PROYECTO FINAL: ACCIDENTES EN ESTADOS UNIDOS

ÍNDICE:

1. Descripción del problema.

Los accidentes de tráfico son una de las principales causas de mortalidad y lesiones graves en todo el mundo, especialmente en países como Estados Unidos, donde la red de carreteras y autopistas es extensa y variada. Comprender los factores asociados con los accidentes, como la severidad de los mismos y las condiciones climáticas en las que ocurren, es esencial para mejorar la seguridad vial, optimizar los recursos de emergencia y diseñar políticas preventivas más efectivas.

El análisis de los accidentes de tráfico en Estados Unidos desde 2016 hasta marzo de 2023 tiene como objetivo identificar patrones relevantes entre la severidad de los accidentes, las condiciones climáticas y las ubicaciones geográficas (por estado). Con este análisis se busca responder preguntas clave como:

- ¿Qué estados presentan mayores promedios de severidad en condiciones climáticas específicas?
- ¿Qué factores climáticos están más correlacionados con accidentes graves?
- ¿Cómo varía el comportamiento del tráfico en función de la ubicación y el clima?

Estos datos pueden proporcionar información valiosa para agencias gubernamentales, compañías de seguros, investigadores de seguridad vial y ciudadanos interesados en mejorar la seguridad vial. Para abordar esta problemática, se utilizará un enfoque de Big Data que permita procesar y analizar un volumen significativo de datos históricos de accidentes de tráfico de forma eficiente y escalable.

La solución desarrollada se centra en aprovechar Apache Spark, una plataforma de procesamiento distribuido, para realizar tareas de filtrado, agregación y análisis avanzado. Este sistema permite extraer estadísticas clave que ayudan a caracterizar el impacto de las condiciones climáticas en la severidad de los accidentes por estado, con un énfasis en identificar áreas de alto riesgo y tendencias peligrosas a lo largo de los años analizados.

2. Necesidad de Big Data y la Nube.

-Big Data:

El conjunto de datos ocupa aproximadamente 3gb de almacenamiento, por lo tanto tiene un volumen de datos muy grande además de una complejidad muy alta para usar un sistema tradicional de análisis, sobre todo por la dificultad de la lectura y análisis de los datos.
Los datos requieren técnicas avanzadas de procesamiento y almacenamiento.

-Cloud:

La infraestructura de la nube permite la escalabilidad necesaria para analizar estos datos de manera eficiente.
Además en nuestro caso tenemos herramientas instaladas en Google Cloud (en este caso Spark) que pueden hacer tareas y cálculos en paralelo que nos permite reducir el tiempo en el que se procesa los datos de forma significativa.

3. Descripción de los datos

El conjunto de datos incluye información detallada sobre accidentes de tráfico ocurridos en Estados Unidos desde 2016 hasta marzo de 2023. Los atributos más relevantes para el análisis son:

- **Severidad del accidente**: Representada por una escala numérica (1 a 4), donde los valores más altos indican mayor gravedad.
- **Ubicación geográfica**: Información precisa del lugar del accidente, como estado, ciudad, y coordenadas geográficas (latitud y longitud). Esto permite estudiar patrones espaciales.
- **Condiciones climáticas**: Variables como temperatura, visibilidad, precipitación, velocidad del viento y una descripción de las condiciones generales (e.g., lluvia ligera, despejado) en el momento del accidente.
- **Factores temporales**: Fechas y horas de inicio y fin del accidente, lo que ayuda a identificar tendencias según franjas horarias o estaciones del año.
- **Factores de infraestructura vial**: Indicadores sobre la presencia de semáforos, cruces, rotondas y otras características de la carretera que podrían influir en la ocurrencia de los accidentes.

Este conjunto de datos se proporciona en formato CSV (Comma-Separated Values), lo que facilita su procesamiento y análisis. El tamaño del conjunto de datos es de 2.9 GB. Ha sido obtenido de Kaggle, una plataforma en línea que ofrece conjuntos de datos públicos para análisis y competencias de ciencia de datos. Lo hemos elegido debido a su amplitud y relevancia, ya que permite analizar patrones en accidentes de tráfico en función de variables como la severidad, las condiciones climáticas y la ubicación, lo cual es crucial para mejorar la seguridad vial.

4. Descripción de la aplicación, modelos de programación, plataforma e infraestructura.

La aplicación es un sistema de análisis de Big Data diseñado para analizar una base de datos de accidentes en Estados Unidos.
Objetivo principal: Relacionar las condiciones climatológicas con la gravedad de los accidentes.

Funcionalidades principales:

1. Relacionar condiciones climatológicas y severidad de accidentes.
2. Clasificar accidentes por estado.
3. Calcular el promedio de severidad de los accidentes por estado.
4. Obtener el top 5 de combinaciones de estado y clima con los promedios de severidad más altos.

