Predicción Avanzada de LTV, CAC y ROMI para Showz

Resumen Ejecutivo

Problema: Showz busca optimizar su inversión en marketing digital comprendiendo mejor el valor a largo plazo de sus clientes (LTV) y el costo asociado a su adquisición (CAC) por diferentes canales. La asignación actual del presupuesto podría no ser óptima para maximizar el retorno de la inversión (ROMI).

Solución: Se desarrolló un sistema de predicción basado en Machine Learning utilizando datos históricos de visitas, órdenes y costos de marketing (junio 2017 - junio 2018). Este sistema predice dos métricas clave para cada usuario:

1. **LTV_180:** Ingreso total esperado en los 180 días posteriores a su primera visita.
2. **CAC_source_30:** Costo de adquisición promedio estimado para la fuente que lo atrajo, medido en los 30 días posteriores a su conversión.

Metodología:

- Se realizó una limpieza exhaustiva de los datos brutos y se aplicó ingeniería de características avanzada para crear variables descriptivas del comportamiento del usuario, patrones temporales e interacciones de marketing.
- Se entrenaron y validaron múltiples modelos de regresión (desde baselines lineales hasta ensambladores avanzados como Gradient Boosting) utilizando un enfoque de validación temporal estricto para simular un entorno de producción realista.
- Se seleccionaron los modelos con mejor rendimiento (Gradient Boosting optimizado con validación cruzada temporal) para LTV y CAC, evaluados mediante MAE y MAPE.
- Se utilizaron técnicas de explicabilidad (Importancia de Variables por Gain/Permutation, SHAP) para identificar los factores clave que influyen en LTV y CAC.

Hallazgos Clave:

- Los modelos de **Gradient Boosting optimizados** demostraron ser los más precisos en la validación temporal.
- Para **LTV**, el **ingreso total previo (`revenue_total`)** es el predictor dominante, seguido por el **valor promedio de orden (`avg_order_value`)** y la **frecuencia de compra (`avg_days_between_orders`)**. El análisis SHAP mostró una relación casi exponencial entre `revenue_total` y el LTV predicho, potenciada por un `avg_order_value` alto.
- Para **CAC**, el análisis de permutación destacó la importancia de la variable `is_weekend_session_False` (indica sesiones entre semana) y `is_churned_True` (usuarios inactivos). El análisis SHAP confirmó que el **costo promedio histórico por usuario (`avg_cost_per_user`)** y el **mes de la sesión (`session_month`)** también son drivers importantes, con interacciones complejas.
- El análisis de error segmentado por `device` mostró errores promedio significativos y relativamente similares para ambos dispositivos en CAC, sugiriendo que el tipo de dispositivo no es el principal factor de incertidumbre en estas predicciones.
- Se identificaron fuentes de marketing (específicamente **source_id 9 y 10**) con un **ROMI predicho significativamente positivo** (~3.7 y ~8.0 respectivamente), indicando un alto potencial de rentabilidad, mientras otras fuentes mostraron ROMI predicho negativo.

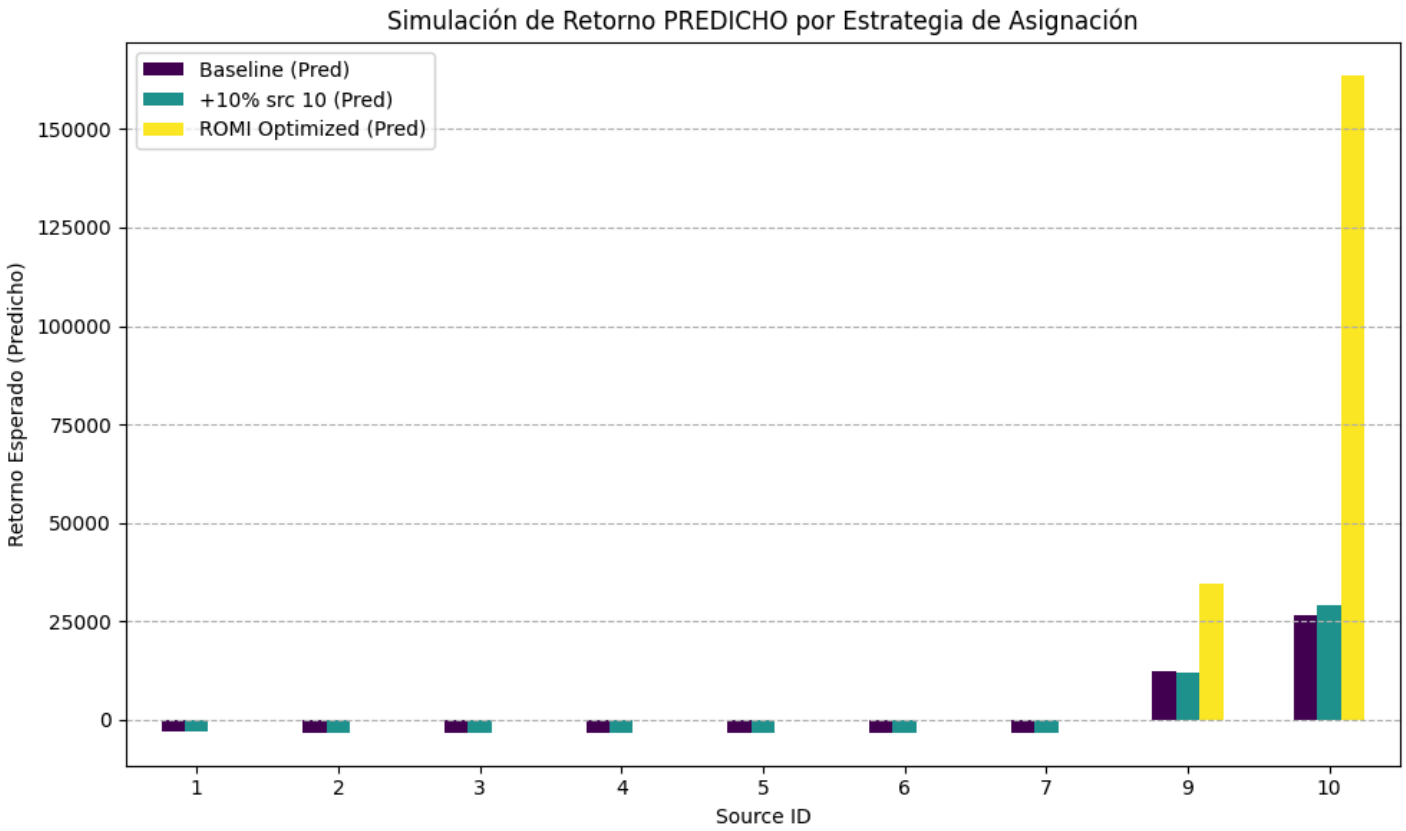
Recomendaciones Estratégicas:

- Se simuló el impacto de diferentes estrategias de asignación de presupuesto utilizando las predicciones de LTV y CAC.
- La estrategia **"ROMI Optimized"**, que reasigna el presupuesto total ($30,000) proporcionalmente a las fuentes con ROMI predicho positivo (fuentes 9 y 10), **generó un retorno total estimado de $198,392**, superando drásticamente el escenario baseline (asignación igualitaria, retorno ~$15,623) y el escenario de aumento puntual (+10% a fuente 10, retorno ~$18,078).
- **Se recomienda adoptar la estrategia "ROMI Optimized"**, concentrando la inversión en las fuentes 9 (aprox. 31.5% del presupuesto) y 10 (aprox. 68.5%), lo que representa una **ganancia potencial estimada de $182,769** frente a la asignación actual.

Próximos Pasos:

- Implementar la asignación de presupuesto recomendada para el próximo ciclo.

- Monitorear continuamente el rendimiento real de LTV, CAC y ROMI por fuente para validar las predicciones.
- Re-entrenar periódicamente los modelos con datos nuevos para mantener su precisión y explorar mejoras en el feature engineering para refinar las predicciones.




```
Asignación de presupuesto recomendada (ROMI Optimized):
   source_id    budget  budget_percentage
0          1      0.00                0.0
1          2      0.00                0.0
2          3      0.00                0.0
3          4      0.00                0.0
4          5      0.00                0.0
5          6      0.00                0.0
6          7      0.00                0.0
7          9   9451.28               31.5
8         10  20548.72               68.5
```

Detalles Técnicos (Enfoque CRISP-DM)

1. Comprensión del Negocio (Business Understanding)

- **Objetivos del Negocio:** Optimizar la asignación del presupuesto de marketing digital de Showz para maximizar el Retorno de la Inversión (ROI), identificar los canales de adquisición más rentables a largo plazo, y comprender los factores clave que determinan el valor del cliente (LTV) y su costo de adquisición (CAC).
- **Objetivos de Minería de Datos:**
  - Construir modelos de regresión supervisada para predecir el **LTV_180** (ingreso acumulado de un usuario en los 180 días posteriores a su primera sesión).
  - Construir modelos de regresión supervisada para predecir el **CAC_source_30** (costo promedio de adquisición asociado a la fuente del usuario, medido en los 30 días posteriores a su primera conversión).
  - Identificar las características (features) más influyentes en las predicciones de LTV y CAC mediante técnicas de explicabilidad.
  - Simular diferentes escenarios de asignación de presupuesto de marketing basados en el ROMI predicho ((LTV_pred - CAC_pred) / CAC_pred) para recomendar una estrategia óptima.
- **Criterios de Éxito:**
  - Modelos predictivos con un rendimiento (medido por MAE y MAPE) significativamente superior a modelos baseline simples en un conjunto de validación temporalmente posterior a los datos de entrenamiento.
  - Identificación clara y interpretable de los principales impulsores (drivers) de LTV y CAC.

- Una recomendación de estrategia de asignación de presupuesto basada en simulaciones que demuestre un potencial de mejora cuantificable en el ROMI general en comparación con la estrategia actual (baseline).
- Código reproducible y bien documentado.

2. Comprensión de los Datos (Data Understanding)

- **Fuentes de Datos:** Se utilizaron tres archivos CSV proporcionados:
  - `visits_log_us.csv`: Contiene información sobre las sesiones de los usuarios, incluyendo ID de usuario (`Uid`), dispositivo (`Device`), fuente (`Source Id`), y timestamps de inicio (`Start Ts`) y fin (`End Ts`). (~360k filas).
  - `orders_log_us.csv`: Registra las órdenes realizadas, incluyendo ID de usuario (`Uid`), timestamp de compra (`Buy Ts`), y el ingreso generado (`Revenue`). (~50k filas).
  - `costs_us.csv`: Detalla los costos de marketing diarios por fuente (`source_id`), fecha (`dt`), y monto (`costs`). (~2.5k filas).
- **Exploración Inicial (EDA - `01_EDA.ipynb`):**
  - Se cargaron los datos crudos en DataFrames de Pandas.
  - Se realizó una inspección inicial utilizando `head()`, `info()`, `isnull().sum()`, y `duplicated().sum()`.
  - Los datos brutos mostraron estar completos en términos de valores nulos en columnas clave y no presentaron duplicados evidentes.
  - Las columnas de fecha/hora estaban en formato `object` y requerían conversión.
  - Los nombres de las columnas no seguían un estándar uniforme (e.g., `Start Ts`, `Source Id`).
  - Se confirmó la necesidad de unir estas tablas usando `Uid` y `Source Id` / `source_id` como claves.

