Smart Tariff & Fee Assistant (STAFF)

CMPT-310 Project Proposal — Group 26

Group Members:

- Kazi Boni Amin (kba109@sfu.ca) — 301591325

- Ertugrul Yurtseven (eya38@sfu.ca) — 301588905

- Brayden Yee (@sfu.ca) — 301559654

- Guneet Bajaj (gba74@sfu.ca) — 301637797

1. Project Idea

Problem

When Canadians ship packages internationally, they must select the correct HS (Harmonized System) code for customs. Product descriptions are often informal (e.g., "Sony-style wireless headphones w/ battery"), making it difficult to find the correct code. Wrong classifications can cause delays or extra fees. Some products also contain restricted items (e.g., lithium batteries) that sellers may not recognize.

Proposed Solution

We will build an AI system that takes a casual product description and predicts the most likely HS code category. It will also flag restricted items and, as an optional extension, estimate shipping fees.

- Input: Product description text (and optionally weight/value)
- Output: Top 3 HS code predictions with confidence scores and any restriction warnings

AI Techniques to Use

- Text Classification into 10–15 HS categories
- Decision Tree (ID3) built from scratch
- K-Nearest Neighbors (KNN) built from scratch
- BERT embeddings for text features
- Rule-based keyword detection for restricted items

System Diagram

User Input → BERT Embedding → Classifier (Tree/KNN) → HS Code Prediction
↓
Rule Checker → ⚠ Flags

2. Tools and Resources

- Languages/Libraries: Python 3, NumPy, Pandas, Matplotlib
- Text Embeddings: Hugging Face Transformers (BERT)
- Evaluation: scikit-learn (metrics only)
- UI (Optional): Streamlit

Dataset: We will build a synthetic dataset (500–1000 entries) using public product lists (e.g., Datablist), manually labeling them into 10–15 categories. Both clean and noisy (with typos/abbreviations) versions will be prepared to test robustness.

3. Project Plan / Timeline

Milestone 1 (Oct 19)

- Build dataset (clean + noisy)
- Implement Decision Tree classifier
- Create basic CLI for predictions
- Run baseline evaluation (accuracy + confusion matrix)

Milestone 2 (Nov 16)

- Integrate BERT embeddings
- Implement KNN classifier
- Add restricted item keyword detection
- Compare Decision Tree vs. KNN and BERT vs. TF-IDF
- Visualize evaluation metrics
- (Optional) Add fee estimation or simple UI

4. Minimal Viable System

- Input: Product description text
- Output: Top-1 HS code prediction + restricted item flag
- One Decision Tree classifier (from scratch)
- Basic CLI for testing
- Evaluation plots showing accuracy

Even this simple system is useful: it gives sellers quick HS code suggestions and flags common shipping issues, reducing manual lookup time.

5. Why It Uses AI?

HS code classification is a real text classification problem. Manual rules are impractical, as users describe products in informal ways. Our system learns from examples using Decision Trees and KNN, which are transparent and suitable for a from-scratch implementation. This makes the project both realistic and educational.

Appendix: Project File Structure

```
STAFF/
│
├── data/
│   ├── raw/                  # Original CSV files
│   ├── processed/            # Clean & noisy labeled datasets
│   └── README.md
│
├── src/
│   ├── decision_tree.py      # ID3 implementation
│   ├── knn.py                # KNN implementation
│   ├── text_processing.py    # BERT + TF-IDF processing
│   ├── rule_based.py         # Keyword detection
│   └── main.py               # CLI entry point
│
├── notebooks/
│   ├── EDA.ipynb
│   ├── baseline_model.ipynb
│   └── evaluation.ipynb
│
├── reports/
│   └── proposal/
│       └── STAFF_Project_Proposal.docx
│
├── requirements.txt
├── README.md
└── .gitignore
```