Minería de datos: PEC1

Autor: Vinicio Alejandro Naranjo Mosquera

Marzo 2024

Contents

Ejercicio 1

Propón un proyecto completo de minería de datos. La organización de la respuesta tiene que coincidir en las fases típicas del ciclo de vida de un proyecto de minería de datos. **No hay que realizar las tareas de la fase**.

Se espera por lo tanto que se responda de forma argumentada a las siguientes preguntas (Metodología *CRISP-DM*):

1. **Comprensión del negocio** - ¿Qué necesita el negocio?
2. **Comprensión de los datos** - ¿Qué datos tenemos/necesitamos? ¿Están limpios?
3. **Preparación de los datos** - ¿Cómo organizamos los datos para el modelado?
4. **Modelado** - ¿Qué técnicas de modelado debemos aplicar?
5. **Evaluación** - ¿Qué modelo cumple mejor con los objetivos del negocio?
6. **Implementación** - ¿Cómo acceden los interesados a los resultados?

Para cada fase indica cuál es el objetivo de la fase y el producto que se obtendrá. Utiliza ejemplos de qué y cómo podrían ser las tareas. Si hay alguna característica que hace diferente el ciclo de vida de un proyecto de minería respecto a otros proyectos indícalo.

Extensión mínima: 400 palabras

> Escribe aquí la respuesta a la pregunta

1. Comprensión del Negocio

Objetivo: Identificar y definir claramente los problemas y oportunidades financieras personales, con el fin de optimizar la gestión de gastos e inversiones.

Producto: Un documento de definición del proyecto que esboza los objetivos específicos del análisis, como identificar patrones de gasto, optimizar la planificación de inversiones, y sugerir estrategias de ahorro.

Ejemplo: Determinar los periodos del año donde el gasto es más alto y identificar oportunidades de inversión basadas en rendimientos históricos.

2. Comprensión de los Datos

Objetivo: Evaluar la disponibilidad, calidad y relevancia de los datos recolectados para los objetivos del análisis.

Producto: Un informe de evaluación de datos que detalle las fuentes de datos, su estructura, y cualquier problema de calidad o limpieza identificado.

Ejemplo: Revisar los registros de transacciones para verificar completitud, categorización correcta de gastos e ingresos, y la existencia de datos sobre rendimiento de inversiones.

3. Preparación de los Datos

Objetivo: Transformar y organizar los datos brutos en un formato adecuado para el modelado, incluyendo la limpieza, integración, y selección de variables.

Producto: Un conjunto de datos preparado y listo para el modelado, con variables seleccionadas y transformadas según sea necesario.

Ejemplo: Crear variables agregadas como el total mensual de gastos, total de ingresos, y el rendimiento de la inversión; normalizar los montos de transacciones.

4. Modelado

Objetivo: Aplicar técnicas de minería de datos para construir modelos que cumplan con los objetivos del análisis, como predecir patrones de gasto o identificar el mejor momento para invertir.

Producto: Modelos de clasificación y regresión desarrollados, validados y optimizados para predecir variables de interés.

Ejemplo: Usar modelos de regresión para predecir el total de gastos mensuales y modelos de clasificación para identificar categorías de gastos con mayor probabilidad de aumento.

5. Evaluación

Objetivo: Seleccionar el mejor modelo basándome en su rendimiento y su alineación con los objetivos del negocio.

Producto: Un informe de evaluación que compara los modelos en base a métricas de rendimiento y su relevancia para los objetivos del análisis.

Ejemplo: Comparar modelos utilizando métricas como la precisión para clasificación y RMSE para regresión, seleccionando aquellos que mejor predicen los patrones de gasto.

6. Implementación

Objetivo: El objetivo principal aquí es traducir los modelos analíticos y sus insights en funcionalidades concretas dentro de una aplicación de gestión de capital accesible y útil para un amplio espectro de usuarios, desde novatos en finanzas hasta expertos. Busco que esta herramienta ayude a los usuarios a tomar decisiones financieras informadas basadas en la predicción de gastos, recomendaciones de inversión, y seguimiento de rendimientos.

Producto: El producto final será una aplicación multiplataforma (disponible en web, iOS, y Android) que integra los modelos de predicción financiera directamente en la interfaz de usuario. La app permitirá a los usuarios ingresar, visualizar, y analizar sus datos financieros personales, ofreciendo predicciones personalizadas y recomendaciones basadas en sus hábitos de gasto e inversión.

Ejemplo: Dentro de la aplicación, un usuario podría: Ver Predicciones de Gastos: Basándose en su historial de gastos, la app podría mostrar predicciones sobre futuros gastos en diferentes categorías. Obtener Recomendaciones de Inversión: Según el perfil de riesgo y los objetivos financieros del usuario, la app podría sugerir oportunidades de inversión ajustadas a sus necesidades. Seguimiento del Rendimiento de Inversiones: Permitir a los usuarios seguir el rendimiento de sus inversiones actuales y compararlo con las predicciones pasadas para ajustar estrategias. Personalización y Aprendizaje: La app aprendería de las interacciones del usuario, refinando sus predicciones y recomendaciones para ser cada vez más precisas y relevantes.

Para desarrollar una aplicación multiplataforma de gestión de capital, utilizaria React para el frontend web y móvil con React Native, Django para un backend robusto, y una combinación de bases de datos relacionales para datos de usuario y no relacionales para gastos, ingresos, y predicciones. Esta stack tecnológica ofrece un equilibrio perfecto entre rendimiento, escalabilidad y desarrollo eficiente.

Ejercicio 2

A partir del juego de datos utilizado en el ejemplo de la PEC, realiza las tareas previas a la generación de un modelo de minería de datos explicadas en los módulos "El proceso de minería de datos" y "Preprocesado de los datos y gestión de características".

Puedes utilizar de referencia el ejemplo de la PEC, pero procura cambiar el enfoque y analizar los datos en función de las diferentes dimensiones que presentan los datos. Así, no se puede utilizar la combinación de variables utilizada en el ejemplo: "FATALS","DRUNK_DR","VE_TOTAL","VE_FORMS","PVH_INVL","PEDS","PERSONS Se debe analizar cualquier otra combinación que puede incluir (o no) algunas de estas variables con otras nuevas.

Opcionalmente y valorable se pueden añadir al estudio datos de otros años para realizar comparaciones temporales (https://www.nhtsa.gov/file-downloads?p=nhtsa/downloads/FARS/) o añadir otros hechos a estudiar relacionados, por ejemplo, el consumo de drogas en los accidentes (https://static.nhtsa.gov/nhtsa/downloads/FARS/2020/National/FARS2020NationalCSV.zip)

Escribe aquí la respuesta a la pregunta

1. Comprensión del Negocio y de los Datos

Objetivo del Análisis:

Predecir la severidad de los accidentes basándose en la presencia de sustancias como drogas, climatologia y visibilidad.

```
# Cargamos el juego de datos
path = 'accident.CSV'
accidentData <- read.csv(path, row.names=NULL)

# Segundo set de datos sobre el consumo de drogas:
path2 = 'drugs.csv'
drugData <- read.csv(path2, row.names=NULL)
```

