🔬 Data Preprocessing to Model Selection Rationale

Pharma Sales Analysis and Forecasting at Small Scale

> **This document explains the logical connection between specific data preprocessing techniques and the choice of multiple forecasting models (ARIMA, Prophet, LSTM), demonstrating why each preprocessing step is essential and how time series analysis informs model selection for different drug categories.**

Table of Contents

---

1. Executive Summary

The Core Question

Why use these specific preprocessing techniques, and how does time series analysis inform which model is best for each drug category?

Quick Answer

Preprocessing Technique	Problem Solved	Model Selection Impact
Temporal Aggregation	Reduces 600K transactions to 302 weekly observations	Enables statistical modeling (small n)
Seasonality Analysis	Detects annual/weekly patterns	Seasonal → SARIMA/Prophet; Non-seasonal → ARIMA
Stationarity Testing	Checks if mean/variance are stable	Non-stationary → Differencing (d > 0)

Preprocessing Technique	Problem Solved	Model Selection Impact
ACF/PACF Analysis	Identifies autocorrelation patterns	Determines ARIMA (p, q) parameters
Approximate Entropy	Measures predictability	High entropy → Expect lower accuracy
MinMax Scaling	Normalizes data to [0,1]	Required for LSTM neural networks
Sequence-to-Supervised	Converts time series to X,y format	Enables LSTM training

Model Selection Summary

Drug Category	Characteristics	Best Model
N02BE (Paracetamol)	Strong seasonality, low entropy	**SARIMA/Prophet**
R03 (Respiratory)	Strong seasonality, high outliers	**Prophet**
R06 (Antihistamines)	Strong seasonality	**SARIMA/Prophet**
M01AB/M01AE (Anti-inflammatory)	Weak patterns, high entropy	**Naïve baseline**
N05B/N05C (Sedatives)	High randomness	**Average baseline**

---

2. Data Characteristics & Challenges

2.1 Original Data Structure

Characteristic	Value
Raw Records	600,000 transactions
Time Period	6 years (2014-2019)
Source	Single pharmacy POS system
Categories	8 ATC drug classifications

Drug Categories (ATC Classification):

Code	Drug Type	Expected Patterns
M01AB	Anti-inflammatory (Diclofenac)	Weather-dependent
M01AE	Anti-inflammatory (Ibuprofen)	Weather-dependent
N02BA	Analgesics (Aspirin)	General demand
N02BE	Analgesics (Paracetamol)	Cold/flu season

Code	Drug Type	Expected Patterns
N05B	Anxiolytics	Prescription-driven
N05C	Hypnotics/Sedatives	Prescription-driven
R03	Respiratory drugs	Seasonal (allergies)
R06	Antihistamines	Seasonal (allergies)

2.2 Key Challenges Requiring Specific Preprocessing

Challenge 1: Too Many Observations for Analysis

Problem:

- 600,000 raw transactions
- High noise at transaction level
- Computational expense for modeling

Solution: Aggregate to weekly frequency (302 observations)

Challenge 2: Unknown Seasonality Patterns

Problem:

- Some drugs have seasonal demand (flu season, allergy season)
- Others have constant demand (prescription medications)
- Need to identify patterns before choosing models

Solution: Seasonal decomposition + box plot analysis

Challenge 3: Mixed Stationarity

Problem:

- Statistical models (ARIMA) require stationary data
- Some series have trends, others don't
- Differencing degree varies by category

Solution: ADF and KPSS tests per category

Challenge 4: Unknown ARIMA Parameters

Problem:

- ARIMA requires (p, d, q) parameters
- Different categories need different parameters
- Manual selection is subjective

Solution: ACF/PACF analysis + Auto-ARIMA

Challenge 5: LSTM Data Format Requirements

Problem:

- ☑ → [y]
- Raw time series is sequential, not tabular
- Neural networks need normalized data

