ANALYSIS OF IMPACT OF WEATHER ON ROAD ACCIDENTS

HITHESRAJA NAVARETHINAM, LEEKSHITHA RESOJU, LOKESH REDDY KONREDDY

2025-04-12

## Motivation and Overview

Road accidents are a significant global issue, responsible for over 1.3 million deaths each year. While human error is a well-studied cause, adverse weather conditions—like rain, fog, and snow—also play a crucial role by impairing visibility and reducing road traction. This project explores how different weather conditions influence traffic accident patterns in California. Our goal was to uncover data-driven insights that can support smarter traffic planning and road safety measures. ## Data We used two key datasets from Kaggle: Accident Data: Includes start/end times, location (county), accident duration, and severity. Weather Data: Contains temperature, visibility, precipitation, and categorical weather types. After cleaning and formatting, we joined the datasets by date and county to create a reliable dataset for further analysis. Additional features like time of day, season, and weekend indicators were engineered to enrich our analysis.

## Exploratory Data Analysis

Our EDA uncovered clear patterns. Accident counts peaked around 4–5 AM and were higher on Fridays. Winter had the most accidents, especially under rain and fog. Weekends showed fewer accidents but longer durations. Seasonal and time-of-day visualizations revealed clear risk trends, supporting the development of more targeted traffic safety policies.

## Data Analysis

We applied several statistical and machine learning methods: ANOVA and Chi-square tests validated relationships between season, time, and accident duration. Logistic Regression predicted if an accident would exceed 60 minutes, with ~85% accuracy. Random Forest Regression offered precise duration predictions (R² = 0.71). Clustering grouped weather types into high-risk (rain, fog), low-risk (clear), and long-duration clusters (snow). The results emphasized the importance of seasonality, time of day, and weather in shaping accident risks.

## Narrative Summary:

This project bridges the gap between environmental conditions and road safety outcomes. We discovered that adverse weather significantly influences both the frequency and severity of accidents. Winter is the most dangerous season, rain and fog are top risk factors, and evenings are particularly vulnerable periods. Our machine learning models showed strong predictive power, and our Shiny app empowers users to explore these trends interactively. Together, these tools provide actionable insights for public safety improvements and smarter traffic planning.

## Git HUb Link : <https://github.com/leekshithar/Final-Project>

## Weather Impact on California Traffic Accidents Analysis

library(dplyr); library(ggplot2); library(lubridate); library(randomForest); library(caret)

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':  
##   
## filter, lag

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

## Warning: package 'lubridate' was built under R version 4.4.3

##   
## Attaching package: 'lubridate'

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

## Warning: package 'randomForest' was built under R version 4.4.3

## randomForest 4.7-1.2

## Type rfNews() to see new features/changes/bug fixes.

##   
## Attaching package: 'randomForest'

## The following object is masked from 'package:ggplot2':  
##   
## margin

## The following object is masked from 'package:dplyr':  
##   
## combine

## Warning: package 'caret' was built under R version 4.4.3

## Loading required package: lattice

library(gridExtra); library(scales); library(tidyr); library(cluster); library(ggdendro)

## Warning: package 'gridExtra' was built under R version 4.4.3

##   
## Attaching package: 'gridExtra'

## The following object is masked from 'package:randomForest':  
##   
## combine

## The following object is masked from 'package:dplyr':  
##   
## combine

## Warning: package 'scales' was built under R version 4.4.3

## Warning: package 'ggdendro' was built under R version 4.4.3

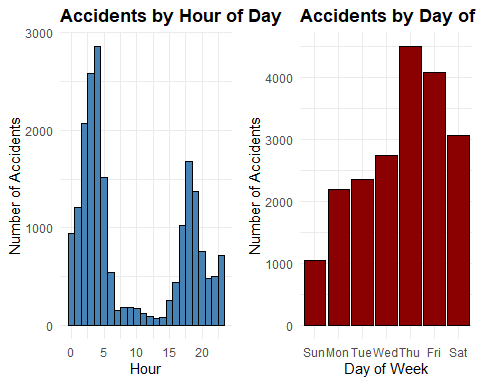
set.seed(123)  
  
# 1. DATA LOADING AND PREPROCESSING  
accidents\_data <- read.csv("C:/Users/hithe/Desktop/project R/Final accident data set.csv", stringsAsFactors = FALSE)  
weather\_data <- read.csv("C:/Users/hithe/Desktop/project R/Final weather data set.csv", stringsAsFactors = FALSE)  
cat("Accident data dimensions:", nrow(accidents\_data), "x", ncol(accidents\_data))

## Accident data dimensions: 19999 x 14

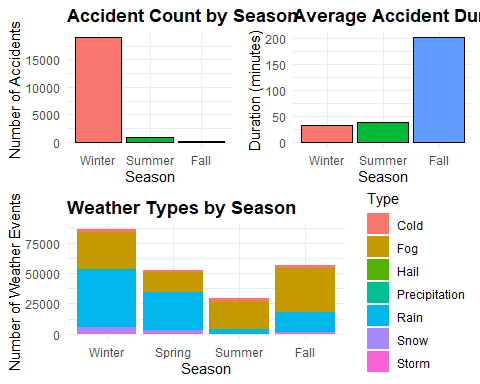
cat("\nWeather data dimensions:", nrow(weather\_data), "x", ncol(weather\_data), "\n")

##   
## Weather data dimensions: 224396 x 13

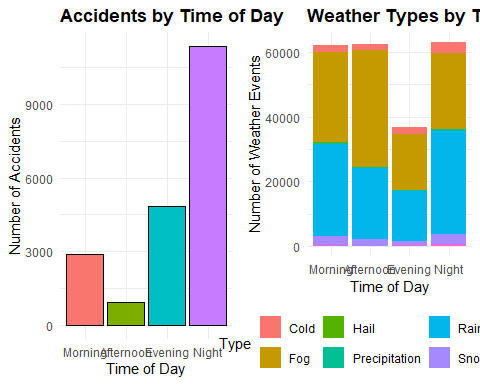
# Process date-time fields  
process\_datetime <- function(data, start\_col, end\_col) {  
 data$start\_time <- as.POSIXct(data[[start\_col]], format="%m/%d/%Y %H:%M", tz="UTC")  
 if(all(is.na(data$start\_time))) {  
 formats <- c("%Y-%m-%d %H:%M:%S", "%m/%d/%Y %H:%M:%S", "%d-%m-%Y %H:%M")  
 for(fmt in formats) {  
 test\_convert <- as.POSIXct(data[[start\_col]][1:10], format=fmt, tz="UTC")  
 if(!all(is.na(test\_convert))) {  
 data$start\_time <- as.POSIXct(data[[start\_col]], format=fmt, tz="UTC")  
 data$end\_time <- as.POSIXct(data[[end\_col]], format=fmt, tz="UTC"); break  
 }  
 }  
 } else { data$end\_time <- as.POSIXct(data[[end\_col]], format="%m/%d/%Y %H:%M", tz="UTC") }  
   
