# trafficCrash

Frank Guo

2024-04-15

## Milestone 2 - Crashes Analysis

**By Xiaodong Guo**

### 1. Goal.

Our project has a significant objective- to construct models identifying the pivotal factors contributing to severe crashes. This crucial task is based on the data provided by the New Zealand Transport Agency(NZTA).

### 2. Data Source.

The original data, meticulously collected, originated from the Waka Kotahi NZ Transport Agency's open data portal(the tutor provided the link in the assignment piece). We specifically downloaded the dataset named "Crash Analysis System (CAS) data" from the "Crash" catalogue, which encompasses all traffic crashes reported to us by the NZ Police. The data format is a "CVS" file. It was created on 3/25/2020 and last updated on 3/14/2024.

The data includes crash datas from 2000 to 2023.

### 3. Data Processing.

```
#load data

library(tidyverse)
library(xgboost)

#file_path <- file.choose()
set.seed(230)
data <- read.csv("../data/Crash_Analysis_System_(CAS)_data.csv", header = TRUE, sep = ",")

dim(data)
```

```
## [1] 821744     72
```

```
#glimpse(data)
```

We load the data from csv flie. The dataset we got have 72 columns,and 821744 rows.The first things we can do is to drop columns not related to our objective.Thus, we select following columns by common sense of mine(not sure if 100% correct,maybe need some hint from Lisa Chen).There are also have some description columns, that is long string values to desript an event or street name. That looks no sense,should drop them too. Like "crashLocation1","crashLocation2".Than drop columns with almost all values are Null,that is crashRoadSideRoad" "intersection".
Define crashSeverity == "Fatal Crash" and crashSeverity == "Serious Crash" as severe crashes given numeric value 1,

Define crashSeverity == "Minor Crash" | crashSeverity == "Non-Injury Crash" as not severe crashes given numeric value 0.
The "crashSeverity" Label will be the target label.

Predictors like "weatherA" and "weatherB" ,could be combined as one attribute "weather",much easier to display and dealing with it later.

Replace the ""","Null","None" value in region with "Other",replace the " " in other character attributes with "others".

List all the attributes with Na value: "advisorySpeed" "bicycle" "bridge" "bus" "carStationWagon"
"cliffBank" "ditch" "fence" "guardRail" "houseOrBuilding"
"kerb" "moped" "motorcycle" "NumberOfLanes" "otherVehicleType"
"overBank" "parkedVehicle" "pedestrian" "postOrPole" "roadworks"
"schoolBus" "slipOrFlood" "speedLimit" "strayAnimal" "suv"
"taxi" "temporarySpeedLimit" "trafficIsland" "trafficSign" "train"
"tree" "truck" "unknownVehicleType" "vanOrUtility" "vehicle"
"waterRiver"
They are all numberic, and fill with 0 is reasonable.

The whole dataset looks categorical.So we convert all attributes in to categorical attributes.

```
#clean data


columns_to_drop <- c("X","Y","OBJECTID","areaUnitID","crashDirectionDescription","","crashDistance","tla
"crashFinancialYear","fatalCount","debris","meshblockId","northing","easting","objectThrownOrDropped","r
"otherObject","phoneBoxEtc","seriousInjuryCount")

data <- select(data, -one_of(columns_to_drop))

na_percentage <- colMeans(is.na(data))
columns_with_high_na <- names(na_percentage[na_percentage > 0.99])
#print(columns_with_high_na)

data <- data %>% select(-columns_with_high_na)

#table(data$crashSeverity)

data <- data %>%
  mutate(crashSeverity = ifelse(crashSeverity == "Fatal Crash" | crashSeverity == "Serious Crash", 1,
                    ifelse(crashSeverity == "Minor Crash" | crashSeverity == "Non-Injury Crash", 0,
                           2))) %>%  filter(crashSeverity != 2)

#table(data$crashSeverity)

#table(data$weatherA)
#table(data$weatherB)

data$weatherA <- ifelse(data$weatherA %in% c("None", "Null"), "Other", data$weatherA)
data$weatherB<- ifelse(data$weatherB %in% c("None", "Null"), "", data$weatherB)

data <- data %>% unite(weatherA,weatherB,col=weather,sep=" ")


table(data$region)
```