Estructura de los datos de accident.csv: ****

```
structure = str(accidentData)
```

```
## 'data.frame':    35766 obs. of  81 variables:
##  $ STATE       : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ STATENAME   : chr  "Alabama" "Alabama" "Alabama" "Alabama" ...
##  $ ST_CASE     : int  10001 10002 10003 10004 10005 10006 10007 10008 10009 10010 ...
##  $ VE_TOTAL    : int  1 4 2 1 1 2 1 2 2 2 ...
##  $ VE_FORMS    : int  1 4 2 1 1 2 1 2 2 2 ...
##  $ PVH_INVL    : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ PEDS        : int  0 0 0 0 0 0 1 0 0 0 ...
##  $ PERSONS     : int  4 6 2 5 1 3 1 2 4 3 ...
##  $ PERMVIT     : int  4 6 2 5 1 3 1 2 4 3 ...
##  $ PERNOTMVIT  : int  0 0 0 0 0 0 1 0 0 0 ...
##  $ COUNTY      : int  51 73 117 15 37 103 73 25 45 95 ...
##  $ COUNTYNAME  : chr  "ELMORE (51)" "JEFFERSON (73)" "SHELBY (117)" "CALHOUN (15)" ...
##  $ CITY        : int  0 350 0 0 0 0 330 0 0 1500 ...
```

```
##  $ CITYNAME    : chr  "NOT APPLICABLE" "BIRMINGHAM" "NOT APPLICABLE" "NOT APPLICABLE" ...
##  $ DAY         : int  1 2 2 3 4 4 7 8 9 10 ...
##  $ DAYNAME     : int  1 2 2 3 4 4 7 8 9 10 ...
##  $ MONTH       : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ MONTHNAME   : chr  "January" "January" "January" "January" ...
##  $ YEAR        : int  2020 2020 2020 2020 2020 2020 2020 2020 2020 2020 ...
##  $ DAY_WEEK    : int  4 5 5 6 7 7 3 4 5 6 ...
##  $ DAY_WEEKNAME: chr  "Wednesday" "Thursday" "Thursday" "Friday" ...
##  $ HOUR        : int  2 17 14 15 0 16 19 7 20 10 ...
##  $ HOURNAME    : chr  "2:00am-2:59am" "5:00pm-5:59pm" "2:00pm-2:59pm" "3:00pm-3:59pm" ...
##  $ MINUTE      : int  58 18 55 20 45 55 23 15 0 2 ...
##  $ MINUTENAME  : chr  "58" "18" "55" "20" ...
##  $ NHS         : int  0 0 0 0 0 0 0 0 0 1 ...
##  $ NHSNAME     : chr  "This section IS NOT on the NHS" "This section IS NOT on the NHS" "This sectio
##  $ ROUTE       : int  4 6 3 4 4 3 4 4 4 2 ...
##  $ ROUTENAME   : chr  "County Road" "Local Street - Municipality" "State Highway" "County Road" ...
##  $ TWAY_ID     : chr  "cr-4" "martin luther king jr dr" "sr-76" "CR-ALEXANDRIA WELLINGTON RD" ...
##  $ TWAY_ID2    : chr  "" "" "us-280" "" ...
##  $ RUR_URB     : int  1 2 1 1 1 1 2 1 1 1 ...
##  $ RUR_URBNAME : chr  "Rural" "Urban" "Rural" "Rural" ...
##  $ FUNC_SYS    : int  5 4 4 7 5 4 4 5 5 3 ...
##  $ FUNC_SYSNAME: chr  "Major Collector" "Minor Arterial" "Minor Arterial" "Local" ...
##  $ RD_OWNER    : int  2 4 1 2 2 1 4 2 2 1 ...
##  $ RD_OWNERNAME: chr  "County Highway Agency" "City or Municipal Highway Agency" "State Highway Agen
##  $ MILEPT      : int  0 0 49 0 0 390 0 0 0 3019 ...
##  $ MILEPTNAME  : chr  "None" "None" "49" "None" ...
##  $ LATITUDE    : num  32.4 33.5 33.3 33.8 32.8 ...
##  $ LATITUDENAME: chr  "32.43313333" "33.48465833" "33.29994167" "33.79507222" ...
##  $ LONGITUD    : num  -86.1 -86.8 -86.4 -85.9 -86.1 ...
##  $ LONGITUDNAME: chr  "-86.09485" "-86.83954444" "-86.36964167" "-85.88348611" ...
##  $ SP_JUR      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ SP_JURNAME  : chr  "No Special Jurisdiction" "No Special Jurisdiction" "No Special Jurisdiction"
##  $ HARM_EV     : int  42 12 34 42 42 12 8 12 12 12 ...
##  $ HARM_EVNAME : chr  "Tree (Standing Only)" "Motor Vehicle In-Transport" "Ditch" "Tree (Standing On
##  $ MAN_COLL    : int  0 6 0 0 0 2 0 1 1 2 ...
##  $ MAN_COLLNAME: chr  "The First Harmful Event was Not a Collision with a Motor Vehicle in Transport
##  $ RELJCT1     : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ RELJCT1NAME : chr  "No" "No" "No" "No" ...
##  $ RELJCT2     : int  1 1 3 1 1 1 3 1 8 1 ...
##  $ RELJCT2NAME : chr  "Non-Junction" "Non-Junction" "Intersection-Related" "Non-Junction" ...
##  $ TYP_INT     : int  1 1 3 1 1 1 2 1 1 1 ...
##  $ TYP_INTNAME : chr  "Not an Intersection" "Not an Intersection" "T-Intersection" "Not an Intersecti
##  $ WRK_ZONE    : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ WRK_ZONENAME: chr  "None" "None" "None" "None" ...
##  $ REL_ROAD    : int  4 1 4 4 4 1 1 1 1 1 ...
##  $ REL_ROADNAME: chr  "On Roadside" "On Roadway" "On Roadside" "On Roadside" ...
##  $ LGT_COND    : int  2 3 1 1 2 2 3 1 2 1 ...
##  $ LGT_CONDNAME: chr  "Dark - Not Lighted" "Dark - Lighted" "Daylight" "Daylight" ...
##  $ WEATHER     : int  1 2 2 10 2 1 1 1 10 10 ...
##  $ WEATHERNAME : chr  "Clear" "Rain" "Rain" "Cloudy" ...
##  $ SCH_BUS     : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ SCH_BUSNAME : chr  "No" "No" "No" "No" ...
##  $ RAIL        : chr  "0000000" "0000000" "0000000" "0000000" ...
##  $ RAILNAME    : chr  "Not Applicable" "Not Applicable" "Not Applicable" "Not Applicable" ...
```

```
##  $ NOT_HOUR    : int  99 17 14 99 0 17 19 7 20 10 ...
##  $ NOT_HOURNAME: chr  "Unknown" "5:00pm-5:59pm" "2:00pm-2:59pm" "Unknown" ...
##  $ NOT_MIN     : int  99 18 58 99 45 0 23 21 0 3 ...
##  $ NOT_MINNAME : chr  "Unknown" "18" "58" "Unknown" ...
##  $ ARR_HOUR    : int  3 17 15 99 0 17 19 7 20 10 ...
##  $ ARR_HOURNAME: chr  "3:00am-3:59am" "5:00pm-5:59pm" "3:00pm-3:59pm" "Unknown EMS Scene Arrival Hou
##  $ ARR_MIN     : int  10 26 15 99 55 19 29 28 10 7 ...
##  $ ARR_MINNAME : chr  "10" "26" "15" "Unknown EMS Scene Arrival Minutes" ...
##  $ HOSP_HR     : int  99 99 99 99 88 18 88 88 99 10 ...
##  $ HOSP_HRNAME : chr  "Unknown" "Unknown" "Unknown" "Unknown" ...
##  $ HOSP_MN     : int  99 99 99 99 88 51 88 88 99 29 ...
##  $ HOSP_MNNAME : chr  "Unknown EMS Hospital Arrival Time" "Unknown EMS Hospital Arrival Time" "Unknou
##  $ FATALS      : int  3 1 1 1 1 1 1 1 1 1 ...
##  $ DRUNK_DR    : int  1 0 0 0 0 0 0 0 0 0 ...
```

35766 registros **** Estructura de los datos de drogas: ****

```
structure = str(drugData)
```

```
## 'data.frame':    107141 obs. of  9 variables:
##  $ STATE       : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ STATENAME   : chr  "Alabama" "Alabama" "Alabama" "Alabama" ...
##  $ ST_CASE     : int  10001 10001 10001 10001 10002 10002 10002 10002 10002 10002 ...
##  $ VEH_NO      : int  1 1 1 1 1 1 1 1 2 3 ...
##  $ PER_NO      : int  1 2 3 4 1 1 1 2 1 1 ...
##  $ DRUGSPEC    : int  1 0 1 1 1 1 1 0 0 0 ...
##  $ DRUGSPECNAME: chr  "Whole Blood" "Test Not Given" "Whole Blood" "Whole Blood" ...
##  $ DRUGRES     : int  1 0 1 1 151 402 177 0 0 0 ...
##  $ DRUGRESNAME : chr  "Tested, No Drugs Found/Negative" "Test Not Given" "Tested, No Drugs Found/Nega
```

Resumen del archivo drugs.csv

Número total de entradas (filas): 107,141 Número total de columnas: 9 Columnas y sus tipos de datos: Este conjunto de datos incluye principalmente datos numéricos (int64) y algunos campos categóricos (object).

Columnas destacadas:

- **STATE, STATENAME:** Estado donde ocurrió el accidente.
- **ST_CASE:** Número de caso, una clave que podría usarse para relacionar estos datos con el conjunto de datos de accidentes.
- **VEH_NO:** Número de vehículo involucrado en el accidente.
- **PER_NO:** Número de la persona involucrada en el análisis de drogas.
- **DRUGSPEC:** Código especificando el tipo de especimen tomado para la prueba de drogas.
- **DRUGSPECNAME:** Nombre del tipo de especimen.
- **DRUGRES:** Resultado de la prueba de drogas.
- **DRUGRESNAME:** Nombre del resultado de la prueba de drogas (por ejemplo, "Tested, No Drugs Found/Negative", "FENTANYL").

Este conjunto de datos proporciona información detallada sobre los resultados de las pruebas de drogas relacionadas con los accidentes, incluyendo si se encontraron drogas en los involucrados y qué tipo de drogas fueron detectadas.

2. Preprocesamiento de Datos

Limpieza

Identificar y tratar valores faltantes, errores o valores atípicos. En el caso del consumo de drogas, verificar la consistencia de los datos entre los distintos sets.

El siguiente paso será la limpieza de datos, mirando si hay valores vacíos o nulos.

```
print('NA')
```

Comprobando set de datos de drugsData

```
## [1] "NA"
```

```
colSums(is.na(drugData))
```

```
##         STATE     STATENAME       ST_CASE        VEH_NO        PER_NO      DRUGSPEC
##             0             0             0             0             0             0
## DRUGSPECNAME      DRUGRES  DRUGRESNAME
##             0             0             0
```

```
print('Blancos')
```

```
## [1] "Blancos"
```

```
colSums(drugData=="")
```

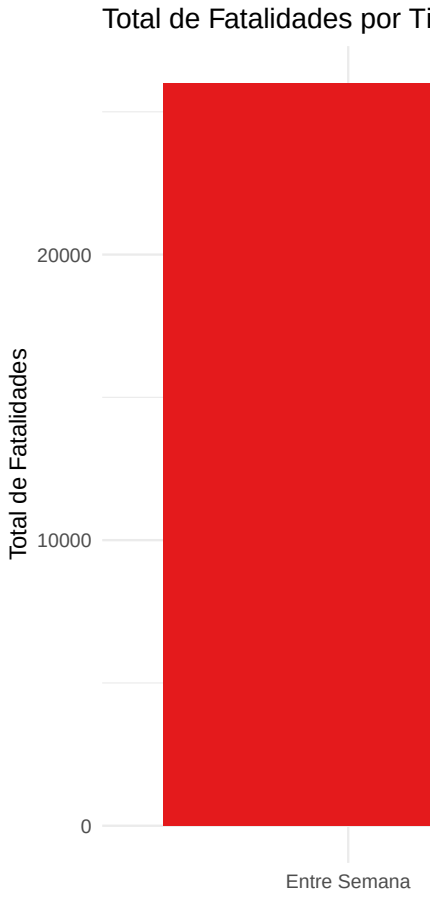
```
##         STATE     STATENAME       ST_CASE        VEH_NO        PER_NO      DRUGSPEC
##             0             0             0             0             0             0
## DRUGSPECNAME      DRUGRES  DRUGRESNAME
##             0             0             0
```

Conjunto de Datos de Accidentes (accident): Como ya sabemos por el analisis anterior tenemos que TWAY_ID2(vía de tránsito (2004)) tiene 26,997 valores en blanco y como no es relvante para nuestro estudio no haremos nada con estos datos.

