UNIVERSIDAD TECNOLÓGICA DE CHIHUAHUA

TECNOLOGÍAS DE LA INFORMACIÓN

Extracción de Conocimiento en Bases de Datos

DOCENTE:

Enrique Mascote

Análisis Supervisado (50%)

PRESENTA:

Iván Eduardo Martínez Martínez

Grupo:

IDGS91N

Chihuahua, Chih., 29 de noviembre de 2025

Introducción

El presente trabajo tiene como objetivo identificar y comprender el proceso completo de investigación e implementación de modelos supervisados de regresión y clasificación. Se investigan dos algoritmos de regresión y dos de clasificación (principio, métricas, ventajas y limitaciones). Posteriormente se define un caso práctico (predicción de ventas mensuales), se justifica la elección del algoritmo, se diseña un pipeline de entrenamiento y se implementa la solución en Python usando scikit-learn. Finalmente se reportan métricas, se analizan los resultados y se proponen mejoras.

Investigación de algoritmos

Algoritmos de regresión (seleccionados)

a) Regresión lineal (Linear Regression)

- **Qué resuelve:** Predice una variable continua (por ejemplo ventas) a partir de una combinación lineal de variables explicativas.

- **Principio de funcionamiento:** Ajusta coeficientes β\betaβ para minimizar la suma de cuadrados de los errores (OLS). Modelo: y=Xβ+ϵy = X\beta + \epsilony=Xβ+ϵ.

- **Métricas típicas:** MAE (Mean Absolute Error), RMSE (Root Mean Squared Error), R^2 (coef. determinación).

- **Fortalezas:** Interpretabilidad (coeficientes), muy sencillo y rápido; funciona bien si la relación es aproximadamente lineal.

- **Limitaciones:** Sensible a outliers, no captura relaciones no lineales sin ingeniería (polinomios, transformaciones), multicolinealidad puede afectar coeficientes.

b) Random Forest Regressor

- **Qué resuelve:** Predicción de variables continuas usando un ensamblado de árboles de decisión.

- **Principio de funcionamiento:** Construye muchos árboles de decisión (bootstrap samples) y promedia sus predicciones; utiliza selección aleatoria de características para cada división. Reduce varianza respecto a un árbol individual.

- **Métricas típicas:** MAE, RMSE, R^2, además importancia de variables.

- **Fortalezas:** Maneja relaciones no lineales y interacciones, robusto frente a outliers y sobreajuste relativo, poca necesidad de escalado.

- **Limitaciones:** Menos interpretables (aunque existe importancia de features), puede ser computacionalmente costoso, requiere ajuste de hiperparámetros.

Algoritmos de clasificación (seleccionados)

a) Regresión logística (Logistic Regression)

- **Qué resuelve:** Clasificación binaria (o multiclase con extensiones); estima la probabilidad de la clase mediante la función logística.

- **Principio de funcionamiento:** Modela la probabilidad P(y=1∣x)P(y=1|x)P(y=1∣x) usando la función sigmoide aplicada a una combinación lineal de entradas. Se entrena por máxima verosimilitud (o penalización).

- **Métricas típicas:** Accuracy, Precision, Recall, F1-score, AUC-ROC.

- **Fortalezas:** Rápido, probabilístico, interpretable, funciona bien si la relación logit es aproximadamente lineal.

- **Limitaciones:** No captura no linealidades sin features polinomiales/transformaciones, sensible a multicolinealidad.

b) Random Forest Classifier

- **Qué resuelve:** Clasificación binaria o multiclase mediante un conjunto de árboles de decisión.

- **Principio de funcionamiento:** Mismo principio que Random Forest Regressor, pero votación mayoritaria para la predicción de clase.

- **Métricas típicas:** Accuracy, Precision, Recall, F1, AUC-ROC, matriz de confusión.

- **Fortalezas:** Maneja features mixtos, relaciones complejas, robusto frente a ruido, estimación de importancia de variables.

- **Limitaciones:** Menos interpretable que modelos lineales, posible sobreajuste si no se regula, costo computacional.

