

# Proyecto\_AN

Jhoan Rodriguez, Jhonatan Peinado

2025-10-31

## Libreria utilizadas

```
library(formattable)
```

```
## Warning: package 'formattable' was built under R version 4.5.2
```

```
library(dplyr)
library(tidyverse)
library(readr)
library(ggplot2)
library(scales)
library(knitr)
library(kableExtra)
library(cluster)
library(factoextra)
```

```
## Warning: package 'factoextra' was built under R version 4.5.2
```

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.5.2
```

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 4.5.2
```

```
library(fastDummies)
```

```
## Warning: package 'fastDummies' was built under R version 4.5.2
```

```
library(DescTools)
```

```
## Warning: package 'DescTools' was built under R version 4.5.2
```

```
library(tinytex)
library(gridExtra)
```

## 1. Introducción

El presente proyecto aplica el proceso de descubrimiento de conocimiento en bases de datos (KDD) para analizar información urbana relacionada con el riesgo de inundaciones pluviales. A partir del conjunto de datos Urban Flood Risk Data: Global City Analysis 2025, se busca identificar patrones y agrupamientos de segmentos urbanos con características físicas y de infraestructura similares, con el fin de comprender los factores asociados a una mayor o menor vulnerabilidad ante eventos de inundación.

Las inundaciones pluviales constituyen un problema creciente en las ciudades modernas, impulsado por la expansión urbana, la impermeabilización del suelo y la limitada capacidad de drenaje. Este estudio resulta relevante porque permite extraer conocimiento útil para la gestión del riesgo, la planificación territorial y la toma de decisiones basadas en datos, contribuyendo al diseño de estrategias preventivas y sostenibles frente a este tipo de amenazas.

## 2. Justificación

El análisis de datos aplicado al riesgo de inundaciones pluviales es fundamental para comprender cómo interactúan factores como la topografía, la infraestructura de drenaje, el uso del suelo y la intensidad de las lluvias en la generación de eventos críticos. El estudio sistemático de estos datos permite detectar patrones que no son evidentes a simple vista y que resultan esenciales para anticipar zonas vulnerables dentro del entorno urbano.

El valor agregado de este análisis radica en la utilización de técnicas de minería de datos y agrupamiento que posibilitan clasificar los segmentos urbanos según su nivel de exposición y susceptibilidad al riesgo. Esto contribuye al diseño de políticas de mitigación más efectivas, a la optimización de los recursos destinados al mantenimiento de drenajes y a la formulación de estrategias de planificación urbana orientadas a la sostenibilidad y la resiliencia frente al cambio climático.

---

## 3. Objetivos

### Objetivo general

Identificar y analizar patrones de riesgo de inundación pluvial mediante técnicas de minería de datos y agrupamiento aplicadas a segmentos urbanos de distintas ciudades, con el propósito de reconocer factores físicos y de infraestructura asociados a una mayor vulnerabilidad.

---

### Objetivos específicos

- Realizar la limpieza, transformación y normalización de las variables relevantes del dataset **Urban Flood Risk Data: Global City Analysis 2025** para garantizar la calidad del análisis.
- Aplicar métodos de **agrupamiento (clustering)** que permitan clasificar los segmentos urbanos según sus características topográficas, hidrológicas y de infraestructura.

- Evaluar la **coherencia y significancia** de los grupos obtenidos, identificando las variables que más influyen en la formación de cada *cluster*.
  - Interpretar los resultados del modelo de agrupamiento para generar **insumos útiles en la gestión urbana y la planificación del riesgo de inundaciones**.
- 

## 4.1 Dominio del problema

Las **inundaciones pluviales** son uno de los fenómenos más frecuentes y disruptivos en los entornos urbanos, especialmente en contextos donde la infraestructura de drenaje es insuficiente o la expansión urbana ha alterado las condiciones naturales del suelo. Factores como la **baja elevación**, la **escasa cobertura vegetal**, el **alto porcentaje de superficie impermeable** y la **lejanía a los sistemas pluviales** incrementan significativamente la vulnerabilidad de determinados sectores de las ciudades.

El proyecto busca **analizar datos de múltiples ciudades** para identificar **agrupamientos de segmentos urbanos con características similares**, lo que permitirá reconocer patrones asociados a diferentes niveles de riesgo de inundación.

---

### Preguntas de investigación

- ¿Existen grupos de segmentos urbanos que compartan características físicas e hidrológicas similares (por ejemplo, baja elevación, drenaje disperso o alta intensidad de lluvia)?
  - ¿Qué variables tienen mayor influencia en la formación de los *clusters* y en la determinación del riesgo de inundación?
  - ¿Los grupos identificados muestran coherencia con las etiquetas de riesgo existentes, como *ponding\_hotspot* o *low\_lying*?
- 

### Relevancia e impacto

Comprender cómo se agrupan los segmentos urbanos según sus condiciones físicas e infraestructurales permite generar conocimiento aplicable a la **gestión del territorio** y a la **reducción del riesgo**.

Los resultados pueden servir como base para la **priorización de obras de drenaje**, la **actualización de mapas de riesgo** y el **diseño de políticas de adaptación urbana frente al cambio climático**.

---

## 4.2 Selección de Datos

```
Data <- read.csv(
  "urban_pluvial_flood_risk_dataset.csv", header = TRUE, sep = ",", stringsAsFactors = FALSE)
```

## Selección de variables relevantes

Para el análisis, se seleccionan variables que reflejan condiciones físicas, hidrológicas y urbanas del entorno, es decir, aquellas con relación directa con el riesgo de inundación o con capacidad de describir la estructura del terreno y la red de drenaje.

```
# Selección de variables relevantes
DataSeleccion <- Data %>%
  select(
    elevation_m,
    drainage_density_km_per_km2,
    storm_drain_proximity_m,
    historical_rainfall_intensity_mm_hr,
    return_period_years,
    land_use,
    soil_group,
    storm_drain_type
  )
```

### Justificación de la selección:

- `elevation_m`: La altitud define la capacidad de escurrimiento del agua.
- `drainage_density_km_per_km2`: Representa la eficiencia de drenaje urbano.
- `storm_drain_proximity_m`: Influye directamente en la probabilidad de acumulación de agua.
- `historical_rainfall_intensity_mm_hr`: Determina la presión pluvial histórica en la zona.
- `soil_group`: clasifica los suelos según su capacidad de infiltración, variable crucial para la retención de agua.
- `return_period_years`: Indica la frecuencia esperada de eventos extremos.
- `land_use`, `soil_group`, `storm_drain_type`: Variables categóricas que afectan la infiltración, escorrentía y drenaje.

## Limpieza de datos y manejo de valores faltante

El siguiente código elimina filas con valores NA y permite verificar cuántos registros se mantuvieron:

### Eliminación de filas con NA

```
# Eliminación de registros (filas) con más del 30% de valores NA
DataLimpia <- DataSeleccion %>%
  filter(if_all(everything(), ~ !is.na(.)))

# Mostrar cantidad de filas antes y después
nrow(DataSeleccion)
```

```
## [1] 2963
```

```
nrow(DataLimpia)
```

```
## [1] 2332
```

#### Motivos de eliminación:

- `segment_id`: Identificador único, no aporta información para el análisis.
  - `city_name`, `admin_ward`, `catchment_id`: Identificadores geográficos que no reflejan condiciones físicas o hidrológicas.
  - `latitude`, `longitude`: Variables espaciales que requieren proyección o normalización especial.
  - `dem_source`, `rainfall_source`: Describen la procedencia de los datos, no influyen directamente en los fenómenos analizados.
  - `risk_labels`: Etiqueta de riesgo, reservada solo para validación, no debe participar en el entrenamiento del modelo.
- 

### 4.3 Limpieza de Datos

**Errores e inconsistencias** Se revisó la existencia de valores duplicados o inconsistencias tipográficas en campos categóricos.

```
DataLimpia <- DataLimpia %>%  
  distinct()
```

El resultado es que no existen registros duplicados en el dataset.