3. Preparación de los Datos (Data Preparation)

- **Limpieza (`01_EDA.ipynb` / `prepare_data.py`):**
  - Se renombraron las columnas a formato *snake_case* (e.g., `start_ts`, `source_id`).
  - Se convirtieron las columnas de timestamp (`start_ts`, `end_ts`, `buy_ts`, `dt`) a objetos `datetime` de Pandas.
  - Se aseguraron los tipos de datos correctos para identificadores (`Uid` como `uint64`, `source_id` como `int64`) y valores numéricos (`Revenue`, `costs` como `float64`).
  - Se eliminaron duplicados (aunque no se encontraron en la carga inicial).
  - Se guardaron los dataframes limpios como `visits_clean.csv`, `orders_clean.csv`, `costs_clean.csv` en la carpeta `data/processed/`.
- **Ingeniería de Características (`02_FeatureEngineering.ipynb` y `src/features.py`):**
  - Se cargaron los dataframes limpios.
  - Se crearon múltiples características a nivel de usuario (`uid`) mediante agregación y cálculos entre tablas:
    - **Características de Comportamiento:** Se calcularon métricas como número total de sesiones (`n_sessions`), duración promedio y desviación estándar de sesiones (`avg_session_duration`, `session_duration_std`), número total de órdenes (`n_orders`), ingreso total (`revenue_total`), valor promedio por orden (`avg_order_value`), tiempo desde la primera sesión hasta la primera compra (`conversion_delay_days`), tiempo entre la primera y última compra (`order_span_days`), órdenes por sesión (`orders_per_session`), tiempo promedio entre órdenes (`avg_days_between_orders`), días desde la última sesión (`days_since_last_session`), y una bandera de abandono (`is_churned`) si `days_since_last_session > 30`.
    - **Características Temporales:** Se extrajeron componentes de la fecha/hora de la *primera sesión* (`first_session`): mes (`session_month`), día (`session_day`), trimestre (`session_quarter`), día de la semana (`session_weekday`), hora (`session_hour`), y si fue fin de semana (`is_weekend_session`). También se extrajo el día de la semana de la *primera conversión* (`conversion_weekday`).
    - **Características de Marketing:** Se incluyó el dispositivo (`device`) y la fuente (`source_id`) de la última sesión. Se calcularon métricas agregadas por fuente: tasa de conversión promedio de la fuente (`source_conversion_rate`) y costo promedio por usuario de la fuente (`avg_cost_per_user`).
  - **Cálculo de Variables Objetivo (Targets):**
    - `LTV_180`: Se determinó la fecha de la primera sesión de cada usuario (`first_session`) y se sumaron los ingresos (`Revenue` de `orders`) ocurridos dentro de los 180 días posteriores a esa fecha.

        - `CAC_source_30`: Se identificó la `source_id` asociada a la sesión que precedió inmediatamente a la primera conversión (`first_order`). Luego, se sumaron los costos de esa `source_id` durante los 30 días posteriores a la fecha de conversión y se calculó un costo promedio (aunque el notebook `features.py` parece calcular el costo total asociado en ese período, se interpreta como target para el modelo).
    - **Características de Cohorte:** Se agruparon usuarios por mes de adquisición (basado en `first_session`). Se calcularon métricas promedio para cada cohorte: LTV promedio (`ltv_cohort_avg`), CAC promedio (`cac_cohort_avg`), y tasa de conversión promedio (`conversion_rate_cohort`). Estas se asignaron de nuevo a cada usuario según su mes de cohorte.
    - **Dataset Final:** Todas las características calculadas y las variables objetivo se unieron en un único DataFrame a nivel de usuario (`uid`). Se eliminaron columnas intermedias (como fechas de sesiones/órdenes individuales) y se guardó como `final_dataset.csv` en `data/processed/`. Este dataset contenía aproximadamente 228k filas y 33 columnas.

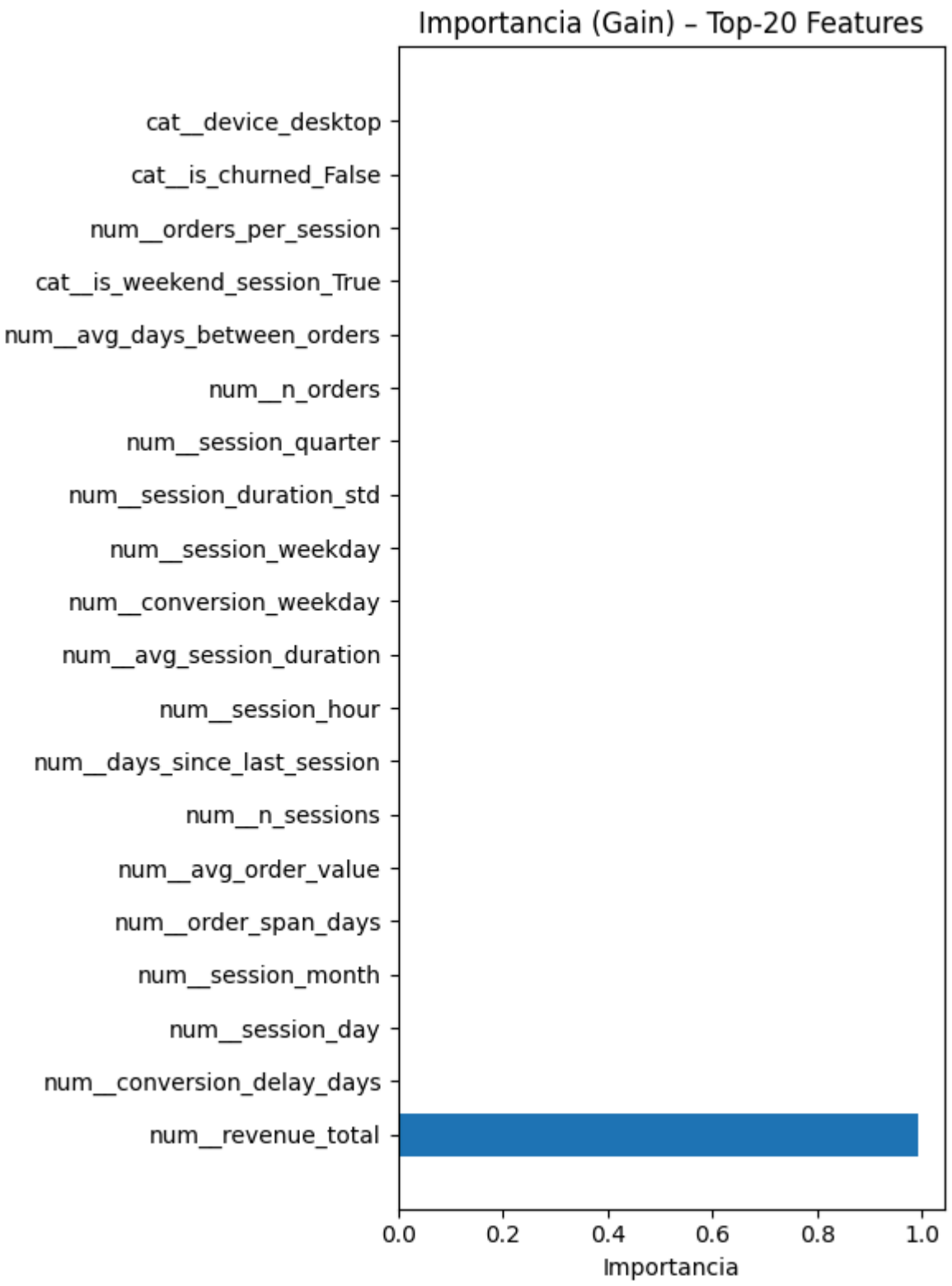
4. Modelado (Modeling)

- **Enfoque:** Se definieron dos tareas de regresión separadas, una para predecir `LTV_180` y otra para `CAC_source_30`, utilizando el mismo conjunto de características preparadas.
- **División de Datos (`03_ModelTraining.ipynb` y `src/train.py`):** Se implementó una división temporal basada en la columna `first_session` para simular un escenario realista donde se entrena con datos pasados y se evalúa en datos futuros:
    - **Conjunto de Entrenamiento:** Usuarios cuya `first_session` fue antes del '2018-01-01'.
    - **Conjunto de Validación:** Usuarios cuya `first_session` fue entre '2018-01-01' y '2018-04-01' (usado para evaluación final en `Final_Project_Showz_LTV_CAC.ipynb` y como set de validación implícito en `GridSearchCV`).
    - **Conjunto de Prueba:** Usuarios cuya `first_session` fue el '2018-04-01' o después (no utilizado explícitamente para reportar métricas finales en los notebooks proporcionados, pero definido en la función `load_and_prepare_data`).
    - Se eliminaron filas con valores nulos en la variable objetivo (`LTV_180` o `CAC_source_30`) de cada conjunto respectivo.
- **Preprocesamiento (`src/train.py`, `03_ModelTraining.ipynb`):** Se definió un `Pipeline` de preprocesamiento utilizando `ColumnTransformer`:
    - **Variables Numéricas:** Se imputaron los valores faltantes con la mediana (`SimpleImputer(strategy='median')`) y luego se escalaron usando estandarización (`StandardScaler`).
    - **Variables Categóricas/Booleanas:** Se imputaron los valores faltantes con la moda (`SimpleImputer(strategy='most_frequent')`) y se aplicó codificación One-Hot (`OneHotEncoder(handle_unknown='ignore')`).
- **Modelos Entrenados (`03_ModelTraining.ipynb`):** Se entrenó una variedad de modelos de regresión, cada uno dentro de un `Pipeline` que incluía el paso de preprocesamiento:
    - **Baselines:** LinearRegression, Ridge, SGDRegressor.
    - **Modelos Avanzados:** RandomForestRegressor, XGBRegressor, LGBMRegressor, CatBoostRegressor.
    - **Ensamblador:** StackingRegressor (utilizando los modelos avanzados como estimadores base y Ridge como meta-estimador).
- **Entrenamiento (`03_ModelTraining.ipynb`):** Los pipelines de cada modelo se ajustaron (`fit`) utilizando los datos del conjunto de entrenamiento (X_train, y_train de 2017). Los modelos entrenados (pipelines completos) se serializaron y guardaron como archivos `.pkl` en la carpeta `/models/` (e.g., `LTV_180_lgb.pkl`, `CAC_source_30_rf.pkl`).

