Documentación Prueba Técnica DataKnow

# Sebastian Mariño

# 1. Carga de Información

La realización de este ejercicio fue desarrollada mediante el lenguaje de programación Python. En un primer acercamiento al problema, se utilizó la función open para abrir el archivo OFEI1204.txt ubicado en la carpeta '../Datos3'. El modo 'r' especifica que el archivo se abrirá en modo lectura. El contenido del archivo se leyó y almacenó en la variable txt\_data. Posteriormente, se imprimió el contenido del archivo para identificar las particularidades de la Data, en donde se notó que cada tabla de información venía delimitado por el titulo agente y líneas vacías.

Luego, se procedió a la extracción de información relevante del texto. Se dividió el contenido del archivo en secciones utilizando el método split con el delimitador "AGENTE:". Cada sección resultante contenía información sobre un agente específico. Se recorrieron estas secciones, extrayendo el nombre del agente y la tabla de datos asociada. La información de cada agente se agregó a la lista agentes, y las tablas de datos se procesaron y almacenaron en la lista dfs.

Después de completar la extracción de información, se utilizó la biblioteca pandas para manipular y analizar los datos de manera eficiente. Se concatenaron todas las tablas de datos en un único DataFrame llamado df\_final. Se filtraron las filas del DataFrame donde la columna "AGENTE" era igual a "EEPPM" para verificar la correcta lectura del archivo ya que se comparó esta salida con el archivo .txt original. Luego, se filtraron las filas donde la columna con índice 1 era igual a "D" y se reorganizaron las columnas. Se eliminó la columna 'TIPO' y se renombraron las columnas de acuerdo con los requisitos del problema.

Finalmente, se realizaron ajustes en el formato de algunas columnas, convirtiendo las columnas que representan horas a tipo de dato float. El DataFrame resultante se guardó en un archivo CSV llamado 'tabla\_punto\_1.csv' en la carpeta '../resultados'.

En resumen, el código realiza la extracción, procesamiento y manipulación de datos de un archivo de texto, utilizando Python y pandas, para obtener un DataFrame final que satisface los requisitos específicos del problema planteado.

# 2. Manipulación de datos

La implementación de este código en Python se realizó con el objetivo de manipular y analizar datos contenidos en un archivo Excel ('Datos Maestros VF.xlsx') y un archivo de texto ('dDEC1204.TXT'). Aquí se detalla paso a paso el procedimiento:

En primer lugar, se seleccionaron las columnas relevantes del archivo Excel utilizando la biblioteca pandas. Se leyó el archivo 'Datos Maestros VF.xlsx' y se extrajeron las columnas especificadas en la lista columnas\_seleccionadas. El DataFrame resultante se llamó df\_master\_data.

Luego, se filtraron las filas de df\_master\_data para obtener información específica sobre la empresa "EMGESA" y sobre centrales de tipo hidroeléctrico ("H") o térmico ("T"). El DataFrame resultante se denominó df\_master\_data\_emgesa.

A continuación, se leyó el archivo de texto 'dDEC1204.TXT' con la función pd.read\_csv, especificando el delimitador y la codificación adecuados. Se realizaron ajustes en el DataFrame resultante (df\_dec\_1204), eliminando la última fila y renombrando las columnas con nombres descriptivos.

Se llevó a cabo una fusión (pd.merge) entre los DataFrames df\_master\_data\_emgesa y df\_dec\_1204 utilizando la columna 'CENTRAL (dDEC, dSEGDES, dPRU…)' como clave de unión, y se especificó un tipo de fusión de tipo 'left'. El DataFrame resultante se llamó df\_merged\_emgesa.

A continuación, se seleccionaron las columnas que representan las horas ('Hora\_1', 'Hora\_2', ..., 'Hora\_24') y se calculó la suma de estas columnas para obtener la columna 'Total\_Horas'. Se filtraron las filas donde 'Total\_Horas' era mayor que 0, y el DataFrame resultante se llamó df\_tabla\_punto\_2.

Finalmente, se guardó el DataFrame df\_tabla\_punto\_2 en un archivo CSV llamado 'tabla\_punto\_2.csv' en la carpeta '../resultados'.

En resumen, este código realiza una serie de manipulaciones de datos utilizando pandas, fusionando información de un archivo Excel y un archivo de texto, y generando un DataFrame final que satisface los requisitos establecidos en el problema.

# 3. Prueba de SQL

Como primer paso se corrió la Query otorgada, la cual crea las tablas EMPLEADO,VACACIONES y se lleno con la información especificada.

Texto

Descripción generada automáticamente

Figura . Creación Tablas

A continuación, se muestran todas las Querys realizadas para obtener los datos solicitados y los respectivos resultados.