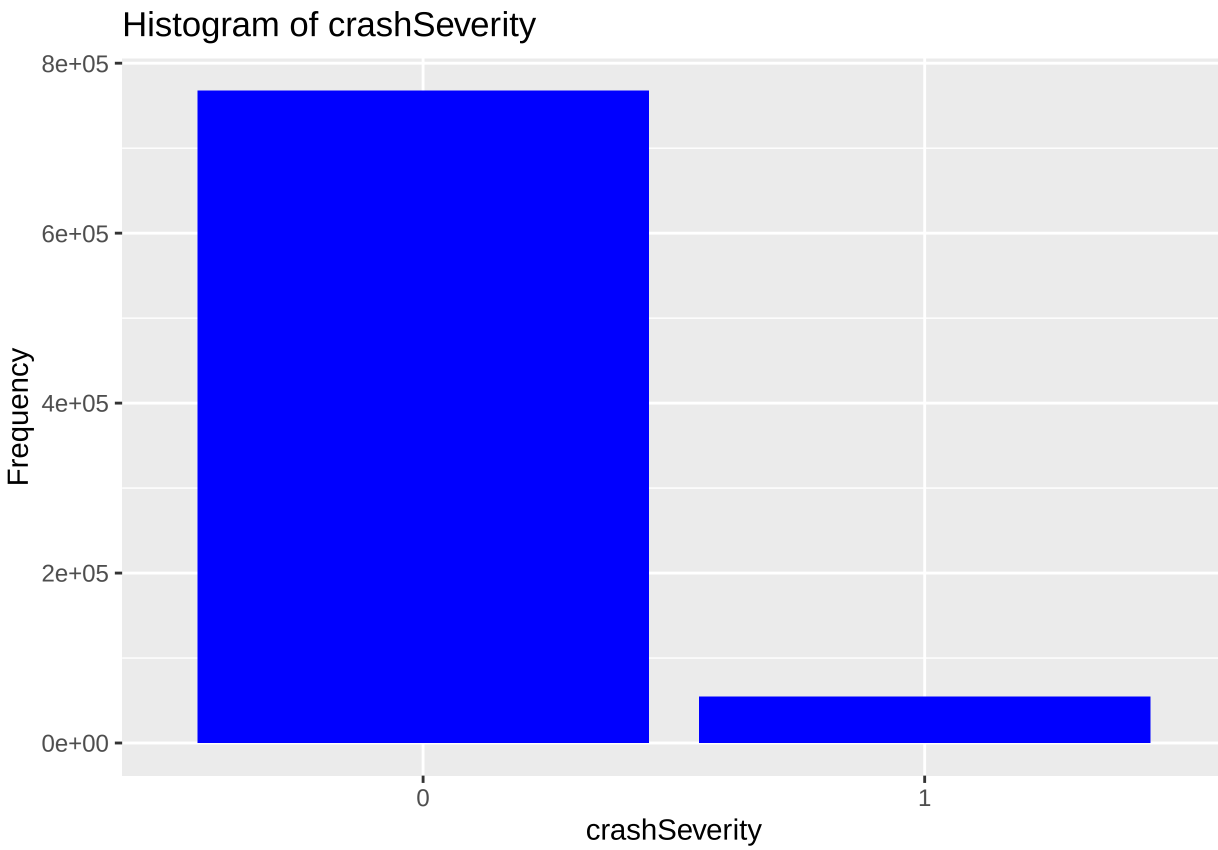
```
##
##                                          Auckland Region      Bay of Plenty Region
##                     3188                          285346                     47177
##         Canterbury Region                Gisborne Region         Hawke's Bay Region
##                    82146                            9784                     32388
## Manawatū-Whanganui Region             Marlborough Region             Nelson Region
##                    46329                            8266                      8076
##          Northland Region                   Otago Region          Southland Region
##                    33299                           44574                     20234
##           Taranaki Region                  Tasman Region            Waikato Region
##                    18604                            7541                     87849
##         Wellington Region              West Coast Region
##                    79725                            7218
```

```
data$region <- ifelse(data$region %in% c("None", "Null"), "Other", data$region)

na_columns <- sapply(data, function(x) any(is.na(x)))

columns_with_na <- names(data)[na_columns]
#print(columns_with_na)

data <- data %>%
  mutate_at(vars(one_of(columns_with_na)), ~replace_na(., 0))


processed_data <- data
processed_data[processed_data == ""] <- "others"

processed_data[] <- lapply(processed_data, function(x) as.factor(x))
data <- processed_data

rm(processed_data)

#glimpse(data)
```

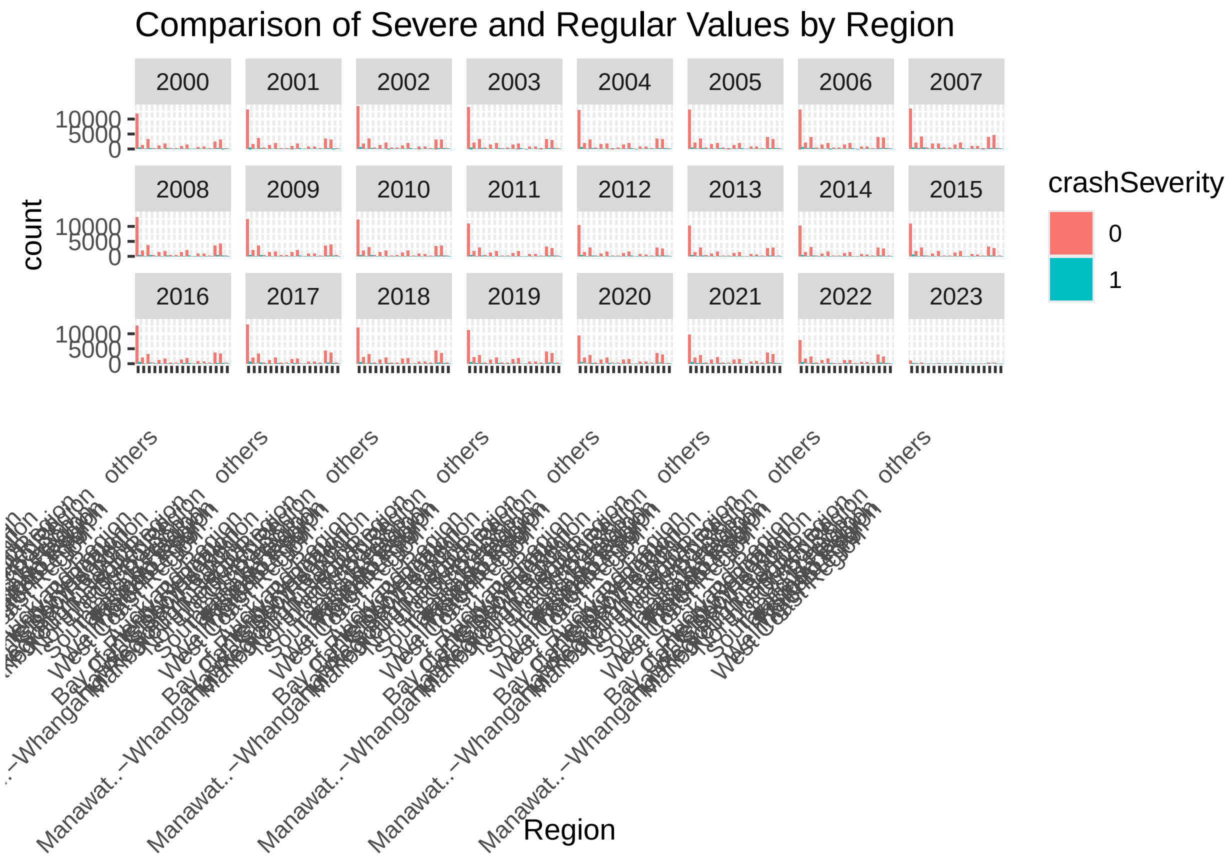
## 4. Data Exploration

Because all the attributes are categorical attributes, we decide to explore the freqency the most important attributes.
We Plot the target labels and some relations between attributes like crashYear, region, weather and light which is intuitively think that will be key attribute of the cause of incident.
We Can see total crashes decreased by year, but servere crashes not.Also, we found the problem that unbalanced data occurs in predictors.For examples, regarding as weather, we can see most of the crashes(severe and regular) happened in fine day rather than the other weather conditions.
From All of those plots, we found the unbalance of the data.There are 767290 regular crashes,but only 54454 severe crashes. Our target is to find the cause of severe crashes,but the severe observations' size is really small compares to the regular crashes.