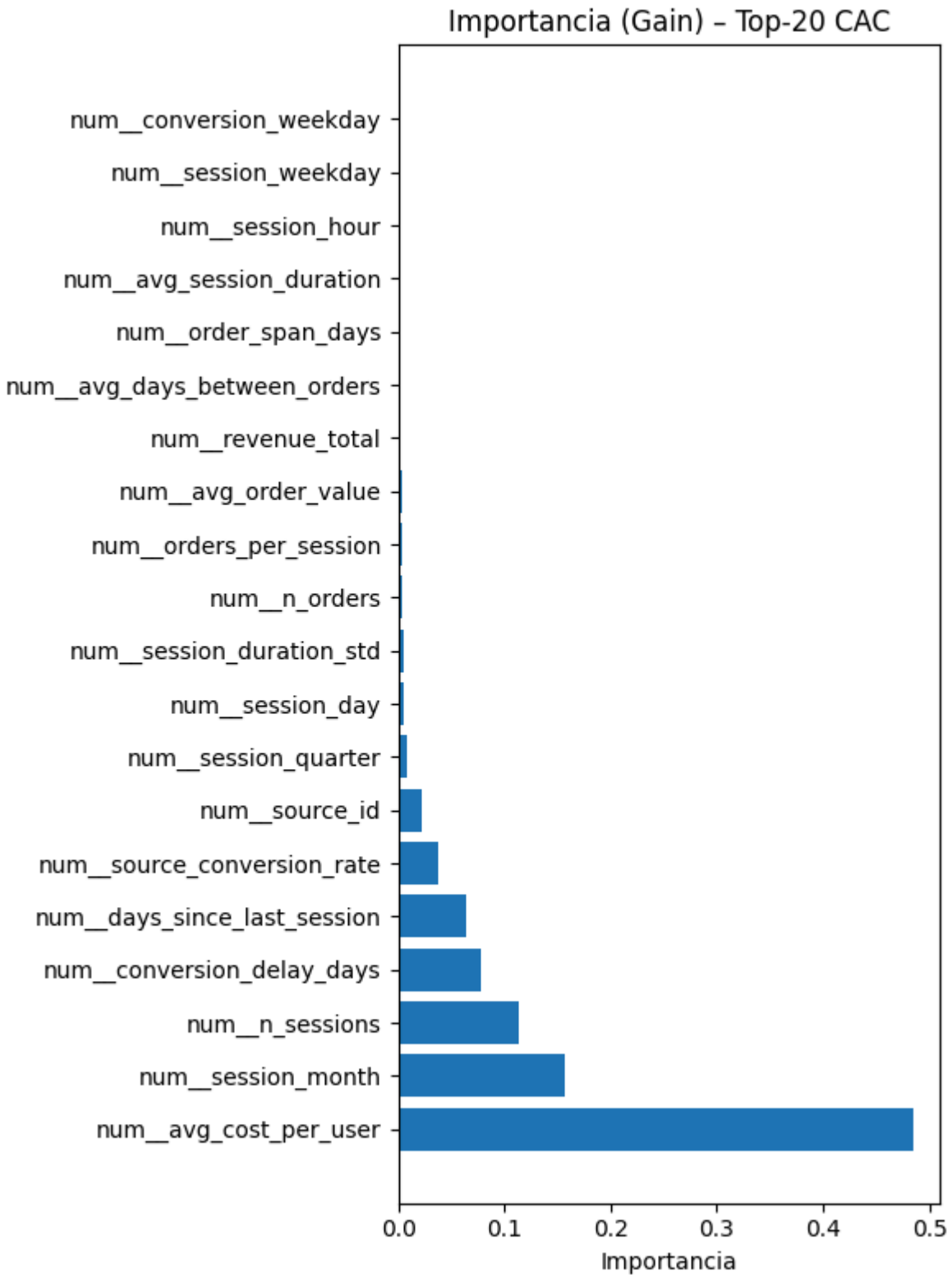
5. Evaluación (Evaluation)

- **Métricas (`evaluation.py`):** Se definieron funciones para calcular el Error Absoluto Medio (MAE), la Raíz del Error Cuadrático Medio (RMSE) y el Error Porcentual Absoluto Medio (MAPE). Se dio prioridad a MAE y MAPE para la interpretación del negocio debido a su escala (error absoluto y porcentual, respectivamente).
- **Validación Inicial (`Final_Project_Showz_LTV_CAC.ipynb`):** Se cargaron todos los modelos entrenados desde `/models/` y se evaluaron en el conjunto de validación (H1 2018). Se compararon sus métricas (MAE, RMSE, MAPE). Los modelos basados en árboles (RF, XGB, LGB, CatBoost) y el Stacking mostraron mejor rendimiento que los lineales.
- **Validación Cruzada y Ajuste de Hiperparámetros (`Final_Project_Showz_LTV_CAC.ipynb`):**
    - Se implementó `GridSearchCV` específicamente para `GradientBoostingRegressor` (no entrenado previamente en `03_ModelTraining`).
    - Se utilizó `TimeSeriesSplit(n_splits=5)` como estrategia de validación cruzada para respetar la naturaleza temporal de los datos dentro del conjunto de entrenamiento (datos de 2017).

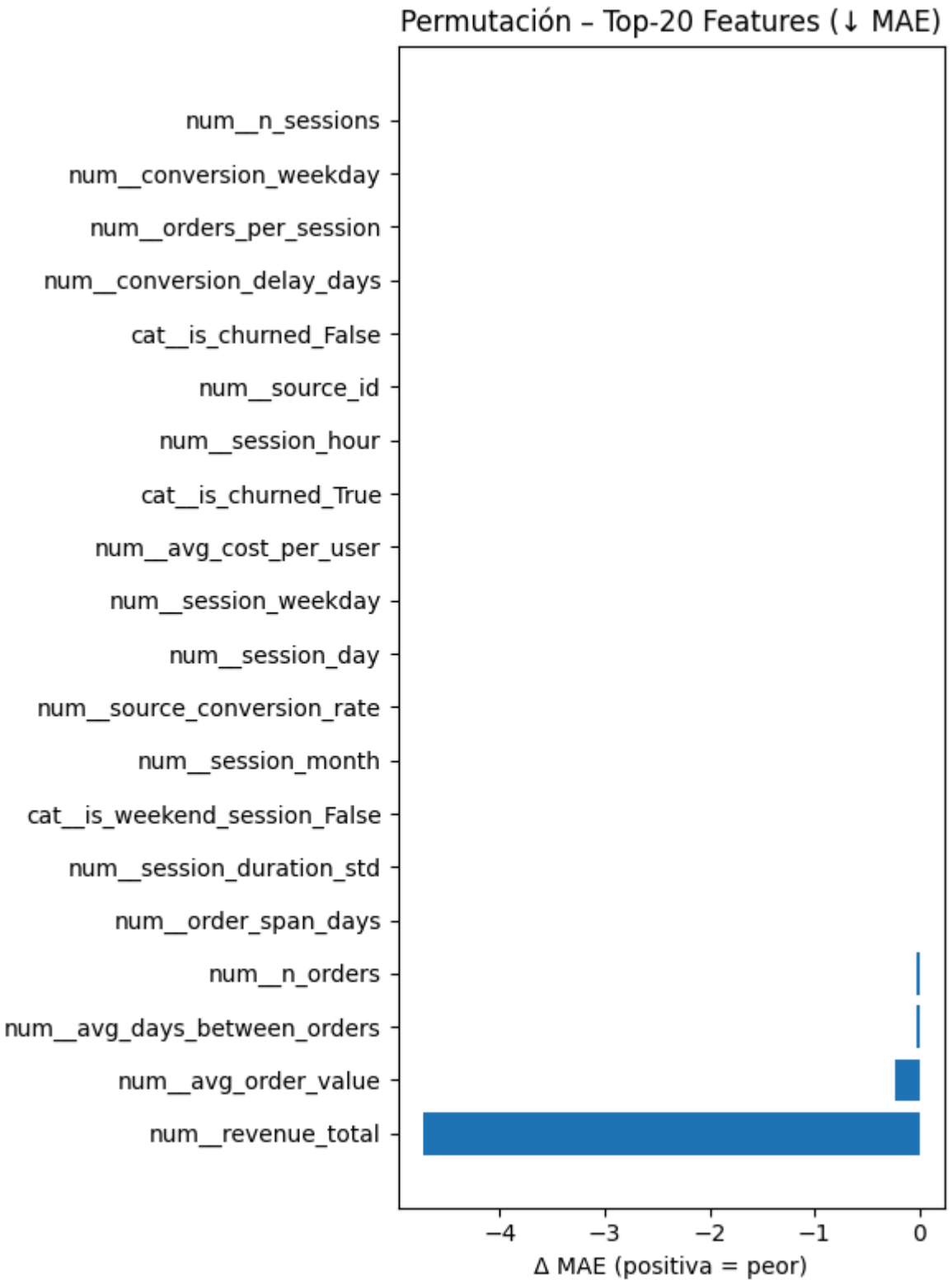
    - Se optimizaron `n_estimators`, `learning_rate`, y `max_depth`.
    - La métrica de puntuación utilizada fue el MAE negativo (`make_scorer(mean_absolute_error, greater_is_better=False)`).
- **Modelo Seleccionado (`Final_Project_Showz_LTV_CAC.ipynb`):** El `GradientBoostingRegressor` con los mejores hiperparámetros encontrados mediante `GridSearchCV` con `TimeSeriesSplit` (denominado `best_ltv_model` y `best_cac_model` en el notebook) fue seleccionado como el modelo final para las fases de explicabilidad y simulación, debido a su robusta validación temporal.
    - Rendimiento (Validación H1 2018):
        - LTV (best_tscv): MAE ≈ 0.156, RMSE ≈ 2.95, MAPE ≈ 4.98%
        - CAC (best_tscv): MAE ≈ 2837, RMSE ≈ 4047, MAPE ≈ 67.2%
- **Explicabilidad (`Final_Project_Showz_LTV_CAC.ipynb`):** Se aplicaron técnicas sobre los modelos seleccionados (`best_ltv_model`, `best_cac_model`) utilizando el conjunto de validación (H1 2018) o una muestra de él.
    - **Importancia de Variables (Gain):**
        - *LTV:* El análisis de 'gain' mostró que `num_revenue_total` tiene una importancia dominante (casi 1.0), sugiriendo una fuerte dependencia del modelo en esta única variable. Las demás características tuvieron contribuciones marginales en comparación.



