Universidad Panamericana
Maestría en Ciencia de Datos
Datos Masivos

Proyecto Final: *Pipeline Distribuido de Predicción para Iowa Liquor Sales en GCP*

Enrique Ulises Báez Gómez Tagle, Luis Alejandro Guillén Alvarez

3 de diciembre de 2025

Índice

1 Dataset utilizado **3**
1.1 Fuente y descripción . . . 3
1.2 Cardinalidades y dimensiones . . . 3
1.3 Calidad de los datos . . . 3
1.4 Distribución de ventas . . . 4
1.5 Top 10 categorías por volumen de ventas . . . 4
1.6 Justificación de selección . . . 4

2 Descripción de la arquitectura implementada **5**
2.1 Diagrama de arquitectura . . . 5
2.2 Flujo de datos . . . 5
2.3 Componentes de la arquitectura . . . 6

3 Desarrollo de la ruta elegida: Procesamiento Distribuido con PySpark **6**
3.1 Selección y exportación del dataset hacia GCS . . . 6
3.1.1 Proceso de exportación . . . 6
3.1.2 Configuración de Cloud Run . . . 7
3.1.3 Métricas de tiempo de ejecución . . . 7
3.1.4 Verificación de estructura y consistencia . . . 8
3.2 Procesamiento distribuido en Dataproc . . . 9
3.2.1 Configuración de clusters . . . 9
3.2.2 Lectura del dataset desde GCS mediante PySpark . . . 10
3.2.3 Aplicación de limpieza, filtrado y transformación . . . 11
3.2.4 Monitoreo de ejecución . . . 12
3.3 Modelado predictivo en PySpark (***PLANNED***) . . . 14
3.3.1 Modelo seleccionado . . . 14
3.3.2 Entrenamiento del modelo . . . 14
3.3.3 Métricas de evaluación . . . 14
3.4 Evaluación comparativa entre configuraciones de cluster . . . 14
3.4.1 Métricas de tiempo de ejecución . . . 14
3.4.2 Análisis de latencia, paralelismo y escalabilidad . . . 15

4 Métricas, gráficas y análisis de resultados **15**
4.1 Interpretación de resultados 15
4.2 Justificación del muestreo 15
4.3 Evaluación del desempeño del modelo 15

5 Análisis crítico del enfoque **15**
5.1 Ventajas del enfoque elegido 15
5.2 Limitaciones del enfoque elegido 15

6 Conclusiones **15**

7 Código utilizado **15**
7.1 Script principal de PySpark 15
7.2 Repositorio de código fuente 15

8 Referencias **16**

1. Dataset utilizado

1.1. Fuente y descripción

El dataset utilizado proviene de **BigQuery Public Data** y contiene registros de ventas de licores en el estado de Iowa, Estados Unidos. Este conjunto de datos es mantenido por el Iowa Department of Commerce y está disponible públicamente para análisis.

- **Fuente:** BigQuery Public Data - `bigquery-public-data.iowa_liquor_sales.sales`
- **Tamaño:** 32,816,143 registros
- **Periodo:** 2012-01-03 a 2025-10-31 (13.8 años)
- **Características principales:**
  - `date`: Fecha de la transacción
  - `store_number`: Identificador de la tienda
  - `city`: Ciudad donde se realizó la venta
  - `category`: Categoría del producto
  - `item_number`: Identificador del producto
  - `sale_dollars`: Monto de la venta (variable objetivo)
  - `bottles_sold`: Cantidad de botellas vendidas
  - `volume_sold_liters`: Volumen vendido en litros

1.2. Cardinalidades y dimensiones

El dataset presenta alta cardinalidad en múltiples dimensiones, lo que lo hace ideal para procesamiento distribuido:

Cuadro 1: Cardinalidades del dataset Iowa Liquor Sales.

Dimensión	Valores Únicos
Tiendas	3,337
Ciudades	504
Productos	15,183
Categorías	185

1.3. Calidad de los datos

El análisis exploratorio reveló una excelente calidad de datos con mínimos valores faltantes:

Cuadro 2: Valores nulos por campo.

Campo	Valores Nulos	Porcentaje
sale_dollars	10	0.00003 %
category	16,974	0.052 %
city	84,575	0.258 %
Total	**101,559**	**0.31 %**

Calidad general: 99.69 % de datos completos, lo que indica un dataset de alta calidad para modelado predictivo.

1.4. Distribución de ventas

La distribución de la variable objetivo (`sale_dollars`) muestra las siguientes características:

Cuadro 3: Distribución de ventas en dólares.

Percentil	Valor (USD)
P50 (Mediana)	$78.66
P90	$269.88
P99	$1,185.60

1.5. Top 10 categorías por volumen de ventas

Las categorías más vendidas representan una parte significativa del volumen total de transacciones:

Cuadro 4: Top 10 categorías por ventas totales.