Conjunto de Datos de Drogas (drugs): No hay datos nulos o faltantes en ninguna de las columnas.

```
#install.packages("dplyr")
# Cargamos las librerías necesarias
library(readr)
library(dplyr)
library(ggplot2)

# Seleccionamos las columnas de interés
```

```r
accidentDataSelected <- accidentData %>%
  select(ST_CASE, STATE, STATENAME, FATALS, DAY_WEEK)

# Añadir columna 'Tipo_de_día'
accidentDataSelected <- accidentDataSelected %>%
  mutate(DAY_TYPE = ifelse(DAY_WEEK %in% c(1, 7), "Fin de Semana", "Entre Semana"))

# Asumiendo que accidentDataSelected ya tiene la columna DAY_TYPE correctamente añadida
fatalitiesByDayType <- accidentDataSelected %>%
  group_by(DAY_TYPE) %>%
  summarise(Total_Fatalities = sum(FATALS))

# Crear gráfico
p <- ggplot(fatalitiesByDayType, aes(x = DAY_TYPE, y = Total_Fatalities, fill = DAY_TYPE)) +
  geom_bar(stat = "identity") +
  theme_minimal() +
  labs(title = "Total de Fatalidades por Tipo de Día",
       x = "Tipo de Día",
       y = "Total de Fatalidades") +
  scale_fill_brewer(palette = "Set1")

ggsave("total_fatalidades_por_tipo_de_dia.png", plot = p, width = 8, height = 6)
# Mostrar el gráfico
print(p)
```



Vamos a comprobar si los fines de semana ocurren mas accidentes mortales:

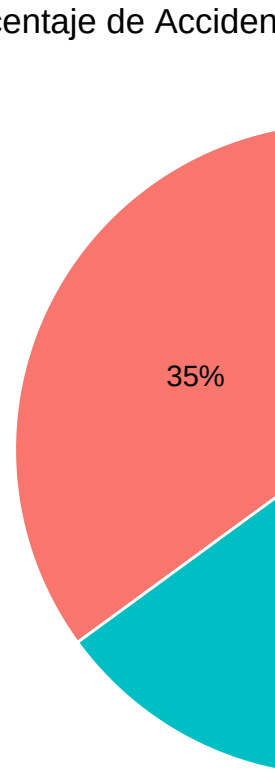
```
# Calcular el total de fatalidades
total_fatalities <- sum(fatalitiesByDayType$Total_Fatalities)
#total_fatalities

# Añadir una columna de porcentaje
fatalitiesByDayType$Percentage <- (fatalitiesByDayType$Total_Fatalities / total_fatalities) * 100

# Ver los resultados
print(fatalitiesByDayType)
```

Calculamos el porcentage de diferencia entre las dos columnas:

```
## # A tibble: 2 x 3
##   DAY_TYPE      Total_Fatalities Percentage
##   <chr>                    <int>      <dbl>
## 1 Entre Semana             25998       67.0
## 2 Fin de Semana            12826       33.0
```

Del total 38824 de las fatalidades registradas, el 66.96% ocurrieron entre semana, mientras que el 33.04% sucedieron durante el fin de semana. Esto indica que hay una mayor cantidad de muertes durante los días de semana en comparación con los fines de semana dentro de este conjunto de datos.

```r
# Primero, necesitamos unir los datasets por 'ST_CASE'
mergedData <- merge(accidentData, drugData, by = "ST_CASE")

# Definimos una nueva columna basada en la clasificación de resultados de drogas
mergedData <- mergedData %>%
  mutate(DRUG_RESULT_ADJUSTED = case_when(
    DRUGRES %in% 100:996 ~ "Positivo",
    DRUGRES == 998 ~ "Positivo",
    DRUGRES %in% c(95,997, 999) ~ "Desconocido",
    TRUE ~ "Negativo o No Probado"
  ))

# Identificar accidentes con un muerto o mas
fatalAccidents <- accidentData %>%
  filter(FATALS > 0) %>%
  select(ST_CASE, FATALS)

# Encontrar accidentes con al menos un positivo en drogas
accidentsWithDrugs <- mergedData %>%
  filter(DRUG_RESULT_ADJUSTED == "Positivo") %>%
  distinct(ST_CASE) %>%
  inner_join(fatalAccidents, by = "ST_CASE")

# Contar estos accidentes
accidentsCount <- nrow(accidentsWithDrugs)

# Calcular el total de accidentes fatales
totalFatalAccidents <- nrow(fatalAccidents)

# Calcular el número de accidentes sin resultados positivos en drogas
accidentsWithoutDrugs <- totalFatalAccidents - accidentsCount

# Preparar datos para el gráfico
dataForPlot <- data.frame(
  Category = c("Con Drogas", "Sin Drogas"),
  Count = c(accidentsCount, accidentsWithoutDrugs)
)

# Calcular porcentajes
dataForPlot$Percentage <- dataForPlot$Count / sum(dataForPlot$Count) * 100

# Crear gráfico
ggplot(dataForPlot, aes(x = "", y = Percentage, fill = Category)) +
  geom_bar(width = 1, stat = "identity", color = "white") +
  coord_polar("y", start = 0) +
  theme_void() +
  labs(title = "Porcentaje de Accidentes Fatales con y sin Resultados Positivos en Drogas",
       fill = "Categoría") +
  geom_text(aes(label = paste0(round(Percentage, 1), "%")), position = position_stack(vjust = 0.5))
```



Voy a buscar una relación entre accidentes fatales y el consumo de drogas:

```
# Mostrar el gráfico
ggsave("accidentes_drogas_porcentaje.png", width = 8, height = 6)

# Imprimir los resultados
cat("Número de accidentes con un muerto o mas y al menos un positivo en drogas:", accidentsCount, "\n")
```

```
## Número de accidentes con un muerto o mas y al menos un positivo en drogas: 12534
```

```
cat("Número total de accidentes fatales:", totalFatalAccidents, "\n")
```

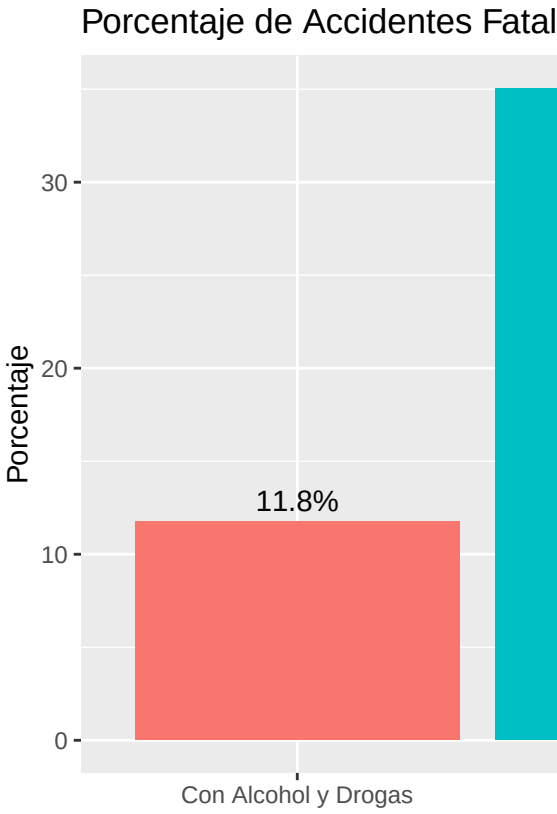
```
## Número total de accidentes fatales: 35766
```

35% de las fatalidades ocurrieron en casos donde se encontró almenos un resultado positivo en las pruebas de drogas. 65% de las fatalidades están asociadas a casos donde los resultados de las pruebas de drogas fueron negativos o no se realizaron.

Este análisis resalta la importancia de las medidas preventivas y la educación como herramientas clave para mejorar la seguridad vial y reducir el número de accidentes fatales relacionados con el consumo de drogas.

```
# Unimos los datasets por 'ST_CASE'
mergedData <- merge(accidentData, drugData, by = "ST_CASE")
```

```r
# Definimos 'DRUG_RESULT_ADJUSTED' basado en la clasificación de los resultados de drogas
mergedData <- mergedData %>%
  mutate(DRUG_RESULT_ADJUSTED = case_when(
    DRUGRES %in% 100:996 ~ "Positivo",
    DRUGRES == 998 ~ "Positivo",
    DRUGRES %in% c(95,997, 999) ~ "Desconocido",
    TRUE ~ "Negativo o No Probado"
  ))

# Filtrar accidentes con un muerto o más
fatalAccidents <- accidentData %>%
  filter(FATALS > 0)

# Contar todos los accidentes fatales únicos
totalFatalAccidents <- nrow(fatalAccidents)

# Encontrar accidentes fatales con al menos un positivo en drogas
accidentsWithDrugs <- mergedData %>%
  filter(DRUG_RESULT_ADJUSTED == "Positivo") %>%
  distinct(ST_CASE)
accidentsWithDrugsCount <- nrow(accidentsWithDrugs)

# Calcular accidentes con al menos un resultado positivo en drogas y alcohol
accidentsWithAlcoholAndDrugs <- mergedData %>%
  filter(DRUNK_DR > 0, DRUG_RESULT_ADJUSTED == "Positivo") %>%
  distinct(ST_CASE)

# Contar estos accidentes
accidentsWithAlcoholAndDrugsCount <- nrow(accidentsWithAlcoholAndDrugs)

# Preparar datos para el gráfico comparativo
dataForPlot <- data.frame(
  Category = c("Con Drogas", "Con Alcohol y Drogas"),
  Count = c(nrow(accidentsWithDrugs), accidentsWithAlcoholAndDrugsCount)
)

# Calcular porcentajes basados en el total de accidentes fatales
dataForPlot$Percentage <- dataForPlot$Count / totalFatalAccidents * 100

# Crear gráfico
ggplot(dataForPlot, aes(x = Category, y = Percentage, fill = Category)) +
  geom_col() +
  labs(title = "Porcentaje de Accidentes Fatales con Drogas y Alcohol",
       y = "Porcentaje",
       x = "") +
  geom_text(aes(label = sprintf("%.1f%%", Percentage)), vjust = -0.5)
```