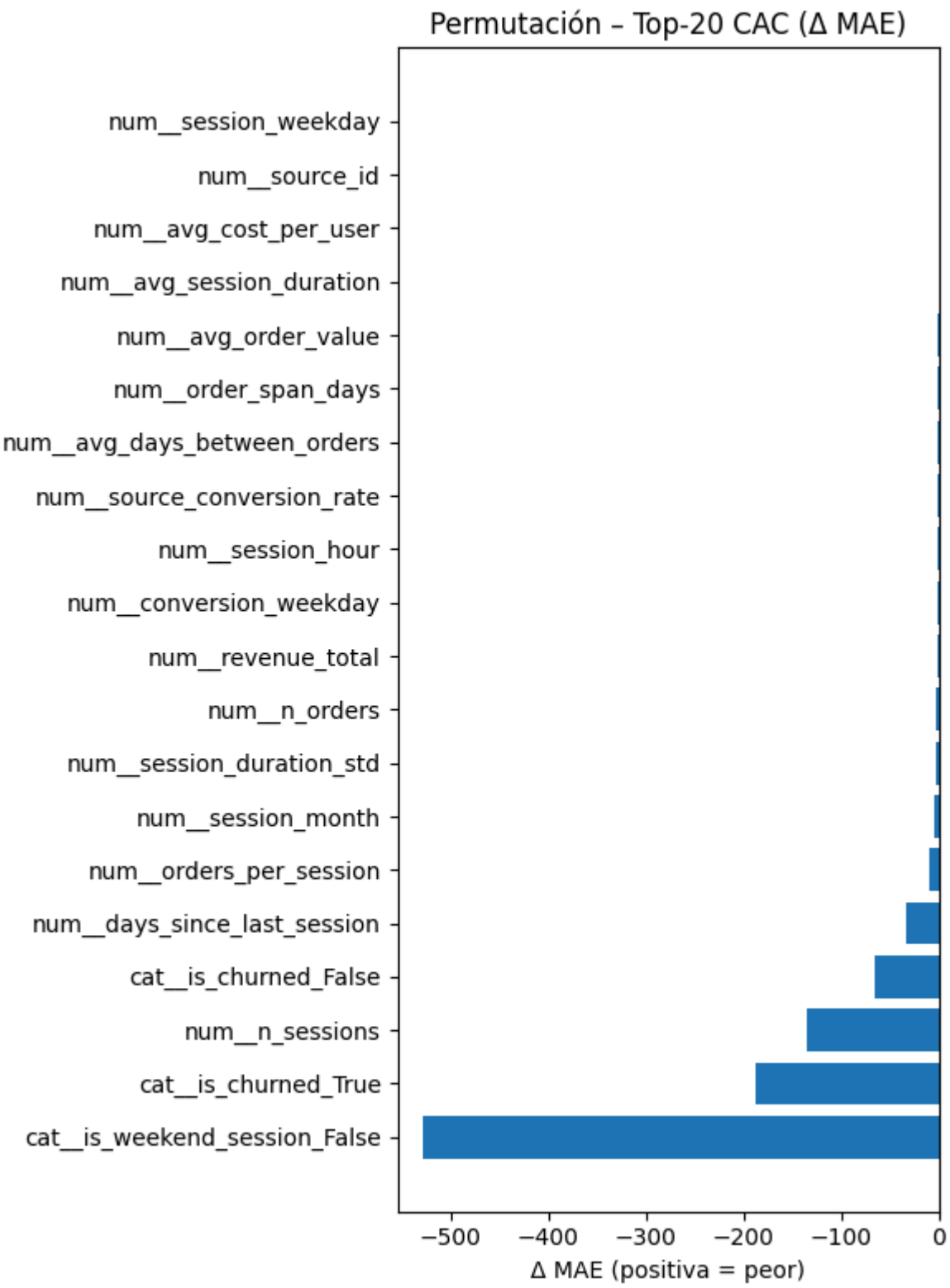
        - *CAC:* `num__avg_cost_per_user` fue la característica más importante (importancia ~0.5), seguida por `num__session_month` y `num__n_sessions`, indicando que el costo histórico y la temporalidad/frecuencia de sesiones son relevantes.



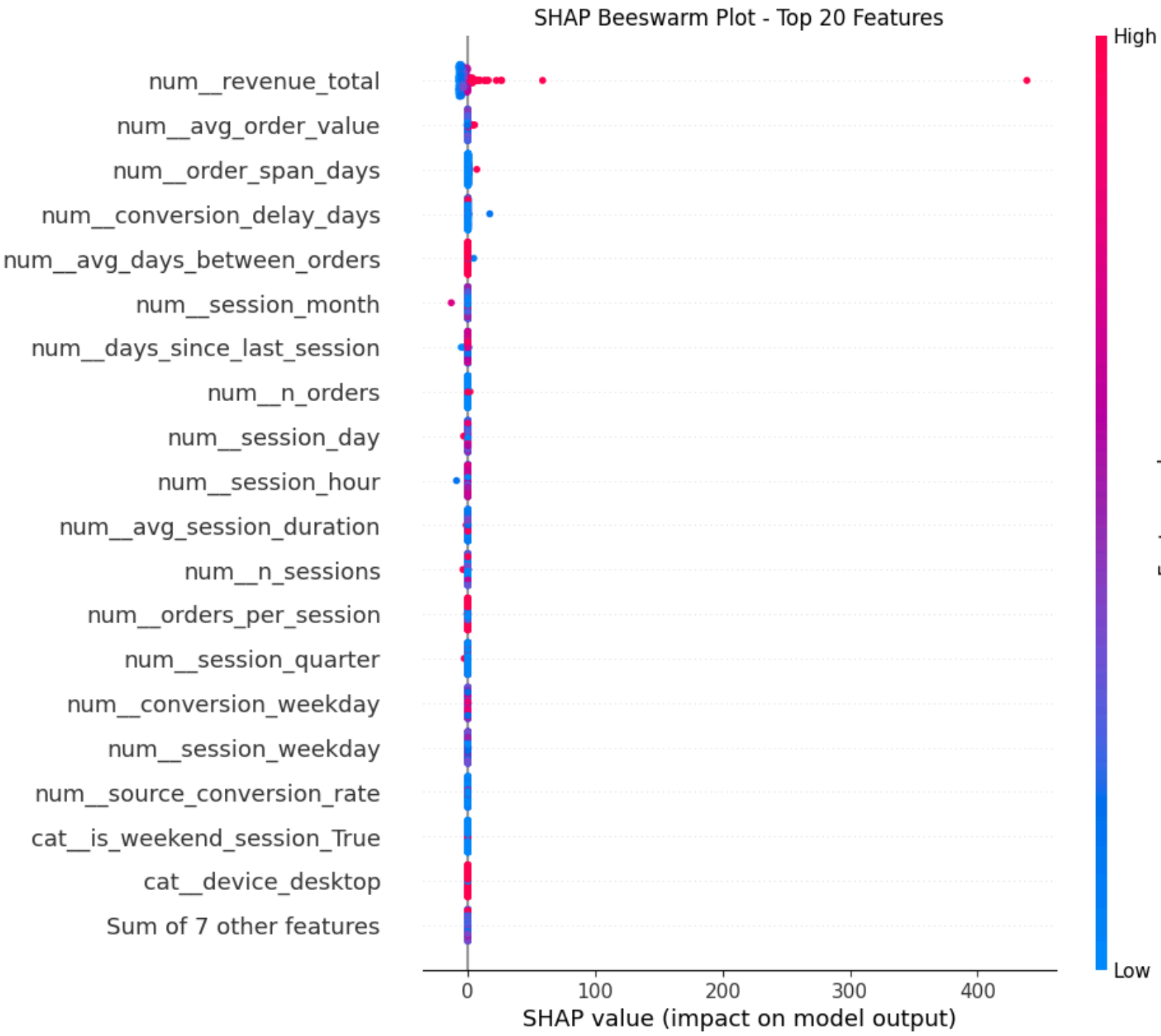
- **Importancia de Variables (Permutation):**
  - *LTV:* Este método confirmó la criticidad de `num_revenue_total` (mayor impacto negativo en MAE al permutar), pero también resaltó la importancia de `num_avg_order_value` y `num_avg_days_between_orders`, proporcionando una visión más equilibrada del impacto en el rendimiento predictivo.



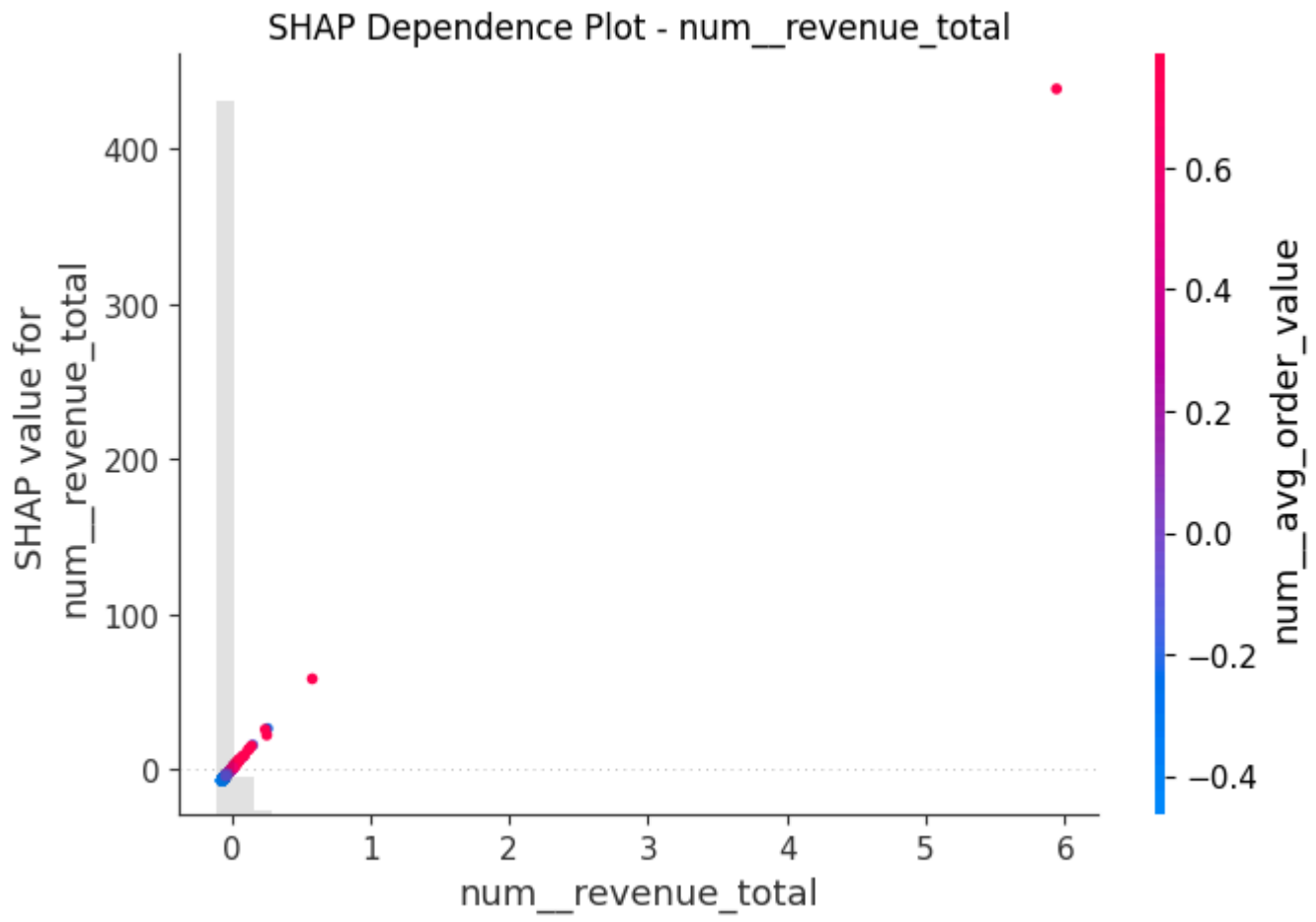
- *CAC:* Sorprendentemente, `cat__is_weekend_session_False` (sesiones entre semana) y `cat__is_churned_True` mostraron el mayor impacto en el MAE al ser permutadas, seguidas por `num__n_sessions`. Esto sugiere que estas características categóricas y de comportamiento son cruciales para la precisión del modelo CAC, más allá de lo indicado por el 'gain'.