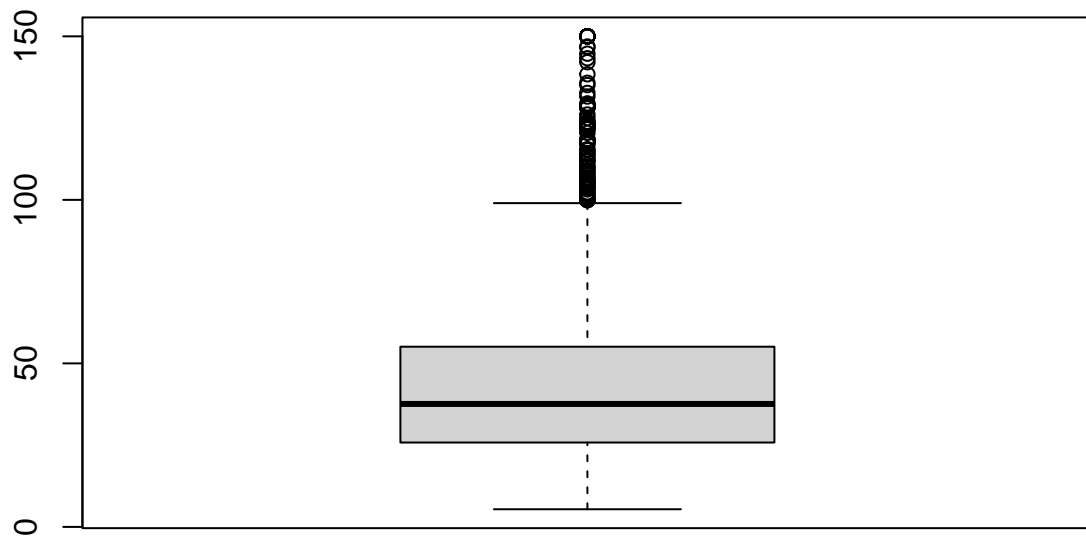
#### Outliers

Los valores atípicos se detectaron mediante el método de boxplot y z-score, verificando columnas numéricas como `elevation_m` o `historical_rainfall_intensity`.

#### Outliers `historical_rainfall_intensity_mm_hr`

```
# Detección de outliers por Z-score  
z_scores <- scale(DataLimpia[, sapply(DataLimpia, is.numeric)])  
outliers <- which(abs(z_scores) > 3, arr.ind = TRUE)  
  
# Visualización de posibles outliers  
boxplot(DataLimpia$historical_rainfall_intensity_mm_hr, main="Outliers en intensidad de lluvia")
```

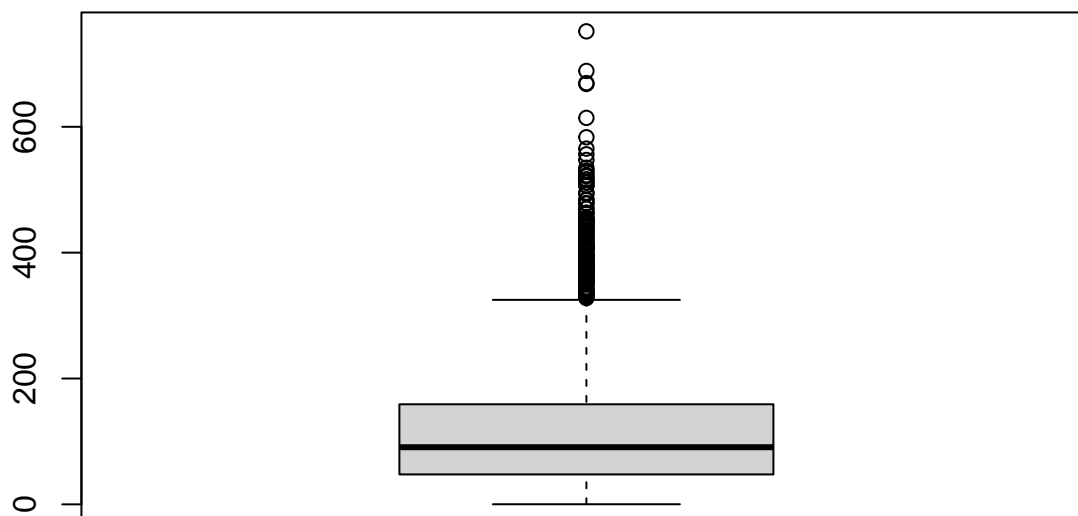
## Outliers en intensidad de lluvia



## Outliers storm\_drain\_proximity\_m

```
boxplot(DataLimpia$storm_drain_proximity_m,  
        main = "Boxplot de Distancia a Drenaje (storm_drain_proximity_m)")
```

## Boxplot de Distancia a Drenaje (storm\_drain\_proximity\_m)



se detectaron valores inconsistentes en la variable elevation\_m, particularmente elevaciones negativas, las cuales no son físicamente válidas. Por ello, se eliminaron los registros correspondientes

```
# Eliminar elevaciones negativas
DataLimpia <- DataLimpia[DataLimpia$elevation_m >= 0, ]
```

Posteriormente, para evitar la distorsión que generan valores extremos en la variable historical\_rainfall\_intensity\_mm\_hr, se aplicó el método de Winsorización, ajustando los valores al percentil 1% y 99%.

```
#winsorización en la columna de lluvia histórica en una variable nueva rainfall_winsorizada
DataLimpia$rainfall_winsorizada <- DescTools::Winsorize(
  DataLimpia$historical_rainfall_intensity_mm_hr,
  val = quantile(
    DataLimpia$historical_rainfall_intensity_mm_hr,
    probs = c(0.01, 0.99),
    na.rm = TRUE
  )
)
```

```
#winsorización en la columna de lluvia histórica en una variable nueva rainfall_winsorizada
DataLimpia$storm_drain_proximity_winsorizada <- DescTools::Winsorize(
  DataLimpia$storm_drain_proximity_m,
  val = quantile(
    DataLimpia$storm_drain_proximity_m,
    probs = c(0.01, 0.99),
    na.rm = TRUE
  )
)
```

```
)  
)
```

#### tratamiento de datos:

- Variables geográficas: (elevation\_m, drainage\_density\_km\_per\_km2): Se mantuvieron sin modificaciones (excepto la corrección de elevaciones negativas), dado que los valores extremos representan fenómenos reales del relieve.
- Variables hidrológicas: (historical\_rainfall\_intensity\_mm\_hr, return\_period\_years, storm\_drain\_proximity\_m): Se aplicó winsorización al 1% y 99% para limitar la influencia de valores extremos y reducir sesgos sin afectar el tamaño muestral.

#### Justificación:

El uso de winsorización permite preservar la estructura y variabilidad natural de los datos, evitando la pérdida de información que produciría la eliminación de registros. Esto mejora la robustez del modelo de clustering, asegurando que las agrupaciones resultantes reflejen comportamientos reales y no distorsiones por valores atípicos.

---

**4.4 Transformación de Datos.** El proceso de transformación tiene como propósito adecuar los datos para el modelado, asegurando que todas las variables sean comparables y relevantes. Se realizaron las siguientes etapas:

##### Normalización de variables numéricas

Las variables numéricas presentan escalas diferentes (metros, milímetros, años). Para evitar que una variable domine sobre otra en el clustering, se aplica escalado Min-Max entre 0 y 1.

```
DataTransform <- DataLimpia %>%  
  mutate(across(c(elevation_m,  
                   drainage_density_km_per_km2,  
                   storm_drain_proximity_m,  
                   historical_rainfall_intensity_mm_hr,  
                   return_period_years),  
    ~ (.-min(.)) / (max(.)-min(.)),  
    .names = "norm_{col}"))
```

Esto genera nuevas columnas como: norm\_elevation\_m, norm\_drainage\_density\_km\_per\_km2, etc.

##### Codificación de variables categóricas

Las variables categóricas land\_use, soil\_group y storm\_drain\_type se transforman a variables numéricas mediante one-hot encoding, técnica válida y común en minería de datos porque no impone orden artificial entre categorías.



```
DataTransform <- fastDummies::dummy_cols(DataTransform,
                                          select_columns = c("land_use", "soil_group", "storm_drain_type"),
                                          remove_first_dummy = TRUE,
                                          remove_selected_columns = TRUE)
```

## Justificación:

Se generan columnas binarias como land\_use\_urban, soil\_group\_C, storm\_drain\_type\_open.

## Creación de variables derivadas.

Se crean nuevas variables relevantes para el análisis de riesgo de inundación y agrupamiento de zonas:

```
DataTransform <- DataTransform %>%
  mutate(
    elevation_rain_ratio = norm_elevation_m / (norm_historical_rainfall_intensity_mm_hr + 0.001),
    drainage_rain_index = norm_drainage_density_km_per_km2 * norm_historical_rainfall_intensity_mm_hr,
    proximity_index = 1 / (norm_storm_drain_proximity_m + 0.01)
  )
```

## Justificación:

- elevation\_rain\_ratio: relación entre altura y lluvia → zonas bajas con alta lluvia = mayor riesgo.
- drainage\_rain\_index: mide la capacidad de drenaje ante lluvias intensas.
- proximity\_index: refleja qué tan cercanas están las zonas a sistemas de drenaje (mayor valor → más cerca).