Rank	Categoría	Ventas Totales (USD)	Transacciones
1	1012100.0	$495,078,200	2,778,490
2	1031100.0	$441,329,100	2,988,622
3	1011200.0	$288,427,900	1,859,256
4	1081600.0	$219,643,200	1,360,017
5	1062400.0	$169,326,700	861,360
6	1022200.0	$152,794,300	668,286
7	1031080.0	$145,760,500	1,265,930
8	1022100.0	$143,383,100	849,580
9	1011400.0	$119,534,300	538,956
10	1011100.0	$117,536,600	1,213,606
Total Top 10		**$2,292,813,900**	**15,384,103**

1.6. Justificación de selección

Este dataset fue seleccionado por las siguientes razones:

1. **Volumen masivo:** Con más de 32 millones de registros, cumple ampliamente con el requisito de $\geq$32M registros y justifica el uso de procesamiento distribuido con PySpark en Dataproc.
2. **Datos temporales:** El rango de 13.8 años permite análisis de series temporales y patrones estacionales, ideal para feature engineering temporal.
3. **Alta dimensionalidad:** La combinación de 15K+ productos, 185 categorías, 3.3K tiendas y 504 ciudades proporciona un espacio de características rico para modelado predictivo.
4. **Calidad excepcional:** Con 99.69 % de datos completos, minimiza la necesidad de imputación compleja y permite enfocarse en transformaciones y modelado.
5. **Variable objetivo continua:** `sale_dollars` es una variable continua ideal para regresión lineal, permitiendo predecir montos de venta basados en características de productos, ubicación y temporalidad.
6. **Disponibilidad pública:** Al estar en BigQuery Public Data, facilita la reproducibilidad del proyecto y el acceso sin restricciones de licenciamiento.
7. **Relevancia práctica:** Los modelos predictivos de ventas tienen aplicaciones directas en optimización de inventario, planificación de demanda y estrategias de pricing.

2. Descripción de la arquitectura implementada

2.1. Diagrama de arquitectura

La arquitectura implementada sigue un patrón de medallion con dos capas (Bronze y Gold) sobre Google Cloud Platform, integrando servicios de almacenamiento, procesamiento distribuido y análisis de datos masivos.

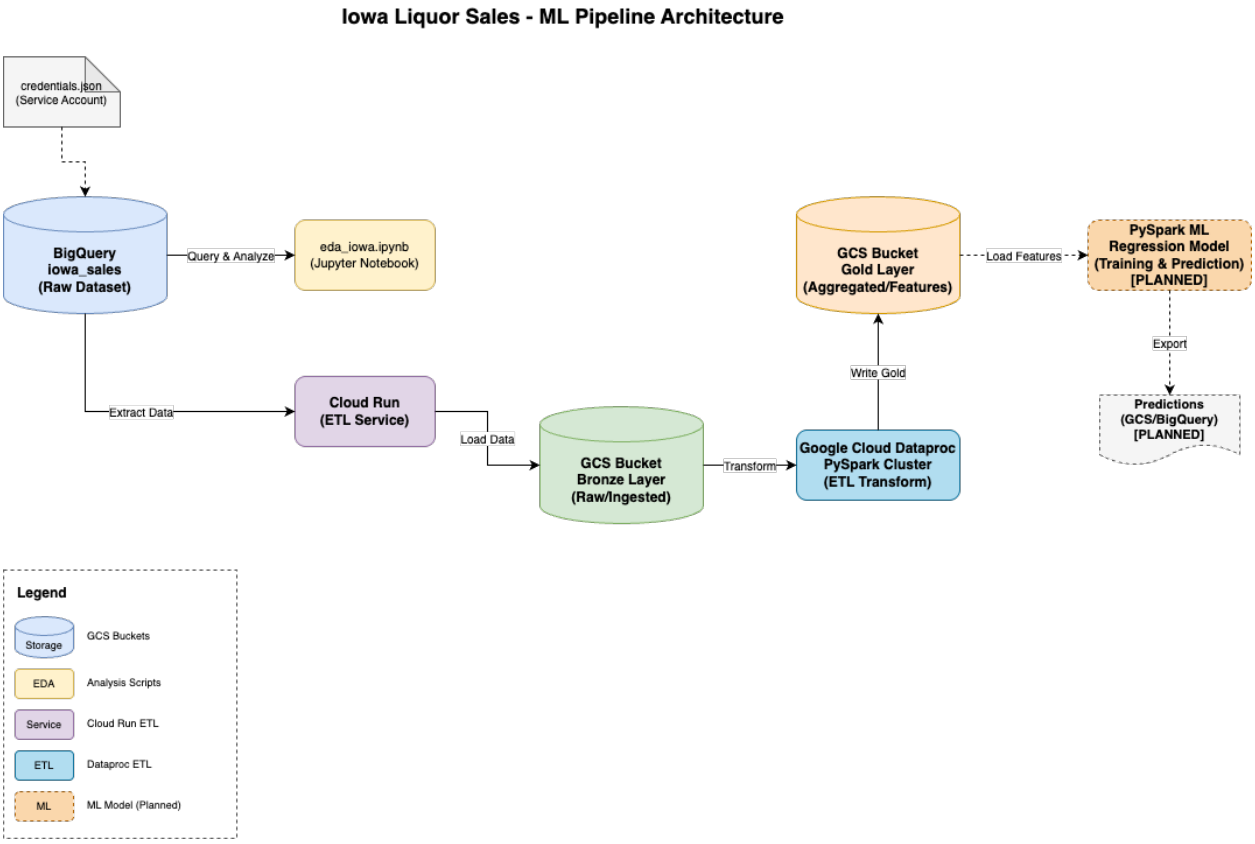

Figura 1: Arquitectura del pipeline distribuido: BigQuery → Cloud Run → GCS Bronze → Dataproc → GCS Gold → ML Model (***PLANNED***).

