DESAFÍO TÉCNICO MERCADO LIBRE

En el contexto del Marketplace de Mercadolibre, se necesita un algoritmo para predecir si un artículo publicado en el mercado es nuevo o usado. Su tarea consiste en diseñar un modelo de aprendizaje automático para predecir si un artículo es nuevo o usado y luego evaluarlo con datos de prueba. Para facilitar esta tarea, se proporciona un conjunto de datos en `MLA_100k.jsonlines` y una función para leer dicho conjunto de datos en `build_dataset`. Para la evaluación, deberá elegir una métrica adecuada y también argumentar por qué se eligió dicha métrica.
Los entregables son:

- Este archivo incluye todo el código necesario para definir y evaluar un modelo.
- Un archivo de texto con una breve explicación de los criterios aplicados para elegir la métrica y el rendimiento alcanzado con ella.

Dataset:
Conjunto de datos de train con un conjunto de items del Marketplace de Mercado Libre ya etiquetados como “New” o “Used” y un conjunto de datos de test para evaluar el modelo.
Train: 90000 items, 40 columnas de datos.
Test: 10000 items, 39 columnas de datos.

Formato dataset:
Como el formato en que se disponibilizan los datos es una lista de jsons, y en algunos casos anidados, se formatean por medio de una función para obtener un dataframe, generando diferentes columnas en base a los json y en algunos casos se generan variables derivadas como:

- age_days: antigüedad del anuncio, en días (derivada de ‘last_updated’ y ‘date_created’)
- title_len: longitud del título (número de caracteres)
- title_has_new: `True` si el título contiene “nuevo/nueva/new”
- title_has_used: `True` si el título contiene “usado/used”
- price_log: logaritmo natural del precio
- sold_ratio: proporción de ventas
- n_pictures: número total de fotos
- max_picture_px: resolución máxima de las fotos (ancho × alto)
- has_secure_picture: `True` si al menos una foto tiene URL segura (https)

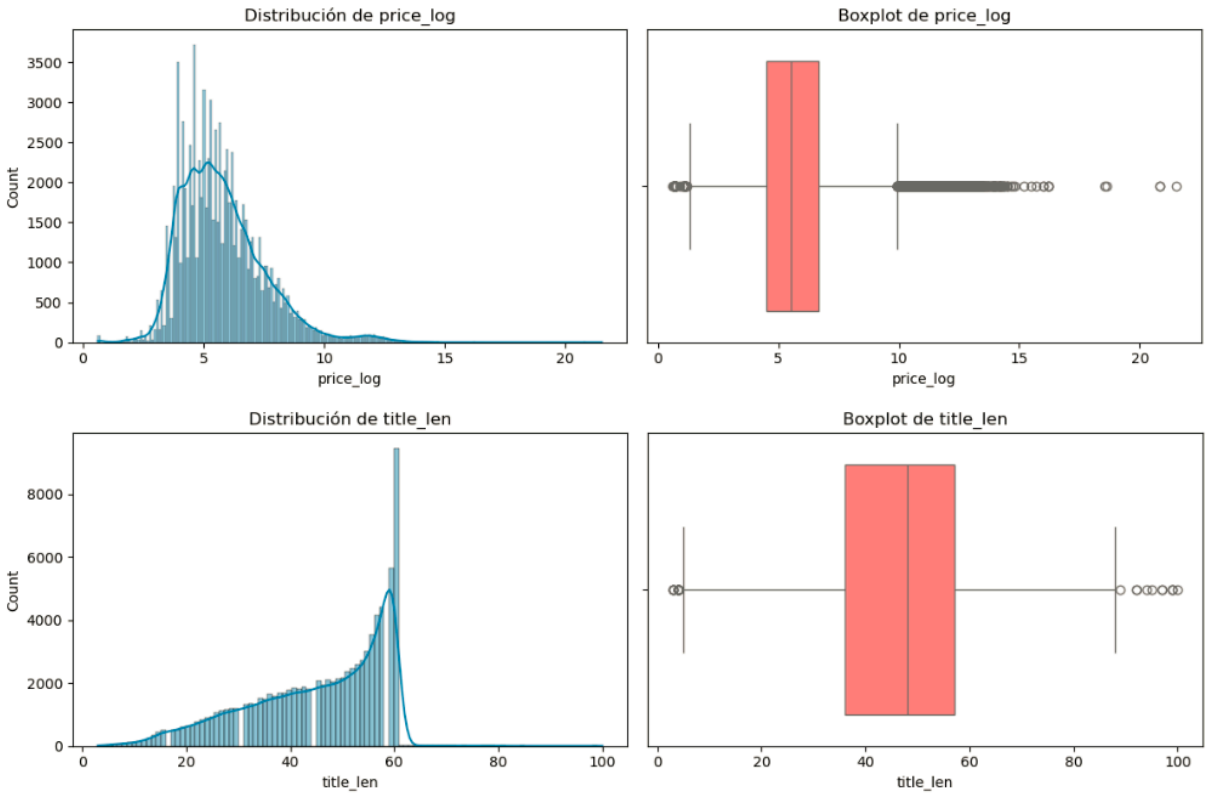
No se incluyen las variables anidadas como “attributes”, debido a la disparidad de los campos como marca, color, y gran cantidad de nulos, cercanos al 100%.