Modelo de Programación

La aplicación utiliza el modelo de programación basado en Resilient Distributed Datasets (RDDs).

Operaciones Principales

1. **Transformaciones:**
    - **`filter`:** Filtra datos relevantes.
    - **`map`:** Transforma líneas de texto en pares clave-valor.
    - **`reduceByKey`:** Agrega datos por clave, sumando valores asociados.
    - **`sortBy`:** Ordena los resultados.
2. **Acciones:**
    - **`takeOrdered`:** Obtiene los registros con mayor severidad promedio.
    - **`saveAsTextFile`:** Escribe los resultados en el sistema de archivos.

Ventajas de RDDs en esta aplicación

- Distribución automática de datos entre nodos del clúster, acelerando el procesamiento.
- Tolerancia a fallos de Spark, garantizando la confiabilidad del análisis.

Plataforma

La aplicación está diseñada para ejecutarse en Apache Spark, aprovechando sus características clave:

- Compatibilidad con múltiples lenguajes: Usamos Python con PySpark.
- Distribución automática de datos y tareas: Spark distribuye de forma eficiente entre los nodos del clúster, mejorando rapidez y eficiencia.

Infraestructura

1) **Hardware:**

   Utilizamos un clúster de Dataproc de Google Cloud con múltiples nodos (Master Node que coordina el cluster y Worker Nodes que ejecutan las diferentes tareas).

2) **Software:**

   Apache Spark: Instalado en el clúster.

   Python (PySpark): Lenguaje principal de implementación.

3) **Almacenamiento:**

   Usamos un Bucket de Google Cloud para subir la entrada (un archivo `.csv` con los datos) y almacenar la salida (los resultados del análisis).

5. Diseño del software

El uso de este análisis de accidentes de tráfico se centra en identificar patrones entre la severidad de los accidentes, las condiciones climáticas y la ubicación geográfica. A continuación se describen los pasos clave del proceso, desde la carga del dataset hasta la obtención de los resultados:

1. **Carga de Datos:** El primer paso consiste en cargar el archivo CSV que contiene los datos de los accidentes. Este archivo es procesado por Apache Spark, lo que permite manejar grandes volúmenes de datos de forma eficiente.

```
# Leer el archivo de entrada
lines = sc.textFile(sys.argv[1])
```

2. **Filtrado de Datos:** Se filtran las líneas que contienen información incompleta o no válida, como registros con valores nulos o encabezados de columnas.

```
# Filtrar las líneas que no tienen encabezado y que tienen severidad válida (no vacía)
filtered_lines = lines.filter(lambda line: "ID" not in line.split(',')[0] and
line.split(',')[2] != '' and
line.split(',')[14] != '' and
line.split(',')[28] != '')
```

3. **Transformación de Datos:** Los datos relevantes, como el estado, las condiciones climáticas y la severidad, se extraen y transforman en pares clave-valor para facilitar su agregación.

```
# Transformar los datos relevantes ((State, Weather_Condition), (Severity, 1))
# Si la condición del clima o el estado son nulos, asignamos "Otros"
data = filtered_lines.map(lambda line: line.split(',')) \
                     .map(lambda fields: (
                         (fields[14],  # Estado
                          fields[28]),  # Condición climática
                         (int(fields[2]), 1)  # Severidad y conteo
                     ))
```

4. **Agregación de Datos:** Los datos se agrupan por estado y condición climática, y luego se suman los valores de severidad y el conteo de accidentes para cada grupo.

```
# Reducir por clave ((State, Weather_Condition)) para sumar severidad y contar accidentes
aggregated = data.reduceByKey(lambda acc, value: (acc[0] + value[0], acc[1] + value[1]))
```

5. **Cálculo del Promedio de Severidad:** Se calcula el promedio de severidad por estado y condición climática dividiendo la suma de severidad por el número de accidentes registrados.

```
# Calcular el promedio de severidad por estado y condición climática
averaged = aggregated.mapValues(lambda x: (round(x[0] / x[1], 2), x[1]))
```

6. **Ordenación de Resultados:** Los resultados se ordenan primero por estado y luego por el promedio de severidad (de mayor a menor).

```
# Ordenar primero por estado y luego por promedio de severidad
sorted_result = averaged.sortBy(lambda x: (x[0][0], -x[1][0]))
```

7. **Visualización de Resultados:** Los resultados se guardan en archivos de salida.

```
# Guardar el resultado en el archivo de salida
sorted_result.map(lambda x: f"{x[0][0]},{x[0][1]},{x[1][0]:.2f},{x[1][1]}").saveAsTextFile(sys.argv[2])
```

