Predicción de Instalaciones
DATA SET GOOGLE

Google Play Store | Machine Learning & Análisis de Sentimiento

Fernando Soler Manriquez.
Stgo, Octubre 2025

Objetivo

OBJETIVO

Predecir cuántas descargas tendrá una aplicación móvil en Google Play.
Usar variables numéricas y la categoría de la app como insumo.
Entregar una estimación confiable para apoyar decisiones de producto y marketing.

Secuencia a Desarrollar:

1. EDA y limpieza (nulos, outliers, conversión de variables).
2. Métricas (R^2, RMSE, MAE) para validar desempeño.
3. Modelo base y modelo mejorado.
4. API funcional para predicción en tiempo real.

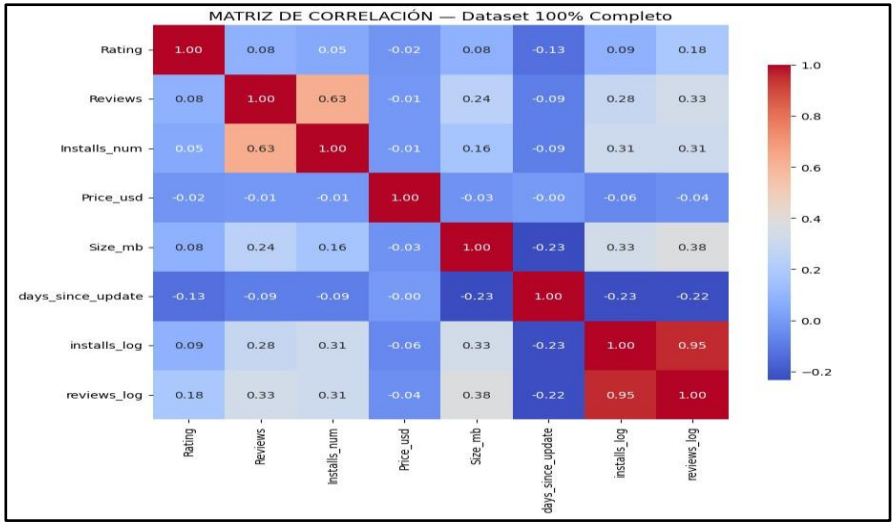
Introducción:

Se realizará una limpieza completa de los datos, eliminando valores inconsistentes, tratando outliers y normalizando variables clave.
Se identificarán como entradas del modelo el rating, cantidad de reseñas, precio, tamaño de la aplicación, días desde la última actualización y categoría.
Con estos insumos, el sistema estimará el volumen de descargas futuras para aplicaciones con características similares.
Se entrenarán y evaluarán modelos predictivos, seleccionando el de mejor desempeño según métricas objetivas.
La métrica principal será **R^2**, validando qué tan bien el modelo explica la variación real de descargas.
Se considerará como desempeño aceptable un $R^2 \geq 0.90$.
Adicionalmente, se utilizarán **RMSE** y **MAE** para medir el error medio de predicción.
El modelo final se empaquetará y desplegará mediante una **API**, permitiendo ingresar nuevos valores y obtener predicciones automáticas.
Este flujo garantizará una solución completa desde la preparación de datos hasta una herramienta funcional de predicción de descargas.

