Pipeline Guide - LOE Forecasting System

Quick Start

```
cd Main_project

# Run full pipeline (uses config.py settings)
python scripts/run_pipeline.py
```

Configuration (`src/config.py`)

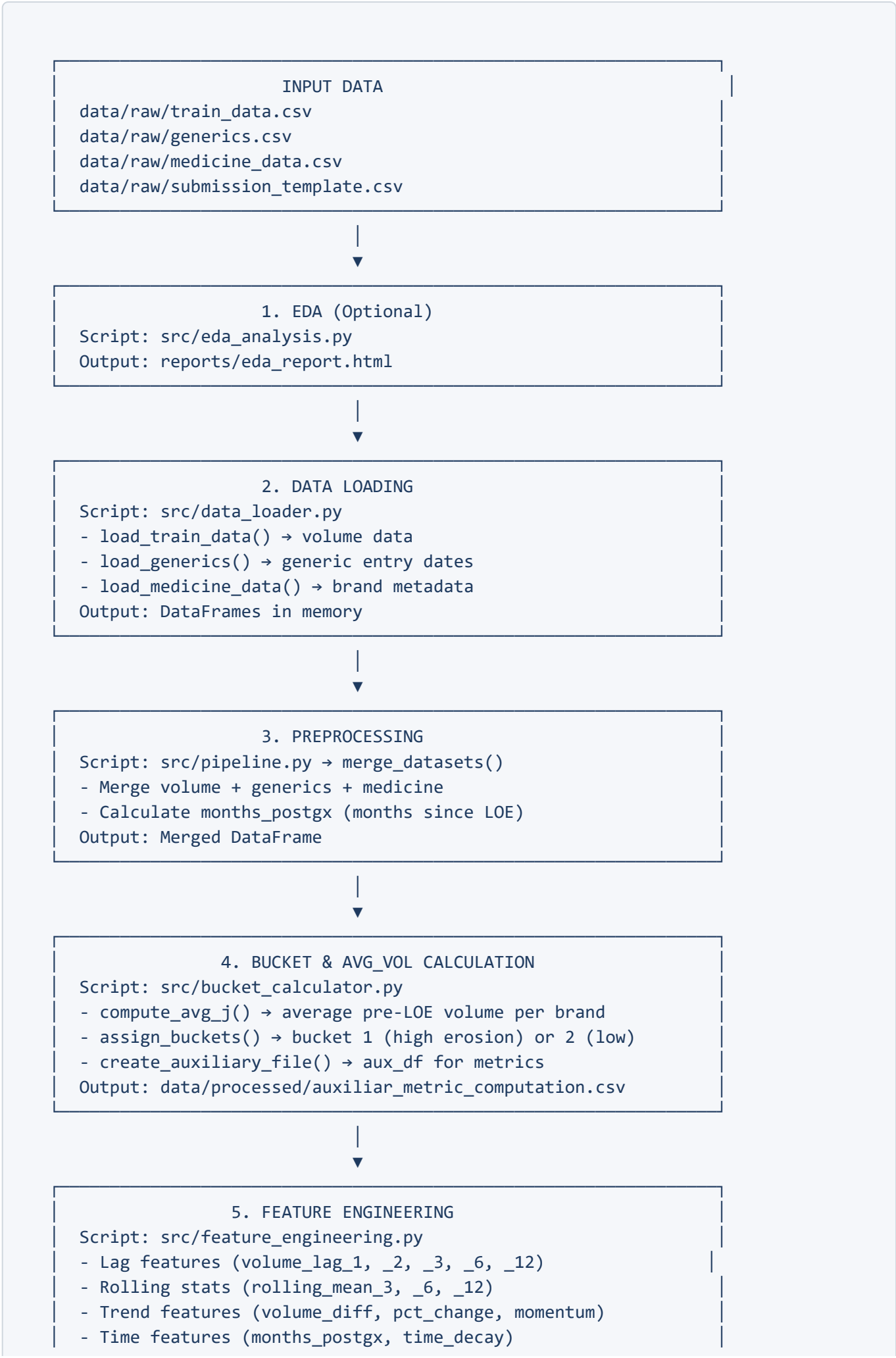
All pipeline settings are controlled here. **No CLI arguments needed.**

