

# PROYECTO AURELION EN GOOGLE IA STUDIO CON

ACTUAL 21/11/2025

Proyecto Aurelion Machine Learning y Modelo de Entrenamiento

## Contexto del Proyecto

[Descargar README.md](#)

Documentación oficial para la defensa del Sprint 3. Este archivo README.md es tu portada en Visual Studio Code.

### ⚠ Nota Importante: Corrección de Data Leakage

¿Por qué eliminamos 'frequency' del entrenamiento?

Definimos que "Fiel" = "Frecuencia ≥ 2". Si usamos la frecuencia como dato de entrada, el modelo "hace trampa" porque ya sabe la respuesta. Para el Sprint 3, hemos removido esta variable de las **Features (X)** para obligar al modelo a predecir basándose en otros patrones (Recencia, Gasto, etc.).

## ☰ Guía de Defensa (Los 10 Puntos)

#1 Objetivo del Modelo

#2 Descripción Dataset (X e y)

#3 Preprocesamiento

#4 División Train/Test

#5 Selección Algoritmo

#6 Entrenamiento .fit()

#7 Predicciones .predict()

#8 Métricas Evaluación

#9 Modelo Final implementado

#10 Gráficos y Conclusiones

Tip para VSC: Descarga el README.md arriba. Contiene esta tabla mapeada a los archivos reales para que no te pierdas durante la presentación.

## grid Variables del Modelo (X e y)

ROL	VARIABLE	ESTATUS
y (Target)	is_fidelizado	Objetivo (1=Fiel, 0=Ocasional)
X (Excluido)	frequency	ELIMINADO para evitar Data Leakage (Trampa).
X (Feature)	recency_days	Input válido (Días sin venir)
X (Feature)	monetary_log	Input válido (Gasto Total Normalizado)

## Ingeniería de Features (RFM)

Análisis visual de las variables predictoras

↗ Pantalla Completa



### Resumen Ejecutivo

Generado por Gemini 2.5 Flash



Regenerar Reporte

Buenos días, Directiva. Permítanme presentarles un resumen ejecutivo sobre el estado actual de nuestra cartera de clientes en Aurelion Retail, fundamentado en nuestros datos de RFM más recientes.

#### Estatus de Cartera

Nuestra cartera de clientes actual se compone de 67 individuos, de los cuales 37 son clientes leales, formando una base estable para nuestras operaciones. Contamos también con 30 clientes ocasionales, lo que representa un segmento con potencial de desarrollo en su recurrencia. El ticket promedio de compra en el periodo analizado es de \$ 39.573,39, ofreciendo una métrica clave sobre el valor transaccional de nuestra clientela.

#### Hallazgo Clave

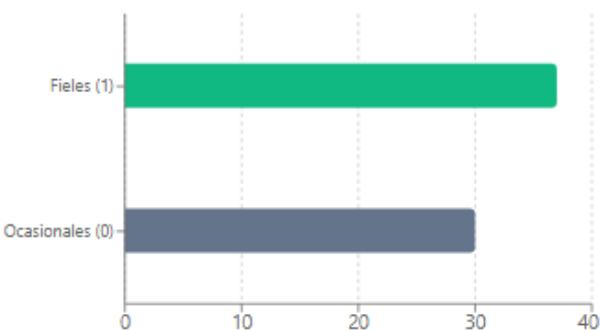
El equilibrio entre clientes leales y ocasionales subraya una oportunidad estratégica para impulsar la fidelización y el valor a largo plazo de una parte significativa de nuestra base de usuarios.

#### Estrategia Sugerida

Para fortalecer la retención y aumentar la lealtad, proponemos implementar un programa de comunicación segmentado dirigido específicamente a nuestros clientes ocasionales. Este programa debería incluir incentivos personalizados y promociones exclusivas diseñadas para fomentar su segunda compra y establecer un patrón de recurrencia, buscando así moverlos progresivamente hacia el segmento de clientes leales.

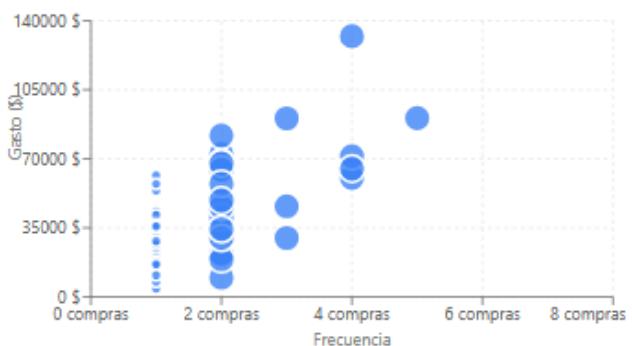
### Distribución del Target

Proporción de clientes Fieles (1) vs Ocasionales (0)



### Patrón: Frecuencia vs Gasto

Relación entre veces que compran y cuánto gastan



### Dataset Procesado (RFM)

67 registros listos para el modelo

Descargar CSV

Ocultar Tabla

ID_CLIENTE	RECENCY_DAYS	FREQUENCY	MONETARY	CATEGORIA_PREFERIDA	IS_FIDELIZADO
1	94	2	\$ 72.448,00	ALIMENTOS	FIEL (1)
2	77	1	\$ 22.150,00	ALIMENTOS	OCASIONAL (0)
3	133	1	\$ 33.310,00	ALIMENTOS	OCASIONAL (0)
5	3	4	\$ 132.158,00	ALIMENTOS	FIEL (1)
6	136	2	\$ 48.878,00	ALIMENTOS	FIEL (1)
8	171	1	\$ 61.503,00	ALIMENTOS	OCASIONAL (0)
9	95	2	\$ 41.840,00	LIMPIEZA	FIEL (1)
10	9	1	\$ 25.860,00	LIMPIEZA	OCASIONAL (0)
12	60	2	\$ 49.518,00	ALIMENTOS	FIEL (1)
13	59	1	\$ 13.188,00	ALIMENTOS	OCASIONAL (0)

## Simulador de Entrenamiento ML

Visualiza cómo el algoritmo aprende a separar clientes Fieles de Ocasionales.

