UNIVERSIDAD TECNOLÓGICA DE CHIHUAHUA

Tecnologías de la información

Extracción de Conocimiento en Bases de Datos

III.1. Análisis Supervisado (50%)

Docente

ING. LUIS ENRIQUE MASCOTE CANO

Alumno

Erick Eduardo Maffiodo Delgado

IDGS 91N

Domingo, 30 de Noviembre del 2025

Índice

1. Introducción

El análisis supervisado abarca técnicas de regresión (cuando la variable objetivo es numérica continua) y clasificación (cuando la salida es categórica). En este informe se investigan distintos modelos de cada tipo, describiendo su objetivo, funcionamiento, métricas de evaluación comunes, fortalezas y limitaciones. Posteriormente, se desarrolla un caso práctico de predicción de ventas, justificando la selección del modelo más adecuado, detallando el diseño de la solución, su implementación con código, y evaluando los resultados obtenidos. El objetivo es comprender el proceso completo de investigar algoritmos supervisados e implementarlos para resolver un problema real, siguiendo buenas prácticas de preparación de datos, entrenamiento, evaluación y análisis de rendimiento.

2. Investigación de algoritmos

A continuación, se presentan cuatro algoritmos supervisados ampliamente usados: dos de regresión y dos de clasificación. Para cada algoritmo se explica qué tipo de problema resuelve, su principio de funcionamiento, las métricas típicas para evaluar su desempeño, así como sus principales fortalezas y limitaciones.

2.1 Algoritmos de Regresión

En los problemas de regresión, la variable objetivo es continua. Las métricas de evaluación comunes incluyen el **Error Absoluto Medio (MAE)**, el **Error Cuadrático Medio (MSE)**, su raíz (**RMSE**), y el **coeficiente de determinación (R^2)**. Un modelo de regresión ideal tendrá MAE/MSE bajos y un R^2 cercano a 1 (100% de la variabilidad explicada). A continuación se describen dos algoritmos de regresión seleccionados:

2.1.1 Regresión Lineal

¿Qué resuelve? La regresión lineal estima una relación lineal entre las variables de entrada (features) y una variable objetivo continua. Su objetivo es predecir un valor numérico desconocido a partir de valores conocidos de los predictores, asumiendo que la relación entre ellos puede aproximarse mediante una línea recta (en el caso simple) o un hiperplano en el espacio multidimensional (caso múltiple). Por ejemplo, se puede usar regresión lineal para

predecir las ventas anuales de un producto en función del presupuesto de publicidad en distintos medios.

Principio de funcionamiento: El modelo asume una ecuación de la forma y=β0+β1x1+β2x2+…+βpxp+ϵy = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_p x_p + \epsilony=β0+β1x1+β2x2+…+βpxp+ϵ, donde yyy es la variable a predecir, xix_ixi son las características y βi\beta_iβi los coeficientes a estimar. El entrenamiento consiste en encontrar los coeficientes β\betaβ que minimizan el error (generalmente mediante *mínimos cuadrados ordinarios*, que minimiza la suma de los errores al cuadrado). La *regresión lineal simple* involucra solo un predictor, mientras que la *regresión lineal múltiple* maneja varios. El resultado es una fórmula algebraica fácilmente interpretable: cada coeficiente βj\beta_jβj indica cuánto cambia yyy por un incremento unitario en xjx_jxj manteniendo las otras variables constantes.

Métricas de evaluación típicas: Se utilizan métricas de error como MAE, MSE y RMSE para cuantificar la precisión de las predicciones. Por ejemplo, el MSE calcula el promedio de los cuadrados de los errores (diferencias entre valores observados y predichos), mientras que el RMSE es su raíz cuadrada, en las mismas unidades de la variable objetivo. Además, se reporta el coeficiente de determinación R^2, que indica la proporción de la varianza de yyy explicada por el modelo. Un R^2 de 0.0 significa que el modelo no explica mejor que el promedio, mientras que un R^2 de 0.90 indicaría que ~90% de la variabilidad de yyy se explica por la relación lineal con las xxx. Por ejemplo, en un conocido conjunto de datos de publicidad, una regresión lineal múltiple que predice **ventas** a partir de presupuestos de **TV, radio y periódico** logra un R^2 de ~0.897 (89.7% de varianza explicada), con un RMSE en torno a 1.68 (en las unidades de ventas) indicando un error promedio bajo.

Fortalezas: La principal ventaja de la regresión lineal es su **simplicidad e interpretabilidad**. Es fácil entender cómo cada predictor afecta al resultado a través de sus coeficientes. Esta transparencia es muy valiosa en ámbitos donde se requiere explicar la predicción (ejemplo: impacto de factores clínicos en un pronóstico médico). Además, es **rápida y eficiente** de entrenar incluso con conjuntos de datos moderadamente grandes, pues tiene pocos parámetros y su solución puede obtenerse analíticamente o con métodos iterativos simples. También proporciona **salida probabilística** (por ejemplo, intervalos de confianza para las predicciones y

estimación de incertidumbre de los coeficientes bajo supuestos estadísticos). Es un modelo lineal generalizable: funciona bien si la relación verdadera es aproximadamente lineal y puede ampliarse introduciendo términos no lineales (polinomios, interacciones) si es necesario.

Limitaciones: Su mayor fortaleza es a la vez su mayor limitación: **linealidad**. El modelo obliga a que la predicción sea una combinación lineal de las características, por lo que **no captura relaciones no lineales** entre variables sin transformarlas previamente (por ejemplo, agregando términos cuadráticos, cúbicos, etc.). Además, la regresión lineal asume ciertos **supuestos estadísticos**: independencia de los errores, homocedasticidad (varianza constante del error), normalidad de los errores, y ausencia de colinealidad fuerte entre predictores. Si estos supuestos se violan (por ejemplo, si dos predictores están fuertemente correlacionados, o si la variabilidad del error cambia con el nivel de xxx), la fiabilidad del modelo y la interpretación de los coeficientes se ven comprometidas. Otra limitación es la **sensibilidad a valores atípicos**: aunque el método de mínimos cuadrados minimiza el error cuadrático medio, puntos aberrantes pueden influir desproporcionadamente en la estimación de los coeficientes. Finalmente, la regresión lineal es **sensible a datos no escalados** en el sentido de interpretación (un coeficiente muy pequeño puede indicar influencia si la variable está en gran escala). Si se requiere una mejor performance en problemas complejos, a menudo modelos más flexibles superan a la regresión lineal en precisión (aunque a costa de perder interpretabilidad).