2.2. Flujo de datos

El pipeline implementa un flujo de datos end-to-end con las siguientes etapas:

1. **Fuente de datos (BigQuery):** El dataset público `iowa_liquor_sales` (32M+ registros) sirve como origen de datos. Los scripts de EDA (`eda_iowa.py` y `eda_iowa.ipynb`) realizan análisis exploratorio inicial directamente sobre BigQuery.

2. **Extracción (Cloud Run):** Un servicio ETL desplegado en Cloud Run ejecuta `bronze_extract.py`, que extrae datos desde BigQuery y los carga en formato Parquet particionado hacia la capa Bronze en Google Cloud Storage.

3. **Capa Bronze (GCS):** Almacenamiento de datos crudos en formato Parquet con particionamiento temporal, preservando la estructura original para trazabilidad y reproducibilidad.

4. **Transformación (Dataproc):** Un cluster de Dataproc ejecuta `gold_transform.py` con PySpark, aplicando limpieza, transformaciones y feature engineering sobre los datos Bronze. El procesamiento distribuido permite manejar el volumen masivo de forma eficiente.

5. **Capa Gold (GCS):** Datos limpios, transformados y enriquecidos con features derivadas, almacenados en formato Parquet particionado y optimizados para consumo analítico y modelado ML.
6. **Modelado ML (***PLANNED***):** Modelo de regresión lineal con PySpark MLlib entrenado sobre la capa Gold para predicción de ventas, con evaluación de métricas (R^2, RMSE, MAE) y comparación de performance entre configuraciones de cluster.

2.3. Componentes de la arquitectura

- **BigQuery:** Fuente de datos pública (`bigquery-public-data.iowa_liquor_sales.sales`)
- **Cloud Run:** Servicio ETL serverless para extracción batch hacia capa Bronze
- **GCS Bronze Layer:** Almacenamiento de datos crudos en formato Parquet particionado
- **Dataproc (PySpark):** Cluster de procesamiento distribuido para transformación y feature engineering
- **GCS Gold Layer:** Datos refinados listos para análisis y modelado
- **Terraform:** Infraestructura como código para provisionar clusters Dataproc con diferentes configuraciones
- **ML Model (***PLANNED***):** Modelo de regresión PySpark MLlib para predicción de ventas

3. Desarrollo de la ruta elegida: Procesamiento Distribuido con PySpark

3.1. Selección y exportación del dataset hacia GCS

3.1.1. Proceso de exportación

La extracción de datos desde BigQuery hacia Google Cloud Storage se implementó mediante un servicio ETL desplegado en Cloud Run, diseñado para ejecutarse como job batch serverless. El proceso consta de tres etapas principales:

Etapa 1: Creación de tabla temporal con particionamiento. El script `bronze_extract.py` ejecuta una consulta SQL que selecciona todos los registros del dataset público y agrega columnas derivadas de año y mes para facilitar el particionamiento posterior en PySpark:

```
SELECT
    *,
    EXTRACT(YEAR FROM date) as year,
    EXTRACT(MONTH FROM date) as month
FROM `bigquery-public-data.iowa_liquor_sales.sales`
```

Esta consulta materializa una tabla temporal en el dataset `ml_work.bronze_temp` del proyecto, permitiendo una exportación eficiente sin modificar la fuente original.

Etapa 2: Exportación a formato Parquet. Utilizando la API de BigQuery, se exportan los datos desde la tabla temporal hacia Google Cloud Storage en formato Parquet, un formato columnar optimizado para procesamiento distribuido:

- **Destino:** `gs://iowa-liquor-medallion-ml/bronze/iowa_sales/*.parquet`
- **Formato:** Parquet (columnar, comprimido)
- **Particionamiento:** Múltiples archivos generados automáticamente por BigQuery

Etapa 3: Limpieza y registro de métricas. Una vez completada la exportación, se elimina la tabla temporal y se registran las métricas de tiempo en un archivo JSON almacenado en GCS para trazabilidad.

3.1.2. Configuración de Cloud Run

El servicio se despliega mediante un contenedor Docker con las siguientes características:

- **Imagen base:** `python:3.11-slim`
- **Dependencias:** `google-cloud-bigquery`, `google-cloud-storage`, `pandas`, `pyarrow`, `db-dtypes`
- **Tipo de ejecución:** Cloud Run Job (batch, no HTTP)
- **Variables de entorno:**
  - `PROJECT_ID`: `secure-cipher-475203-k2`
  - `BUCKET_NAME`: `iowa-liquor-medallion-ml`