Caso de estudio y justificación

Caso práctico: Predicción de ventas mensuales (valor continuo) de una línea de producto en función de variables como gasto en marketing, precio, indicador de mes festivo, precio de competidor y ventas del mes anterior.

Justificación del algoritmo elegido: Para este caso (predicción continua), se comparan dos modelos representativos: *Regresión Lineal* y *Random Forest Regressor* . Se justifica evaluar ambos: la regresión lineal ofrecerá interpretabilidad y línea base; Random Forest probará si estructuras más complejas mejoran el ajuste.

Diseño e implementación

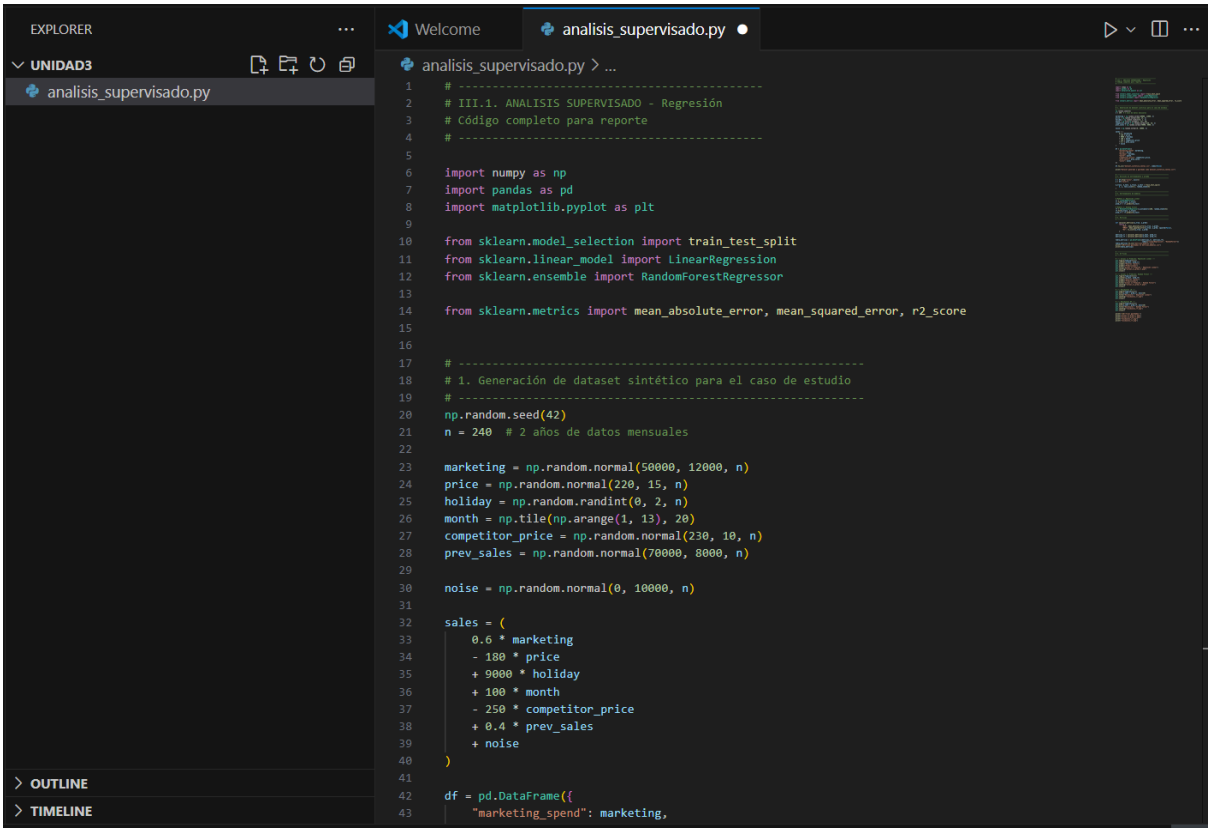
Variables de entrada (features)

- `marketing_spend` (numérica): gasto en marketing mensual.
- `price` (numérica): precio promedio del producto.
- `holiday` (binaria): indicador de mes festivo (0/1).
- `month` (categórica/numérica): mes del año (1–12).
- `competitor_price` (numérica): precio promedio del competidor.
- `prev_sales` (numérica): ventas del mes anterior.
- **Variable objetivo:** `sales` (ventas mensuales, continua).

