Veridion Challenge - Data Science track

Horatiu Andrei Palaghiu

May 10, 2025

Abstract

This project develops a scalable **company classifier** for an insurance taxonomy using unstructured data (descriptions, tags, and sector labels). Without ground-truth labels, the solution emphasizes **heuristic validation** over traditional metrics, leveraging embeddings and zero-shot learning while prioritizing **real-world relevance**. Deliverables include an annotated dataset, modular code, and a discussion of trade-offs between precision and scalability.

Contents

1 Introduction

2 Data preparation & preprocessing

2.1 Dataset

The initial data cleaning was performed in the Veridion_Challenge_preprocessing.ipynb file. While the dataset was surprisingly clean, several standard preprocessing steps were necessary.

2.1.1 Text Normalization

The clean_text() function performs essential text normalization:

- Converts input strings to lowercase for consistency
- Compresses multiple whitespaces into single spaces
- Removes leading/trailing whitespace
- Safely handles non-string inputs

Applied dataframe-wide using applymap(), this function ensures uniform text formatting across all columns.

Moreover, I sepparately got rid of stop words and added lemmatization. I used NLTK's English stopword list to filter out common function words, and applied WordNetLemmatizer to reduce words to their base forms. These computationally intensive steps may be omitted for large-scale datasets where preserving the full vocabulary is preferred over processing time.

2.1.2 Duplicate Handling

The dataset contained semantic duplicates where different companies shared identical descriptions while having distinct business_tags. These were identified by: (1) detecting duplicate descriptions while excluding missing values, and (2) manually inspecting the resulting subset of potential duplicates. The deduplication strategy prioritized data richness by preserving entries with the most comprehensive business_tags.

Notably, three cases featured identical generic descriptions (i.e. "*The company is a business analysis and English translation service provider*") that actually represented distinct businesses when examined in the context of their sector, category, and niche classifications. These were retained as valid unique entries, demonstrating the importance of holistic record examination beyond simple text matching. The manual verification process ensured no genuine businesses were erroneously removed while effectively consolidating true duplicates.

2.1.3 Missing Values

The dataset's missing values were systematically addressed through a multi-stage process. Initial inspection revealed no records with three or more missing values, while 27 entries contained exactly two missing values - consistently in both sector and category columns. For single missing values, analysis showed these occurred exclusively in the description field, which were filled with empty strings using fillna() as meaningful imputation wasn't feasible.

For the sector and category gaps, I employed FastText classification models optimized with Optuna hyperparameter tuning. This automated approach successfully filled:

- 25/27 missing sector values (92.6% completion)
- 12/27 missing category values (44.4% completion)

The remaining gaps (2 sector and 15 category entries) were manually imputed based on business descriptions and tags.
This hybrid approach - combining automated classification with manual review - proved fast and effective for this static dataset, though would require different handling for streaming big data where new categories may emerge.

2.2 Labels

The label standardization process was performed in the Veridion_challenge_labels_preprocessing.ipynb file, focusing on the insurance taxonomy dataset. The workflow consisted of:

2.2.1 Label Cleaning

The clean_labels() function processed the taxonomy labels through:

- Case normalization (conversion to lowercase)
- Special character removal (preserving alphanumerics, spaces, hyphens, and slashes)
- Whitespace standardization (multiple spaces → single space)
- Leading/trailing whitespace trimming
- Tokenization using NLTK's word_tokenize()

2.2.2 Similarity Detection

Potential duplicates were identified using cosine similarity comparison (threshold = 0.65)

The process outputted a cleaned label list and identified semantic duplicates that I removed.

2.2.3 Label Description Generation with LLMs

To enhance the performance of our zero-shot classification approach, I employed both ChatGPT and DeepSeek to generate short, but comprehensive descriptions for each insurance taxonomy label. This auxiliary step proved valuable when I observed that zero-shot models could be

significantly improved with richer context for each category. I then created a column label_description to save them.

2.3 Export

The cleaned dataset was saved as clean_data.csv, while the labels as clean_labels+description.csv

3 Modelling

With the cleaned datasets loaded in Main_Veridion_Challenge.ipynb, I implemented a zero-shot approach using SentenceTransformer embeddings and BART classifier (after testing but rejecting ROBERTA for being too slow), followed by few-shot learning strategies for refinement.

3.1 Zero-Shot with SentenceTransformer + BART

3.1.1 Direct company-centered approach

I first implemented a hybrid classifier combining SentenceTransformer embeddings with BART-large-MNLI predictions. The model achieved strong precision (12.5% single-label predictions) but limited coverage (16.6% total labeled). The ensemble filtered predictions through:

- Semantic similarity threshold (0.3)
- BART confidence threshold (0.3)
- Weighted consensus scoring (30% ST + 70% BART)

While accurate (evidenced by manual inspection), this conservative approach proved too restrictive - especially on labels, as barely 4 of them appeared with treshold greater than 0.6.

The ConsensusInsuranceClassifier combines predictions from a Sentence Transformer and BART model, using min_agreement=1 to maximize label coverage while maintaining confidence. By accepting labels that meet the threshold in either model (rather than requiring both to agree), the classifier ensures broader applicability—particularly useful when one model is uncertain but the other provides strong signals. This approach strikes a balance: it avoids missing valid labels due to strict consensus while still filtering out low-confidence predictions through individual thresholds (st_threshold=0.7 and bart_threshold=0.6). The result is a more inclusive yet reliable output compared to methods demanding unanimous agreement.

3.1.2 Inversed taxonomy-centered approach

To improve recognition of rare or underrepresented labels, I implemented an **inverted matching approach** using SentenceTransformer and BART-large-MNLIto identify companies most aligned with each label's description.
This method "reverse-engineers" the problem — instead of predicting labels for companies, it retrieves companies that best exemplify each label. For rare labels, this helps surface **edge-case examples** that might otherwise be overlooked in standard classification. The approach leverages the same embedding model (all-MiniLM-L6-v2) used in the consensus classifier, ensuring consistency, while the similarity scores provide interpretable confidence metrics. By analyzing these matches, I could validate label definitions, refine thresholds, and even augment training data for challenging categories.

3.2 Few-shot classification

4 Conclusion & Further Remarks

4.1 What probably doesn't work for big data

4.2 Possible improvements