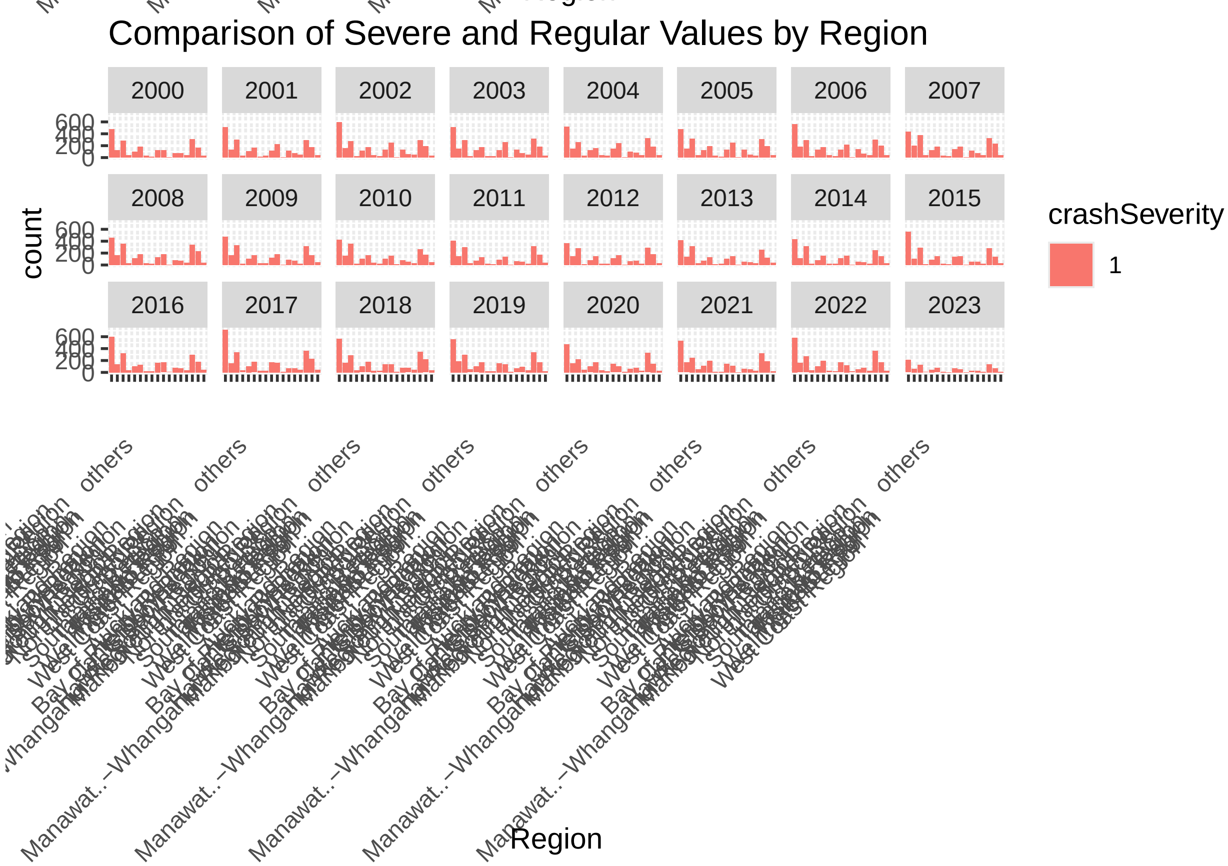
| Var1 | Freq |
|---|---|
| 0 | 767290 |
| 1 | 54454 |




```
## `summarise()` has grouped output by 'crashYear', 'region'. You can override
## using the `.groups` argument.
```