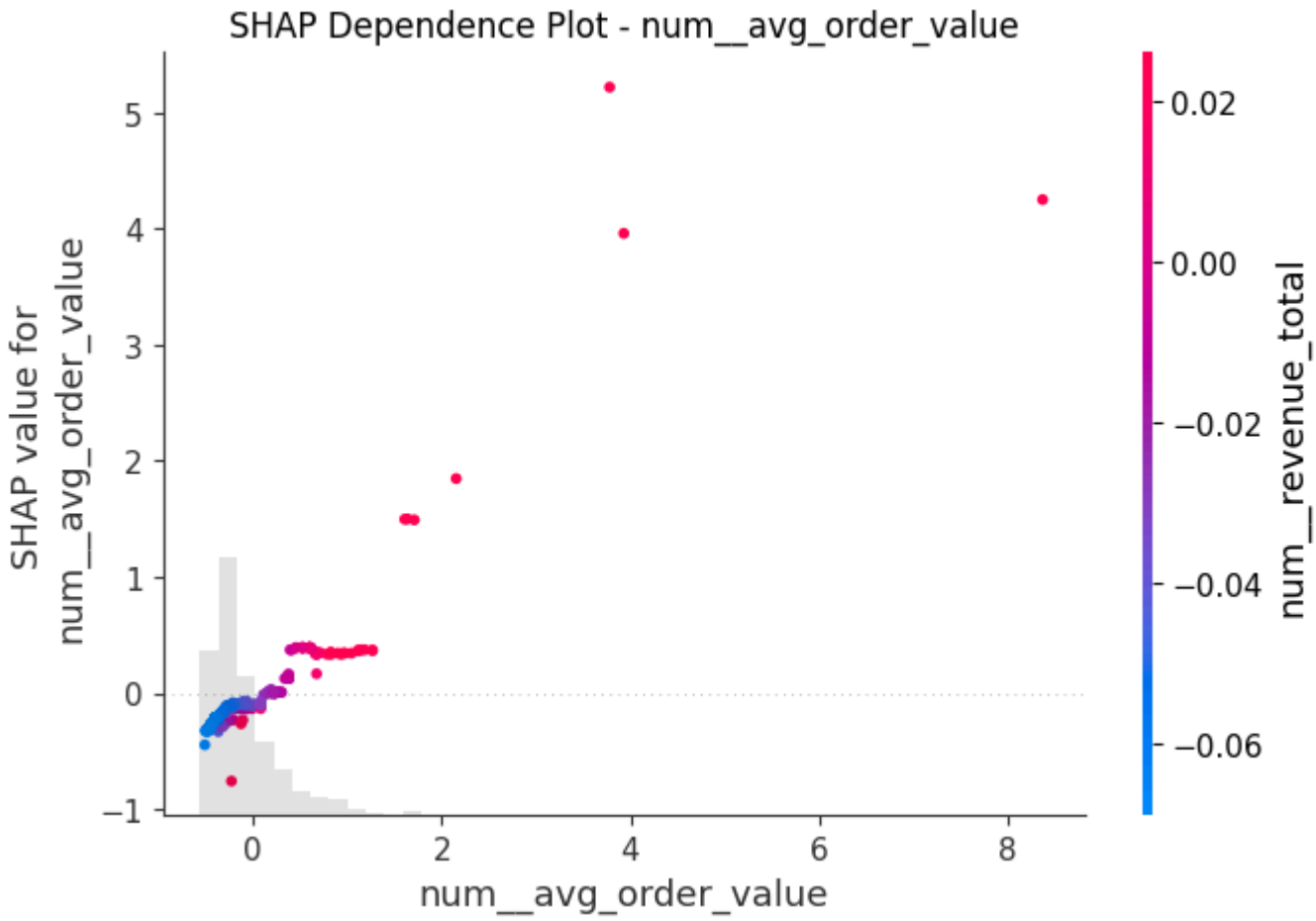
- **Valores SHAP:** Se generaron valores SHAP para una muestra de 500 observaciones del conjunto de validación usando `shap.TreeExplainer`.
  - *LTV:*
    - El gráfico *beeswarm* confirmó visualmente el dominio de `num_revenue_total`, `num_avg_order_value`, etc., en la explicación de las predicciones.



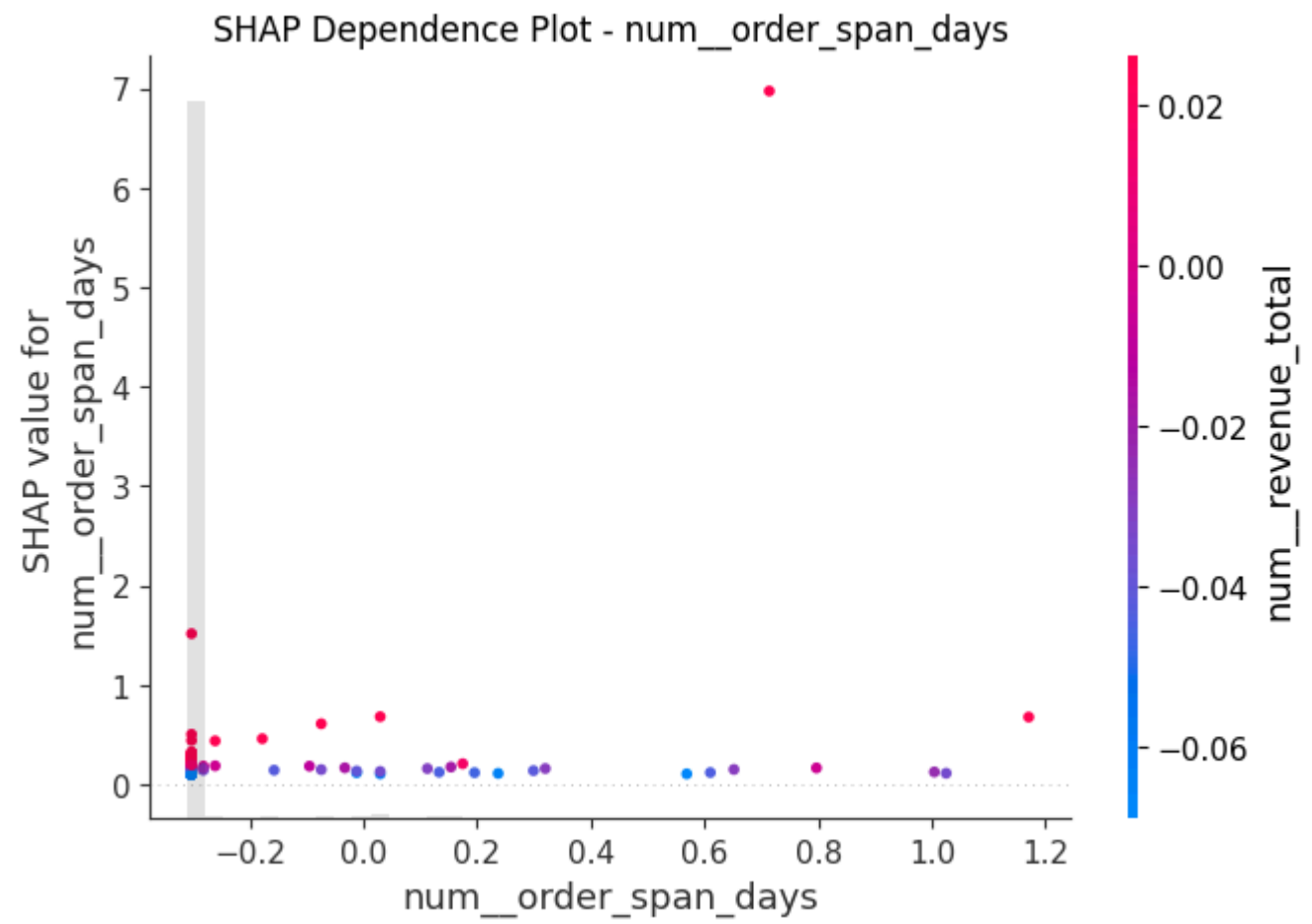
- Los gráficos de *dependencia* mostraron:
  - `num_revenue_total`: Relación positiva fuerte y exponencial con LTV_pred, interacción con `avg_order_value`.



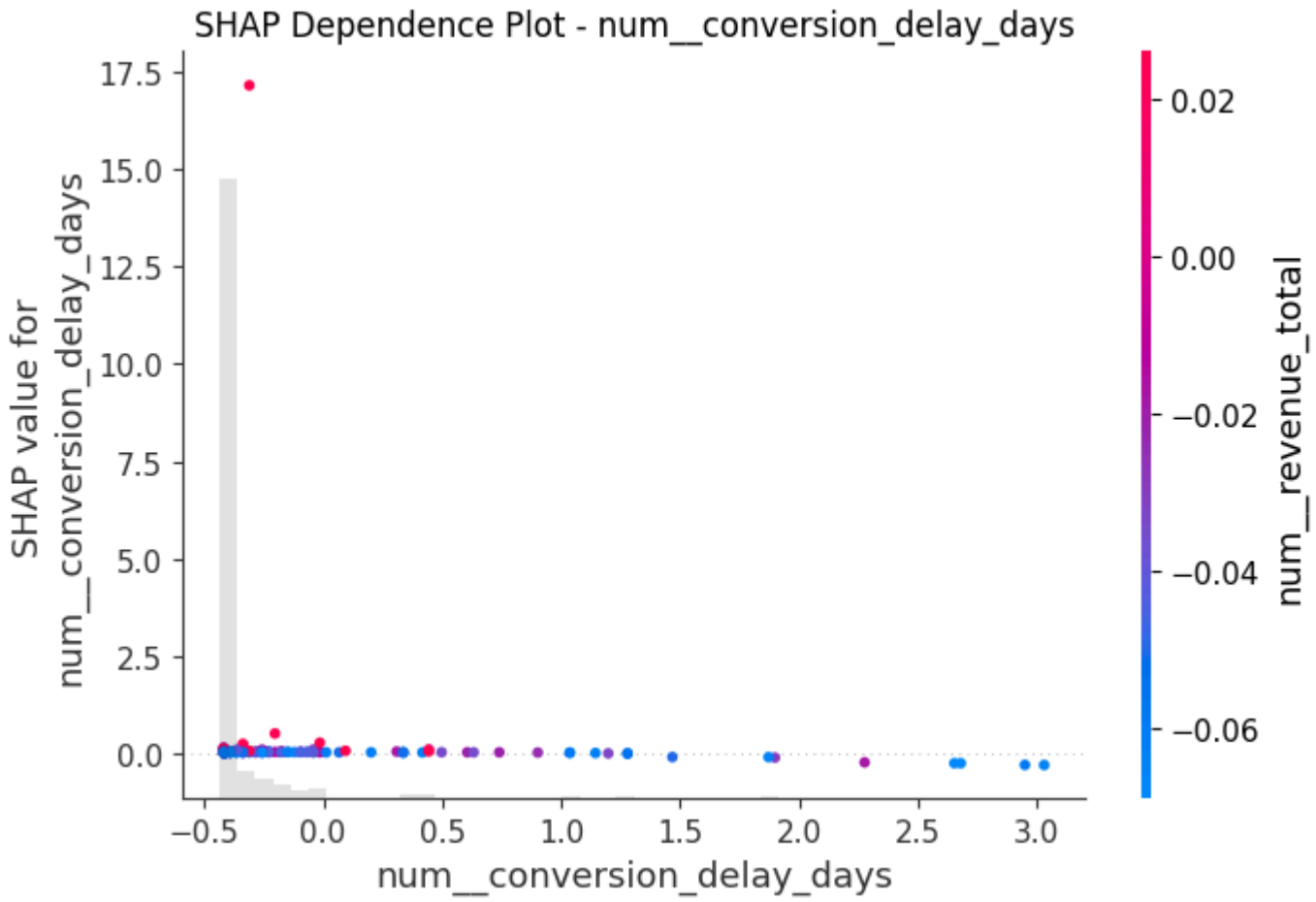
  - `num_avg_order_value`: Relación positiva, pero más compleja, con casos de alto impacto. Interacción con `revenue_total`.