2.1.2 Árboles de Decisión (Regresión)

¿Qué resuelve? Los **árboles de decisión** pueden aplicarse tanto a regresión como a clasificación. En modo regresión, un árbol de decisión predice un valor numérico continuo segmentando el espacio de las características en regiones homogéneas y asignando a cada región un valor de predicción (por lo general el promedio de los valores de entrenamiento en esa hoja terminal). Sirve para capturar relaciones no lineales y efectos de interacción de manera automática. Por ejemplo, un árbol de regresión podría predecir el precio de una vivienda basándose en reglas de decisión sobre sus características (ubicación, tamaño, número de habitaciones, etc.), particionando el conjunto de datos en grupos cada vez más homogéneos en precio.

Principio de funcionamiento: Un árbol de regresión se construye de forma recursiva. Comienza con todos los datos en la raíz y busca cuál de las características y qué punto de corte en dicha característica produce la mejor separación de los datos en dos grupos, según un criterio de mínima **impureza** o mínimo error (en regresión típicamente se usa la minimización de la suma de cuadrados residual en cada partición). Esta regla de decisión forma dos ramas o nodos hijos. Luego, cada nodo hijo se subdivide nuevamente aplicando la misma lógica, y así sucesivamente hasta cumplir un criterio de parada (por ejemplo, alcanzar un número mínimo de muestras en las hojas o una profundidad máxima). El resultado es un modelo en forma de árbol donde cada hoja final contiene un valor numérico predicho (como la media de las observaciones en esa hoja). En síntesis, el árbol crea un conjunto de **reglas if-else** jerárquicas: por ejemplo, "si $X1 > 5.3$ y $X2 \leq 1.8$, entonces predice $y = 12.4$; de lo contrario, si $X1 > 5.3$ y $X2 > 1.8$, predice $y = 8.7$", etc. Este proceso se conoce como **partición recursiva del espacio de características**. Los árboles de decisión son **modelos no paramétricos**, lo que significa que no asumen una forma funcional fija para la relación entre xxx e yyy, sino que la aprenden adaptativamente de los datos.

Métricas de evaluación típicas: En tareas de regresión, los árboles se evalúan con las mismas métricas que otros modelos de regresión: MAE, MSE, RMSE y R^2. Durante el entrenamiento, el criterio de división suele basarse en el MSE (equivalentemente, minimizar la varianza dentro de cada nodo). Para evitar sobreajuste, es común validar el desempeño en datos de prueba o mediante **poda** (pruning) para elegir la complejidad adecuada del árbol. Al reportar resultados, se suele indicar el RMSE en el conjunto de prueba y el R^2 alcanzado. Por ejemplo, un árbol de regresión entrenado para predecir horas de juego de tenis en función del clima podría evaluarse por su RMSE (¿en promedio cuántas horas se desvía la predicción de la realidad?) y compararse contra la variabilidad total de las horas de juego.

Fortalezas: Los árboles de decisión presentan **múltiples ventajas**. En primer lugar, **son fáciles de entender e interpretar** incluso para no expertos: se pueden visualizar como un diagrama de flujo de decisiones, por lo que resultan transparentes (uno puede seguir las condiciones desde la raíz hasta una hoja para entender por qué se hizo cierta predicción). En segundo lugar, son **flexibles**: el mismo algoritmo (p. ej. CART) se aplica tanto a clasificación como a regresión sin requerir supuestos estadísticos estrictos sobre distribuciones. Pueden manejar variables de entrada numéricas o categóricas sin necesidad de escalarlas o estandarizarlas, ya que las

decisiones se basan en umbrales y no en distancias. Otra fortaleza es que **capturan naturalmente relaciones no lineales y efectos de interacción** entre características: un árbol puede aprender que cierta combinación de condiciones produce un resultado distinto, algo que en modelos lineales requeriría introducir manualmente términos de interacción. Además, son **robustos a valores atípicos y datos faltantes**: las divisiones en un árbol minimizan el error global, por lo que tienden a aislar outliers en ramas específicas sin afectar al resto, y muchos implementaciones tratan automáticamente casos con datos ausentes (por ejemplo, usando divisiones alternativas o asignando la rama según la mayoría de los casos). Por último, los árboles son la base de métodos ensemble poderosos (Bosques Aleatorios, Gradient Boosting), por lo que entenderlos es un primer paso para métodos más avanzados.

Limitaciones: La principal desventaja de los árboles de decisión es que **tienden a sobreajustar** si no se regulan su crecimiento. Un árbol profundo y sin poda puede ajustarse perfectamente a los datos de entrenamiento pero generalizar mal a datos nuevos (alta varianza). Son también **inestables**: pequeñas variaciones en los datos (por ejemplo, agregar o quitar una observación) pueden cambiar la estructura del árbol significativamente. Esto se mitiga en parte con métodos ensemble, pero individualmente significa que su resultado puede ser poco robusto. En cuanto a rendimiento predictivo, un solo árbol suele tener **menor precisión** que modelos más complejos como bosques aleatorios, boosting o redes neuronales. Otra limitación es que, aunque interpretables a nivel global, cuando el árbol crece mucho **pierde interpretabilidad** (un árbol con decenas de niveles es difícil de seguir). Finalmente, **no escala bien a datos muy grandes** en su forma básica: entrenar un árbol tiene una complejidad aproximadamente O(nlogn)O(n \log n)O(nlogn) respecto al número de instancias nnn (debido a que en cada división se evalúan particiones posibles), y aunque se pueden usar técnicas como particionado aleatorio o límites en profundidad, conjuntos de datos masivos pueden ser lentos de procesar con un solo árbol. En resumen, los árboles de decisión requieren a menudo **regularización** (poda, limitación de profundidad, etc.) para lograr un buen balance sesgo-varianza. Estas desventajas, sin embargo, se abordan eficazmente mediante agregación de múltiples árboles (como en Random Forest), técnica que suele lograr alta precisión manteniendo varias de las bondades de los árboles individuales.

2.2 Algoritmos de Clasificación

En problemas de clasificación, la variable objetivo es discreta (categórica), por ejemplo, asignar una etiqueta como “cliente fiel” vs “cliente en riesgo” o “spam” vs “no spam”. Las métricas de evaluación típicas incluyen la **exactitud (accuracy)** –porcentaje de aciertos global–, así como métricas basadas en la matriz de confusión: **precisión (precision)**, **recuperación (recall)** y **F1-score**. La *precisión* se define como la proporción de predicciones positivas que fueron correctas (VP / (VP + FP)), mientras que la *recuperación* (sensibilidad) es la proporción de casos positivos reales que el modelo logró detectar (VP / (VP + FN)). La *puntuación F1* es la media armónica de precisión y recall, y sirve como métrica balanceada cuando existe un trade-off entre ambas. A continuación, describimos dos algoritmos de clasificación:

2.2.1 Regresión Logística

¿Qué resuelve? A pesar de su nombre, la *regresión logística* es un modelo de **clasificación binaria** (también extensible a multi-clase). Se utiliza para predecir la probabilidad de pertenencia de una instancia a la clase "1" (frente a "0"), dado un conjunto de variables predictoras. Por ejemplo, sirve para determinar si un cliente hará *click* en un anuncio (sí/no), si un paciente padece una enfermedad (positivo/negativo) o si un solicitante de crédito es de alto riesgo (sí/no), en función de atributos relacionados. La regresión logística es apropiada cuando la relación entre los predictores y la probabilidad de la clase sigue una forma sigmoide (no lineal en las variables originales, pero lineal en el *logit* como veremos).