Solution: Sequence splitting + MinMax scaling

---

3. Preprocessing Pipeline Overview

Complete Three-Phase Pipeline

```
┌─────────────────────────────────────────────────────────────────┐
│              PHASE 1: DATA PREPARATION                          │
│  600K transactions → Aggregate → 302 weekly observations        │
└─────────────────────────────────────────────────────────────────┘
                            │
                            ▼
┌─────────────────────────────────────────────────────────────────┐
│              PHASE 2: TIME SERIES ANALYSIS                      │
│   ┌────────────┐   ┌────────────┐   ┌──────────────┐           │
│   │ Seasonality │   │Stationarity│   │Autocorrelation│          │
│   │  Analysis   │   │  Testing   │   │   Analysis   │           │
│   └────────────┘   └────────────┘   └──────────────┘           │
│         │                │                 │                    │
│         ▼                ▼                 ▼                    │
│  Seasonal vs       d parameter       p, q parameters            │
│  Non-seasonal      (0 or 1)          (ACF/PACF)                 │
└─────────────────────────────────────────────────────────────────┘
                            │
                            ▼
┌─────────────────────────────────────────────────────────────────┐
│              PHASE 3: MODEL-SPECIFIC PREPROCESSING              │
│   ┌────────────┐   ┌────────────┐   ┌────────────┐             │
│   │   ARIMA    │   │  Prophet   │   │    LSTM    │             │
│   │  Raw data  │   │ ds, y cols │   │ MinMaxScale│             │
│   │  Optional  │   │ Seasonality│   │ Sequence   │             │
│   │  log/diff  │   │   config   │   │ splitting  │             │
│   └────────────┘   └────────────┘   └────────────┘             │
└─────────────────────────────────────────────────────────────────┘
                            │
                            ▼
┌─────────────────────────────────────────────────────────────────┐
│                      MODEL FORECASTING                          │
```

```
│  • Baseline: Naïve, Seasonal Naïve, Average                    │
│  • Statistical: ARIMA, SARIMA, Auto-ARIMA                      │
│  • Modern: Prophet, LSTM (Vanilla, Stacked, Bidirectional)     │
└────────────────────────────────────────────────────────────────┘
```

---

4. Preprocessing Step 1: Temporal Aggregation

4.1 Why Weekly Aggregation?

Frequency	Rows	Pros	Cons
Hourly	~52,560	Fine-grained patterns	Too noisy, computational
Daily	~2,190	Good for analysis	Still noisy for forecasting
Weekly ✓	302	Stable patterns	Optimal for small-scale
Monthly	~72	Very smooth	Too few observations

Why 302 Weekly Observations is Ideal:

Consideration	Weekly Data Advantage
Statistical power	302 > 100 observations minimum for ARIMA
Noise reduction	Smooths daily fluctuations
Seasonality capture	52 weeks × 6 years = clear annual patterns
Computational	Fast model training

4.2 Available Aggregation Files

```
# Multiple granularities available
saleshourly.csv    # ~52,560 rows - for daily pattern analysis
salesdaily.csv     # ~2,190 rows  - for seasonality detection
salesweekly.csv    # 302 rows     - PRIMARY forecasting dataset
salesmonthly.csv   # ~72 rows     - for trend visualization
```

How This Enables Model Selection:

- **Daily data:** Used for seasonality box plots (detect weekly/monthly patterns)
- **Weekly data:** Used for actual forecasting (optimal sample size)
- **Monthly data:** Used for trend visualization

---

5. Preprocessing Step 2: Time Series Analysis

5.1 Seasonality Detection

Box Plot Analysis:

```
# Monthly seasonality detection
sns.boxplot(data=dfatc_daily, x='Month', y='N02BE')  # Paracetamol
sns.boxplot(data=dfatc_daily, x='Month', y='R03')   # Respiratory
```

Results Summary:

Category	Annual Seasonality	Weekly Seasonality	Implication
N02BE	☑ Strong (Winter peak)	Moderate	Use SARIMA/Prophet with yearly seasonality
R03	☑ Strong (Spring peak)	Weak	Use SARIMA/Prophet with yearly seasonality
R06	☑ Strong (Spring peak)	Weak	Use SARIMA/Prophet with yearly seasonality
M01AB	◯ Weak	Weak	Try both seasonal and non-seasonal
M01AE	◯ Weak	Weak	Try both seasonal and non-seasonal
N05B	✕ None	None	Use non-seasonal ARIMA
N05C	✕ None	None	Use non-seasonal ARIMA
N02BA	◯ Weak	Weak	Try both approaches