1. **Seleccionar nombre, apellido y salario de todos los empleados:**

**Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente**

Figura . Nombre apellido y salario

1. **Seleccionar nombre, apellido y salario de todos los empleados que ganen más de 4 millones:**

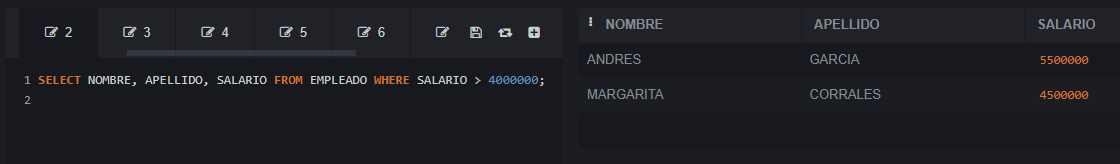
****

Figura . Empleados que ganas más de 4M

1. **Contar los empleados por sexo:**

**Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente**

Figura . Conteo por sexo

1. **Seleccionar los empleados que no han hecho solicitud de vacaciones:**

**Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente**

Figura . Empleados que no han solicitado vacaciones

1. **Seleccionar los empleados que tengan más de una solicitud de vacaciones y mostrar cuántas solicitudes tienen:**

**Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente**

Figura . Empleados que han solicitado vacaciones más de una vez

1. **Determinar el salario promedio de los empleados:**

**Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente**

Figura . Salario promedio

1. **Determinar la cantidad de días promedio solicitados de vacaciones por cada empleado:**

**Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente**

Figura . Dias poromedio solicitados por empleado

1. **Seleccionar el empleado que ha solicitado la mayor cantidad de días de vacaciones:**

**Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente**

Figura . Empleado que más ha solicitado vacaciones en total

1. **Consultar la cantidad de días aprobados y rechazados por cada empleado, mostrar 0 en caso de no tener solicitudes:**

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Figura . Tabla de días aprobados y rechazados de vacaciones

# 4. Prueba de AWS

***1. Configuración roles en AWS***

En la Figura 11 se observan los roles creados para las dos instancias requeridas para el desarrollo de este ejercicio, estas son s3 y redshift

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Figura . Roles

***2. Configure un Clúster de pruebas de Redshift con los requerimientos mínimos necesarios***

Se crea el Clúster de prueba para ingresar los datos necesarios.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Figura .Clúster

***3. Copie la información a Redshift usando algún editor de queries como SQL Workbench/J - Home (sql-workbench.eu)***

Mediante SQL workbench se conecta al clúster creado como se muestra en la siguiente Figura.

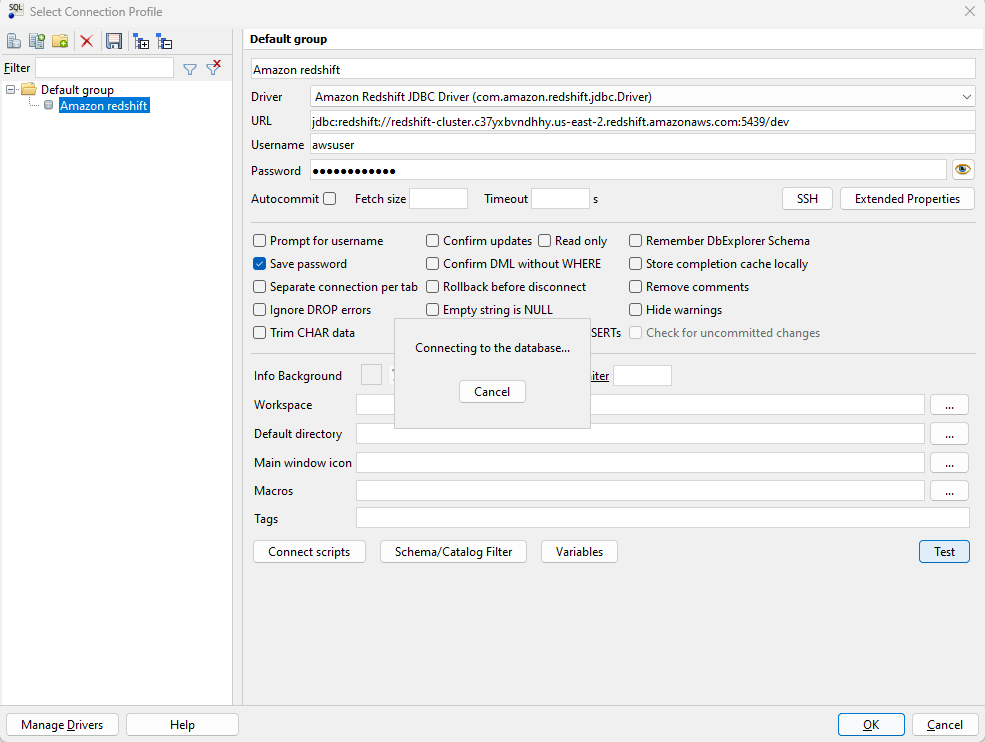
******

Figura . Workbench

Posteriormente se cargaron las tablas, a continuación, se muestra un ejemplo de Query SQl para cargar una tala, esto se repitió para los 7 archivos de la base de datos.

COPY events FROM 'C:\Users\{tu\_ruta}\tickitdb \allevents\_pipe.txt'

DELIMITER ',' -- Cambia a tu delimitador

IGNOREHEADER 1 -- Si hay un encabezado en el archivo

CSV; -- O el formato que estés utilizando

Ya que tickitdb es una base de dase de datos de prueba para redshift, otra forma de cargarla es cargando directamente desde la creación del Clúster del siguiente modo.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Figura . Carga de Tickit directamente desde la creación del Clúster

A continuación, se muestra la base de datos relacional tickit.