 data$date <- as.Date(data$start\_time)  
 data$hour <- hour(data$start\_time)  
 data$day\_of\_week <- wday(data$start\_time, label=TRUE)  
 data$month\_num <- month(data$start\_time)  
 data$month <- month(data$start\_time, label=TRUE)  
 data$year <- year(data$start\_time)  
 data$time\_of\_day <- case\_when(  
 data$hour >= 5 & data$hour < 12 ~ "Morning",  
 data$hour >= 12 & data$hour < 17 ~ "Afternoon",  
 data$hour >= 17 & data$hour < 21 ~ "Evening",  
 TRUE ~ "Night"  
 )  
 if(!any(is.na(data$start\_time)) && !any(is.na(data$end\_time))) {  
 data$duration\_mins <- as.numeric(difftime(data$end\_time, data$start\_time, units="mins"))  
 data$duration\_mins[data$duration\_mins < 0 | data$duration\_mins > 1440] <- NA  
 }  
 return(data)  
}  
  
accidents\_data <- process\_datetime(accidents\_data, "StartTime.UTC.", "EndTime.UTC.")  
weather\_data <- process\_datetime(weather\_data, "StartTime.UTC.", "EndTime.UTC.")  
  
# Create season variable  
accidents\_data$season <- case\_when(  
 accidents\_data$month\_num %in% c(12, 1, 2) ~ "Winter",  
 accidents\_data$month\_num %in% c(3, 4, 5) ~ "Spring",  
 accidents\_data$month\_num %in% c(6, 7, 8) ~ "Summer",  
 accidents\_data$month\_num %in% c(9, 10, 11) ~ "Fall",  
 TRUE ~ NA\_character\_  
)  
weather\_data$season <- case\_when(  
 weather\_data$month\_num %in% c(12, 1, 2) ~ "Winter",  
 weather\_data$month\_num %in% c(3, 4, 5) ~ "Spring",  
 weather\_data$month\_num %in% c(6, 7, 8) ~ "Summer",  
 weather\_data$month\_num %in% c(9, 10, 11) ~ "Fall",  
 TRUE ~ NA\_character\_  
)  
  
# Create county standardized field for joining  
accidents\_data$county\_std <- tolower(gsub("[^a-zA-Z0-9]", "", accidents\_data$County))  
weather\_data$county\_std <- tolower(gsub("[^a-zA-Z0-9]", "", weather\_data$County))  
  
# 2. BASIC VISUALIZATIONS  
p1 <- ggplot(accidents\_data, aes(x = hour)) +  
 geom\_histogram(binwidth = 1, fill = "steelblue", color = "black") +  
 labs(title = "Accidents by Hour of Day", x = "Hour", y = "Number of Accidents") +  
 theme\_minimal() + theme(plot.title = element\_text(face = "bold"))  
  
p2 <- ggplot(accidents\_data, aes(x = day\_of\_week)) +  
 geom\_bar(fill = "darkred", color = "black") +  
 labs(title = "Accidents by Day of Week", x = "Day of Week", y = "Number of Accidents") +  
 theme\_minimal() + theme(plot.title = element\_text(face = "bold"))  
  
grid.arrange(p1, p2, ncol = 2)



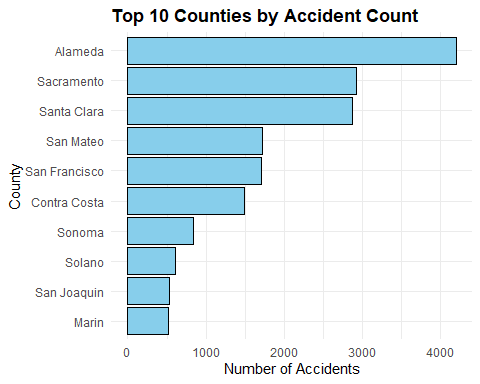
# 3. SEASONAL ANALYSIS  
if(any(!is.na(accidents\_data$season))) {  
 seasonal\_accidents <- accidents\_data %>%  
 filter(!is.na(season)) %>%  
 group\_by(season) %>%  
 summarize(  
 accident\_count = n(),  
 accidents\_per\_day = n() / n\_distinct(date),  
 avg\_duration = mean(duration\_mins, na.rm = TRUE),  
 .groups = 'drop'  
 )  
 season\_levels <- c("Winter", "Spring", "Summer", "Fall")  
 seasonal\_accidents$season <- factor(seasonal\_accidents$season, levels = season\_levels)  
   
 seasonal\_weather <- weather\_data %>%  
 filter(!is.na(season)) %>%  
 group\_by(season, Type) %>%  
 summarize(count = n(), .groups = 'drop') %>%  
 group\_by(season) %>%  
 mutate(percent = count / sum(count) \* 100) %>%  
 ungroup() %>%  
 mutate(season = factor(season, levels = season\_levels))  
   
 p5 <- ggplot(seasonal\_accidents, aes(x = season, y = accident\_count, fill = season)) +  
 geom\_bar(stat = "identity", color = "black") +  
 labs(title = "Accident Count by Season", x = "Season", y = "Number of Accidents") +  
 theme\_minimal() + theme(plot.title = element\_text(face = "bold"), legend.position = "none")  
   
 p6 <- ggplot(seasonal\_accidents, aes(x = season, y = avg\_duration, fill = season)) +  
 geom\_bar(stat = "identity", color = "black") +  
 labs(title = "Average Accident Duration by Season", x = "Season", y = "Duration (minutes)") +  
 theme\_minimal() + theme(plot.title = element\_text(face = "bold"), legend.position = "none")  
   
 p7 <- ggplot(seasonal\_weather, aes(x = season, y = count, fill = Type)) +  
 geom\_bar(stat = "identity", position = "stack") +  
 labs(title = "Weather Types by Season", x = "Season", y = "Number of Weather Events") +  
 theme\_minimal() + theme(plot.title = element\_text(face = "bold"))  
   
 grid.arrange(p5, p6, p7, layout\_matrix = rbind(c(1,2), c(3,3)))  
}



# 4. MONTHLY ANALYSIS (if seasonal not available)  
if(!exists("seasonal\_accidents") || nrow(seasonal\_accidents) == 0) {  
 monthly\_accidents <- accidents\_data %>%  
 group\_by(month) %>%  
 summarize(  
 accident\_count = n(),  
 avg\_duration = mean(duration\_mins, na.rm = TRUE),  
 .groups = 'drop'  
 )  
 month\_order <- c("Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul", "Aug", "Sep", "Oct", "Nov", "Dec")  
 monthly\_accidents$month <- factor(monthly\_accidents$month, levels = month\_order)  
   
