**Final Project Report**

Group 3: Jehan Bugli

**Overview**

This project tags congressional bills with one of [32 policy area designations](https://www.congress.gov/help/field-values/policy-area), replacing an important yet tedious task that is currently completed manually by the Congressional Research Service (CRS). This uses classical methods, with a focus on interpretability, to support practical use.

The CRS assists legislators with [policy analysis](https://www.congress.gov/crs-products), [official bill summaries](https://www.congress.gov/help/bill-summaries), and more; however, their job is rapidly becoming more difficult due to [rapidly increasing bill introductions](https://rollcall.com/2025/03/05/publishing-pileup-congressional-bills-slow-to-reach-public/) causing significant backlogs for official text releases, which in turn affects third parties reliant upon their processing efforts.

This report will:

* Discuss the data collection and processing efforts involved
* Describe the chosen modeling approach and experiment setup
* Outline the experiment results and conclusions

**Data collection and processing**

Data was sourced from the [GovInfo bulk data repository](https://www.govinfo.gov/bulkdata) which stores XML-formatted bill data for recent congressional sessions; this dates back to the 113th session (2013-2015).

To start, I downloaded bill text for each bill in the available sessions. This includes every version; for instance, if a bill received amendments between introduction and enactment, both the amended and original versions would be included. These were saved as a collection of session-specific .zip files.

I also downloaded associated data for each bill, including policy area tags and other CRS-processed items, in the same format as noted earlier.

Before further processing, I opted to “flatten” both folders, unzipping contents into two main directories.

Then, I converted each flattened folder into a tabular dataset. The folder with bill text was processed to generate a parquet file storing each file’s name and bill text.

Using BeautifulSoup’s XML parser, I searched for valid legislative body tags (e.g. ‘legis-body’, ‘preamble’) and discarded components such as the title, publishing date, sponsors, sessions, and more. While some of these items certainly may have provided some predictive value, I was concerned that such modeling may step outside of the NLP realm and muddy the results’ interpretability.

The folder with additional bill data was processed to generate a parquet file storing policy areas alongside text version names.

These text version names were used to join the two files and construct my finalized input dataset.

**Modeling approaches**

At its core, my approach uses Term Frequency – Inverse Document Frequency (TF-IDF). TF-IDF emerged out of pioneering information retrieval research as a method to quantify word importance.

This combines two key metrics:

* **Term Frequency (TF):** How frequently a term appears within a single document
* **Inverse Document Frequency (IDF):** How rare a term is across the entire document collection

The TF-IDF score multiplies these metrics to weight terms that are frequent within a given document but relatively rare across the entire collection of documents. This weighting can prove useful for many tasks, including feature extraction for classification.

I trained three different models on these extracted features:

* **LinearSVC (Support Vector Classifier):** attempts to find the "best" separating hyperplane that divides the TF-IDF feature space, distinguishing between different policy area categories. It aims to maximize the distance between the closest data points (support vectors) of different classes.
* **Logistic Regression:** predicts the probability of an instance belonging to a particular class using a logistic (sigmoid) function applied to a linear combination of the input features (TF-IDF scores).
* **Multinomial Naive Bayes:** calculates the probability of a document belonging to each policy area based on the frequency of terms within it, making a "naive" assumption that the features are independent of each other given the class.

I compared the options above using grid search cross-validation; this explores various parameter combinations across the TF-IDF vectorizer and these classifiers, comparing performance on weighted recall to return the best-performing combination.

For each combination, this search performs 3-fold cross-validation, where the data is split into 3 subsets; for each “fold”, a model is trained using 2 subsets and the remaining one serves as a validation set. This prevents overfitting, where the model is unable to generalize effectively from its training data.

My grid search included different hyperparameter configurations, such as:

* **C**: A regularization parameter penalizing misclassified examples; a larger C value prioritizes accurate classification but risks overfitting.
* **Maximum iterations**:The number of allowed iterations, altered to ensure that models are converging

I focused on macro recall as my primary metric. Recall is the ratio of true positives over all actual positive instances for a class (including both true positives and false negatives); my methodology averages recall across all classes without regard for imbalance. This decision is made out of perceived practical utility; for a policy affairs professional, catching every relevant piece of legislation (each “true positive”) for their potentially niche field is of the utmost importance, even if that arrives alongside a number of false positives.

**Results**

The linear SVC model displayed the strongest performance by far, reaching nearly 94% recall with grid search hyperparameter tuning!

This served as an encouraging sign for my line of investigation. Logistic regression followed close behind, while multinomial naïve Bayes was a distant third.

While the LinearSVC model displayed excellent results, however, performance differed across different policy areas!

These results support the notion that a classical approach can form a very robust classification mechanism for the federal legislation corpus.

The poor MultinomialNB performance suggests that this model’s assumption of independence may not hold; individual TF-IDF feature impacts on categorization cannot be viewed independently!

This makes sense intuitively; terms found jointly in a document could imply classifications that the individual terms would not, such as “pharmacy benefit manager” and “energy security”.