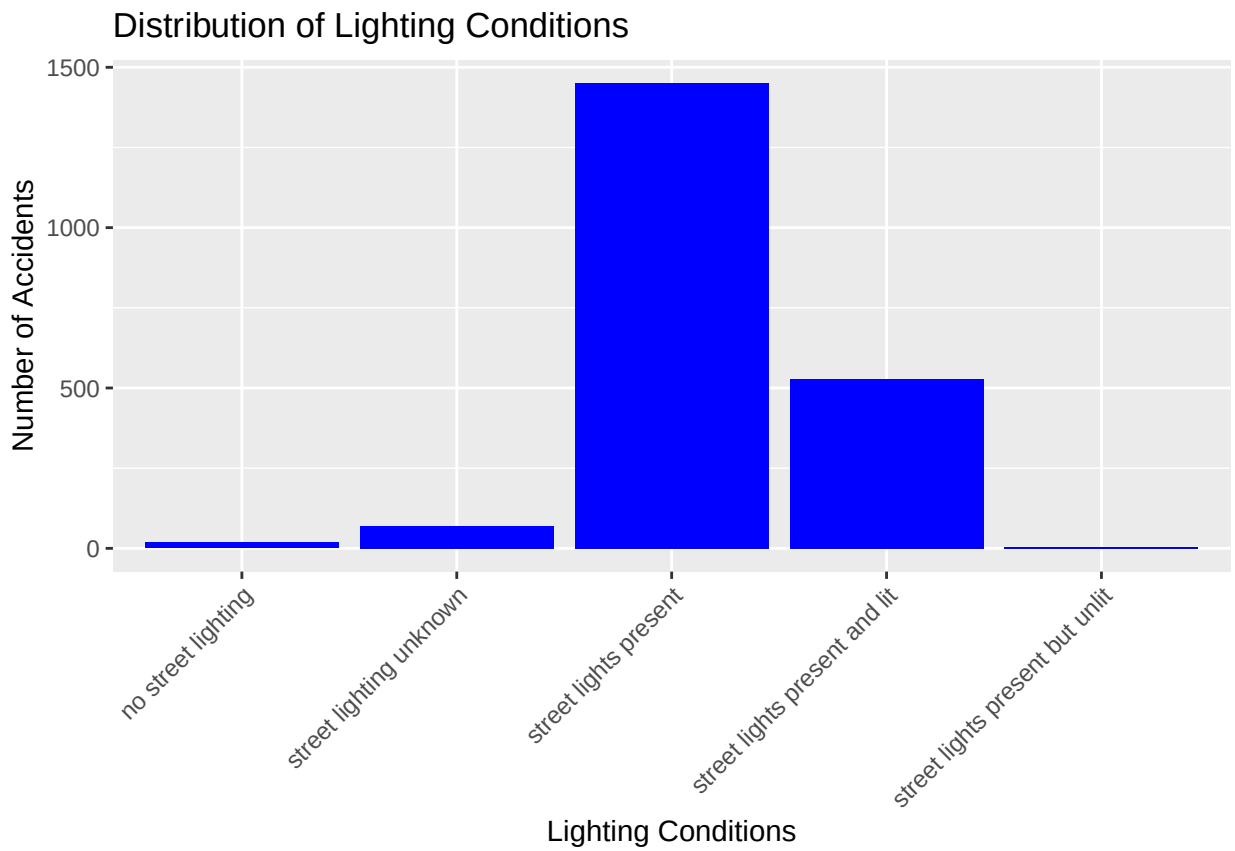
Task 10: Displaying Distribution of Casualty Severity across different Road Surface.

```
ggplot(data, aes(x = `Road Surface`, fill = `Casualty Severity`)) +
  geom_bar(position = "dodge") +
  labs(title = "Distribution of Accidents based on Road Surface Conditions",
       x = "Road Surface Conditions",
       y = "Number of Accidents",
       fill = "Casualty Severity") +
    scale_fill_manual(values = colors) +
  theme_minimal()
```



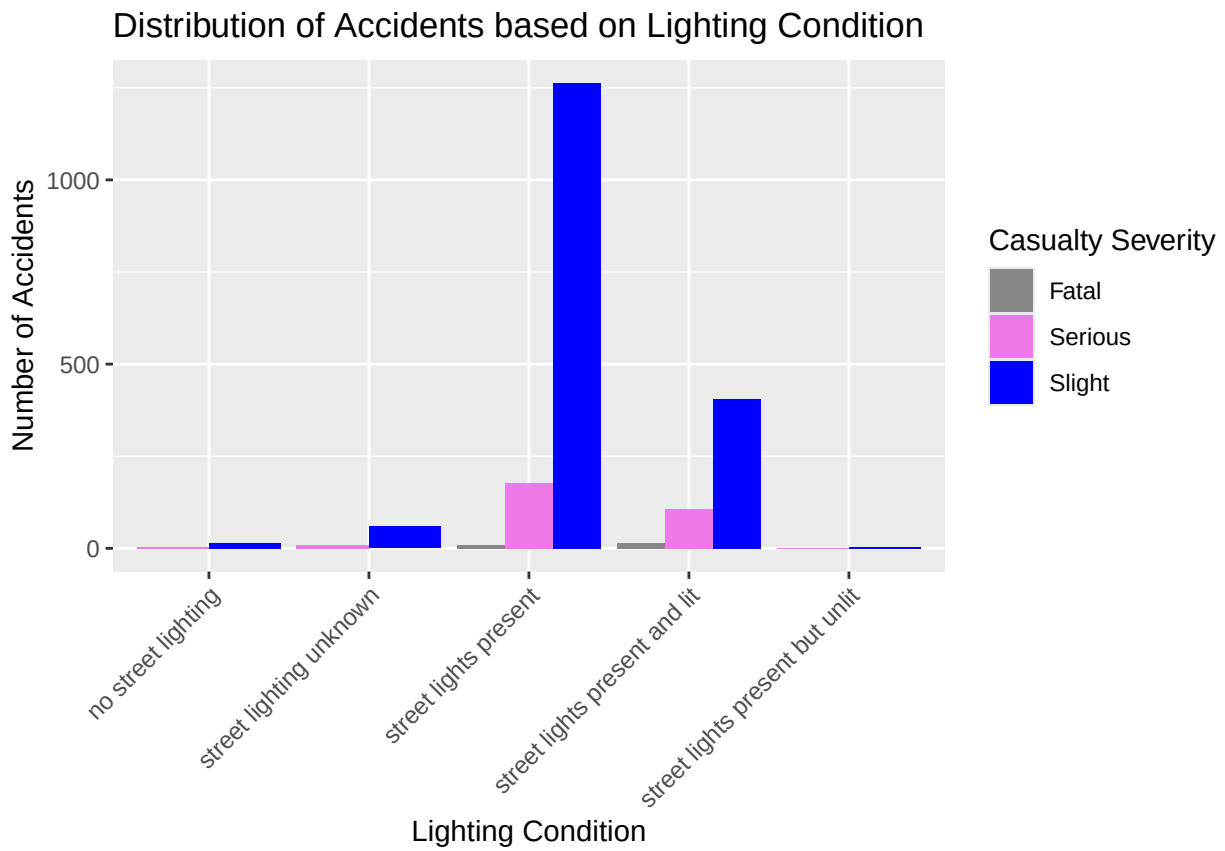
Task 11: Displaying Distribution of Lighting Conditions.

```
ggplot(data, aes(x = `Lighting Conditions`)) +
  geom_bar(fill = "blue") +
  labs(title = "Distribution of Lighting Conditions",
       x = "Lighting Conditions",
       y = "Number of Accidents") +
    theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



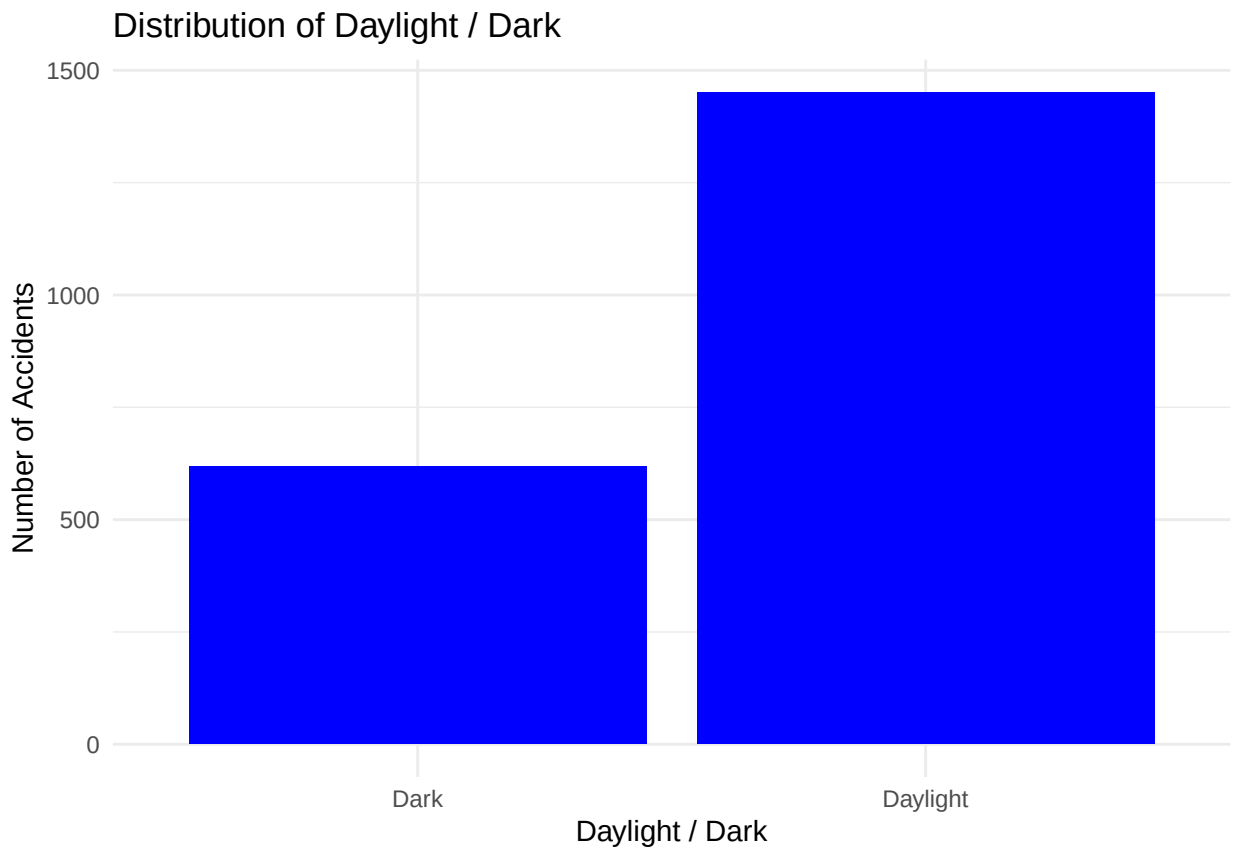
Task 12: Displaying Distribution of Casualty Severity across different Lighting Condition.