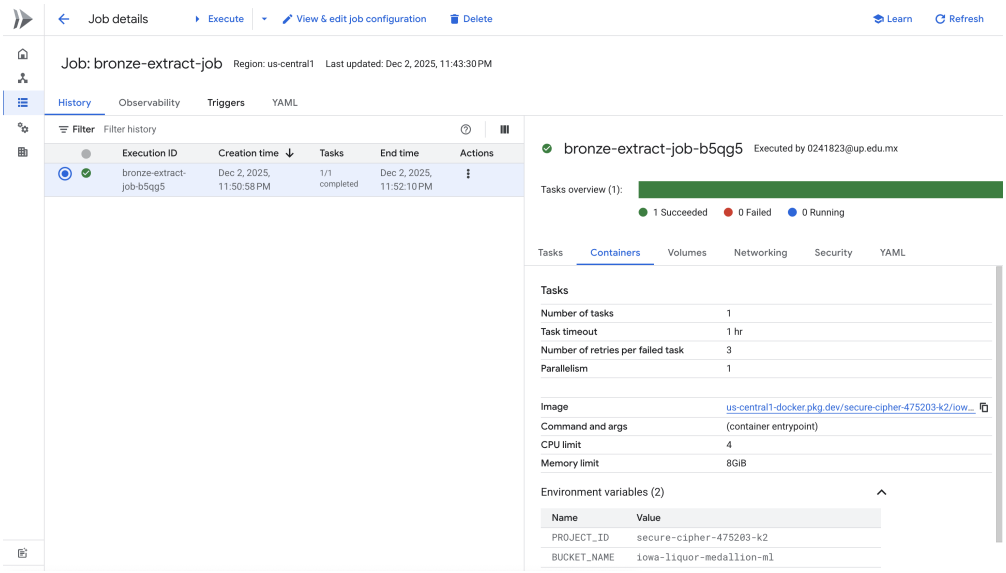

Figura 2: Configuración y historial de ejecución del Cloud Run Job para extracción Bronze.

3.1.3. Métricas de tiempo de ejecución

La fase de extracción Bronze completó exitosamente con las siguientes métricas:

Cuadro 5: Tiempos de ejecución de la fase Bronze (Cloud Run).

Etapa	Tiempo
Creación de tabla temporal	5.51s
Exportación a Parquet (GCS)	2.81s
Limpieza de recursos	0.16s
Tiempo total	**8.51s (0.14 min)**

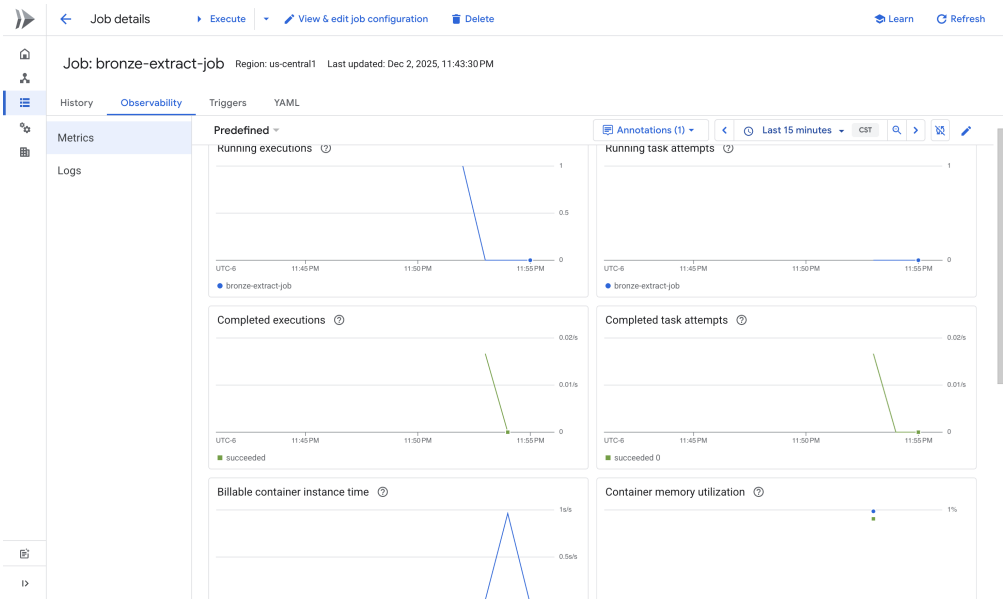

Figura 3: Métricas de observabilidad del Cloud Run Job mostrando ejecución exitosa.

3.1.4. Verificación de estructura y consistencia

Una vez completada la exportación, se verificó la estructura del bucket de GCS y la integridad de los datos:

Estructura del bucket. El bucket `iowa-liquor-medallion-ml` queda entonces con la siguiente carpeta:

- `bronze/iowa_sales/`: Datos crudos en formato Parquet (32,816,143 registros)

Archivos Parquet en capa Bronze. BigQuery generó múltiples archivos Parquet para optimizar la exportación paralela. Cada archivo contiene un subconjunto de los registros totales:

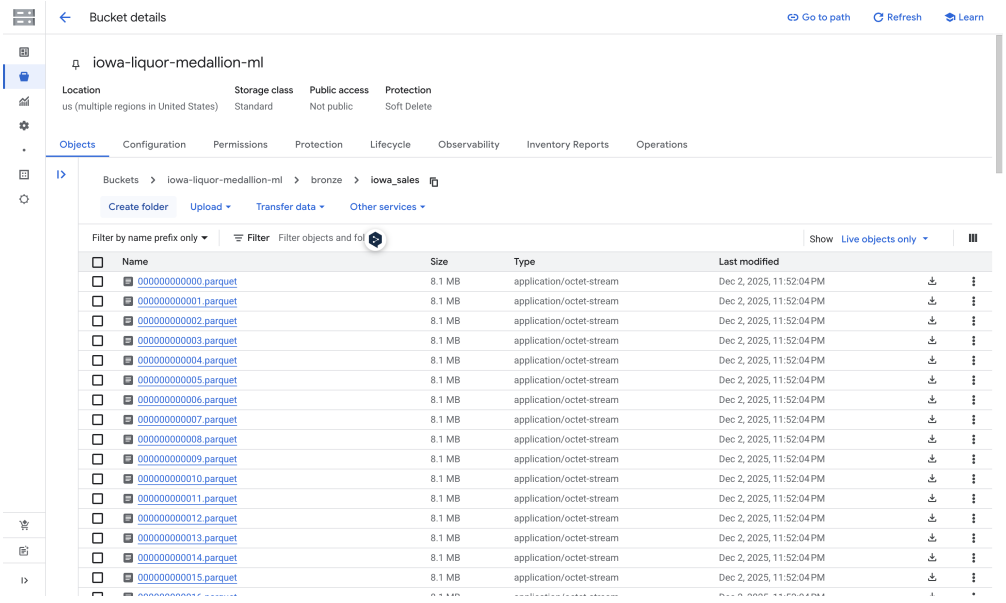

Figura 4: Archivos Parquet en la capa Bronze listos para procesamiento distribuido.

Validaciones realizadas.

- **Conteo de registros:** 32,816,143 registros exportados (coincide con el dataset original)
- **Formato:** Parquet columnar con compresión Snappy
- **Esquema:** Todas las columnas originales + columnas derivadas `year` y `month`
- **Integridad:** Sin errores de exportación, todos los archivos accesibles
- **Trazabilidad:** Métricas de tiempo registradas en `job_timing_bronze.json`

3.2. Procesamiento distribuido en Dataproc

3.2.1. Configuración de clusters

Se provisionaron dos configuraciones de clusters Dataproc mediante **infraestructura automatizada como código (Terraform)**, permitiendo despliegues reproducibles y parametrizables para evaluar el impacto del tamaño y tipo de máquina en el rendimiento del procesamiento distribuido:

Cluster 1: Configuración estándar (n1-standard). Cluster con perfil balanceado de CPU y memoria, optimizado para cargas de trabajo generales:

Cuadro 6: Configuración del Cluster 1 (n1-standard-3w).

Componente	Tipo de Máquina	Recursos
Master	n1-standard-2	2 vCPUs, 7.5 GB RAM
Workers (3x)	n1-standard-2	2 vCPUs, 7.5 GB RAM (cada uno)
Total		**8 vCPUs, 30 GB RAM**

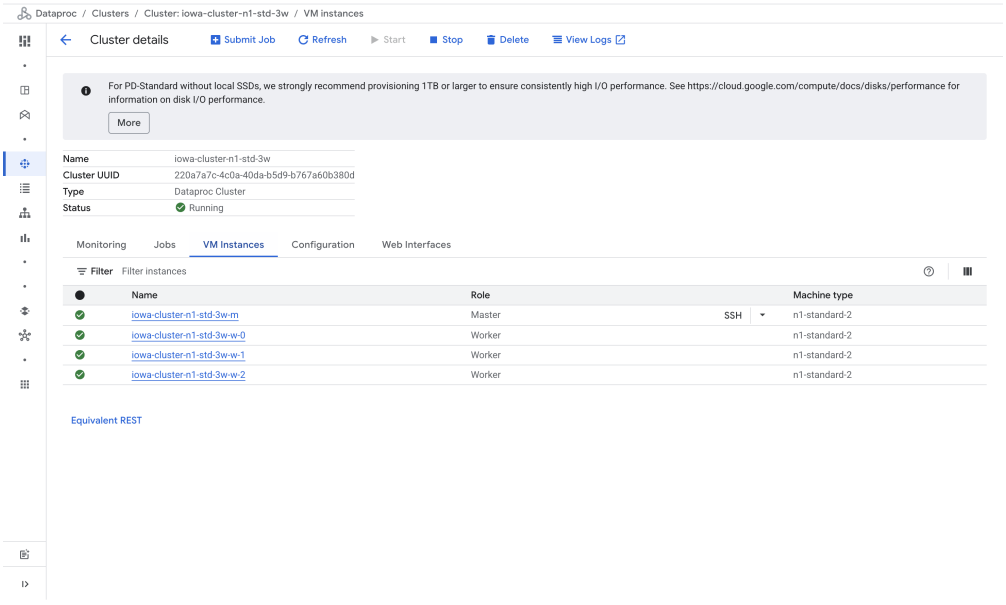

Figura 5: Configuración de instancias VM del Cluster 1 en Dataproc.

Cluster 2: Configuración high-memory (n2-highmem). Cluster con perfil de alta memoria, optimizado para cargas de trabajo intensivas en memoria:

Cuadro 7: Configuración del Cluster 2 (n2-highmem-4w).

