Memoria del Módulo: Clasificación de Precios de Cartas Pokémon TCG

-Introducción al Módulo

Este documento detalla el desarrollo, la implementación y la evaluación del módulo de clasificación de precios para cartas de Pokémon Trading Card Game (TCG). El objetivo específico de este módulo es categorizar el precio_actual de una carta en una de cuatro clases predefinidas: 'Bulk', 'Asequible', 'Cara' o 'Coleccionista'. Esto proporciona una valoración granular, esencial para funcionalidades de filtrado y recomendación en el sistema principal del proyecto.

-Fase 1: Adquisición y Preparación de Datos

En esta fase, el enfoque fue en consolidar las características necesarias para la clasificación a partir de datos BigQuery.

1.1. Fuente de Datos y Características Iniciales

Se recuperan metadatos y precios (actuales y del mes previo) de las tablas de BigQuery. Los precios del último snapshot (t0) y del snapshot anterior (t-1) son fundamentales.

1.2. Ingeniería de Características Clave

Se crean dos características esenciales para el modelo de clasificación:

price_change_from_prev: Representa el cambio porcentual de precio (price_t0 - price_t_minus_1) / price_t_minus_1. Se imputan los valores nulos para garantizar la completitud.

price_category_target: Nuestra variable objetivo. Se clasifica el price_t0 de cada carta en 'Bulk' (<5€), 'Asequible' (<50€), 'Cara' (<200€) o 'Coleccionista' (>=200€) usando umbrales predefinidos. Se excluyen los registros sin una categoría clara.

Las características de entrada para el modelo se definieron globalmente para consistencia:

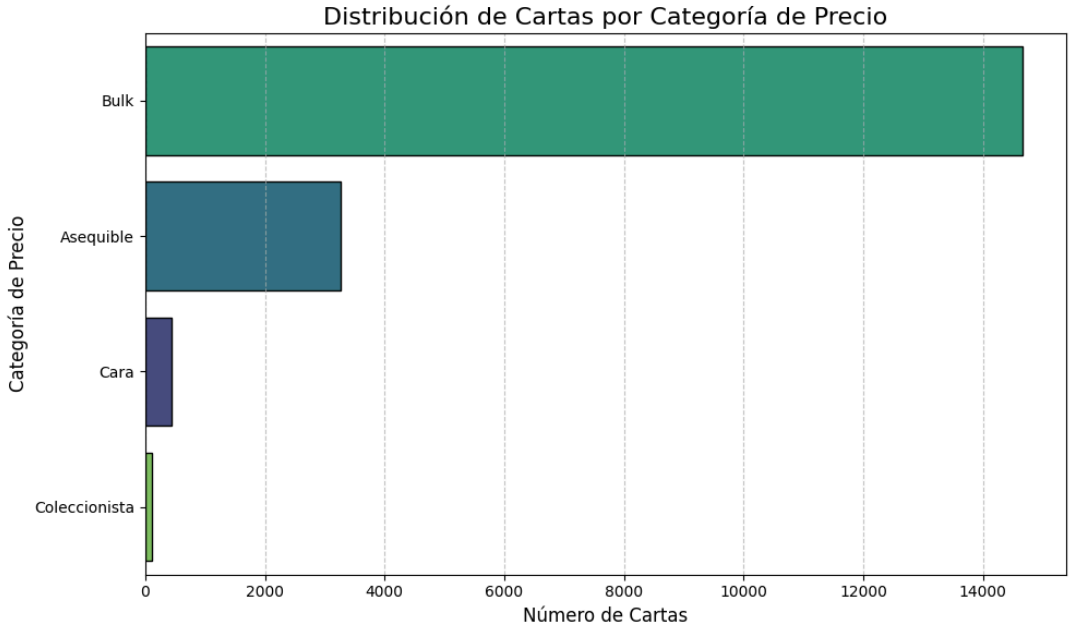
- **Numéricas**: price_t_minus_1, price_change_from_prev.
- **Categóricas**: artist_name, pokemon_name, rarity, set_name, types, supertype, subtypes.

