Capstone Project Proposal for legal document text classification

1. Project Overview

Unstructured free text is a prosperous source of information; however, extracting insights from it can be a complicated and highly time-consuming task. In recent years, thanks to many breakthroughs in the field of Natural Language Processing (NLP), which is a subfield of Machine Learning (ML), extracting actionable insight has significantly simplified. One of the essential parts of NLP is text classification which allows Data scientists to transform an unstructured corpus of text data into a structured format. During the past couple of years, novel algorithms such as Transformers[1], Universal Language Model Fine-tuning for Text Classification[2] and pre-trained language models such as Universal Sentence Encoder[3] and Pre-training of Deep Bidirectional Transformers for Language Understanding (BERT)[4] has improved the quality of text classification tasks while requiring much less tagged data to achieve state of the art results.

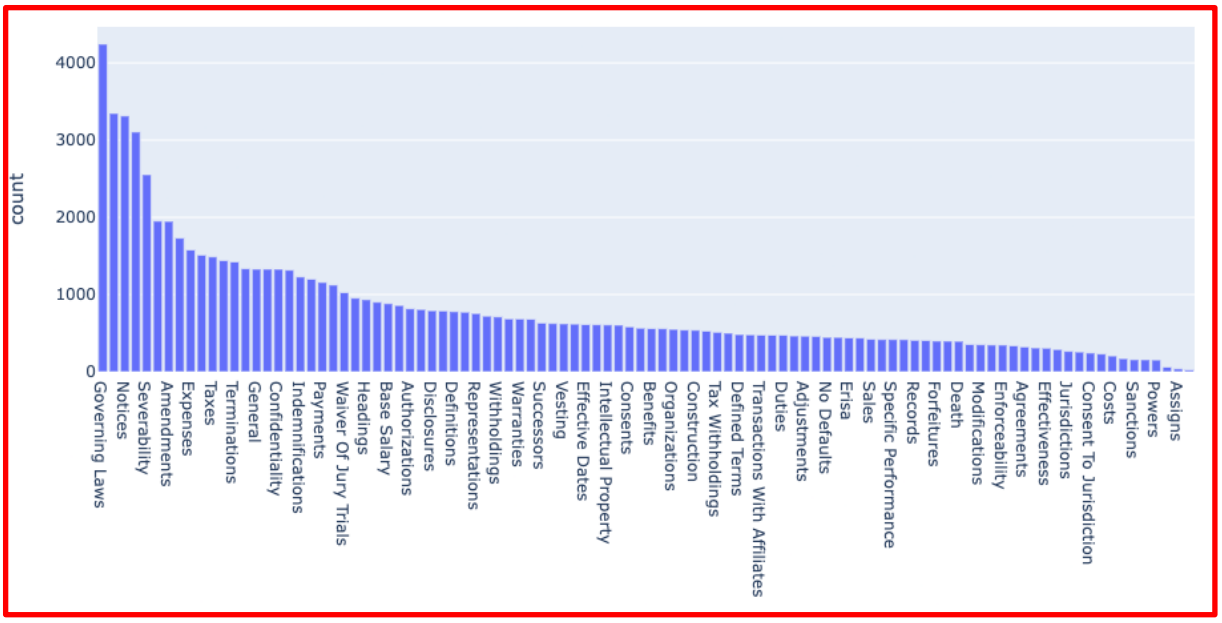
2. Problem statement

This work aims to develop a text classification pipeline that can go from raw data, which can be in an S3 bucket or any relational vs non-relational databases, to a secure API and scalable endpoint. The involving tasks are:

1. Download and process a dataset
2. Text cleaning and vectorization
3. Hyperparameter tuning and Text classification
4. Deploying the best model to a secure REST full API

3. Dataset

In this work I have used LEDGAR[1] dataset which is a multilabel corpus of legal provisions in contracts with label set of over 12’000 labels annotated in almost 100’000 provisions in over 60’000 contracts. The data was obtained from the US Securities and Exchange Commission (SEC) filings, that are publicly available. Each label represents the single main topic of the relevant contract.



As we can see from the above image, the number of available text for each "clause_type" ranges from ~4000 to ~20. In this work I will make a text classification for the top 5 classes and will add and "other" class to capture anything else. The following is the list of "clause_type" that I'm going to train a text classification model for:

1. Governing Laws
2. Counterparts
3. Notices
4. Entire Agreements
5. Severability
6. Others

4. Solution statement:

To solve this proposed problem, we will do the following:

1. Design and develop a Sagemaker pipeline based on Bring Your Own Container (BYOC) approach to:
    1. extract, transform and load (ETL) the dataset
    2. Hyperparameter tuning and model selection
    3. model deployment to live endpoint

2. Develop a CI-CD pipeline to execute the pipeline on code change on the protected branches
3. Use Universal Sentence Encoder, to transform textual data into a vector of floats
4. Train a Random forest model for the classification task

5. Evaluation metrics

The evaluation metric for this work is going to be a multi-class F1-Score. This metric is a harmonic mean of precision and recall, which is the preferred metric for unbalanced datasets like we have here.

$$\text{F-1 Score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

The above equation is used to calculated F1-Score.

6. Project design

The following is my proposal for the MLOps pipeline:

- Data processing:
  - Load raw data
  - Data transformation
  - Feature extraction
  - Save data to S3
- Model training:
  - Track the experiment with MLFlow
- Model evaluation:
  - Run model test before deployment
- Accuracy check to check whether or not the F1-Score will be higher than 80%:
  - If True: the deployment pipeline to create the REST API endpoint is triggered
  - If False: pipeline fail
- Model deployment
  - Deploying the model with SageMaker endpoint

7. Benchmark:

Due to the specific nature of this project with the suggested changes, there is no detailed study for a benchmark model performance. However, since we have six classes to identify in this dataset, if we assume that we have a random classifier to pick each of these classes at random with a normal distribution we get to the accuracy of ~ 16%.

8. References

1. Attention Is All You Need (31st Conference on Neural Information Processing Systems (NIPS 2017))(https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf)

2. Universal Language Model Fine-tuning for Text Classification [https://arxiv.org/abs/1801.06146]

3. Universal Sentence Encoder[https://arxiv.org/abs/1803.11175]

4. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding [https://arxiv.org/abs/1810.04805]

5. LEDGAR: A Large-Scale Multi-label Corpus for Text Classification of Legal Provisions in Contracts [Don Tuggener, Pius von Däniken, Thomas Peetz, Mark Cieliebak] (https://aclanthology.org/2020.lrec-1.155/)