Dahlia Time-to-Depletion Modeling

🎯 Objective: Predict how many days it will take for an inventory SKU to deplete after listing, using regression modeling.

# 1. Environment Setup

• Installed: catboost, feature\_engine

• Imported: pandas, numpy, matplotlib, plotly, sklearn, BigQuery libs

# 2. Data Extraction & Merging

• Pulled inventory, products, offers, orders, categories from BigQuery.

• Merged related tables and aligned schemas using SKU, seller, expiry\_date.

# 3. Data Cleaning & Feature Formatting

• Removed missing brand/category/subcategory rows.

• Normalized text: lowercase, removed underscores, dropped duplicates.

• Converted date columns to datetime; fixed inconsistent data types.

# 4. Core vs Non-Core Inventory

• Split inventory into CORE and NON\_CORE streams.

• Handled differently due to differing update frequency and completeness.

# 5. Offer-Inventories Merge

• For each inventory record, merged closest matching offer (if exists).

• Computed `offer\_day\_diff\_updated\_inv` as time lag to offers.

# 6. Inventory Depletion Tracking

• Shifted inventory rows to compare previous and current state.

• Computed `depletion` and `time` (in days).

• Applied conditional logic based on product category thresholds.

# 7. Dummy Generation for Time Variation

• Added synthetic examples to teach model about:

– High/low shelf life → scaled time accordingly

– High/low depletion → scaled time in opposite direction

• Ensured balanced exposure across inventory patterns

# 8. Listing Type Assignment

• Classified listings as: `fresh`, `obsolete`, `excess`, `damaged`, `made\_to\_order`

• Based on shelf life, recovery rate, and priority logic

# 9. Liquidation Channel Simulation

• Exploded each row into multiple channels (pollen, retailer, disposal, etc.)

• Applied weighted depletion per channel based on SKU priority (p1, p2, p3)

• Corrected `time` and `shelf\_life\_remaining\_days` to preserve numeric integrity

# 10. Final Preprocessing for Regression

• Removed rows with invalid or missing `time`, `total\_units\_prev`, or `depletion`.

• Computed `depletion\_percent = depletion / total\_units\_prev`

• Imputed missing shelf life with priority-based medians.

# 11. Training Dataset Preparation

• Features: sku\_number, brand, category, seller, listing\_condition, etc.

• Target: `time` (days between listings)

• Categorical columns: encoded using CatBoost handling.

# 12. Model Training: CatBoost Regressor

• Objective: RMSE minimization to predict listing depletion time

• Parameters: learning\_rate = 0.8, depth = 5, early stopping = 25 rounds

• Trained with ~470K examples after filtering and deduplication

# ✅ Business Value

• Enables prioritizing listings with low expected shelf time

• Improves liquidation planning by predicting sales velocity

• Supports dynamic pricing and markdown planning with actionable time estimates