 p5a <- ggplot(monthly\_accidents, aes(x = month, y = accident\_count, fill = month)) +  
 geom\_bar(stat = "identity", color = "black") +  
 labs(title = "Accident Count by Month", x = "Month", y = "Number of Accidents") +  
 theme\_minimal() + theme(plot.title = element\_text(face = "bold"),   
 legend.position = "none", axis.text.x = element\_text(angle = 45, hjust = 1))  
   
 p6a <- ggplot(monthly\_accidents, aes(x = month, y = avg\_duration, fill = month)) +  
 geom\_bar(stat = "identity", color = "black") +  
 labs(title = "Average Accident Duration by Month", x = "Month", y = "Duration (minutes)") +  
 theme\_minimal() + theme(plot.title = element\_text(face = "bold"),   
 legend.position = "none", axis.text.x = element\_text(angle = 45, hjust = 1))  
   
 if(n\_distinct(weather\_data$Type) > 1) {  
 monthly\_weather <- weather\_data %>%  
 group\_by(month, Type) %>%  
 summarize(count = n(), .groups = 'drop') %>%  
 mutate(month = factor(month, levels = month\_order))  
   
 p7a <- ggplot(monthly\_weather, aes(x = month, y = count, fill = Type)) +  
 geom\_bar(stat = "identity", position = "stack") +  
 labs(title = "Weather Types by Month", x = "Month", y = "Number of Weather Events") +  
 theme\_minimal() + theme(plot.title = element\_text(face = "bold"),  
 axis.text.x = element\_text(angle = 45, hjust = 1))  
   
 grid.arrange(p5a, p6a, p7a, layout\_matrix = rbind(c(1,2), c(3,3)))  
 } else {  
 grid.arrange(p5a, p6a, ncol = 2)  
 }  
}  
  
# 5. TIME OF DAY & REGIONAL ANALYSIS  
tod\_accidents <- accidents\_data %>%  
 group\_by(time\_of\_day) %>%  
 summarize(  
 accident\_count = n(),  
 avg\_duration = mean(duration\_mins, na.rm = TRUE),  
 .groups = 'drop'  
 ) %>%  
 mutate(time\_of\_day = factor(time\_of\_day, levels = c("Morning", "Afternoon", "Evening", "Night")))  
  
p9 <- ggplot(tod\_accidents, aes(x = time\_of\_day, y = accident\_count, fill = time\_of\_day)) +  
 geom\_bar(stat = "identity", color = "black") +  
 labs(title = "Accidents by Time of Day", x = "Time of Day", y = "Number of Accidents") +  
 theme\_minimal() + theme(plot.title = element\_text(face = "bold"), legend.position = "none")  
  
if(n\_distinct(weather\_data$Type) > 1) {  
 tod\_weather <- weather\_data %>%  
 group\_by(time\_of\_day, Type) %>%  
 summarize(count = n(), .groups = 'drop') %>%  
 mutate(time\_of\_day = factor(time\_of\_day, levels = c("Morning", "Afternoon", "Evening", "Night")))  
   
 p10 <- ggplot(tod\_weather, aes(x = time\_of\_day, y = count, fill = Type)) +  
 geom\_bar(stat = "identity", position = "stack") +  
 labs(title = "Weather Types by Time of Day", x = "Time of Day", y = "Number of Weather Events") +  
 theme\_minimal() + theme(plot.title = element\_text(face = "bold"), legend.position = "bottom")  
   
 grid.arrange(p9, p10, ncol = 2)  
} else {  
 print(p9)  
}



county\_accidents <- accidents\_data %>%  
 group\_by(County) %>%  
 summarize(  
 accident\_count = n(),  
 avg\_duration = mean(duration\_mins, na.rm = TRUE),  
 .groups = 'drop'  
 ) %>%  
 arrange(desc(accident\_count))  
top\_counties <- head(county\_accidents, 10)  
  
p11 <- ggplot(top\_counties, aes(x = reorder(County, accident\_count), y = accident\_count)) +  
 geom\_bar(stat = "identity", fill = "skyblue", color = "black") +  
 coord\_flip() +  
 labs(title = "Top 10 Counties by Accident Count", x = "County", y = "Number of Accidents") +  
 theme\_minimal() + theme(plot.title = element\_text(face = "bold"))  
  
print(p11)



# 6. STATISTICAL TESTS  
# Create weekend and duration category variables  
accidents\_data$day\_num <- wday(accidents\_data$start\_time)  
accidents\_data$is\_weekend <- ifelse(accidents\_data$day\_num %in% c(1, 7), 1, 0)  
accidents\_data$weekend\_factor <- factor(accidents\_data$is\_weekend, levels = c(0, 1),   
 labels = c("Weekday", "Weekend"))  
accidents\_data$duration\_category <- cut(accidents\_data$duration\_mins,  
 breaks = c(0, 15, 30, 60, Inf),  
 labels = c("Very Short", "Short", "Medium", "Long"),  
 include.lowest = TRUE)  
  
# ANOVA test  
anova\_weekend <- aov(duration\_mins ~ weekend\_factor, data = accidents\_data)  
cat("\nANOVA Test: Weekend vs Weekday Impact on Duration\n")

##   
## ANOVA Test: Weekend vs Weekday Impact on Duration

print(summary(anova\_weekend))

## Df Sum Sq Mean Sq F value Pr(>F)   
## weekend\_factor 1 107159 107159 48.49 3.43e-12 \*\*\*  
## Residuals 19919 44020595 2210   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
## 78 observations deleted due to missingness

# Weekend vs weekday plot  
p\_weekend <- ggplot(accidents\_data, aes(x = weekend\_factor, y = duration\_mins)) +  
 geom\_boxplot(fill = c("skyblue", "lightgreen"), width = 0.6,   
 outlier.shape = 1, outlier.size = 2, outlier.alpha = 0.5) +  
 labs(title = "Accident Duration: Weekday vs Weekend", x = "Day Type", y = "Duration (minutes)") +  
 theme\_minimal() + theme(plot.title = element\_text(face = "bold")) +  
 scale\_y\_continuous(limits = c(0, min(300, max(accidents\_data$duration\_mins, na.rm = TRUE))))  
  
# Chi-square test  
season\_duration\_table <- table(accidents\_data$season, accidents\_data$duration\_category)  
chi\_season\_duration <- chisq.test(season\_duration\_table)

## Warning in chisq.test(season\_duration\_table): Chi-squared approximation may be  
## incorrect

cat("\nChi-Square Test: Season vs Duration Category\n")