-Fase 2: Análisis Exploratorio de Datos (EDA)

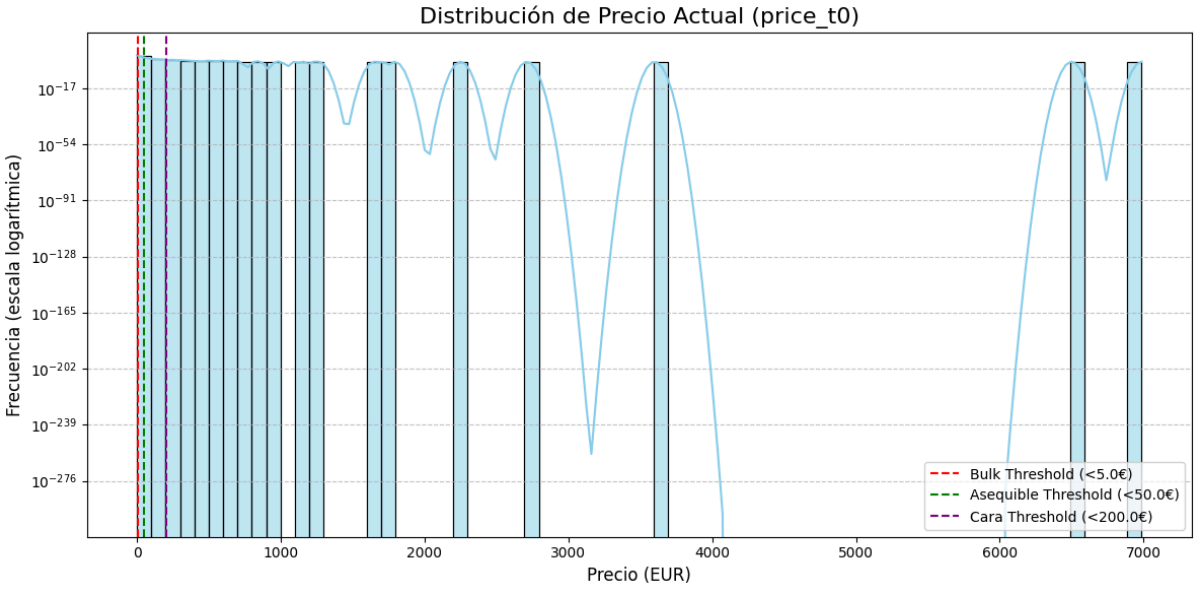
El EDA fue crucial para entender la distribución y las relaciones de los datos, guiando las decisiones de preprocesamiento y modelado.

2.1. Hallazgos Clave y Justificaciones

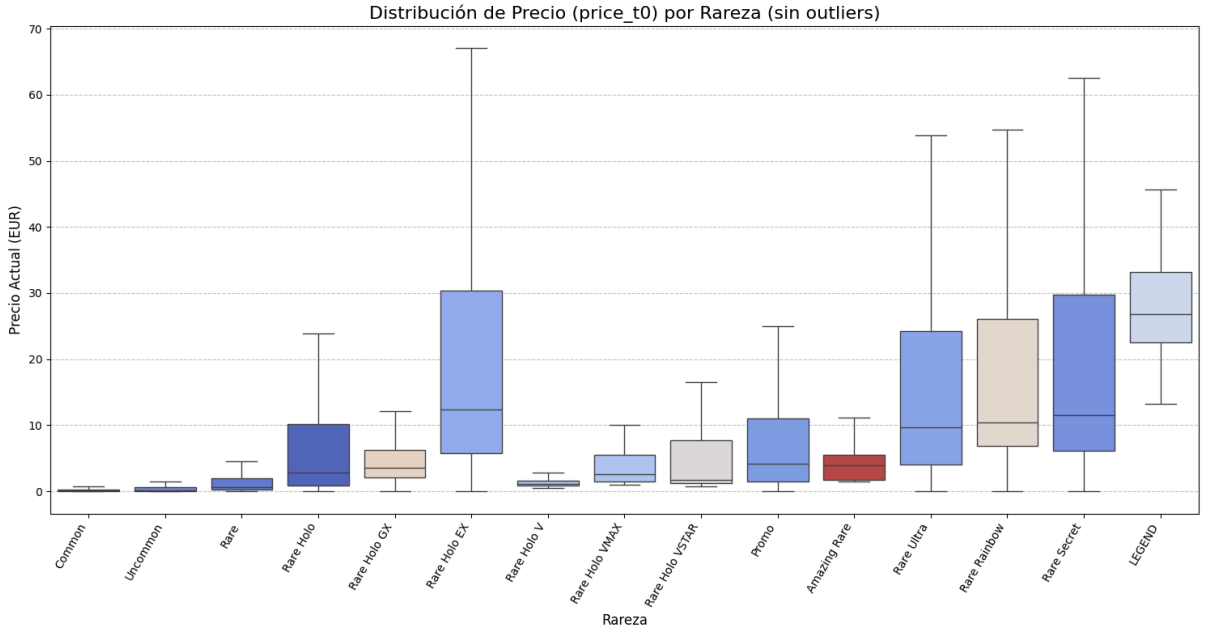
- Desbalance de Clases: El análisis de price_category_target reveló un desbalance severo (e.g., 'Bulk' >> 'Coleccionista').
    - Implicación: La evaluación del modelo debe priorizar métricas por clase (Precision, Recall, F1-score) y el F1-score ponderado (weighted avg), en lugar de solo Accuracy. El stratify en la división de datos es esencial.