Diagrama

Descripción generada automáticamente

Figura . TickitDb

***4. Responda las siguientes preguntas usando comandos de consultas SQL:***

Median el redshift query editor se realizarán las consultas, a continuación, se muestran las tablas cargadas:

Captura de pantalla de un teléfono celular

Descripción generada automáticamente

Figura . Redshift query editor v2

A continuación, se muestran las Querys para resolver las preguntas planteadas:

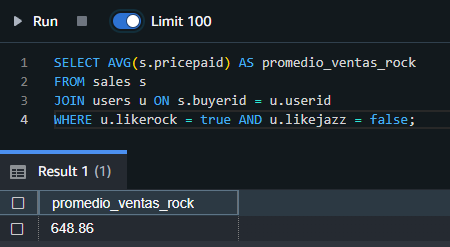
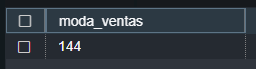
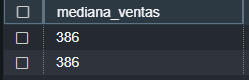
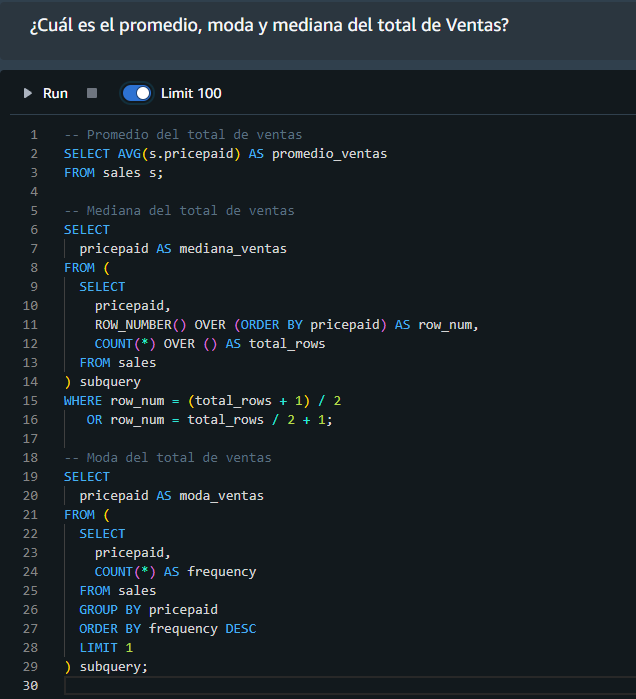
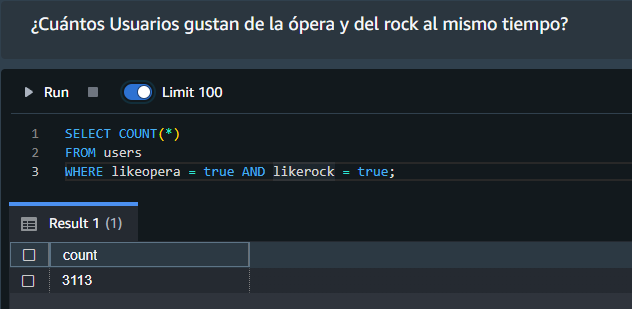
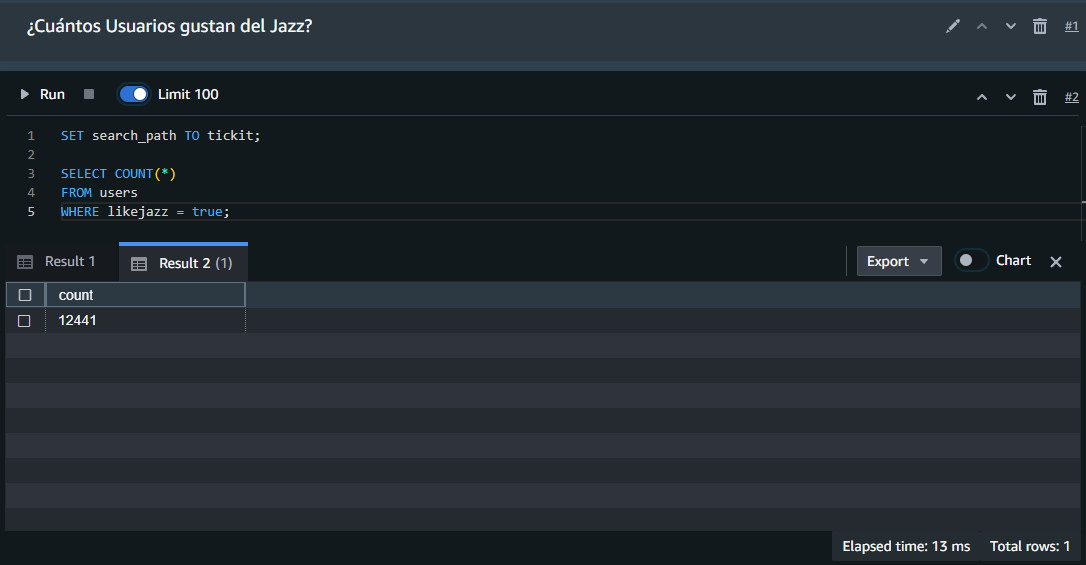


Figura . Querys para resolver las preguntas

***4. En una nueva tabla junte la información (Nombre de usuario, Apellido de usuario, Correo de usuario, Nombre del evento, lugar del evento, Fecha del evento, Cantidad y Total vendidos) y expórtela usando Redshift a un bucket predefinido de S3.:***

Se crea una instancia de S3 y se crea el bucket para almacenar una nueva tabla.