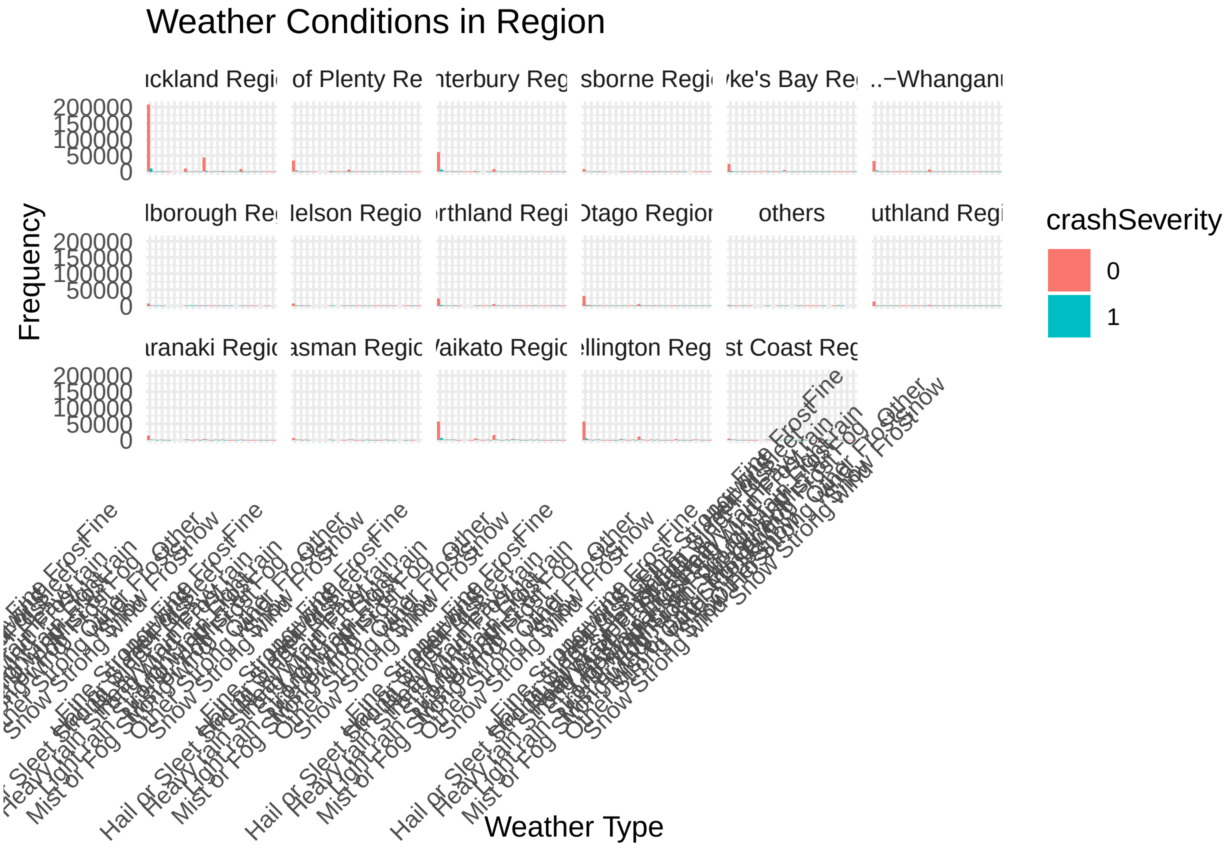


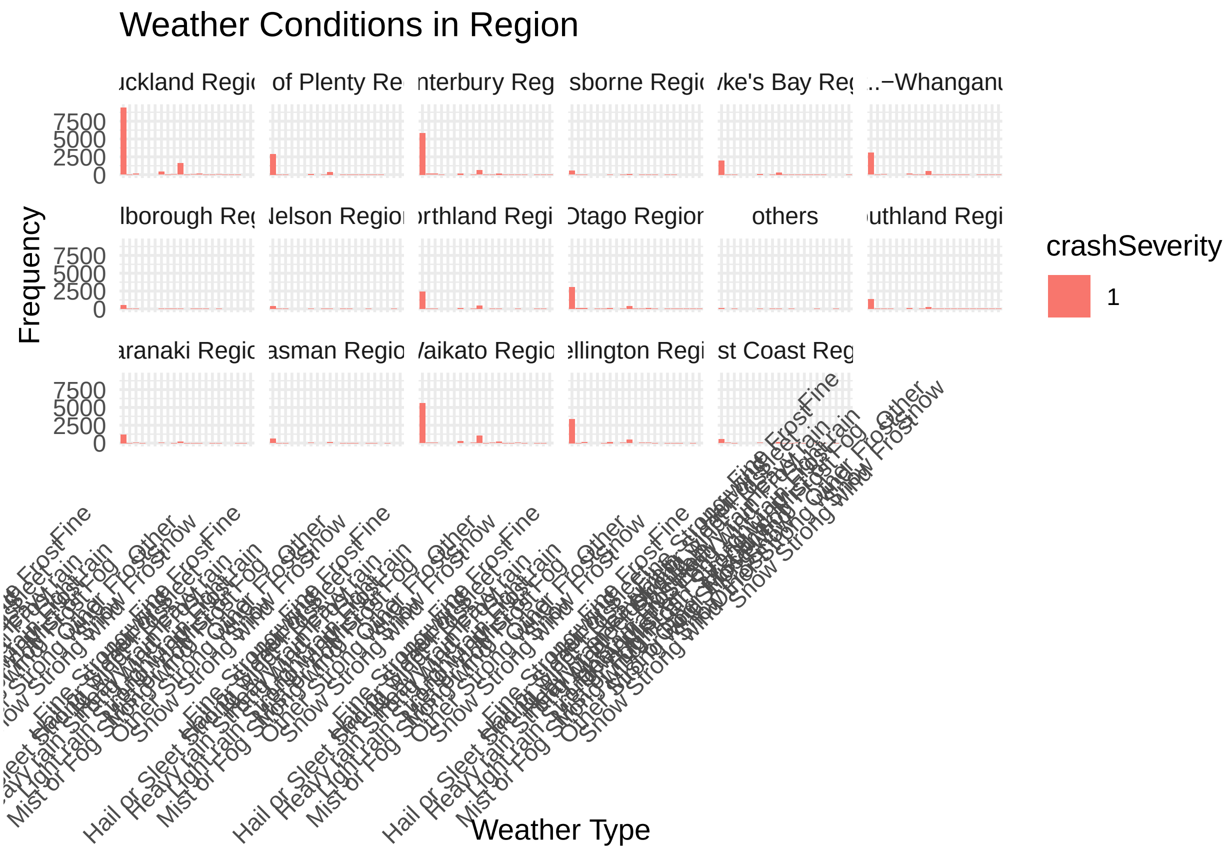
Comparison of Severe and Regular Values by Region
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
count
crashSeverity
0
1
others
Region