```
ggplot(data, aes(x = `Lighting Conditions`, fill = `Casualty Severity`)) +
  geom_bar(position = "dodge") +
  labs(title = "Distribution of Accidents based on Lighting Condition",
       x = "Lighting Condition",
       y = "Number of Accidents",
       fill = "Casualty Severity") +
  scale_fill_manual(values = colors) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



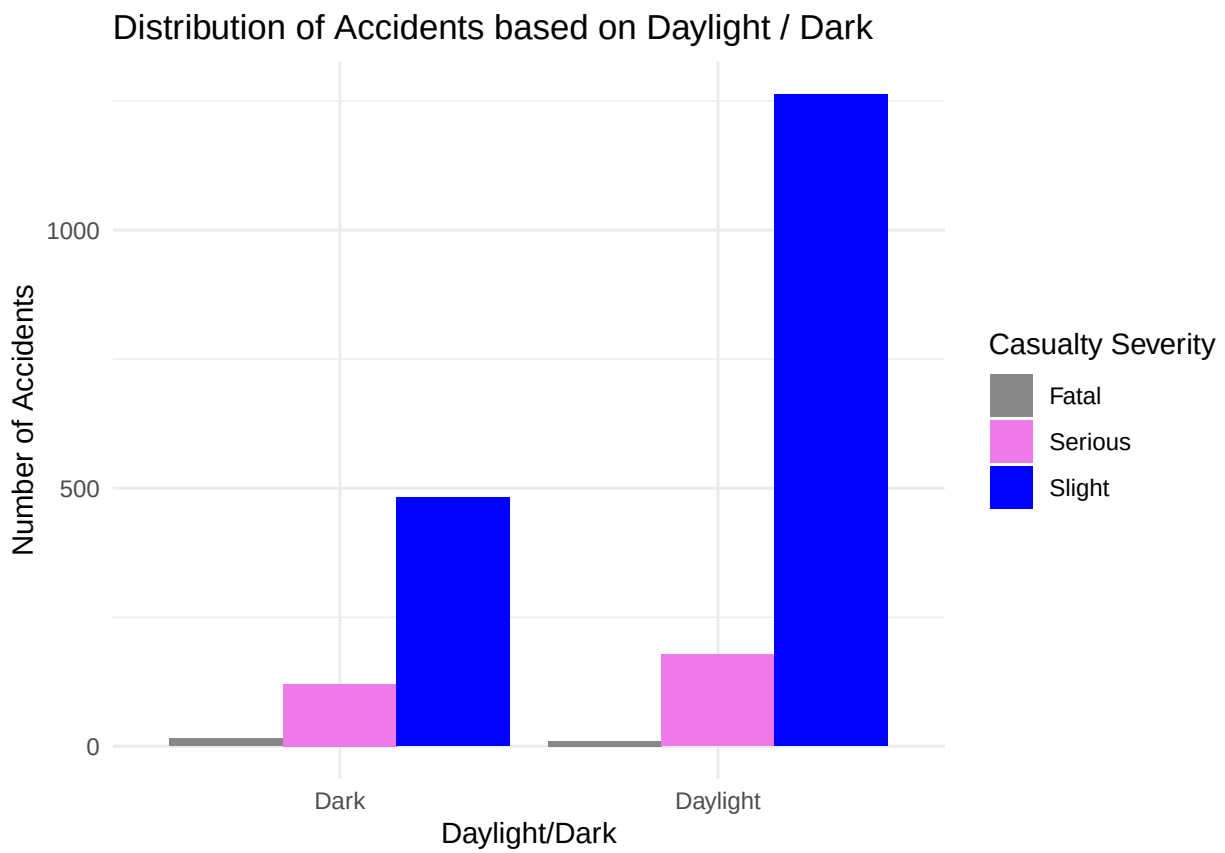
Task 13: Displaying Distribution of Daylight/Dark.

```
ggplot(data, aes(x = `Daylight/Dark`)) +
  geom_bar(fill = "blue") +
  labs(title = "Distribution of Daylight / Dark",
       x = "Daylight / Dark",
       y = "Number of Accidents") +
  theme_minimal()
```



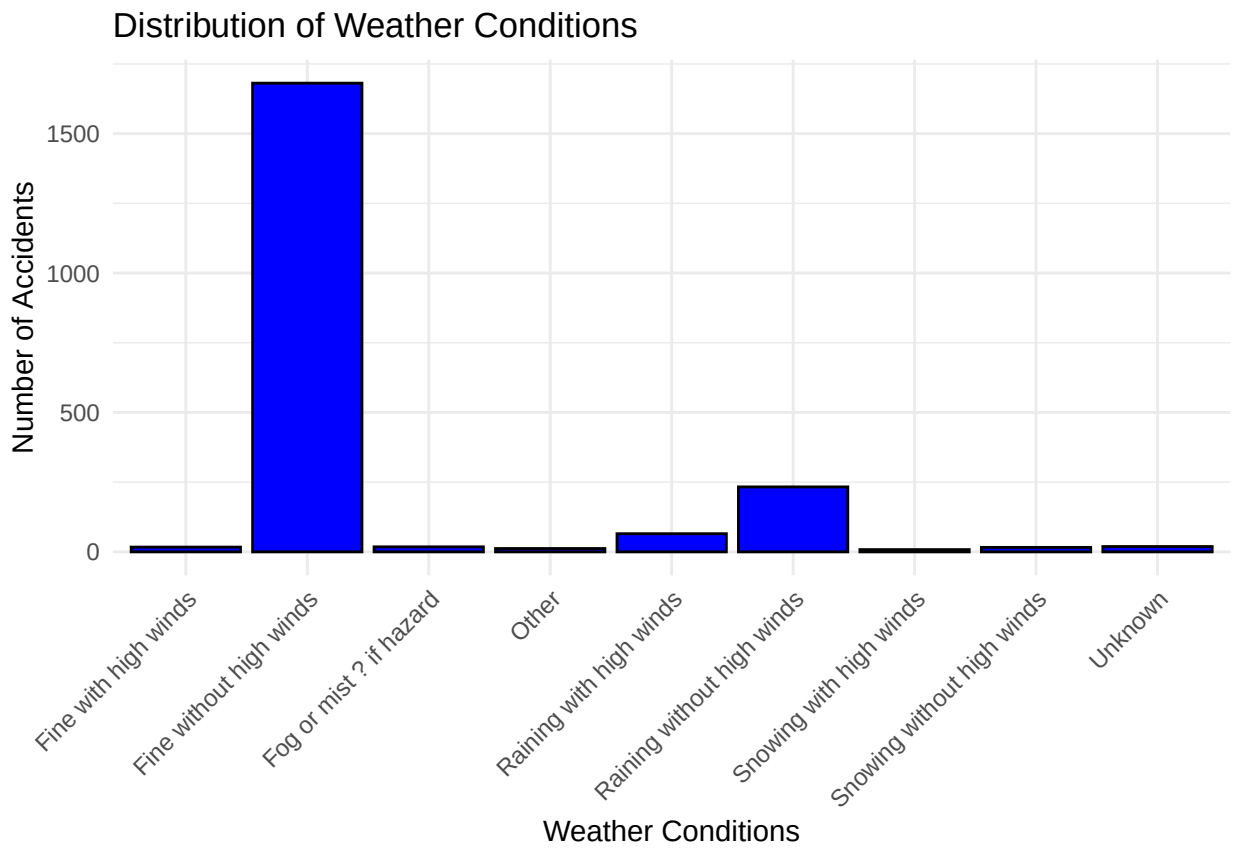
Task 14: Displaying Distribution of Casualty Severity across Dayligh/Dark.

```
ggplot(data, aes(x = `Daylight/Dark`, fill = `Casualty Severity`)) +
  geom_bar(position = "dodge") +
  labs(title = "Distribution of Accidents based on Daylight / Dark",
       x = "Daylight/Dark",
       y = "Number of Accidents",
       fill = "Casualty Severity") +
    scale_fill_manual(values = colors) +
  theme_minimal()
```