Componente	Tipo de Máquina	Recursos
Master	n2-highmem-4	4 vCPUs, 32 GB RAM
Workers (4x)	n2-highmem-2	2 vCPUs, 16 GB RAM (cada uno)
Total		**12 vCPUs, 96 GB RAM**

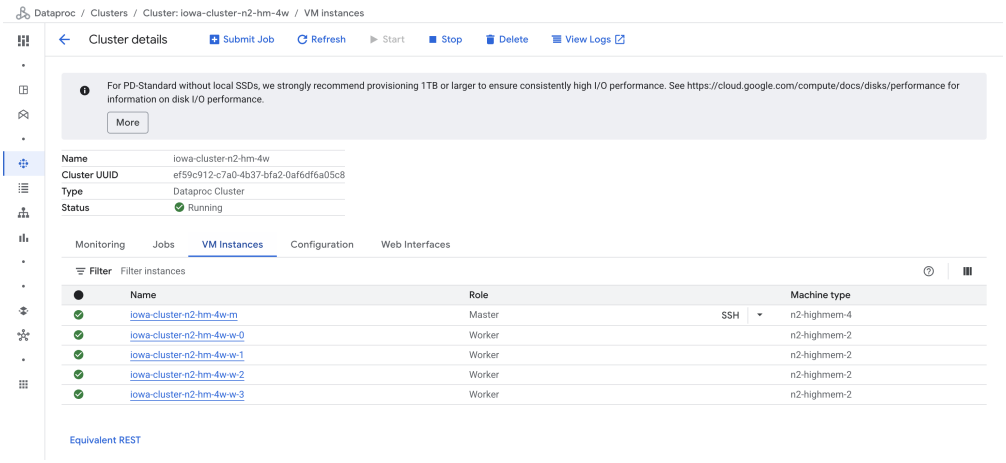

Figura 6: Configuración de instancias VM del Cluster 2 en Dataproc.

Infraestructura como código (Terraform). Los clusters se provisionan mediante módulos de Terraform con variables parametrizables:

```
# cluster1.tfvars
cluster_name        = "iowa-cluster-n1-std-3w"
master_machine_type = "n1-standard-2"
worker_machine_type = "n1-standard-2"
num_workers         = 3

# cluster2.tfvars
cluster_name        = "iowa-cluster-n2-hm-4w"
master_machine_type = "n2-highmem-4"
worker_machine_type = "n2-highmem-2"
num_workers         = 4
```

3.2.2. Lectura del dataset desde GCS mediante PySpark

El script `gold_transform.py` inicializa una sesión de Spark y lee los datos de la capa Bronze almacenados en formato Parquet:

```
BRONZE_PATH = f"gs://{BUCKET}/bronze/iowa_sales"
df = spark.read.parquet(BRONZE_PATH)
records_read = df.count()  # 32,816,143 registros
```

PySpark distribuye automáticamente la lectura de los múltiples archivos Parquet entre los workers del cluster, aprovechando el paralelismo para optimizar el tiempo de carga.

3.2.3. Aplicación de limpieza, filtrado y transformación

El procesamiento de la capa Gold se divide en tres etapas principales:

1. Limpieza de datos. Se aplican filtros para eliminar registros con valores nulos o inconsistentes en campos críticos:

```
df_clean = df.filter(
    (F.col("sale_dollars").isNotNull()) & (F.col("sale_dollars") > 0)
    & (F.col("bottles_sold").isNotNull()) & (F.col("bottles_sold") > 0)
    & (F.col("volume_sold_liters").isNotNull())
    & (F.col("volume_sold_liters") > 0)
).dropDuplicates()
```

Resultado: De 32,816,143 registros iniciales, se limpiaron 32,801,412 registros (99.96 % de retención), eliminando solo 14,731 registros (0.04 %) con datos inconsistentes.

2. Feature engineering. Se generan características derivadas para enriquecer el dataset y mejorar el potencial predictivo:

- `day_of_week`: Día de la semana (1-7) extraído de la fecha
- `quarter`: Trimestre del año (1-4)
- `is_weekend`: Indicador binario (1 si es fin de semana, 0 si no)
- `price_per_bottle`: Precio unitario calculado como `sale_dollars / bottles_sold`
- `volume_per_bottle`: Volumen unitario calculado como `volume_sold_liters / bottles_sold`

```
df_features = (
    df_clean
    .withColumn("day_of_week", F.dayofweek("date"))
    .withColumn("quarter", F.quarter("date"))
    .withColumn("is_weekend",
        F.when(F.dayofweek("date").isin([1, 7]), 1).otherwise(0))
    .withColumn("price_per_bottle",
        F.col("sale_dollars") / F.col("bottles_sold"))
    .withColumn("volume_per_bottle",
        F.col("volume_sold_liters") / F.col("bottles_sold"))
)
```

3. Escritura particionada a capa Gold. Los datos transformados se escriben en formato Parquet con particionamiento por año y mes para optimizar consultas futuras:

```
GOLD_PATH = f"gs://{BUCKET}/gold_{CLUSTER_NAME}/iowa_sales"
df_features.write.mode("overwrite")
    .partitionBy("year", "month")
    .parquet(GOLD_PATH)
```