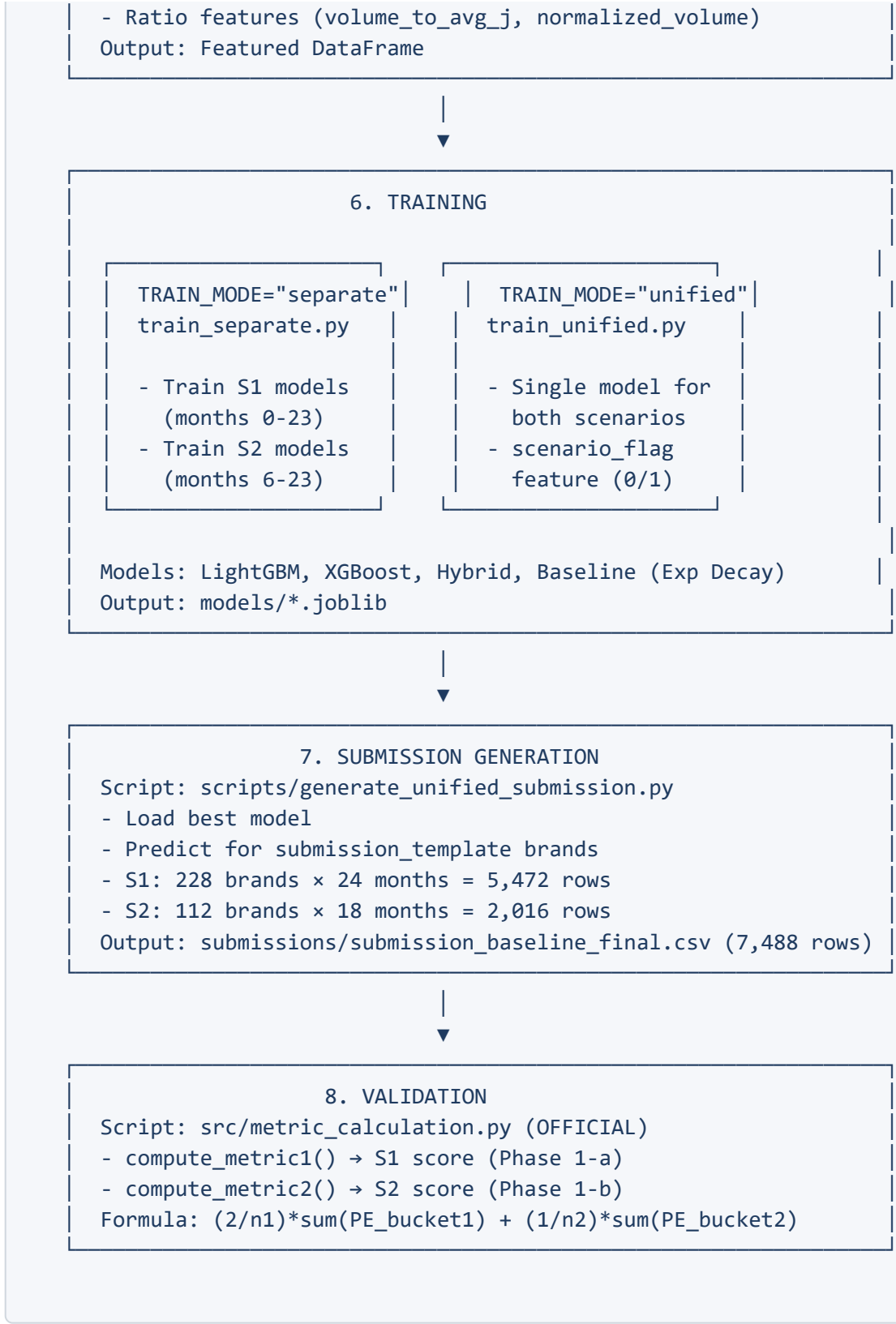
Setting	Options	Description
`TRAIN_MODE`	`"separate"` / `"unified"`	Train S1/S2 separately or single unified model
`TEST_MODE`	`True` / `False`	Use subset of brands for quick testing
`RUN_SCENARIO`	`"both"` / `"s1"` / `"s2"`	Which scenarios to run
`RUN_EDA`	`True` / `False`	Run exploratory data analysis
`RUN_TRAINING`	`True` / `False`	Train models
`RUN_SUBMISSION`	`True` / `False`	Generate submission file
`RUN_VALIDATION`	`True` / `False`	Validate submission format

Models Enabled

```
MODELS_ENABLED = {
    'baseline_exp_decay': True,
    'lightgbm': True,
    'xgboost': True,
    'hybrid_lightgbm': True,
    'arima': False,  # Slow
}
```

Pipeline Flow


INPUT DATA
data/raw/train_data.csv
data/raw/generics.csv
data/raw/medicine_data.csv
data/raw/submission_template.csv
1. EDA (Optional)
Script: src/eda_analysis.py
Output: reports/eda_report.html
2. DATA LOADING
Script: src/data_loader.py
- load_train_data() → volume data
- load_generics() → generic entry dates
- load_medicine_data() → brand metadata
Output: DataFrames in memory
3. PREPROCESSING
Script: src/pipeline.py → merge_datasets()
- Merge volume + generics + medicine
- Calculate months_postgx (months since LOE)
Output: Merged DataFrame
4. BUCKET & AVG_VOL CALCULATION
Script: src/bucket_calculator.py
- compute_avg_j() → average pre-LOE volume per brand
- assign_buckets() → bucket 1 (high erosion) or 2 (low)
- create_auxiliary_file() → aux_df for metrics
Output: data/processed/auxiliar_metric_computation.csv
5. FEATURE ENGINEERING
Script: src/feature_engineering.py
- Lag features (volume_lag_1, _2, _3, _6, _12)
- Rolling stats (rolling_mean_3, _6, _12)
- Trend features (volume_diff, pct_change, momentum)
- Time features (months_postgx, time_decay)