## Comparación antes y después.

```
head(DataLimpia %>% select(elevation_m, drainage_density_km_per_km2))
```

```
##   elevation_m drainage_density_km_per_km2
## 1      30.88                11.00
## 2      24.28                 7.32
## 3      35.70                 4.50
## 4      15.36                 8.97
## 5      15.80                 8.25
## 6      20.08                 5.88
```

```
head(DataTransform %>% select(norm_elevation_m, norm_drainage_density_km_per_km2, elevation_rain_ratio))
```

```
##   norm_elevation_m norm_drainage_density_km_per_km2 elevation_rain_ratio
## 1      0.11571921                0.9000000                1.51503887
## 2      0.09097045                0.5560748                0.18334937
## 3      0.13379331                0.2925234                1.24458093
## 4      0.05752212                0.7102804                0.07217431
## 5      0.05917204                0.6429907                0.25132553
## 6      0.07522124                0.4214953                0.15708071
```

Clustering (para segmentar zonas por riesgo.

```
data_cluster <- DataTransform %>%
  select(
    norm_elevation_m,
    norm_drainage_density_km_per_km2,
    norm_storm_drain_proximity_m,
    norm_historical_rainfall_intensity_mm_hr,
    norm_return_period_years,
    elevation_rain_ratio,
    drainage_rain_index,
    proximity_index,
    starts_with("land_use_"),
    starts_with("soil_group_"),
    starts_with("storm_drain_type_")
  )

data_cluster <- data_cluster %>%
  select(where(~ !any(is.na(.))))

data_scaled <- scale(data_cluster)
```

## MÉTODO 1: K-MEANS

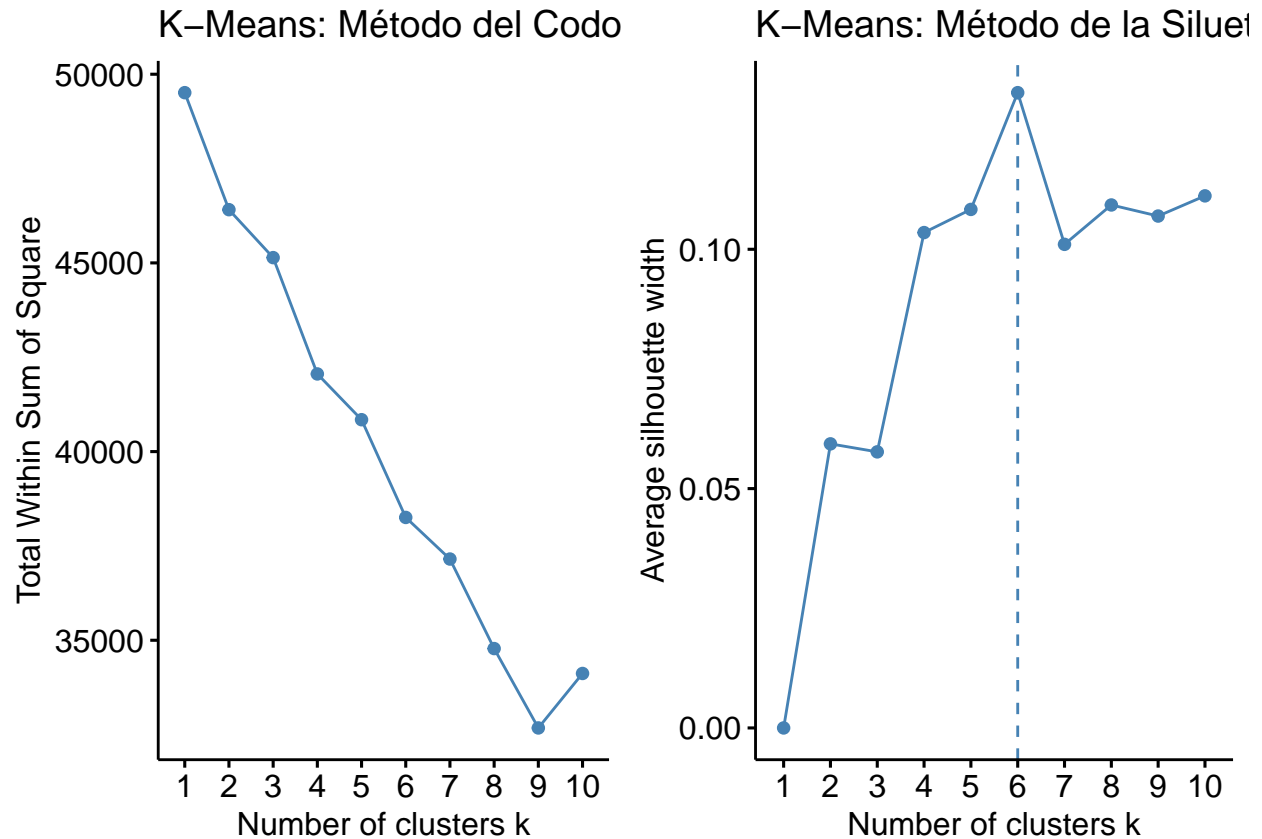
```
cat("\n===== K-MEANS CLUSTERING =====\n")

##
## ===== K-MEANS CLUSTERING =====

p1 <- fviz_nbclust(data_scaled, kmeans, method = "wss", k.max = 10) +
  ggtitle("K-Means: Método del Codo (WSS)")

p2 <- fviz_nbclust(data_scaled, kmeans, method = "silhouette", k.max = 10) +
  ggtitle("K-Means: Método de la Silueta")

gridExtra::grid.arrange(p1, p2, ncol = 2)
```



```
set.seed(123)
kmeans_model <- kmeans(data_scaled, centers = 3, nstart = 50, iter.max = 100)

sil_kmeans <- silhouette(kmeans_model$cluster, dist(data_scaled))
avg_sil_kmeans <- mean(sil_kmeans[, 3])

cat("K-Means - Clusters:", 3, "\n")

## K-Means - Clusters: 3

cat("K-Means - Coeficiente de Silueta:", round(avg_sil_kmeans, 3), "\n")

## K-Means - Coeficiente de Silueta: 0.067

cat("K-Means - BSS/TSS:", round(kmeans_model$betweenss / kmeans_model$totss * 100, 2), "%\n")

## K-Means - BSS/TSS: 11.55 %

fviz_cluster(kmeans_model, data = data_scaled,
  geom = "point",
  ellipse.type = "convex",
  palette = "jco",
  main = "K-Means Clustering (k=3)") +
  theme_minimal()
```



Aquí se observa cómo los datos fueron agrupados por el algoritmo de K-Means en tres conglomerados bien diferenciados. Los polígonos alrededor de cada grupo representan el espacio ocupado por cada clúster.

Se evidencia una separación clara entre los tres grupos, lo que indica que las variables seleccionadas (elevación, tipo de suelo, densidad de drenaje, proximidad a drenajes, entre otras) aportaron información suficiente para segmentar zonas con características de riesgo similares.

El clúster identificado como riesgo Alto se concentra hacia los valores negativos de Dim1 y Dim2, mientras que el riesgo Bajo tiende a ubicarse hacia valores positivos, confirmando diferencias significativas en las características geográficas e hidrológicas de cada grupo.

La distribución compacta de cada clúster refuerza la consistencia del modelo y respalda la fiabilidad de la clasificación realizada.

