ETL Pipeline Report

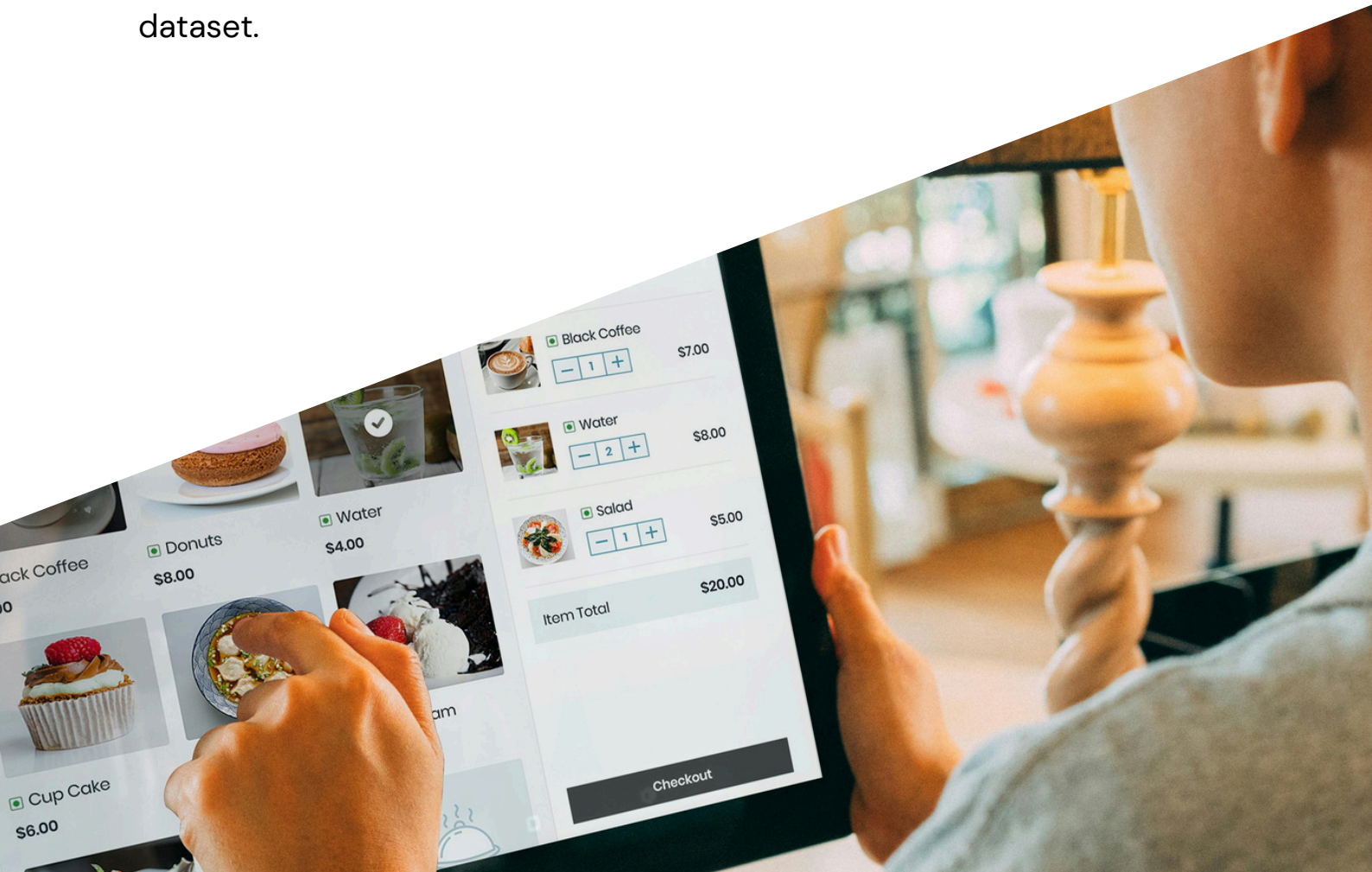
DATA NINJAS

Sales Data Cleaning & Transformation

2025

Overview

The dataset sample_dirty_dataset.csv was processed using a custom ETL (Extract, Transform, Load) pipeline to clean and standardize the data. The resulting cleaned dataset, sales_data_cleaned.csv, contains 242 rows and 12 columns. This report details the cleaning operations performed, the methods used, and the rationale behind each step, addressing issues like duplicates, missing values, inconsistent formats, and data validity specific to the dataset.



Data Characteristics

Original Columns

- name
- email
- birthdate
- signup_date
- category
- purchase_amount

Vision

- Missing values in name (5), email (17), and purchase_amount (12 "N/A" values).
- Inconsistent category values (e.g., "furniture" vs. "Furniture").
- Invalid/future birthdate values (e.g., 2007-02-10).
- Future signup_date values (e.g., 2025-04-14).
- Non-numeric purchase_amount entries ("N/A").
- Unnormalized date formats and potential whitespace in strings.
- No duplicates found.