Vamos a ver la relacion entre accidentes mortales, drogas y alchol

```
# Mostrar el gráfico
ggsave("accidentes_alcohol_drogas_porcentaje.png", width = 8, height = 6)

cat("Total de accidentes fatales ", totalFatalAccidents)
```

```
## Total de accidentes fatales  35766
```

```
cat("\nTotal de accidentes fatales con drogas ", accidentsWithDrugsCount)
```

```
##
## Total de accidentes fatales con drogas  12534
```

```
cat("\nTotal de accidentes fatales con drogas y alcohol", accidentsWithAlcoholAndDrugsCount)
```

```
##
## Total de accidentes fatales con drogas y alcohol 4205
```

El gráfico muestra claramente la relación entre accidentes fatales y el consumo de drogas, con y sin la combinación de alcohol. De los 35,766 accidentes fatales registrados, 4205 (aproximadamente el 11.8%) implicaron tanto drogas como alcohol, mientras que un porcentaje significativamente mayor de 35% estuvo asociado exclusivamente con drogas. Esto sugiere que, si bien el alcohol sigue siendo un factor considerable en los accidentes de tráfico con fatalidades, el uso de drogas representa una preocupación más predominante.

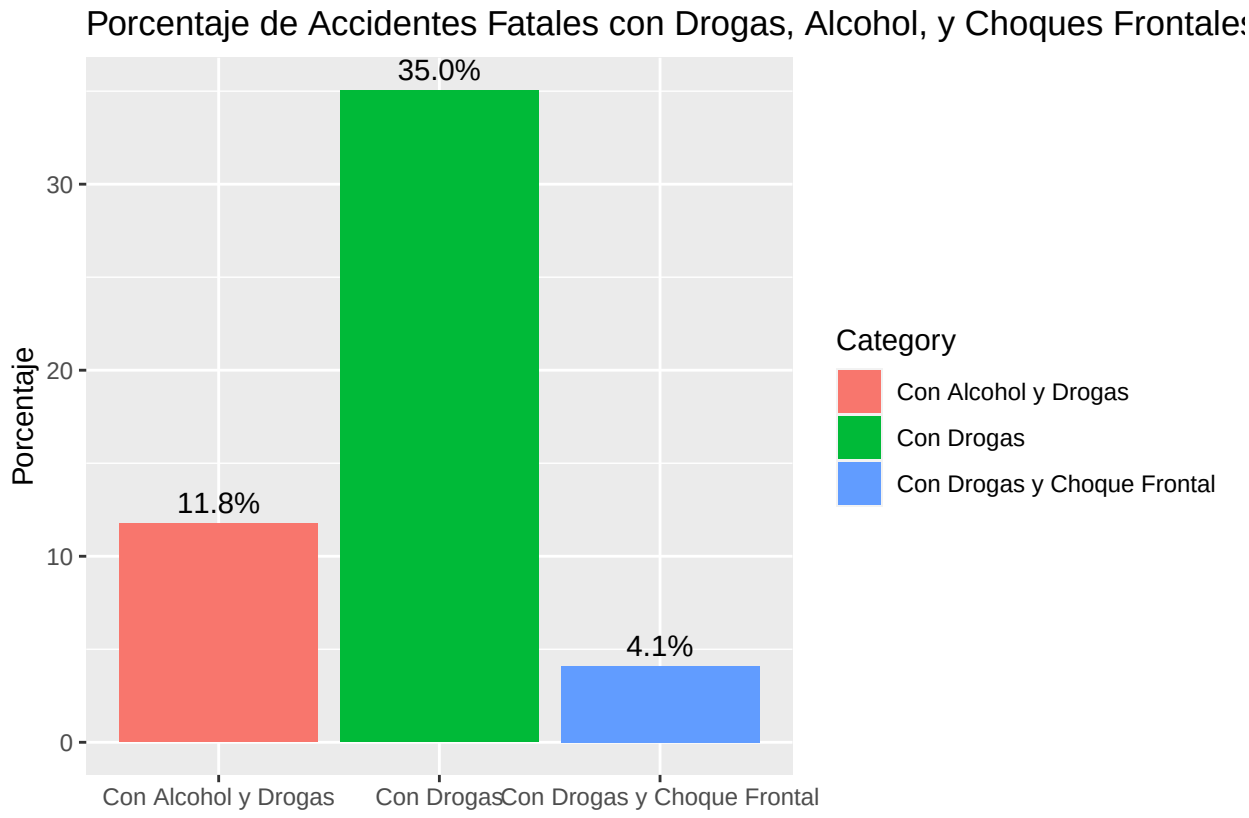
El gráfico y los números indican que del total de accidentes fatales (35,766), un 11.8% de los accidentes involucraron tanto alcohol como drogas. Este porcentaje está incluido en el 35% de accidentes asociados con drogas, lo que significa que el 35% no es exclusivamente drogas, sino que incluye casos con solo drogas y casos con drogas y alcohol

```r
# Asumiendo que mergedData ya contiene 'DRUG_RESULT_ADJUSTED'

# Identificar accidentes fatales con al menos un positivo en drogas que fueron choques frontales (MAN_C
accidentsWithDrugsFrontalCollision <- mergedData %>%
  filter(DRUG_RESULT_ADJUSTED == "Positivo", MAN_COLL == 2) %>%
  distinct(ST_CASE)

# Contar estos accidentes
accidentsWithDrugsFrontalCollisionCount <- nrow(accidentsWithDrugsFrontalCollision)

# Añadir al gráfico comparativo la nueva categoría de "Con Drogas y Choque Frontal"
# Asegurándonos de que el nuevo data frame tenga las mismas columnas que 'dataForPlot'
dataForPlotExpanded <- rbind(dataForPlot,
                             data.frame(Category = "Con Drogas y Choque Frontal",
                                        Count = accidentsWithDrugsFrontalCollisionCount,
                                        Percentage = NA)) # Añadimos NA para la columna 'Percentage' qu

# Recalcular porcentajes basados en el total de accidentes fatales para asegurarnos de que todos los po
dataForPlotExpanded$Percentage <- dataForPlotExpanded$Count / totalFatalAccidents * 100
print(dataForPlotExpanded)
```

Ahora vamos a ver la relación entre accidentes mortales frontales y drogas:

```
##                       Category Count Percentage
## 1                   Con Drogas 12534  35.044456
## 2        Con Alcohol y Drogas  4205  11.756976
## 3 Con Drogas y Choque Frontal  1463   4.090477
```

```r
# Crear gráfico actualizado
ggplot(dataForPlotExpanded, aes(x = Category, y = Percentage, fill = Category)) +
  geom_col() +
  labs(title = "Porcentaje de Accidentes Fatales con Drogas, Alcohol, y Choques Frontales",
       y = "Porcentaje",
       x = "") +
  geom_text(aes(label = sprintf("%.1f%%", Percentage)), vjust = -0.5)
```




```
# Mostrar el gráfico actualizado
ggsave("accidentes_alcohol_drogas_choque_frontal_porcentaje.png", width = 8, height = 6)
```

Con Drogas (35%): Una cantidad significativa de los accidentes fatales involucran drogas, lo que resalta la influencia significativa del consumo de drogas en la ocurrencia de accidentes mortales.

Con Alcohol y Drogas (11.8%): Una parte considerable de los accidentes fatales incluye tanto alcohol como drogas. Esto apunta a que la combinación de estas sustancias es un factor de riesgo relevante en los accidentes de tráfico mortales.

Con Drogas y Choque Frontal (4.1%): Una menor, pero aún notable, proporción de accidentes fatales implica el uso de drogas en colisiones frontales. Este tipo de colisión es particularmente peligroso y a menudo resulta en consecuencias graves debido a la fuerza del impacto.

Estos insights indican que el consumo de drogas es un factor importante en los accidentes de tráfico mortales y que es aún más crítico cuando se combina con alcohol. Además, los choques frontales, que son altamente peligrosos por sí mismos, son más graves cuando están asociados con el uso de drogas. Estos datos deben servir para informar políticas de tráfico y estrategias de intervención dirigidas a reducir la conducción bajo la influencia de sustancias psicoactivas para mejorar la seguridad vial.