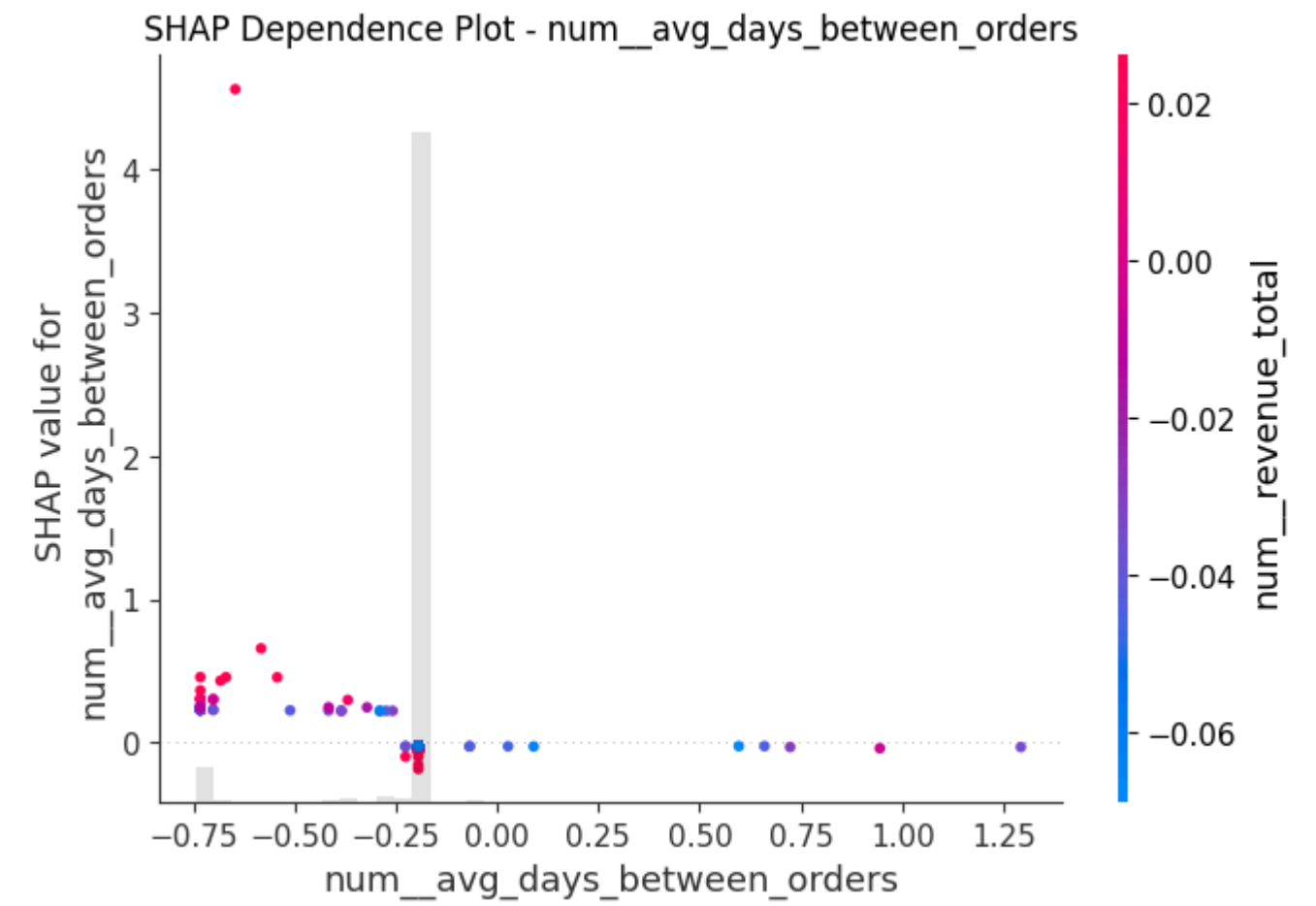
- `num_order_span_days`: Impacto cercano a cero para la mayoría, pero casos atípicos con alta influencia positiva.



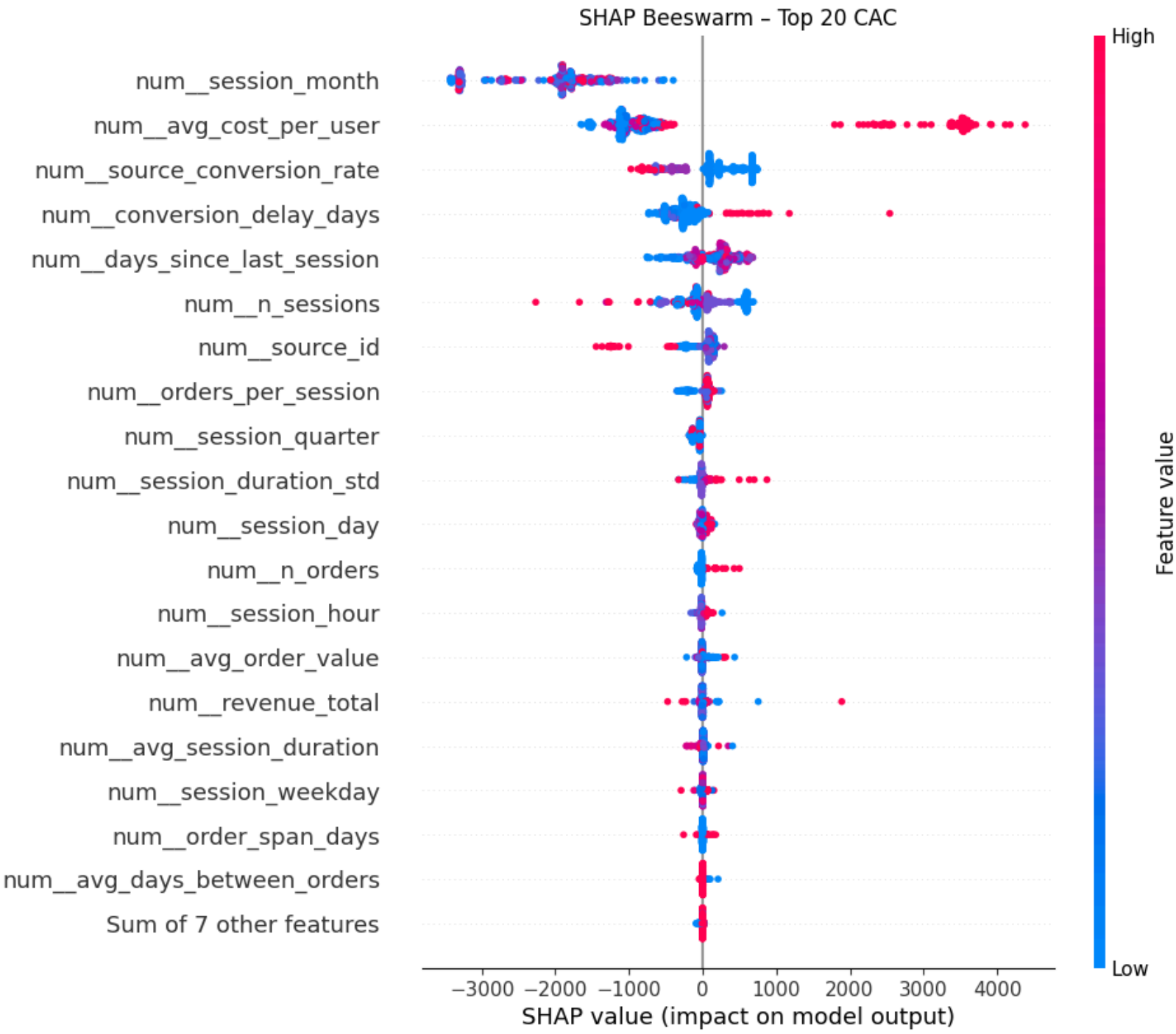
- `num_conversion_delay_days`: Mayormente neutral, excepto un caso atípico con alto impacto positivo. Posible interacción débil con `source_conversion_rate`.