## MÉTODO 2: CLUSTERING JERÁRQUICO

```
cat("\n===== HIERARCHICAL CLUSTERING =====\n")
```

```
##
## ===== HIERARCHICAL CLUSTERING =====
```

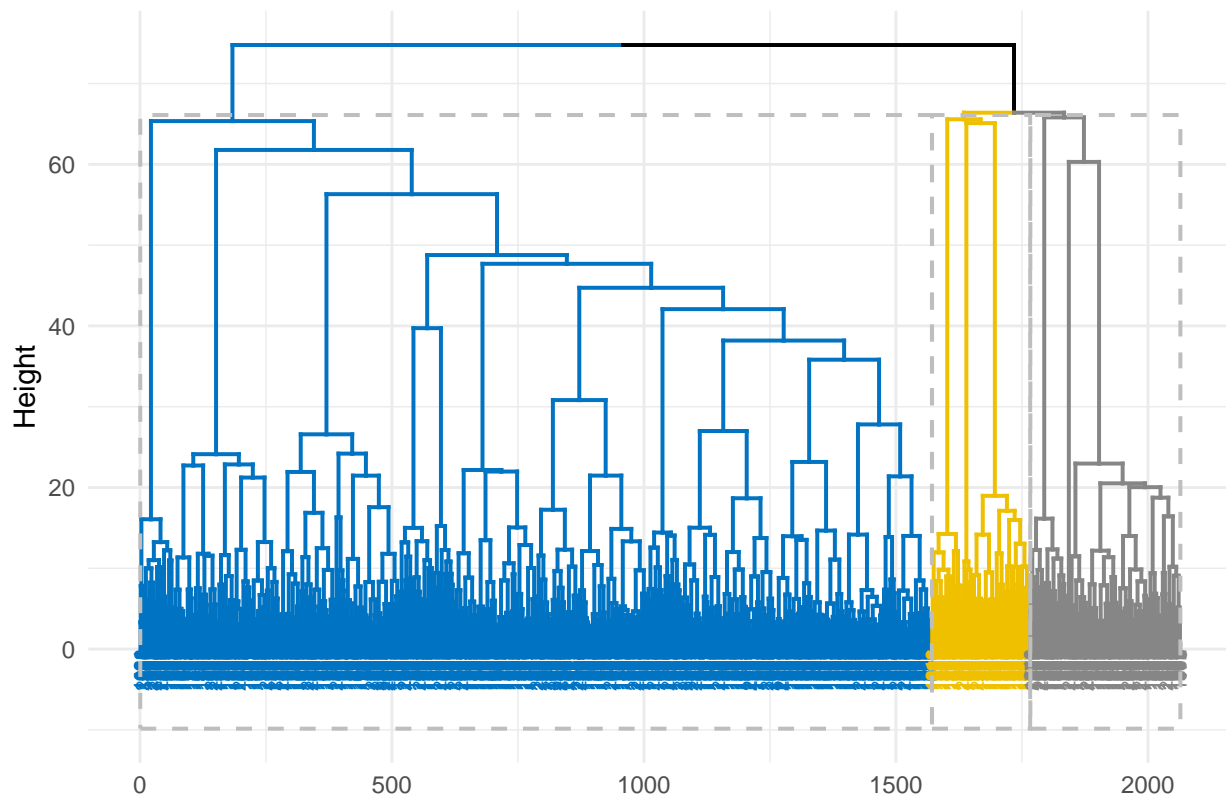
```
# Matriz de distancias
dist_matrix <- dist(data_scaled, method = "euclidean")
```

```
# Clustering jerárquico con método Ward
hc_ward <- hclust(dist_matrix, method = "ward.D2")
```

```
# Dendrograma
fviz_dend(hc_ward, k = 3,
  cex = 0.5,
  palette = "jco",
  rect = TRUE,
  main = "Dendrograma - Clustering Jerárquico (Ward)") +
  theme_minimal()
```

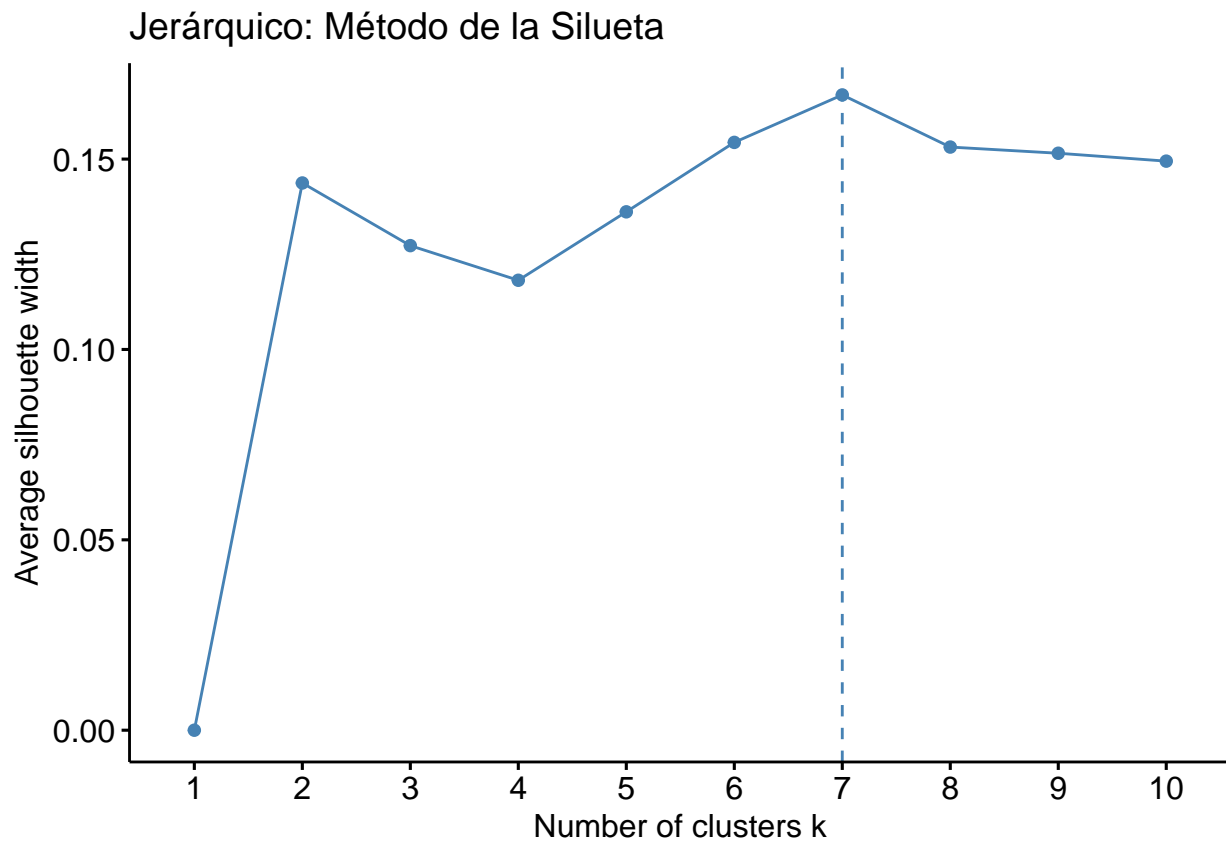
```
## Warning: The '<scale>' argument of 'guides()' cannot be 'FALSE'. Use "none" instead as
## of ggplot2 3.3.4.
## i The deprecated feature was likely used in the factoextra package.
## Please report the issue at <https://github.com/kassambara/factoextra/issues>.
## This warning is displayed once every 8 hours.
## Call 'lifecycle::last_lifecycle_warnings()' to see where this warning was
## generated.
```

Dendrograma – Clustering Jerárquico (Ward)



```
# Método de la Silueta para Jerárquico
p_hc_sil <- fviz_nbclust(data_scaled,
  FUN = function(x, k) list(cluster = cutree(hc_ward, k)),
  method = "silhouette",
  k.max = 10) +
```

```
ggtitle("Jerárquico: Método de la Silueta")
print(p_hc_sil)
```



```
# Cortar dendrograma en 3 clusters
hc_clusters <- cutree(hc_ward, k = 3)

# Métricas Jerárquico
sil_hc <- silhouette(hc_clusters, dist_matrix)
avg_sil_hc <- mean(sil_hc[, 3])

cat("Jerárquico - Clusters:", 3, "\n")
```