Seasonal Decomposition:

```
from statsmodels.tsa.seasonal import seasonal_decompose
result = seasonal_decompose(df['N02BE'], freq=52, model='additive')
# Extracts: trend, seasonal, residual components
```

Residual Analysis Results:

Category	Residual % of Observed	Interpretation
N02BE	~15%	**High predictability** (trend+season explain 85%)

Category	Residual % of Observed	Interpretation
R06	~15%	**High predictability**
R03	~30%	Moderate predictability
M01AB	~25%	Moderate predictability
N05C	~35%	**Low predictability** (high noise)

5.2 Stationarity Testing

ADF Test Results:

```
from statsmodels.tsa.stattools import adfuller
dftest = adfuller(df['N02BA'], regression='ct', autolag='AIC')
```

Category	ADF Statistic	P-value	Result
M01AB	-4.21	0.004	☑ Stationary (d=0)
M01AE	-4.58	0.001	☑ Stationary (d=0)
N02BA	-2.71	**0.249**	✗ Non-stationary (d=1)
N02BE	-3.92	0.011	☑ Stationary (d=0)
N05B	-4.33	0.003	☑ Stationary (d=0)
N05C	-4.87	0.000	☑ Stationary (d=0)
R03	-4.12	0.006	☑ Stationary (d=0)
R06	-3.78	0.018	☑ Stationary (d=0)

KPSS Test Results:

Category	Result	Trend Stationarity
N02BE	$P < 0.05$	Non-stationary (has trend)
R03	$P < 0.05$	Non-stationary (has trend)
R06	$P < 0.05$	Non-stationary (has trend)
Others	$P > 0.05$	Trend stationary

Implication for Model Parameters:

Category	ADF Result	KPSS Result	ARIMA d

Category	ADF Result	KPSS Result	ARIMA d
N02BA	Non-stationary	Stationary	**d=1**
N02BE, R03, R06	Stationary	Non-stationary	Consider d=1 for trend
Others	Stationary	Stationary	**d=0**

5.3 Autocorrelation Analysis (ACF/PACF)

Purpose:

Plot	Shows	Determines
ACF	Correlation at each lag	MA order (q)
PACF	Direct correlation (controlling intermediate lags)	AR order (p)

Example Parameter Selection:

```
# If PACF cuts off at lag 2 → p = 2
# If ACF cuts off at lag 1 → q = 1
# Result: ARIMA(2, d, 1)
```

Selected ARIMA Parameters:

Category	(p, d, q)	Reasoning
M01AB	(0, 0, 0)	No significant autocorrelation (random walk)
M01AE	(2, 0, 0)	PACF cutoff at lag 2
N02BA	(5, 1, 1)	Complex pattern, needs differencing
N02BE	(0, 0, 0) seasonal	Seasonality dominates
N05B	(2, 0, 0)	AR(2) process
N05C	(0, 0, 5)	MA(5) process
R03	(0, 0, 0) seasonal	Seasonality dominates
R06	(0, 0, 0) seasonal	Seasonality dominates

5.4 Approximate Entropy (Predictability Score)

```
def ApEn(U, m, r):  # Approximate Entropy calculation
    ...
```

Results (Higher = Less Predictable):

```
Drug        Approximate Entropy    Predictability
─────────────────────────────────────────────────
N02BE       ████████████           HIGHEST (easiest)
R06         █████████████          High
R03         ███████████████        High
N05B        █████████████████      Moderate
N05C        ███████████████████    Moderate
N02BA       █████████████████████  Low
M01AB       ██████████████████████ Low
M01AE       ████████████████████████ LOWEST (hardest)
```

Implication:

Entropy Level	Categories	Model Expectation
Low entropy	N02BE, R06, R03	Models should perform well
High entropy	M01AE, M01AB	Even best models may not beat Naïve

---

6. Preprocessing Step 3: Data Transformation for Models

6.1 ARIMA: Minimal Preprocessing

```
# ARIMA works on raw or differenced data
X = df['M01AB'].values
model = ARIMA(X, order=(p, d, q))
```

Why Minimal?