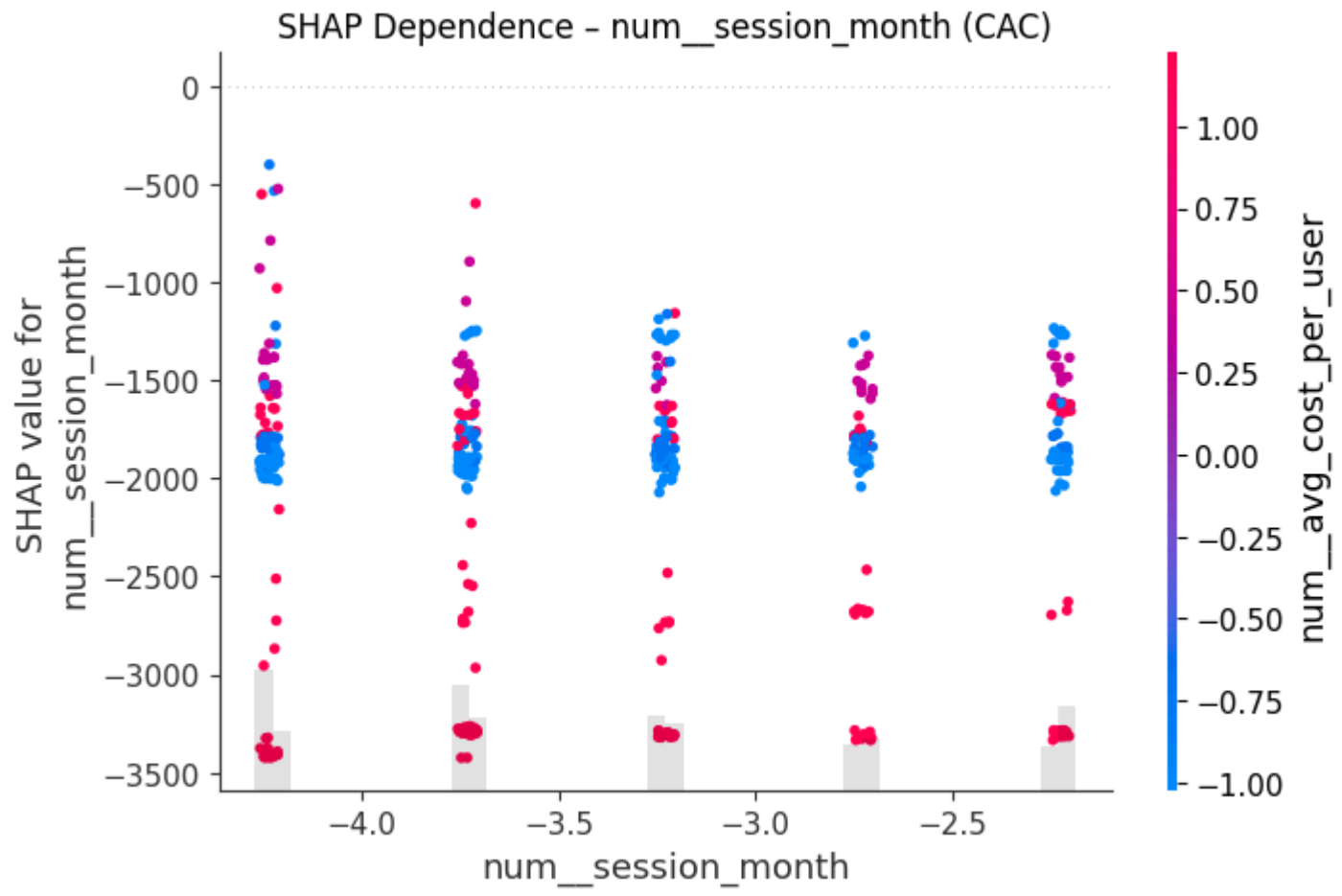
- `num_avg_days_between_orders`: Valores negativos (compras frecuentes/agrupadas) se asocian con mayor LTV predicho. Interacción con `revenue_total`.



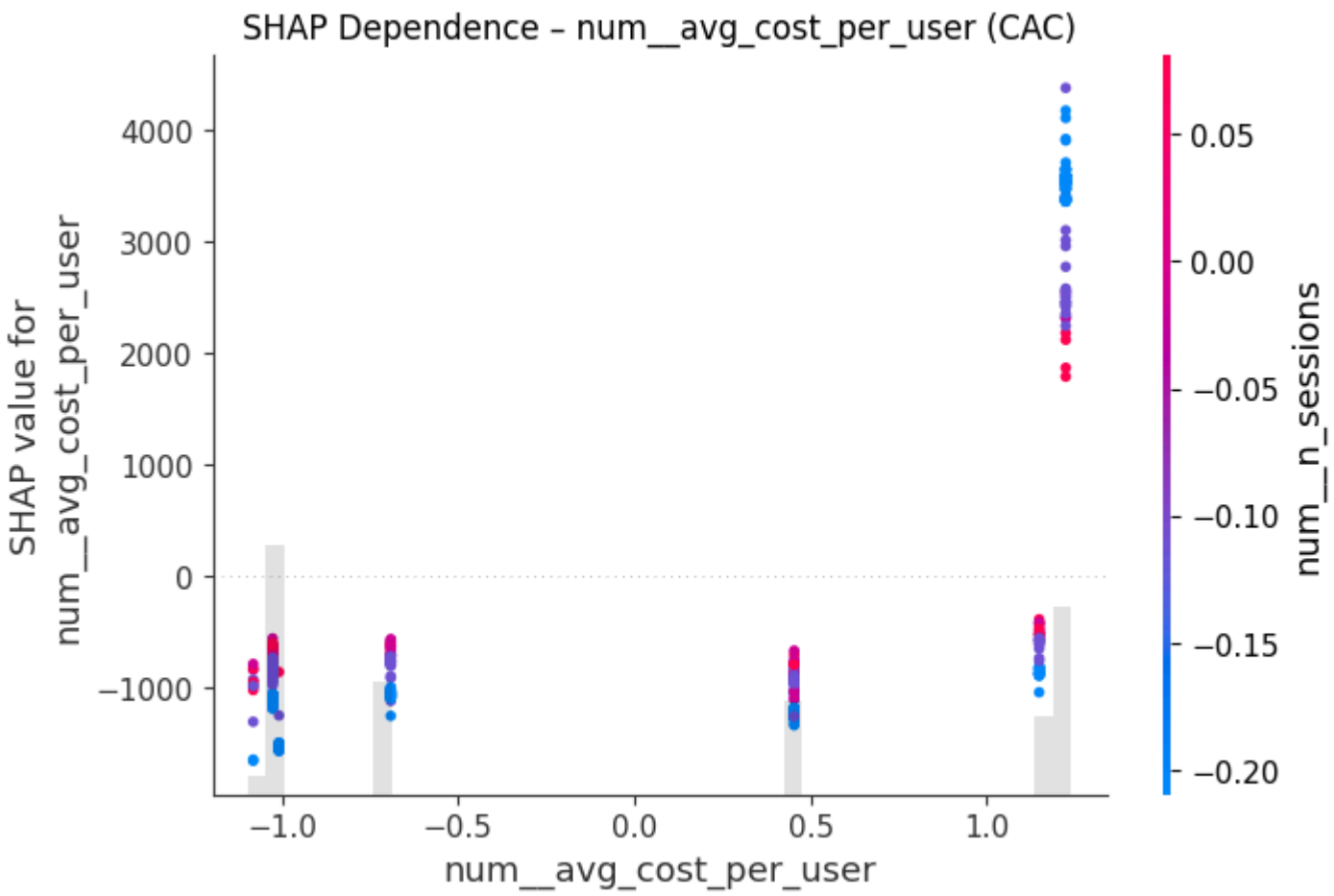
- *CAC:*
  - El gráfico *beeswarm* visualizó la importancia global de features como `session_month`, `avg_cost_per_user`, etc.



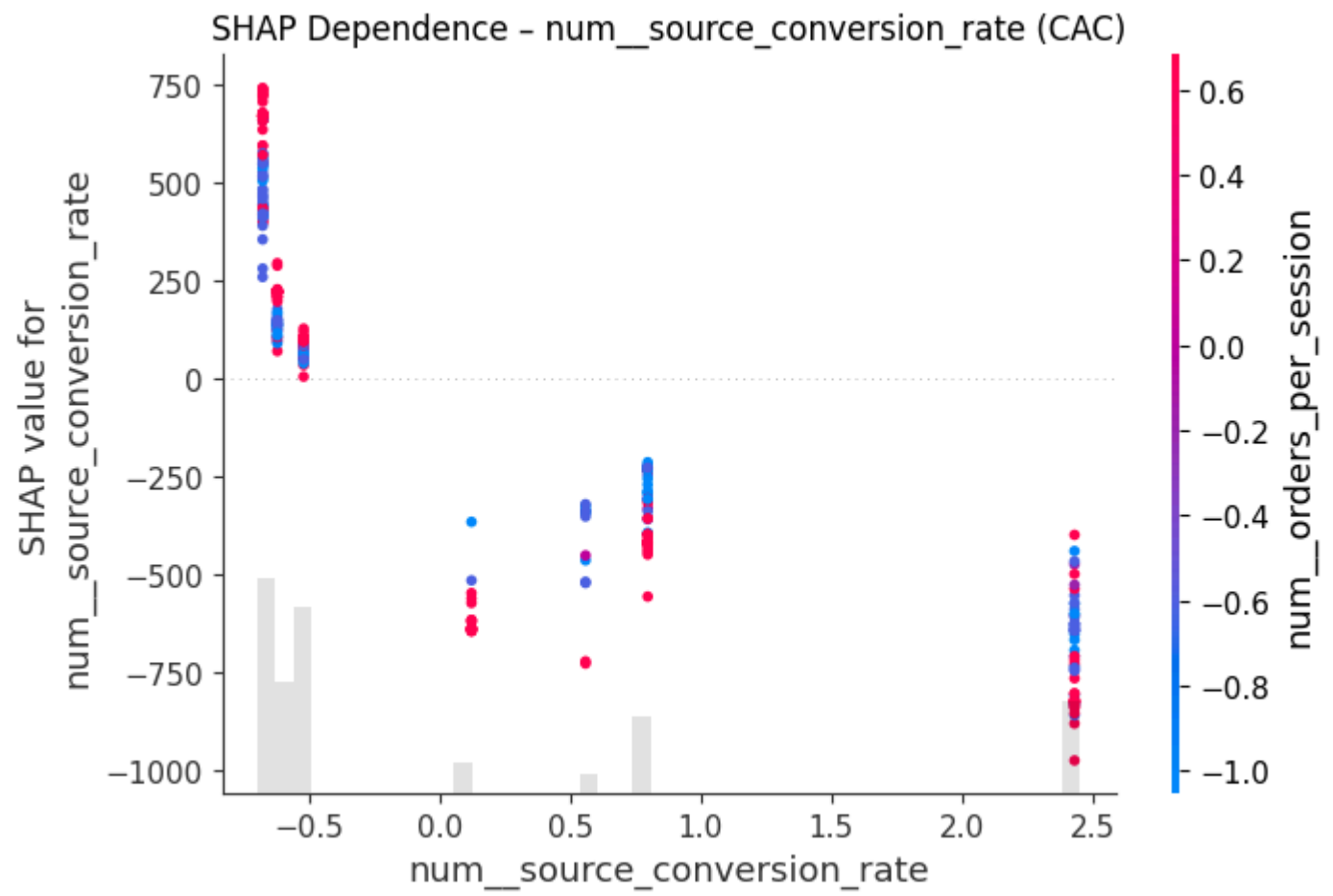
- Los gráficos de *dependencia* mostraron:
  - `num_session_month`: Valores más altos (meses específicos) tienden a disminuir el CAC predicho.