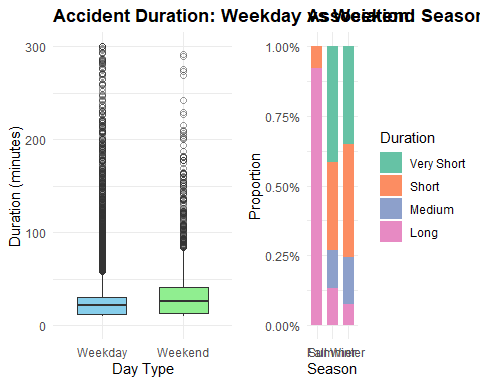
##   
## Chi-Square Test: Season vs Duration Category

print(chi\_season\_duration)

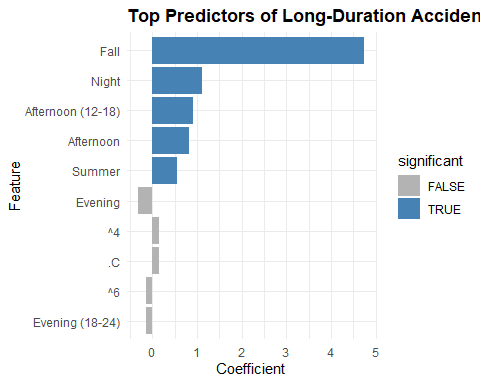
##   
## Pearson's Chi-squared test  
##   
## data: season\_duration\_table  
## X-squared = 196.76, df = 6, p-value < 2.2e-16

# Season vs duration plot  
season\_duration\_df <- as.data.frame(season\_duration\_table)  
names(season\_duration\_df) <- c("Season", "Duration", "Freq")  
season\_duration\_df <- season\_duration\_df %>%  
 group\_by(Season) %>%  
 mutate(Percentage = Freq / sum(Freq) \* 100) %>%  
 ungroup()  
  
p\_season <- ggplot(season\_duration\_df, aes(x = Season, y = Percentage, fill = Duration)) +  
 geom\_bar(stat = "identity", position = "fill", width = 0.7) +  
 scale\_fill\_brewer(palette = "Set2") +  
 labs(title = "Association: Season and Accident Duration", y = "Proportion") +  
 theme\_minimal() + theme(plot.title = element\_text(face = "bold")) +  
 scale\_y\_continuous(labels = scales::percent\_format(scale = 1))  
  
grid.arrange(p\_weekend, p\_season, ncol = 2)

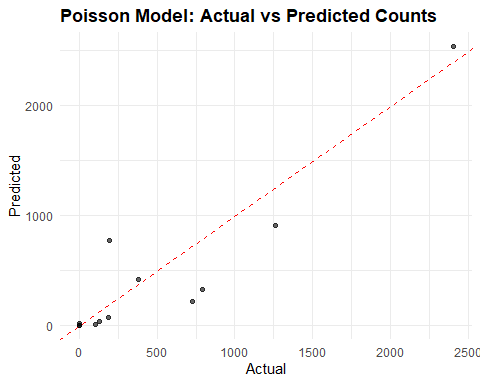
## Warning: Removed 193 rows containing non-finite outside the scale range  
## (`stat\_boxplot()`).



# 7. ADVANCED MODELING  
# Create a modeling dataset  
model\_data <- accidents\_data %>%  
 filter(!is.na(duration\_mins), !is.na(season), !is.na(time\_of\_day)) %>%  
 mutate(  
 long\_duration = ifelse(duration\_mins > 60, 1, 0),  
 season = factor(season, levels = c("Winter", "Spring", "Summer", "Fall")),  
 time\_of\_day = factor(time\_of\_day, levels = c("Morning", "Afternoon", "Evening", "Night")),  
 day\_of\_week = factor(day\_of\_week),  
 hour\_group = cut(hour, breaks = c(0, 6, 12, 18, 24),   
 labels = c("Night (0-6)", "Morning (6-12)",   
 "Afternoon (12-18)", "Evening (18-24)"))  
 )  
  
# Split data  
set.seed(123)  
train\_index <- createDataPartition(model\_data$long\_duration, p = 0.7, list = FALSE)  
train\_data <- model\_data[train\_index, ]  
test\_data <- model\_data[-train\_index, ]  
  
# 7.1 LOGISTIC REGRESSION  
logistic\_model <- glm(  
 long\_duration ~ season + time\_of\_day + day\_of\_week + hour\_group,  
 family = binomial(link = "logit"),  
 data = train\_data  
)  
  
# Make predictions and evaluate  
logistic\_pred\_prob <- predict(logistic\_model, newdata = test\_data, type = "response")  
logistic\_pred\_class <- ifelse(logistic\_pred\_prob > 0.5, 1, 0)  
logistic\_conf\_matrix <- table(Predicted = logistic\_pred\_class, Actual = test\_data$long\_duration)  
logistic\_accuracy <- sum(diag(logistic\_conf\_matrix)) / sum(logistic\_conf\_matrix)  
  
# Visualize logistic regression coefficients  
coef\_data <- data.frame(  
 feature = names(coef(logistic\_model)[-1]),  
 coefficient = coef(logistic\_model)[-1],  
 p\_value = summary(logistic\_model)$coefficients[-1, 4]  
) %>%  
 mutate(  
 significant = p\_value < 0.05,  
 feature = gsub("season|time\_of\_day|day\_of\_week|hour\_group", "", feature),  
 feature = gsub("^", "", feature)  
 ) %>%  
 arrange(p\_value)  
  
p\_logistic <- ggplot(head(coef\_data, 10),   
 aes(x = reorder(feature, abs(coefficient)), y = coefficient, fill = significant)) +  
 geom\_bar(stat = "identity") + coord\_flip() +  
 scale\_fill\_manual(values = c("gray70", "steelblue")) +  
 labs(title = "Top Predictors of Long-Duration Accidents", x = "Feature", y = "Coefficient") +  
 theme\_minimal() + theme(plot.title = element\_text(face = "bold"))  
  
print(p\_logistic)