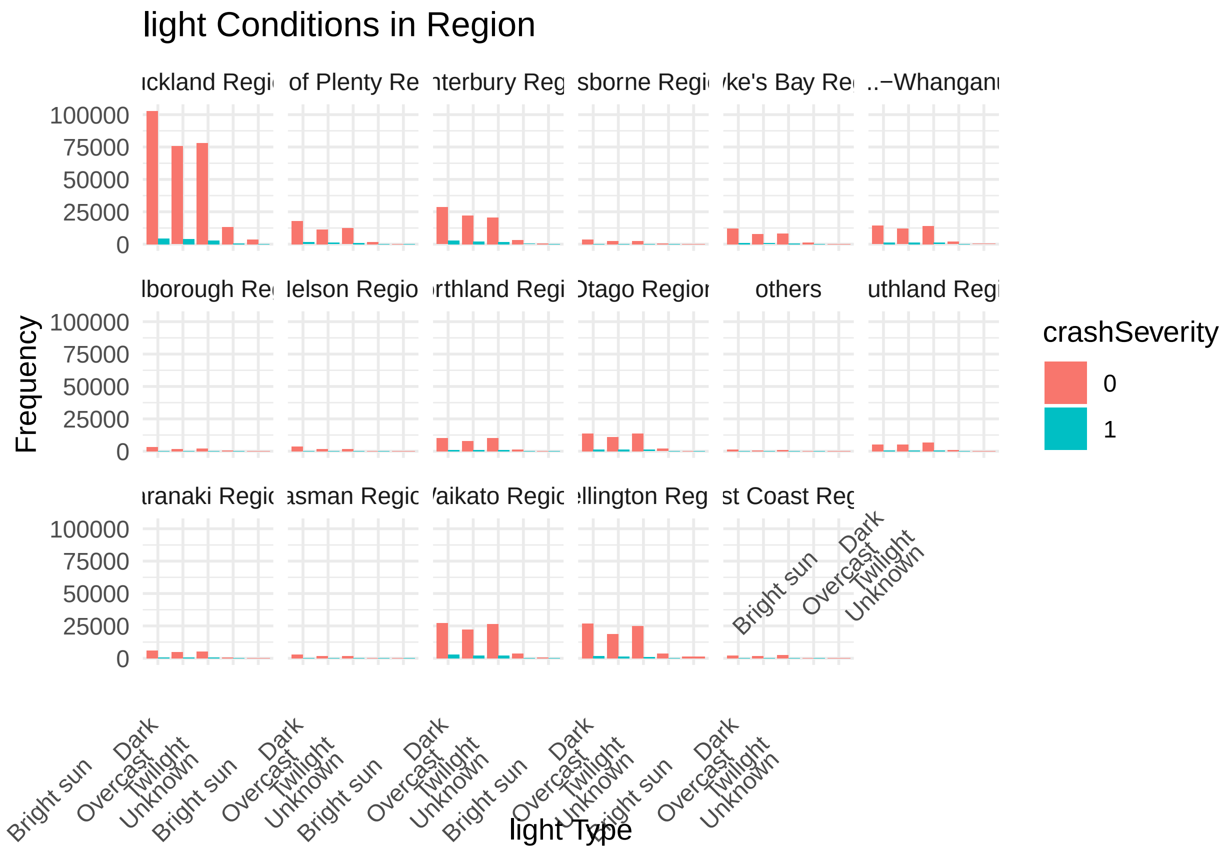
Comparison of Severe and Regular Values by Region
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
count
crashSeverity
1
others
Region

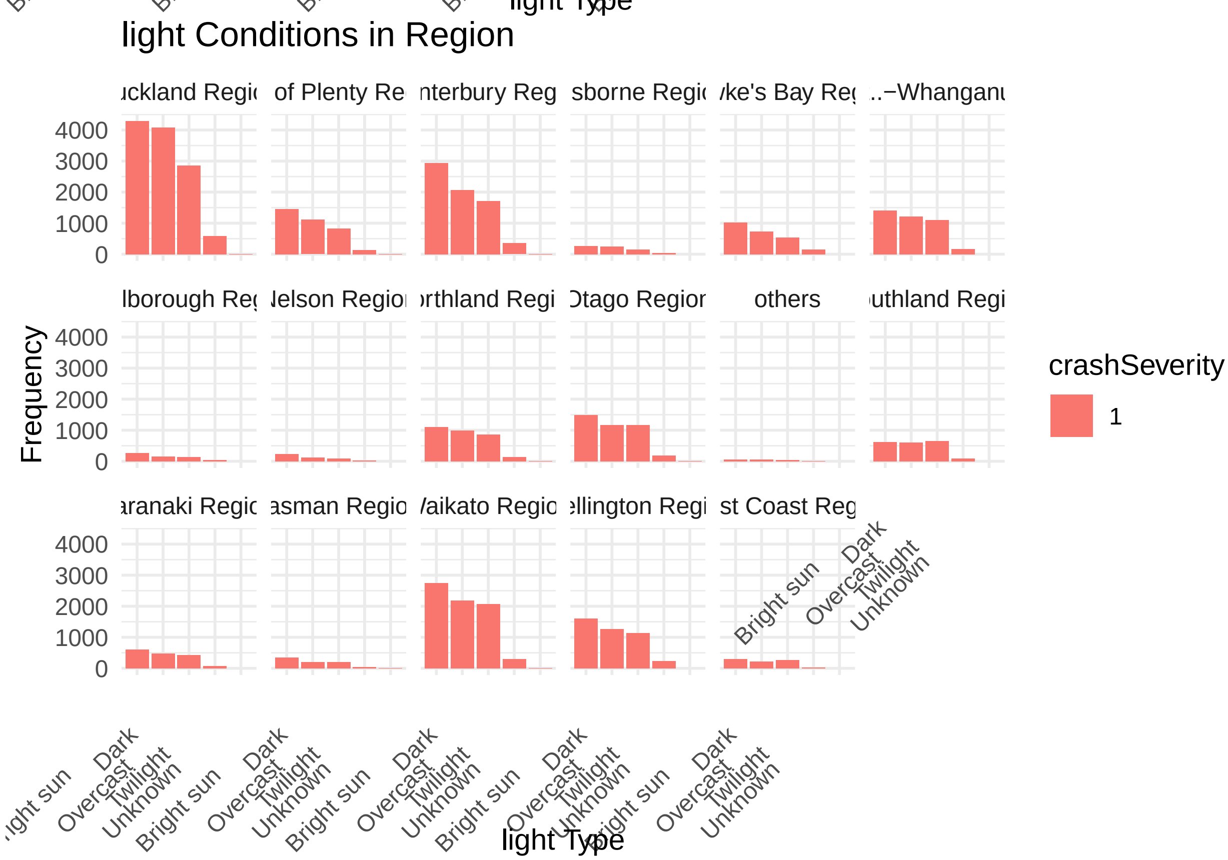
```
## `summarise()` has grouped output by 'region', 'weather'. You can override using
## the `.groups` argument.
```






```
## `summarise()` has grouped output by 'region', 'light'. You can override using
## the `.groups` argument.
```


light Conditions in Region
crashSeverity
0
1
Frequency
light Type
Bright sun
Overcast
Dark
Twilight
Unknown
others


light Conditions in Region
crashSeverity
1
Frequency
light Type
Bright sun
Overcast
Dark
Twilight
Unknown
others