8. **Obtención del Top 5:** A continuación, se obtiene el Top 5 de los estados con la mayor severidad promedio para las condiciones climáticas específicas.

```
# Obtener el top 5 de estados con la condición climática
top_5 = averaged.takeOrdered(5, key=lambda x: -x[1][0])
```

9. **Visualización de Resultados:** Los resultados del Top 5 se guardan en otro archivo de salida. El resultado contiene el estado, la condición climática, el promedio de severidad y el número de accidentes.

```
# Convertir el top 5 en un RDD y guardarlo en el archivo de salida
sc.parallelize(top_5) \
  .map(lambda x: f"{x[0][0]},{x[0][1]},{x[1][0]},{x[1][1]}") \
  .saveAsTextFile(sys.argv[3])
```

6. USO

Para realizar todas las pruebas y generar los outputs (resultados), hemos llevado a cabo los siguientes pasos, basándonos en el laboratorio 4 de Spark:

1) **Configuración del clúster:**
   Una vez creado el fichero anterior, procedimos a configurar un clúster mediante el siguiente comando:

```
gcloud dataproc clusters create mycluster --region=europe-southwest1 \
--master-machine-type=e2-standard-4 --master-boot-disk-size=50 \
--worker-machine-type=e2-standard-4 --worker-boot-disk-size=50 \
--enable-component-gateway
```

2) **Carga de archivos al bucket:**
   A continuación, añadimos los archivos .py en el bucket que se creó durante los primeros laboratorios, así como el dataset de accidentes de tráfico.

3) **Generación de outputs:**
   Con el bucket y el clúster completamente configurados, ejecutamos el siguiente comando para generar los dos outputs:

```
BUCKET=gs://central-mission-436716-i4
gcloud dataproc jobs submit pyspark --cluster mycluster
--region=europe-southwest1 $BUCKET/codigo.py  --
$BUCKET/US_Accidents_March23.csv $BUCKET/prueba_normal
$BUCKET/prueba_top
```

   Siendo codigo.py el código en python, US_Accidents_March23.csv el dataset, y prueba_normal y prueba_top los outputs.

4) **Verificación de los outputs:**
Finalmente, comprobamos el contenido de los outputs generados utilizando este comando:

```
gcloud storage ls $BUCKET/output
gcloud storage cat $BUCKET/output/* | more
```

En este caso deberíamos de cambiar output por prueba_normal y prueba_top.

Este apartado nos daría los siguientes resultados:

prueba_normal:

```
AL,Freezing Drizzle,3.00,1
AL,Showers in the Vicinity,3.00,2
AL,Light Rain Showers,3.00,1
AL,Patches of Fog,2.82,17
AL,Thunderstorms and Rain,2.78,36
AL,Light Freezing Fog,2.68,28
AL,Snow,2.67,6
AL,Mist,2.61,54
AL,Heavy Thunderstorms and Rain,2.61,59
AL,Heavy Snow,2.60,5
AL,Thunderstorm,2.54,132
AL,Light Thunderstorms and Rain,2.48,107
AL,Scattered Clouds,2.43,1825
```

prueba_top:

```
SD,Smoke,4.0,1
WI,Light Rain Showers,4.0,2
DE,Ice Pellets,4.0,2
NE,Heavy Snow / Windy,4.0,1
MO,Squalls / Windy,4.0,1
```

Para acceder a los resultados completos, los outputs se encuentran en el proyecto GROUP2, el cual está asociado al correo electrónico: alvarodschez@gmail.com.

7. Evaluación de rendimiento

Cálculo del Speed Up

Para evaluar el rendimiento del sistema, se realizaron pruebas utilizando 2 nodos y 4 nodos. En cada caso, se midieron los tiempos de ejecución correspondientes, los cuales se presentan en las capturas adjuntas. A partir de estos tiempos, se calculó el Speed Up utilizando la fórmula:

Speed up = TIMEsequential / TIMEparallel

donde:

- TIMEsequential es el tiempo de ejecución en un nodo (o secuencial).
- TIMEparallel es el tiempo de ejecución utilizando nodos.

Este análisis permite ver el incremento en la eficiencia al aumentar el número de nodos empleados.