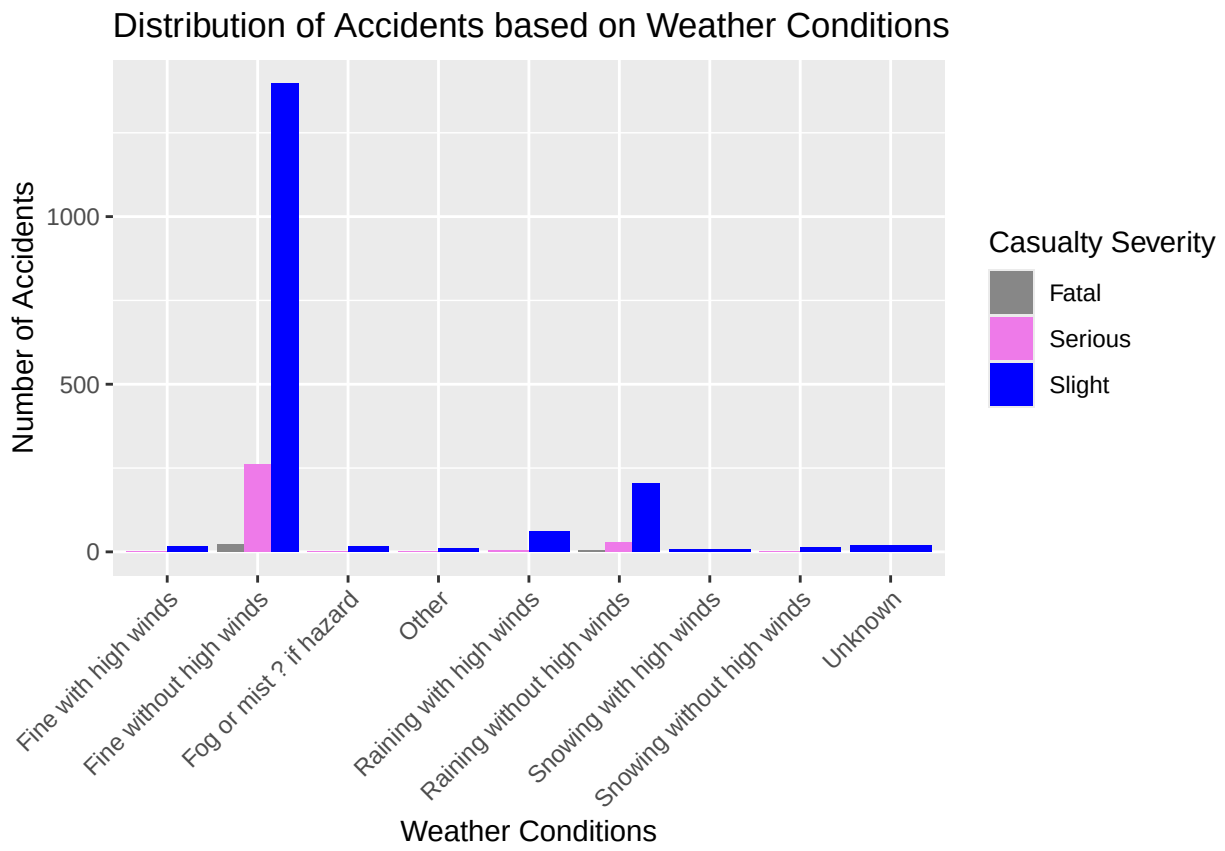
Task 15: Displaying Distribution of Weather Conditions.

```
ggplot(data, aes(x = `Weather Conditions`)) +
  geom_bar(fill = "blue", color = "black", stat = "count") +
  labs(title = "Distribution of Weather Conditions",
       x = "Weather Conditions",
       y = "Number of Accidents") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



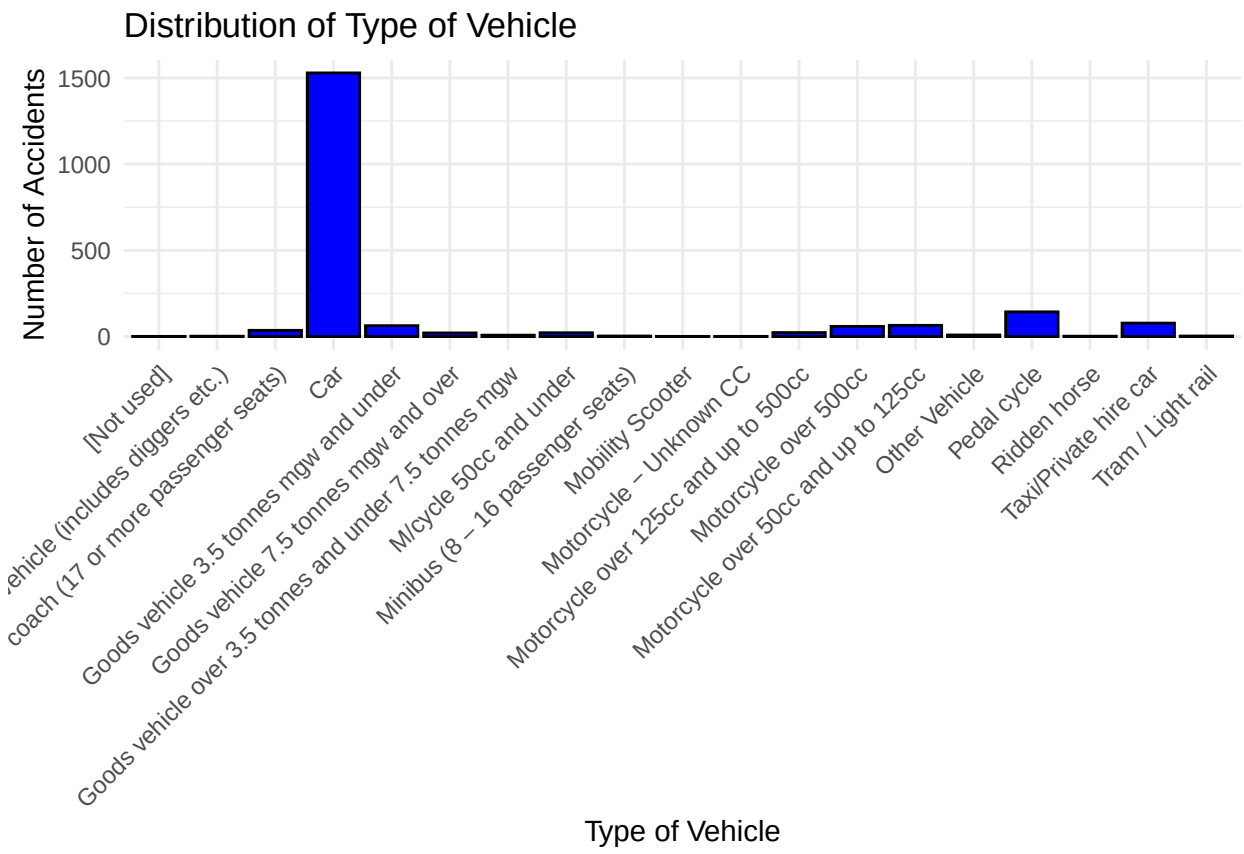
Task 16: Displaying Distribution of Casualty Severity across Weather Conditions.

```
ggplot(data, aes(x = `Weather Conditions`, fill = `Casualty Severity`)) +
  geom_bar(position = "dodge") +
  labs(title = "Distribution of Accidents based on Weather Conditions",
       x = "Weather Conditions",
       y = "Number of Accidents",
       fill = "Casualty Severity") +
    scale_fill_manual(values = colors) +
   theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



