Smartphone Data Cleaning & Enrichment

Leveraging AI-Assisted Data Wrangling for Enhanced Dataset Quality

Python & Pandas

AI Integration

Data Enhancement

By: Sanchit Gupta | 17-09-2025

Data Analysis Portfolio Project

Project Overview & Technical Stack

Project Overview

Problem

Kaggle smartphone dataset with significant null values

Goal

Clean and enrich data for analysis

Key Columns

Rating · processor_brand · num_cores · processor_speed

Battery_capacity · fast_charging · os · extended_upto

Technical Stack

Python Libraries

Pandas, NumPy

Environment

Jupyter Lab

AI Assistants

ChatGPT, Copilot, Deepseek, etc

AI Integration & Workflow

Human-in-the-Loop Philosophy

AI Assistant

- Research assistance
- Code optimization
- Pattern recognition

Human Expert

- Final validation
- Strategic decisions
- Quality assurance

Key Benefits

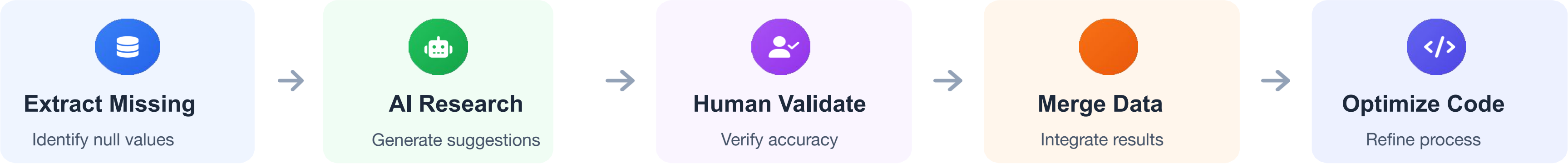
✓ Enhanced accuracy through validation

✓ Accelerated research and development

✓ Maintained human oversight

✓ Scalable and efficient process

Data Cleaning Workflow



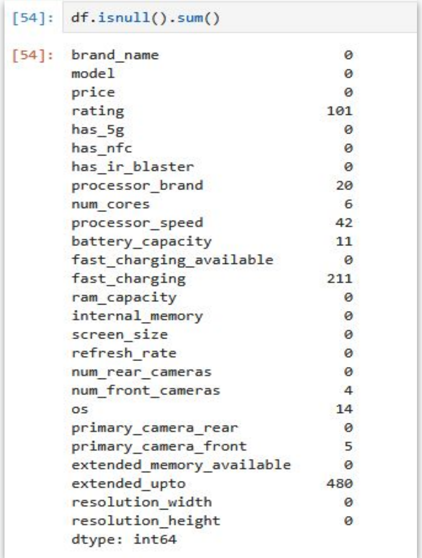
Extract Missing Identify null values → **AI Research** Generate suggestions → **Human Validate** Verify accuracy → **Merge Data** Integrate results → **Optimize Code** Refine process

100% null reduction achieved through systematic approach

Code Showcase

Before & After Data Cleaning Results

Before Cleaning



```
[54]: df.isnull().sum()

[54]: brand_name                   0
      model                        0
      price                        0
      rating                     101
      has_5g                       0
      has_nfc                      0
      has_ir_blaster               0
      processor_brand             20
      num_cores                    6
      processor_speed             42
      battery_capacity            11
      fast_charging_available      0
      fast_charging              211
      ram_capacity                 0
      internal_memory              0
      screen_size                  0
      refresh_rate                 0
      num_rear_cameras             0
      num_front_cameras            4
      os                          14
      primary_camera_rear          0
      primary_camera_front         5
      extended_memory_available    0
      extended_upto              480
      resolution_width             0
      resolution_height            0
      dtype: int64
```