Valores nulos:

- Vemos que la variable 'original_price' tiene nulos tanto en X_train_df como en X_test_df. Con un porcentaje de casi el 100% de nulos en ambos casos.
- Se deciden crear dos nuevas variables:
  - `has_discount` = True solo si original_price no es nulo.
  - `discount_pct` = NaN si original_price es nulo o ≤0, o si no hay diferencia válida con price.

Y luego eliminar la columna 'original_price' por tener alta correlación con 'discount_pct'.

Análisis univariado: Distribuciones y outliers
Para las variables numéricas se analizan sus distribuciones y boxplot para ver la existencia de outliers.



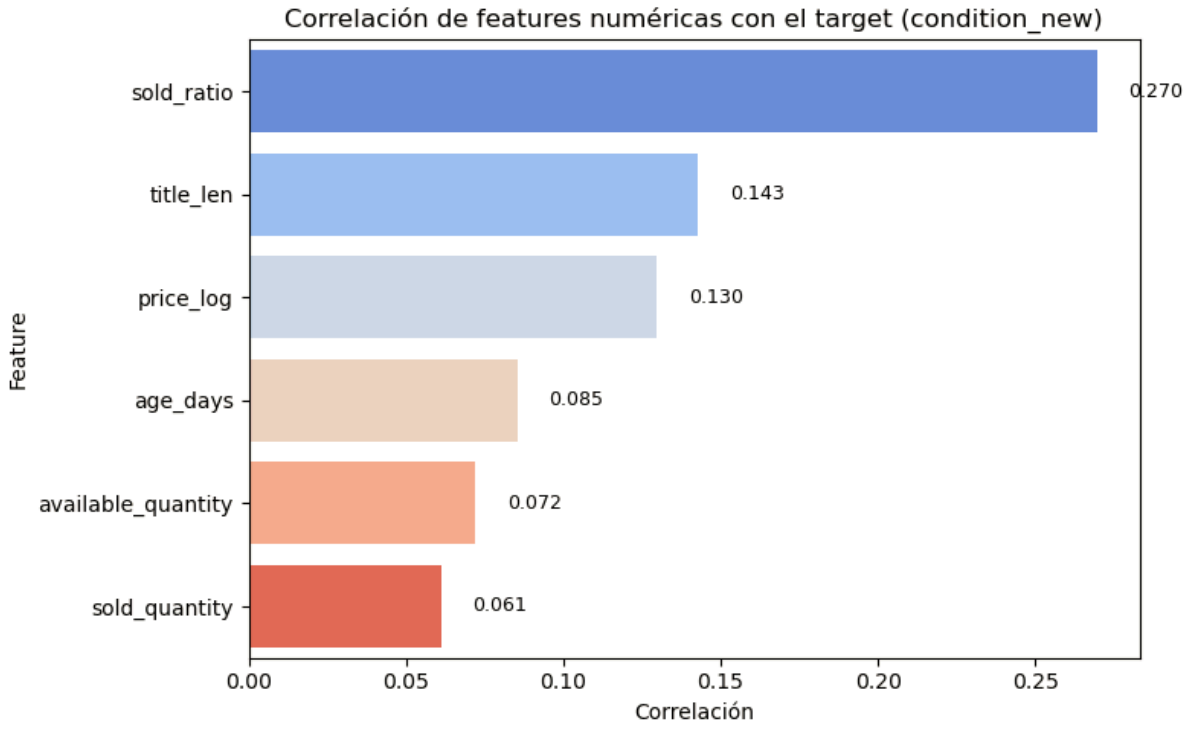
- Se observa mayor cantidad de outliers en las variables: 'price', 'sold_quantity', 'available_quantity', 'initial_quantity', 'age_days', 'price_log', 'sold_ratio'.
- Y en menor cantidad se observan outliers en la variable 'title_len'.
- Por el momento se deciden dejar dichos outliers, hasta realizar un análisis multivariado con respecto a la variable target y ver su correlación.
- Variables numéricas con gran amplitud serán escaladas/normalizadas.
- También analizaremos su varianza y multicolinealidad, para ver si hay variables a eliminar.