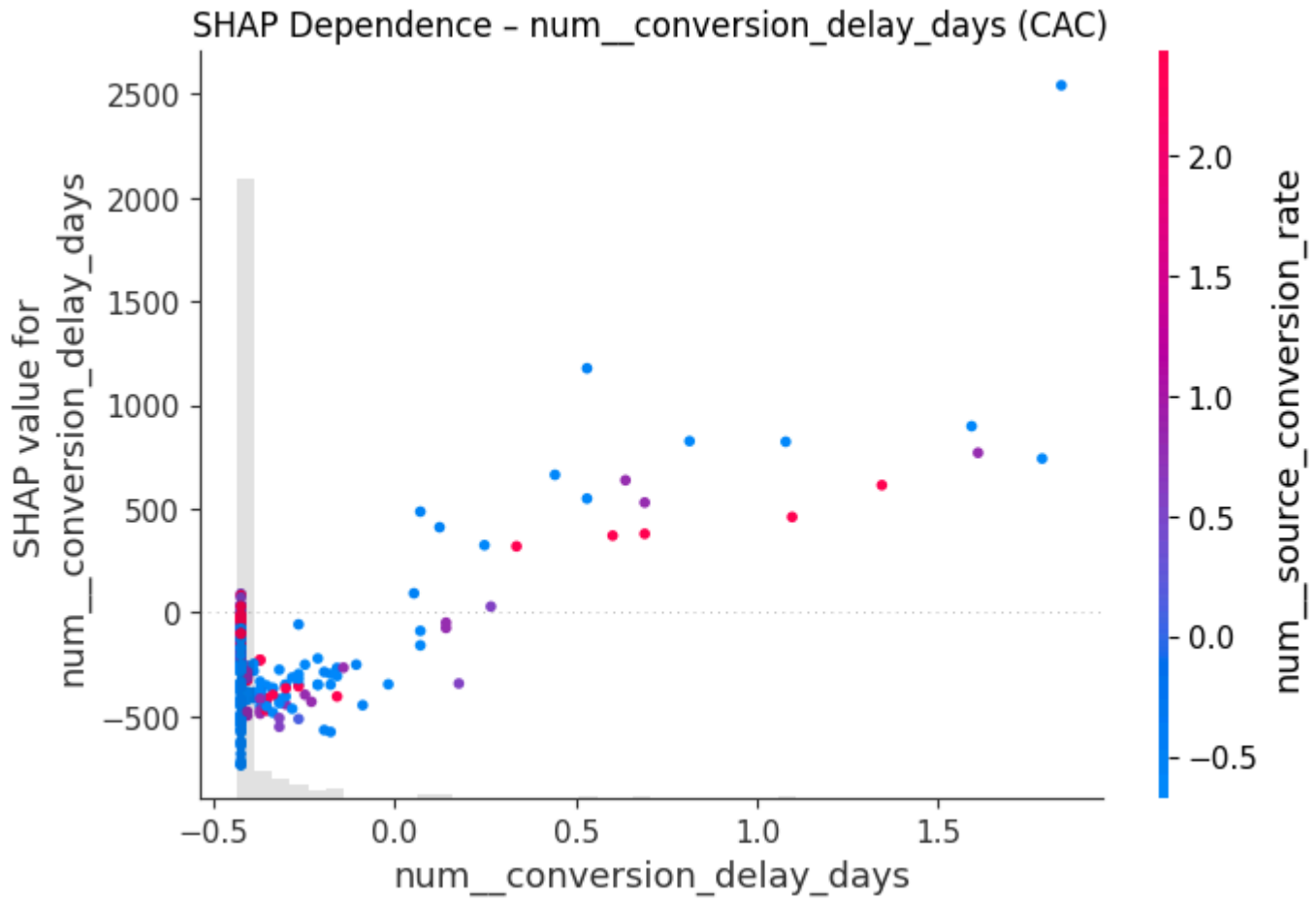
  - `num_avg_cost_per_user`: Relación positiva fuerte y esperada con el CAC predicho, con interacción notable con `n_sessions` (alto costo + pocas sesiones = CAC muy alto).



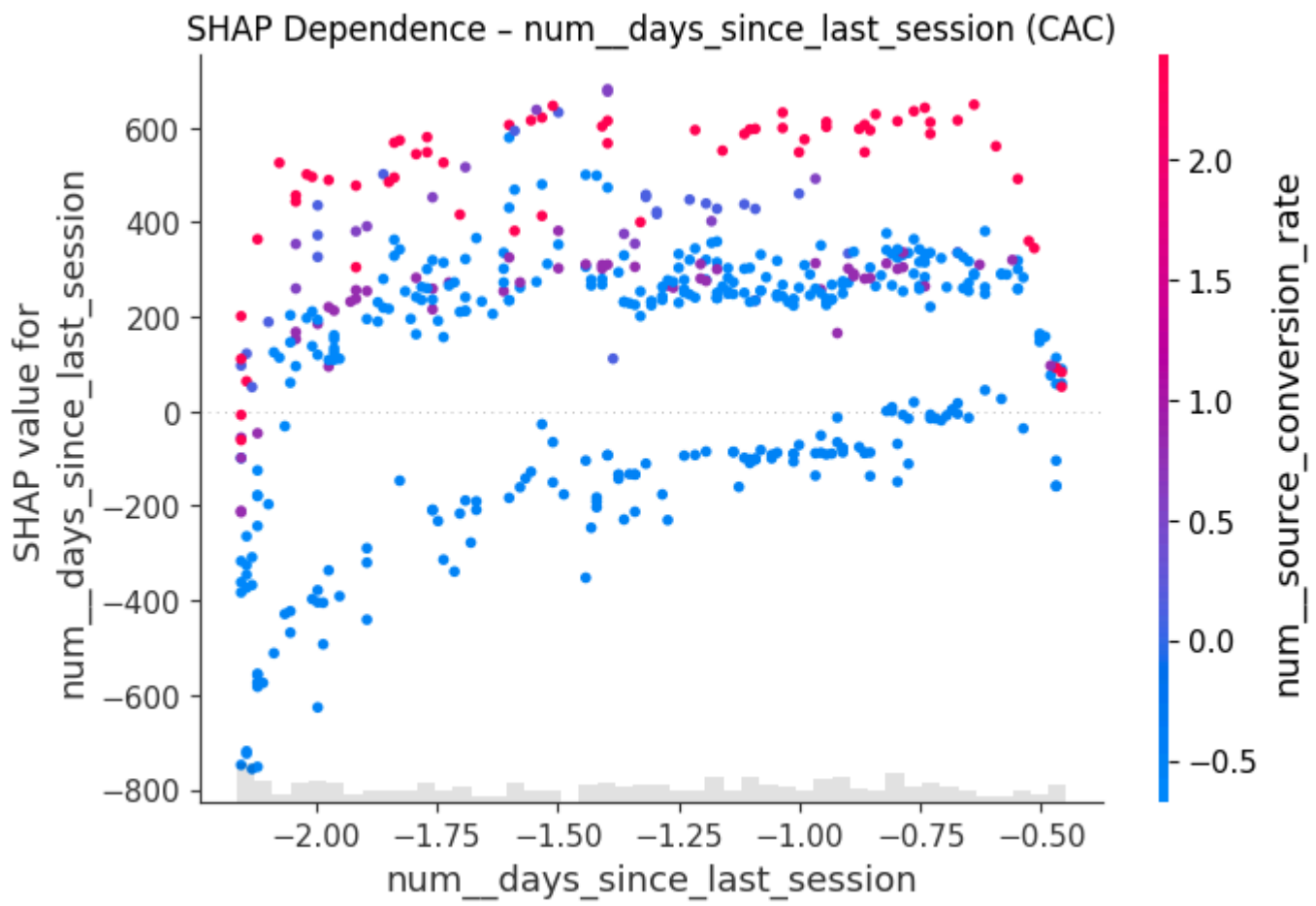
- `num_source_conversion_rate`: Tasas bajas aumentan el CAC predicho, tasas altas lo disminuyen, pero con variabilidad.



- `num_conversion_delay_days`: Mayor retraso generalmente aumenta el CAC predicho, posible interacción con `source_conversion_rate`.



- `num_days_since_last_session`: Mayor inactividad tiende a aumentar el CAC predicho, posible interacción con `source_conversion_rate`.



- **Análisis de Error por Segmento (`Final_Project_Showz_LTV_CAC.ipynb`):**
  - *LTV:* Se calculó el MAE agrupando por `device`. El MAE fue ligeramente superior para `desktop` (0.204) que para `touch` (0.122), indicando que el modelo es un poco menos preciso para usuarios de escritorio.
  - *CAC:* El MAE por `device` fue alto pero similar: `desktop` (2571) y `touch` (2454). Esto sugiere que, si bien el error general del modelo CAC es considerable, el tipo de dispositivo no es un factor principal que explique diferencias en la precisión de la predicción del CAC.

6. Despliegue (Deployment - Estrategia Simulada)

- **Aplicación del Modelo (`Estrategia_de_marketing_basada_en_simulación.ipynb`):** Los modelos finales seleccionados (`LTV_180_best_tscv.pkl`, `CAC_source_30_best_tscv.pkl`) se cargaron y se utilizaron para generar predicciones (`LTV_180_pred`, `CAC_source_30_pred`) para todos los usuarios en el `final_dataset.csv`.
- **Estimación de ROMI:**

    - Se calculó el ROMI predicho a nivel de usuario: `predicted_romi = (LTV_180_pred - CAC_source_30_pred) / CAC_source_30_pred`. Se aplicó `clip(lower=0.01)` a `CAC_source_30_pred` para evitar divisiones por cero.
    - Se agruparon los resultados por `source_id` y se calculó el `predicted_romi_avg` para cada fuente. Las fuentes 9 y 10 mostraron los ROMIs promedio más altos y positivos (3.67 y 7.97 respectivamente), mientras que las demás fuentes mostraron ROMIs promedio negativos.
- **Simulación de Escenarios (Presupuesto Total = $30,000):**
    - *Escenario Baseline:* Presupuesto distribuido equitativamente entre las 9 fuentes con datos (~$3,333 por fuente). Se calculó el retorno total multiplicando el presupuesto por fuente por su `predicted_romi_avg` y sumando. **Retorno Total Predicho: ~$15,623**.
    - *Escenario +10% a Fuente Top:* Se identificó la fuente 10 como la de mayor ROMI. Se incrementó su presupuesto en un 10% y se rebalanceó el presupuesto restante proporcionalmente entre las demás fuentes para mantener el total en $30,000. Se calculó el retorno total. **Retorno Total Predicho: ~$18,079**.
    - *Escenario ROMI Optimized:* El presupuesto de $30,000 se asignó únicamente a las fuentes con `predicted_romi_avg > 0` (fuentes 9 y 10), de forma proporcional a su ROMI predicho. Se calculó el retorno total. **Retorno Total Predicho: ~$198,392**.
- **Recomendación Final:** Basándose en las simulaciones con los datos predichos, se recomendó adoptar la estrategia **"ROMI Optimized"** por su retorno esperado significativamente superior.
- **Asignación de Presupuesto Recomendada:**
    - Fuente 9: 31.5% ($9,451.28)
    - Fuente 10: 68.5% ($20,548.72)
    - Otras Fuentes: 0%
- **Beneficio Cuantificado:** La ganancia estimada de la estrategia óptima sobre el escenario baseline, basada en las predicciones del modelo, es de **~$182,769**.
- **Próximos Pasos Sugeridos:**
    1. Implementar gradualmente la nueva asignación de presupuesto, comenzando por las fuentes 9 y 10.
    2. Establecer un sistema de monitorización para rastrear el LTV, CAC y ROMI reales por fuente y comparar con las predicciones.
    3. Planificar reentrenamientos periódicos del modelo (ej. trimestralmente) con datos actualizados para mantener la precisión y adaptarse a posibles cambios en el mercado o comportamiento del usuario.
    4. Investigar mejoras en la ingeniería de características, especialmente para LTV, buscando variables que complementen la fuerte influencia del `revenue_total`. Explorar por qué el modelo CAC tiene un MAE elevado y si se puede mejorar.