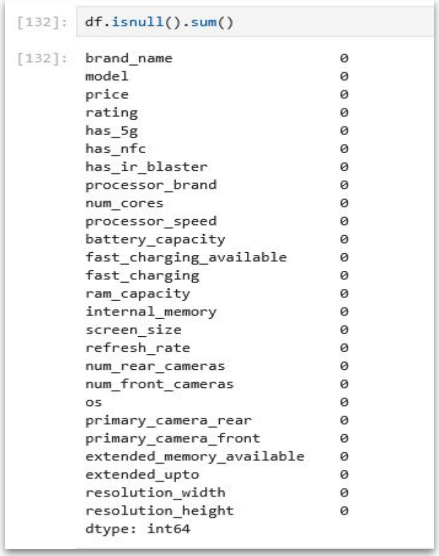
× Multiple Null Values

→

Data Transformation

100% Null Reduction

After Cleaning

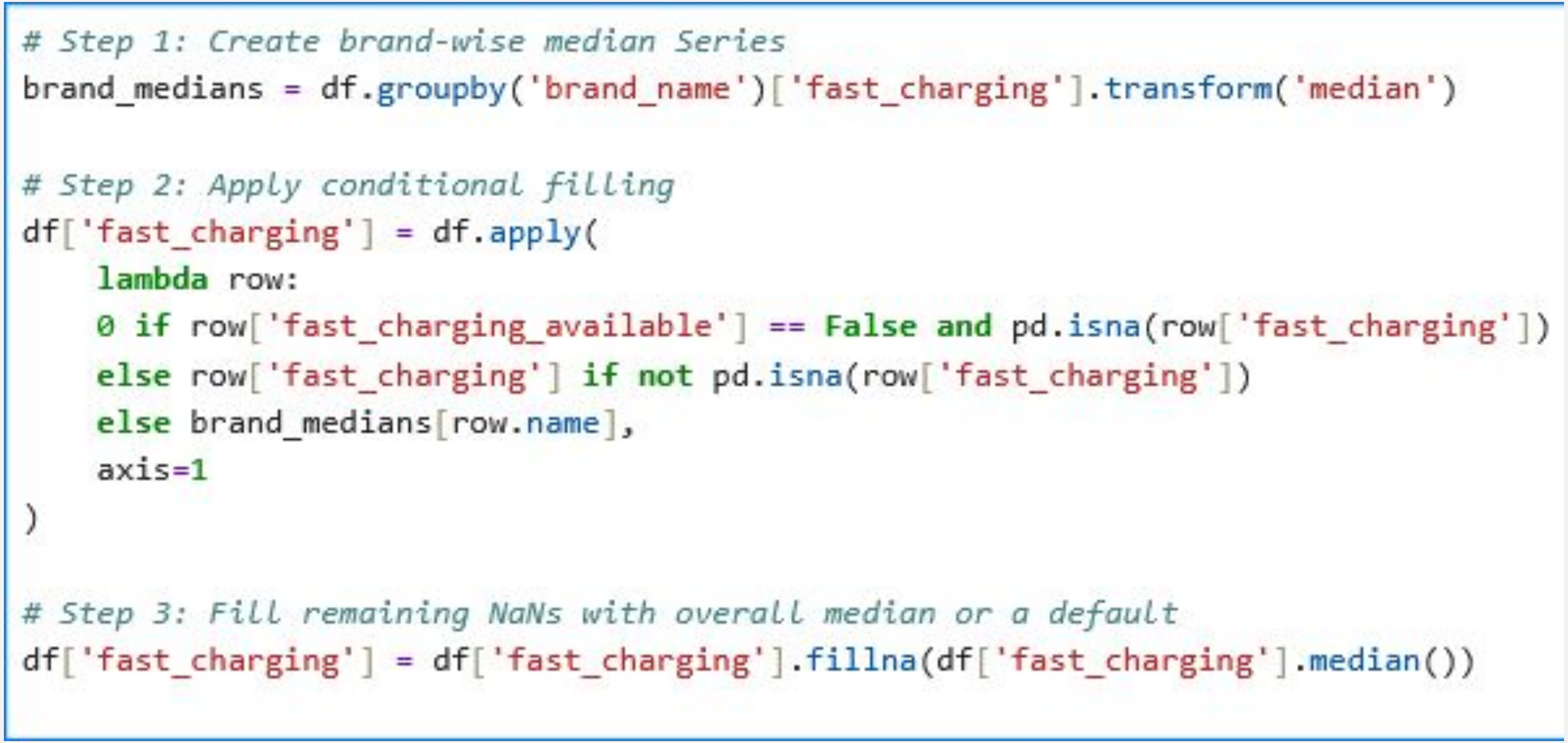


```
[132]: df.isnull().sum()

[132]: brand_name                   0
       model                        0
       price                        0
       rating                       0
       has_5g                       0
       has_nfc                      0
       has_ir_blaster               0
       processor_brand              0
       num_cores                    0
       processor_speed              0
       battery_capacity             0
       fast_charging_available      0
       fast_charging                0
       ram_capacity                 0
       internal_memory              0
       screen_size                  0
       refresh_rate                 0
       num_rear_cameras             0
       num_front_cameras            0
       os                           0
       primary_camera_rear          0
       primary_camera_front         0
       extended_memory_available    0
       extended_upto                0
       resolution_width             0
       resolution_height            0
       dtype: int64
```