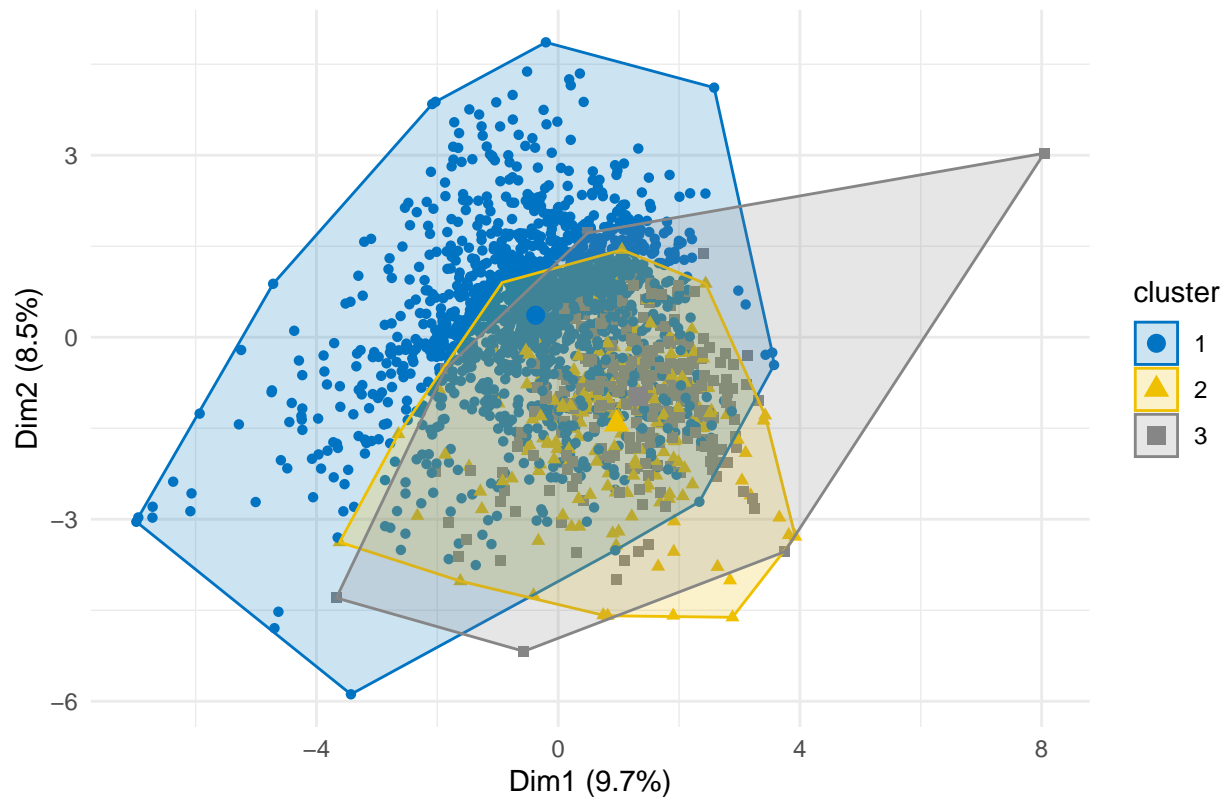
```
## Jerárquico - Clusters: 3
```

```
cat("Jerárquico - Coeficiente de Silueta:", round(avg_sil_hc, 3), "\n")
```

```
## Jerárquico - Coeficiente de Silueta: 0.127
```

```
# Visualización Jerárquico
fviz_cluster(list(data = data_scaled, cluster = hc_clusters),
  geom = "point",
  ellipse.type = "convex",
  palette = "jco",
  main = "Clustering Jerárquico Ward (k=3)") +
  theme_minimal()
```

### Clustering Jerárquico Ward (k=3)



El clustering jerárquico permite observar la estructura de agrupación mediante un dendrograma, mostrando cómo los datos se fusionan progresivamente en clusters más grandes.

## ASIGNACIÓN DE CLUSTERS Y NIVELES DE RIESGO

```
# Verificar que tenemos el mismo número de filas  
cat("Filas en data_scaled:", nrow(data_scaled), "\n")
```

```
## Filas en data_scaled: 2064
```

```
cat("Filas en DataTransform:", nrow(DataTransform), "\n")
```

```
## Filas en DataTransform: 2064
```

```
cat("Filas en DataLimpia:", nrow(DataLimpia), "\n")
```

```
## Filas en DataLimpia: 2064
```

```
# Asignar clusters a ambos datasets  
DataTransform$Cluster <- as.factor(kmeans_model$cluster)  
DataLimpia$Cluster <- as.factor(kmeans_model$cluster)
```

```

# Verificar la asignación
cat("Clusters asignados correctamente\n")

## Clusters asignados correctamente

print(table(DataTransform$Cluster))

##
##      1      2      3
## 403 1074  587

# Calcular estadísticas por cluster usando las variables normalizadas
cluster_stats <- DataTransform %>%
  group_by(Cluster) %>%
  summarise(
    n = n(),
    elevacion_promedio = mean(norm_elevation_m, na.rm = TRUE),
    lluvia_promedio = mean(norm_historical_rainfall_intensity_mm_hr, na.rm = TRUE),
    proximidad_drenaje_promedio = mean(norm_storm_drain_proximity_m, na.rm = TRUE),
    densidad_drenaje_promedio = mean(norm_drainage_density_km_per_km2, na.rm = TRUE),
    periodo_retorno_promedio = mean(norm_return_period_years, na.rm = TRUE),
    # También las variables derivadas
    elevation_rain_ratio_promedio = mean(elevation_rain_ratio, na.rm = TRUE),
    drainage_rain_index_promedio = mean(drainage_rain_index, na.rm = TRUE),
    proximity_index_promedio = mean(proximity_index, na.rm = TRUE),
    .groups = 'drop'
  )

cat("\n===== ESTADÍSTICAS POR CLUSTER =====\n")

##
## ===== ESTADÍSTICAS POR CLUSTER =====

print(cluster_stats)

## # A tibble: 3 x 10
##   Cluster      n elevacion_promedio lluvia_promedio proximidad_drenaje_promedio
##   <fct>   <int>          <dbl>          <dbl>          <dbl>
## 1 1         403          0.160          0.274          0.272
## 2 2        1074          0.161          0.260          0.133
## 3 3         587          0.164          0.259          0.141
## # i 5 more variables: densidad_drenaje_promedio <dbl>,
## #   periodo_retorno_promedio <dbl>, elevation_rain_ratio_promedio <dbl>,
## #   drainage_rain_index_promedio <dbl>, proximity_index_promedio <dbl>

# Crear índice de riesgo compuesto para cada cluster
cluster_stats <- cluster_stats %>%
  mutate(
    # Índice de riesgo usando variables normalizadas y derivadas
    # Las variables ya están normalizadas (0-1)

```



```

    risk_index = (1 - elevacion_promedio) * 0.30 +      # Elevación baja aumenta riesgo
                  lluvia_promedio * 0.25 +             # Lluvia alta aumenta riesgo
                  proximidad_drenaje_promedio * 0.20 +  # Lejos de drenaje aumenta riesgo
                  (1 - drainage_rain_index_promedio) * 0.15 + # Bajo índice drenaje-lluvia aumenta riesgo
                  (1 - proximity_index_promedio) * 0.10  # Bajo índice proximidad aumenta riesgo
  ) %>%
  arrange(desc(risk_index))

cat("\n===== ÍNDICE DE RIESGO POR CLUSTER =====\n")

```

```

##
## ===== ÍNDICE DE RIESGO POR CLUSTER =====

```

```

print(cluster_stats %>% select(Cluster, risk_index))

```

```

## # A tibble: 3 x 2
##   Cluster risk_index
##   <fct>      <dbl>
## 1 1          0.0239
## 2 3         -0.807
## 3 2         -0.850

```

```

# Asignar nivel de riesgo según el índice
cluster_stats <- cluster_stats %>%
  mutate(
    risk_level = case_when(
      risk_index >= quantile(risk_index, 0.67) ~ "Alto",
      risk_index >= quantile(risk_index, 0.33) ~ "Medio",
      TRUE ~ "Bajo"
    )
  )

cat("\n===== PERFILES DE RIESGO POR CLUSTER =====\n")

```

```

##
## ===== PERFILES DE RIESGO POR CLUSTER =====

```

```

print(cluster_stats %>% select(Cluster, risk_index, risk_level,
                              elevacion_promedio, lluvia_promedio,
                              proximidad_drenaje_promedio,
                              elevation_rain_ratio_promedio))

```

```

## # A tibble: 3 x 7
##   Cluster risk_index risk_level elevacion_promedio lluvia_promedio
##   <fct>      <dbl> <chr>          <dbl>          <dbl>
## 1 1          0.0239 Alto           0.160          0.274
## 2 3         -0.807 Medio          0.164          0.259
## 3 2         -0.850 Bajo           0.161          0.260
## # i 2 more variables: proximidad_drenaje_promedio <dbl>,
## #   elevation_rain_ratio_promedio <dbl>