Análisis multivariado: con respecto al target 'condition'

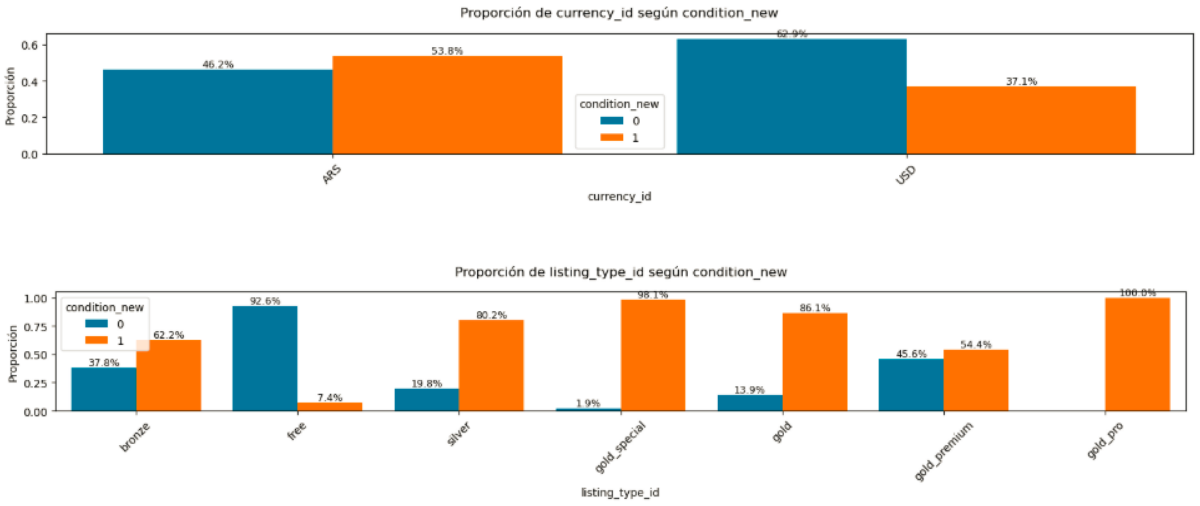
<u>Variables numéricas:</u>

- No tener en cuenta para el modelo las variables constantes (varianza cero), ya que no aportan información al modelo:
  - `n_pictures`
  - `max_picture_px`
  - `n_other_payments`
- Eliminación de la multicolinealidad. La idea es que si dos variables están altamente correlacionadas (p. ej., corr > 0,9), solo nos queda una:

- Eliminamos `initial_quantity` porque está altamente correlacionada con `available_quantity`.
- Mantener sólo las variables numéricas con alta correlación con el target
    - sold_ratio
    - title_len
    - price_log
    - age_days
    - available_quantity
    - sold_quantity



<u>Variables categóricas y booleanas</u>:
Realizamos visualizaciones con gráficos de barra para ver su distribución con respecto a la variable target.



Variables eliminadas y motivo:

- title: tiene mucha variabilidad (89008 casos únicos).
- price: baja correlación con el target.
- category_id: tiene mucha variabilidad (10491 casos únicos).
- site_id: es constante ('MLA').
- initial_quantity: Eliminamos `initial_quantity` porque está altamente correlacionada con `available_quantity`.
- date_created: Se deja la variable derivada 'age_days'.
- last_updated: Se deja la variable derivada 'age_days'.
- stop_time: Se deja la variable derivada 'age_days'.
- start_time: Se deja la variable derivada 'age_days'.
- seller_country: Es casi constante con Country=AR, y 1 caso con ''.
- seller_state: tiene variabilidad (25 casos únicos). Se podría pensar en dejar las más frecuentes y poner una categoría 'others'
- seller_city: tiene mucha variabilidad (89008 casos únicos).
- seller_lat: tiene mucha variabilidad.
- seller_long: tiene mucha variabilidad.
- shipping_local_pick_up: Hay variaciones mínimas con respecto al target, eliminar la variable.
- n_descriptions: baja correlación con el target.
- n_other_payments: es constante, varianza cero.
- n_pictures: es constante, varianza cero.
- has_secure_picture: es constante en False, varianza cero.
- max_picture_px: es constante, varianza cero.
- has_cash: Es constante en False.
- has_bank_transfer: Es constante en False.
- discount_pct: casi nula correlación con el target.