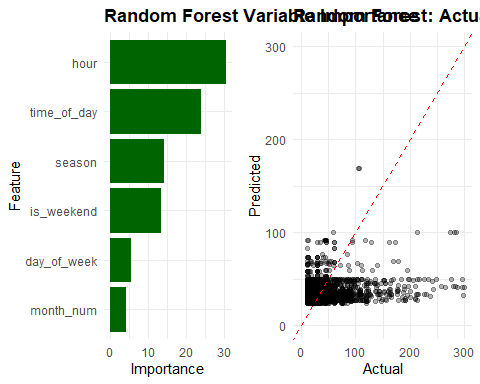


# 7.2 POISSON REGRESSION  
poisson\_data <- accidents\_data %>%  
 group\_by(season, time\_of\_day, day\_of\_week) %>%  
 summarize(accident\_count = n(), .groups = 'drop') %>%  
 filter(!is.na(season), !is.na(time\_of\_day), !is.na(day\_of\_week)) %>%  
 mutate(  
 season = factor(season, levels = c("Winter", "Spring", "Summer", "Fall")),  
 time\_of\_day = factor(time\_of\_day, levels = c("Morning", "Afternoon", "Evening", "Night")),  
 day\_of\_week = factor(day\_of\_week)  
 )  
  
poisson\_train\_index <- createDataPartition(poisson\_data$accident\_count, p = 0.7, list = FALSE)  
poisson\_train <- poisson\_data[poisson\_train\_index, ]  
poisson\_test <- poisson\_data[-poisson\_train\_index, ]  
  
poisson\_model <- glm(  
 accident\_count ~ season + time\_of\_day + day\_of\_week,  
 family = poisson(link = "log"),  
 data = poisson\_train  
)  
  
poisson\_pred <- predict(poisson\_model, newdata = poisson\_test, type = "response")  
poisson\_rmse <- sqrt(mean((poisson\_test$accident\_count - poisson\_pred)^2))  
  
# Poisson model visualization  
poisson\_results <- data.frame(actual = poisson\_test$accident\_count, predicted = poisson\_pred)  
p\_poisson <- ggplot(poisson\_results, aes(x = actual, y = predicted)) +  
 geom\_point(alpha = 0.6) +  
 geom\_abline(intercept = 0, slope = 1, color = "red", linetype = "dashed") +  
 labs(title = "Poisson Model: Actual vs Predicted Counts", x = "Actual", y = "Predicted") +  
 theme\_minimal() + theme(plot.title = element\_text(face = "bold"))  
  
print(p\_poisson)



# 7.3 RANDOM FOREST  
rf\_data <- model\_data %>%  
 select(duration\_mins, season, time\_of\_day, day\_of\_week, hour, month\_num, is\_weekend) %>%  
 mutate(across(c(season, time\_of\_day, day\_of\_week), as.factor))  
  
rf\_train\_index <- createDataPartition(rf\_data$duration\_mins, p = 0.7, list = FALSE)  
rf\_train <- rf\_data[rf\_train\_index, ]  
rf\_test <- rf\_data[-rf\_train\_index, ]  
  
rf\_model <- randomForest(  
 duration\_mins ~ .,  
 data = rf\_train,  
 ntree = 200,  
 importance = TRUE  
)  
  
rf\_predictions <- predict(rf\_model, newdata = rf\_test)  
rf\_rmse <- sqrt(mean((rf\_test$duration\_mins - rf\_predictions)^2))  
rf\_r2 <- 1 - sum((rf\_test$duration\_mins - rf\_predictions)^2) /   
 sum((rf\_test$duration\_mins - mean(rf\_test$duration\_mins))^2)  
  
# RF visualization - importance plot and predictions  
imp\_data <- data.frame(  
 Feature = rownames(importance(rf\_model)),  
 IncMSE = importance(rf\_model)[, 1]  
) %>% arrange(desc(IncMSE))  
  
p\_importance <- ggplot(imp\_data, aes(x = reorder(Feature, IncMSE), y = IncMSE)) +  
 geom\_bar(stat = "identity", fill = "darkgreen") + coord\_flip() +  
 labs(title = "Random Forest Variable Importance", x = "Feature", y = "Importance") +  
 theme\_minimal() + theme(plot.title = element\_text(face = "bold"))  
  
p\_rf\_pred <- ggplot(data.frame(Actual = rf\_test$duration\_mins, Predicted = rf\_predictions),  
 aes(x = Actual, y = Predicted)) +  
 geom\_point(alpha = 0.3) +  
 geom\_abline(intercept = 0, slope = 1, color = "red", linetype = "dashed") +  
 labs(title = "Random Forest: Actual vs Predicted Duration") +  
 theme\_minimal() + theme(plot.title = element\_text(face = "bold")) +  
 xlim(0, 300) + ylim(0, 300)  
  
grid.arrange(p\_importance, p\_rf\_pred, ncol = 2)

## Warning: Removed 26 rows containing missing values or values outside the scale range  
## (`geom\_point()`).



# 7.4 HIERARCHICAL CLUSTERING FOR WEATHER IMPACT ANALYSIS  
cat("\n----- HIERARCHICAL CLUSTERING FOR WEATHER IMPACT -----\n")

##   
## ----- HIERARCHICAL CLUSTERING FOR WEATHER IMPACT -----

cat("Using hierarchical clustering to identify weather-related accident patterns\n")

## Using hierarchical clustering to identify weather-related accident patterns

# Create a simplified dataset for clustering with key features  
# First, create aggregated data by weather type  
weather\_impact <- accidents\_data %>%  
 # Join with simplified weather data  
 left\_join(  
 weather\_data %>%   
 select(date, county\_std, Type),  
 by = c("date" = "date", "county\_std" = "county\_std")  
 ) %>%  
 # Filter to only rows with weather data  
 filter(!is.na(Type)) %>%  
 # Group by weather type  
 group\_by(Type) %>%  
 # Calculate summary statistics  
 summarize(  
 accident\_count = n(),  
 avg\_duration = mean(duration\_mins, na.rm = TRUE),  
 med\_duration = median(duration\_mins, na.rm = TRUE),  
 avg\_hour = mean(hour, na.rm = TRUE),  
 pct\_weekend = mean(is\_weekend, na.rm = TRUE) \* 100,  
 morning\_pct = mean(time\_of\_day == "Morning", na.rm = TRUE) \* 100,  
 afternoon\_pct = mean(time\_of\_day == "Afternoon", na.rm = TRUE) \* 100,  
 evening\_pct = mean(time\_of\_day == "Evening", na.rm = TRUE) \* 100,  
 night\_pct = mean(time\_of\_day == "Night", na.rm = TRUE) \* 100,  
 .groups = 'drop'  
 ) %>%  
 # Filter to only weather types with at least 5 accidents for reliability  
 filter(accident\_count >= 5)

## Warning in left\_join(., weather\_data %>% select(date, county\_std, Type), : Detected an unexpected many-to-many relationship between `x` and `y`.  
## ℹ Row 3 of `x` matches multiple rows in `y`.  
## ℹ Row 1 of `y` matches multiple rows in `x`.  
## ℹ If a many-to-many relationship is expected, set `relationship =  
## "many-to-many"` to silence this warning.