```

```

# Mapear niveles de riesgo a ambos datasets
risk_mapping <- setNames(cluster_stats$risk_level, cluster_stats$Cluster)
DataTransform$risk_level <- risk_mapping[as.character(DataTransform$Cluster)]
DataLimpia$risk_level <- risk_mapping[as.character(DataLimpia$Cluster)]

# Convertir a factor con niveles ordenados
DataTransform$risk_level <- factor(DataTransform$risk_level,
                                   levels = c("Bajo", "Medio", "Alto"))
DataLimpia$risk_level <- factor(DataLimpia$risk_level,
                                levels = c("Bajo", "Medio", "Alto"))

cat("\n===== MAPEO COMPLETADO =====\n")

```

```

##
## ===== MAPEO COMPLETADO =====

```

```

cat("Niveles de riesgo asignados a DataTransform y DataLimpia\n")

```

```

## Niveles de riesgo asignados a DataTransform y DataLimpia

```

```

# Distribución final
cat("\n===== DISTRIBUCIÓN FINAL POR NIVEL DE RIESGO =====\n")

```

```

##
## ===== DISTRIBUCIÓN FINAL POR NIVEL DE RIESGO =====

```

```

print(table(DataLimpia$risk_level))

```

```

##
##  Bajo Medio  Alto
##  1074   587   403

```

```

cat("\nPorcentajes:\n")

```

```

##
## Porcentajes:

```

```

print(round(prop.table(table(DataLimpia$risk_level)) * 100, 2))

```

```

##
##  Bajo Medio  Alto
##  52.03 28.44 19.53

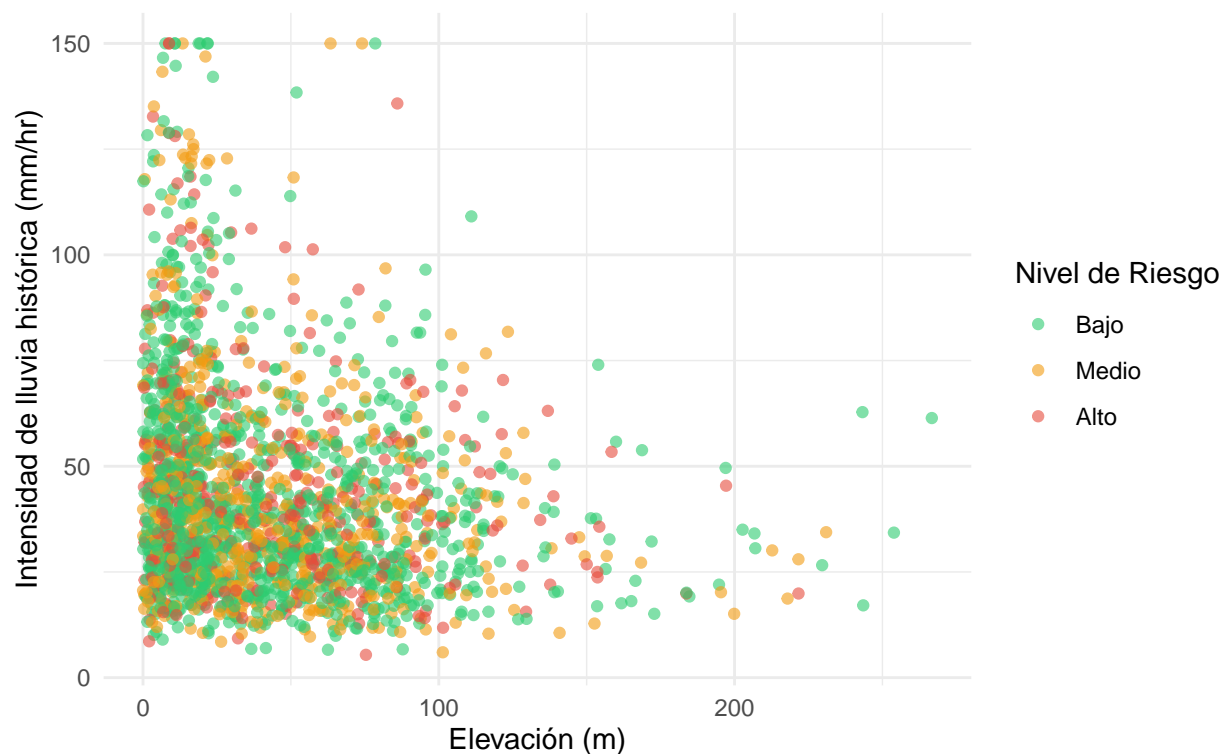
```

## VISUALIZACIONES FINALES

```
# Visualización 1: Distribución de riesgo (Elevación vs Lluvia)
ggplot(DataLimpia, aes(x = elevation_m, y = historical_rainfall_intensity_mm_hr,
                      color = risk_level)) +
  geom_point(alpha = 0.6, size = 1.5) +
  scale_color_manual(values = c("Bajo" = "#2ecc71",
                                "Medio" = "#f39c12",
                                "Alto" = "#e74c3c")) +
  labs(title = "Distribución de Riesgo de Inundación por Cluster",
       subtitle = "K-Means con k=3",
       x = "Elevación (m)",
       y = "Intensidad de lluvia histórica (mm/hr)",
       color = "Nivel de Riesgo") +
  theme_minimal() +
  theme(legend.position = "right",
        plot.title = element_text(face = "bold", size = 14))
```