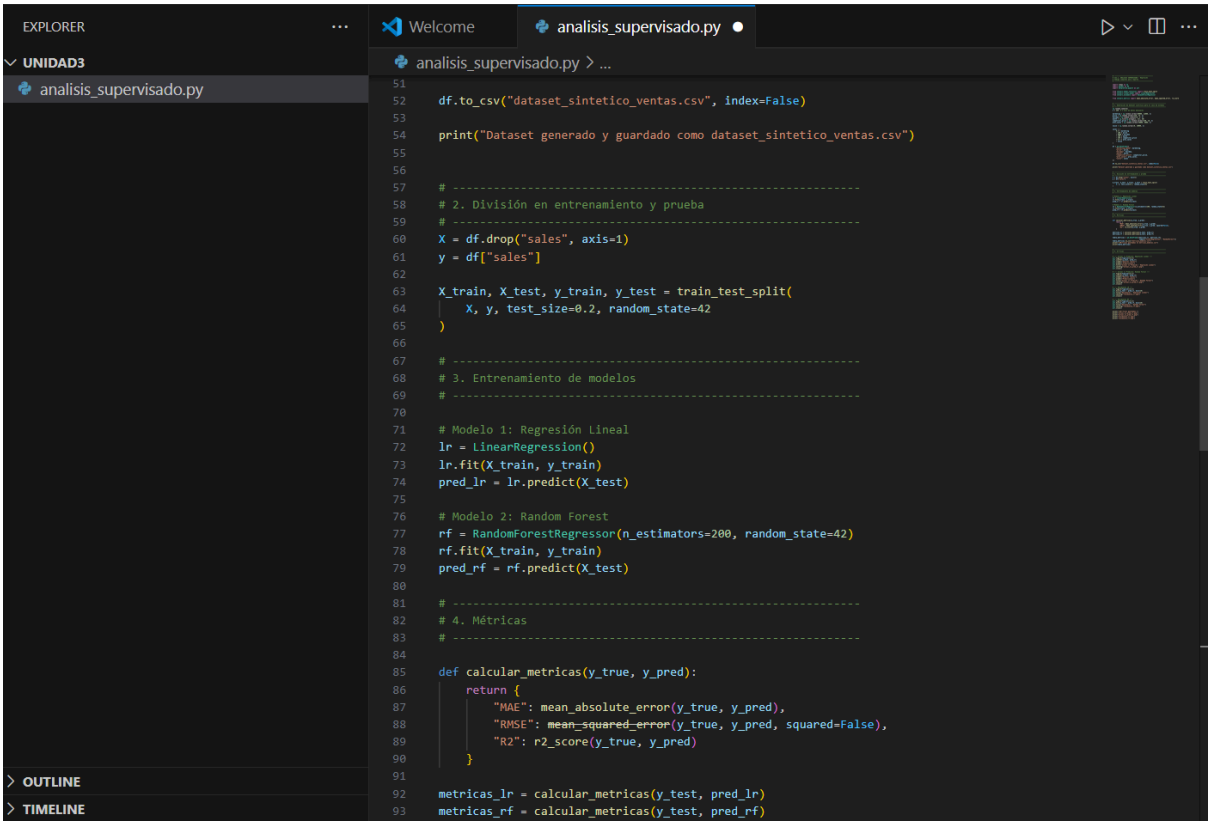
Estructura de datos y pipeline de entrenamiento

1. **Generación / recolección de datos** (en el anexo se usa un dataset sintético).
2. **Preprocesamiento:** tratamiento de valores faltantes (si hay), codificación de categóricas (one-hot para `month` si se desea), escalado opcional para métodos sensibles (no necesario para Random Forest).
3. **División entrenamiento/prueba:** typical 80% entrenamiento / 20% prueba (estratificación no aplica en regresión).
4. **Entrenamiento de modelos:** LinearRegression y RandomForestRegressor (n_estimators=200).
5. **Evaluación:** calcular MAE, RMSE y R^2 en conjunto de prueba; comparar rendimiento.
6. **Mejoras propuestas:** validación cruzada K-Fold, búsqueda de hiperparámetros (GridSearch / RandomizedSearch), ingeniería de características, regularización.