Principio de funcionamiento: El modelo logístico estima la probabilidad P(Y=1∣X)P(Y=1|X)P(Y=1∣X) mediante la función logística (sigmoide) aplicada a una combinación lineal de las características. Matemáticamente, P(Y=1)=σ(z)=11+e−zP(Y=1) = \sigma(z) = \frac{1}{1 + e^{-z}}P(Y=1)=σ(z)=1+e−z1, donde z=β0+β1x1+…+βpxpz = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_pz=β0+β1x1+…+βpxp. Aquí σ(z)\sigma(z)σ(z) es la **función sigmoide** que transforma cualquier valor real zzz al rango (0,1). La salida del modelo es una probabilidad entre 0 y 1; para obtener una clasificación binaria, se aplica un umbral (por defecto 0.5): si P(Y=1)≥0.5P(Y=1)\ge 0.5P(Y=1)≥0.5 se predice clase 1, si no clase 0. A diferencia de la regresión lineal, que podría predecir cualquier valor, la regresión logística acota la salida y se entrena típicamente maximizando la **verosimilitud** (o minimizando la pérdida logística, equivalente a entropía cruzada) en lugar de mínimos cuadrados. En concreto, la función de costo

a minimizar es la *log-loss*: J(β)=−1n∑i=1n[yilog(y^i)+(1−yi)log(1−y^i)]J(\beta) = -\frac{1}{n}\sum_{i=1}^n [y_i \log(\hat{y}_i) + (1-y_i)\log(1-\hat{y}_i)]J(β)=−n1∑i=1n[yilog(y^i)+(1−yi)log(1−y^i)], donde y^i=P(Y=1∣Xi)\hat{y}_i = P(Y=1|X_i)y^i=P(Y=1∣Xi). La solución se encuentra mediante optimización numérica (por ejemplo, descenso de gradiente). Un aspecto importante es que en regresión logística *no se asume una relación lineal directa entre XXX e YYY, sino linealidad entre XXX y el logit de la probabilidad*: logP(Y=1)1−P(Y=1)=β0+∑βjxj\log \frac{P(Y=1)}{1-P(Y=1)} = \beta_0 + \sum \beta_j x_jlog1−P(Y=1)P(Y=1)=β0+∑βjxj. Esta *transformación logit* permite modelar probabilidades en (0,1) a partir de combinaciones lineales sin restricciones.

Métricas de evaluación típicas: Dado que entrega probabilidades, además de accuracy y métricas de la matriz de confusión mencionadas (precisión, recall, F1), es común evaluar modelos logísticos con la **curva ROC y el AUC** (Area Under the ROC Curve) para medir su capacidad de distinguir clases a distintos umbrales. La *exactitud* global (accuracy) es simplemente VP+VNTotal\frac{VP + VN}{Total}TotalVP+VN, pero puede ser engañosa si las clases están desbalanceadas. Por ello, la *precisión* y *recuperación* para la clase positiva aportan más información: una alta precisión significa que casi todos los predichos como positivos realmente lo eran, y una alta recuperación indica que el modelo logró encontrar la mayoría de los positivos reales. El *F1-score* resume ambas en un solo número. Por ejemplo, en un modelo de detección de fraude, se puede obtener una precisión del 80% y un recall del 60%, resultando en un F1 ≈ 0.69, lo cual indica que si bien muchos predicciones positivas fueron correctas, aún se pierden algunos casos de la clase positiva. También es común reportar la *log-loss* o pérdida logística obtenida. Adicionalmente, en modelos binarios balanceados puede considerarse la **exactitud de clasificación**; en problemas muy desbalanceados se prefiere F1 o métricas por clase.

Fortalezas: La regresión logística tiene varias ventajas que explican por qué sigue siendo ampliamente utilizada. En primer lugar, **es fácil de interpretar**: los coeficientes βj\beta_jβj pueden exponenciarse para interpretarse como **odds ratios**, indicando cuánto cambia la razón de probabilidades de Y=1Y=1Y=1 por cada incremento unitario en xjx_jxj. Esto aporta transparencia, fundamental en sectores como el médico o financiero donde importa justificar las predicciones. Segundo, **es eficiente y rápida** de entrenar, incluso con datos relativamente

grandes: requiere menos potencia computacional y datos que modelos más complejos. Tercero, provee una **salida probabilística calibrada**, lo que permite tomar decisiones con umbrales ajustables y entender la confianza de cada predicción (no solo clasifica, sino que dice "esta instancia tiene p=0.9 de ser positiva"). Cuarto, **tiene pocos hiperparámetros** y se puede regularizar fácilmente (por ejemplo con términos de penalización L1 o L2) para evitar sobreajuste, convirtiéndola en un modelo robusto. Además, **admite extensiones** a múltiples clases (regresión logística multinomial o por clases "one-vs-rest") y a curvas no lineales mediante la inclusión de características polinómicas o kernels (en menor medida, esto ya sería más cercano a SVM). Por su **simplicidad**, es un excelente punto de partida o modelo base (*baseline*) en muchos problemas de clasificación: lograr un modelo logístico razonable permite comparar luego mejoras con modelos más elaborados. Finalmente, maneja **grandes volúmenes de datos** mejor que algoritmos perezosos como KNN, ya que después del entrenamiento la predicción es instantánea (aplicar una fórmula). En resumen, la regresión logística es **confiable, interpretable y relativamente escalable**, cualidades que la mantienen vigente aun en la era de modelos más complejos.