El tiempo secuencial (tiempo de ejecución en un nodo) es el siguiente:

```
24/12/18 19:09:46 INFO GoogleCloudStorageFileSystemImpl: Successfully repaired
24/12/18 19:09:50 INFO GoogleCloudStorageFileSystemImpl: Successfully repaired
Tiempo total de ejecución: 121.19 segundos
```

A partir de este tiempo hemos podido calcular el Speed up con 2 nodos y 4 nodos:

Resultados con 2 Nodos

En el caso de 2 nodos, los tiempos de ejecución se compararon con el tiempo secuencial para calcular el Speed Up. La captura adjunta muestra el tiempo obtenido, y el cálculo se realizó utilizando la fórmula mencionada anteriormente:

```
24/12/18 19:16:00 INFO GoogleCloudStorageFileSystemImpl: Successfully repaired
24/12/18 19:16:02 INFO GoogleCloudStorageFileSystemImpl: Successfully repaired
Tiempo total de ejecución: 67.51 segundos
```

A partir de este tiempo hemos obtenido el siguiente Speed up:

Speed up = 121.19 / 67.52 = 1.795

Resultados con 4 Nodos

Al emplear 4 nodos, se repitió el mismo procedimiento para medir los tiempos de ejecución y calcular el Speed Up. En la siguiente captura se puede observar la mejora en el rendimiento al incrementar la cantidad de nodos, evidenciando una mayor eficiencia del sistema:

```
24/12/18 19:22:26 INFO GoogleCloudStorageFileSystemImpl: Successfully repaired
24/12/18 19:22:28 INFO GoogleCloudStorageFileSystemImpl: Successfully repaired
Tiempo total de ejecución: 59.63 segundos
```

A partir de este tiempo hemos obtenido el siguiente Speed up:

Speed up = 121.19 / 59.63 = 2.032

Optimización del Código

Un aspecto relevante de la implementación es que se utilizó el mismo código para realizar dos funcionalidades distintas. Esta reutilización permitió optimizar el desarrollo y reducir el tiempo total de ejecución. Si las funcionalidades se hubieran desarrollado por separado, el tiempo necesario habría sido considerablemente mayor, tanto en términos de ejecución como de mantenimiento del código.

¿De qué manera se han obtenido los tiempos para el cálculo?

En base al código especificado anteriormente hemos realizado los siguientes cambios

1) Importar Time:

```
from pyspark import SparkConf, SparkContext
from time import time
import sys
```

2) Guardar el tiempo inicial en una variable start_time:

```
start_time = time()
```

3) Al final del código hacer print del tiempo final menos el tiempo inicial que ha sido guardado en la variable start_time:

```
print(f"Tiempo total de ejecución: {time() - start_time:.2f} segundos")
```

8. Características avanzadas

Novedades de la aplicación

En comparación con ejercicios anteriores, lo novedoso de esta aplicación es la implementación del cálculo del Top 5 de estados con mayor severidad de accidentes bajo condiciones climáticas específicas. Además, se optimizó el flujo de procesamiento para generar dos outputs distintos utilizando un único código, lo que simplifica y mejora la eficiencia del proceso.

En resumen, la aplicación aprovecha la capacidad de procesamiento distribuido de Apache Spark, la infraestructura escalable de Google Cloud, y las buenas prácticas del laboratorio de Spark para ejecutar el análisis de accidentes de tráfico de manera eficiente y obtener resultados útiles para la mejora de la seguridad vial.

9. Conclusiones

En definitiva, el análisis de los accidentes de tráfico en Estados Unidos entre 2016 y 2023 ha permitido identificar patrones significativos entre la severidad de los accidentes y las condiciones climáticas, como la lluvia ligera y la niebla, que están asociadas a una mayor gravedad. El uso de Big Data y Apache Spark ha sido clave para procesar grandes volúmenes de datos de manera eficiente, lo que facilita la identificación de áreas y condiciones de alto riesgo. Estos resultados son fundamentales para enfocar los esfuerzos preventivos y mejorar la seguridad vial en las zonas más vulnerables.

10. Referencias.

Kaggle (Para la obtención del dataset):

Kaggle es una plataforma en línea que ofrece un espacio para la compartición de conjuntos de datos, competiciones de ciencia de datos y recursos educativos. Es ampliamente utilizada por la comunidad de científicos de datos para acceder a datos públicos, aprender y participar en desafíos relacionados con el análisis de datos. En este trabajo, se utilizó un conjunto de datos de Kaggle sobre accidentes de tráfico en Estados Unidos entre 2016 y 2023.
https://www.kaggle.com

Apuntes del Classroom de la Asignatura (Para la implementación del código):

Los apuntes proporcionados en la asignatura sobre Batch Processing con Apache Spark fueron esenciales para la implementación de este análisis. En particular, el tema de Spark explica cómo procesar grandes volúmenes de datos de manera distribuida, utilizando RDDs y operaciones como map, filter, reduceByKey, y sortBy. Estos conceptos fueron aplicados en el procesamiento de los datos de accidentes de tráfico, lo que nos ha permitido obtener resultados de manera eficiente y en tiempos reducidos.