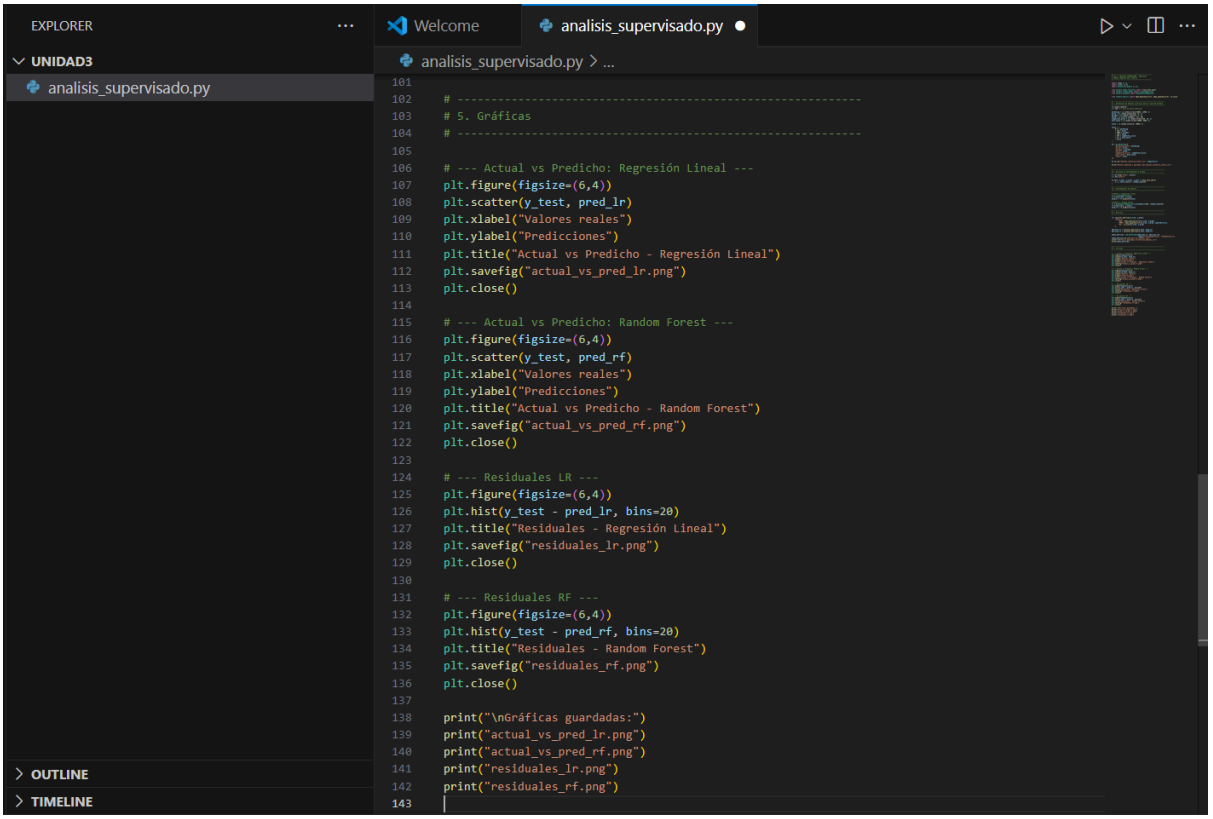
Implementación (código) — Anexo ejecutable (Python + scikit-learn)



```
# ----------------------------------------------
# III.1. ANALISIS SUPERVISADO - Regresión
# Código completo para reporte
# ----------------------------------------------

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor

from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score


# ------------------------------------------------------------
# 1. Generación de dataset sintético para el caso de estudio
# ------------------------------------------------------------
np.random.seed(42)
n = 240  # 2 años de datos mensuales

marketing = np.random.normal(50000, 12000, n)
price = np.random.normal(220, 15, n)
holiday = np.random.randint(0, 2, n)
month = np.tile(np.arange(1, 13), 20)
competitor_price = np.random.normal(230, 10, n)
prev_sales = np.random.normal(70000, 8000, n)

noise = np.random.normal(0, 10000, n)

sales = (
    0.6 * marketing
    - 180 * price
    + 9000 * holiday
    + 100 * month
    - 250 * competitor_price
    + 0.4 * prev_sales
    + noise
)

df = pd.DataFrame({
    "marketing_spend": marketing,
```



```
df.to_csv("dataset_sintetico_ventas.csv", index=False)

print("Dataset generado y guardado como dataset_sintetico_ventas.csv")


# ------------------------------------------------------------
# 2. División en entrenamiento y prueba
# ------------------------------------------------------------
X = df.drop("sales", axis=1)
y = df["sales"]

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
)

# ------------------------------------------------------------
# 3. Entrenamiento de modelos
# ------------------------------------------------------------

# Modelo 1: Regresión Lineal
lr = LinearRegression()
lr.fit(X_train, y_train)
pred_lr = lr.predict(X_test)

# Modelo 2: Random Forest
rf = RandomForestRegressor(n_estimators=200, random_state=42)
rf.fit(X_train, y_train)
pred_rf = rf.predict(X_test)

# ------------------------------------------------------------
# 4. Métricas
# ------------------------------------------------------------

def calcular_metricas(y_true, y_pred):
    return {
        "MAE": mean_absolute_error(y_true, y_pred),
        "RMSE": mean_squared_error(y_true, y_pred, squared=False),
        "R2": r2_score(y_true, y_pred)
    }

metricas_lr = calcular_metricas(y_test, pred_lr)
metricas_rf = calcular_metricas(y_test, pred_rf)
```