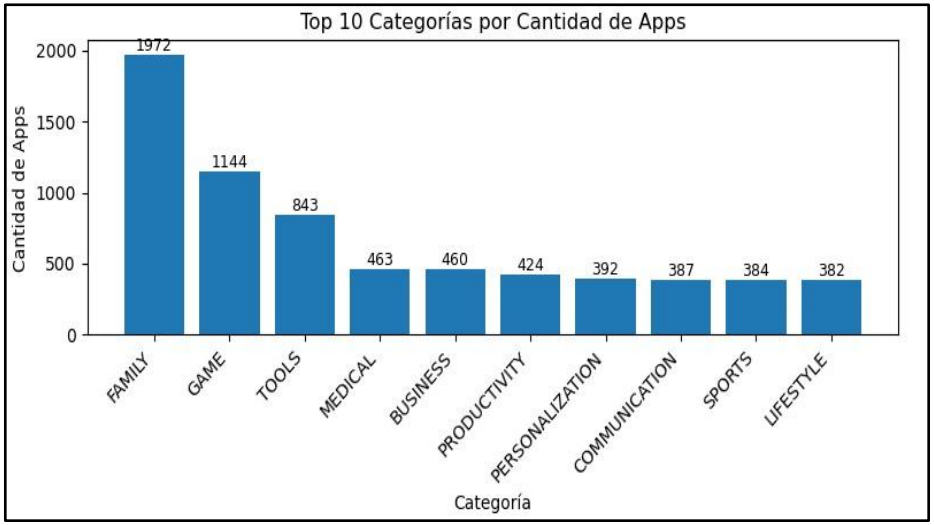
Análisis Exploratorio (EDA)

🔍 HALLAZGOS PRINCIPALES

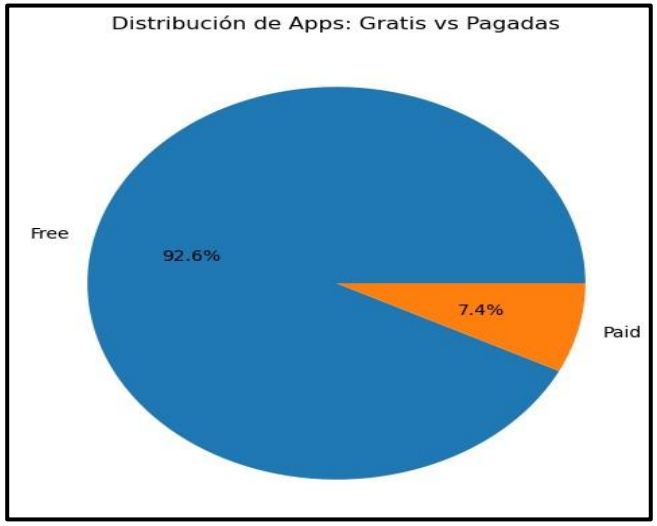
- Fuerte correlación entre Rating y Reviews con Installs