- ARIMA handles differencing internally (d parameter)
- No scaling needed (regression-based)
- Optional: Log transform for stabilizing variance

6.2 Prophet: Column Renaming + Seasonality Config

```python
# Prophet requires specific column names
dfg = df.rename(columns={'datum': 'ds', 'N02BE': 'y'})

# Configure seasonality for seasonal series
model = Prophet(...)
model.add_seasonality(name='yearly', period=365.25, fourier_order=13)
```

Prophet Configuration by Category:

Category	Yearly Seasonality	Fourier Order
N02BE	☑ Yes	13
R03	☑ Yes	13
R06	☑ Yes	13
Others	✖ No	N/A

6.3 LSTM: Full Preprocessing Pipeline

Step 1: MinMax Scaling

```python
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler(feature_range=(0, 1))
X_scaled = scaler.fit_transform(X.reshape(-1, 1))
```

Why Scale to [0, 1]?

Reason	Explanation
Gradient flow	Prevents vanishing/exploding gradients
Faster convergence	Normalized inputs train faster
Equal feature weight	All features contribute equally

Step 2: Sequence-to-Supervised Transformation

```python
def split_sequence(sequence, n_steps=5):
    X, y = [], []
    for i in range(len(sequence)):
        end_ix = i + n_steps
        if end_ix > len(sequence)-1:
            break
```

```
            X.append(sequence[i:end_ix])
            y.append(sequence[end_ix])
        return array(X), array(y)
```

Transformation Example (n_steps=5):

```
Original: [10, 15, 12, 18, 20, 22, 19, 25]

Transformed:
X                       y
[10, 15, 12, 18, 20] → 22
[15, 12, 18, 20, 22] → 19
[12, 18, 20, 22, 19] → 25
```

Step 3: Reshape for LSTM Input

```
X = X.reshape((X.shape[0], X.shape[1], n_features))
# Shape: (samples, timesteps, features) = (n, 5, 1)
```

LSTM Input Format:

Dimension	Meaning	Value
Samples	Number of training examples	~247
Timesteps	Lookback window	5 weeks
Features	Variables per timestep	1 (univariate)

---

7. Model Selection by Drug Category

7.1 Decision Framework

```
                         ┌──────────────────────┐
                         │ Is there strong      │
                         │ seasonality?         │
                         └──────────────────────┘
                                    │
                   ┌────────────────┴────────────────┐
                   │ YES                             │ NO
                   ▼                                 ▼
```

```
        ┌─────────────────┐         ┌─────────────────┐
        │ SARIMA/Prophet  │         │ Is there        │
        │ (seasonal=True) │         │ autocorrelation?│
        └─────────────────┘         └─────────────────┘
                 │                           │
                 │                  ┌────────┴────────┐
                 │                  │ YES             │ NO
                 │                  ▼                 ▼
                 │          ┌──────────────┐  ┌──────────────┐
                 │          │ ARIMA with   │  │ Naïve/Average│
                 │          │ p,q from     │  │ Baseline     │
                 │          │ ACF/PACF     │  │              │
                 │          └──────────────┘  └──────────────┘
                 ▼
        ┌─────────────────┐
        │ High Residuals? │
        │ (>25%)          │
        └─────────────────┘
               │
        ┌──────┴──────┐
        │ YES         │ NO
        ▼             ▼
      Expect        Expect
      ~Naïve        Good
      accuracy      accuracy
```

7.2 Category-Specific Model Selection

N02BE (Paracetamol) - Best Case

Analysis Result	Implication
Strong annual seasonality	Use SARIMA/Prophet
Low approximate entropy	High predictability
Low residuals (~15%)	Trend+season explain most variance
Stationary by ADF	d=0

Best Models: SARIMA(0,0,0)(P,D,Q,52), Prophet with yearly seasonality

---

R03 (Respiratory) - Good Case with Outliers

Analysis Result	Implication
Strong spring seasonality	Use SARIMA/Prophet
High outliers	Prophet robust to outliers
Moderate residuals (~30%)	Some unexplained variance

Best Models: Prophet (handles outliers better than ARIMA)

M01AE (Ibuprofen) - Challenging Case

Analysis Result	Implication
Weak/no seasonality	Non-seasonal models
Highest approximate entropy	Very unpredictable
High residuals	Random component dominates

Best Models: Naïve baseline likely competitive with complex models

---

N05C (Sedatives) - Most Difficult

Analysis Result	Implication
No seasonality	Non-seasonal models
High randomness	Prescription-driven
Highest residuals (~35%)	Mostly noise

Best Models: Average baseline, complex models won't help

---

8. Why Multiple Models Were Used

8.1 Research Question

> "Can modern time-series forecasting methods outperform Naïve baselines for small-scale pharmaceutical sales prediction?"