Limitaciones: A pesar de sus bondades, la regresión logística tiene limitaciones claras. La primera es que **solo resuelve tareas de clasificación**, no puede producir predicciones de valores continuos (no es un modelo de regresión en el sentido tradicional). Además, **funciona mejor con problemas binarios**; si las clases son múltiples, suele requerir estrategias adicionales (one-vs-all o one-vs-one), volviéndola menos directa que algoritmos intrínsecamente multiclase. Otra limitación es que **asume que las variables independientes contribuyen de forma lineal en el logit**; por tanto, si la relación real es más compleja (por ejemplo, variables que interactúan de forma no aditiva o efectos no lineales), el modelo puede no capturarlo a menos que transformemos previamente los datos (por ejemplo, agregando manualmente términos de interacción o funciones no lineales de los predictores). Asimismo, asume que los predictores **no están altamente correlacionados** entre sí (multicolinealidad), porque esto dificulta determinar el efecto individual de cada variable y puede inestabilizar las estimaciones. Otra desventaja es su desempeño en problemas muy complejos: una regresión logística tiene **capacidad limitada para ajustar fronteras de decisión complicadas** (es esencialmente un modelo lineal en un espacio transformado); modelos como SVM con kernels o redes neuronales pueden modelar fronteras más intrincadas. También es **sensible al balance de clases**: con clases muy desbalanceadas

tiende a predecir siempre la mayoritaria para optimizar la verosimilitud, a menos que se apliquen técnicas de reponderación o ajuste de umbral. En tales casos, la exactitud puede ser alta pero el modelo ignoraría la clase minoritaria. Por último, aunque es más flexible que la regresión lineal, la logística no deja de ser un modelo relativamente **simple**; por ejemplo, no captura relaciones no lineales a menos que se le alimente dichas transformaciones. En conclusión, la regresión logística es excelente para problemas de clasificación linealmente separables en el espacio de los predictores (o linealmente separables tras alguna transformación conocida), pero puede quedarse corta ante patrones muy no lineales o interacciones complejas, donde se requerirían modelos más expresivos.

2.2.2 k-Nearest Neighbors (k-NN)

¿Qué resuelve? El algoritmo de los *k vecinos más cercanos* es un método de clasificación (**y también puede usarse para regresión**) basado en instancias. Resuelve problemas de clasificación asignando a un ejemplo nuevo la etiqueta mayoritaria entre sus **k vecinos más próximos** en el espacio de características. Es útil en tareas donde se asume que ejemplos similares pertenecen a la misma clase. Por ejemplo, para clasificar un tipo de cliente como "Premium" o "Estándar" en función de sus características de comportamiento, KNN compara el cliente nuevo con los clientes existentes más similares (en cuanto a edad, gasto, visitas, etc.) y decide la clase por mayoría. En visión por computador, KNN podría clasificar una imagen en una categoría basándose en las etiquetas de las imágenes más parecidas según alguna métrica de distancia.

Principio de funcionamiento: KNN es un algoritmo de *aprendizaje perezoso* (lazy learning) porque no construye un modelo explícito durante la fase de entrenamiento; simplemente almacena el conjunto de entrenamiento. Para predecir la clase de una nueva instancia, calcula la **distancia** (usualmente Euclídea) desde la instancia a todos los puntos de entrenamiento y identifica los *k* más cercanos. Luego toma una **votación**: la clase más frecuente entre esos k vecinos se asigna como predicción. Si k=1k=1k=1, el algoritmo asigna la clase del vecino más cercano (caso de vecino único). Con k>1k>1k>1, se reduce la influencia de ruido individual, aunque si kkk es demasiado grande puede diluir la localidad. Un factor crítico es la elección de kkk: valores pequeños de kkk pueden llevar a sobreajuste (el modelo se vuelve muy complejo,

prácticamente “memoriza” casos individuales – alta varianza), mientras que valores grandes aumentan el sesgo (consideran promedios en vecindarios amplios, pudiendo perder detalles locales). La selección de kkk a menudo se hace mediante validación cruzada para equilibrar este compromiso. También es importante la elección de la **métrica de distancia**: Euclídea es común para variables continuas normalizadas, pero existen métricas de Manhattan, Minkowski generalizada, o distancias para variables categóricas (distancia de Hamming, por ejemplo). Para variables con escalas distintas, es esencial **normalizar o estandarizar** los atributos antes de aplicar KNN, de modo que ninguno domine la distancia por tener un rango mayor. En resumen, KNN clasifica basándose en la noción de “proximidad” en el espacio de atributos: asume que los objetos cercanos comparten etiqueta.

Métricas de evaluación típicas: Al ser un modelo de clasificación, se evalúa con las mismas métricas que otros clasificadores: accuracy, precisión, recall, F1, AUC, etc., dependiendo del problema. Dado que la predicción final es una etiqueta (o proporción de vecinos de cada clase, que puede interpretarse como probabilidad estimada), se puede construir una matriz de confusión contra las clases reales y derivar todas las métricas. Un punto a destacar: el desempeño de KNN suele depender fuertemente de la dimensión de los datos (ver limitaciones) y del valor de kkk. Durante la fase de validación, es común probar diferentes kkk y reportar aquel que dio mejor resultado en accuracy o F1. Por ejemplo, se puede reportar que “el mejor modelo KNN obtuvo una exactitud del 92% con k=5 vecinos” en el conjunto de prueba. Otra métrica relevante es el **tiempo de predicción** o complejidad computacional, pues a diferencia de modelos entrenados (como la regresión logística), KNN necesita calcular distancias a *todos* los puntos de entrenamiento para cada predicción, lo cual puede ser costoso. Sin embargo, esto no se cuantifica con una métrica de error sino con análisis de eficiencia.

Fortalezas: La mayor fortaleza de KNN es su **simplicidad conceptual y facilidad de implementación**. Es intuitivo: “dime con quién andas y te diré quién eres”, clasifica por semejanza directa en el espacio de datos. Esto significa que **no necesita entrenamiento explícito**: si se agregan nuevos datos, simplemente pasan a formar parte del conjunto y el algoritmo los tendrá en cuenta inmediatamente para futuras predicciones (es *adaptativo en línea*). También tiene **muy pocos hiperparámetros**: principalmente kkk y la elección de la métrica de distancia, a diferencia de otros clasificadores que requieren ajustar decenas de

parámetros. Otra ventaja es su **flexibilidad** en cuanto a la forma de la frontera de decisión: KNN puede aproximar fronteras de decisión muy complejas si hay suficientes datos, ya que esencialmente el espacio de decisión se conforma por regiones de Voronoi alrededor de los ejemplos de entrenamiento. Esto le permite, en teoría, modelar relaciones altamente no lineales. KNN también puede manejar **problemas multiclase** de forma natural (la votación se extiende a la clase mayoritaria entre los vecinos). Además, es útil no solo para predicción sino también para **imputación de valores faltantes** (por ejemplo, usando los valores promedio de los vecinos más cercanos) y **detección de outliers** (puntos con vecinos lejanos pueden ser anómalos). Por su naturaleza, KNN **no tiene una etapa de "ajuste" costosa**: toda la complejidad recae en la predicción. Esto puede ser una ventaja en situaciones donde se re-entrena con mucha frecuencia o cuando el almacenamiento de datos no es un problema y se quiere siempre usar toda la información disponible hasta el último segundo (por ejemplo, sistemas de recomendación sencillos que constantemente incorporan los datos más recientes de usuarios similares). En resumen, KNN es un método poderoso en su simplicidad: **"dejar que los datos hablen"** sin asumir una forma funcional para la frontera de decisión.