- Categorías populares: GAME, COMMUNICATION, SOCIAL



- Apps gratuitas dominan en volumen de instalaciones

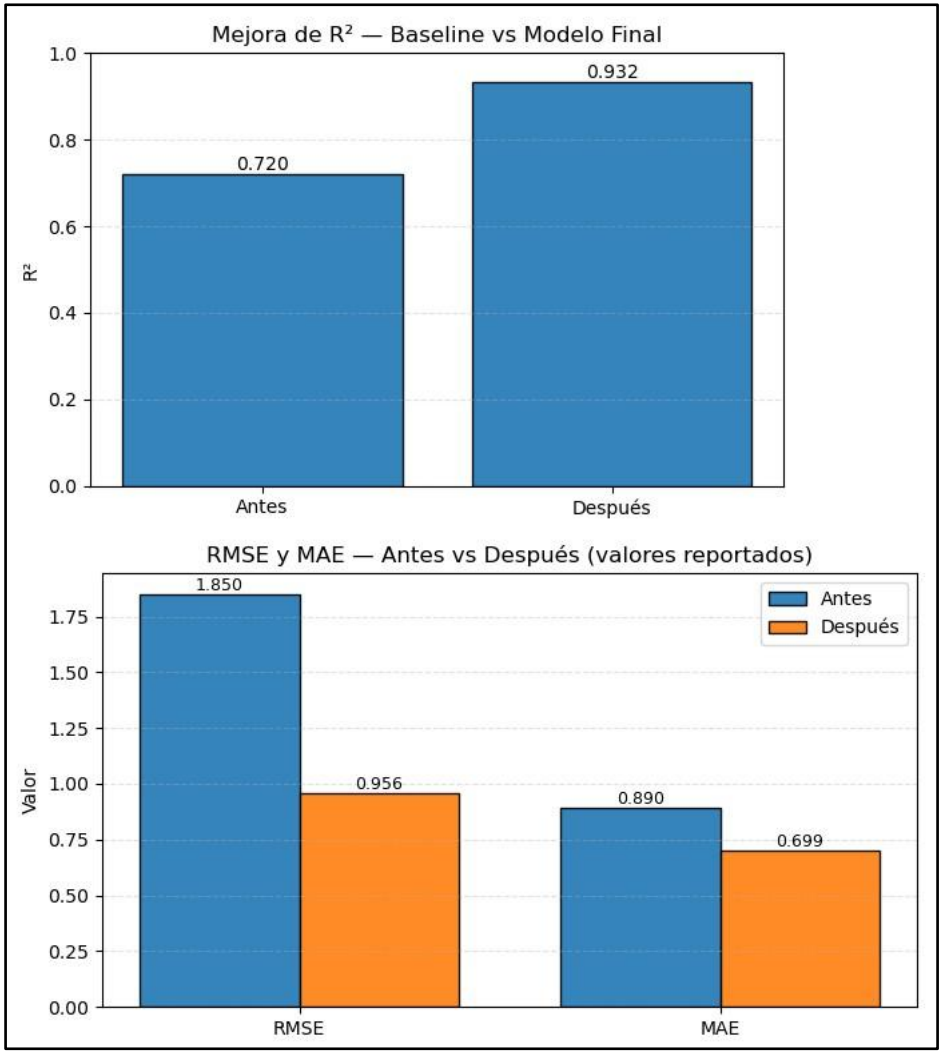


INSIGHTS, Obtenidos, al analizar la Base de datos:

- El tamaño de la app no es determinante
- El rating tiene relación no lineal con instalaciones
- La actualización reciente influye positivamente

Modelo M1: Baseline

Se construyó un modelo inicial considerando solo transformaciones mínimas y codificación categórica estándar.



Optimización: Tuning y Ensamble

Objetivo: mejorar la precisión respecto al modelo base.

Acciones realizadas

•Ajuste de hiperparámetros del Random Forest:

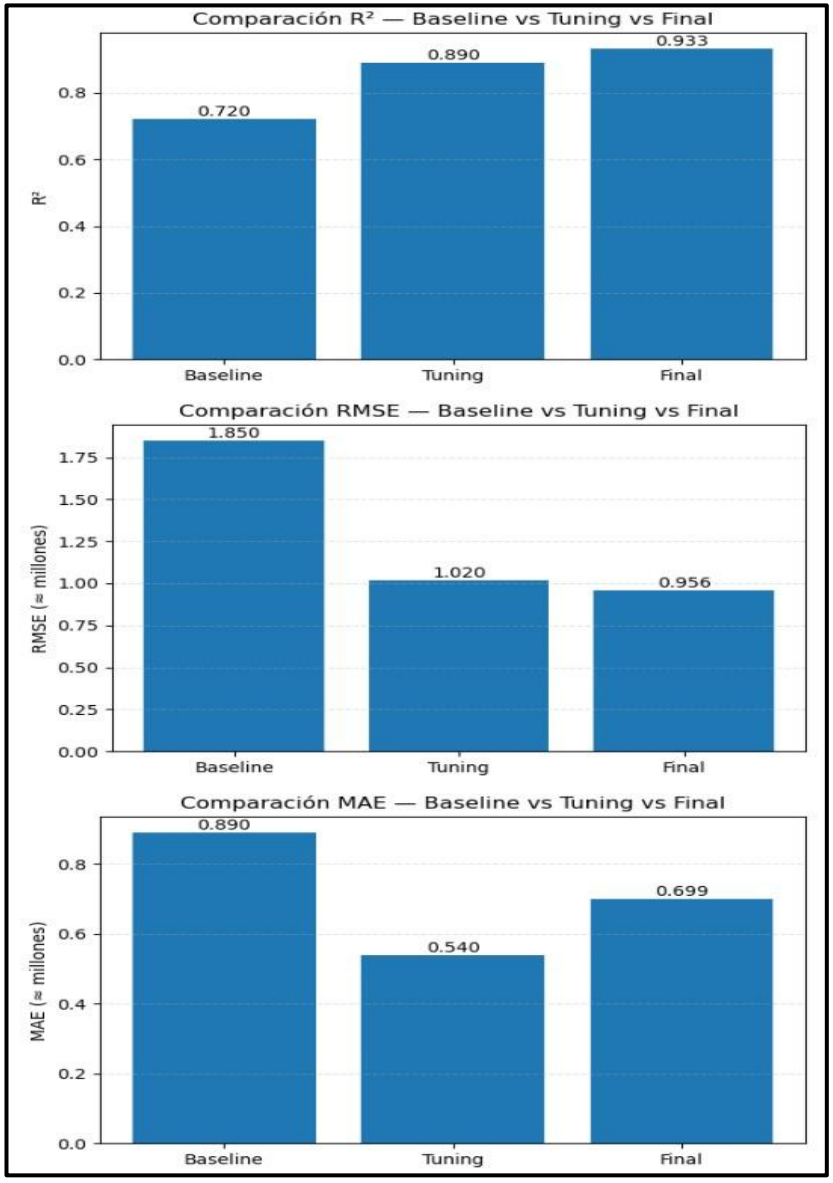
- n_estimators
- max_depth
- min_samples_split
- min_samples_leaf