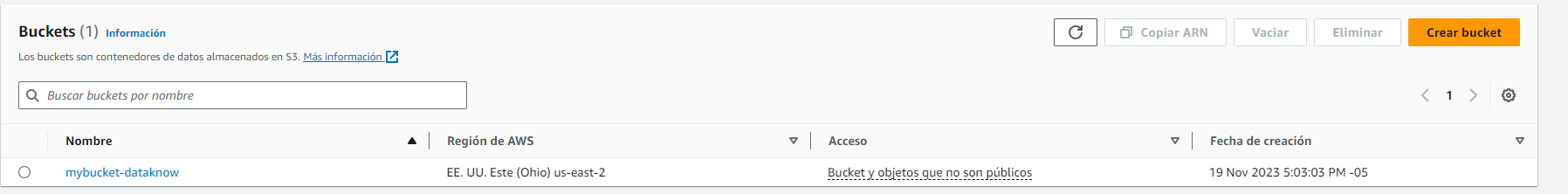


Figura . creación bucket

Posteriormente con la siguiente Query se crea la “nueva\_tabla”

Texto

Descripción generada automáticamente

Figura . Crear nueva tabla

Mediante la siguiente Query se carga la nueva en la ubicación deseada dentro del Bucket

Texto

Descripción generada automáticamente con confianza media

Figura . Envío de tabla a S3

Se revisa dentro del bucket el archivo cargado de la siguiente manera:

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Figura . Carga del archivo csv

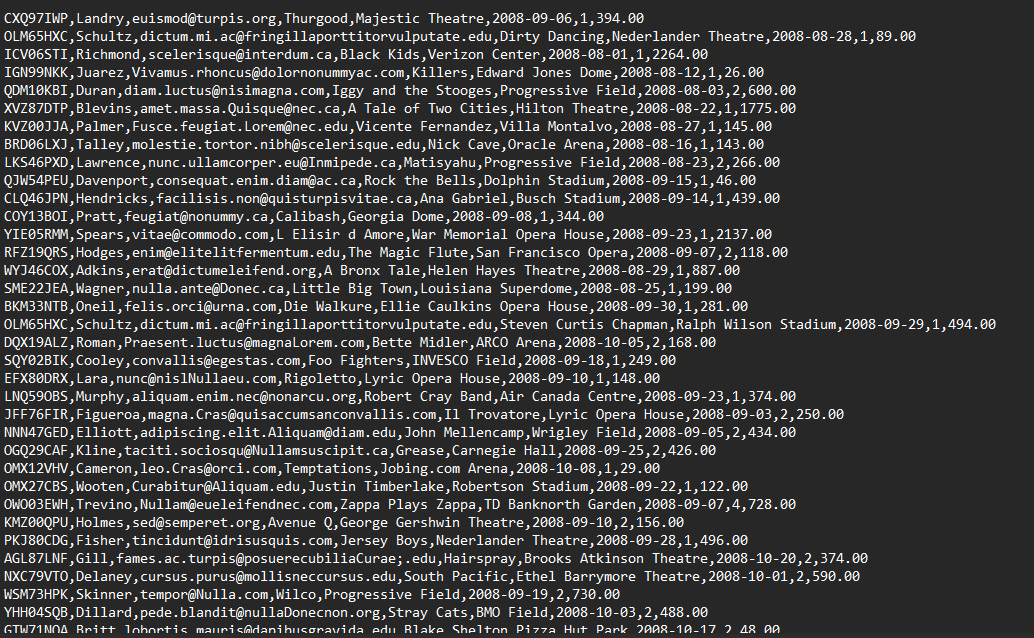


Figura . CSV generado

***6. (Opcional) Sobre la data exportada en el punto 5, y usando cualquier información adicional que desee, cree una sesión de SageMaker y realice un modelo de Forecast con cualquier técnica de su preferencia, para pronosticar las ventas para los siguientes 7 días desde el final del histórico de datos. Tenga en cuenta que la fecha de la venta se encuentra en la variable:***

Se crea una instancia de Sagemaker como se muestra en la siguiente figura:

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Figura . Sagemaker

Se implementa el codigo en un jupyter dentro del entorno de sagemaker.



Figura . Sagemaker

El codigo implementado se puede encontrar dentro de la carpeta code. A continuación, se muestra la predicción.