Se mantienen las siguientes 17 variables:

Numéricas

- 'sold_ratio',
- 'title_len',
- 'price_log',
- 'age_days',
- 'available_quantity',
- 'sold_quantity'

Categóricas

- 'currency_id',
- 'listing_type_id',
- 'buying_mode',
- 'status',
- 'accepts_mercadopago',
- 'shipping_mode'

Booleanas

- 'shipping_free_shipping',
- 'title_has_new',
- 'title_has_used',
- 'has_discount'

Target booleano

- 'condition_num'

Transformación y escalado de variables numéricas

Ya que la mayoría de las variables numéricas están altamente sesgadas y presentan outliers extremos, utilizaremos métodos de transformación y escalado para mejorar la performance de los modelos.
La transformación debe realizarse tanto en train como en test.

Se aplica RobutScaler a las variables numéricas.

Codificación de variables categóricas

- Debemos pasar las variables categóricas a numéricas por medio de encoding, ya que los modelos trabajan con variables numéricas.
- Aplicarlo tanto a train como a test.
- Codificamos todas las variables categóricas de un usando **OneHotEncoder** con *drop='first'* para evitar multicolinealidad.
- Quedaron 27 variables, todas numéricas:
    - 'sold_ratio',
    - 'title_len',
    - 'price_log',
    - 'age_days',
    - 'available_quantity',
    - 'sold_quantity',
    - 'accepts_mercadopago',
    - 'shipping_free_shipping',
    - 'title_has_new',
    - 'title_has_used',
    - 'has_discount',
    - 'currency_id_USD',
    - 'listing_type_id_free',
    - 'listing_type_id_gold',
    - 'listing_type_id_gold_premium',
    - 'listing_type_id_gold_pro',
    - 'listing_type_id_gold_special',
    - 'listing_type_id_silver',
    - 'buying_mode_buy_it_now',
    - 'buying_mode_classified',
    - 'status_closed',

- 'status_not_yet_active',
- 'status_paused',
- 'shipping_mode_me1',
- 'shipping_mode_me2',
- 'shipping_mode_not_specified'
- 'condition_num',

Modelo baseline

Métricas del modelo

En el caso que estamos prediciendo si un item en venta es nuevo/usado, tenemos dos enfoques posibles:

- Si el modelo se va a usar para detectar productos nuevos con confianza, conviene priorizar alto el 'precision'.
- Si en cambio lo importante es no perder ningún producto nuevo (aunque se cuelen usados), entonces priorizamos tener alto el 'recall'.

Como no tenemos claro estos objetivos de negocio vamos a optar por una métrica balanceada:

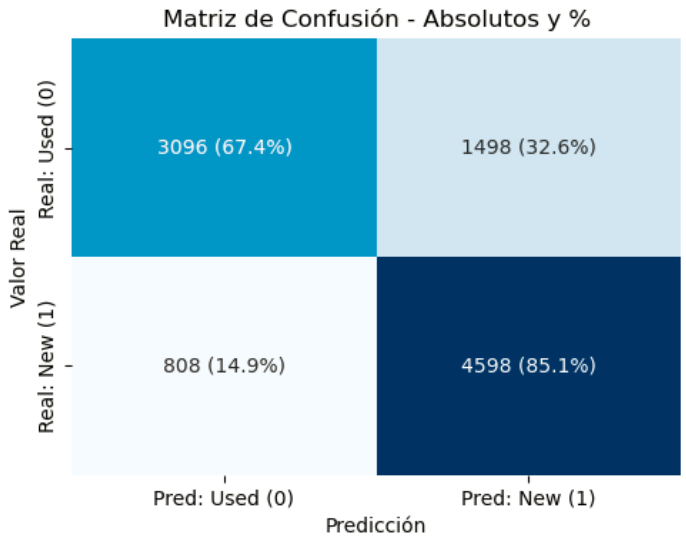
- **Maximizar F1-score**: Se elige el umbral donde la combinación de 'precision' y recall está más balanceada. Es decir, queremos un equilibrio entre detectar la mayor cantidad posible de productos nuevos ('recall'), pero sin etiquetar demasiados productos usados como si fueran nuevos ('precision')

Aplicamos como modelo "**Baseline**", una **Regresión Logística**, para predecir la variable binaria "condition_num". Ya que es un modelo simple, rápido e interpretable. Y suele ser una buena base para problemas binarios.