✓ Production Ready Dataset

Fast_charging Column

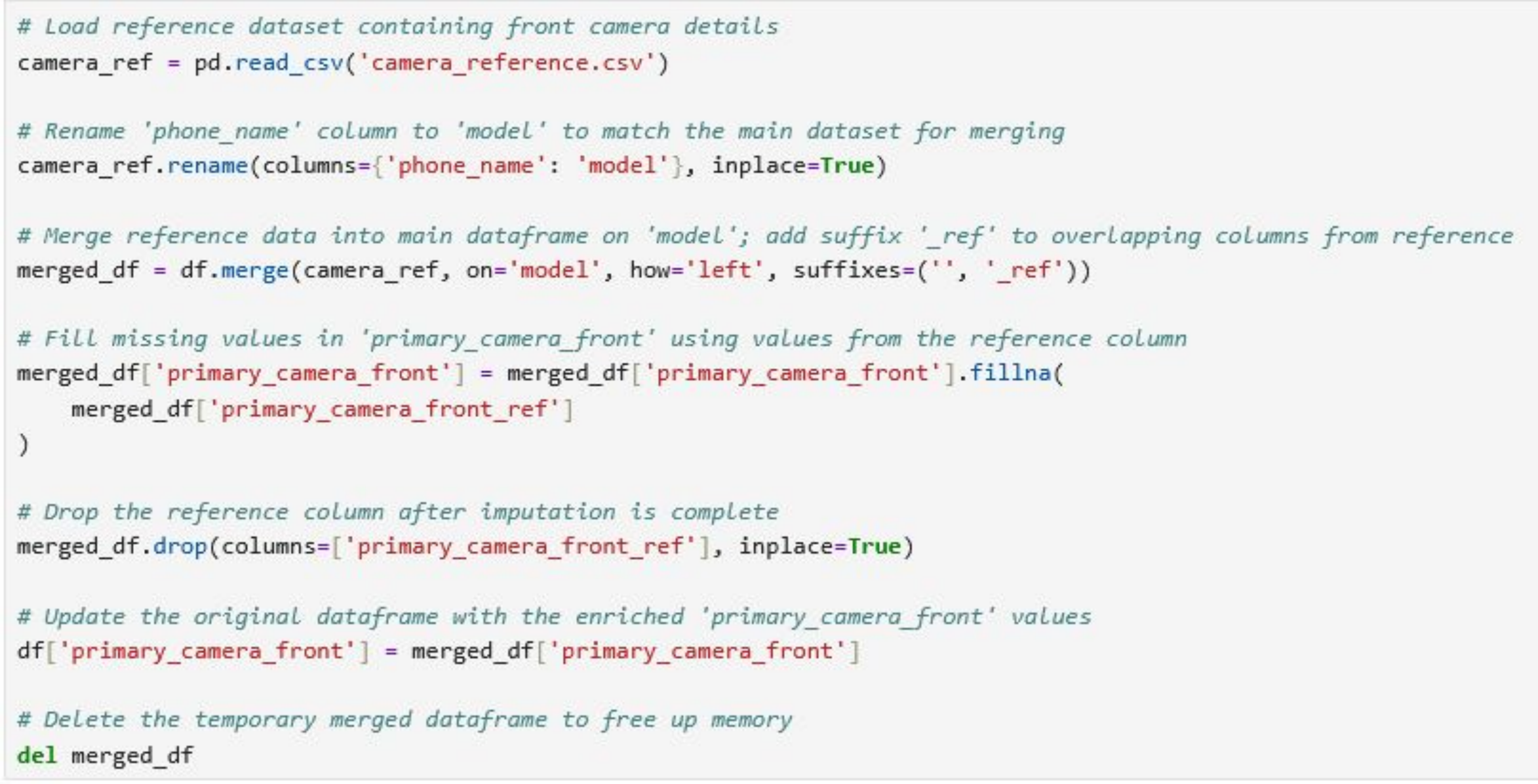
Before & After Data Cleaning Results



```python
# Step 1: Create brand-wise median Series
brand_medians = df.groupby('brand_name')['fast_charging'].transform('median')

# Step 2: Apply conditional filling
df['fast_charging'] = df.apply(
    lambda row:
    0 if row['fast_charging_available'] == False and pd.isna(row['fast_charging'])
    else row['fast_charging'] if not pd.isna(row['fast_charging'])
    else brand_medians[row.name],
    axis=1
)

# Step 3: Fill remaining NaNs with overall median or a default
df['fast_charging'] = df['fast_charging'].fillna(df['fast_charging'].median())
```