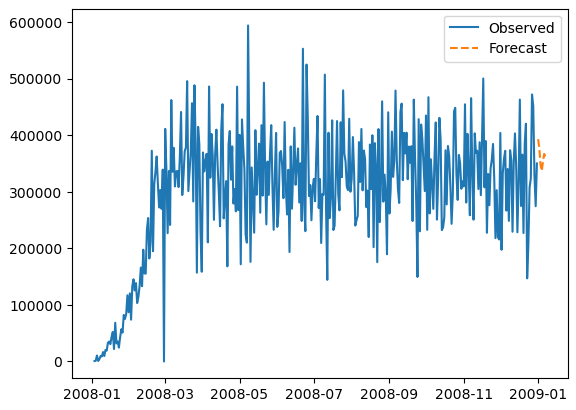


Figura . Forecast

# 5. Prueba de Azure

***1.Desplegar base de datos en SQL, con la base de pruebas Adventure Works***

Se crea la base de datos SQL en azure como se muestra en la siguiente figura, posteriormente ya se tiene una base de datos con la dataset de pruebas.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Figura . Creación base de datos

***2. Realizar un pipeline con Azure Datafactory, utilizando data flow, para realizar la carga de una base de datos, crear 5 indicadores.***

En la siguiente figura se muestra la creación del Dataflow desde Azure:

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Figura . Dataflow

Los siguientes ítems no se lograron desarrollar debido a una falla en la cuenta personal de Azure, pero los pasos siguientes si se hubiera desarrollado el punto serian:

* se selecciona "Nuevo" para crear un nuevo Data Flow. En el Data Flow, se utilizan las actividades "Source" para extraer datos de la base de datos SQL y las actividades "Sink" para cargar datos en el Data Lake. Se configuran las transformaciones necesarias entre las actividades Source y Sink utilizando las funcionalidades de transformación de Data Flow.
* se agregan actividades "Copy Data" entre las transformaciones según sea necesario. Se configuran las actividades "Copy Data" para mover datos entre la base de datos SQL y el Data Lake.
* se guarda y publica el Data Flow y el pipeline después de configurar todas las actividades y transformaciones.
* se puede ejecutar el pipeline manualmente para probar las transformaciones y asegurarse de que todo funcione correctamente.

# 6. Prueba de modelación analítica

En el transcurso del análisis y modelado de datos, se procedió a cargar el conjunto de datos de entrenamiento desde el archivo 'train.csv'. Se identificaron varias columnas que no solo eran consideradas irrelevantes para la predicción de la variable objetivo FRAUDE, sino que también presentaban un alto número de valores nulos. La decisión de eliminar estas columnas se basó en la premisa de que su inclusión no aportaría significativamente al modelo y podría introducir ruido innecesario en el análisis. Después de remover filas con valores nulos y eliminar estas columnas, el conjunto de datos de entrenamiento no experimentó una reducción significativa en su tamaño, si no se hubiesen eliminado las columnas irrelevantes el conjunto de datos sufriría una drástica disminución de casi la mitad de los datos. Este proceso de limpieza y simplificación de características no solo contribuyó a mejorar la eficiencia computacional, sino que también permitió centrarse en las variables más relevantes para el objetivo de predecir transacciones fraudulentas.

Posteriormente, se dividió el conjunto de datos en conjuntos de entrenamiento y prueba, y se ajustaron modelos de Random Forest y SVM. La optimización de hiperparámetros se realizó mediante las técnicas de búsqueda GridSearchCV y RandomizedSearchCV. Se evaluó el rendimiento de los modelos, destacándose el Random Forest por su alta precisión, recall y f1-score. En el siguiente paso, se aplicó el modelo seleccionado a un conjunto de datos de prueba previamente cargado. Se garantizó la consistencia en el orden de las columnas y se aplicó la codificación one-hot, ajustando la columna 'COD\_PAIS' según el mapeo ISO previamente creado. Se generaron probabilidades en lugar de predicciones binarias para ofrecer flexibilidad en la interpretación de los resultados. Los resultados se guardaron en un archivo CSV, incluyendo las probabilidades de fraude. Se optó por el modelo de Random Forest con hiperparámetros optimizados mediante GridSearchCV debido a su robusto rendimiento, especialmente en la minimización de falsos positivos. Este enfoque se seleccionó con la consideración de la crítica asociada a acusar falsamente de fraude, priorizando así la precisión en la detección de transacciones legítimas. Los resultados detallados del rendimiento de los modelos se presentan en el informe, y el archivo CSV resultante se generó para su análisis y revisión adicionales.