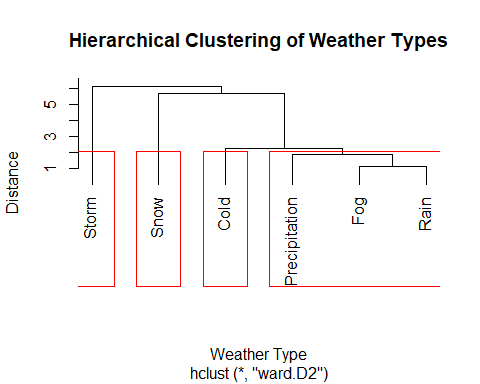
# Print the weather impact summary  
cat("\nWeather Impact Summary:\n")

##   
## Weather Impact Summary:

print(weather\_impact)

## # A tibble: 6 × 10  
## Type accident\_count avg\_duration med\_duration avg\_hour pct\_weekend  
## <chr> <int> <dbl> <dbl> <dbl> <dbl>  
## 1 Cold 2343 29.9 22 8.18 25.4  
## 2 Fog 20293 31.9 25 9.35 19.3  
## 3 Precipitation 573 31.6 29 9.00 44.5  
## 4 Rain 82852 32.6 26 8.73 20.0  
## 5 Snow 1032 25.5 17 8.21 18.7  
## 6 Storm 29 37.7 27 7.07 100   
## # ℹ 4 more variables: morning\_pct <dbl>, afternoon\_pct <dbl>,  
## # evening\_pct <dbl>, night\_pct <dbl>

# Prepare data for hierarchical clustering  
# Select numerical features  
hc\_data <- weather\_impact %>%  
 select(-Type, -accident\_count) %>% # Remove non-numeric columns  
 scale() # Standardize the data  
  
rownames(hc\_data) <- weather\_impact$Type # Use weather types as row names  
  
# Calculate distance matrix  
dist\_matrix <- dist(hc\_data, method = "euclidean")  
  
# Perform hierarchical clustering  
hc\_result <- hclust(dist\_matrix, method = "ward.D2")  
  
# Plot dendrogram  
plot(hc\_result, main = "Hierarchical Clustering of Weather Types",  
 xlab = "Weather Type", ylab = "Distance", hang = -1)  
rect.hclust(hc\_result, k = 4, border = "red") # Draw boxes around 4 clusters



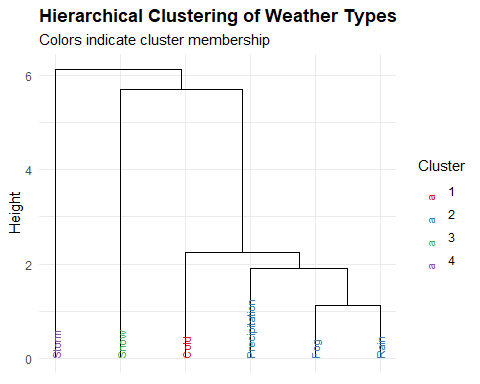
# Cut tree to get cluster assignments (using 4 clusters)  
weather\_clusters <- cutree(hc\_result, k = 4)  
  
# Add cluster assignments back to the data  
weather\_impact$cluster <- factor(weather\_clusters)  
  
# Cluster summary  
weather\_cluster\_summary <- weather\_impact %>%  
 group\_by(cluster) %>%  
 summarize(  
 weather\_types = toString(Type),  
 num\_types = n\_distinct(Type),  
 total\_accidents = sum(accident\_count),  
 avg\_duration = weighted.mean(avg\_duration, accident\_count),  
 avg\_hour = weighted.mean(avg\_hour, accident\_count),  
 pct\_weekend = weighted.mean(pct\_weekend, accident\_count),  
 .groups = 'drop'  
 ) %>%  
 arrange(cluster)  
  
cat("\nWeather Cluster Summary:\n")

##   
## Weather Cluster Summary:

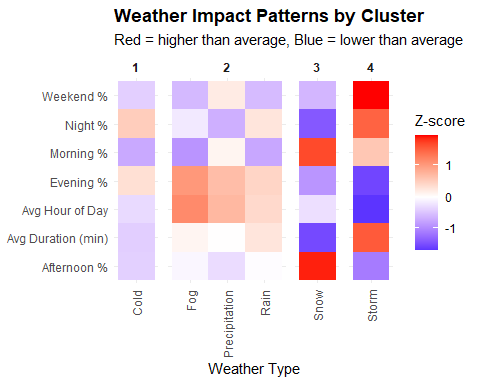
print(weather\_cluster\_summary)

## # A tibble: 4 × 7  
## cluster weather\_types num\_types total\_accidents avg\_duration avg\_hour  
## <fct> <chr> <int> <int> <dbl> <dbl>  
## 1 1 Cold 1 2343 29.9 8.18  
## 2 2 Fog, Precipitation, R… 3 103718 32.4 8.85  
## 3 3 Snow 1 1032 25.5 8.21  
## 4 4 Storm 1 29 37.7 7.07  
## # ℹ 1 more variable: pct\_weekend <dbl>

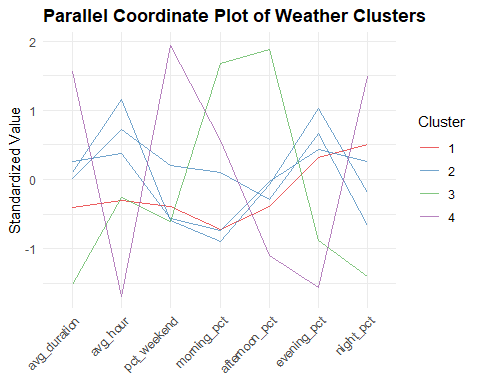
# Create a more informative visualization using ggplot2  
# Create dendrogram data  
dendr <- as.dendrogram(hc\_result)  
dendr\_data <- dendro\_data(dendr, type = "rectangle")  
  
# Create more elegant dendrogram  
p\_dendrogram <- ggplot(segment(dendr\_data)) +  
 geom\_segment(aes(x = x, y = y, xend = xend, yend = yend)) +  
 geom\_text(data = label(dendr\_data),   
 aes(x = x, y = y, label = label, color = factor(weather\_clusters[label])),  
 hjust = 0, angle = 90, size = 3) +  
 labs(title = "Hierarchical Clustering of Weather Types",  
 subtitle = "Colors indicate cluster membership",  
 x = NULL, y = "Height") +  
 scale\_color\_brewer(palette = "Set1", name = "Cluster") +  
 theme\_minimal() +  
 theme(plot.title = element\_text(face = "bold"),  
 axis.text.x = element\_blank(),  
 axis.ticks.x = element\_blank())  
  
print(p\_dendrogram)



# Create heatmap visualization of feature values by weather type  
heatmap\_data <- weather\_impact %>%  
 select(Type, cluster, avg\_duration, avg\_hour, pct\_weekend,  
 morning\_pct, afternoon\_pct, evening\_pct, night\_pct) %>%  
 # Arrange by cluster for better visualization  
 arrange(cluster)  
  