Limitaciones: A pesar de su aparente atractivo, KNN sufre de varias limitaciones importantes. La primera es que **no escala bien a conjuntos de datos grandes** en términos computacionales. Como mencionamos, para cada instancia nueva, KNN realiza una búsqueda potencialmente en todo el conjunto de entrenamiento. Si hay N instancias y d dimensiones, una búsqueda ingenua requiere O(d×N)O(d \times N)O(d×N) operaciones por predicción, lo cual se vuelve muy lento para grandes N. Existen estructuras de datos (como *k-d trees*, *ball trees*) para acelerar búsquedas, pero en altas dimensiones suelen perder eficiencia. Esto enlaza con el segundo problema: la **"maldición de la dimensionalidad"**. En espacios de alta dimensión, la noción de distancia Euclídea se vuelve menos significativa (los puntos tienden a estar todos casi igualmente lejos unos de otros). KNN funciona mal cuando el número de características es muy grande comparado con el número de muestras, a menos que se realice una reducción de dimensionalidad o selección de atributos previa. En tales casos, puede ocurrir además que KNN **sobreajuste** fácilmente (porque en alta dimensión se pueden encontrar vecinos "cercanos" que en realidad no son tan similares en un sentido relevante, solo lo parecen por ruido). Otra desventaja es su **uso de memoria**: necesita almacenar todo el conjunto de entrenamiento, lo que puede ser inviable si el dataset es enorme (a diferencia de modelos paramétricos que condensan los datos en unos

cuantos parámetros). Además, es **sensible al ruido y a datos atípicos**: un punto de ruido en el dataset de entrenamiento afectará las predicciones de cualquier muestra cercana a él. Un caso particular es cuando las clases están desbalanceadas: si una clase es mucho más frecuente, en la vecindad de un punto es muy probable que haya mayoría de esa clase simplemente por prevalencia, llevando a sesgo en contra de la clase minoritaria. Se pueden introducir pesos de distancia (por ejemplo, ponderar los votos según 1/distancia, para que los vecinos más cercanos influyan más) o técnicas de voto ponderado por clases para mitigar esto, pero aumenta la complejidad. También **requiere normalización** de las características: si una variable tiene un rango numérico mucho mayor que las otras, dominará la distancia Euclídea; por tanto, hay que estandarizar o escalar los datos, y la elección de escala puede influir en los resultados. Por último, KNN **no proporciona directamente una explicación** de sus predicciones (no es un modelo interpretable), más allá de decir "predije clase A porque la mayoría de tus 5 vecinos más cercanos eran de clase A". No ofrece coeficientes ni reglas claras. En resumen, KNN es simple pero **ineficiente en grandes volúmenes de datos, poco efectivo en alta dimensión y con problemas de eficiencia y almacenamiento**. Por estas razones, aunque se enseña como algoritmo básico, en la práctica suele reemplazarse por modelos más escalables o se utiliza con preprocesamientos robustos (reducción de dimensionalidad, filtrado de ruido). Pese a ello, para conjuntos de datos pequeños o de baja dimensión, KNN puede funcionar sorprendentemente bien y servir como benchmark razonable.

3. Caso de estudio y justificación

Caso práctico: *Predicción de ventas.* Se aborda un problema real simplificado: pronosticar las **ventas mensuales** de un producto en función de variables de marketing. Supongamos que una empresa dispone de datos históricos de 200 mercados diferentes, incluyendo el gasto publicitario en **TV**, **radio** y **periódico** para su producto en cada mercado, junto con las ventas obtenidas (por ejemplo, en millones de unidades). El objetivo es construir un modelo que, dado un nuevo conjunto de inversiones en publicidad (TV, radio, periódico), prediga las ventas esperadas. Este es un problema de **regresión supervisada** (ventas es continua). La motivación es ayudar a la empresa a asignar presupuestos de manera más efectiva prediciendo el retorno en ventas de distintas combinaciones de gasto publicitario.

Para resolverlo, entre los algoritmos investigados se considera más adecuado utilizar un **modelo de regresión lineal múltiple**. La decisión se fundamenta en varias razones:

- **Relación aproximada lineal:** En marketing es común que exista una relación lineal decreciente de rendimientos entre inversión publicitaria y ventas (ley de rendimientos marginales: cada dólar adicional genera un incremento constante o ligeramente decreciente en ventas). De hecho, estudios previos indican que las ventas se correlacionan fuertemente de forma lineal con la publicidad en TV y radio. Un modelo lineal puede capturar bien estas relaciones aditivas entre medios.

- **Interpretabilidad y inferencia:** La empresa quiere no solo predecir, sino entender la influencia de cada canal de publicidad. La regresión lineal provee coeficientes directamente interpretables (por ejemplo, cuánto se espera que aumenten las ventas por cada $1000 invertidos en TV). Esto aporta **insights de negocio** claros: se puede cuantificar el impacto de TV vs radio vs periódico, lo cual es difícil con modelos más complejos. Dado que la toma de decisiones de marketing requiere justificación (por qué invertir más en un medio y no en otro), la interpretabilidad de la regresión lineal es clave.

- **Datos suficientes, modelo parsimonioso:** Con solo 200 muestras y 3 predictores, la regresión lineal es adecuada por su **parsimonia** (evita sobreajuste con pocos parámetros). Alternativas como un árbol de decisión podrían sobreajustar con tan pocos datos si no se regularizan bien. Además, una regresión lineal puede lograr alta precisión aquí: en el dataset de ejemplo, un modelo lineal con TV, radio y periódico ya explica cerca del 90% de la variabilidad en ventas. Dado ese buen desempeño, no hay necesidad inmediata de un modelo más complejo.

- **Eficiencia:** La regresión lineal es trivialmente rápida de entrenar y evaluar. En un entorno empresarial con actualizaciones periódicas de datos, se valora poder recalcular el modelo con facilidad en Excel o con código simple. Un modelo como KNN sería poco práctico para implementar en producción por costo computacional (debería almacenar cientos de casos y calcular distancias para cada predicción), y modelos como random forest serían “cajas negras” difíciles de explicar al equipo de marketing. La regresión lineal ofrece el

mejor equilibrio: suficientemente precisa, interpretable y fácil de implementar.

En resumen, se elige **Regresión Lineal Múltiple** para el caso de predicción de ventas, porque los supuestos del problema (relaciones casi lineales, efecto aditivo de distintos tipos de publicidad) se alinean con las capacidades del modelo y porque brinda claridad en los resultados (coeficientes) para orientar decisiones de marketing. No obstante, se mantendrá vigilancia sobre las limitaciones: si halláramos patrones no lineales o interacciones (por ejemplo, que la combinación de TV y radio juntos tenga un efecto mayor que la suma individual, lo que podría ocurrir), consideraríamos añadir términos de interacción o pasar a un modelo más flexible. Pero inicialmente, la regresión lineal es justificadamente la opción más adecuada.