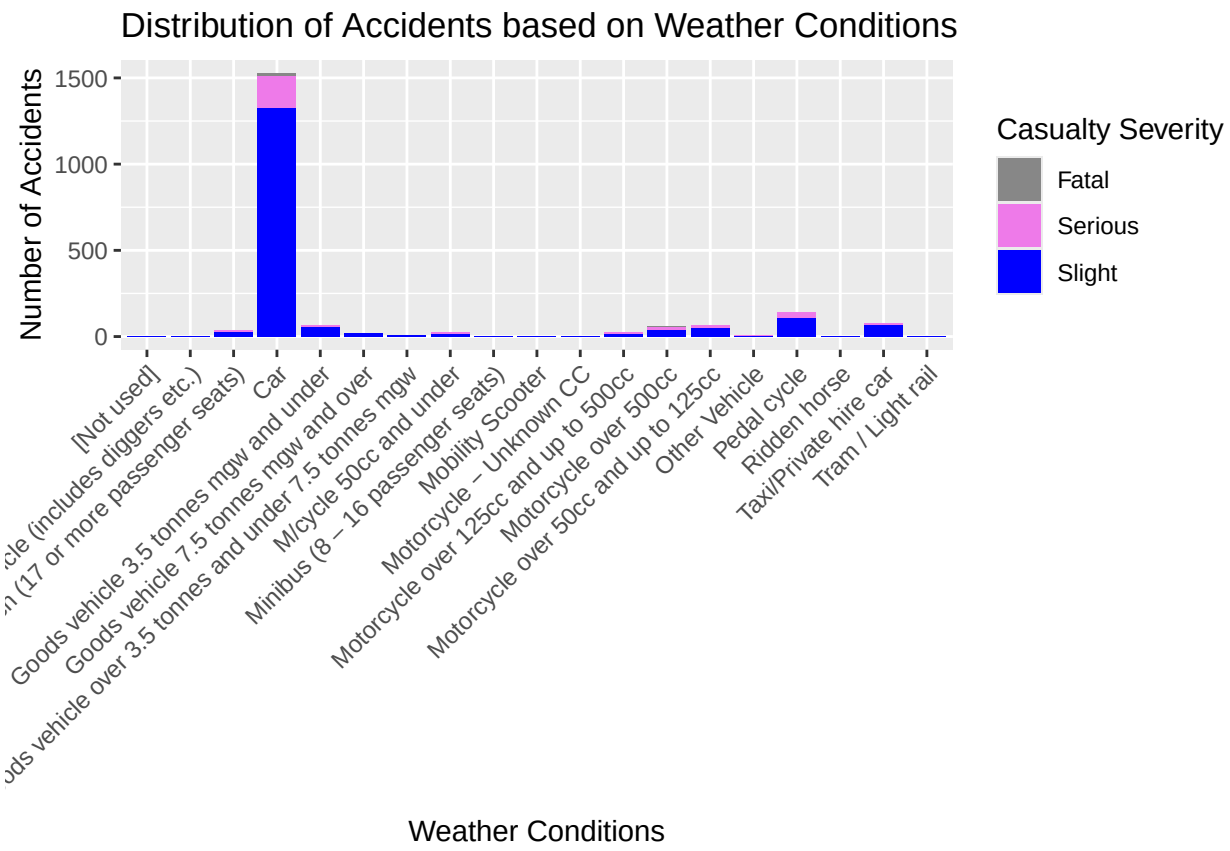
Task 17: Displaying Distribution of Type of Vehicle.

```
ggplot(data, aes(x = `Type of Vehicle`)) +
  geom_bar(fill = "blue", color = "black", stat = "count") +
  labs(title = "Distribution of Type of Vehicle",
       x = "Type of Vehicle",
       y = "Number of Accidents") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



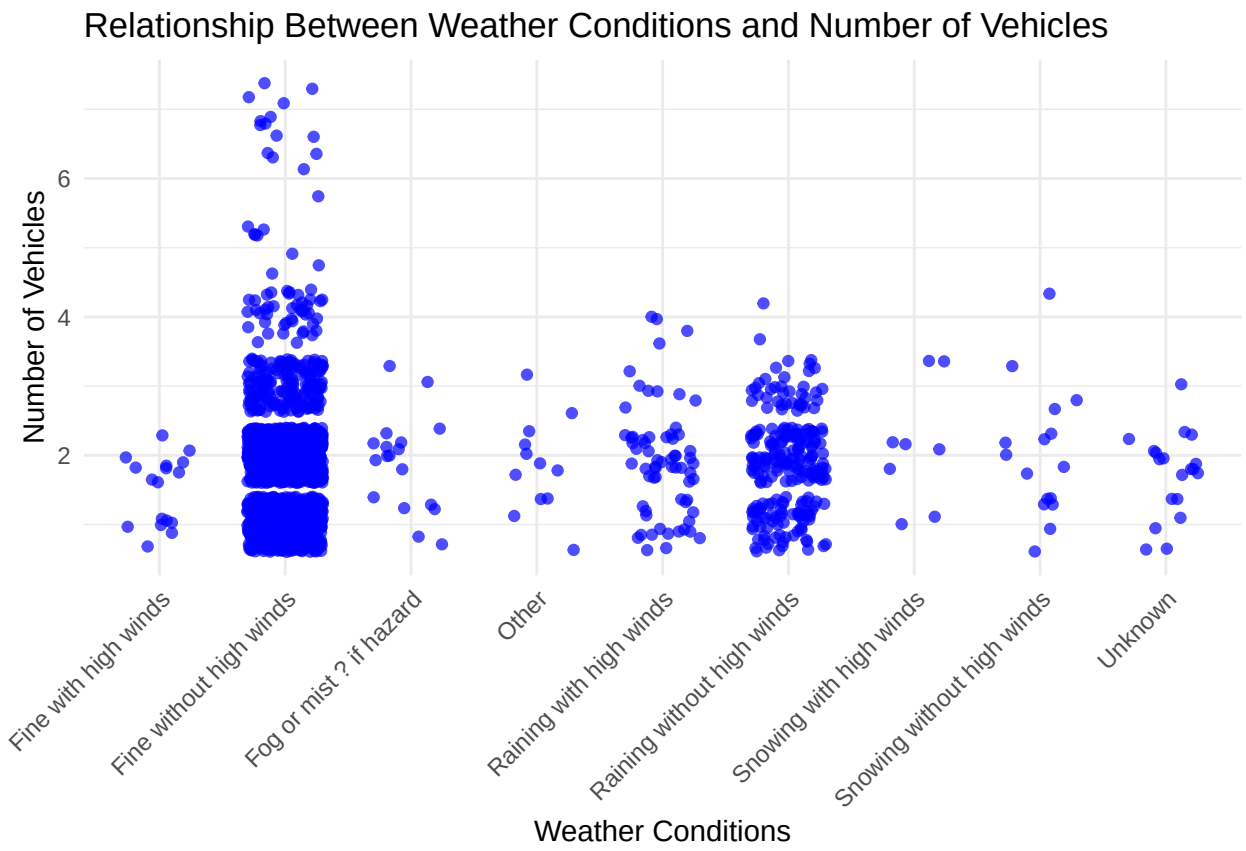
Task 18: Displaying Distribution of Casualty Severity across Type of Vehicle

```
ggplot(data, aes(x = `Type of Vehicle`, fill = `Casualty Severity`)) +
  geom_bar(position = "stack") +
  labs(title = "Distribution of Accidents based on Weather Conditions",
       x = "Weather Conditions",
       y = "Number of Accidents",
       fill = "Casualty Severity") +
    scale_fill_manual(values = colors) +
   theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



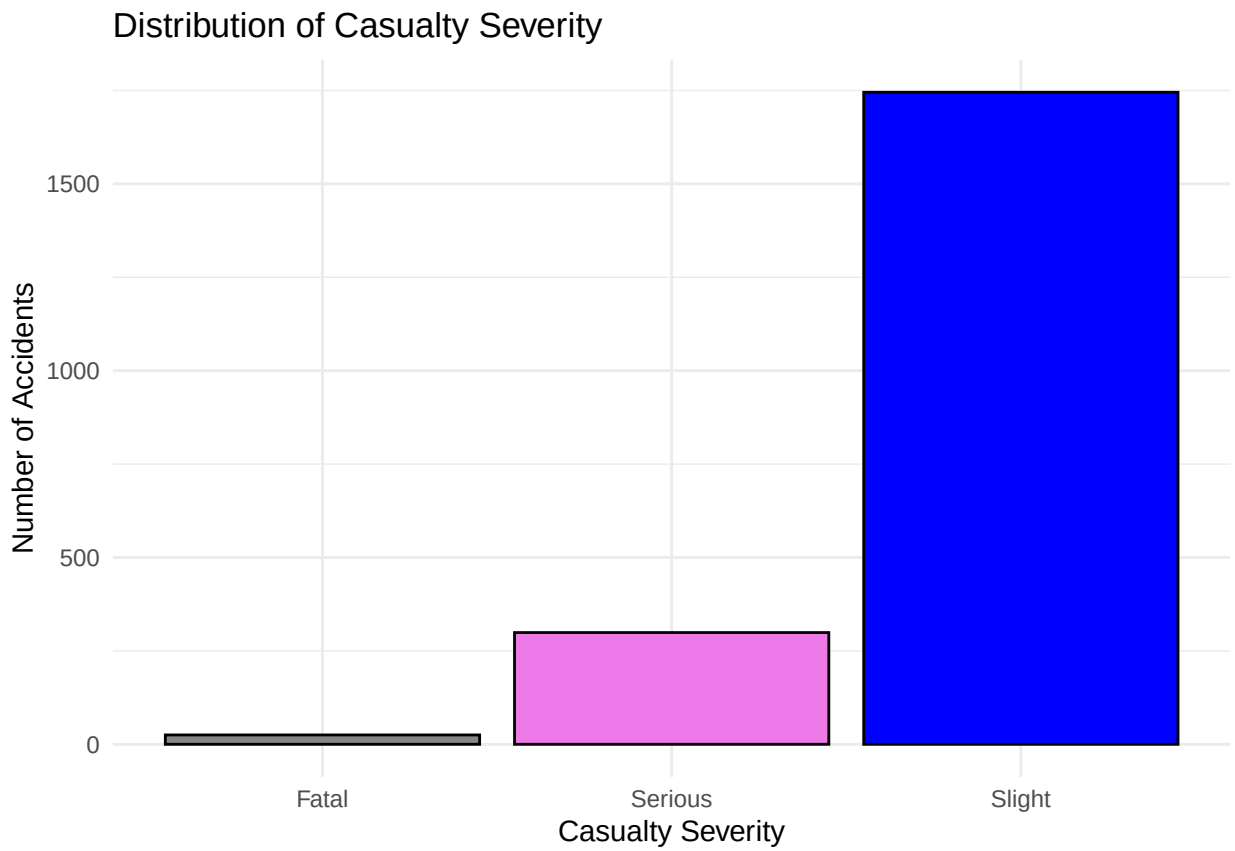
Task 19: Displaying Relationship between weather conditions and Number of vehicles.

```
ggplot(data, aes(x = `Weather Conditions`, y = `Number of Vehicles`)) +
  geom_jitter(color = "blue", alpha = 0.7, width = 0.3) +
  labs(title = "Relationship Between Weather Conditions and Number of Vehicles ",
       x = "Weather Conditions",
       y = "Number of Vehicles") +
  theme_minimal()+
theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



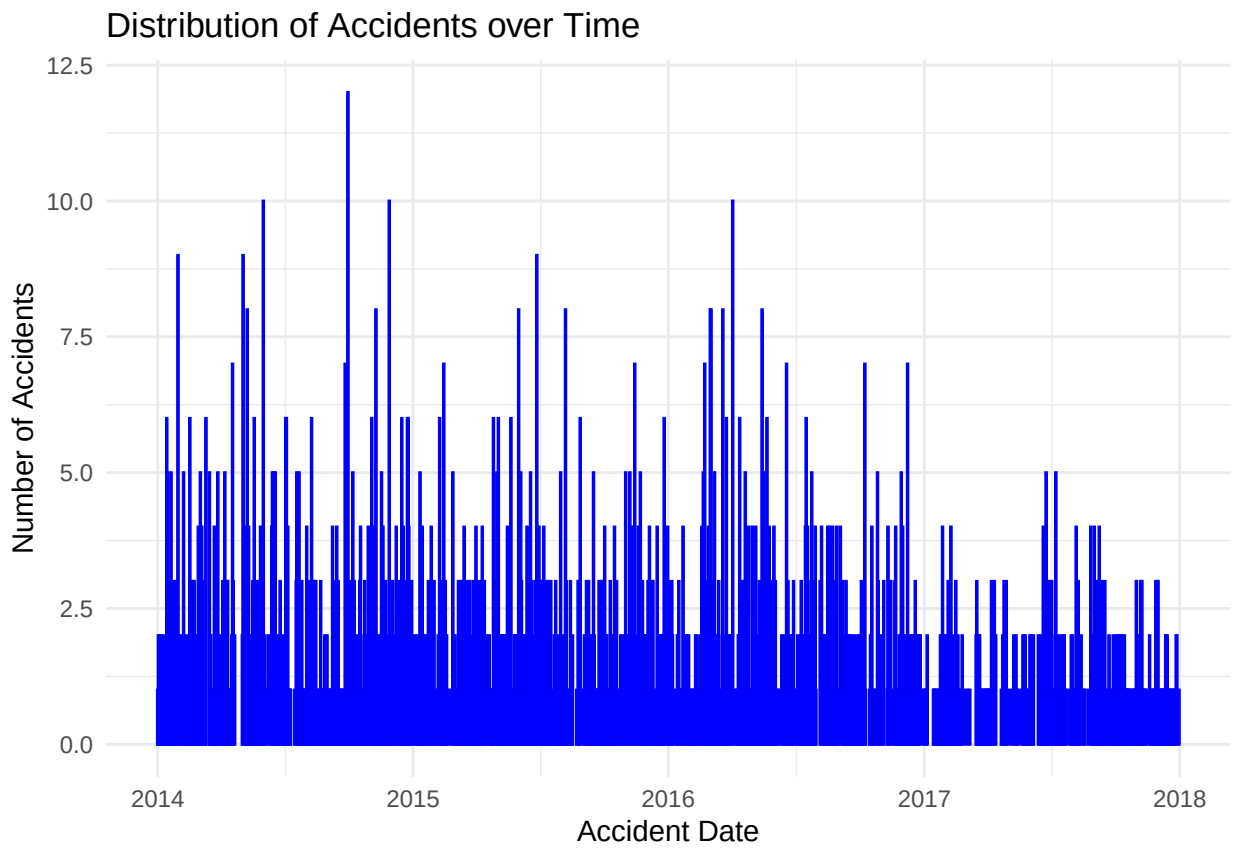
Task 20: Displaying Distribution of Casualty Severity.

