Kola Market Inventory Recommendation Prototype

Mbeleck Berle

August 7, 2025

Abstract

This document describes the design, implementation, and deployment of an inventory recommendation system for two Ghanaian towns—Accra and Tamale. We combine simulated demographic and climate data, Google Trends analysis, and a Random Forest regression model exposed via a FastAPI service. We also outline a WhatsApp integration suggestion using Azure Bot Service.

Contents

1 Introduction

Data-driven inventory planning can reduce stockouts and waste. We focus on Accra (urban/coastal) and Tamale (savannah) to illustrate:

- Data sources and assumptions
- Google Trends demand analysis
- Machine learning–based recommendations
- REST API deployment with FastAPI
- Suggested WhatsApp bot integration

2 Data Sources & Assumptions

- **Demographics:** World Bank population data for Greater Accra and Northern regions.
- **Climate:** World Bank historical climate portal for rainfall and temperature.
- **Product Demand:** Google Trends search interest for key products over the past 12 months.
- **Assumptions:**
  - Trends scores (0–100) represent relative weekly demand.
  - Synthetic region/quarter features to diversify training data.
  - Product list: rice, maize, canned fish, cooking oil, bottled water.

3 Data Acquisition

3.1 Demographic & Climate Data

Load or simulate CSVs into `pandas`:

```
df_pop = pd.DataFrame({Region: [...], Population: [...]})
df_climate = pd.DataFrame({
  Region: [...],
  Avg_Annual_Rainfall_mm: [...],
  Avg_Temp_C: [...]
})
```

3.2 Google Trends Data

Use `pytrends` to fetch weekly interest:

```
pytrends.build_payload(products, timeframe="today 12-m", geo="GH")
trends = pytrends.interest_over_time()
```

4 Data Integration & Product Trends

1. Map World Bank regions to towns.
2. Compute 12-month mean trend score per product.
3. Create `product_trends` DataFrame.

5 Visualization

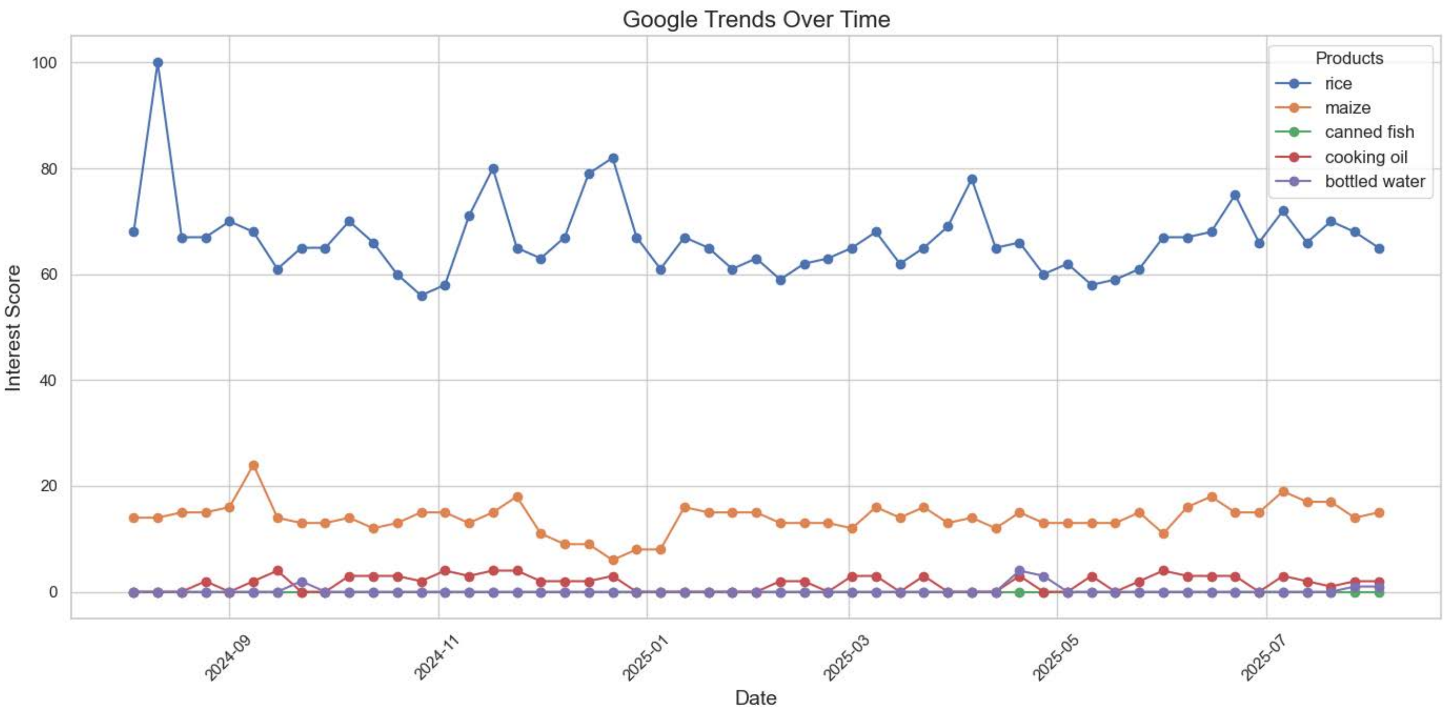

Figure 1: Weekly Google Trends scores.

6 Machine Learning

6.1 Preparation

1. Melt trends to long format.
2. Add `region`, `quarter` features.
3. Label-encode categories.

6.2 Model

```
rf = RandomForestRegressor(n_estimators=100, random_state=42)
rf.fit(X_train, y_train)
r2 = rf.score(X_test, y_test)
```

Test R^2: `0.XX`

7 FastAPI Service

Expose recommendations via a REST API:

7.1 Setup

```
pip install fastapi uvicorn joblib
```

7.2 app/main.py

```
from fastapi import FastAPI
import joblib, pandas as pd

app = FastAPI()
rf = joblib.load("models/rf_model.joblib")
le_r = joblib.load("models/le_region.joblib")
le_q = joblib.load("models/le_quarter.joblib")
le_p = joblib.load("models/le_product.joblib")

@app.get("/recommend")
def recommend(town: str, quarter: str, top_n: int = 5):
    te = le_r.transform([town.title()])[0]
    qe = le_q.transform([quarter.upper()])[0]
    products = le_p.classes_
    pe = le_p.transform(products)
    df = pd.DataFrame({
      "region_enc": te,
      "quarter_enc": qe,
      "product_enc": pe
    })
    preds = rf.predict(df)
    top = sorted(zip(products, preds), key=lambda x: -x[1])[:top_n]
    return [{"product": p, "score": float(s)} for p,s in top]
```

7.3 Run

```
uvicorn app.main:app --reload --port 8000
```

8 WhatsApp Integration Suggestion

Use Azure Bot Service + Azure Communication Services:

- Deploy FastAPI on Azure App Service.
- Create Azure Bot, configure WhatsApp channel.
- Bot calls `/recommend` endpoint, returns top products.

9 Limitations & Future Work

- **Limitations:** Normalized scores sales volumes; simulated features.
- **Future:**
  - Real-time sales/weather integration.
  - Time-series forecasting (ARIMA, LSTM).
  - Interactive web dashboard.