## 5. Analytical Plan

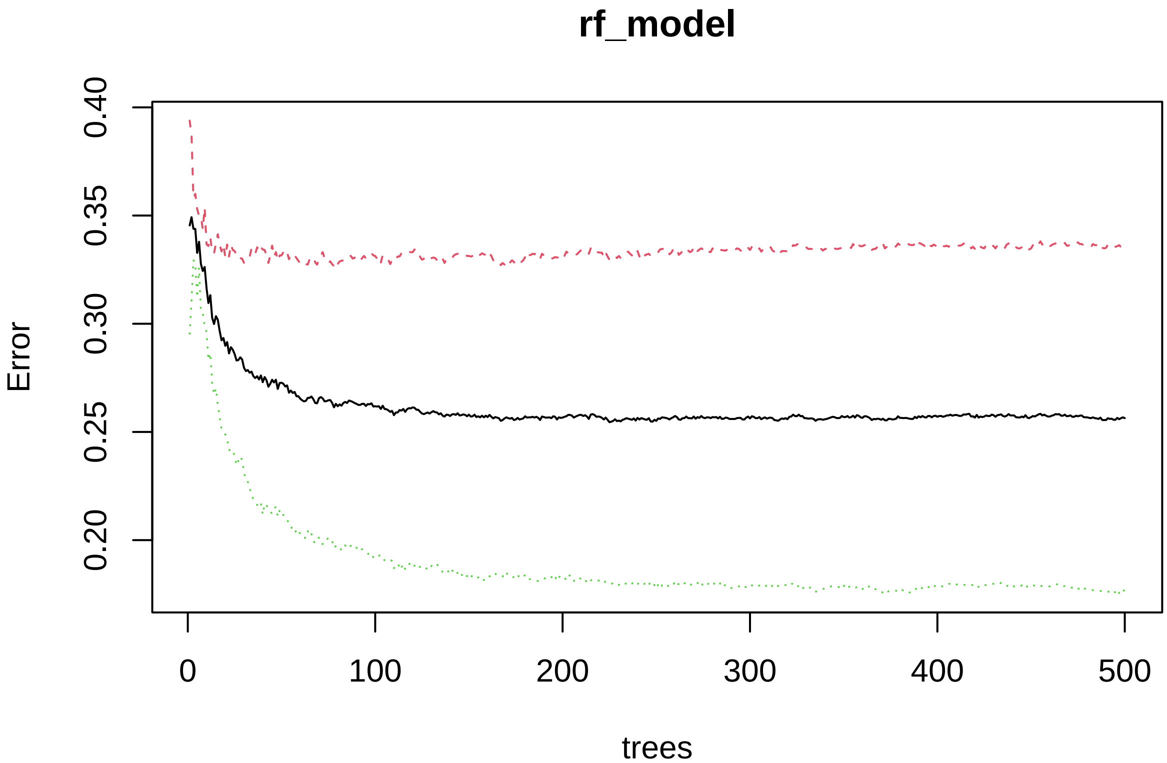
Because of the unbalanc of the dataset. We decisied to equally sample from severe crashes and regular crashes. Because it is catalog dataset, we will use ensemble based on decision tree. Random forest is my prefer to fit the data. We will use 80% of the sample data as training data, and 20% of the sample data as test data. After fit, Finding the most significant predictor in a Random Forest model.That involves analyzing the importance scores of the predictors. Random Forest calculates the importance of each predictor by measuring how much the accuracy of the model decreases when the values of that predictor are randomly permuted while keeping other variables constant. The larger the decrease in accuracy, the more important the predictor is considered.

```
#sample data

# Split data by class
class_data <- split(data, data$crashSeverity)

# Determine desired sample size (e.g., proportionally to the original class distribution)
desired_sample_size <- 5000  # Adjust as needed

# Sample each class
sampled_data <- lapply(class_data, function(class_subset) {
  # Determine the sample size for this class
  class_size <- nrow(class_subset)
  class_sample_size <- min(class_size, desired_sample_size)

  # Sample observations from this class
  sampled_indices <- sample(1:class_size, size = class_sample_size, replace = TRUE)

  # Return the sampled subset
  return(class_subset[sampled_indices, ])
})

# Combine sampled subsets
balanced_data <- do.call(rbind, sampled_data)
```

### fit a random forest model

```
#fit a model and test

training_idx <- sample(nrow(balanced_data), nrow(balanced_data)*0.8)
test_idx <-(1:nrow(balanced_data))[-training_idx]

training_data <- balanced_data[training_idx,]
test_data <- balanced_data[test_idx,]

#train_Y <- training_data$crashSeverity

#train_X <- training_data %>% select(-crashSeverity)


test_Y <- test_data$crashSeverity
test_X <- test_data %>% select(-crashSeverity)

library(randomForest)

rf_model <- randomForest(crashSeverity ~ ., data = training_data)
```

```r
plot(rf_model)
```




```r
predictions <- predict(rf_model, newdata = test_X)

accuracy <- mean(predictions == test_Y)

print(accuracy)
```

```
## [1] 0.734
```