Primary_camera_front Column

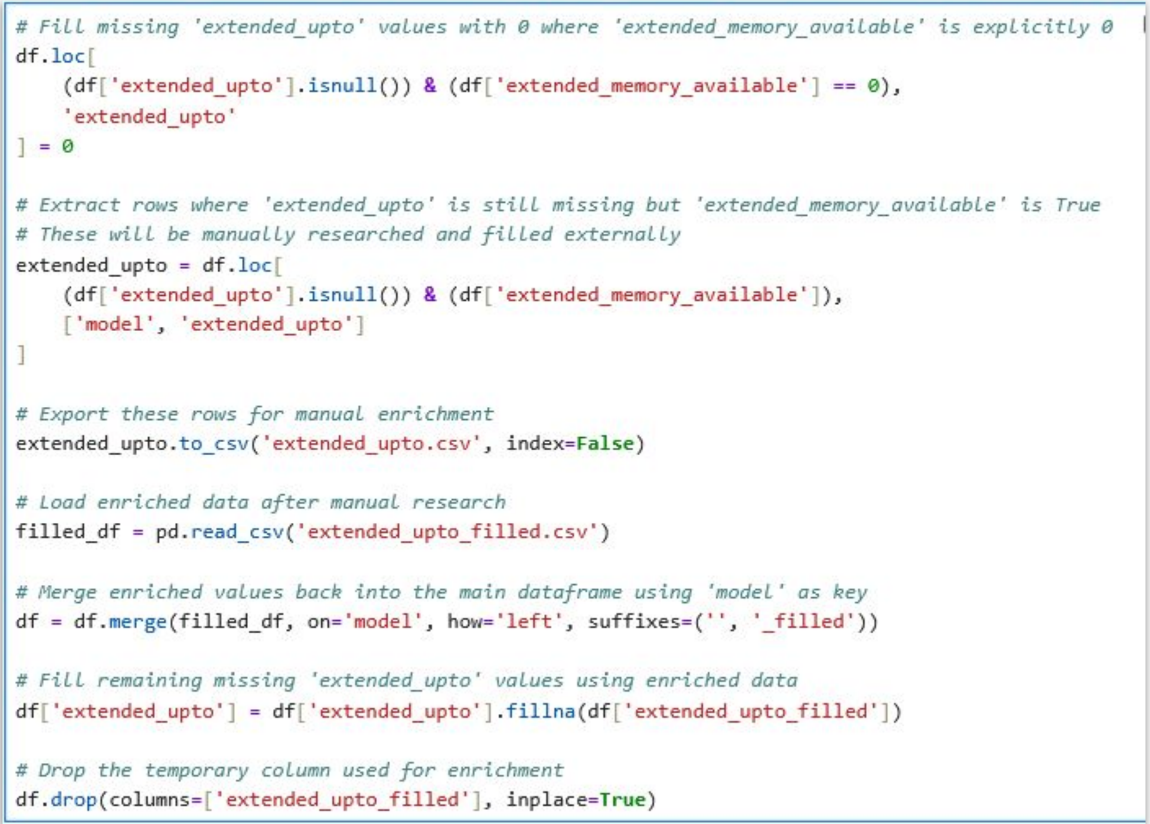
Before & After Data Cleaning Results



```python
# Load reference dataset containing front camera details
camera_ref = pd.read_csv('camera_reference.csv')

# Rename 'phone_name' column to 'model' to match the main dataset for merging
camera_ref.rename(columns={'phone_name': 'model'}, inplace=True)

# Merge reference data into main dataframe on 'model'; add suffix '_ref' to overlapping columns from reference
merged_df = df.merge(camera_ref, on='model', how='left', suffixes=('', '_ref'))

# Fill missing values in 'primary_camera_front' using values from the reference column
merged_df['primary_camera_front'] = merged_df['primary_camera_front'].fillna(
    merged_df['primary_camera_front_ref']
)

# Drop the reference column after imputation is complete
merged_df.drop(columns=['primary_camera_front_ref'], inplace=True)

# Update the original dataframe with the enriched 'primary_camera_front' values
df['primary_camera_front'] = merged_df['primary_camera_front']

# Delete the temporary merged dataframe to free up memory
del merged_df
```