```
ggplot(data, aes(x = `Casualty Severity`, fill = `Casualty Severity`)) +
  geom_bar(fill = colors, color = "black") +  # Use random colors
  labs(title = "Distribution of Casualty Severity",
       x = "Casualty Severity",
       y = "Number of Accidents") +
  theme_minimal()
```



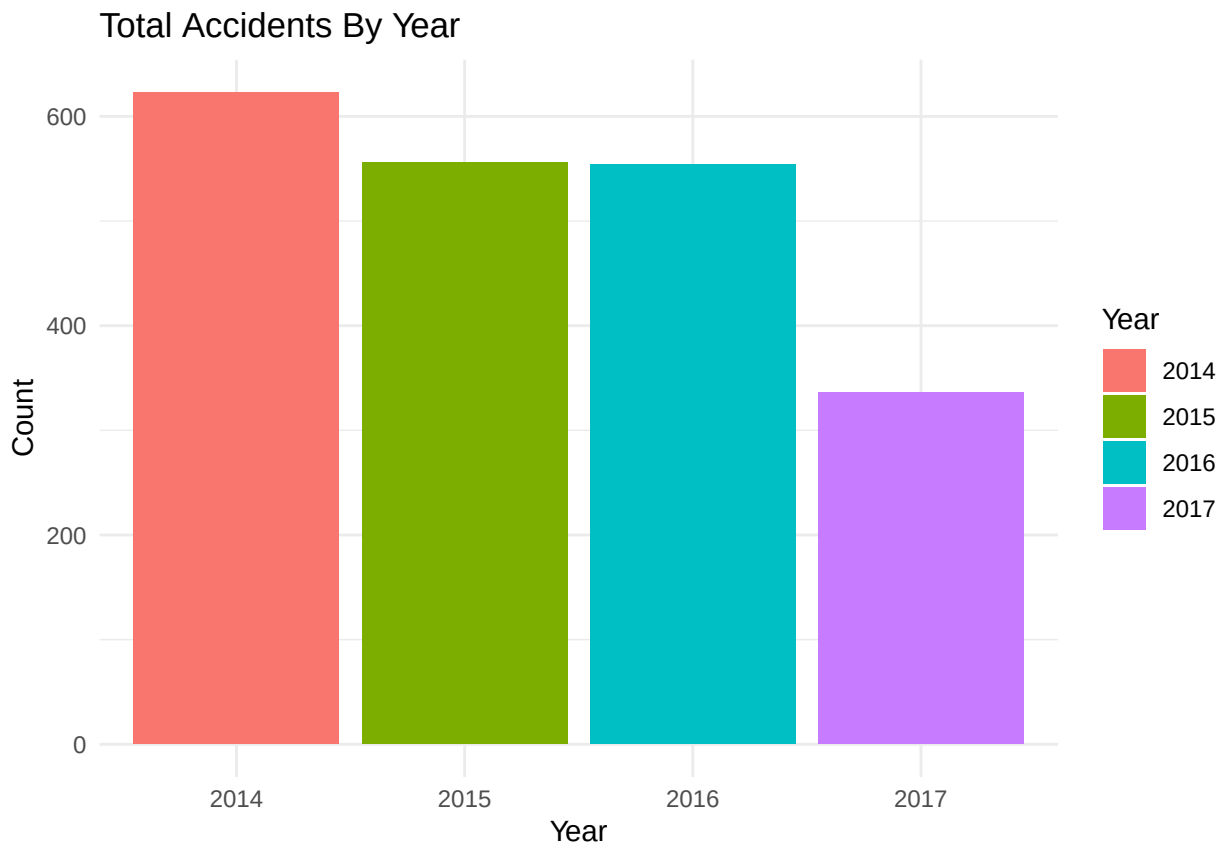
Task 21: Displaying distribution of Accidents over time.

```
ggplot(data, aes(x = `Accident Date`)) +
  geom_bar(fill = "red", color = "blue") +
  labs(title = "Distribution of Accidents over Time",
       x = "Accident Date",
       y = "Number of Accidents") +
  theme_minimal()
```



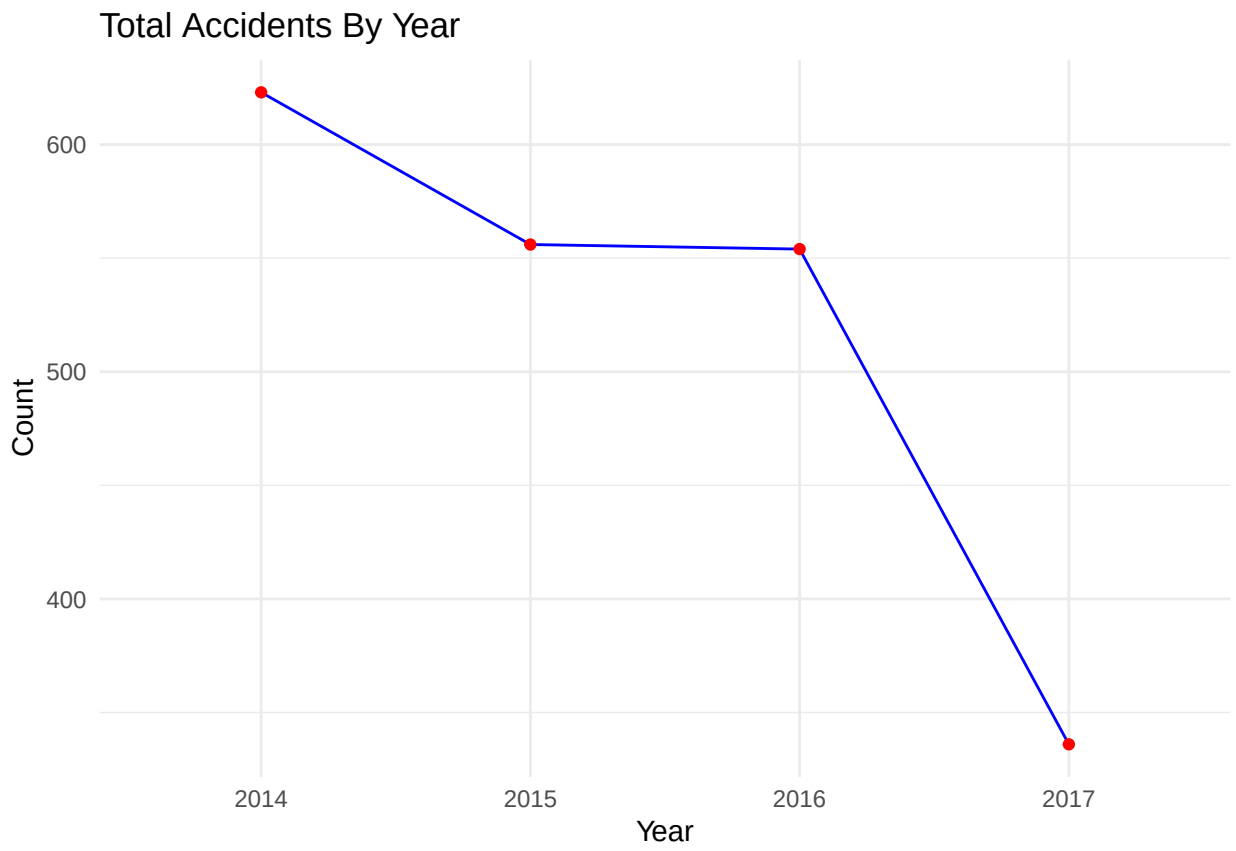
Task 22:Displaying total accidents by year in bar graph.