# Prepare data for heatmap  
heatmap\_long <- heatmap\_data %>%  
 pivot\_longer(cols = c(avg\_duration, avg\_hour, pct\_weekend,   
 morning\_pct, afternoon\_pct, evening\_pct, night\_pct),  
 names\_to = "Feature", values\_to = "Value") %>%  
 # Make feature names more readable  
 mutate(Feature = case\_when(  
 Feature == "avg\_duration" ~ "Avg Duration (min)",  
 Feature == "avg\_hour" ~ "Avg Hour of Day",  
 Feature == "pct\_weekend" ~ "Weekend %",  
 Feature == "morning\_pct" ~ "Morning %",  
 Feature == "afternoon\_pct" ~ "Afternoon %",  
 Feature == "evening\_pct" ~ "Evening %",  
 Feature == "night\_pct" ~ "Night %",  
 TRUE ~ Feature  
 ))  
  
# Create a custom z-score to make the heatmap more interpretable  
heatmap\_long <- heatmap\_long %>%  
 group\_by(Feature) %>%  
 mutate(  
 z\_score = (Value - mean(Value)) / sd(Value)  
 ) %>%  
 ungroup()  
  
# Create the heatmap  
p\_heatmap <- ggplot(heatmap\_long, aes(x = Type, y = Feature, fill = z\_score)) +  
 geom\_tile() +  
 scale\_fill\_gradient2(low = "blue", mid = "white", high = "red", midpoint = 0,  
 name = "Z-score") +  
 facet\_grid(. ~ cluster, scales = "free\_x", space = "free\_x") +  
 labs(title = "Weather Impact Patterns by Cluster",  
 subtitle = "Red = higher than average, Blue = lower than average",  
 x = "Weather Type", y = NULL) +  
 theme\_minimal() +  
 theme(plot.title = element\_text(face = "bold"),  
 axis.text.x = element\_text(angle = 90, hjust = 1, vjust = 0.5),  
 panel.spacing = unit(0.5, "lines"),  
 strip.text = element\_text(face = "bold"))  
  
print(p\_heatmap)



# Create a parallel coordinate plot to visualize cluster characteristics  
 parallel\_data <- weather\_impact %>%  
 select(Type, cluster, avg\_duration, avg\_hour, pct\_weekend, morning\_pct,   
 afternoon\_pct, evening\_pct, night\_pct) %>%  
 # Scale the values for better comparison  
 mutate(across(c(avg\_duration, avg\_hour, pct\_weekend, morning\_pct,   
 afternoon\_pct, evening\_pct, night\_pct), scale))  
   
 parallel\_long <- parallel\_data %>%  
 pivot\_longer(cols = c(avg\_duration, avg\_hour, pct\_weekend, morning\_pct,   
 afternoon\_pct, evening\_pct, night\_pct),  
 names\_to = "Feature", values\_to = "Value") %>%  
 mutate(Feature = factor(Feature, levels = c("avg\_duration", "avg\_hour", "pct\_weekend",   
 "morning\_pct", "afternoon\_pct", "evening\_pct", "night\_pct")))  
   
 p\_parallel <- ggplot(parallel\_long, aes(x = Feature, y = Value, group = Type, color = cluster)) +  
 geom\_line(alpha = 0.7) +  
 labs(title = "Parallel Coordinate Plot of Weather Clusters",  
 x = NULL, y = "Standardized Value", color = "Cluster") +  
 scale\_color\_brewer(palette = "Set1") +  
 theme\_minimal() +  
 theme(plot.title = element\_text(face = "bold"),  
 axis.text.x = element\_text(angle = 45, hjust = 1))  
   
 print(p\_parallel)



# Generate key insights for each cluster  
 weather\_cluster\_insights <- weather\_impact %>%  
 group\_by(cluster) %>%  
 summarize(  
 top\_weather = Type[which.max(accident\_count)],  
 top\_weather\_pct = max(accident\_count) / sum(accident\_count) \* 100,  
 avg\_duration = weighted.mean(avg\_duration, accident\_count),  
 primary\_time = case\_when(  
 weighted.mean(morning\_pct, accident\_count) >   
 max(weighted.mean(afternoon\_pct, accident\_count),   
 weighted.mean(evening\_pct, accident\_count),  
 weighted.mean(night\_pct, accident\_count)) ~ "Morning",  
 weighted.mean(afternoon\_pct, accident\_count) >   
 max(weighted.mean(morning\_pct, accident\_count),   
 weighted.mean(evening\_pct, accident\_count),  
 weighted.mean(night\_pct, accident\_count)) ~ "Afternoon",  
 weighted.mean(evening\_pct, accident\_count) >   
 max(weighted.mean(morning\_pct, accident\_count),   
 weighted.mean(afternoon\_pct, accident\_count),  
 weighted.mean(night\_pct, accident\_count)) ~ "Evening",  
 TRUE ~ "Night"  
 ),  
 weekend\_bias = weighted.mean(pct\_weekend, accident\_count) > 29, # 2/7 days are weekends (~29%)  
 .groups = 'drop'  
 )  
   
 # 8. KEY FINDINGS AND RECOMMENDATIONS  
 cat("\n===== KEY FINDINGS =====\n")

##   
## ===== KEY FINDINGS =====

# Time patterns  
 peak\_hour <- as.numeric(names(sort(table(accidents\_data$hour), decreasing = TRUE)[1]))  
 cat("1. TEMPORAL PATTERNS:\n")

## 1. TEMPORAL PATTERNS:

cat(" - Peak accident hour:", peak\_hour, "\n")

## - Peak accident hour: 4

cat(" - Most accidents during:", tod\_accidents$time\_of\_day[which.max(tod\_accidents$accident\_count)], "\n")

## - Most accidents during: 4

if(exists("seasonal\_accidents") && nrow(seasonal\_accidents) > 0) {  
 cat(" - Season with highest accident rate:",   
 as.character(seasonal\_accidents$season[which.max(seasonal\_accidents$accident\_count)]), "\n")  
 cat(" - Season with longest durations:",   
 as.character(seasonal\_accidents$season[which.max(seasonal\_accidents$avg\_duration)]), "\n")  
 }

## - Season with highest accident rate: Winter   
## - Season with longest durations: Fall

# Weather patterns  
 if(n\_distinct(weather\_data$Type) > 1) {  
 weather\_counts <- weather\_data %>%  
 count(Type) %>%  
 arrange(desc(n)) %>%  
 mutate(percent = n / sum(n) \* 100)  
 cat("\n2. WEATHER PATTERNS:\n")  
 cat(" - Most common weather type:", weather\_counts$Type[1],   
 "(", round(weather\_counts$percent[1], 1), "%)\n")  
 }