Resultados Modelo Baseline de Regresión Logística:

Métricas de test

```
=== Logistic Regression best F1 threshold ===
ROC-AUC: 0.8570843139992955

Classification Report:
              precision    recall  f1-score   support

           0       0.79      0.67      0.73      4594
           1       0.75      0.85      0.80      5406

    accuracy                           0.77     10000
   macro avg       0.77      0.76      0.76     10000
weighted avg       0.77      0.77      0.77     10000
```



Conclusiones del modelo de regresión logística:

- **TN (arriba-izquierda)**: un 67,4% de ítems usados fueron correctamente clasificados como usados, esto evita prometer “nuevo” cuando no lo es (protege la confianza).

- **FP (arriba-derecha)**: un 32,6% de los usados fueron clasificados como nuevos, esto representa un *costo de reputación* (riesgo de falsas promesas). Pero el porcentaje es bajo o aceptable.

- **FN (abajo-izquierda)**: un 14,9% de ítems nuevos fueron clasificados como usados, esto implica una *pérdida de oportunidad* (podríamos no destacar un producto que sí es nuevo).

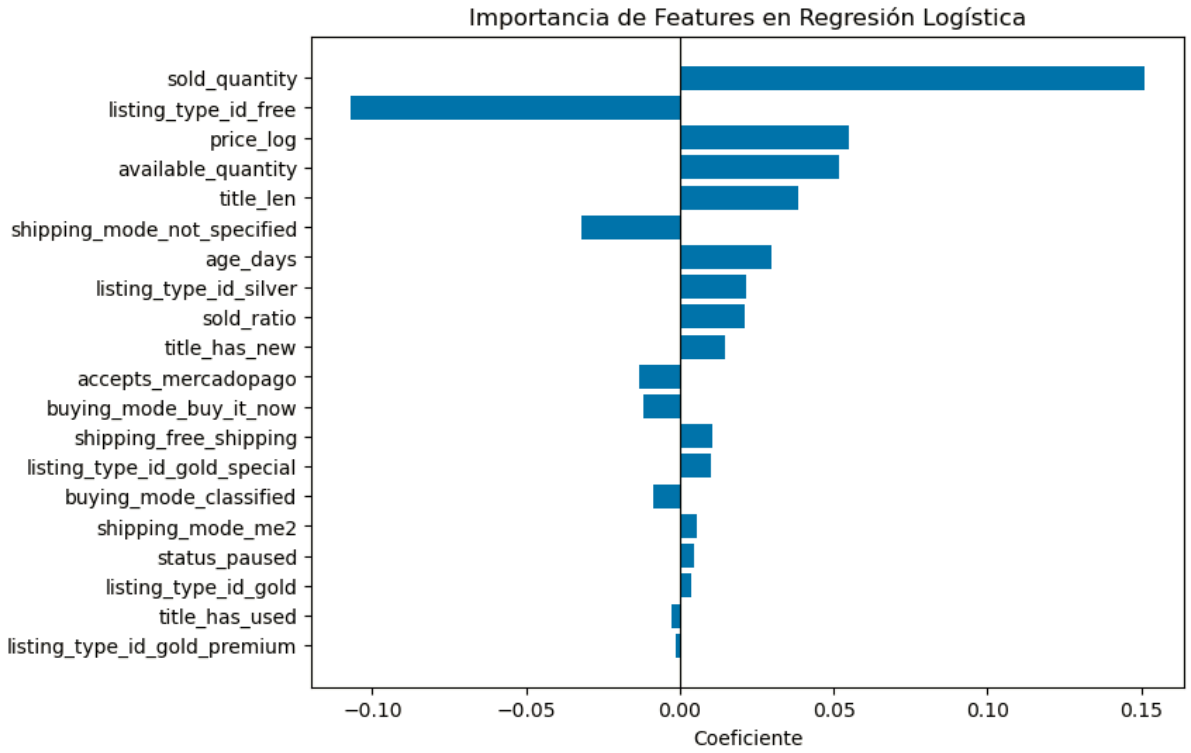
- **TP (abajo-derecha)**: un 85,1% de los ítems nuevos fueron correctamente clasificados, este es nuestro objetivo principal ya que quisiéramos promover productos nuevos.