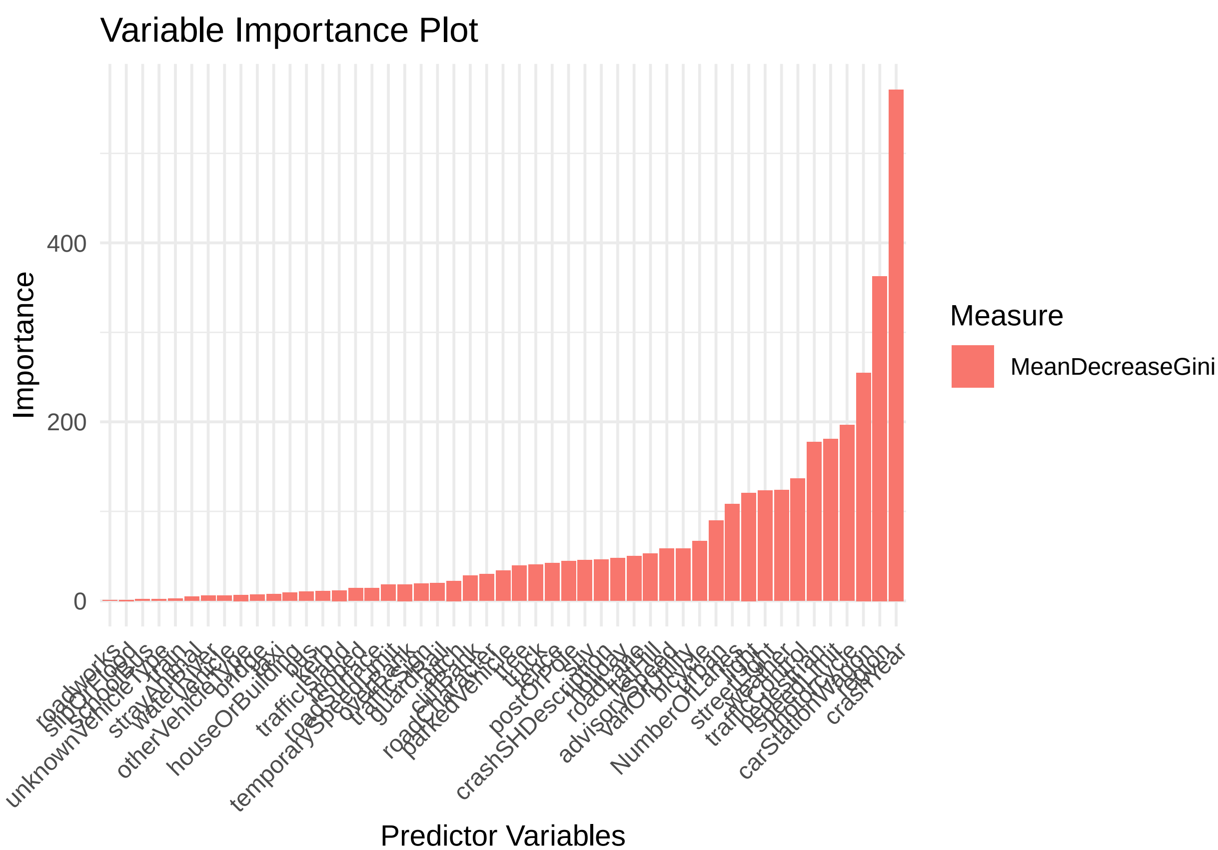
```r
knitr::kable(table(predictions,test_Y))
```

| | 0 | 1 |
|---|---|---|
| 0 | 653 | 181 |
| 1 | 351 | 815 |

**analysis the importance**

| | MeanDecreaseGini |
|---|---|
| crashYear | 571.32273 |
| region | 362.82414 |
| carStationWagon | 254.95824 |
| motorcycle | 196.26954 |
| speedLimit | 180.78257 |
| pedestrian | 177.59802 |
| trafficControl | 136.55451 |
| weather | 123.78827 |

| | MeanDecreaseGini |
|---|---:|
| streetLight | 123.31054 |
| light | 120.80554 |
| NumberOfLanes | 108.10591 |
| urban | 89.57288 |
| bicycle | 66.82217 |
| vanOrUtility | 58.64939 |
| advisorySpeed | 58.21941 |
| flatHill | 52.68257 |
| roadLane | 50.15274 |
| holiday | 47.84270 |
| crashSHDescription | 46.19015 |
| suv | 45.52350 |




## 6. Summarise

We fitted a random forest model based on sample data. and the predict accuracy is 0.734, that is quit good. From Sorted importance table and bar char.We can find the most important factors: crashYear 571.3227293 region 362.8241371 carStationWagon 254.9582385 motorcycle 196.2695437 speedLimit 180.7825677 pedestrian 177.5980152 trafficControl 136.5545083 weather 123.7882740 streetLight 123.3105373 light 120.8055409 NumberOfLanes 108.1059101 urban 89.5728839 bicycle 66.8221703 vanOrUtility 58.6493884 advisorySpeed 58.2194140 flatHill 52.6825686 roadLane 50.1527358 holiday 47.8426982 crashSHDescription 46.1901514 suv 45.5235026