## Distribución de Riesgo de Inundación por Cluster

K-Means con k=3



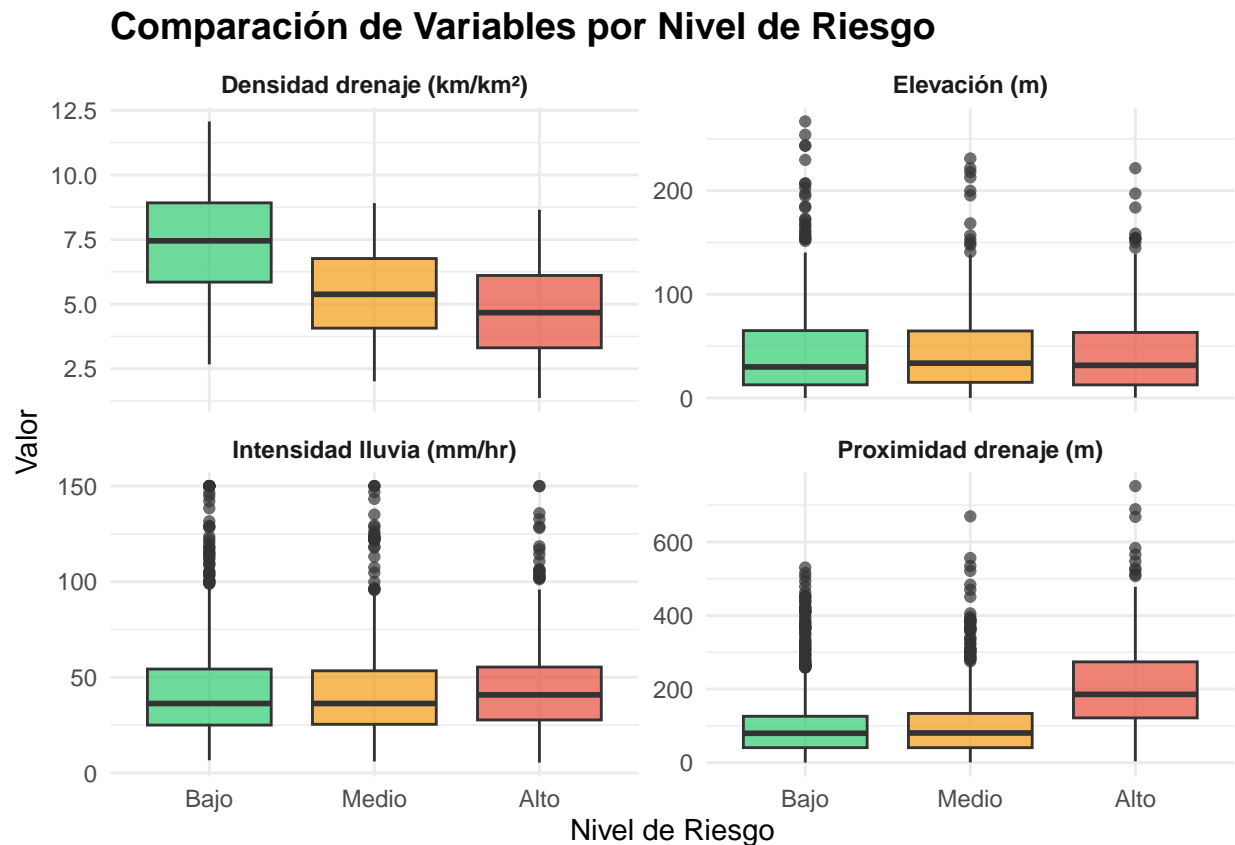
Se evidencia la distribución del riesgo de inundación según los clústeres obtenidos, con base en dos variables ambientales clave: elevación (m) y la intensidad histórica de lluvia (mm/hr). Las áreas con menor elevación y mayor intensidad de lluvia histórica presentan mayor concentración de puntos correspondientes al clúster de riesgo Alto.

```
# Visualización 2: Boxplots comparativos
DataLimpia %>%
  select(risk_level, elevation_m, historical_rainfall_intensity_mm_hr,
         drainage_density_km_per_km2, storm_drain_proximity_m) %>%
```

```

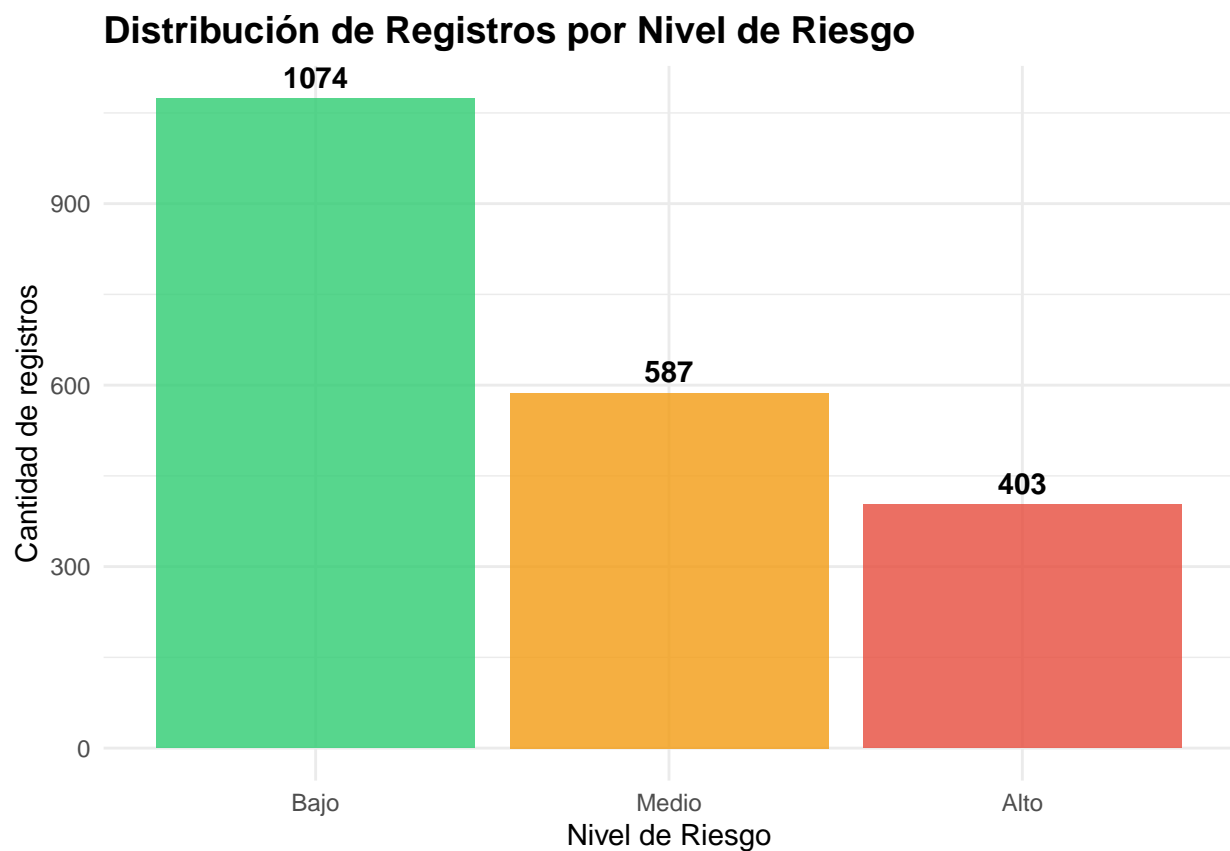
pivot_longer(cols = ~risk_level, names_to = "variable", values_to = "value") %>%
ggplot(aes(x = risk_level, y = value, fill = risk_level)) +
geom_boxplot(alpha = 0.7) +
facet_wrap(~variable, scales = "free_y", ncol = 2,
  labeller = labeller(variable = c(
    "elevation_m" = "Elevación (m)",
    "historical_rainfall_intensity_mm_hr" = "Intensidad lluvia (mm/hr)",
    "drainage_density_km_per_km2" = "Densidad drenaje (km/km²)",
    "storm_drain_proximity_m" = "Proximidad drenaje (m)"
  ))) +
scale_fill_manual(values = c("Bajo" = "#2ecc71",
  "Medio" = "#f39c12",
  "Alto" = "#e74c3c")) +
labs(title = "Comparación de Variables por Nivel de Riesgo",
  x = "Nivel de Riesgo",
  y = "Valor") +
theme_minimal() +
theme(legend.position = "none",
  plot.title = element_text(face = "bold", size = 14),
  strip.text = element_text(face = "bold"))

```



Los boxplots muestran la distribución de cada variable clave según el nivel de riesgo asignado, confirmando la coherencia de la clasificación.

```
# Visualización 3: Barras de distribución
ggplot(DataLimpia, aes(x = risk_level, fill = risk_level)) +
  geom_bar(alpha = 0.8) +
  geom_text(stat = 'count', aes(label = after_stat(count)),
           vjust = -0.5, size = 4, fontface = "bold") +
  scale_fill_manual(values = c("Bajo" = "#2ecc71",
                               "Medio" = "#f39c12",
                               "Alto" = "#e74c3c")) +
  labs(title = "Distribución de Registros por Nivel de Riesgo",
       x = "Nivel de Riesgo",
       y = "Cantidad de registros") +
  theme_minimal() +
  theme(legend.position = "none",
       plot.title = element_text(face = "bold", size = 14))
```



```
# ESTADÍSTICAS FINALES POR NIVEL DE RIESGO
cat("\n===== ESTADÍSTICAS DESCRIPTIVAS POR NIVEL DE RIESGO =====\n")
```

```
##
## ===== ESTADÍSTICAS DESCRIPTIVAS POR NIVEL DE RIESGO =====
```

```
# Estadísticas usando datos originales de DataLimpia
stats_finales <- DataLimpia %>%
  group_by(risk_level) %>%
```

```

summarise(
  n = n(),
  porcentaje = round(n() / nrow(DataLimpia) * 100, 2),
  elevacion_media = round(mean(elevation_m, na.rm = TRUE), 2),
  elevacion_sd = round(sd(elevation_m, na.rm = TRUE), 2),
  lluvia_media = round(mean(historical_rainfall_intensity_mm_hr, na.rm = TRUE), 2),
  lluvia_sd = round(sd(historical_rainfall_intensity_mm_hr, na.rm = TRUE), 2),
  proximidad_drenaje_media = round(mean(storm_drain_proximity_m, na.rm = TRUE), 2),
  densidad_drenaje_media = round(mean(drainage_density_km_per_km2, na.rm = TRUE), 2),
  .groups = 'drop'
)

print(stats_finales)

```

```

## # A tibble: 3 x 9
##   risk_level      n porcentaje elevacion_media elevacion_sd lluvia_media
##   <fct>         <int>      <dbl>          <dbl>      <dbl>      <dbl>
## 1 Bajo          1074        52.0            42.9        39.3        42.9
## 2 Medio          587        28.4            43.8        37.5        42.8
## 3 Alto          403        19.5            42.6        37.7        45.1
## # i 3 more variables: lluvia_sd <dbl>, proximidad_drenaje_media <dbl>,
## #   densidad_drenaje_media <dbl>

```

```

# Estadísticas usando variables transformadas de DataTransform
cat("\n===== ESTADÍSTICAS CON VARIABLES NORMALIZADAS Y DERIVADAS =====\n")