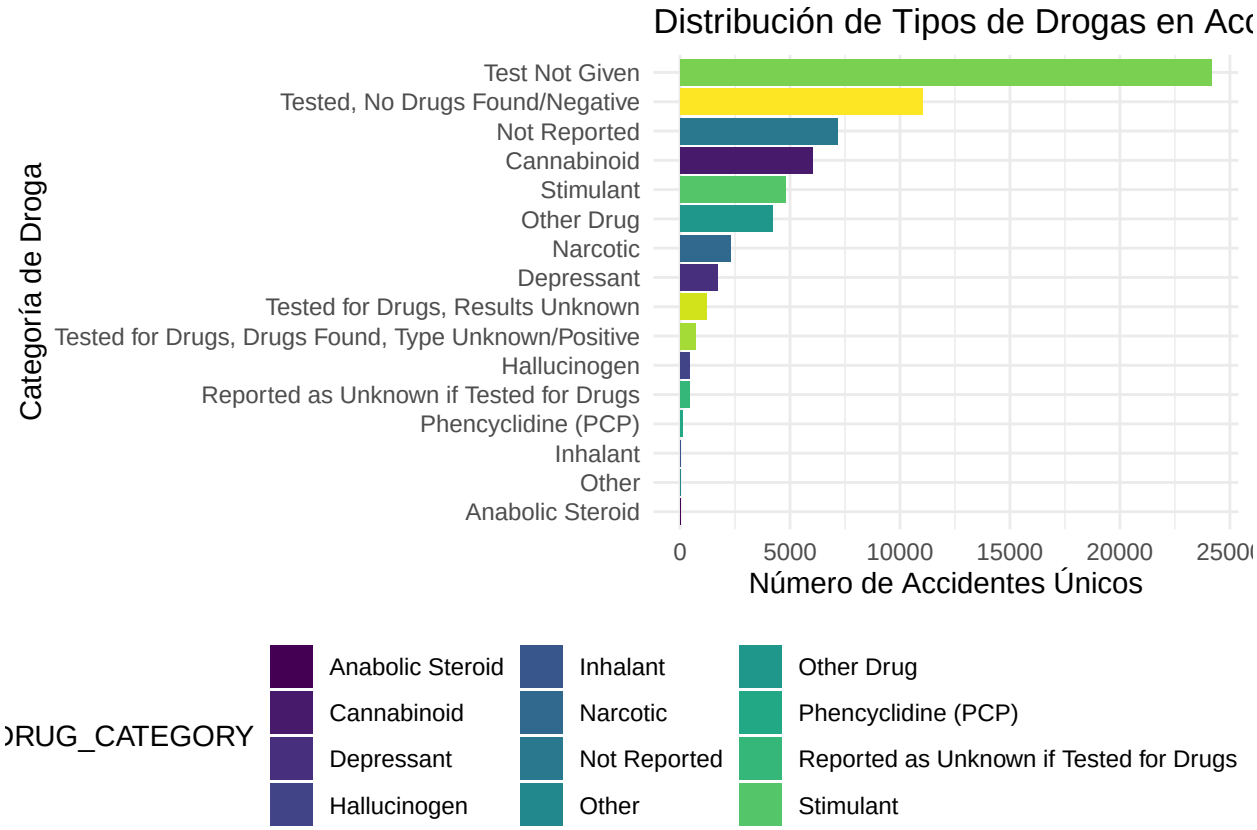
```
# Cargar los datos
accident_data <- read_csv('accident.CSV', locale = locale(encoding = "latin1"))
drug_data <- read_csv('drugs.csv', locale = locale(encoding = "latin1"))
```

```r
# Unir los datos basados en 'ST_CASE'
merged_data <- inner_join(accident_data, drug_data, by = 'ST_CASE')

# Clasificar los resultados de las pruebas de drogas
merged_data <- merged_data %>%
  mutate(DRUG_CATEGORY = case_when(
    DRUGRES == 0 ~ "Test Not Given",
    DRUGRES == 1 ~ "Tested, No Drugs Found/Negative",
    DRUGRES == 95 ~ "Not Reported",
    DRUGRES >= 100 & DRUGRES <= 295 ~ "Narcotic",
    DRUGRES >= 300 & DRUGRES <= 395 ~ "Depressant",
    DRUGRES >= 400 & DRUGRES <= 495 ~ "Stimulant",
    DRUGRES >= 500 & DRUGRES <= 595 ~ "Hallucinogen",
    DRUGRES >= 600 & DRUGRES <= 695 ~ "Cannabinoid",
    DRUGRES >= 700 & DRUGRES <= 795 ~ "Phencyclidine (PCP)",
    DRUGRES >= 800 & DRUGRES <= 895 ~ "Anabolic Steroid",
    DRUGRES >= 900 & DRUGRES <= 995 ~ "Inhalant",
    DRUGRES == 996 ~ "Other Drug",
    DRUGRES == 997 ~ "Tested for Drugs, Results Unknown",
    DRUGRES == 998 ~ "Tested for Drugs, Drugs Found, Type Unknown/Positive",
    DRUGRES == 999 ~ "Reported as Unknown if Tested for Drugs",
    TRUE ~ "Other"
  ))

# Conteo de accidentes únicos por categoría de droga
drugTypeCounts <- merged_data %>%
  group_by(DRUG_CATEGORY) %>%
  summarise(Count = n_distinct(ST_CASE)) %>%
  arrange(desc(Count))

print(drugTypeCounts)
```

Ahora voy a ver la relacion de los diferentes tipos de droga con su consumo y la implicación en los accidentes

```
## # A tibble: 16 x 2
##    DRUG_CATEGORY                                        Count
##    <chr>                                                <int>
##  1 Test Not Given                                       24178
##  2 Tested, No Drugs Found/Negative                      11041
##  3 Not Reported                                          7159
##  4 Cannabinoid                                           6022
##  5 Stimulant                                             4797
##  6 Other Drug                                            4188
##  7 Narcotic                                              2317
##  8 Depressant                                            1687
##  9 Tested for Drugs, Results Unknown                     1187
## 10 Tested for Drugs, Drugs Found, Type Unknown/Positive   721
## 11 Hallucinogen                                           451
## 12 Reported as Unknown if Tested for Drugs                448
## 13 Phencyclidine (PCP)                                    134
## 14 Inhalant                                                 9
## 15 Other                                                    6
## 16 Anabolic Steroid                                         1
```

```r
# Visualización de la distribución de tipos de droga
ggplot(drugTypeCounts, aes(x = reorder(DRUG_CATEGORY, Count), y = Count, fill = DRUG_CATEGORY)) +
  geom_col() +
  coord_flip() +
  labs(title = "Distribución de Tipos de Drogas en Accidentes Únicos",
       x = "Categoría de Droga",
       y = "Número de Accidentes Únicos") +
  theme_minimal() +
  scale_fill_viridis_d() +
  theme(legend.position = "bottom")
```



Test Not Given (24,178): La mitad de los casos no incluyeron una prueba de drogas, implicando ausencia de datos sobre la presencia de drogas en estos incidentes. Esto podría reflejar políticas de prueba selectiva, limitaciones de recursos, o la naturaleza específica de cada incidente.

Tested, No Drugs Found/Negative (11,041): Un número significativo de pruebas no reveló la presencia de drogas, indicando resultados negativos. Esto muestra que, aunque se realizan pruebas, no siempre se encuentran sustancias ilegales.

Cannabinoid (6,022): Los cannabinoides representan la tercera categoría más común, sugiriendo que esta clase de sustancia es frecuentemente detectada en accidentes de tráfico donde se realizan pruebas de drogas.

Not Reported (7,159): Una proporción relevante de los informes carece de datos sobre los resultados de las pruebas de drogas, lo cual plantea desafíos para la interpretación de la incidencia total de drogas en accidentes de tráfico y podría influir en la formulación de políticas.

Stimulant (4,797): Los estimulantes también se detectan comúnmente, lo que podría indicar su uso prevalente entre individuos involucrados en accidentes de tráfico.

Other Drug (4,188): Una cantidad considerable de accidentes están vinculados con drogas que no encajan en las otras categorías específicas, sugiriendo una diversidad de sustancias implicadas.

Narcotic (2,317): Aunque menos frecuentes que los cannabinoides y estimulantes, los narcóticos están presentes en una cantidad significativa de accidentes, requiriendo atención debido a sus efectos potentes.

Depressant (1,687): Los depresores, aunque no tan comunes como otras categorías, están asociados con un número notable de accidentes.

Tested for Drugs, Results Unknown (1,187): Existe un grupo de casos donde, a pesar de haberse realizado pruebas, los resultados no se conocen o no se reportaron.

Tested for Drugs, Drugs Found, Type Unknown/Positive (721): Una fracción menor de incidentes donde las pruebas de drogas fueron positivas, pero no se pudo determinar el tipo de droga.

Esta revisión refleja un panorama de la presencia de drogas en accidentes de tráfico, destacando la frecuencia de casos sin prueba de drogas, la incidencia de resultados negativos, y la presencia notable de cannabinoides y estimulantes entre los positivos. La diversidad de sustancias implicadas subraya la complejidad de abordar la conducción bajo la influencia de drogas.

3. Ingeniería de Características:

Normalización/Escalado:

Selección de Variables Relevantes

Del primer conjunto de datos (accidentes):

- **ST_CASE:** Identificador de accidente, esencial para unir los conjuntos de datos.
- **FATALS:** Muertes, para medir la severidad.
- **DRUNK_DR:** Conductores ebrios, indica la influencia del alcohol.
- **LGT_COND:** Condición lumínica, podría afectar la visibilidad y la probabilidad de accidentes graves.
- **WEATHER:** Condiciones meteorológicas, factores externos que pueden influir en la severidad.

Del segundo conjunto de datos (consumo de drogas):

- **ST_CASE:** Para vincular con el conjunto de datos de accidentes.
- **DRUGRES:** Resultado de la prueba de drogas, indicador clave de la presencia de sustancias.

```
# Cargamos el juego de datos
path = 'accident.CSV'
accidents_df <- read.csv(path, row.names=NULL, encoding = 'latin1')

# Segundo set de datos sobre el consumo de drogas:
path2 = 'drugs.csv'
drugs_df <- read.csv(path2, row.names=NULL)
#print(accidents_df)
```

```
library(dplyr)
# Seleccionar variables relevantes del primer conjunto de datos
accidents_selected <- select(accidents_df, ST_CASE, FATALS, DRUNK_DR, LGT_COND, WEATHER)

# Seleccionar variables relevantes del segundo conjunto de datos
# Nota: ST_CASE ya está incluido, así que solo necesitamos añadir DRUGRES
```

```r
drugs_selected <- select(drugs_df, ST_CASE, DRUGRES)

# Integrar los datos
# Aquí optaremos por una agregación de los resultados de las pruebas de drogas por caso,
# ya que un accidente puede tener múltiples personas involucradas y por lo tanto múltiples registros en
drugs_aggregated <- drugs_selected %>%
  group_by(ST_CASE) %>%
  summarise(DRUGRES = list(DRUGRES))

# Unir los dos conjuntos de datos seleccionados
data_merged <- merge(accidents_selected, drugs_aggregated, by = "ST_CASE", all.x = TRUE)

# Visualizar las primeras filas del conjunto de datos integrado
head(data_merged)
```

```
##   ST_CASE FATALS DRUNK_DR LGT_COND WEATHER                      DRUGRES
## 1   10001      3        1        2       1                    1, 0, 1, 1
## 2   10002      1        0        3       2 151, 402, 177, 0, 0, 0, 0, 0
## 3   10003      1        0        1       2                         0, 0
## 4   10004      1        0        1      10       600, 600, 605, 0, 0, 0
## 5   10005      1        0        2       2                            0
## 6   10006      1        0        2       1     316, 151, 417, 401, 0, 0
```