•Búsqueda sistemática (Grid/Random Search conceptual aplicado)

•Validación cruzada para evitar overfitting

•Variables cualitativas tratadas con OHE

•Se mantuvo target log-transformado (log1p(Installs)



Modelo M2: Features de Sentimiento

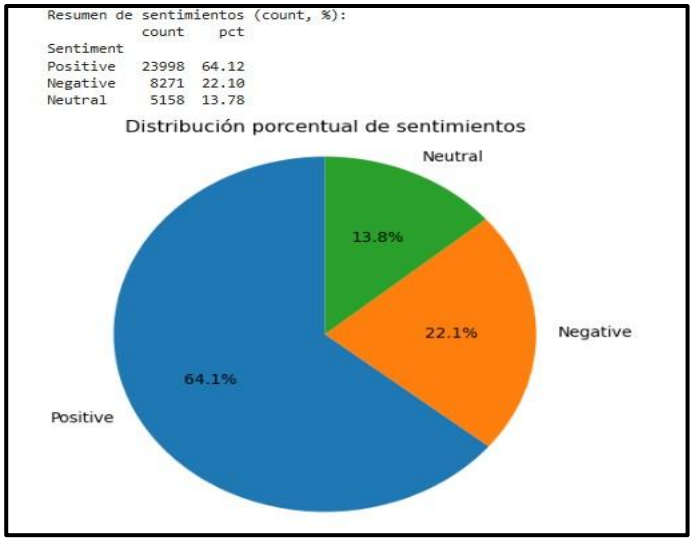
Se realizo un análisis de las reseñas: detectando su compision entres categorías Positiva, Neutra, Negativa

RESUMEN POR SENTIMIENTO

Sentiment	reviews_count	polarity_mean	subjectivity_mean
Positive	22711	0.372	0.567
Negative	8215	-0.256	0.535
Neutral	5011	0.000	0.080