2025

Cleaning Operations Performed

- Duplicate Handling
- Whitespace Trimming
- Name Cleaning and Imputation
- Email Standardization and Imputation

Duplicate Handling

How

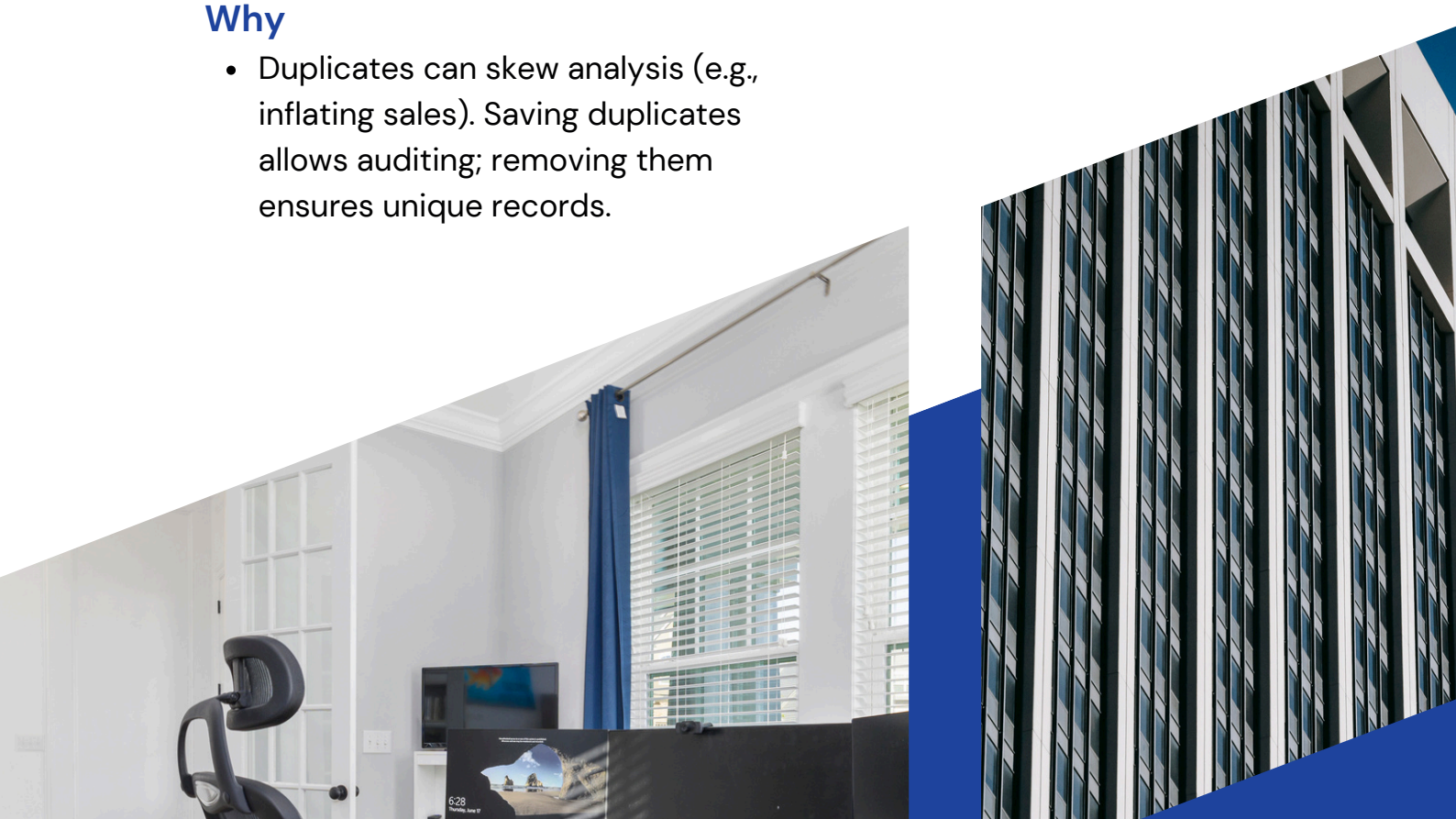
- Checked for duplicates using df.duplicated(). If found, saved to sales_duplicates.csv and removed using df.drop_duplicates(). Initialized duplicates as None to avoid UnboundLocalError.

Result

- No duplicates found (0 rows).

Why

- Duplicates can skew analysis (e.g., inflating sales). Saving duplicates allows auditing; removing them ensures unique records.



Cleaning Operations Performed Cont.

Whitespace Trimming

How

- Applied df[col].str.strip() to all string columns (name, email, category) for columns with object dtype.