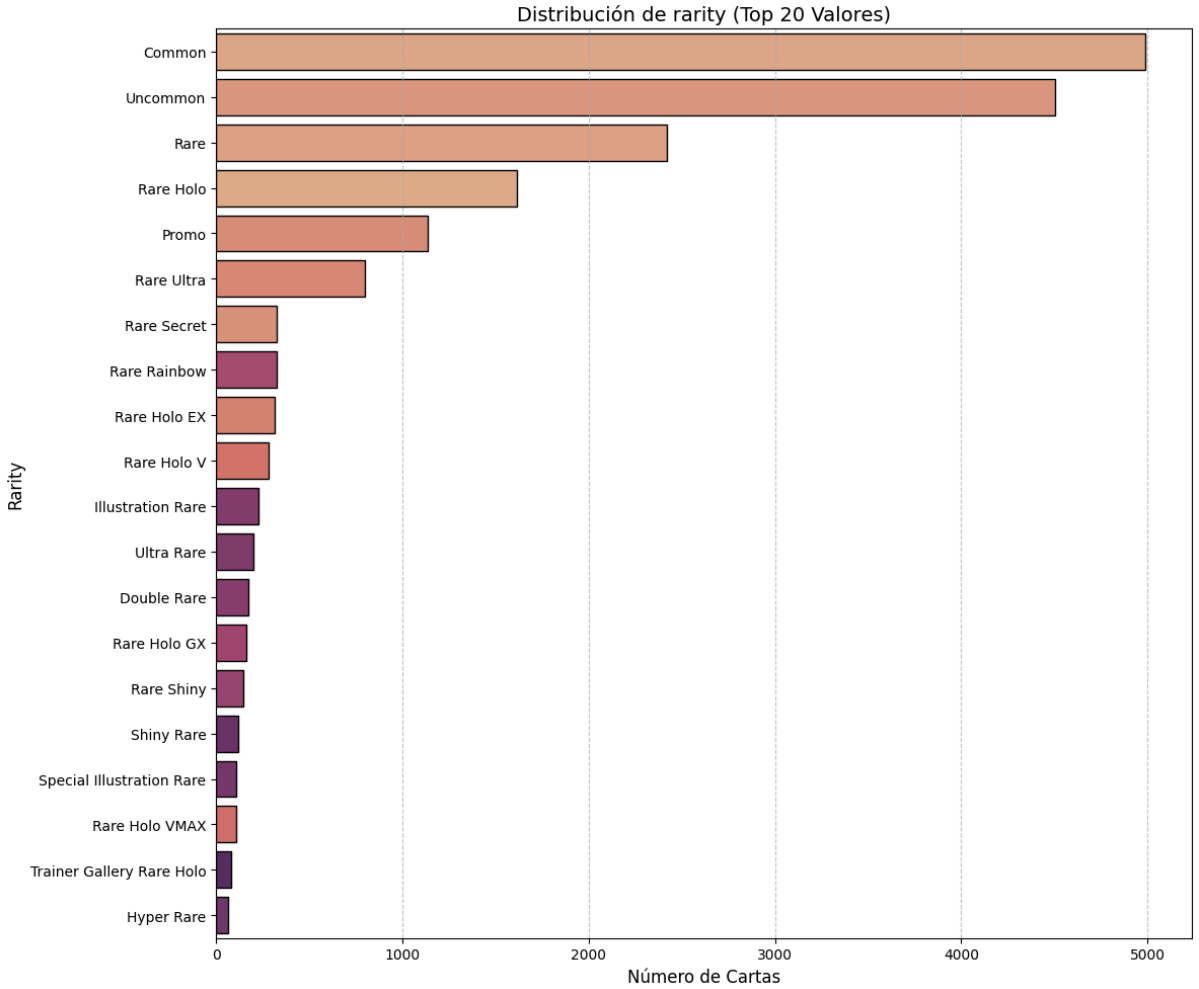


- Distribución Numérica: price_t0 y price_t_minus_1 están fuertemente sesgados a la derecha (muchas cartas baratas). price_change_from_prev se concentra en cero.
    - Implicación: Se justifica el escalado de variables numéricas (con StandardScaler) para algoritmos sensibles a la magnitud.