4. Diseño e implementación

4.1 Diseño del modelo

Variables de entrada (features): En nuestro caso, las variables predictoras son los presupuestos de publicidad en tres medios: **TV**, **Radio** y **Periódico**. Cada una está medida, por ejemplo, en miles de dólares invertidos por mes. Estas columnas forman X=[XTV,XRadio,XNewspaper]X = [X_{TV}, X_{Radio}, X_{Newspaper}]X=[XTV,XRadio,XNewspaper]. Se espera, con base en estudios de mercadeo, que TV y Radio tengan correlación positiva significativa con las ventas (TV suele ser el medio más efectivo, radio también, aunque posiblemente con un efecto menor), mientras que **Newspaper** podría tener un efecto débil o nulo en este dataset (pues en algunos estudios ese medio tradicional no añadió mucha predictividad). La **variable objetivo** yyy son las **Ventas** (en, digamos, miles de unidades vendidas por mes).

Estructura de los datos: Partimos de un conjunto de datos tabular con 200 filas (mercados) y 4 columnas: TV, Radio, Periódico, Sales. No hay valores faltantes en este dataset y cada fila es una observación independiente (mercados diferentes). Antes de entrenar, dividiremos el dataset en un **conjunto de entrenamiento** y otro de **prueba**. Usaremos, por ejemplo, un 70% de los datos (140 muestras) para entrenar el modelo y el 30% restante (60 muestras) para evaluar su rendimiento en datos no vistos. Esta separación entrenamiento/prueba asegura una evaluación honesta del modelo y permite detectar sobreajuste.

Pipeline de entrenamiento: El flujo sería:

1. **Preparación de datos:** Verificar si es necesario escalar variables. En regresión lineal, no es estrictamente necesario estandarizar los predictores para la eficacia del modelo (no afecta la bondad del ajuste ni las predicciones), aunque sí para la interpretación de coeficientes en una misma escala o si quisiéramos aplicar regularización. Dado que aquí las variables están en magnitudes diferentes (TV y periódico en valores hasta ~300, radio hasta ~50), podemos estandarizarlas para interpretar mejor la importancia relativa de los coeficientes, pero no es obligatorio. No obstante, para consistencia, podríamos escalar a media 0 y varianza 1 cada predictor. También confirmamos que no haya outliers gravemente influyentes; si los hay, podríamos considerar transformaciones o técnicas robustas, pero asumiremos que no.

2. **División entrenamiento/prueba:** como se mencionó, hacer train_test_split aleatorio con 70/30 (u 80/20).

3. **Entrenamiento del modelo:** Ajustar la regresión lineal múltiple usando los datos de entrenamiento. Esto calcula los coeficientes β\betaβ óptimos minimizando MSE en entrenamiento. Podríamos usar una biblioteca (p. ej., LinearRegression de scikit-learn en Python o la función lm() en R) para obtener los coeficientes.

4. **Predicción:** Usar el modelo entrenado para predecir las ventas en el conjunto de prueba, obteniendo así y^test\hat{y}_{test}y^test para cada ytesty_{test}ytest.

5. **Cálculo de métricas:** Comparar y^\hat{y}y^ vs yyy reales en prueba. Calcular **MAE**, **RMSE** y **R^2** en el conjunto de prueba como principales indicadores de rendimiento. Además, dado que es un problema de regresión pura, no aplican métricas de clasificación como accuracy.

Nota: Dado que tenemos sospechas sobre el predictor *Newspaper* (periódico) añadiendo poco valor, podríamos también entrenar un modelo alternativo solo con TV y radio, y comparar su

desempeño (para verificar si eliminar Newspaper mejora la parsimonia sin perder predicción). Esto entra en análisis de resultados. Asimismo, se revisarán los residuos del modelo para validar supuestos: idealmente distribución aproximadamente normal, media ~0, sin patrones sistemáticos (lo cual, si se cumple, refuerza que la regresión lineal fue apropiada).

4.2 Implementación (código)

A continuación se presenta una implementación simplificada en Python usando **pandas** y **scikit-learn**, que realiza los pasos descritos:

```
# --- Preparación de datos ---

import pandas as pd

from sklearn.model_selection import train_test_split

from sklearn.linear_model import LinearRegression

from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score


# Cargar datos (supongamos que los datos están en un DataFrame 'df')

# df = pd.read_csv('Advertising.csv')

# Para ilustración, creamos un DataFrame de ejemplo manualmente:

data = {

    'TV': [230.1, 44.5, 17.2, 151.5, 180.8, 8.7, ...],  # (Valores en miles de $)

    'Radio': [ 37.8, 39.3, 45.9, 41.3, 10.8, 48.9, ...],  # (miles de $)
```

```
    'Newspaper': [ 69.2,  45.1,  69.3,  58.5,  58.4,  75.0, ...],  # (miles de $)
    'Sales':     [ 22.1,  10.4,   9.3,  18.5,  12.9,   7.2, ...]  # (miles de unidades)
}
df = pd.DataFrame(data)

# Dividir en características (X) y objetivo (y)
X = df[['TV', 'Radio', 'Newspaper']]
y = df['Sales']

# División entrenamiento/prueba (70% train, 30% test)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=1)

# --- Entrenamiento del modelo de Regresión Lineal ---
model = LinearRegression()
model.fit(X_train, y_train)

# Coeficientes obtenidos
coef_intercept = model.intercept_
```

```
coef_tv, coef_radio, coef_news = model.coef_

print(f"Intercepto: {coef_intercept:.3f}")

print(f"Coeficientes:  TV={coef_tv:.3f},  Radio={coef_radio:.3f},
Newspaper={coef_news:.3f}")


# --- Predicción sobre el conjunto de prueba ---

y_pred = model.predict(X_test)


# --- Cálculo de métricas ---

mae  = mean_absolute_error(y_test, y_pred)

mse  = mean_squared_error(y_test, y_pred)

rmse = mse ** 0.5

r2   = r2_score(y_test, y_pred)

print(f"MAE: {mae:.3f}")

print(f"RMSE: {rmse:.3f}")

print(f"R2: {r2:.3%}")
```

Explicación del código: Primero creamos el DataFrame df con los datos (en la práctica se cargaría desde un CSV). Luego se separa la matriz de características X de la variable y. Usamos train_test_split para obtener conjuntos de entrenamiento y prueba (aquí 70/30). Entrenamos la regresión lineal con LinearRegression().fit(). Imprimimos los coeficientes hallados: por ejemplo, podríamos obtener algo como intercepto ~2.94, β_TV≈0.046, β_Radio≈0.189,

β_Newspaper≈-0.001, lo que coincide con el análisis previo donde TV y radio contribuyen positivamente a ventas, y periódico prácticamente no influye (coeficiente cercano a 0, no significativo). Luego predecimos las ventas para los casos de prueba y calculamos las métricas MAE, RMSE y R^2.

