**Informe de Implementación**

**Descripción del Flujo de Trabajo**

Para resolver el desafío de predecir la probabilidad de compra, se implementó un flujo de trabajo de ML distribuido utilizando Apache Spark y su librería MLlib. El proceso fue el siguiente:

1. **Preparación de Datos:** Se partió de un conjunto de datos con información de navegación y compras de los clientes. Se definieron las variables predictoras (características) y la variable objetivo (la etiqueta label que indica si el cliente comprará o no).
2. **Preprocesamiento y Vectorización:** Se construyó un Pipeline de MLlib para automatizar la transformación de los datos. Este pipeline convierte las variables categóricas en formato numérico con StringIndexer y OneHotEncoder, y luego consolida todas las características en una única columna vectorial (VectorAssembler), que es el formato requerido por los algoritmos de MLlib.
3. **Selección de Modelo y Ajuste:** Se seleccionó el algoritmo Random Forest por su robustez. Para encontrar la configuración óptima del modelo, se utilizó un CrossValidator. Este proceso prueba automáticamente diferentes combinaciones de hiperparámetros (como el número de árboles y su profundidad) y selecciona la que ofrece el mejor rendimiento.
4. **Entrenamiento y Evaluación:** El pipeline completo, incluyendo el CrossValidator, se entrenó sobre un conjunto de datos de entrenamiento (80% del total). Posteriormente, el rendimiento del mejor modelo encontrado se evaluó sobre un conjunto de prueba (20% restante) para obtener una estimación imparcial de su desempeño. Pero al tener tan pocos datos, pueden haber errores de estimación debido a que el conjunto de datos no es significativo.

**Resultados y Métricas del Modelo**

El rendimiento del modelo final en el conjunto de datos de prueba fue medido con las siguientes métricas, adecuadas para problemas de clasificación binaria:

* **Área bajo la curva ROC (AUC-ROC):** Esta métrica indica la capacidad del modelo para distinguir entre las dos clases (clientes que comprarán y los que no). Un valor cercano a 1.0 representa un modelo excelente.
* **Área bajo la curva de Precisión-Recall (AUC-PR):** Esta métrica es especialmente útil cuando las clases están desbalanceadas (por ejemplo, si hay muchos más no compradores que compradores). Evalúa la habilidad del modelo para identificar correctamente a los clientes que sí comprarán.

**Ambos valores tuvieron de valor “1.0”**

**Recomendaciones y Futuras Mejoras**

1. **Ingeniería de Características:** El mayor impacto en el rendimiento de un modelo suele venir de la calidad de sus características. Se recomienda crear nuevas variables más sofisticadas, como "frecuencia de compra”,"tiempo promedio entre sesiones" o "categorías de productos favoritas".
2. **Evaluación de Otros Modelos:** Se sugiere comparar el rendimiento del Random Forest con otros algoritmos potentes como **Gradient-Boosted Trees** que a menudo pueden ofrecer una mayor precisión.
3. **Escalado y Automatización:** Para un entorno de producción, se debe ejecutar el flujo sobre un clúster de Spark adecuadamente dimensionado (ej. en AWS EMR, Google Dataproc o Databricks). Además, se recomienda orquestar la ejecución periódica del entrenamiento del modelo utilizando herramientas como Apache Airflow.