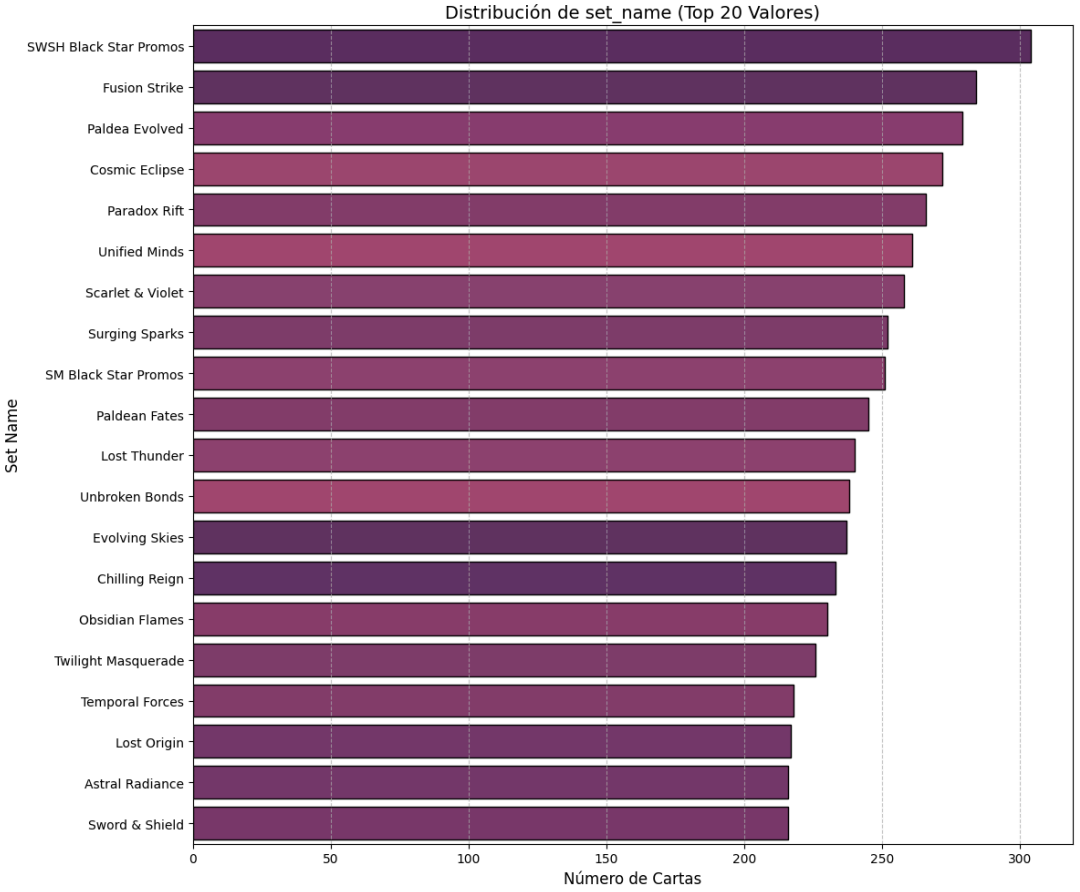


- Influencia Categórica: rarity, supertype y set_name muestran una correlación directa y fuerte con el precio de la carta.
    - Implicación: Son características altamente predictivas. Su codificación adecuada es vital.

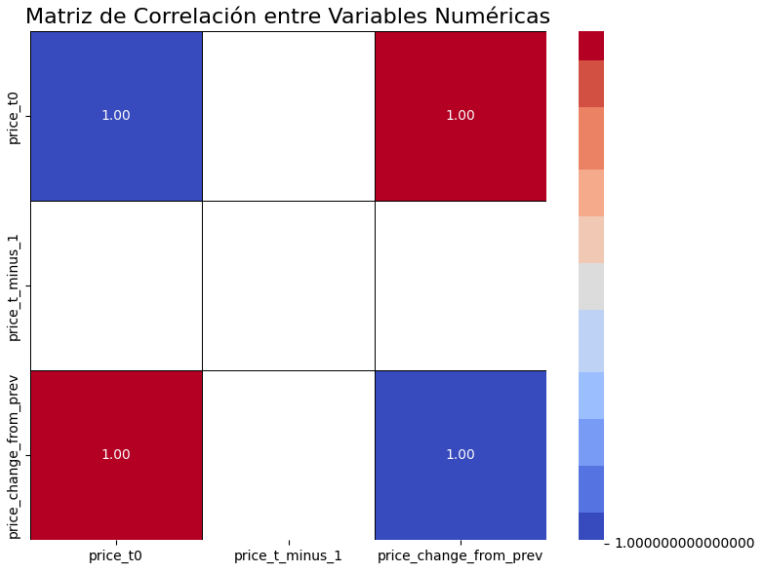




- Alta Cardinalidad: pokemon_name, artist_name, set_name tienen muchos valores únicos.
    - Implicación: Serán manejadas por OneHotEncoder, generando un espacio de alta dimensionalidad.