To answer this, we need to **benchmark** multiple approaches.

8.2 Model Comparison Strategy

Model Type	Models Used	Purpose
Baselines	Naïve, Seasonal Naïve, Average	Lower bound benchmark
Statistical	ARIMA, SARIMA, Auto-ARIMA	Traditional approaches
Modern ML	Prophet	Facebook's approach
Deep Learning	LSTM variants	Neural network approach

8.3 Why Each Model Type?

Naïve / Seasonal Naïve

Model	Formula	Use Case
Naïve	$\hat{y}t = y{t-1}$	Random walk data
Seasonal Naïve	$\hat{y}t = y{t-52}$	Seasonal data

Purpose: If complex models can't beat these, the data is inherently unpredictable.

ARIMA / SARIMA

Model	Captures	Use Case
ARIMA(p,d,q)	Trend, autocorrelation	Non-seasonal series
SARIMA(p,d,q)(P,D,Q,m)	+ Seasonality	Seasonal series

Purpose: Standard statistical approach for time series.

Prophet

Strength	How
Handles holidays	Built-in holiday effects
Robust to outliers	Heavy-tailed uncertainty
Automatic seasonality	Fourier series decomposition
Business-friendly	Interpretable components

Purpose: Modern alternative to ARIMA, especially good for seasonal data with outliers.

LSTM

Architecture	Purpose
Vanilla LSTM	Baseline neural network
Stacked LSTM	Capture complex patterns
Bidirectional LSTM	Use future context

Purpose: Test if deep learning adds value for small-scale data.

8.4 Expected Findings

Based on preprocessing analysis:

Category	Expected Best Performer
N02BE, R03, R06	SARIMA/Prophet (seasonal patterns)
M01AB, M01AE	Naïve (high randomness)
N05B, N05C	Average (no patterns)

Category	Expected Best Performer
N02BA	ARIMA (some autocorrelation)

---

9. Preprocessing-Model Synergy

9.1 Complete Preprocessing-to-Model Mapping

Preprocessing Step	Analysis Result	Model Configuration
Aggregation	302 weekly points	Enables all models
Seasonality analysis	N02BE, R03, R06 seasonal	SARIMA(m=52), Prophet(yearly)
ADF test	N02BA non-stationary	ARIMA with d=1
KPSS test	Trend in N02BE, R03, R06	Consider detrending
ACF/PACF	Varies by category	Category-specific (p, q)
Entropy	High for M01AB, M01AE	Expect Naïve competitive
MinMax scaling	For LSTM	Neural network ready
Sequence splitting	5-step lookback	LSTM input format

9.2 How Preprocessing Enables Model Performance

```
┌─────────────────────────────────────────────────────────────────┐
│                    PREPROCESSING BENEFITS                       │
├─────────────────────────────────────────────────────────────────┤
│ 1. Weekly aggregation                                           │
│    → Stable patterns emerge from noise                          │
│    → 302 points sufficient for statistical models               │
│                                                                 │
│ 2. Seasonality detection                                        │
│    → Correctly configure SARIMA seasonal period (m=52)          │
│    → Add yearly seasonality to Prophet for N02BE, R03, R06      │
│                                                                 │
│ 3. Stationarity testing                                         │
│    → Set d=0 for most categories (already stationary)           │
│    → Set d=1 for N02BA (needs differencing)                     │
│                                                                 │
│ 4. ACF/PACF analysis                                            │
│    → Optimal (p, q) for each category                           │
│    → Avoids overfitting from too many parameters                │
│                                                                 │
│ 5. Entropy analysis                                             │
│    → Set realistic expectations                                 │
│    → M01AE high entropy → don't expect miracles                 │
│                                                                 │
```

```
│ 6. MinMax scaling + sequence splitting                     │
│    → LSTM can train without gradient problems              │
│    → Supervised format enables neural network training     │
└────────────────────────────────────────────────────────────┘
```