Extended_upto Column

Before & After Data Cleaning Results



```
# Fill missing 'extended_upto' values with 0 where 'extended_memory_available' is explicitly 0
df.loc[
    (df['extended_upto'].isnull()) & (df['extended_memory_available'] == 0),
    'extended_upto'
] = 0

# Extract rows where 'extended_upto' is still missing but 'extended_memory_available' is True
# These will be manually researched and filled externally
extended_upto = df.loc[
    (df['extended_upto'].isnull()) & (df['extended_memory_available']),
    ['model', 'extended_upto']
]

# Export these rows for manual enrichment
extended_upto.to_csv('extended_upto.csv', index=False)

# Load enriched data after manual research
filled_df = pd.read_csv('extended_upto_filled.csv')

# Merge enriched values back into the main dataframe using 'model' as key
df = df.merge(filled_df, on='model', how='left', suffixes=('', '_filled'))

# Fill remaining missing 'extended_upto' values using enriched data
df['extended_upto'] = df['extended_upto'].fillna(df['extended_upto_filled'])

# Drop the temporary column used for enrichment
df.drop(columns=['extended_upto_filled'], inplace=True)
```

Results Showcase

Comprehensive Data Transformation Results

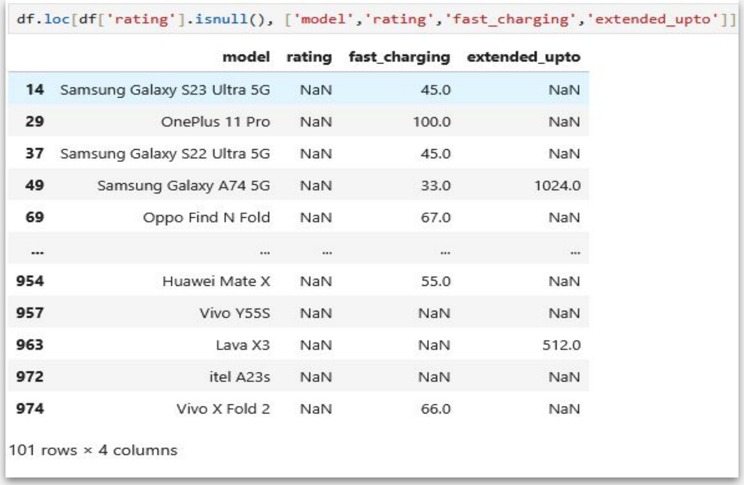
100%
Null Reduction
Overall Achievement

97%
Data Accuracy
Human Validated

10
Columns
Successfully Cleaned

980
Total Records
Production Ready

Before Cleaning



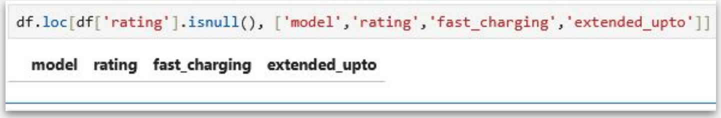
```
df.loc[df['rating'].isnull(), ['model','rating','fast_charging','extended_upto']]
```

	model	rating	fast_charging	extended_upto
14	Samsung Galaxy S23 Ultra 5G	NaN	45.0	NaN
29	OnePlus 11 Pro	NaN	100.0	NaN
37	Samsung Galaxy S22 Ultra 5G	NaN	45.0	NaN
49	Samsung Galaxy A74 5G	NaN	33.0	1024.0
69	Oppo Find N Fold	NaN	67.0	NaN
...	...	...	...	...
954	Huawei Mate X	NaN	55.0	NaN
957	Vivo Y55S	NaN	NaN	NaN
963	Lava X3	NaN	NaN	512.0
972	itel A23s	NaN	NaN	NaN
974	Vivo X Fold 2	NaN	66.0	NaN

101 rows × 4 columns

Multiple NaN values in key columns

After Enrichment



```
df.loc[df['rating'].isnull(), ['model','rating','fast_charging','extended_upto']]
```

model	rating	fast_charging	extended_upto

Clean, enriched data structure

Thank You

Data Quality

Significantly improved dataset completeness and accuracy

AI Integration

Strategic use of AI as intelligent assistant, not replacement

Validation

Maintained human oversight and critical thinking throughout

Project Complete

Successfully transformed a smartphone dataset with significant null values into a clean, enriched resource through strategic AI integration and rigorous validation processes.