ML Challenge 2025 — Smart Product Pricing (Team Apex)

Submission Date: 13/10/2025

Name	GitHub	LinkedIn	Department @ IIT Patna
S Akash	@Akasxh	LinkedIn	EEE
Anirudh D Bhat	@RudhaTR	LinkedIn	CSE
Akash S	@akash1764591	LinkedIn	EEE
Ammar Ahmad	@ammarrahmad	LinkedIn	AI & DS

1 Executive Summary

We tackle **product price prediction from semi-structured catalog text** as a **text regression** problem. After building a **hybrid data preparation pipeline** (Regex for fast parsing + a targeted small LLM for imputation), we fine-tune **DeBERTa-v3-large** with a lightweight regression head.

- **Codes and Drive** and **Github**
- **Best validation SMAPE:** 42.22% on a 7,500-sample hold-out split
- **Other metrics:** MAE: $9.44, R^2: 0.29
- **Training size:** 67,499 samples; **epochs:** 10

Key insight: **pragmatic feature engineering**—extracting value/unit/name via Regex and using an LLM *only where needed*—yields cleaner inputs at a fraction of the compute of fully LLM-driven structuring.

2 Problem & Data

2.1 Task

Predict a continuous **price** from a textual **catalog_content** field (semi-structured lines with optional bullets). Images were available but largely redundant with text for this challenge.

2.2 Dataset Fields

- `serial_id`, `price` (target), `image_link`, `catalog_content` (primary feature)

2.3 EDA Highlights

- Right-skewed target → apply `log1p(price)` during training
- Semi-structured text with pattern:
  1. Item name / short description
  2. Numeric value/quantity (e.g., 12, 16.9)
  3. Unit (e.g., oz, count)
  4. Longer description / bullets (optional)
- Image information mostly duplicated in text → **text-only** approach favored

3 Methodology

3.1 Data Preparation (Hybrid Regex + Targeted LLM)

1. **Regex extraction** of **item name**, **value**, **unit**, plus remaining text as **description**.
2. **Targeted LLM imputation** for rows where Regex fails.
3. Sensible defaults for residual nulls (e.g., `unit="count"`, `description="no description"`).
4. Structured prompt string fed to the model:

```
Category: [category] [SEP] description: [description] [SEP] Amount: [value] [unit
```

5. Apply `log1p` transform to `price` for training; use `expm1` for inference.

3.2 Model Architecture

- Backbone: `microsoft/deberta-v3-large`
- Head: MLP regression head on CLS/pooled representation
- Loss: Huber

3.3 Training Setup (Key Hyperparameters)

- Max sequence length: 160
- Dropout: 0.2
- Encoder LR: 2e-5; Head LR: 1e-3
- Batch size: 16
- Epochs: 10

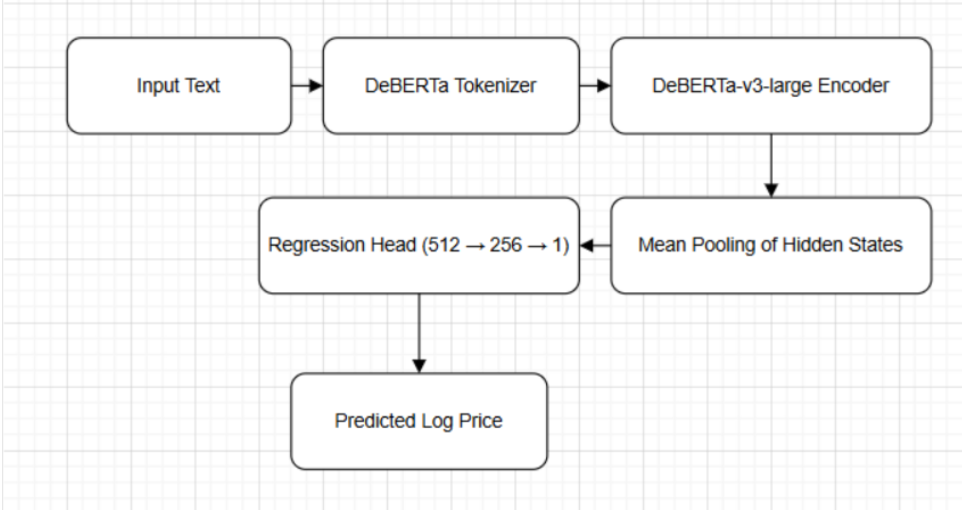

Figure 1: Data preparation and model pipeline (replace with your image).

4 Results

SMAPE is the primary metric:

$$\text{SMAPE} = \frac{1}{n} \sum \frac{|\text{predicted} - \text{actual}|}{(|\text{actual}| + |\text{predicted}|)/2}$$

4.1 Final Hold-out (7,500 samples)

- SMAPE: 42.22%
- MAE: $9.44
- R^2: 0.29
- Training epochs: 10; MLP head layers: 512→256; Train Loss (Huber) @ epoch 10: 0.0276

4.2 Selected Experiment Snapshots (Ablations)

A) Head depth/epochs vs. performance:

Epochs	MLP Head (dims)	MAE ($)	R^2	SMAPE (%)
5	512→256	9.53	0.28	42.69
5	768→384	9.63	0.29	42.86
7	512→256	9.50	0.28	42.68
10	**512→256**	**9.44**	**0.29**	**42.22**

B) Model family comparisons (same pipeline):

Approach	Validation SMAPE (%)
DeBERTa-v3-large + head (final)	**42.22**
DeBERTa-v3-base + head	43.59
DistilBERT + head (baseline)	45.00
Sentence Embeddings + Neural Net	49.00
Sentence Embeddings + XGBoost	52.00
BERT Encodings + XGBoost	50.00
LLM SFT (phi-3-mini-instruct, 1 epoch)	48.00 (slow)
IFT (Qwen-7B / phi-3-mini-instruct)	Abandoned (compute)

5 Discussion & Lessons

- Data ¿ Model Size (alone): Regex→LLM hybrid delivered cleaner inputs without full LLM structuring.
- Target transformation matters: `log1p` stabilized optimization for heavy-tailed prices.
- Vision deprioritized intentionally: text captured most variance; images not critical under compute constraints.

6 Reproducibility (High-Level)

1. Prepare data: parse `name/value/unit/description`, apply LLM fills, compute `log1p(price)`.
2. Train: fine-tune `microsoft/deberta-v3-large` with Huber loss.
3. Evaluate: compute SMAPE/MAE/R^2 on hold-out.
4. Infer: predict on new `catalog_content`, inverse-transform with `expm1`.

Code & Scripts: Google Drive

7 Appendix

7.1 Additional Experiment Notes (excerpts)

- Epoch 7/7: Train Huber Loss ≈ 0.0470 → Test SMAPE 42.68% (MAE $9.50, R^2 0.28)
- Epoch 10/10: Train Huber Loss 0.0276 → Best Test SMAPE 42.22% (MAE $9.44, R^2 0.29)

7.2 Potential Next Steps

- Lightweight category detection and "premium" heuristics to de-skew residuals
- Targeted vision augmentation only when Regex+LLM confidence is low

- Quantization / LoRA for faster ablations on larger backbones
- Calibrated price intervals (e.g., conformal) for actionable uncertainty

Contact: See team table for GitHub / LinkedIn profiles.