1.Crear Variables Categóricas a partir de DRUGRES

Los resultados de las pruebas de drogas (DRUGRES) son transformadas en categorías más claras, como "Positivo", "Negativo", o "Desconocido". Esto permitirá analizar los datos más fácilmente y realizar comparaciones significativas entre grupos.

```r
clasificar_resultados_drugres_v2 <- function(lista_resultados) {

  # Verificar si la lista está vacía o es NA, lo cual puede pasar si el caso no tiene datos de drogas
  if (is.null(lista_resultados) || length(lista_resultados) == 0 || all(is.na(lista_resultados))) {
    return("Desconocido")
  }

  # todos los elementos son numéricos para la comparación
  lista_resultados <- sapply(lista_resultados, as.numeric)

  # Eliminar NA antes de realizar la clasificación para evitar errores en las comprobaciones
  lista_resultados <- lista_resultados[!is.na(lista_resultados)]

  # Clasificación basada en los códigos proporcionados
  positivos <- any(lista_resultados >= 100 & lista_resultados <= 996 | lista_resultados == 998)
  desconocidos <- any(lista_resultados %in% c(95, 997, 999))

  if (positivos) {
    return("Positivo")
  } else if (desconocidos) {
    return("Desconocido")
  } else {
    return("Negativo o No Probado")
  }
```

```
}

# Aplicar la función ajustada
data_merged$DRUGRES_CAT <- sapply(data_merged$DRUGRES, clasificar_resultados_drugres_v2)

# Visualizar las primeras filas para verificar la transformación
#head(select(data_merged, ST_CASE, DRUGRES, DRUGRES_CAT))
#print(head(data_merged))

resultados_clasificados <- sapply(data_merged$DRUGRES, clasificar_resultados_drugres_v2)
conteos <- table(resultados_clasificados)

# presentar cada conteo individualmente:
cat("\nDesglose de clasificaciones:\n")
```

```
##
## Desglose de clasificaciones:
```

```
cat("Accidentes con positivo en drogas:", conteos["Positivo"], "\n")
```

```
## Accidentes con positivo en drogas: 12534
```

```
cat("Accidentes con Desconocido:", conteos["Desconocido"], "\n")
```

```
## Accidentes con Desconocido: 7315
```

```
cat("Accidentes con Negativo o No Probado:", conteos["Negativo o No Probado"], "\n")
```

```
## Accidentes con Negativo o No Probado: 15917
```

La función sapply() se utiliza para aplicar clasificar_resultados_drugres_v2 a cada elemento de la columna DRUGRES, que se espera contenga listas de resultados de pruebas de drogas.

Positivo: DRUGRES en el rango de 100 a 996, o igual a 998. Desconocido: DRUGRES igual a 95, 997, o 999. Negativo o No Probado: Cualquier otro caso.

El motivo detrás de la clasificación propuesta, especialmente al tratar el código 95 (“Not Reported”) como “Desconocido”, es mantener la precisión y fiabilidad en la identificación de accidentes donde las drogas estaban definitivamente presentes. Al clasificar los resultados no reportados como “Desconocidos”, se evita asumir incorrectamente la presencia o ausencia de drogas, lo que podría distorsionar el análisis. Este enfoque garantiza que solo se consideren “Positivos” aquellos casos con evidencia clara de drogas, mientras que los casos sin información suficiente se manejan de manera conservadora, evitando conclusiones precipitadas. Es esencial para el análisis enfocarse en datos concretos y evitar interpretaciones erróneas o suposiciones sobre datos incompletos o no reportados.

2. Agrupar LGT_COND y WEATHER en Categorías Más Generales Las condiciones de luz (LGT_COND) y el clima (WEATHER) pueden tener muchas categorías específicas que sería útil simplificar para el análisis.

```r
library(dplyr)

# Definir funciones para agrupar LGT_COND y WEATHER en categorías más generales
agrupar_lgt_cond <- function(codigo) {
  if (codigo == 1) {
    return("Luz Diurna")
  } else if (codigo %in% c(2, 3, 6)) {
    return("Oscuridad")
  } else if (codigo %in% c(4, 5)) {
    return("Crepúsculo o Amanecer")
  } else {
    return("Otras")
  }
}

agrupar_weather <- function(codigo) {
  if (codigo %in% c(1, 10)) {
    return("Claro")
  } else if (codigo %in% c(2, 3, 4, 5, 12)) {
    return("Precipitaciones")
  } else if (codigo %in% c(6, 7, 11)) {
    return("Extremas")
  } else {
    return("Otras")
  }
}

# Aplicar las funciones para crear nuevas columnas con las categorías agrupadas
data_merged <- data_merged %>%
  mutate(LGT_COND_CAT = sapply(LGT_COND, agrupar_lgt_cond),
         WEATHER_CAT = sapply(WEATHER, agrupar_weather))

# Visualizar las primeras filas para verificar las nuevas categorías
#head(select(data_merged, ST_CASE, LGT_COND, LGT_COND_CAT, WEATHER, WEATHER_CAT))

# Conteo para LGT_COND_CAT
conteos_lgt_cond_cat <- table(data_merged$LGT_COND_CAT)

# Presentar conteos de LGT_COND_CAT línea por línea
cat("Conteos por categoría de condiciones de luz (LGT_COND_CAT):\n")
```

```
## Conteos por categoría de condiciones de luz (LGT_COND_CAT):
```

```r
for(categoria in names(conteos_lgt_cond_cat)) {
  cat(categoria, ":", conteos_lgt_cond_cat[categoria], "\n")
}
```

```
## Crepúsculo o Amanecer : 1605
## Luz Diurna : 16291
## Oscuridad : 17572
## Otras : 298
```

```r
cat("\n") # Añadir un espacio entre los conteos
```

```r
# Conteo para WEATHER_CAT
conteos_weather_cat <- table(data_merged$WEATHER_CAT)

# Presentar conteos de WEATHER_CAT línea por línea
cat("Conteos por categoría de clima (WEATHER_CAT):\n")
```

```
## Conteos por categoría de clima (WEATHER_CAT):
```

```r
for(categoria in names(conteos_weather_cat)) {
  cat(categoria, ":", conteos_weather_cat[categoria], "\n")
}
```

```
## Claro : 29585
## Extremas : 87
## Otras : 2742
## Precipitaciones : 3352
```

```r
# Visualizar todo el dataframe
#print(data_merged)
```

Condiciones de Luz (LGT_COND) Podemos simplificar las condiciones de luz en categorías como “Luz Diurna”, “Oscuridad” y “Crepúsculo o Amanecer”, junto con “Otras” para incluir casos no reportados o desconocidos.

Luz Diurna: 1 Oscuridad: 2, 3, 6 (incluye oscuridad no iluminada, oscuridad iluminada y oscuridad con iluminación desconocida) Crepúsculo o Amanecer: 4, 5 (dawn y dusk) Otras: 7, 8, 9 (otras, no reportadas, desconocidas) Condiciones Meteorológicas (WEATHER) Para el clima, podríamos agrupar en “Claro”, “Precipitaciones”, “Extremas” y “Otras” para abarcar todas las posibilidades de manera simplificada.

Claro: 1, 10 (claro, nublado sin condiciones adversas) Precipitaciones: 2, 3, 4, 5, 12 (lluvia, granizo, nieve, niebla, lluvia congelante) Extremas: 6, 7, 11 (vientos cruzados severos, arena o suciedad soplada, nieve soplada) Otras: 8, 98, 99 (otras, no reportadas, desconocidas)

Conclusión:

Condiciones de Luz y Clima

Prevalencia de Accidentes en Oscuridad: La mayor cantidad de accidentes en condiciones de oscuridad plantea preguntas sobre la visibilidad y la percepción de riesgo durante la noche. Investigar más a fondo las medidas de seguridad activas y pasivas en estos contextos podría ser beneficioso.

Impacto del Clima: Mientras que la mayoría de los accidentes ocurren en condiciones climáticas claras, los accidentes bajo condiciones de “Precipitaciones” y “Extremas” merecen atención especial. El impacto de estas condiciones en la capacidad de los conductores para reaccionar a tiempo o en la eficacia de los sistemas de seguridad vehicular podría ser un área de estudio valiosa.

ST_CASE: Identificador del caso. FATALS: Número de muertes. DRUNK_DR: Conductores bajo influencia del alcohol. MAN_COLL: Tipo de colisión. DAY_WEEK: Día de la semana. LGT_COND: Código de condición lumínica (original). WEATHER: Código de condición meteorológica (original). DRUGRES: Lista de resultados de pruebas de drogas. DRUGRES_CAT: Categoría de resultados de pruebas de drogas (“Positivo”, “Desconocido”, “Negativo o No Probado”). LGT_COND_CAT: Categorías simplificadas de condiciones lumínicas. WEATHER_CAT: Categorías simplificadas de condiciones meteorológicas.