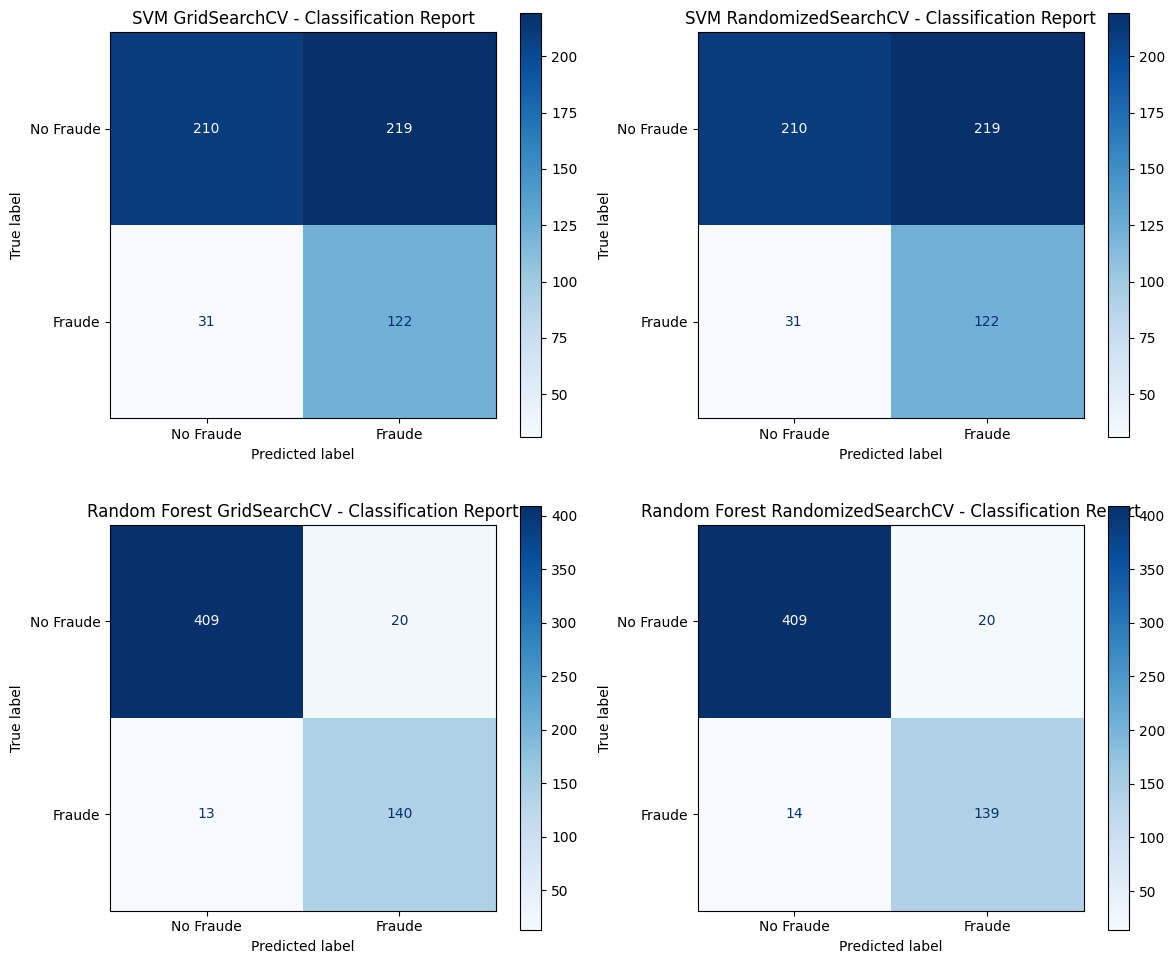


Figura . Matriz de confusión con modelos entrenados

# 7. Arquitectura

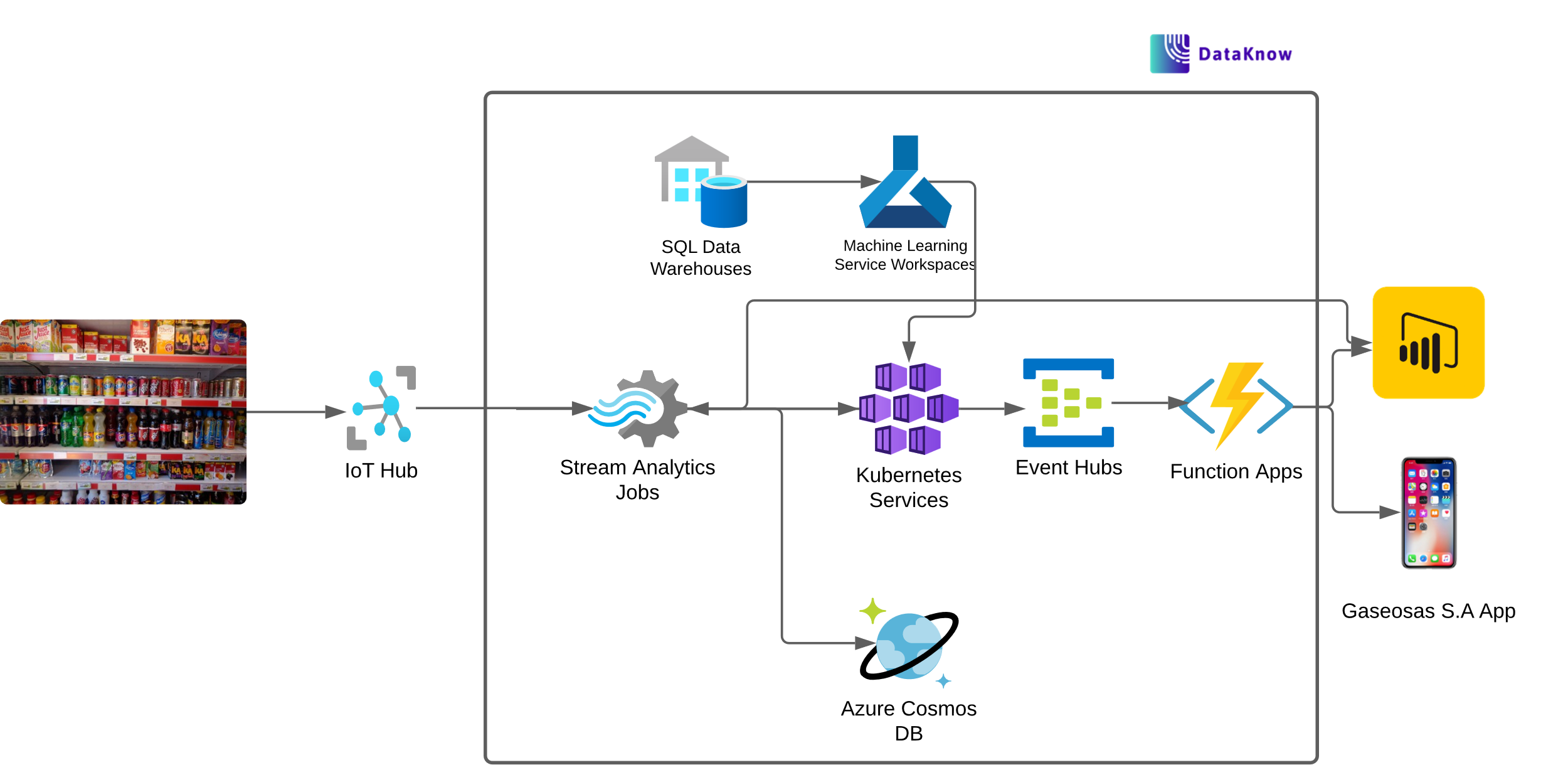


Figura . Arquitectura Solución.

Para el diseño de la arquitectura de la solución analítica en Azure que cumpla con los requerimientos de Gaseosas SA, se tienen la siguiente explicación.

1. **Fuentes de Datos:**
   * **Datos de Ventas:** Almacenados en Azure SQL Data Warehouse para la base de datos histórica debido al gran volumen de datos y la capacidad de escalamiento.
   * **Datos de Dispositivos de Puntos de Venta:** Transmitidos en tiempo real a través de Azure Stream Analytics, que procesa y enruta los datos a diferentes destinos según sea necesario.
2. **Procesamiento de Datos:**
   * Utilizar Azure Databricks para procesar grandes volúmenes de datos de ventas y realizar análisis avanzados para el entrenamiento de modelos predictivos.
3. **Almacenamiento:**
   * **Almacenamiento Histórico:** Azure SQL Data Warehouse para almacenar la base de datos histórica de ventas debido a su capacidad de escalamiento horizontal y capacidades de procesamiento analítico.
   * **Almacenamiento en Tiempo Real:** Azure Cosmos DB para almacenar datos en tiempo real, ya que ofrece baja latencia y es altamente escalable.
4. **Modelado y Entrenamiento:**
   * **Entorno de Modelado:** Azure Machine Learning para desarrollar, entrenar y desplegar modelos predictivos.
   * **Modelos Analíticos:** Los modelos entrenados se implementan en Azure Kubernetes Service para permitir la escalabilidad y el despliegue eficiente.
5. **Gobierno de Datos y Modelos:**
   * **Gestión de Metadatos:** Azure Purview para gestionar y descubrir metadatos en todo el entorno analítico.
   * **Gobierno de Modelos:** Azure Machine Learning registra y rastrea los modelos, y Azure Policy define políticas para la implementación y monitorización.
   * **Seguridad de Datos:** Azure Active Directory para la gestión de identidades y roles, y Azure Key Vault para la gestión segura de claves y secretos.
6. **Análisis en Tiempo Real:**