9.3 Category-Specific Preprocessing-Model Configuration

Seasonal Categories (N02BE, R03, R06)

```
Preprocessing:
  ✓ Weekly aggregation
  ✓ Detect annual seasonality
  ✓ Check stationarity (mostly stationary)
  ✓ MinMax scale for LSTM
        │
        ▼
Model Configuration:
  • SARIMA(0,0,0)(1,1,1,52) - seasonal differencing
  • Prophet + yearly seasonality (fourier_order=13)
  • LSTM with 5-step lookback
```

Non-Seasonal High-Entropy Categories (M01AB, M01AE)

```
Preprocessing:
  ✓ Weekly aggregation
  ✗ No clear seasonality
  ✗ High approximate entropy
  ✓ Stationary
        │
        ▼
Model Configuration:
  • ARIMA(2,0,0) - simple AR model
  • Prophet without yearly seasonality
  • Naïve likely competitive (baseline)
```

---

10. Conclusion

10.1 Summary of Preprocessing-Model Logic

Preprocessing Phase	Technique	Model Impact
Aggregation	600K → 302 weekly	Enables statistical modeling

Preprocessing Phase	Technique	Model Impact
Seasonality	Box plots + decomposition	Seasonal: SARIMA/Prophet; Non-seasonal: ARIMA
Stationarity	ADF + KPSS tests	Determines differencing (d parameter)
Autocorrelation	ACF/PACF plots	Determines AR/MA orders (p, q)
Predictability	Approximate entropy	Sets accuracy expectations
Scaling	MinMax normalization	Required for LSTM
Transformation	Sequence-to-supervised	Enables LSTM training

10.2 Model Selection Decision Tree

```
Time Series Data
       │
       ├── Seasonal? ──────────────────┐
       │   YES                          │ NO
       │   ▼                            ▼
       │  ┌────────────┐           ┌────────────┐
       │  │ SARIMA     │           │ Stationary? │
       │  │ Prophet    │           └────────────┘
       │  └────────────┘                 │
       │                            ┌────┴─────┐
       │                            │YES       │NO
       │                            ▼          ▼
       │                       ARIMA(p,0,q)  ARIMA(p,1,q)
       │                            │
       │                            ▼
       │                       High Entropy?
       │                            │
       │                       ┌────┴────┐
       │                       │YES      │NO
       │                       ▼         ▼
       │                   Naïve     ARIMA/LSTM
       │                   likely    may help
       │                   best
```

10.3 Key Insights

1. **Preprocessing reveals data structure:** Without seasonality/stationarity analysis, model selection is guesswork

2. **Different drugs need different models:** One-size-fits-all doesn't work

    - Seasonal drugs (N02BE, R03, R06) → SARIMA/Prophet
    - Random drugs (M01AE) → Naïve baseline

3. **Simple often beats complex:** For high-entropy data, Naïve outperforms LSTM

4. **Preprocessing is 80% of success:** Correct aggregation, stationarity handling, and parameter selection matter more than model choice

10.4 The Complete Value Chain

```
Raw POS Data (600K transactions)
        │
        ▼
[Weekly Aggregation] → 302 observations
        │
        ▼
[Seasonality Analysis] → Seasonal vs Non-seasonal classification
        │
        ▼
[Stationarity Tests] → Differencing requirements (d parameter)
        │
        ▼
[ACF/PACF] → ARIMA (p, q) parameters
        │
        ▼
[Entropy Analysis] → Predictability expectations
        │
        ▼
[Model-Specific Preprocessing] → Scaling, sequence formatting
        │
        ▼
[Multiple Model Training] → ARIMA, Prophet, LSTM
        │
        ▼
[Performance Comparison] → Best model per category
        │
        ▼
Category-Specific Forecasting Recommendations
```

📊 This document explains the complete rationale from preprocessing to model selection

Understanding why each technique is used ensures appropriate model selection for different time series characteristics