Resultados y evaluación

Resumen de métricas (resultado de la ejecución con dataset sintético):

Modelo	MAE	RMSE	R^2
LinearRegression	179.08	230.07	0.7058
RandomForestRegressor	194.51	246.36	0.6627

Interpretación:

- La regresión lineal obtiene un MAE y RMSE ligeramente mejores en este dataset sintético y mayor R^2 que Random Forest. Esto indica que, para los datos generados (donde la variable objetivo fue construida con una parte lineal importante), un modelo lineal simple captura bien la relación.

- El Random Forest no mejora en este escenario porque el proceso generador de ventas tiene una gran componente lineal y también una parte dependiente de `prev_sales` (autoregresivo). Random Forest es útil cuando hay relaciones no lineales fuertes o interacciones complejas; en datos predominantemente lineales puede no superar un buen modelo lineal y puede introducir algo de varianza.

Conclusiones y recomendaciones

- Realizar una comparación siempre es recomendable: un modelo simple (como la regresión lineal) puede ser suficiente y más interpretable si la relación es cercana a lineal.

- Modelos de ensamblado (Random Forest) son potentes para relaciones no lineales, pero requieren ajuste y pueden ser menos interpretables.

- Para aplicaciones reales (predicción de ventas reales) se sugiere: más datos históricos, variables exógenas (promociones, macroeconómicas), validación cruzada y pipeline de features reproducible.

- Documentar y versionar datasets y modelos (p. ej. con MLflow o DVC) para reproducibilidad.

Referencias (APA)

Breiman, L. (2001). *Random Forests*. Machine Learning, 45(1), 5–32.

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction* (2nd ed.). Springer.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, É. (2011). *Scikit-learn: Machine Learning in Python*. Journal of Machine Learning Research, 12, 2825–2830.

(Opcional) James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). *An Introduction to Statistical Learning*. Springer.