Result

- Removed leading/trailing spaces, ensuring consistency (e.g., " Furniture" → "Furniture").

Why

- Whitespace can cause mismatches in comparisons or grouping, affecting data quality.

Name Cleaning and Imputation

How

- Removed titles (e.g., "Dr.", "Mr.", "DDS") using regex r'^(Dr|Mr|Mrs|Ms|MD|DDS|DVM)\.?\s+'. Imputed missing names with "Unknown".

Result

- Cleaned names (e.g., "Dr. John White DDS" → "John White") and filled 5 missing names with "Unknown".

Why

- Titles are irrelevant for analysis; imputing missing names preserves rows for other analyses (e.g., sales by category).

Cleaning Operations Performed Cont.

Email Standardization and Imputation

How

- Converted email to lowercase and stripped whitespace using str.lower().str.strip(). Imputed missing emails with "no_email@unknown.com (mailto:_email@unknown.com)".

Result

- Standardized emails (e.g., "BJones@Steele.org" → "bjones@steele.org") and filled 17 missing emails.

Why

- Consistent email formats aid matching or deduplication; imputation retains rows for non-email-based analyses.

Date Conversion and Normalization

How

- Converted birthdate and signup_date to datetime64[ns] using pd.to_datetime(errors='coerce'). Dropped rows with invalid dates (NaT). Added birthdate_str and signup_date_str with format dd-mm-yyyy using dt.strftime('%d-%m-%Y').

Result

- Standardized dates (e.g., birthdate 1976-10-03 → birthdate_str "03-10-1976"). Kept datetime columns for calculations.

Why

- Datetime format enables temporal calculations (e.g., age); string columns provide user-requested dd-mm-yyyy format for display/export.

Cleaning Operations Performed Cont.

Age Calculation and Suspicious Age Flagging

How

- Calculated age as datetime.now().year - df['birthdate'].year. Added age_flag column, marking ages <0 or >100 as "Suspicious", else "Valid".

Result

- Added age (e.g., 49 for 1976-10-03) and age_flag (all "Valid" in final data after filtering). Flagged entries like 2007-02-10 (age 18, valid).

Why

- Age is useful for demographic analysis; flagging suspicious ages identifies potential errors without premature deletion.

Signup Date Validation and Days Since Signup

How

- Dropped rows with future signup_date (beyond April 24, 2025) and missing signup_date. Added days_since_signup as (pd.Timestamp.today() - df['signup_date']).dt.days.

Result

- Removed 10 rows with future signup_date (e.g., 2025-04-14). Added days_since_signup (e.g., 485 for 2023-12-26).

Why

- Future dates indicate errors; days_since_signup enables customer tenure analysis.

Cleaning Operations Performed Cont.

Category Standardization

How

- Converted category to lowercase, mapped to standardized values (e.g., "furniture" → "Furniture") using a dictionary, and dropped missing category rows.

Result

- Normalized categories (e.g., "toys" → "Toys"). No missing categories in final data.

Why

- Consistent categories ensure accurate grouping for analysis (e.g., sales by category).

Purchase Amount Cleaning

How

- Converted purchase_amount to numeric with pd.to_numeric(errors='coerce'), replacing "N/A" with NaN. Filled NaN with column mean.

Result

- Converted 12 "N/A" values to mean (~250.57), ensuring all values are numeric.

Why

- Numeric values are required for calculations (e.g., total sales); mean imputation preserves data distribution..

Cleaning Operations Performed Cont.

Data Consistency Check

How

- Added date_consistency column, flagging rows as "Invalid" if signup_date < birthdate, else "Valid"..

Result

- All rows marked "Valid" (no cases where signup_date precedes birthdate).

Why

- Ensures logical consistency, as signup cannot occur before birth.

Final Dataset

Shape

- 242 rows × 12 columns (name, email, birthdate, signup_date, category, purchase_amount, age, age_flag, birthdate_str, days_since_signup, signup_date_str, date_consistency).

Key Transformations

- Removed 10 rows with future signup_date.
- Imputed missing name and email.
- Filled purchase_amount NaNs with mean.
- Added derived columns (age, age_flag, days_since_signup, date_consistency, normalized date strings).

Output

- Saved as sales_data_cleaned.csv.

2025

Rationale Summary

This pipeline ensures a clean, consistent dataset ready for analysis while preserving as much data as possible. Let me know if you need further details or additional cleaning steps!

Data Quality

- Removed duplicates, invalid dates, and standardized formats to ensure reliable analysis.

Data Preservation

- Imputed missing name, email, and purchase_amount to minimize row loss, dropping only where necessary (e.g., missing category, invalid dates).

Usability

- Added derived columns (age, days_since_signup, normalized date strings) to enhance analytical value.

Error Detection

- Flagged suspicious ages and checked date consistency to highlight potential issues without aggressive filtering.