```

```

##
## ===== ESTADÍSTICAS CON VARIABLES NORMALIZADAS Y DERIVADAS =====

```

```

stats_transform <- DataTransform %>%
  group_by(risk_level) %>%
  summarise(
    n = n(),
    elevacion_norm_media = round(mean(norm_elevation_m, na.rm = TRUE), 3),
    lluvia_norm_media = round(mean(norm_historical_rainfall_intensity_mm_hr, na.rm = TRUE), 3),
    elevation_rain_ratio_media = round(mean(elevation_rain_ratio, na.rm = TRUE), 3),
    drainage_rain_index_media = round(mean(drainage_rain_index, na.rm = TRUE), 3),
    proximity_index_media = round(mean(proximity_index, na.rm = TRUE), 3),
    .groups = 'drop'
  )

print(stats_transform)

```

```

## # A tibble: 3 x 7
##   risk_level      n elevacion_norm_media lluvia_norm_media elevation_rain_ratio~1
##   <fct>         <int>      <dbl>          <dbl>          <dbl>
## 1 Bajo          1074        0.161            0.26            1.09
## 2 Medio          587        0.164            0.259           1.22
## 3 Alto          403        0.16             0.274           1.58
## # i abbreviated name: 1: elevation_rain_ratio_media
## # i 2 more variables: drainage_rain_index_media <dbl>,
## #   proximity_index_media <dbl>

```

```
cat("\n===== ANÁLISIS DE CLUSTERING COMPLETADO =====\n")
```

```
##
```

```
## ===== ANÁLISIS DE CLUSTERING COMPLETADO =====
```

```
cat("Método utilizado: K-Means (k=3)\n")
```

```
## Método utilizado: K-Means (k=3)
```

```
cat("Coeficiente de Silueta:", round(avg_sil_kmeans, 3), "\n")
```

```
## Coeficiente de Silueta: 0.067
```

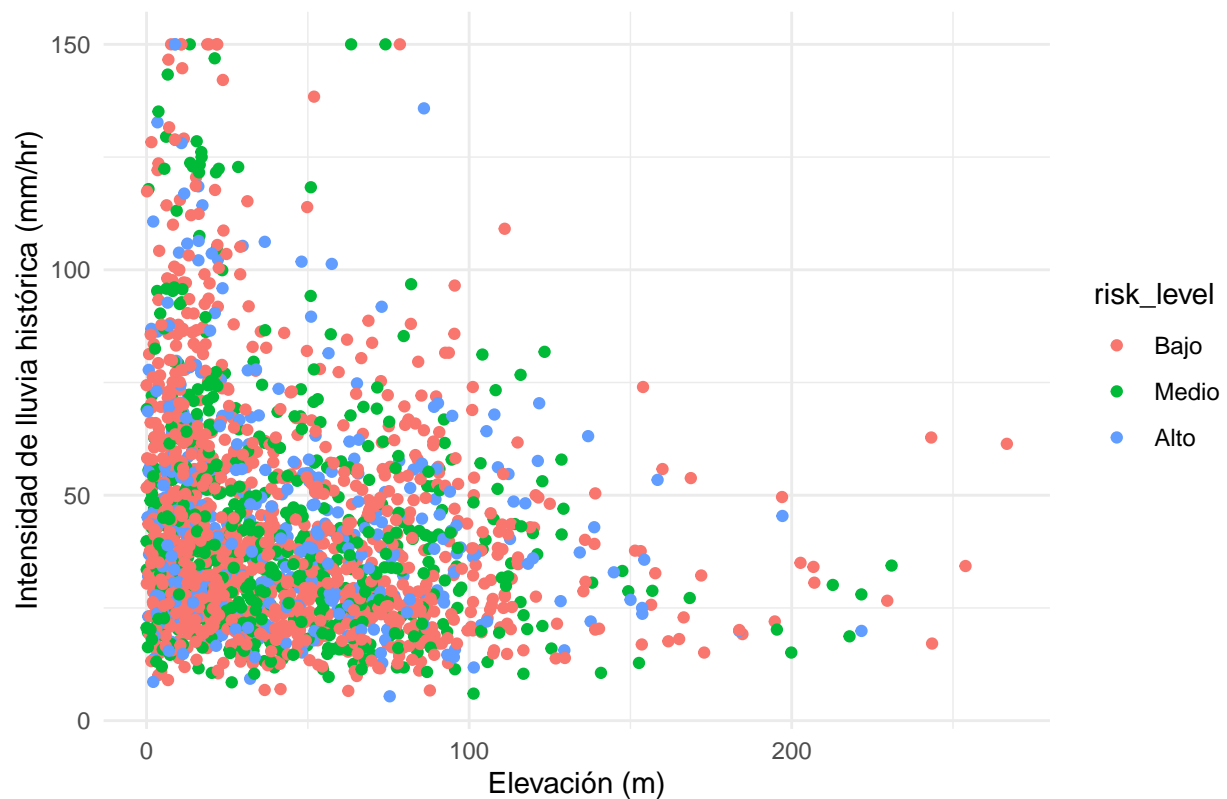
```
cat("Varianza explicada (BSS/TSS):", round(kmeans_model$betweenss / kmeans_model$totss * 100, 2), "%\n")
```

```
## Varianza explicada (BSS/TSS): 11.55 %
```

```
library(ggplot2)
```

```
ggplot(DataLimpia, aes(x = elevation_m, y = historical_rainfall_intensity_mm_hr, color = risk_level)) +  
  geom_point() +  
  labs(title = "Distribución de Riesgo de Inundación por Clúster",  
        x = "Elevación (m)",  
        y = "Intensidad de lluvia histórica (mm/hr)") +  
  theme_minimal()
```

Distribución de Riesgo de Inundación por Clúster



```
# Estadísticas descriptivas por nivel de riesgo
cat("\n===== ESTADÍSTICAS POR NIVEL DE RIESGO =====\n")
```

```
##
## ===== ESTADÍSTICAS POR NIVEL DE RIESGO =====
```

```
DataLimpia %>%
  filter(risk_level != "Sin clasificar") %>%
  group_by(risk_level) %>%
  summarise(
    n = n(),
    elevacion_promedio = round(mean(elevation_m, na.rm = TRUE), 2),
    lluvia_promedio = round(mean(historical_rainfall_intensity_mm_hr, na.rm = TRUE), 2),
    proximidad_drenaje = round(mean(storm_drain_proximity_m, na.rm = TRUE), 2),
    .groups = 'drop'
  ) %>%
  print()
```

```
## # A tibble: 3 x 5
##   risk_level      n elevacion_promedio lluvia_promedio proximidad_drenaje
##   <fct>      <int>          <dbl>          <dbl>          <dbl>
## 1 Bajo        1074          42.9          42.9          100.
## 2 Medio        587          43.8          42.8          106.
## 3 Alto        403          42.6          45.1          205.
```

```
cat("\n===== ANÁLISIS DE CLUSTERING COMPLETADO =====\n")
```

```
##
## ===== ANÁLISIS DE CLUSTERING COMPLETADO =====
```

““

Se evidencia la distribución del riesgo de inundación según los clústeres obtenidos, con base en dos variables ambientales clave: elevación (m) y la intensidad histórica de lluvia (mm/hr). Se observa que:

Las áreas con menor elevación y mayor intensidad de lluvia histórica presentan mayor concentración de puntos correspondientes al clúster de riesgo Alto, indicando mayor susceptibilidad a inundaciones.

El clúster de riesgo Medio se ubica en una zona intermedia tanto en elevación como en precipitación.

El clúster de riesgo Bajo se encuentra mayormente distribuido en zonas con elevaciones más altas y menor intensidad de lluvia, lo cual sugiere condiciones más seguras frente a posibles inundaciones.

## 4.5 Minería de Datos

- Seleccionar y justificar el algoritmo o técnica empleada (clasificación, regresión, clustering, etc.).
- Describir la división de datos (entrenamiento y prueba).
- Presentar las métricas de evaluación (Accuracy, F1-Score, MAE, etc.).
- Incluir visualizaciones que respalden los resultados del modelo.



## **4.6 Interpretación y Evaluación**

- Analizar críticamente los resultados obtenidos.
- Validar el conocimiento descubierto frente a las hipótesis y objetivos planteados.
- Evaluar el valor del conocimiento extraído para el contexto del problema.

## **5. Conclusiones**

- Sintetizar los principales hallazgos.
- Reflexionar sobre el proceso completo y sus limitaciones.
- Proponer posibles trabajos futuros o mejoras.

## **6. Anexos**

- Gráficos, tablas, fragmentos de código, resultados adicionales que complementen el análisis.