- Correlaciones: Alta correlación entre price_t0 y price_t_minus_1. price_change_from_prev aporta información única sobre la dinámica del mercado.



-Fase 3: Preprocesamiento de Features y Modelado

Esta fase comprende la preparación final de los datos y el entrenamiento de los algoritmos.

3.1. Preprocesamiento Aplicado

Codificación One-Hot: Se aplica OneHotEncoder a las características categóricas, transformándolas en un formato binario. El argumento handle_unknown='ignore' es clave para manejar nuevas categorías en producción.

Escalado de Variables: Se aplica StandardScaler a las características numéricas. Esto las normaliza (media 0, desviación estándar 1), asegurando que ninguna característica domine por su magnitud, crucial para modelos como Regresión Logística.

3.2. División del Dataset

Los datos preprocesados se dividieron en conjuntos de entrenamiento (80%) y prueba (20%) utilizando train_test_split.

La opción stratify=y_train_target_series fue fundamental para mantener las proporciones exactas de las clases (incluso las minoritarias) en ambos conjuntos. Esto garantiza que las evaluaciones posteriores sean fiables y no estén sesgadas por una distribución desigual en el test set.

```
print("Separando el dataset en conjuntos de entrenamiento y prueba para una evaluación imparcial del modelo.")

X_train_clf, X_test_clf, y_train_clf, y_test_clf = train_test_split(
    X_classifier_final, y_train_target_series, test_size=0.2, random_state=42, stratify=y_train_target_series
)
print(f"\nDatos divididos: Entrenamiento ({len(X_train_clf)} filas) y Prueba ({len(X_test_clf)} filas).")
print(f"  > Distribución de clases en el conjunto de ENTRENAMIENTO:\n{y_train_clf.value_counts(normalize=True).mul(100).round(2).astype(str) + '%'}")
print(f"  > Distribución de clases en el conjunto de PRUEBA:\n{y_test_clf.value_counts(normalize=True).mul(100).round(2).astype(str) + '%'}")
```

3.3. Selección y Comparación de Modelos

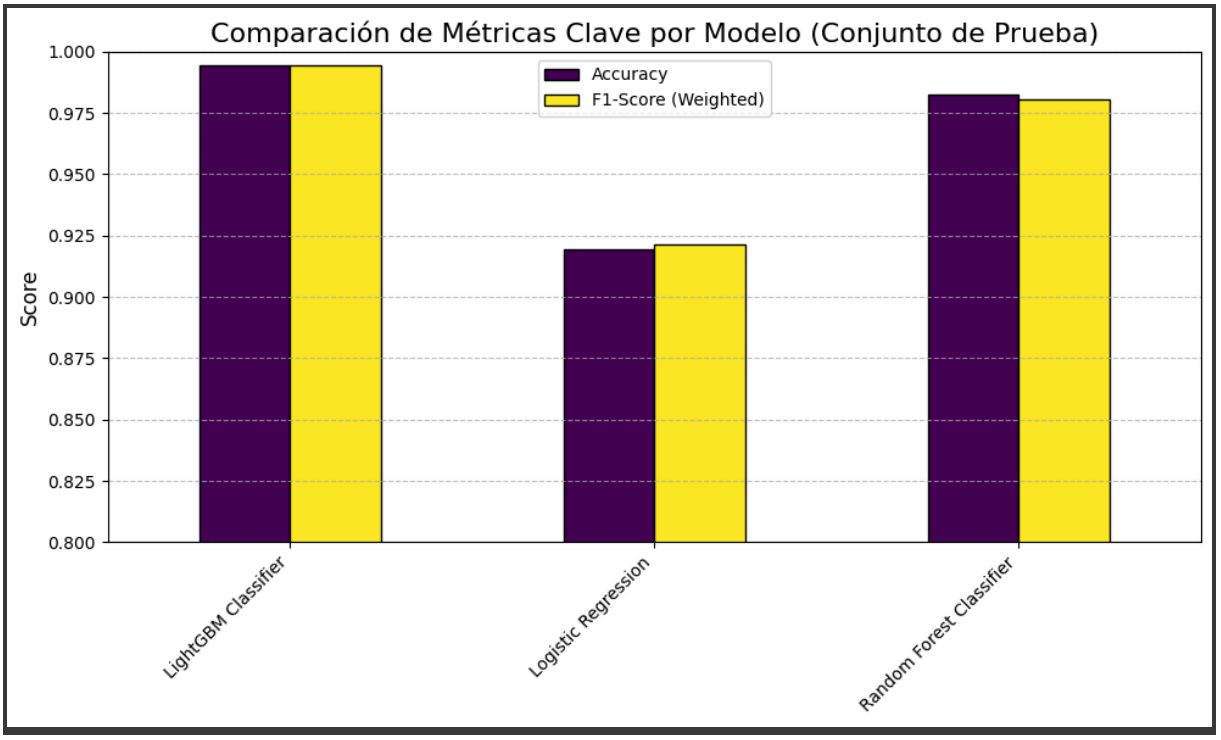
Para justificar la elección del modelo, se entrenaron y evaluaron tres clasificadores comunes:

- **LightGBM Classifier**: Eficiente, rápido y robusto para datos tabulares y alta dimensionalidad.
- **Logistic Regression** (envuelto en OneVsRestClassifier): Modelo lineal como línea base, crucial para validar el beneficio de modelos más complejos. Utiliza class_weight='balanced'.
- **Random Forest Classifier**: Un modelo de ensamble robusto, menos propenso a sobreajuste, también con class_weight='balanced'.

Los modelos fueron comparados en el conjunto de prueba usando Accuracy y F1-Score (Weighted), siendo este último más apropiado por el desbalance de clases.

Resumen de Rendimiento de Modelos en Test Set

	Accuracy	F1-Score (Weighted)
LightGBM Classifier	0.994584	0.994591
Random Forest Classifier	0.982399	0.980619
Logistic Regression	0.919307	0.921498



- **Conclusión del Modelado**: El **LightGBM Classifier** se seleccionó como el modelo principal debido a su superior rendimiento en el F1-Score (Weighted) y Accuracy, demostrando una excelente capacidad de generalización y manejo del desbalance de clases para nuestro problema.

-Fase 4: Evaluación y Fiabilidad del Modelo Final

Se realizó una evaluación exhaustiva del modelo LightGBM Classifier para validar su robustez.

4.1. Validación Cruzada (K-Fold Estratificado)

Se aplicó una validación cruzada con StratifiedKFold (5 folds) sobre el dataset completo preprocesado.

- **Razón de ser**: Esta técnica proporciona una estimación del rendimiento mucho más robusta y fiable que una simple división train_test_split. Al entrenar y validar el modelo múltiples veces con diferentes subconjuntos de datos, se asegura que los resultados no dependan de una división aleatoria y que el modelo sea generalizable. El stratify en los folds es, de nuevo, crucial por el desbalance.
  Resultados de la Validación Cruzada del LightGBM:

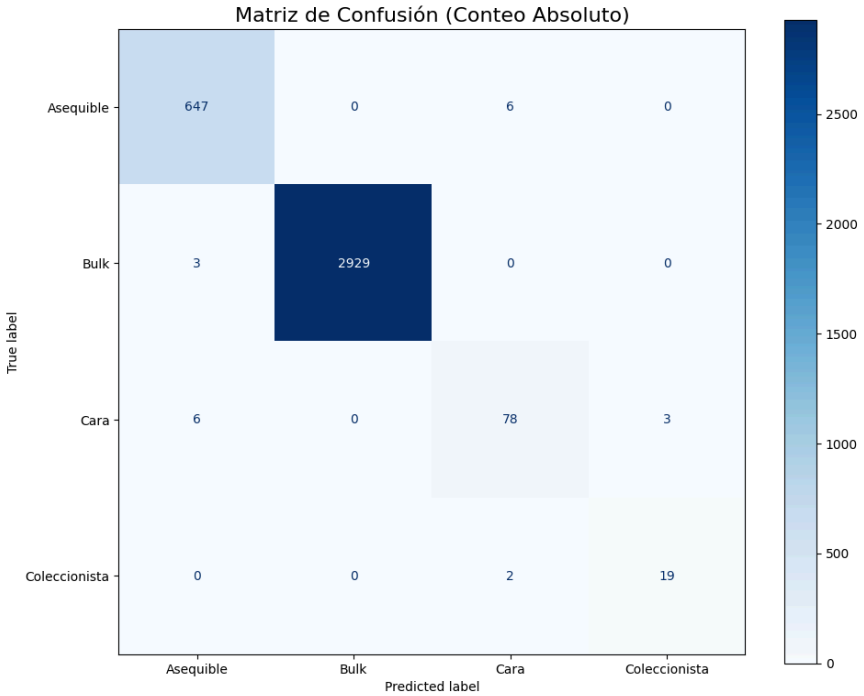
```
Resultados de Validación Cruzada (5-Fold Estratificado) para el LGBMClassifier:
- Accuracy: 0.9959 (Desviación Estándar: 0.0010)
- Precision weighted: 0.9960 (Desviación Estándar: 0.0010)
- Recall weighted: 0.9959 (Desviación Estándar: 0.0010)
- F1 weighted: 0.9959 (Desviación Estándar: 0.0010)
```