ALGORITMO  
Regresión Logística

TASA DE APRENDIZAJE  
**0.01**

ITERACIONES  
100



### ¿Listo para entrenar?

Al hacer clic, el modelo leerá tus 120 clientes y buscará la regla matemática perfecta usando la Tasa de Aprendizaje de 0.01.

Ejecutar Entrenamiento

## Simulador de Entrenamiento ML

Visualiza cómo el algoritmo aprende a separar clientes Fieles de Ocasionales.

**Entrenamiento Finalizado**

Descargar Informe Completo (.md)

ACCURACY GLOBAL

**100%**

(Modelo Perfecto)

PRECISION (FIELES)

**1.00**

Sin errores en positivos

RECALL (FIELES)

**1.00**

Encontró a todos

TP

FP

FN

TN

Matriz de Confusión

Guía Rápida: ¿Cómo leer la Matriz de Confusión?

TP Verdadero Positivo: La IA predijo "Fiel" y el cliente Sí compró 2+ veces. ¡Éxito!

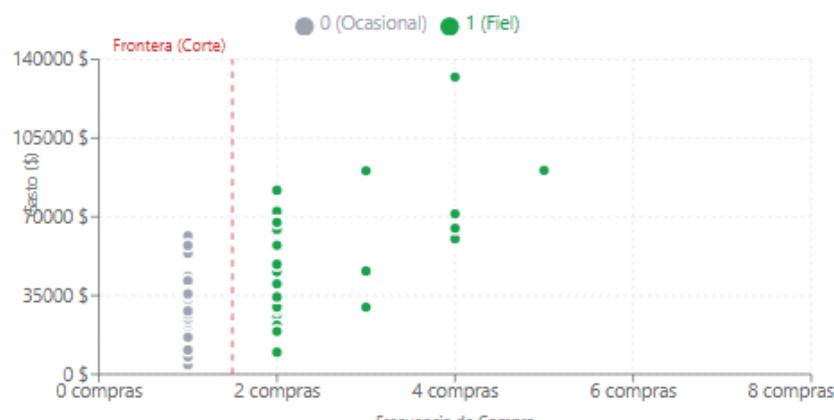
FP Falso Positivo: Predijo "Fiel" pero era Ocasional. (Gastaste marketing por error).

TN Verdadero Negativo: La IA predijo "Ocasional" y el cliente Sí fue de 1 vez. ¡Ahorro!

FN Falso Negativo: Predijo "Ocasional" pero era Fiel. (Perdiste un cliente VIP).

## Frontera de Decisión Visual

Regla: Freq ≥ 2



\* Gráfico estilo Seaborn: Puntos con borde blanco, Grilla gris suave.

## GLOSARIO DE DEFENSA

### 💡 ¿Qué son las Iteraciones?

Es cuántas veces lee el modelo el libro entero (dataset). **100 Iteraciones** = lo leyó 100 veces.  
Si son muchas, *memoriza (overfitting)*; si son pocas, *no aprende*.

### Σ ¿Por qué la curva S?

Porque predecimos **Probabilidad** (0 a 1). La "S" (Síguoide) aplasta valores locos (ej: 1.5) para que siempre den un % válido.

### ▷ ¿Qué es el Optimizador?

Es el método para encontrar la solución. **liblinear** es ideal para datasets pequeños.

## 💡 CONCLUSIÓN EJECUTIVA

"Matemáticamente, la lealtad se decide en la **2da compra**. El monto gastado es irrelevante para la clasificación."

## > Llévalo a Producción

Esta simulación fue visual. Descarga el código Python real (Scikit-Learn) para ejecutarlo en tu VS Code y presentarlo.

[entrenamiento\\_modelo\\_aurelion.py](#)

## Proyecto Aurelion Machine Learning y Modelo de Entrenamiento

### Tu Código Python para el Sprint 3

Este código ahora incluye la generación de gráficos con Matplotlib y Seaborn para que los visualices directamente en tu Notebook de VS Code.

• [script\\_completo\\_sprint3.py](#)

Copiar Código

Descargar.ipynb

```
# =====#
# PROYECTO AURELION - SPRINT 3: CLASIFICACIÓN DE FIDELIDAD (MACHINE LEARNING)
# VERSIÓN: 1.6 (Corrección Data Leakage - Frecuencia Eliminada)
# =====#

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, confusion_matrix
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline

# Configuración visual para gráficos profesionales
sns.set(style="whitegrid")
plt.rcParams['figure.figsize'] = (10, 6)

print("🛠 Iniciando Script de Entrenamiento Aurelion...")

# 1. CARGA DE DATOS
# -----
# Intentamos cargar el archivo CSV generado por la App.
# Asegúrate de que 'master_rfm_aurelion_limpio.csv' esté en la misma carpeta que este script.

# Algunos datos están siendo limpiados aquí
```