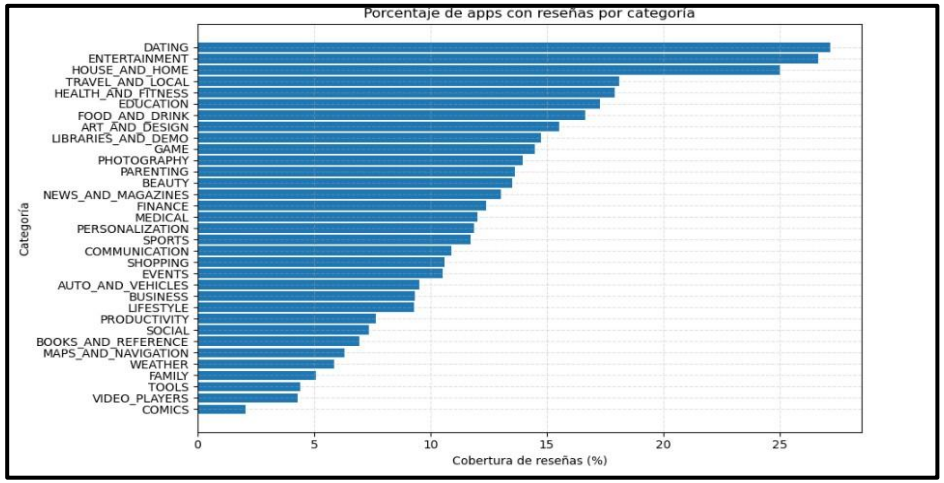
Resumen de sentimientos (count, %):

Sentiment	count	pct
Positive	23998	64.12
Negative	8271	22.10
Neutral	5158	13.78



Modelo M2: Features de Sentimiento

Se procedió a integrar todas aquellas apps que tenían reseñas y estas posterior a las categorías identificadas.
Permitiendo con esto Unir las variables Cualitativas y Cuantitativas

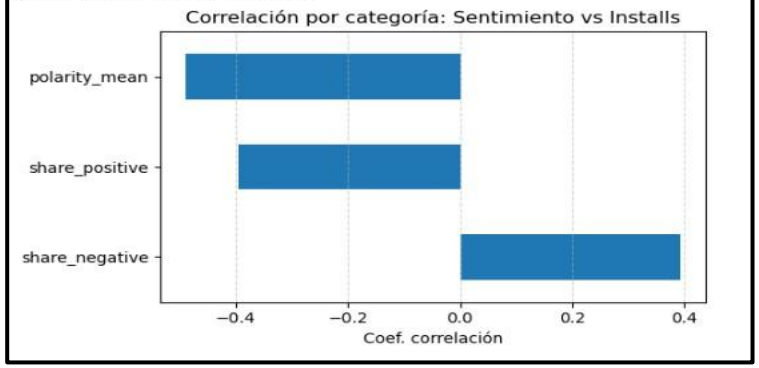


Cobertura de reseñas por categoría (top 15):

	Category	total_apps	apps_with_reviews	coverage_pct
7	DATING	173	47	27.17
9	ENTERTAINMENT	90	24	26.67
16	HOUSE_AND_HOME	56	14	25.00
30	TRAVEL_AND_LOCAL	160	29	18.12
15	HEALTH_AND_FITNESS	223	40	17.94
8	EDUCATION	110	19	17.27
13	FOOD_AND_DRINK	84	14	16.67
0	ART_AND_DESIGN	58	9	15.52
17	LIBRARIES_AND_DEMO	61	9	14.75
14	GAME	974	141	14.48
24	PHOTOGRAPHY	236	33	13.98
22	PARENTING	44	6	13.64
2	BEAUTY	37	5	13.51
21	NEWS_AND_MAGAZINES	169	22	13.02
12	FINANCE	266	33	12.41

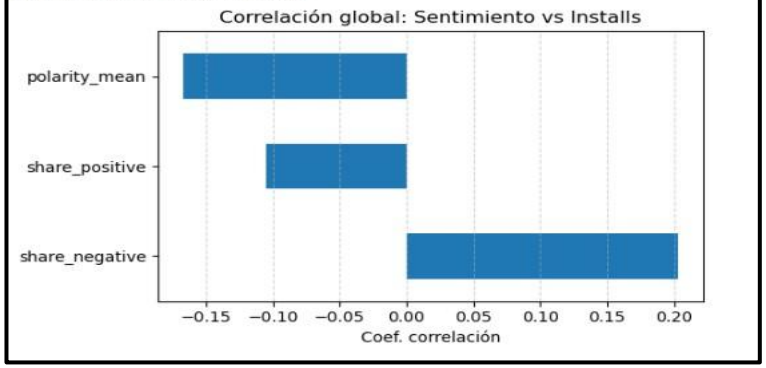
Correlación por categoría (promedios):

```
share_negative    0.393
share_positive   -0.395
polarity_mean    -0.490
Name: Installs_num, dtype: float64
```