##. Appendix:

```r
# Path to your plain text file
text_file <- "rcode.txt"

# Read and include the content of the text file
cat(readLines(text_file), sep = "\n")
```

```
##  library(tidyverse)
## library(xgboost)
##
## #file_path <- file.choose()
## set.seed(230)
## data <- read.csv("../data/Crash_Analysis_System_(CAS)_data.csv", header = TRUE, sep = ",")
##
## dim(data)
##
##
##
## #clean data
##
## columns_to_drop <- c("X","Y","OBJECTID","areaUnitID","crashDirectionDescription","","crashDistance",
## "crashFinancialYear","fatalCount","debris","meshblockId","northing","easting","objectThrownOrDropped
## "otherObject","phoneBoxEtc","seriousInjuryCount")
##
## data <- select(data, -one_of(columns_to_drop))
##
## na_percentage <- colMeans(is.na(data))
## columns_with_high_na <- names(na_percentage[na_percentage > 0.99])
## #print(columns_with_high_na)
##
## data <- data %>% select(-columns_with_high_na)
##
## #table(data$crashSeverity)
##
## data <- data %>%
##   mutate(crashSeverity = ifelse(crashSeverity == "Fatal Crash" | crashSeverity == "Serious Crash", 1
##                         ifelse(crashSeverity == "Minor Crash" | crashSeverity == "Non-Injury Crash", 0
##                                2))) %>%  filter(crashSeverity != 2)
##
## #table(data$crashSeverity)
##
## #table(data$weatherA)
## #table(data$weatherB)
##
## data$weatherA <- ifelse(data$weatherA %in% c("None", "Null"), "Other", data$weatherA)
## data$weatherB<- ifelse(data$weatherB %in% c("None", "Null"), "", data$weatherB)
##
## data <- data %>% unite(weatherA,weatherB,col=weather,sep=" ")
##
##
## table(data$region)
##
## data$region <- ifelse(data$region %in% c("None", "Null"), "Other", data$region)
##
## na_columns <- sapply(data, function(x) any(is.na(x)))
```

```
##
## columns_with_na <- names(data)[na_columns]
## #print(columns_with_na)
##
## data <- data %>%
##   mutate_at(vars(one_of(columns_with_na)), ~replace_na(., 0))
##
##
## processed_data <- data
## processed_data[processed_data == ""] <- "others"
##
## processed_data[] <- lapply(processed_data, function(x) as.factor(x))
## data <- processed_data
##
## rm(processed_data)
##
## #glimpse(data)
##
##
##
## #explore data
##
## library(ggplot2)
##
##
##
## #unbalanced target Label
## knitr::kable(table(data$crashSeverity))
##
## ggplot(data, aes(x = crashSeverity)) +
##   geom_bar(fill = "blue") +
##   labs(title = "Histogram of crashSeverity", x = "crashSeverity", y = "Frequency")
##
##
## group_by_region <- data %>% group_by(crashYear,region,crashSeverity) %>% summarise(count=n())
##
## ggplot(group_by_region, aes(x = region, y = count, fill = crashSeverity)) +
##   geom_bar(stat = "identity", position = "dodge") +
##   labs(title = "Comparison of Severe and Regular Values by Region", x = "Region", y = "count") +
##   theme(axis.text.x = element_text(angle = 45, hjust = 2))+
##   facet_wrap(~crashYear,nrow=3)
##
## group_by_region <- group_by_region %>% filter(crashSeverity==1)
##
## ggplot(group_by_region, aes(x = region, y = count, fill = crashSeverity)) +
##   geom_bar(stat = "identity", position = "dodge") +
##   labs(title = "Comparison of Severe and Regular Values by Region", x = "Region", y = "count") +
##   theme(axis.text.x = element_text(angle = 45, hjust = 2))+
##   facet_wrap(~crashYear,nrow=3)
##
## group_by_weather <- data %>% group_by(region,weather,crashSeverity) %>% summarise(count=n())
## ggplot(group_by_weather, aes(x = weather, y = count, fill = crashSeverity)) +
##   geom_bar(stat = "identity", position = "dodge") +
##   labs(title = "Weather Conditions in Region", x = "Weather Type", y = "Frequency") +
```

```
##   theme_minimal() +
##   theme(axis.text.x = element_text(angle = 45, hjust = 2)) + # Rotate x-axis labels if needed
##   facet_wrap(~region,nrow=3)
##
## group_by_weather <- group_by_weather %>% filter(crashSeverity==1)
## ggplot(group_by_weather, aes(x = weather, y = count, fill = crashSeverity)) +
##   geom_bar(stat = "identity", position = "dodge") +
##   labs(title = "Weather Conditions in Region", x = "Weather Type", y = "Frequency") +
##   theme_minimal() +
##   theme(axis.text.x = element_text(angle = 45, hjust = 2)) + # Rotate x-axis labels if needed
##   facet_wrap(~region,nrow=3)
##
## group_by_light <- data %>% group_by(region,light,crashSeverity) %>% summarise(count=n())
## ggplot(group_by_light, aes(x = light, y = count, fill = crashSeverity)) +
##   geom_bar(stat = "identity", position = "dodge") +
##   labs(title = "light Conditions in Region", x = "light Type", y = "Frequency") +
##   theme_minimal() +
##   theme(axis.text.x = element_text(angle = 45, hjust = 2)) + # Rotate x-axis labels if needed
##   facet_wrap(~region,nrow=3)
##
## group_by_light <- group_by_light  %>% filter(crashSeverity==1)
## ggplot(group_by_light, aes(x = light, y = count, fill = crashSeverity)) +
##   geom_bar(stat = "identity", position = "dodge") +
##   labs(title = "light Conditions in Region", x = "light Type", y = "Frequency") +
##   theme_minimal() +
##   theme(axis.text.x = element_text(angle = 45, hjust = 2)) + # Rotate x-axis labels if needed
##   facet_wrap(~region,nrow=3)
##
##
## #sample data
##
## # Split data by class
## class_data <- split(data, data$crashSeverity)
##
## # Determine desired sample size (e.g., proportionally to the original class distribution)
## desired_sample_size <- 5000  # Adjust as needed
##
## # Sample each class
## sampled_data <- lapply(class_data, function(class_subset) {
##   # Determine the sample size for this class
##   class_size <- nrow(class_subset)
##   class_sample_size <- min(class_size, desired_sample_size)
##
##   # Sample observations from this class
##   sampled_indices <- sample(1:class_size, size = class_sample_size, replace = TRUE)
##
##   # Return the sampled subset
##   return(class_subset[sampled_indices, ])
## })
##
## # Combine sampled subsets
## balanced_data <- do.call(rbind, sampled_data)
##
##
```

```
##
##
## #fit a model and test
##
## #fit a model and test
##
## training_idx <- sample(nrow(balanced_data), nrow(balanced_data)*0.8)
## test_idx <-(1:nrow(balanced_data))[-training_idx]
##
## training_data <- balanced_data[training_idx,]
## test_data <- balanced_data[test_idx,]
##
## #train_Y <- training_data$crashSeverity
##
## #train_X <- training_data %>% select(-crashSeverity)
##
##
## test_Y <- test_data$crashSeverity
## test_X <- test_data %>% select(-crashSeverity)
##
## library(randomForest)
##
## rf_model <- randomForest(crashSeverity ~ ., data = training_data)
##
## plot(rf_model)
##
## predictions <- predict(rf_model, newdata = test_X)
##
## accuracy <- mean(predictions == test_Y)
##
## print(accuracy)
##
## knitr::kable(table(predictions,test_Y))
##
##
##
## #analyse the importance
##
## importance_measures <- importance(rf_model)
## sorted_importance <- importance_df[order(importance_df$MeanDecreaseGini, decreasing = TRUE), , drop =
##
## knitr::kable(head(sorted_importance,20))
##
##
## importance_df <- as.data.frame(importance_measures)
##
## # Add variable names as a column
## importance_df$Variable <- rownames(importance_df)
##
## # Reshape the data for ggplot
## importance_df_long <- tidyr::gather(importance_df, key = "Measure", value = "Importance", -Variable)
##
## # Create the variable importance plot using ggplot
## ggplot(importance_df_long, aes(x = reorder(Variable, Importance), y = Importance, fill = Measure)) +
```

```
##   geom_bar(stat = "identity") +
##   labs(title = "Variable Importance Plot",
##        x = "Predictor Variables",
##        y = "Importance",
##        fill = "Measure") +
##   theme_minimal() +
##   theme(axis.text.x = element_text(angle = 45, hjust = 1))
```