- **El modelo es mejor detectando productos nuevos que usados.**
    - Alta recall (85%) en “new”: casi todos los productos nuevos reales se identifican como tal.
    - Esto es positivo si tu estrategia es destacar productos nuevos y evitar perderlos de vista.

- El costo está en algunos “usados” que se confunden como “nuevos” (precisión 75% en clase “new”).
    - Estos falsos positivos son un riesgo: mostrar productos usados como “nuevos” puede afectar la confianza del cliente.

- Balance general aceptable (77% accuracy, 0.80 F1 en “new”).
    - El modelo encuentra un compromiso entre no dejar escapar demasiados nuevos y no sobre-inflar la categoría.
    - El F1 se maximizó para dar equilibrio entre precisión y recall.

Feature importance del modelo de Regresión Logística:



- Los productos con alto volumen de ventas, precio más alto, gran stock y publicaciones pagas/pro son los que el modelo más fuertemente asocia a “nuevo”.

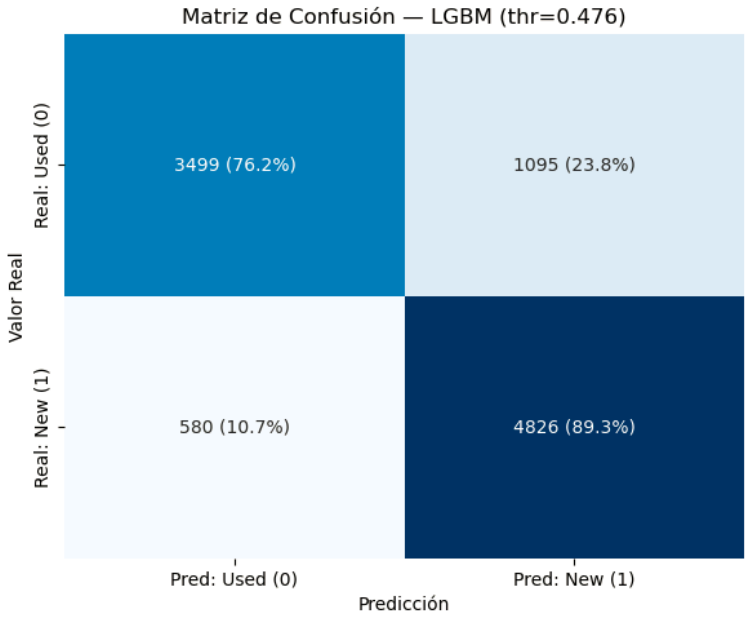
- En cambio, publicaciones con bajo stock, gratis, sin especificar envío, o con poca estructura en la publicación se asocian más a “usado”.