   * Azure Stream Analytics para el análisis en tiempo real de datos de dispositivos de puntos de venta y la comparación con patrones históricos.
7. **Alertas y Acciones Preventivas:**
   * Azure Logic Apps para la automatización de procesos y la generación de alertas en caso de posibles desabastecimientos.
8. **Visualización de Datos:**
   * Utilizar Power BI para crear paneles de control interactivos que permitan a los usuarios explorar los datos y patrones identificados por el sistema analítico.

En cuanto al gobierno de datos tenemos lo siguiente:

1. **Gestión de Metadatos:**
   * Azure Purview para catalogar, descubrir y gestionar metadatos en todos los servicios de datos.
2. **Gobierno de Modelos:**
   * Azure Machine Learning para rastrear, versionar y gestionar el ciclo de vida de los modelos predictivos.
3. **Seguridad de Datos:**
   * Azure Active Directory para gestionar la autenticación y autorización, y Azure Key Vault para la gestión segura de claves y secretos.
4. **Perfiles de Usuario:**
   * Definir roles y permisos en Azure Active Directory para garantizar el acceso adecuado según los perfiles de usuario.
5. **Formación y Concienciación:**
   * Desarrollar programas de formación para los usuarios sobre el uso ético de datos y modelos, con recursos disponibles en Azure Learning Paths.

Esta arquitectura aprovecha servicios específicos de Azure para garantizar la eficiencia, la escalabilidad y la integración fluida entre los diferentes componentes. La gestión de datos y modelos se logra a través de herramientas y servicios dedicados en Azure, proporcionando un gobierno robusto y centralizado.

# 8. Prueba BI

Para el desarrollo de este punto se usó PowerBI, por lo tanto, el primer paso a seguir fue subir las tablas a dicho Software y conectarlas generando una base de datos relacional, esto del siguiente modo:

Interfaz de usuario gráfica, Diagrama

Descripción generada automáticamente

Figura . Unión Tablas

Posteriormente se calcula el número total de usuarios mediante el siguiente Código en DAX, se cuentan todos los valores “user\_uuid” de la tabla CL ORDERS, pero sin contar los repetidos, cada registro solo se cuenta una vez.