```
yearly_data <- data %>%
  mutate(Year = year(`Accident Date`)) %>%
  group_by(Year) %>%
  summarise(Count = n())

yearly_data$Year <- as.factor(yearly_data$Year)

ggplot(yearly_data, aes(x = Year, y = Count, fill = Year)) +
  geom_bar(stat = "identity") +
  labs(title = "Total Accidents By Year",
       x = "Year",
       y = "Count",
       fill = "Year") +
  scale_fill_discrete(name = "Year") +
  theme_minimal()
```



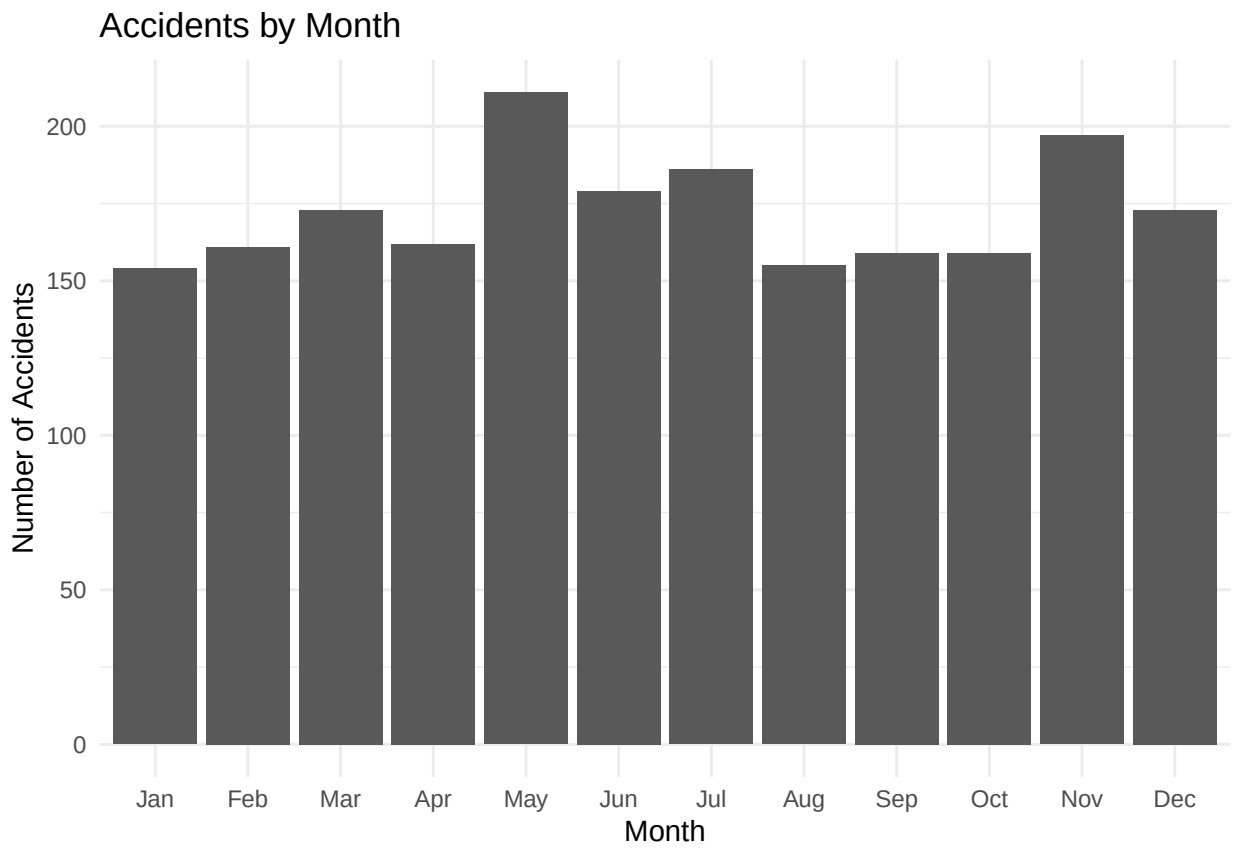
Task 23:Displaying total accidents by year in line graph.

```
ggplot(yearly_data, aes(x = Year, y = Count, group = 1)) +
  geom_line(color = "blue") +
  geom_point(color = "red") +
  labs(title = "Total Accidents By Year",
       x = "Year",
       y = "Count") +
  theme_minimal()
```



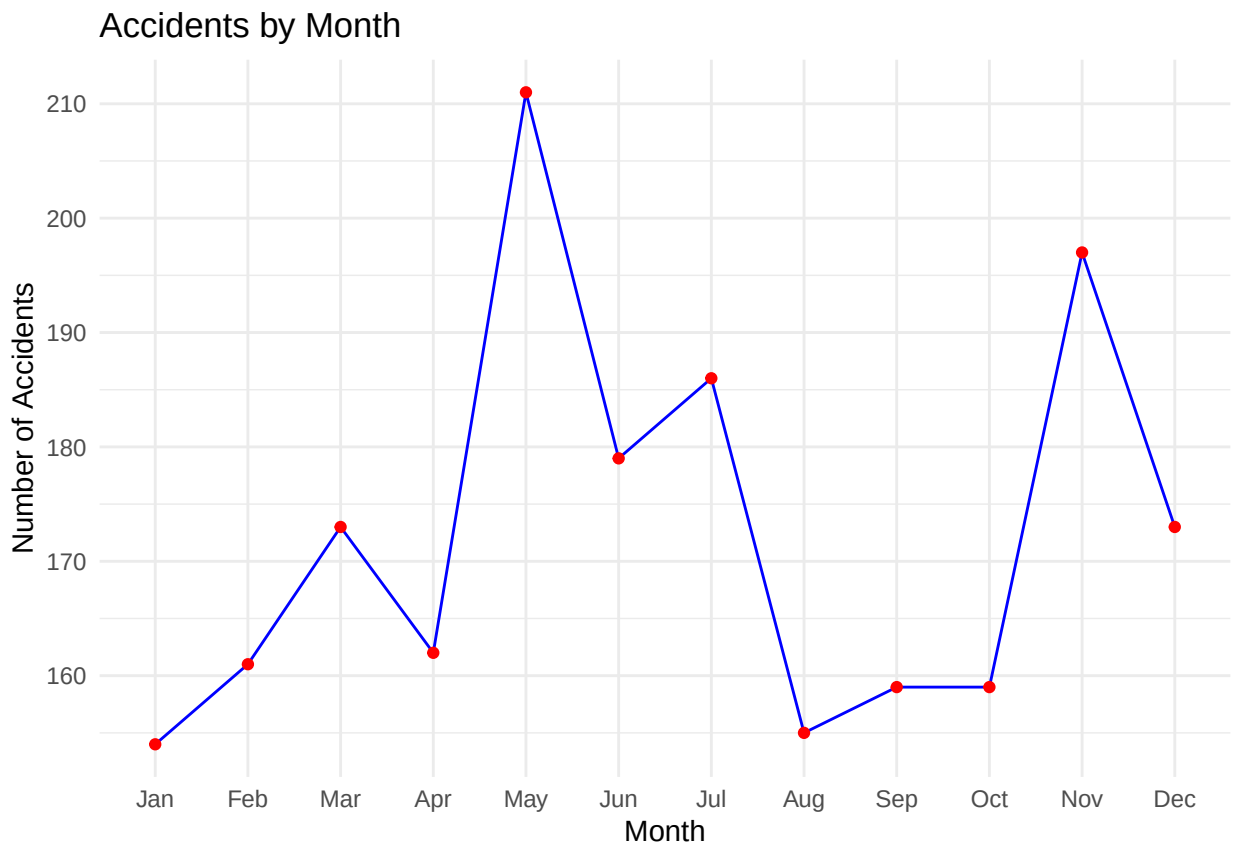
Task 24:Displaying total accidents by month in bar graph.

```
ggplot(data, aes(x = month(`Accident Date`, label = TRUE))) +
  geom_bar(stat = "count") +
  labs(title = "Accidents by Month", x = "Month", y = "Number of Accidents") +
  theme_minimal()
```



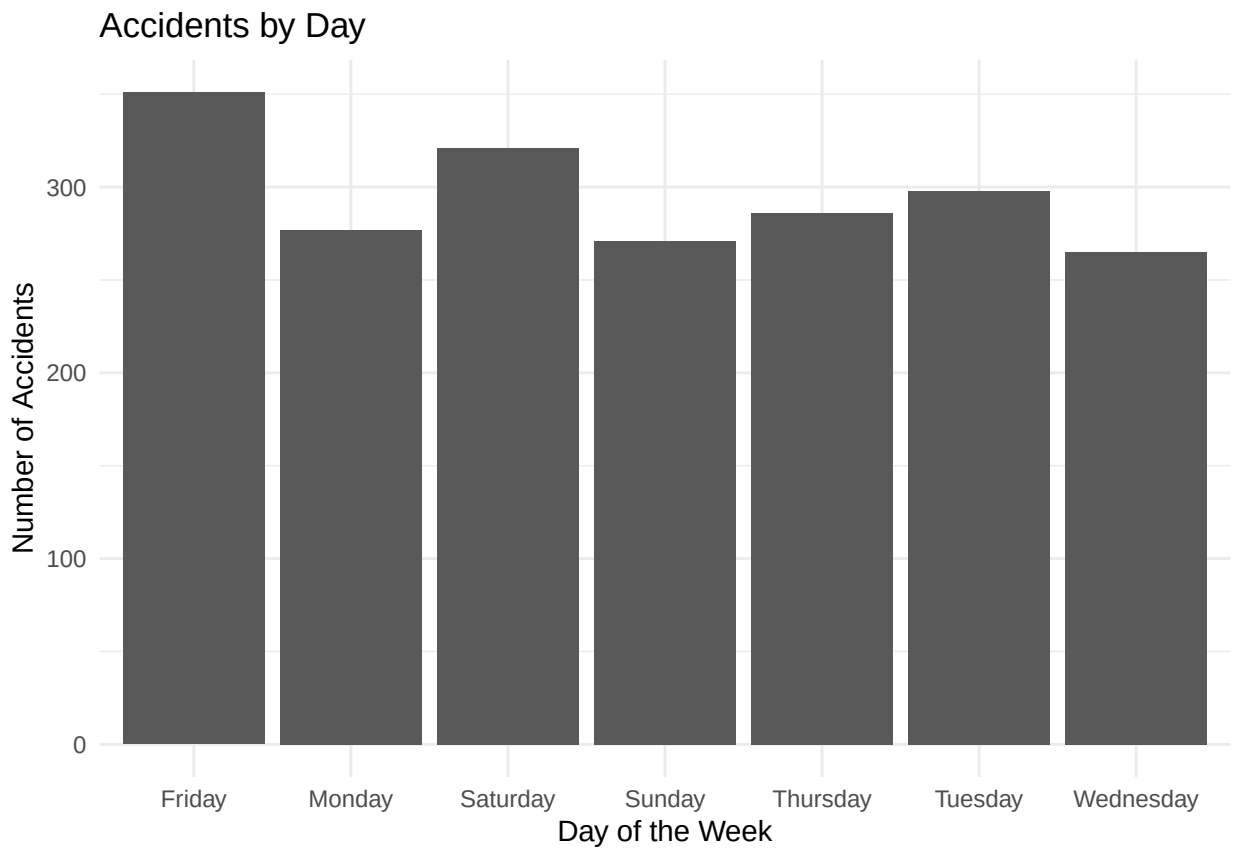
Task 25:Displaying total accidents by month in line graph.

