ISUPPLY COMPETITION

product matching model

By Nada Ahmed

COMPETITION OVERVIEW

- Objective: Build a product matching model for a pharmaceutical marketplace.
- Challenges:
  - Limited data coverage for all products.
  - Different name formats in datasets.
  - Matching both Arabic and English product names.
  - Handling spelling variations and abbreviations.

MATCHING APPROACH

- Goal: Match SKU codes and correctly formatted names from the master file with user-provided product names.
- Process:

1. Extract & Preprocess Data: Normalize product names, handle variations.
2. Use Machine Learning Model: Apply the trained model to predict the best-matching SKU.
3. Assign Correct SKU: Return SKU with a similarity score and confidence level.

ACCEPTANCE CRITERIA & HOW THEY ARE MET

- Similarity Score: Between 0 and 1.
  - Achieved using TF-IDF vectorization and KNN-based distance calculations to quantify similarity.
- Robustness: Handles spelling mistakes, OCR errors (e.g., "Ibuprofn" → "Ibuprofen").
  - Implemented text normalization, number-to-text conversion, and typo tolerance.
- Accuracy: At least 90% on validation data.
  - Model achieved 97% accuracy on test data and 94.3% mean accuracy in cross-validation.

ACCEPTANCE CRITERIA & HOW THEY ARE MET

- Efficiency:
  - Optimized for CPU execution.
  - Matching within ⩽500ms per comparison.
  - Used precomputed TF-IDF vectors for fast lookups.
- Confidence Levels: High, Medium, or Low based on score.
  - Defined confidence thresholds based on predict_proba and similarity score, ensuring uncertain matches are flagged for review.

CHALLENGES FACED

- Limited Data: Not enough examples for all products.
- Data Formatting Issues: Inconsistent product naming styles.
- Language Barrier: Supporting both Arabic and English names.
- Spelling Variations: Handling abbreviations, typos, and OCR errors.

SOLUTIONS & APPROACHES TRIED

1. Traditional Programming (Rule-Based Matching)
   - Problem:
     - Too slow for large datasets.
     - Couldn't cover all variations in product names.
2. Machine Learning Approaches
   - Transfer Learning (Pretrained Models & Embeddings)
     - Issue: Overfitting due to the small dataset.
   - Random Forest & Similar Models
     - Issue: Required high computational power, but only CPU usage was allowed.
   - Word2Vec & Bag of Words
     - Issue: Did not guarantee high accuracy in matching.

OPTIMAL SOLUTION - KNN WITH TF-IDF

Why KNN?

- Finds the nearest match by minimizing distance.
- Naturally calculates similarity between product names.
- Handles new data easily without requiring retraining.

Challenge 1: Setting the Confidence Threshold

- Solution: Use minimum of predict_proba and similarity score
  - Ensures wrong predictions are labeled Low or Medium Confidence.
  - If High Confidence, the prediction is highly reliable.

ADDITIONAL CHALLENGES & REFINEMENTS

- Issue: Similar medicines with small differences (e.g., Item 1 vs. Item 2).
- Solution: Convert numbers into text (e.g., 10 → ten mg) to ensure variations are captured.
- Keeping Price as Numeric Feature: Since prices vary, they were not weighted heavily in training.

ADVANTAGES & LIMITATIONS OF THIS APPROACH

Advantages:

- Training does not require much time or computational power.
- Naturally adapts to different product variations.

Limitations:

- Confidence threshold may need periodic tuning.

PERFORMANCE & OPTIMIZATION

- Execution Speed: Processes 2000 items in ~31 seconds.
- Confidence Check: 8.8% of predictions require manual review.
- Optimizations:
  - Precompute TF-IDF vectors.
  - Use efficient nearest neighbor search for fast matching.

GITHUB

https://github.com/nadaahx/ISupply_competition

THANK YOU