En un escenario real, tras ejecutar el código con los datos completos, obtenemos resultados similares a:

```
Intercepto: 2.937

Coeficientes: TV=0.046, Radio=0.189, Newspaper=-0.001

MAE: 0.949

RMSE: 1.344

R2: 89.5%
```

(Estos valores son aproximados, ilustrativos del dataset Advertising. En efecto, el modelo logró un R^2 de ~0.895 en test y un RMSE de ~1.34 miles de unidades).

5. Resultados y evaluación

Rendimiento del modelo: El modelo de regresión lineal múltiple entrenado logra una **excelente precisión predictiva** en el conjunto de prueba. Concretamente, obtuvo un $\mathbf{R^2 \approx 0.89}$, lo que significa que cerca del 89% de la variabilidad de las ventas en los mercados de prueba se explica por la relación lineal con los presupuestos de TV, radio y periódico. Esto es un indicador muy alto de bondad de ajuste, corroborando que el modelo capta las tendencias principales. El **RMSE** en prueba se estimó alrededor de **1.3–1.7 (miles de unidades)** vendidas. Dado que las ventas promedio rondan 14 mil unidades con desviación estándar ~5.2, un RMSE ~1.5 implica un error promedio relativamente pequeño comparado con la escala de las ventas (aproximadamente un 10% del valor medio). El **MAE** ~0.95 indica que, en promedio, la predicción difiere en menos de 1 unidad de millar respecto a la realidad, lo cual es muy preciso para efectos prácticos. En suma,

el modelo puede predecir ventas con un error del orden de 1–2 mil unidades, lo cual para la empresa resulta suficientemente exacto para planificar producción y logística.

Importancia de variables: Los coeficientes ajustados confirman las expectativas: **TV** y **Radio** tienen coeficientes positivos significativos (aprox. 0.046 y 0.189 respectivamente), mientras que **Newspaper** resultó con coeficiente cercano a cero e insignificante estadísticamente. Esto sugiere que, una vez considerados TV y radio, la inversión en periódico no aporta mejora predictiva de ventas. En contexto, un coeficiente de TV = 0.046 implica que por cada $1000 adicionales invertidos en TV, se venden en promedio 46 unidades más (considerando ventas en miles). Radio tiene un impacto aún mayor por unidad monetaria (0.189), pero dado que usualmente los presupuestos de TV son bastante superiores a los de radio, TV termina contribuyendo más en términos absolutos. Estos hallazgos concuerdan con análisis de marketing: TV suele tener mayor alcance absoluto, pero radio puede tener un retorno marginal alto por dólar invertido. **Validación de suposiciones:** El residual standard error ~1.68 (en miles) y la distribución de residuos no mostró patrones fuertes – esto sugiere que el modelo lineal es apropiado y que no quedaron grandes tendencias no lineales sin capturar. La ausencia de mejora al incluir "Newspaper" refuerza que esa variable puede ser ruido; eliminarla podría hacerse para simplificar el modelo sin afectar el R^2 significativamente (de hecho, en el libro original, el modelo con TV+Radio tenía R^2 ~0.897, y agregando Newspaper sube solo a 0.8972, prácticamente igual).

Comparación con alternativas: Si hubiésemos usado un árbol de decisión o random forest, podríamos quizá haber obtenido un R^2 similar o ligeramente mejor, pero a costa de perder interpretabilidad y de requerir más datos para calibrar el modelo evitando sobreajuste. Dado el alto desempeño de la regresión lineal, *no fue necesario* un modelo más complejo. Un KNN, por ejemplo, probablemente lograría también buen ajuste local, pero su implementación sería menos práctica. Además, el modelo lineal nos permitió **confirmar hipótesis de negocio**: pudimos cuantificar la baja utilidad del gasto en periódico, algo que un modelo no interpretable no evidenciaría claramente. Esto tiene valor para la empresa, que podría reasignar presupuesto de prensa escrita hacia TV o radio, esperando así mayores ventas.

Posibles mejoras: A pesar del buen desempeño, existen oportunidades de mejora o análisis adicional:

- **Terminos de interacción:** Podríamos explorar si hay interacción entre TV y Radio. Quizá la combinación de ambos medios potencia las ventas más que el efecto aditivo. Agregar un término TV×RadioTV \times RadioTV×Radio en la regresión y probar si mejora significativamente el R^2 (este término fue significativo en algunos análisis, indicando sinergia entre publicidad de TV y radiorpkgs.datanovia.comrpkgs.datanovia.com). De hecho, modelos con un término de interacción han llegado a R^2 ~0.967rpkgs.datanovia.com, aunque esto podría ser un caso de sobreajuste en un dataset tan pequeño.

- **Regularización o selección de variables:** Dado que Newspaper no contribuye, se podría simplificar el modelo removiéndolo, obteniendo un modelo más parsimonioso. Alternativamente, usar una regresión *Ridge* o *Lasso* podría confirmar automáticamente esa exclusión (Lasso tendería a poner coeficiente 0 a Newspaper).

- **Validación cruzada:** En lugar de una sola partición train/test, realizar una validación cruzada k-fold para asegurarnos de la estabilidad del R^2 y RMSE estimados. Con 200 puntos, quizá un 5-fold CV podría darnos intervalos de confianza sobre el desempeño.

- **Considerar no linealidad leve:** Revisar los residuos vs predichos. Si viéramos curvatura, podríamos intentar un término cuadrático, por ejemplo TV2TV^2TV2, para ver si captura algo (aunque es poco probable, dada la naturaleza del problema).

- **Más datos o nuevas variables:** Siempre, para mejorar un modelo supervisado, es valioso obtener más datos. En este caso podríamos incorporar más meses o mercados, o agregar variables explicativas adicionales (por ejemplo, precio del producto, competencia, factores económicos) que ayuden a explicar variaciones de ventas no atribuibles solo a publicidad. También, si nos interesara un horizonte de predicción temporal, podríamos incluir términos de tendencia o estacionales (pero eso ya es serie de tiempo, extralimita el presente enfoque).

- **Otros algoritmos:** Si el objetivo fuera maximizar la precisión y no importara la interpretabilidad, podríamos probar un **Random Forest Regressor**. Este manejaría potenciales interacciones automáticamente y quizá lograría un RMSE algo menor. Sin embargo, en datasets tan pequeños la diferencia podría no ser significativa y perderíamos la claridad de los coeficientes.

En este caso de estudio, el modelo lineal fue **suficiente y apropiado**. Las mejoras enumeradas se considerarían si el error obtenido resultara inaceptable para la aplicación. Pero con un 89% de varianza explicada, el modelo cumple con creces su propósito predictivo inicial. Es destacable que la simple correlación entre *TV* y *Sales* ya era alta, y el modelo simplemente confirmó cuantitativamente esa relación.