```
monthly_data <- data %>%
  mutate(Month = month(`Accident Date`, label = TRUE)) %>%
  group_by(Month) %>%
  summarize(Accident_Count = n())

# Create the line plot
ggplot(monthly_data, aes(x = Month, y = Accident_Count, group = 1)) +
  geom_line(color = "blue") +
  geom_point(color = "red") +
  labs(title = "Accidents by Month", x = "Month", y = "Number of Accidents") +
  theme_minimal()
```



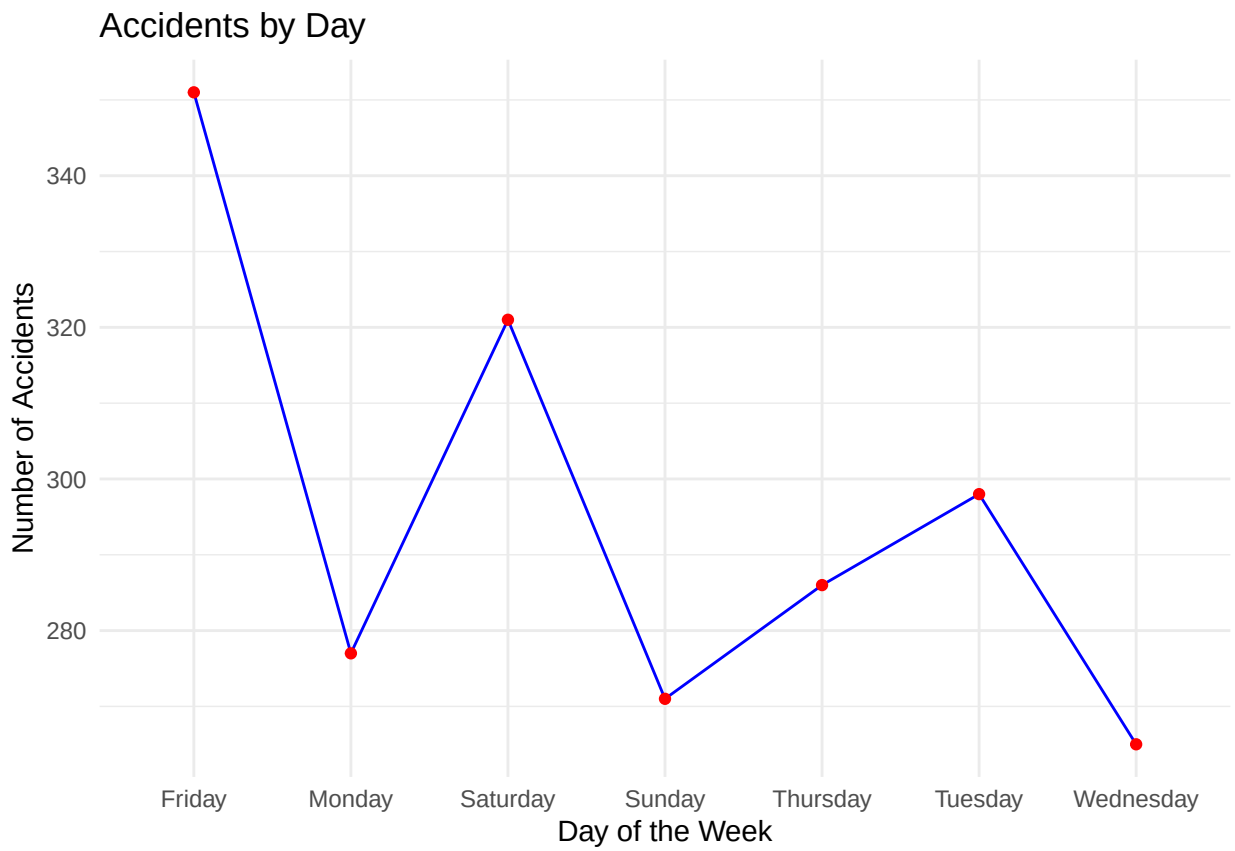
Task 26:Displaying total accidents by day in bar graph.

```
ggplot(data, aes(x = weekdays(`Accident Date`))) +
  geom_bar(stat = "count") +
  labs(title = "Accidents by Day ", x = "Day of the Week", y = "Number of Accidents") +
  theme_minimal()
```



Task 27:Displaying accidents by day in line graph.

```
weekly_data <- data %>%
  mutate(Day = weekdays(`Accident Date`)) %>%
  group_by(Day) %>%
  summarize(Accident_Count = n()) %>%
  arrange(match(Day, c("Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "Sunday")))

# Create the line plot
ggplot(weekly_data, aes(x = Day, y = Accident_Count, group = 1)) +
  geom_line(color = "blue") +
  geom_point(color = "red") +
  labs(title = "Accidents by Day ", x = "Day of the Week", y = "Number of Accidents") +
  theme_minimal()
```



PART 3: Regression

Task 1:Converting specified columns to factors

```
data$`Casualty Class` <- as.factor(data$`Casualty Class`)
data$`Casualty Severity` <- as.factor(data$`Casualty Severity`)
data$`Type of Vehicle` <- as.factor(data$`Type of Vehicle`)
data$`Weather Conditions` <- as.factor(data$`Weather Conditions`)
```

Task 2: Filter out rows with missing values in 'Age of Casualty'

```
train_data <- data %>%
  filter(!is.na(`Age of Casualty`))
```

Task 3:Building a linear regression model using specified predictors

```
lm_model <- lm(`Age of Casualty` ~ `Casualty Class` + `Casualty Severity`
               + `Type of Vehicle` + `Weather Conditions`, data = train_data)
```

Task 4:Print the summary of the linear model

```r
print(summary(lm_model))
```

```
##
## Call:
## lm(formula = `Age of Casualty` ~ `Casualty Class` + `Casualty Severity` +
##     `Type of Vehicle` + `Weather Conditions`, data = train_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -41.011 -13.836  -3.836  11.372  74.470
##
## Coefficients:
##                                                                    Estimate
## (Intercept)                                                         23.5161
## `Casualty Class`Pedestrian                                          -5.7415
## `Casualty Class`Vehicle or pillion passenger                       -10.3380
## `Casualty Severity`Serious                                          -0.7909
## `Casualty Severity`Slight                                           -3.6200
## `Type of Vehicle`Agricultural vehicle (includes diggers etc.)        8.0000
## `Type of Vehicle`Bus or coach (17 or more passenger seats)          35.9231
## `Type of Vehicle`Car                                                23.8357
## `Type of Vehicle`Goods vehicle 3.5 tonnes mgw and under             24.4110
## `Type of Vehicle`Goods vehicle 7.5 tonnes mgw and over              35.0714
## `Type of Vehicle`Goods vehicle over 3.5 tonnes and under 7.5 tonnes mgw  40.8376
## `Type of Vehicle`M/cycle 50cc and under                              8.3874
## `Type of Vehicle`Minibus (8 - 16 passenger seats)                   28.8277
## `Type of Vehicle`Mobility Scooter                                   34.0000
## `Type of Vehicle`Motorcycle - Unknown CC                            64.7415
## `Type of Vehicle`Motorcycle over 125cc and up to 500cc              21.4966
## `Type of Vehicle`Motorcycle over 500cc                              26.2123
## `Type of Vehicle`Motorcycle over 50cc and up to 125cc               11.7457
## `Type of Vehicle`Other Vehicle                                      19.5290
## `Type of Vehicle`Pedal cycle                                        19.6257
## `Type of Vehicle`Ridden horse                                       15.0000
## `Type of Vehicle`Taxi/Private hire car                              26.1783
## `Type of Vehicle`Tram / Light rail                                  12.8920
## `Weather Conditions`Fine without high winds                         -3.8961
## `Weather Conditions`Fog or mist ? if hazard                         -1.8089
## `Weather Conditions`Other                                          -15.0982
## `Weather Conditions`Raining with high winds                         -7.2888
## `Weather Conditions`Raining without high winds                      -6.0312
## `Weather Conditions`Snowing with high winds                         -2.6778
## `Weather Conditions`Snowing without high winds                      -1.7084
## `Weather Conditions`Unknown                                         -7.3029
##                                                                    Std. Error
## (Intercept)                                                           19.8540
## `Casualty Class`Pedestrian                                             1.1744
## `Casualty Class`Vehicle or pillion passenger                           1.0890
## `Casualty Severity`Serious                                             3.9531
## `Casualty Severity`Slight                                              3.8349
## `Type of Vehicle`Agricultural vehicle (includes diggers etc.)         23.1602
## `Type of Vehicle`Bus or coach (17 or more passenger seats)            19.1878
## `Type of Vehicle`Car                                                  18.9229
```