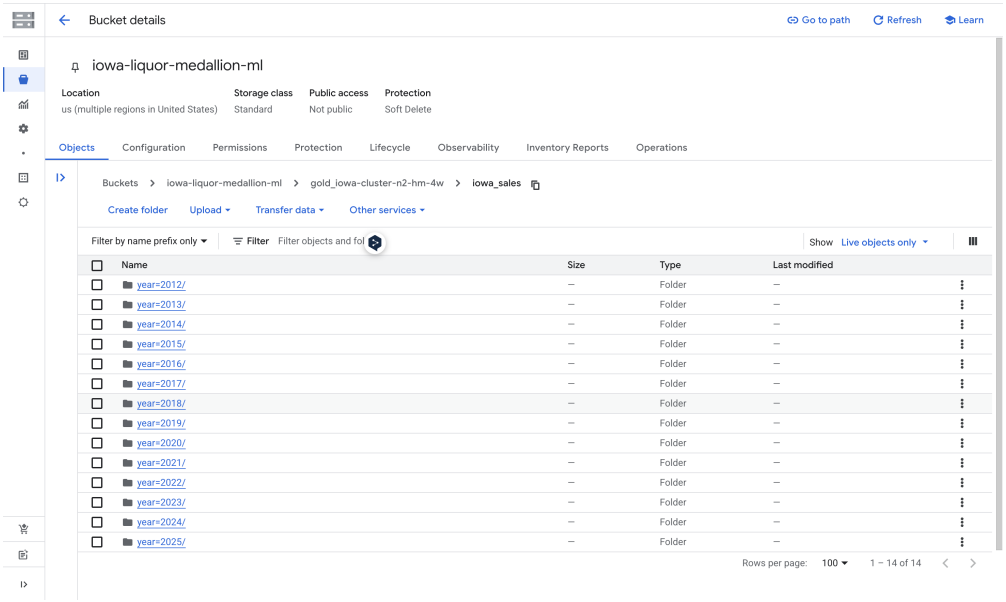

Figura 7: Estructura particionada por año en la capa Gold (Cluster 2).

3.2.4. Monitoreo de ejecución

Durante la ejecución de los jobs, se monitorearon métricas de recursos y tiempos mediante la interfaz de Spark History Server y YARN:

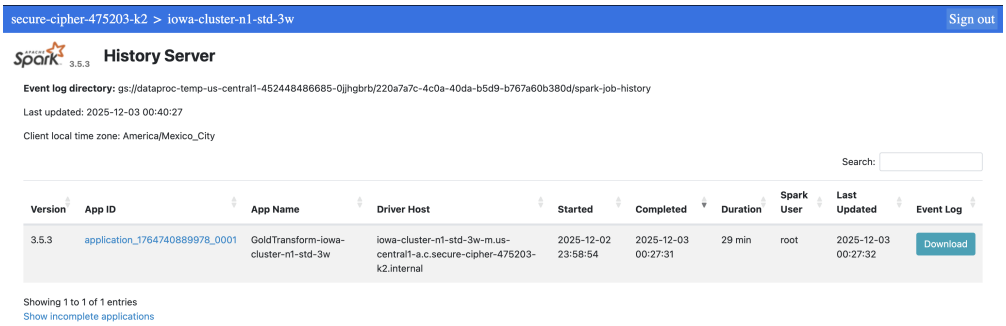

Figura 8: Resumen del job en Spark History Server (Cluster 1).

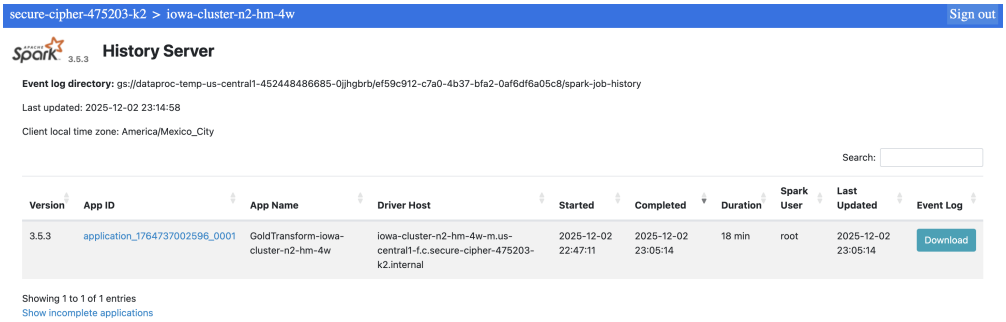

Figura 9: Resumen del job en Spark History Server (Cluster 2).

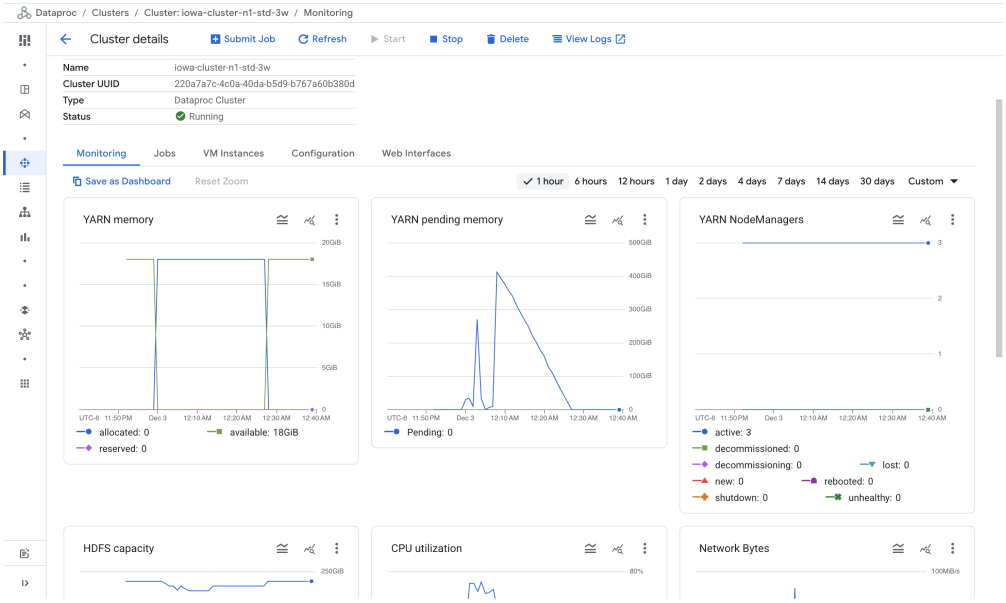

Figura 10: Monitoreo de CPU y memoria durante ejecución (Cluster 1).