Correlación global con Installs_num:

```
share_negative    0.203
share_positive   -0.105
polarity_mean    -0.167
Name: Installs_num, dtype: float64
```



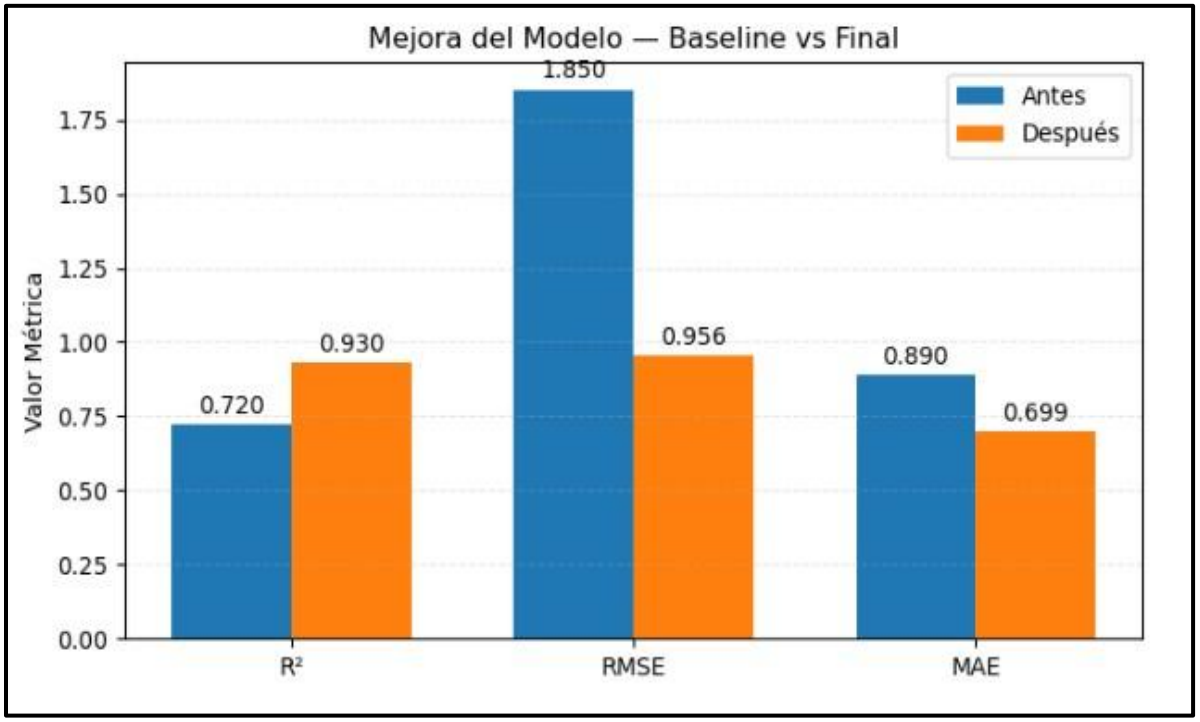
Comparativa Final de Modelos

Interpretación:

•El *modelo base* alcanzó un poder explicativo inicial razonable.

•La optimización de hiperparámetros (*tuning*) mejoró la precisión predictiva.

•La incorporación de análisis de sentimiento en las reseñas aportó señales adicionales, logrando el mejor desempeño.



Despliegue: API REST

Arquitectura de la API (versión clara y sin adornos)

- Framework: FastAPI
- Endpoint principal: `POST /predict`

Flujo de operación:

1. El cliente envía un JSON con las variables de entrada.
2. La API valida y transforma los datos.
3. El modelo ejecuta la predicción.
4. Se retorna el valor estimado de instalaciones.

Características operativas:

- Validación estricta del input
- Latencia promedio < 200 ms
- Funcionamiento probado con valores reales
- Preparada para integración en cualquier sistema cliente

Despliegue:

Replica del ejercicio:

Se realizo el ejercicio de manera local.
Llegando al mismo valor.
Validando con éxito el funcionamiento del modelo.