File Reference

Scripts (`scripts/`)

File	Purpose	Run Command
`run_pipeline.py`	**Main orchestrator** - runs full pipeline	`python scripts/run_pipeline.py`

File	Purpose	Run Command
`train_models.py`	Training entry point (calls train_separate or train_unified)	Called by run_pipeline.py
`generate_unified_submission.py`	Generate submission CSV	Called by run_pipeline.py
`run_full_pipeline.py`	Alternative orchestrator	`python scripts/run_full_pipeline.py`

Source Modules (`src/`)

File	Purpose
`config.py`	**Central configuration** - all settings here
`data_loader.py`	Load CSV files from data/raw/
`pipeline.py`	Merge datasets, coordinate preprocessing
`bucket_calculator.py`	Calculate avg_vol, assign buckets (1=high erosion, 2=low)
`feature_engineering.py`	Create ML features (lags, rolling, trends)
`models.py`	Model classes: GradientBoostingModel, BaselineModels, HybridPhysicsMLModel
`evaluation.py`	Cross-validation, model comparison
`metric_calculation.py`	**OFFICIAL** competition metrics
`eda_analysis.py`	Exploratory data analysis

Training Modules (`src/training/`)

File	Purpose
`train_separate.py`	Separate training: S1 and S2 trained independently
`train_unified.py`	Unified training: single model with scenario_flag feature

Scenario Definitions (`src/scenarios/`)

File	Purpose
`scenarios.py`	Centralized scenario definitions (S1: months 0-23, S2: months 6-23)

Inputs

File	Location	Description

File	Location	Description
`train_data.csv`	`data/raw/`	Historical volume data (country, brand, date, volume)
`generics.csv`	`data/raw/`	Generic entry dates per brand
`medicine_data.csv`	`data/raw/`	Brand metadata
`submission_template.csv`	`data/raw/`	Template with brands/months to predict

Outputs

File	Location	Description
`auxiliar_metric_computation.csv`	`data/processed/`	avg_vol, bucket per brand
`*.joblib`	`models/`	Trained model files
`model_comparison_*.csv`	`reports/`	Training results comparison
`submission_baseline_final.csv`	`submissions/`	**Final submission file**
`eda_report.html`	`reports/`	EDA visualizations

Competition Metrics

Scenario 1 (Phase 1-a) - Zero Actuals

- Predict months 0-23 post-LOE
- No actual data given at prediction time
- PE formula includes terms for months 0-5, 6-11, 12-23

Scenario 2 (Phase 1-b) - Six Actuals

- Predict months 6-23 post-LOE
- Given actual data for months 0-5
- PE formula includes terms for months 6-11, 12-23

Final Score Formula

```
Score = (2/n1) × Σ(PE_bucket1) + (1/n2) × Σ(PE_bucket2)
```

- Bucket 1: High erosion brands (mean_ratio ≤ 0.25) - **weighted 2x**
- Bucket 2: Low erosion brands - **weighted 1x**
- Lower score = better

Example Workflow

```
# 1. Edit config.py as needed
#    Set TRAIN_MODE = "separate" or "unified"
#    Set TEST_MODE = False for full run

# 2. Run pipeline
cd Main_project
python scripts/run_pipeline.py

# 3. Check outputs
#    - models/ for trained models
#    - reports/ for comparison CSVs
#    - submissions/ for final submission
```

Training Modes Explained

Separate Mode (`TRAIN_MODE = "separate"`)

- Trains **two independent models**
- S1 model: trained on months 0-23, predicts with no actuals
- S2 model: trained on months 6-23, uses early months as features
- **Pros**: Specialized models for each scenario
- **Cons**: Less training data per model

Unified Mode (`TRAIN_MODE = "unified"`)

- Trains **one model** for both scenarios
- Uses `scenario_flag` feature (0=S1, 1=S2)
- S2 samples include early post-LOE summary features
- **Pros**: More training data, shared learning
- **Cons**: May not specialize as well

Troubleshooting

Issue	Solution
Import errors	Run from `Main_project/` directory
Missing data	Check `data/raw/` has all CSV files
Memory issues	Set `TEST_MODE = True` in config.py
Slow training	Disable `arima` in MODELS_ENABLED