```
## `Type of Vehicle`Goods vehicle 3.5 tonnes mgw and under                    19.0647
## `Type of Vehicle`Goods vehicle 7.5 tonnes mgw and over                     19.3630
## `Type of Vehicle`Goods vehicle over 3.5 tonnes and under 7.5 tonnes mgw    20.0699
## `Type of Vehicle`M/cycle 50cc and under                                    19.3593
## `Type of Vehicle`Minibus (8 - 16 passenger seats)                          21.8497
## `Type of Vehicle`Mobility Scooter                                          26.7431
## `Type of Vehicle`Motorcycle - Unknown CC                                   26.7689
## `Type of Vehicle`Motorcycle over 125cc and up to 500cc                     19.3669
## `Type of Vehicle`Motorcycle over 500cc                                     19.0772
## `Type of Vehicle`Motorcycle over 50cc and up to 125cc                      19.0601
## `Type of Vehicle`Other Vehicle                                             19.9500
## `Type of Vehicle`Pedal cycle                                               18.9793
## `Type of Vehicle`Ridden horse                                              23.1602
## `Type of Vehicle`Taxi/Private hire car                                     19.0439
## `Type of Vehicle`Tram / Light rail                                         21.8478
## `Weather Conditions`Fine without high winds                                 4.6245
## `Weather Conditions`Fog or mist ? if hazard                                 6.4226
## `Weather Conditions`Other                                                   7.1508
## `Weather Conditions`Raining with high winds                                 5.1779
## `Weather Conditions`Raining without high winds                              4.7686
## `Weather Conditions`Snowing with high winds                                 8.1340
## `Weather Conditions`Snowing without high winds                              6.6045
## `Weather Conditions`Unknown                                                 6.3303
##                                                                          t value
## (Intercept)                                                                1.184
## `Casualty Class`Pedestrian                                                -4.889
## `Casualty Class`Vehicle or pillion passenger                              -9.493
## `Casualty Severity`Serious                                                -0.200
## `Casualty Severity`Slight                                                 -0.944
## `Type of Vehicle`Agricultural vehicle (includes diggers etc.)              0.345
## `Type of Vehicle`Bus or coach (17 or more passenger seats)                 1.872
## `Type of Vehicle`Car                                                       1.260
## `Type of Vehicle`Goods vehicle 3.5 tonnes mgw and under                    1.280
## `Type of Vehicle`Goods vehicle 7.5 tonnes mgw and over                     1.811
## `Type of Vehicle`Goods vehicle over 3.5 tonnes and under 7.5 tonnes mgw    2.035
## `Type of Vehicle`M/cycle 50cc and under                                    0.433
## `Type of Vehicle`Minibus (8 - 16 passenger seats)                          1.319
## `Type of Vehicle`Mobility Scooter                                          1.271
## `Type of Vehicle`Motorcycle - Unknown CC                                   2.419
## `Type of Vehicle`Motorcycle over 125cc and up to 500cc                     1.110
## `Type of Vehicle`Motorcycle over 500cc                                     1.374
## `Type of Vehicle`Motorcycle over 50cc and up to 125cc                      0.616
## `Type of Vehicle`Other Vehicle                                             0.979
## `Type of Vehicle`Pedal cycle                                               1.034
## `Type of Vehicle`Ridden horse                                              0.648
## `Type of Vehicle`Taxi/Private hire car                                     1.375
## `Type of Vehicle`Tram / Light rail                                         0.590
## `Weather Conditions`Fine without high winds                               -0.842
## `Weather Conditions`Fog or mist ? if hazard                               -0.282
## `Weather Conditions`Other                                                 -2.111
## `Weather Conditions`Raining with high winds                               -1.408
## `Weather Conditions`Raining without high winds                            -1.265
## `Weather Conditions`Snowing with high winds                               -0.329
## `Weather Conditions`Snowing without high winds                            -0.259
```

```
## `Weather Conditions`Unknown                                                  -1.154
##                                                                            Pr(>|t|)
## (Intercept)                                                                  0.2364
## `Casualty Class`Pedestrian                                                1.09e-06
## `Casualty Class`Vehicle or pillion passenger                               < 2e-16
## `Casualty Severity`Serious                                                   0.8415
## `Casualty Severity`Slight                                                    0.3453
## `Type of Vehicle`Agricultural vehicle (includes diggers etc.)                0.7298
## `Type of Vehicle`Bus or coach (17 or more passenger seats)                   0.0613
## `Type of Vehicle`Car                                                         0.2080
## `Type of Vehicle`Goods vehicle 3.5 tonnes mgw and under                      0.2005
## `Type of Vehicle`Goods vehicle 7.5 tonnes mgw and over                       0.0702
## `Type of Vehicle`Goods vehicle over 3.5 tonnes and under 7.5 tonnes mgw      0.0420
## `Type of Vehicle`M/cycle 50cc and under                                      0.6649
## `Type of Vehicle`Minibus (8 - 16 passenger seats)                            0.1872
## `Type of Vehicle`Mobility Scooter                                            0.2037
## `Type of Vehicle`Motorcycle - Unknown CC                                     0.0157
## `Type of Vehicle`Motorcycle over 125cc and up to 500cc                       0.2671
## `Type of Vehicle`Motorcycle over 500cc                                       0.1696
## `Type of Vehicle`Motorcycle over 50cc and up to 125cc                        0.5378
## `Type of Vehicle`Other Vehicle                                               0.3277
## `Type of Vehicle`Pedal cycle                                                 0.3012
## `Type of Vehicle`Ridden horse                                                0.5173
## `Type of Vehicle`Taxi/Private hire car                                       0.1694
## `Type of Vehicle`Tram / Light rail                                           0.5552
## `Weather Conditions`Fine without high winds                                  0.3996
## `Weather Conditions`Fog or mist ? if hazard                                  0.7782
## `Weather Conditions`Other                                                    0.0349
## `Weather Conditions`Raining with high winds                                  0.1594
## `Weather Conditions`Raining without high winds                               0.2061
## `Weather Conditions`Snowing with high winds                                  0.7420
## `Weather Conditions`Snowing without high winds                               0.7959
## `Weather Conditions`Unknown                                                  0.2488
##
## (Intercept)
## `Casualty Class`Pedestrian                                                ***
## `Casualty Class`Vehicle or pillion passenger                              ***
## `Casualty Severity`Serious
## `Casualty Severity`Slight
## `Type of Vehicle`Agricultural vehicle (includes diggers etc.)
## `Type of Vehicle`Bus or coach (17 or more passenger seats)                .
## `Type of Vehicle`Car
## `Type of Vehicle`Goods vehicle 3.5 tonnes mgw and under
## `Type of Vehicle`Goods vehicle 7.5 tonnes mgw and over                     .
## `Type of Vehicle`Goods vehicle over 3.5 tonnes and under 7.5 tonnes mgw    *
## `Type of Vehicle`M/cycle 50cc and under
## `Type of Vehicle`Minibus (8 - 16 passenger seats)
## `Type of Vehicle`Mobility Scooter
## `Type of Vehicle`Motorcycle - Unknown CC                                   *
## `Type of Vehicle`Motorcycle over 125cc and up to 500cc
## `Type of Vehicle`Motorcycle over 500cc
## `Type of Vehicle`Motorcycle over 50cc and up to 125cc
## `Type of Vehicle`Other Vehicle
## `Type of Vehicle`Pedal cycle
```

```
## `Type of Vehicle`Ridden horse
## `Type of Vehicle`Taxi/Private hire car
## `Type of Vehicle`Tram / Light rail
## `Weather Conditions`Fine without high winds
## `Weather Conditions`Fog or mist ? if hazard
## `Weather Conditions`Other                                          *
## `Weather Conditions`Raining with high winds
## `Weather Conditions`Raining without high winds
## `Weather Conditions`Snowing with high winds
## `Weather Conditions`Snowing without high winds
## `Weather Conditions`Unknown
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 18.91 on 2019 degrees of freedom
## Multiple R-squared:  0.0784, Adjusted R-squared:  0.06471
## F-statistic: 5.725 on 30 and 2019 DF,  p-value: < 2.2e-16
```

Task 5:Extracting and print the R-squared value

```
r_squared <- summary(lm_model)$r.squared
cat("R-squared value:", r_squared, "\n")
```

```
## R-squared value: 0.07840102
```

Task 6: Filtering out rows with missing values in 'Age of Casualty'

```
missing_age_data <- data %>%
  filter(is.na(`Age of Casualty`))
```

Task 7: Predicting the 'Age of Casualty' for missing values using the linear model

```
predict_missing_age <- function(model, missing_age_data) {
  predict(model, newdata = missing_age_data)
}

predicted_age <- predict_missing_age(lm_model, missing_age_data)
```

Task 8:Imputing the predicted ages into the original data

```
data$`Age of Casualty`[is.na(data$`Age of Casualty`)] <- predicted_age
predicted_age
```

```
##        1        2        3        4        5        6        7        8
## 34.09413 39.83566 34.09413 27.21662 36.92332 37.49655 39.83566 34.09413
##        9       10       11       12       13       14       15       16
## 34.09413 39.83566 39.83566 30.19174 29.49765 26.10491 37.49655 34.09413
##       17       18       19
## 42.90643 35.62574 34.78822
```

Task 9:Rounding the imputed age and convert to integer type

```r
data <- data %>%
  mutate(`Age of Casualty` = round(`Age of Casualty`)) %>%
  mutate(`Age of Casualty` = as.integer(`Age of Casualty`))
```

Task 10:Check for any remaining missing values in the dataset

```r
check_missing_values <- function(data) {
  colSums(is.na(data))
}

missing_values <- check_missing_values(data)
cat("Columns with Missing Values:\n")
```

```
## Columns with Missing Values:
```

```r
print(missing_values[missing_values > 0])
```

```
## named numeric(0)
```

Task 11:Printing the dimensions of the dataset

```r
cat("Dataset dimensions:", dim(data), "\n")
```

```
## Dataset dimensions: 2069 13
```

Task 12:Viewing the data dataset

```r
view(data)
```

Task 13:Writing the modified data to a CSV file

```r
write.csv(data, "regression.csv", row.names = FALSE, quote = FALSE, fileEncoding = "UTF-8")
```