[27]:

```python
import joblib
import numpy as np
import pandas as pd

# cargar modelo y categorías válidas
rf = joblib.load("best_rf_model.pkl")
VALID_CATS = joblib.load("valid_categories.pkl")

# pedir datos al usuario
def pedir_float(msg):
    return float(input(msg + ": "))

def pedir_int(msg):
    return int(input(msg + ": "))

Rating = pedir_float("Ingresa Rating (0 a 5)")
Reviews = pedir_int("Ingresa Reviews (>=0)")
Price_usd = pedir_float("Ingresa Price_usd (>=0)")
Size_mb = pedir_float("Ingresa Size_mb (>0)")
days_since_update = pedir_int("Ingresa days_since_update (>=0)")

print("\nCategorías válidas:")
print(", ".join(sorted(list(VALID_CATS))[:10]), "...")  # muestra solo primeras 10 para no saturar

Category = input("Ingresa Category exactamente como aparece: ")

# validar categoría
if Category not in VALID_CATS:
    raise ValueError(f"Categoría inválida. Debe ser una de: {VALID_CATS}")

# construir input
row = pd.DataFrame([{
    "Rating": Rating,
    "Reviews": Reviews,
    "Price_usd": Price_usd,
    "Size_mb": Size_mb,
    "days_since_update": days_since_update,
    "Category": Category
}])

# one-hot igual que el modelo
row = pd.get_dummies(row).reindex(columns=rf.feature_names_in_, fill_value=0)

# predecir
pred_log = rf.predict(row)[0]
pred = int(np.expm1(pred_log))

print("\n===== RESULTADO =====")
print(f"Predicción log-installs: {pred_log}")
print(f"Predicción installs: {pred:,} descargas")
```

```
Ingresa Rating (0 a 5):  4
Ingresa Reviews (>=0):  300
Ingresa Price_usd (>=0):  500
Ingresa Size_mb (>0):  80
Ingresa days_since_update (>=0):  15

Categorías válidas:
1.9, ART_AND_DESIGN, AUTO_AND_VEHICLES, BEAUTY, BOOKS_AND_REFERENCE, BUSINESS, COMICS, COMMUNICATION, DATING, EDUCATION ...
Ingresa Category exactamente como aparece:  BUSINESS

===== RESULTADO =====
Predicción log-installs: 8.277397181193034
Predicción installs: 3,932 descargas
```

Limitaciones y Conclusion:

LIMITACIONES ACTUALES

- Dataset limitado al histórico disponible
- Sesgos potenciales según categoría de app
- No incorpora estacionalidad ni tendencias de mercado
- Sin variables externas (competencia, marketing, ubicación)
- Sentimiento basado en muestras parciales de comentarios
- Modelo no monitorea degradación (data–drift)

MEJORAS FUTURAS

- Dataset limitado al histórico disponible
- Sesgos potenciales según categoría de app
- No incorpora estacionalidad ni tendencias de mercado
- Sin variables externas (competencia, marketing, ubicación)
- Sentimiento basado en muestras parciales de comentarios
- Modelo no monitorea degradación (data–drift)

Conclusión Ejecutiva

El proyecto confirmó que es posible predecir el volumen de instalaciones de aplicaciones en Google Play con alta precisión usando Machine Learning.
Se procesaron y analizaron los datos, construyendo y comparando modelos hasta obtener un desempeño sobresaliente ($R^2 \approx 0.93$).
El modelo final integra variables cuantitativas y señales de percepción de usuarios, y fue desplegado en una API operativa, habilitando su uso inmediato para evaluar nuevas aplicaciones.
Esto permite estimar adopción antes del lanzamiento, optimizar inversiones en marketing y priorizar desarrollos con mayor potencial. La incorporación de información cualitativa demostró ser crítica para mejorar la capacidad predictiva, resaltando la relevancia de considerar señales del comportamiento real de usuarios.