Conclusión de CV: La baja desviación estándar (+/- XXXX) en todas las métricas valida la consistencia y estabilidad del modelo, confirmando su fiabilidad y minimizando el riesgo de sobreajuste.

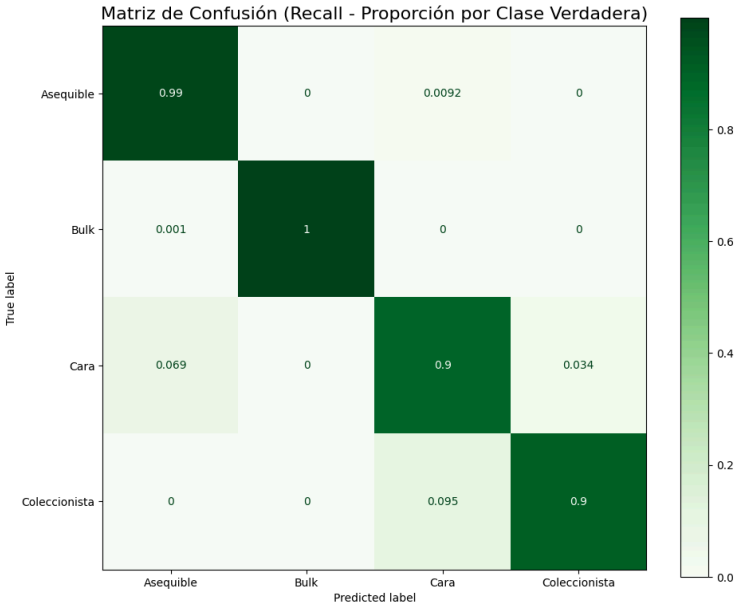
4.2. Matriz de Confusión y Reporte de Clasificación (Conjunto de Prueba)

Para una evaluación granular, se generaron las métricas y visualizaciones en el conjunto de prueba:

- **Matriz de Confusión:**



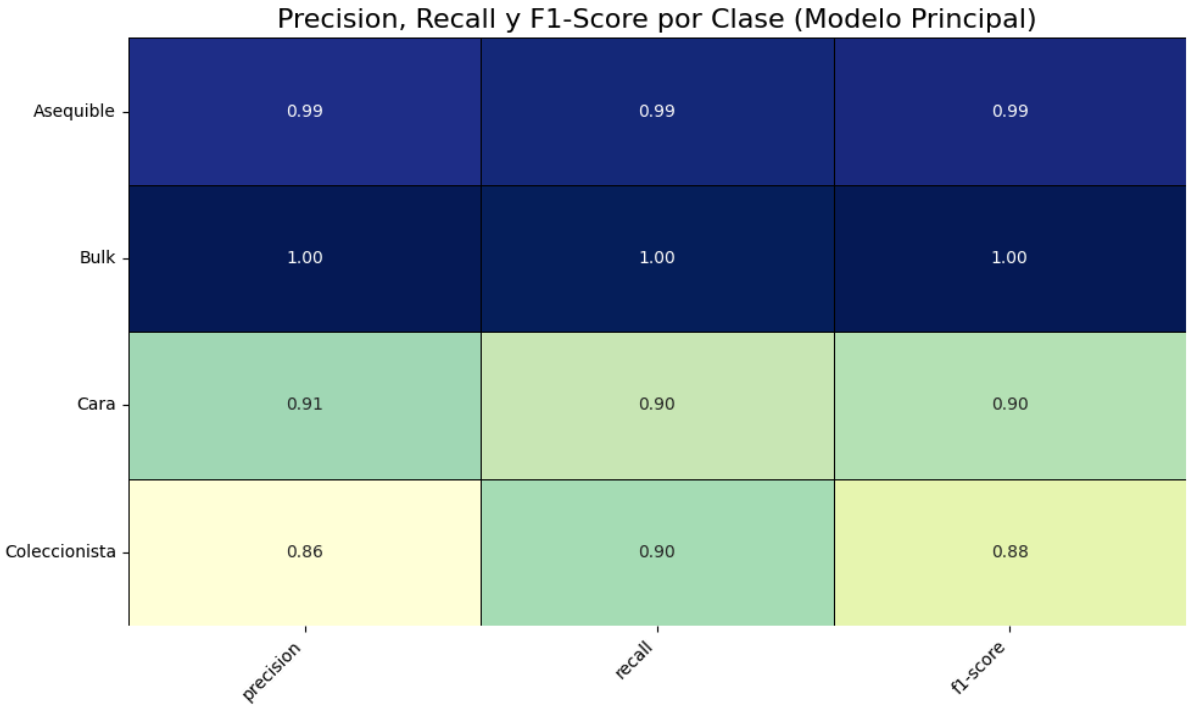
- **Matriz de Confusión (Recall):**



- **Observación:** La matriz muestra un alto acierto general. Para clases minoritarias como 'Cara' y 'Coleccionista', el modelo sigue logrando buenos recall (proporción de casos reales bien identificados), demostrando que no se limita a predecir sólo la clase mayoritaria ('Bulk').

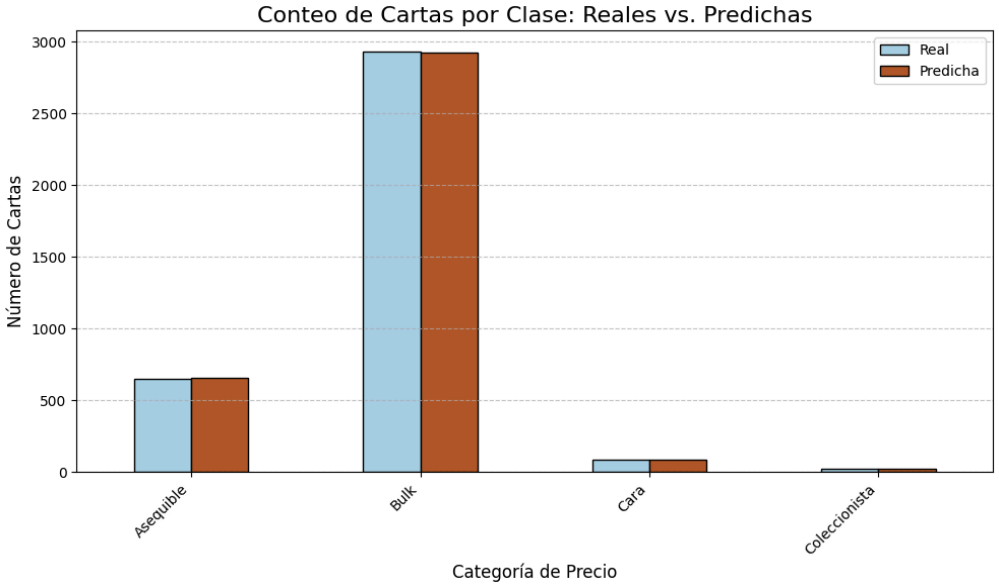
- **Reporte de Clasificación:** Muestra Precision, Recall y F1-score detallado por clase.

```
3. Mostrando el Reporte de Clasificación Detallado del modelo principal:
              precision    recall  f1-score      support
Asequible      0.986280  0.990812  0.988541   653.000000
Bulk           1.000000  0.998977  0.999488  2932.000000
Cara           0.906977  0.896552  0.901734    87.000000
Coleccionista  0.863636  0.904762  0.883721    21.000000
accuracy       0.994584  0.994584  0.994584     0.994584
macro avg      0.939223  0.947776  0.943371  3693.000000
weighted avg   0.994607  0.994584  0.994591  3693.000000
```



- **Observación:** Aunque las clases minoritarias son más desafiantes, sus métricas F1-score son notables para su representación. El modelo es eficaz en todo el espectro de categorías.

- **Conteo de Predicciones Reales vs. Predichas:**



- **Observación:** La distribución de las clases predichas coincide estrechamente con la real en el test set, lo que valida la capacidad del modelo para clasificar fielmente la proporción de cada tipo de carta.

4.3. Conclusión de la Evaluación del Modelo Actual

El LightGBM Classifier presenta un rendimiento excelente y muy fiable para la clasificación de precios. Maneja eficazmente el desbalance de clases y es robusto, sentando una base sólida para su integración en el proyecto principal.