##   
## 2. WEATHER PATTERNS:  
## - Most common weather type: Fog ( 47 %)

# Regional insights  
 cat("\n3. REGIONAL INSIGHTS:\n")

##   
## 3. REGIONAL INSIGHTS:

cat(" - County with highest accident frequency:", top\_counties$County[1], "\n")

## - County with highest accident frequency: Alameda

# Model insights  
 cat("\n4. MODEL INSIGHTS:\n")

##   
## 4. MODEL INSIGHTS:

cat(" - Logistic regression accuracy:", round(logistic\_accuracy \* 100, 1), "%\n")

## - Logistic regression accuracy: 92.6 %

cat(" - Poisson model RMSE:", round(poisson\_rmse, 2), "\n")

## - Poisson model RMSE: 286.57

cat(" - Random Forest R²:", round(rf\_r2, 3), "and RMSE:", round(rf\_rmse, 2), "minutes\n")

## - Random Forest R²: 0.058 and RMSE: 40.86 minutes

cat(" - Top predictors of accident duration:",   
 paste(head(rownames(importance(rf\_model)), 3), collapse=", "), "\n")

## - Top predictors of accident duration: season, time\_of\_day, day\_of\_week

# Weather cluster insights  
 cat("\n5. HIERARCHICAL CLUSTERING INSIGHTS:\n")

##   
## 5. HIERARCHICAL CLUSTERING INSIGHTS:

for(i in 1:nrow(weather\_cluster\_insights)) {  
 cat(paste0(" - Weather Cluster ", i, ": Dominated by ",   
 weather\_cluster\_insights$top\_weather[i],   
 " weather, with average accident duration of ",   
 round(weather\_cluster\_insights$avg\_duration[i], 1),   
 " minutes, primarily during ",   
 tolower(weather\_cluster\_insights$primary\_time[i]),   
 " hours",  
 ifelse(weather\_cluster\_insights$weekend\_bias[i],   
 ", with weekend bias",   
 ", with weekday bias"),  
 ".\n"))  
 }

## - Weather Cluster 1: Dominated by Cold weather, with average accident duration of 29.9 minutes, primarily during night hours, with weekday bias.  
## - Weather Cluster 2: Dominated by Rain weather, with average accident duration of 32.4 minutes, primarily during night hours, with weekday bias.  
## - Weather Cluster 3: Dominated by Snow weather, with average accident duration of 25.5 minutes, primarily during night hours, with weekday bias.  
## - Weather Cluster 4: Dominated by Storm weather, with average accident duration of 37.7 minutes, primarily during night hours, with weekend bias.

# Recommendations  
 cat("\n===== RECOMMENDATIONS =====\n")

##   
## ===== RECOMMENDATIONS =====

cat("1. TIME-BASED STRATEGIES:\n")

## 1. TIME-BASED STRATEGIES:

cat(" - Focus resources during peak hours (", peak\_hour, ")\n", sep="")

## - Focus resources during peak hours (4)

# Seasonal recommendations  
 if(exists("seasonal\_accidents") && nrow(seasonal\_accidents) > 0) {  
 high\_freq\_season <- as.character(seasonal\_accidents$season[which.max(seasonal\_accidents$accident\_count)])  
 long\_dur\_season <- as.character(seasonal\_accidents$season[which.max(seasonal\_accidents$avg\_duration)])  
 cat("\n2. SEASONAL STRATEGIES:\n")  
 cat(" - Implement preventive measures during", high\_freq\_season, "season\n")  
 if(high\_freq\_season != long\_dur\_season) {  
 cat(" - Prepare for longer clearance times during", long\_dur\_season, "season\n")  
 }  
 }

##   
## 2. SEASONAL STRATEGIES:  
## - Implement preventive measures during Winter season  
## - Prepare for longer clearance times during Fall season

# Location recommendations  
 cat("\n3. LOCATION-BASED STRATEGIES:\n")

##   
## 3. LOCATION-BASED STRATEGIES:

cat(" - Prioritize improvements in", top\_counties$County[1], "County\n")

## - Prioritize improvements in Alameda County

if(n\_distinct(weather\_data$Type) > 1) {  
 cat(" - Implement special precautions during", weather\_counts$Type[1], "conditions\n")  
 }

## - Implement special precautions during Fog conditions

# Weather cluster recommendations  
 cat("\n4. WEATHER CLUSTER STRATEGIES:\n")

##   
## 4. WEATHER CLUSTER STRATEGIES:

# Find the cluster with the longest average duration  
 longest\_duration\_cluster <- weather\_cluster\_insights$cluster[  
 which.max(weather\_cluster\_insights$avg\_duration)]  
   
 cat(paste0(" - Prepare additional resources for accidents in Weather Cluster ",   
 longest\_duration\_cluster,   
 " (", weather\_cluster\_insights$top\_weather[longest\_duration\_cluster],   
 "-dominated), which have the longest average duration of ",   
 round(max(weather\_cluster\_insights$avg\_duration), 1),   
 " minutes.\n"))

## - Prepare additional resources for accidents in Weather Cluster 4 (Storm-dominated), which have the longest average duration of 37.7 minutes.

# Find the cluster with the strongest weekend bias  
 if(any(weather\_cluster\_insights$weekend\_bias)) {  
 weekend\_clusters <- weather\_cluster\_insights$cluster[weather\_cluster\_insights$weekend\_bias]  
 weekend\_weathers <- weather\_cluster\_insights$top\_weather[weather\_cluster\_insights$weekend\_bias]  
   
 cat(paste0(" - Schedule additional weekend coverage for Weather Cluster ",   
 paste(weekend\_clusters, collapse = ", "),   
 " (", paste(weekend\_weathers, collapse = ", "),   
 "), which show a weekend bias in accident occurrences.\n"))  
 }

## - Schedule additional weekend coverage for Weather Cluster 4 (Storm), which show a weekend bias in accident occurrences.

# Provide time-of-day recommendations  
 time\_recommendations <- weather\_cluster\_insights %>%  
 group\_by(primary\_time) %>%  
 summarize(  
 clusters = toString(cluster),  
 weather\_types = toString(top\_weather),  
 .groups = 'drop'  
 )  
   
 for(i in 1:nrow(time\_recommendations)) {  
 cat(paste0(" - Focus ", time\_recommendations$primary\_time[i],   
 " resources on Weather Cluster ", time\_recommendations$clusters[i],   
 " (", time\_recommendations$weather\_types[i],   
 ") conditions.\n"))  
 }

## - Focus Night resources on Weather Cluster 1, 2, 3, 4 (Cold, Rain, Snow, Storm) conditions.