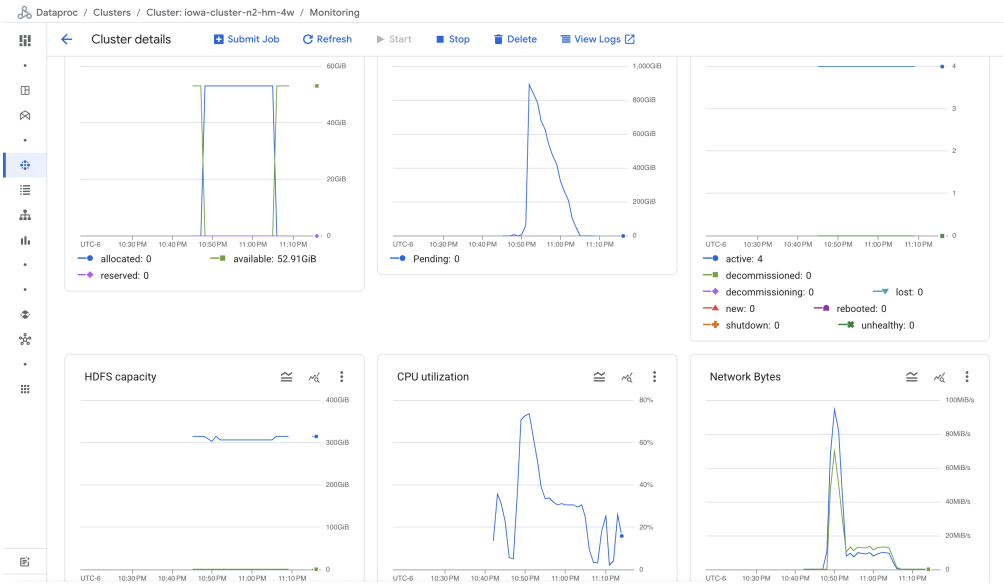

Figura 11: Monitoreo de CPU y memoria durante ejecución (Cluster 2).

3.3. Modelado predictivo en PySpark (***PLANNED***)

3.3.1. Modelo seleccionado

[Descripción del modelo de regresión lineal seleccionado]

3.3.2. Entrenamiento del modelo

[Proceso de entrenamiento sobre Gold layer]

3.3.3. Métricas de evaluación

[Tabla con métricas: R^2, RMSE, MAE]

Cuadro 8: Métricas de evaluación del modelo.

Métrica	Cluster 1	Cluster 2
R^2	[valor]	[valor]
RMSE	[valor]	[valor]
MAE	[valor]	[valor]

3.4. Evaluación comparativa entre configuraciones de cluster

3.4.1. Métricas de tiempo de ejecución

Cuadro 9: Comparativa de tiempos de ejecución.

Etapa	Cluster 1	Cluster 2	Diferencia
Lectura Bronze	[tiempo]	[tiempo]	[%]
Transformación	[tiempo]	[tiempo]	[%]
Escritura Gold	[tiempo]	[tiempo]	[%]
Total	[tiempo]	[tiempo]	[%]

Figura 12: Job UI de Dataproc mostrando métricas de tiempo y recursos.

3.4.2. Análisis de latencia, paralelismo y escalabilidad

[Interpretación de cómo el tamaño del cluster afecta la ejecución]

4. Métricas, gráficas y análisis de resultados

4.1. Interpretación de resultados

[Análisis de los resultados obtenidos]

4.2. Justificación del muestreo

[Explicación de las decisiones de muestreo si aplica]

4.3. Evaluación del desempeño del modelo

[Análisis crítico del desempeño]

5. Análisis crítico del enfoque

5.1. Ventajas del enfoque elegido

Ventaja 1

Ventaja 2

Ventaja 3

5.2. Limitaciones del enfoque elegido

Limitación 1

Limitación 2

Limitación 3

6. Conclusiones

[Conclusiones generales del proyecto]

7. Código utilizado

7.1. Script principal de PySpark

[Referencia al script principal]

7.2. Repositorio de código fuente

`https://github.com/[usuario]/ML-BigData`

8. Referencias

- Google LLC (s. f.). Google Cloud Console. https://console.cloud.google.com/
- Google Cloud. (2024). Crea un clúster de Dataproc con la consola de Google Cloud. https://cloud.google.com/dataproc/docs/quickstarts/create-cluster-console?hl=es-419
- BigQuery Public Data. Iowa Liquor Sales. https://console.cloud.google.com/marketplace/product/iowa-department-of-commerce/iowa-liquor-sales