Paso 1: Prueba de ANOVA

```
# ANOVA
resultado_anova <- aov(FATALS ~ DRUGRES_CAT, data = data_merged)
summary(resultado_anova)
```

```
##                Df Sum Sq Mean Sq F value Pr(>F)
## DRUGRES_CAT     2     14   7.051   60.34 <2e-16 ***
## Residuals   35763   4178   0.117
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Interpretación La conclusión principal aquí es que existe una asociación estadísticamente significativa entre las categorías de resultados de pruebas de drogas y el número de fallecidos en accidentes. Esto significa que, en promedio, el número de fallecidos varía de manera significativa entre al menos dos de las categorías de resultados de pruebas de drogas.

Paso 2: Prueba de TUKEY

```
# Asegurando que data_merged$DRUGRES_CAT es un factor
data_merged$DRUGRES_CAT <- as.factor(data_merged$DRUGRES_CAT)

# Ejecutar el modelo ANOVA
resultado_anova <- aov(FATALS ~ DRUGRES_CAT, data = data_merged)

# Ejecutar el test post-hoc de Tukey
tukey_res <- TukeyHSD(resultado_anova)
print(tukey_res)
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = FATALS ~ DRUGRES_CAT, data = data_merged)
##
## $DRUGRES_CAT
##                                          diff         lwr          upr    p adj
## Negativo o No Probado-Desconocido -0.01369482 -0.02501094 -0.002378695 0.012679
## Positivo-Desconocido               0.03082351  0.01903634  0.042610681 0.000000
## Positivo-Negativo o No Probado     0.04451833  0.03495156  0.054085106 0.000000
```

Interpretación de los Resultados Negativo o No Probado vs. Desconocido: La diferencia estimada entre el número medio de fallecidos para los accidentes con resultados de pruebas de drogas "Negativo o No Probado" y aquellos con resultados "Desconocido" es de -0.013695. Esta diferencia es estadísticamente significativa ($p = 0.0125$), lo que sugiere que hay menos fallecidos en promedio en accidentes donde las pruebas de drogas no se realizaron o resultaron negativas, en comparación con los accidentes donde el resultado de las pruebas de drogas es desconocido.

Positivo vs. Desconocido: La diferencia en el número medio de fallecidos entre los accidentes con resultados de pruebas de drogas "Positivo" y "Desconocido" es de 0.030824. Esta diferencia es altamente significativa ($p < 0.0001$), indicando que hay más fallecidos en promedio en accidentes con resultados de pruebas de drogas positivas en comparación con aquellos donde el resultado es desconocido.

Positivo vs. Negativo o No Probado: La diferencia entre el número medio de fallecidos para los accidentes con resultados de pruebas de drogas "Positivo" y "Negativo o No Probado" es de 0.044518. Esta diferencia también es muy significativa ($p < 0.0001$), lo que significa que hay más fallecidos en promedio en accidentes con resultados de pruebas positivas para drogas en comparación con aquellos donde las pruebas no se realizaron o resultaron negativas.

Conclusiones Los resultados sugieren que los accidentes donde se identificaron drogas a través de las pruebas tienen, en promedio, un mayor número de fallecidos en comparación con los accidentes donde las pruebas resultaron negativas o no se realizaron. Esto podría indicar que la presencia de drogas es un factor que contribuye a la gravedad de los accidentes.

Normalizar variables FATALS y DRUNK_DR

```
# Estandarización de las variables FATALS y DRUNK_DR
data_merged$FATALS <- scale(data_merged$FATALS)
data_merged$DRUNK_DR <- scale(data_merged$DRUNK_DR)

# Verificamos las primeras filas para ver el resultado
#head(data_merged)
```

Codificación

```
#install.packages("fastDummies")
library(fastDummies)

# Aplicar codificación One-Hot a DRUGRES_CAT
data_merged <- dummy_cols(data_merged, select_columns = "DRUGRES_CAT")

# Aplicar codificación One-Hot a LGT_COND_CAT
data_merged <- dummy_cols(data_merged, select_columns = "LGT_COND_CAT")

# Aplicar codificación One-Hot a WEATHER_CAT
data_merged <- dummy_cols(data_merged, select_columns = "WEATHER_CAT")

# Revisar los primeros registros para confirmar los cambios
#head(data_merged)

# Esto convertirá cada elemento de la lista en una cadena de texto
#data_merged$DRUGRES <- sapply(data_merged$DRUGRES, toString)

# Ahora intenta escribir el archivo CSV
#write.csv(head(data_merged, 5), 'data_merged.csv', row.names = FALSE)
```

Aplico la codificación One-Hot a la variable DRUGRES_CAT y el resultado son dos nuevas columnas: DRUGRES_CAT_Negativo o No Probado y DRUGRES_CAT_Positivo. Cada columna representa una de las categorías de la variable original y tiene valores binarios (0 o 1) indicando la ausencia o presencia de esa categoría específica. Realizo el mismo procedimiento con las columnas: LGT_COND_CAT y WEATHER_CAT

4. Reducción de Dimensionalidad

Análisis de Componentes Principales (PCA): Si se incluyen muchas variables, aplicar PCA u otras técnicas para reducir la dimensionalidad, centrando el análisis en las componentes que capturan la mayor parte de la varianza.

```
# columnas que quiero mantener
columnas_deseadas <- c("FATALS", "DRUNK_DR", "DRUGRES_CAT_Desconocido", "DRUGRES_CAT_Negativo o No Proba
                       "LGT_COND_CAT_Crepúsculo o Amanecer", "LGT_COND_CAT_Luz Diurna", "LGT_COND_CAT_Os
                       "WEATHER_CAT_Claro", "WEATHER_CAT_Extremas", "WEATHER_CAT_Otras", "WEATHER_CAT_Pr

# Seleccionando solo las columnas deseadas
data_para_analisis <- select(data_merged, all_of(columnas_deseadas))

# Verificar los primeros registros del conjunto de datos resultante
#head(data_para_analisis)
```

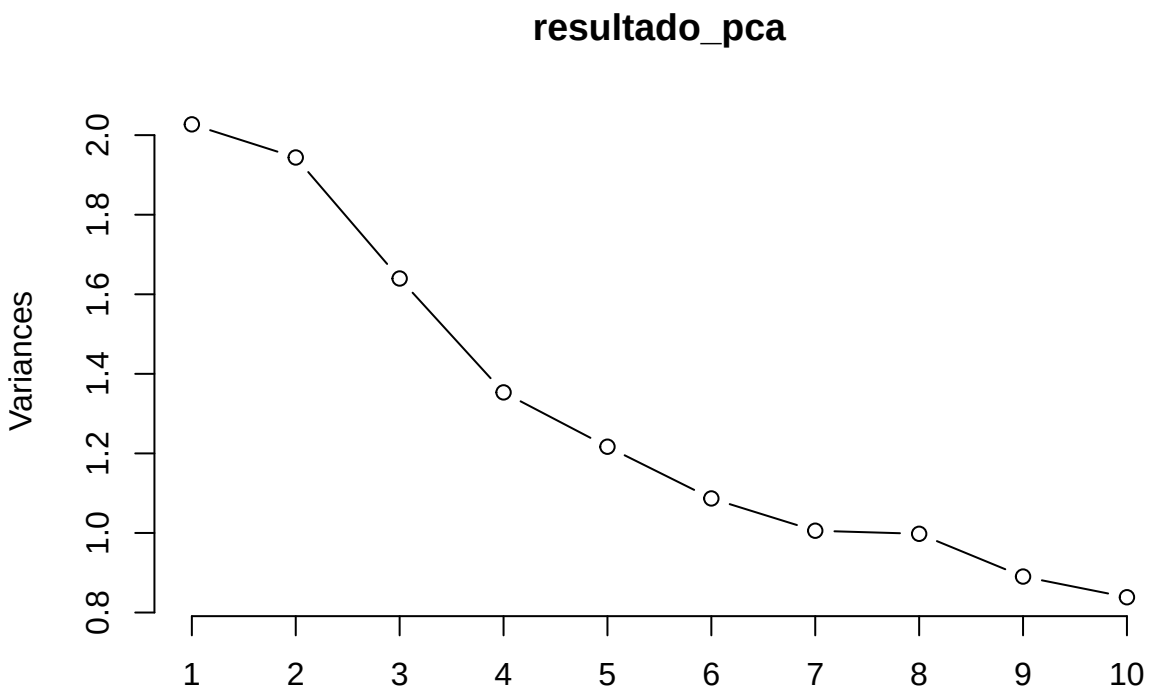
columnas_deseadas es un vector que contiene los nombres de todas las columnas que voy a mantener en mi conjunto de datos final.

Procedo a aplicar PCA

```
# Aplicar PCA
resultado_pca <- prcomp(data_para_analisis, center = TRUE, scale. = TRUE)

# Imprimir un resumen de los resultados del PCA
summary(resultado_pca)
```

```
## Importance of components:
##                           PC1    PC2    PC3    PC4    PC5    PC6     PC7
## Standard deviation     1.4238 1.3942 1.2805 1.1633 1.1031 1.0425 1.00281
## Proportion of Variance 0.1559 0.1495 0.1261 0.1041 0.0936 0.0836 0.07736
## Cumulative Proportion  0.1559 0.3055 0.4316 0.5357 0.6293 0.7129 0.79025
##                           PC8     PC9   PC10      PC11      PC12      PC13
## Standard deviation     0.99896 0.94363 0.9157 1.803e-13 7.082e-14 4.705e-14
## Proportion of Variance 0.07676 0.06849 0.0645 0.000e+00 0.000e+00 0.000e+00
## Cumulative Proportion  0.86701 0.93550 1.0000 1.000e+00 1.000e+00 1.000e+00
```

```
# También puedes visualizar las contribuciones de las variables originales a los componentes principales
plot(resultado_pca, type = "lines")
```



El gráfico muestra la varianza explicada por cada componente principal en PCA. El scree plot es una herramienta visual utilizada para determinar el número de componentes principales a retener. La regla del codo sugiere que debería retener los componentes hasta el punto donde la explicación de la varianza se aplana y deja de caer abruptamente, lo que parece ocurrir alrededor del PC4 o PC5 en el gráfico.

La tabla de importancia de los componentes muestra las desviaciones estándar, la proporción de la varianza explicada por cada componente principal (PC), y la proporción acumulada de la varianza. Se puede ver que los primeros diez componentes explican la totalidad de la varianza en los datos, lo cual es típico cuando el número de componentes seleccionados es igual al número de variables originales. Los últimos tres componentes tienen desviaciones estándar extremadamente pequeñas y no contribuyen a la varianza explicada, lo que indica que no son necesarios.

Basándome en la tabla y el scree plot, parece que podría considerar usar los primeros cuatro o cinco componentes para un análisis posterior, ya que juntos explican más del 50% de la varianza. Esto se alinea con la regla del codo observada en el scree plot.

5. Minería de Datos

Elección de Modelos: Seleccionar modelos adecuados para el objetivo, por ejemplo, modelos de clasificación para predecir la severidad de los accidentes. Validación Cruzada y Ajuste de Parámetros: Aplicar técnicas de validación cruzada para evaluar la robustez de los modelos y ajustar parámetros según sea necesario.