Modelo LigthGBM y comparación de métricas

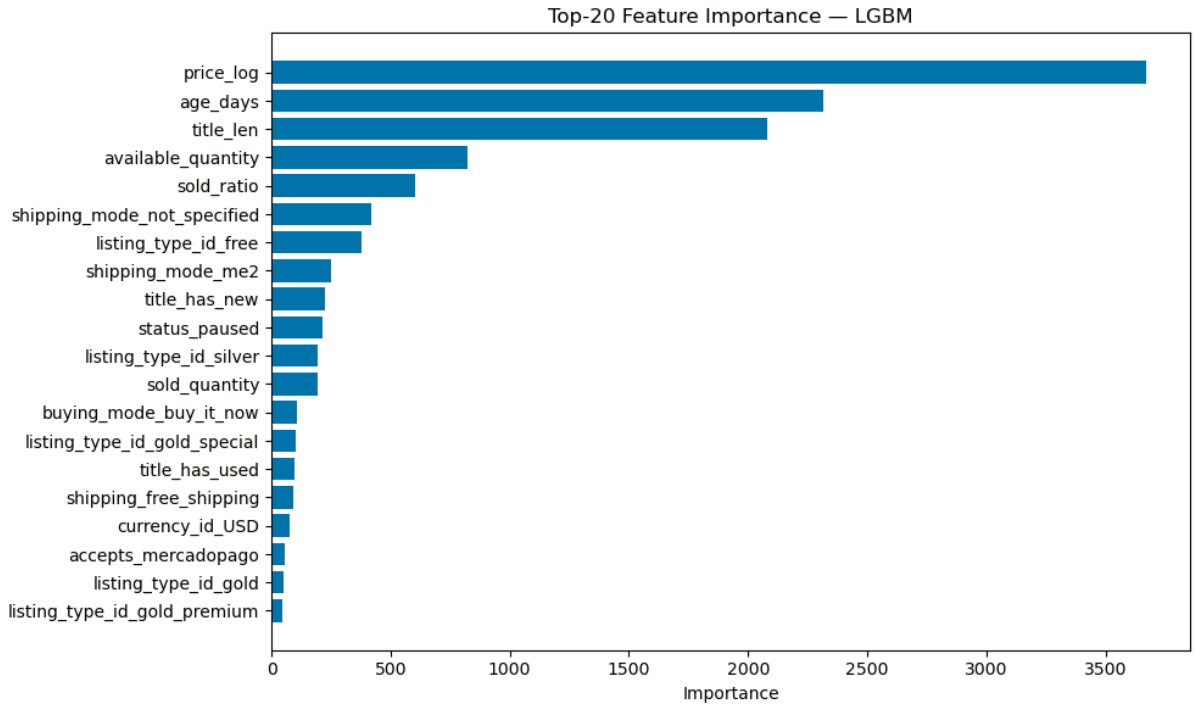
Métricas de test

```
ROC-AUC: 0.9257608687424008

Classification Report:
              precision    recall  f1-score   support

           0       0.86      0.76      0.81      4594
           1       0.82      0.89      0.85      5406

    accuracy                           0.83     10000
   macro avg       0.84      0.83      0.83     10000
weighted avg       0.83      0.83      0.83     10000
```

- Observamos que el modelo LGBM presenta una mejora en el ROC-AUC (de 0.85 a 0.92)
- También el F1 para la clase ‘new’ mejoró (de 0.80 a 0.83)
- F1-Score (equilibrio precisión/recall):
    - Logística: F1 promedio ~0.77
    - LGBM: F1 promedio ~0.83, clara mejora, sobre todo para clase 1 (nuevo).



- Los ítems usados que fueron correctamente clasificados pasaron de un 67,4% a 76,2%.
- Los ítems nuevos que fueron correctamente clasificados pasaron de un 85,1% a 89,3%.



- En cuanto a los features importance de LGBM, algunas observaciones:
    - El precio es el factor principal para diferenciar productos nuevos y usados. En general, los productos nuevos tienden a tener precios más altos, mientras que los usados suelen estar más baratos.
    - Los productos que llevan más tiempo publicados suelen ser usados. Los ítems nuevos tienden a renovarse o venderse más rápido.
    - available_quantity (stock disponible). Stock alto, es más probable que sea nuevo (suelen tener inventario en serie). Usados suelen tener stock único o muy bajo.

En conclusión: **LGBM supera ampliamente a la regresión logística en todas las métricas relevantes**, debería ser un buen modelo candidato para producción.

Posibles mejoras:

- Aplicar TF-IDF a la variable title, para no perder su información y tenerla en formato numérico. Este método reduce el peso de palabras comunes y resalta las que distinguen nuevo vs usado. (Opcional: Embeddings - Word2Vec, BERT, etc.- capturan aún más semántica, pero requieren más cómputo).
- Trabajar en el feature engineering para crear nuevas variables en base a las existentes que presenten mayor correlación con la variable target.
- Tratamiento de outliers, ya que después de escalar, siguen reflejándose en algunas variables numéricas.
- Prueba de otros modelos como LGBM o incluso aplicar ajuste por hiperparámetros, para intentar obtener mejoras.
- Análisis de feature importances para tratar de reducir la cantidad de variables que están tomando los modelos.
- Comparar las métricas de train y de test para verificar que no haya overfitting (altas en train, pero más bajas en test).
- Análisis de interpretabilidad de modelos con librerías como SHAP, ésto nos permite analizar las predicciones individuales. Por ejemplo responde a la pregunta: *“¿Para este producto en particular, qué variables hicieron que lo clasifiquemos como nuevo/usado?”* o para explicar por ej: por qué un producto fue mal clasificado.