Figura . Dax 1

Ahora para calcular el GMV cuadrado.



Figura . Dax 2

Para la cantidad de artículos vendidos.

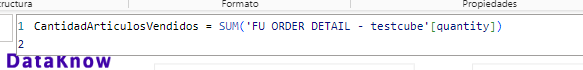


Figura . Dax 3

Para la evolución de las ganancias a lo largo del tiempo se hizo sin DAX de la siguiente manera:

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Figura . Dax 4

Del mismo modo se generaron varios reporte y análisis, estos muestran en la presentación de Power point presente en la misma carpeta donde se encuentra este documento, para más información dirigirse al PowerBI o la presentación.

El dashboard creado a partir de las tablas proporcionadas ofrece una visión integral y detallada de la información clave relacionada con las órdenes y ventas. Este tablero de control, diseñado para facilitar el análisis de datos, incorpora seis filtros esenciales que permiten una personalización completa: País-Ciudad, Zona-Dirección, Barrio, Manager ID, Categoría de Producto y Nombre del Producto.

En la sección de tarjetas, se presentan métricas fundamentales, como el número de usuarios, el GMV al cuadrado, las ganancias totales y el total de artículos vendidos. Estas tarjetas proporcionan una instantánea rápida de los indicadores clave del rendimiento del negocio.

El dashboard incluye un pie chart que visualiza las ganancias por ciudad, lo que facilita la identificación de patrones geográficos en el rendimiento financiero. También se destacan el valor promedio de la orden y el número de productos por orden, brindando información sobre los hábitos de compra de los usuarios.

En la sección de análisis de costos, una tabla presenta las categorías que generan mayores costos. Esto ofrece insights cruciales para la toma de decisiones estratégicas. Las gráficas de barras comparativas muestran las ganancias de diferentes categorías y productos, así como la frecuencia de compra asociada a cada categoría y nombre de producto.

Además, se presentan pie charts que desglosan las ganancias según el canal de creación, el nombre del cupón, el tipo de propiedad y el estado de la orden. Estos gráficos proporcionan una comprensión detallada de las fuentes y condiciones de las transacciones.

En la sección de detalles de usuario, una tabla muestra información específica, como el user ID, dirección, barrio, ganancias generadas por el usuario, tipo de propiedad y ciudad. Acompañando esta tabla, un mapa visualiza las direcciones proporcionadas, facilitando la comprensión geoespacial de la distribución de usuarios.

Finalmente, las gráficas de líneas ofrecen una perspectiva temporal de las ganancias y los pagos, permitiendo la identificación de tendencias a lo largo del tiempo. Este enfoque temporal es esencial para discernir si las fluctuaciones en las ganancias se deben a cambios en el volumen de compras o a la variación en la rentabilidad. En resumen, el dashboard proporciona una herramienta integral para analizar datos, facilitando la identificación de patrones, la toma de decisiones informada y la comprensión profunda del rendimiento del negocio. A partir de este completo dashboard, se pueden derivar varios análisis que ofrecen una visión profunda de las operaciones comerciales y el comportamiento de los clientes.

Algunos de los análisis clave incluyen:

* Análisis Geográfico:Identificación de las ubicaciones geográficas más rentables y la distribución de ganancias por ciudad. Evaluación de la frecuencia de compra y las preferencias de producto en diferentes zonas, barrios o países-ciudades.
* Análisis de Producto: Identificación de las categorías y productos más rentables, lo que permite ajustar estrategias de marketing y gestión de inventario. Comparación de la frecuencia de compra por categoría y por nombre de producto para comprender las preferencias del cliente.
* Análisis de Usuarios: Evaluación del rendimiento individual de los usuarios mediante la tabla detallada, que muestra sus contribuciones en términos de ganancias. Exploración de la relación entre la ubicación del usuario, el tipo de propiedad y el comportamiento de compra.
* Análisis de Órdenes y Transacciones: Observación de métricas clave, como el valor promedio de una orden y el número de productos por orden, para entender el comportamiento de compra típico. Seguimiento del estado de las órdenes (confirmado, cancelado, enviado, etc.) para evaluar el rendimiento operativo y la satisfacción del cliente.
* Análisis de Canales y Cupones: Evaluación del impacto de diferentes canales de creación en las ganancias. Identificación de la efectividad de los cupones en términos de generación de ingresos.
* Análisis Temporal: Posible análisis de la evolución de las ventas a lo largo del tiempo utilizando las tarjetas y visualizaciones temporales. Identificación de patrones estacionales o tendencias de compra a lo largo del tiempo.
* Análisis de Costos y Rentabilidad: Determinación de las categorías que más contribuyen a las ganancias y los productos más rentables. Evaluación del impacto financiero de las estrategias de descuento o promoción.

Estos análisis proporcionan una visión integral de la salud comercial, permitiendo a los responsables de la toma de decisiones identificar oportunidades de mejora, ajustar estrategias y optimizar el rendimiento general del negocio.

Gráfico

Descripción generada automáticamente

Figura . Dashboard Creado