```
# install.packages(c("caret", "randomForest"))
nrow(data_merged)
```

```
## [1] 35766
```

```r
#print(data_merged)
```

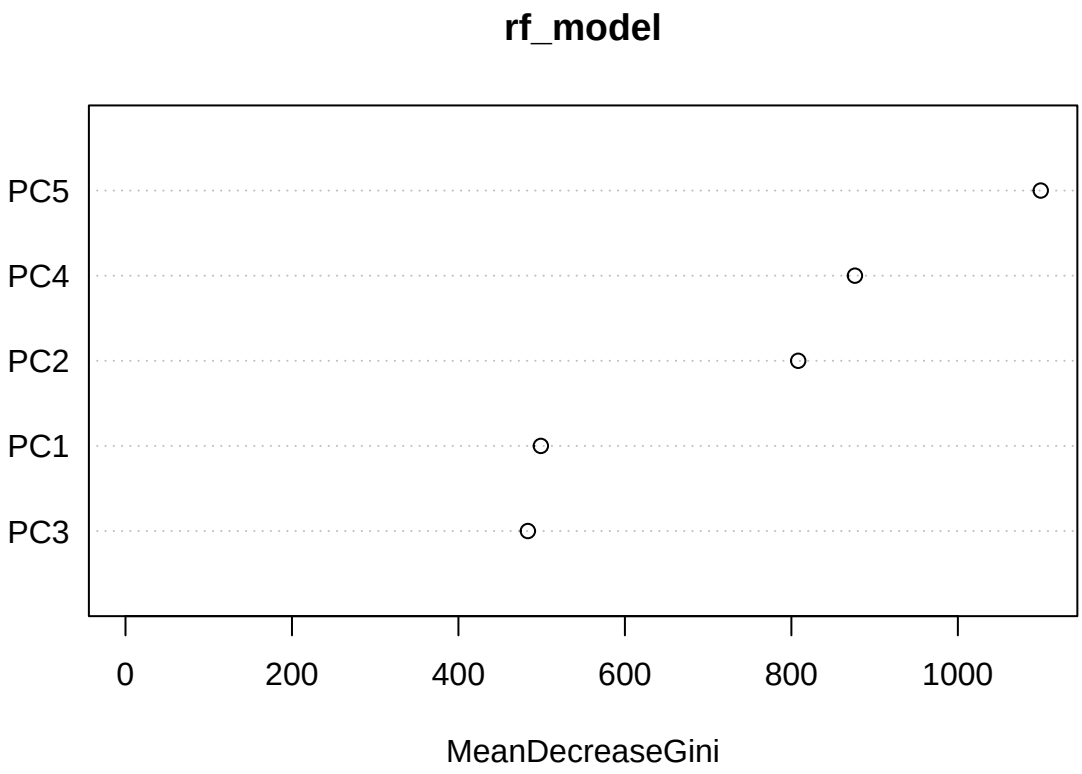
```r
library(caret)
```

```
## Loading required package: lattice
```

```r
library(randomForest)
```

```
## randomForest 4.7-1.1

## 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
```

```r
# Asumiendo resultado_pca es el resultado de tu PCA
# Extraer los componentes principales para usar como características
# N componentes principales basados en análisis anterior
N <- 5
X_pca <- as.data.frame(resultado_pca$x[, 1:N])

# Asegúrate de que la variable objetivo y las características estén alineadas
y <- as.factor(ifelse(data_merged$FATALS > 1, 1, 0))

# Dividir el conjunto de datos en entrenamiento y prueba
set.seed(42) # Para reproducibilidad
trainIndex <- createDataPartition(y, p = .8, list = FALSE)
X_train <- X_pca[trainIndex, ]
X_test <- X_pca[-trainIndex, ]
y_train <- y[trainIndex]
y_test <- y[-trainIndex]

# Convertir y_train y y_test a factores (si aún no lo son)
y_train <- as.factor(y_train)
y_test <- as.factor(y_test)

# Construir y entrenar el modelo de Bosque Aleatorio
rf_model <- randomForest(x = X_train, y = y_train, ntree = 100, mtry = 2)

# Predecir y evaluar el modelo
y_pred <- predict(rf_model, newdata = X_test)
accuracy <- mean(y_pred == y_test)

# Importancia de las características (basado en los componentes principales, no directamente interpretal
feature_importances <- varImpPlot(rf_model)
```




```
# Mostrar resultados
cat("Precisión del modelo:", accuracy, "\n")
```

```
## Precisión del modelo: 0.9987418
```

Bosque Aleatorio tiene una precisión extremadamente alta, ¡casi del 99.9%! Esto es impresionante, pero también puede ser una señal de alerta, ya que tal nivel de precisión es raro en aplicaciones del mundo real y podría indicar un sobreajuste. En otras palabras, el modelo podría estar demasiado ajustado a los datos de entrenamiento y por lo tanto no generalizaría bien a datos nuevos o no vistos. Aquí hay algunas consideraciones a tomar en cuenta:

Revisa el Equilibrio de Clases: Si una clase es mucho más frecuente que la otra, la alta precisión podría simplemente reflejar el desequilibrio de clases. Validación Cruzada: Para asegurarte de que el modelo es robusto y generalizable, realiza una validación cruzada y verifica las métricas de desempeño en cada iteración. Conjunto de Pruebas Independiente: Es vital tener un conjunto de datos de prueba que no haya sido visto por el modelo durante el entrenamiento para evaluar el rendimiento del modelo de manera objetiva. La gráfica generada muestra la importancia de cada componente principal basada en el índice Gini. Un componente con un alto MeanDecreaseGini es más importante para las predicciones del modelo. Sin embargo, dado que estos son componentes principales y no las variables originales, esta importancia no se traduce directamente a la interpretabilidad de las características originales. Esto es algo inherente al uso de PCA: aunque simplifica y posiblemente mejora el rendimiento del modelo, reduce la interpretabilidad directa.

Si la alta precisión se mantiene consistente a través de varias evaluaciones y es genuina, eso es excelente. Pero siempre es prudente estar vigilantes contra el sobreajuste y asegurarse de que los resultados sean válidos y aplicables en el contexto de los datos.

```r
library(caret)
library(randomForest)

# Asumimos que resultado_pca es el resultado de PCA
N <- 5
# Creamos un dataframe con los componentes principales y la variable objetivo
data_pca <- data.frame(Severity = as.factor(ifelse(data_merged$FATALS > 1, 1, 0)),
                       PCA_Features = resultado_pca$x[, 1:N])

# Convertimos los nombres de los componentes principales a un vector de nombres de columnas
predictors_pca <- paste0("PCA_Features.PC", 1:N)

# Configuración de control de entrenamiento para validación cruzada
train_control <- trainControl(method="cv", number=10)

# Entrenar el modelo con validación cruzada usando los componentes principales
rf_grid <- expand.grid(mtry=2:4) # Grid de hiperparámetros a ajustar, ajusta según necesidad

set.seed(42)
rf_model_cv <- train(
  Severity ~ .,
  data = data_pca,
  method = "rf",
  trControl = train_control,
  tuneGrid = rf_grid,
  ntree = 100
)

# Revisar los resultados
print(rf_model_cv)
```

Validacion cruzada

```
## Random Forest
##
## 35766 samples
##     5 predictor
##     2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 32190, 32189, 32190, 32189, 32189, 32190, ...
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##   2     0.9995247  0.9963826
##   3     0.9994128  0.9955304
##   4     0.9993849  0.9953196
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 2.
```

Corregir las clases: En situaciones con un desequilibrio de clases tan pronunciado, la precisión no es la mejor métrica para evaluar el rendimiento del modelo, ya que puede ser engañosamente alta. En cambio, deberías considerar las siguientes métricas:

Recall/Sensitivity (Sensibilidad): La proporción de positivos reales que se identificaron correctamente. Esto es crucial en este caso porque se quiere saber cuántos de los accidentes con más de 1 muerte el modelo puede identificar correctamente.

Precision (Precisión): La proporción de predicciones positivas que son correctas. En este caso, sería el número de veces que el modelo predice más de 1 muerte y en realidad es correcto.

F1 Score: Combinar la precisión y el recall en una métrica. Es útil cuando las distribuciones de clase son desequilibradas.

ROC-AUC: El área bajo la curva del Receptor Operativo Característico (ROC) es una métrica de rendimiento para clasificadores binarios que toma en cuenta tanto la tasa de verdaderos positivos como la de falsos positivos.

Confusion Matrix (Matriz de Confusión): Una tabla que te permite visualizar el rendimiento de un algoritmo de clasificación. Es especialmente útil cuando las clases están desbalanceadas.

Para abordar el desequilibrio de clases,se podría considerar las siguientes estrategias:

Oversampling de la clase minoritaria: Se puede aumentar artificialmente la frecuencia de la clase minoritaria (registros con más de 1 muerte).

Undersampling de la clase mayoritaria: Se puede reducir la frecuencia de la clase mayoritaria (registros con 1 muerte).

Synthetic Minority Over-sampling Technique (SMOTE): Esta es una técnica de oversampling que crea ejemplos sintéticos de la clase minoritaria a partir de ejemplos reales.

Ajustar los pesos de clase en el modelo: Algunos modelos permiten asignar un mayor peso a la clase minoritaria durante el entrenamiento para que el modelo preste más atención a esos casos.