Modeling Promotional Pasta Sales with Hierarchical Poisson Autoregression

Group 2

Contents

1 Introduction

We analyze pasta sales data from an Italian grocery store, consisting of 116 items across four brands, observed daily from 2014 to 2018. Our goal is to model item-level sales with dependence on past observations, promotional status, and latent brand effects.

We compare likelihood-based models (EM, Newton-Raphson), Bayesian inference (MCMC), and machine learning benchmarks (Random Forests, Hidden Markov Models), using predictive accuracy as the main evaluation metric.

1.1 Background

1.2 Goals

2 Data Description

```r
data("data_set_tidy")
glimpse(data_set_tidy)
```

```
## Rows: 212,164
## Columns: 5
## $ DATE  <date> 2014-01-02, 2014-01-02, 2014-01-02, 2014-01-02, 2014-01-02, 201~
## $ brand <chr> "B1", "B1", "B1", "B1", "B1", "B1", "B1", "B1", "B1", "B1", "B1"~
## $ item  <chr> "1", "2", "3", "4", "5", "6", "7", "8", "9", "10", "11", "12", "~
## $ QTY   <dbl> 7, 3, 0, 2, 3, 1, 0, 4, 0, 0, 0, 0, 7, 2, 5, 2, 0, 7, 1, 0, 0, 4~
## $ PROMO <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
```

Each row corresponds to daily sales for a specific item, along with brand and promotion indicators.

3 Model Derivation

3.1 Extended Poisson Autoregressive Model

Let the sales outcome for item i at time t be denoted by y_{it}, where:

$$i \in \{1, \dots, n\}, \quad t \in \{1, \dots, T\}, \quad g_i \in \{1, \dots, B\}$$

Here, g_i denotes the brand (group) to which item i belongs.

We define the model as:

$$y_{it} \mid m_{it} \sim \text{Poisson}(m_{it})$$

$$m_{it} = \sum_{l=1}^{q} \beta_{i,l} y_{i,t-l} + \left(1 - \sum_{l=1}^{q} \beta_{i,l}\right) \exp\left(\mu_{g_i} + \eta_i + \mathbf{x}_{it}^\top \boldsymbol{\gamma}_i + f_t\right)$$

Model components:

- $\beta_{i,l}$: autoregressive coefficients for item i
- μ_{g_i}: brand-level intercept
- η_i: item-specific deviation
- $\boldsymbol{\gamma}_i$: covariate effects (e.g., promotion)
- f_t: global time-specific effect (e.g., seasonality)

3.2 Likelihood Function

Let $\boldsymbol{\theta}$ denote the full set of model parameters. The full likelihood across all items and times is:

$$\mathcal{L}(\boldsymbol{\theta}) = \prod_{i=1}^{n} \prod_{t=q+1}^{T} \frac{m_{it}^{y_{it}} e^{-m_{it}}}{y_{it}!}$$

The corresponding log-likelihood, used for optimization and posterior inference, is:

$$\log \mathcal{L}(\boldsymbol{\theta}) = \sum_{i=1}^{n} \sum_{t=q+1}^{T} \left[y_{it} \log(m_{it}) - m_{it} - \log(y_{it}!)\right]$$

Let $\boldsymbol{\theta}$ denote the full set of model parameters. The complete-data likelihood is:

$$\mathcal{L}(\boldsymbol{\theta}) = \prod_{i=1}^{n} \prod_{t=q+1}^{T} \frac{m_{it}^{y_{it}} e^{-m_{it}}}{y_{it}!}$$

The log-likelihood is:

$$\log \mathcal{L}(\boldsymbol{\theta}) = \sum_{i=1}^{n} \sum_{t=q+1}^{T} \left[y_{it} \log(m_{it}) - m_{it} - \log(y_{it}!) \right]$$

4 Estimation Methods

4.1 EM Algorithm

- E-step: Compute expected values of latent variables (e.g., item intercepts η_i)
- M-step: Maximize the expected log-likelihood with respect to model parameters

4.2 Newton-Raphson

- Use gradients and Hessians of the log-likelihood to iteratively update parameters

4.3 Bayesian Inference (MCMC)

- Fit the full hierarchical model in Stan
- Use priors on all parameters, including hierarchical priors:

$$\gamma i \sim \mathcal{N}(\boldsymbol{\mu} g_i, \rho_{g_i} \mathbf{I}), \quad \mu_{g_i} \sim \mathcal{N}(0, \sigma_\mu^2), \quad \eta_i \sim \mathcal{N}(0, \sigma_\eta^2)$$

5 Model Evaluation and Predictive Performance

We evaluate models using holdout-based prediction: the final 10–20% of each item's time series is reserved for testing.

```
# Example: results <- evaluate_holdout(...)
# knitr::kable(results)
```

** Evaluation Metrics ** - Root Mean Squared Error (RMSE) - Mean Absolute Error (MAE) - Log Predictive Density (for Bayesian models) - Posterior predictive interval coverage (for MCMC)

6 Machine Learning Benchmarks

6.1 Random Forest

```
# rf_fit <- fit_random_forest(...)
```

6.2 Hidden Markov Model

```
# hmm_fit <- fit_hmm(...)
```

7 Summary and Discussion

- Which methods give the most accurate predictions for sparse, autocorrelated data?
- What is the trade-off between hierarchical shrinkage and item-specific flexibility?
- Are machine learning models (e.g., random forests) better suited for forecasting than structured generative models?