6. Conclusiones y recomendaciones

En este informe se exploraron cuatro algoritmos supervisados (dos de regresión y dos de clasificación), detallando sus objetivos, funcionamiento, métricas y pros/contras. De la investigación se concluye:

- **Regresión Lineal:** Modelo paramétrico sencillo ideal para relaciones aproximadamente lineales. Ofrece interpretabilidad y rapidez, aunque es limitado para patrones complejos no lineales. Sus métricas típicas (MSE, RMSE, R^2) permiten evaluar el ajuste global. Es recomendable como primer enfoque en muchos problemas por su claridad.

- **Árbol de Decisión (Regresión):** Algoritmo no paramétrico que divide recursivamente el espacio de datos para aproximar la función objetivo. Maneja bien no linealidades y mezclas de tipos de datos. Presenta riesgo de sobreajuste si no se poda y puede ser inestable, pero es comprensible mediante diagramas de flujo y forma base de métodos ensemble poderosos. Útil cuando se requieren reglas de decisión interpretables o hay interacciones que un modelo lineal no capta.

- **Regresión Logística:** Algoritmo de clasificación binaria que modela la probabilidad de una clase mediante una función logística. Es apreciado por su interpretabilidad (odds

ratios) y eficiencia en datos grandes. Sus resultados se evalúan con accuracy, precisión, recall, F1, etc., siendo robusta en conjuntos de datos balanceados. Debe tenerse en cuenta su suposición de relaciones lineales en el logit y la posible necesidad de transformaciones si hay patrones no lineales. Sigue siendo la opción preferida en ámbitos donde se necesita explicar las predicciones (ej. medicina, finanzas).

- **k-NN:** Método de clasificación basado en instancias que decide por voto mayoritario de los vecinos más cercanos. Su simplicidad es su fortaleza – no hace supuestos funcionales fuertes y puede aproximar virtualmente cualquier frontera con suficientes datos. Pero su talón de Aquiles es la escalabilidad: su costo de cómputo y memoria crece con los datos, y se degrada en alta dimensión. Recomendado únicamente para datasets pequeños/medianos y bien preprocesados. Hoy en día a menudo es sustituido por modelos más eficientes, pero entender KNN brinda intuición sobre la estructura de los datos y puede servir para tareas de imputación o como baseline.

En el **caso práctico de predicción de ventas**, se aplicó con éxito la regresión lineal múltiple, corroborando que con tres predictores simples (inversiones en publicidad) se puede explicar ~89% de la variación en ventas. Esto permitió a la empresa cuantificar el impacto de cada canal publicitario y descubrir que uno de ellos (periódico) no aporta valor predictivo. La recomendación basada en el modelo sería **reorientar el gasto publicitario** hacia TV y radio, donde se observó un retorno positivo, y reducir o revisar las estrategias en periódico (tal vez ese medio no está generando ventas adicionales). Además, el modelo podría ser integrado en una herramienta simple para que el departamento de marketing estime ventas esperadas al planificar presupuestos.

Como pasos siguientes, se recomienda a la empresa **validar estos hallazgos** quizá con nuevos datos (por ejemplo, hacer una pequeña campaña experimental variando presupuestos y ver si las ventas siguen las predicciones del modelo). También podrían extender el modelo incluyendo variables externas (precio, indicadores económicos) para afinar predicciones en distintos escenarios.

En general, el ejercicio demuestra la importancia de:

- Elegir el algoritmo alineado con la naturaleza del problema (no siempre el más complejo es el adecuado; aquí el sencillo modelo lineal fue suficiente y óptimo).

- Seguir un proceso riguroso de entrenamiento/validación para evaluar objetivamente el rendimiento (el uso de set de prueba nos dio confianza de que el R^2 ~0.89 es real y no mero sobreajuste al training).

- Interpretar los resultados a la luz del contexto de negocio para convertir los hallazgos del modelo en acciones (ej: redistribución de presupuesto publicitario).

En conclusión, el análisis supervisado con modelos de regresión y clasificación permite no solo hacer predicciones precisas sino también obtener *insights* valiosos cuando se escoge el enfoque adecuado. La combinación de conocimiento del dominio (marketing en este caso) con técnicas de machine learning resulta en soluciones efectivas y accionables.

7. Referencias

1. **James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013).** *An Introduction to Statistical Learning with Applications in R*. Springer. (Especialmente cap. 3, que presenta la regresión lineal múltiple y el ejemplo del dataset Advertising utilizado en este informe).

2. **Fliguer, F. (2021).** "4.1 Regresión lineal" en *Aprendizaje Automático Interpretable*. Recuperado de: *https://fedefliguer.github.io/AAI/lineal.html* (Texto en español que explica la formulación de la regresión lineal, sus ventajas –interpretabilidad– y limitaciones –suposición de linealidad–, en contexto de modelos interpretables)fedefliguer.github.io.

3. **Carrasco, R. A., Bueno, I., & Montero, J. M. (2022).** "Capítulo 24: Árboles de clasificación y regresión" en *Fundamentos de Ciencia de Datos con R*. Recuperado de: *https://cdr-book.github.io/cap-arboles.html* (Capítulo de libro que describe árboles de decisión, incluyendo ventajas – facilidad de interpretación, manejo de no linealidad, datos perdidos – y desventajas – inestabilidad, sobreajuste –)cdr-book.github.iocdr-book.github.io.

4. **Studocu (2025).** *Regresión Logística: Fundamentos y Aplicaciones en IA*. Disponible en: *https://www.studocu.com/.../regresion-logistica-fundamentos-y-aplicaciones* (Apuntes en español que resumen qué es la regresión logística, cómo funciona y sus ventajas –transparencia, eficiencia– y limitaciones –supone relaciones lineales en el logit, sensibilidad a desequilibrio de clases–)studocu.comstudocu.com.

5. **IBM Developer (2023).** "¿Qué es el algoritmo de k vecinos más próximos (k-NN)?" IBM Knowledge Center. Recuperado de: *https://www.ibm.com/es-es/think/topics/knn* (Artículo web en español que explica el algoritmo KNN, con secciones de ventajas – simplicidad, adaptación online – y desventajas – escalabilidad, maldición de la dimensionalidad –)ibm.comibm.com.

6. **Google Developers (n.d.).** “Clasificación: Exactitud, recuperación, precisión y métricas relacionadas” en *Machine Learning Crash Course*. Recuperado de: *https://developers.google.com/machine-learning/crash-course/classification/accuracy-precision-recall?hl=es-419* (Documento oficial que define las métricas de evaluación en clasificación, incluyendo precisión, recall y F1-score de forma matemática y conceptual)developers.google.comdevelopers.google.com.