Intermediate Presentation – Classification of Text ML-Approach

Master Internship: Approaching Information System Challenges with Natural Language Processing (IN2106, IN2130)

Patrick Ahrend

Garching, 5. December 2023

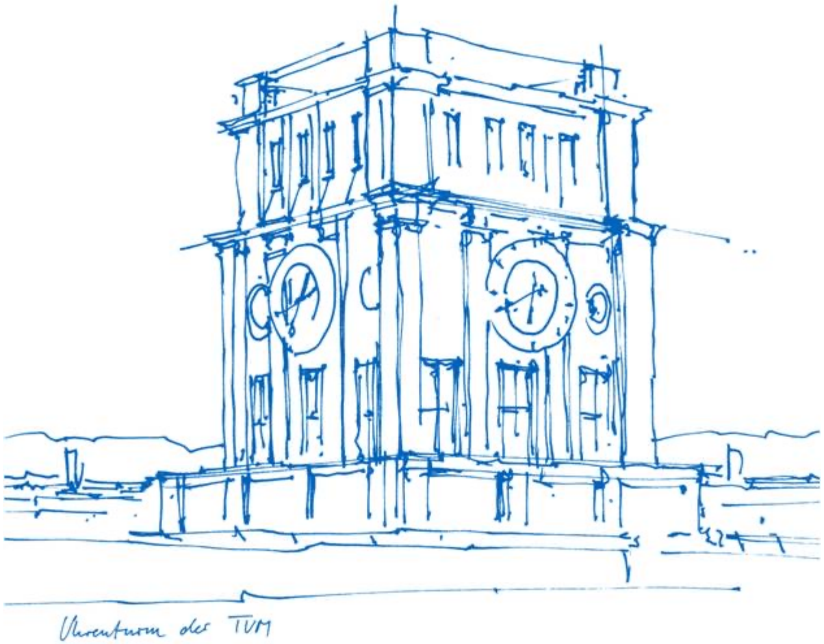
Agenda

1) Project Goal

2) Project Plan

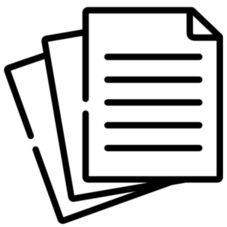
3) Data Gathering

4) Repository Structure

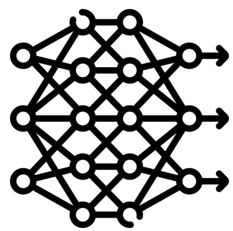


Process Discovery Machine Learning Pipeline

Input



Legal text



Output

{ Relevant: “0”}
{ Not Relevant: ”1”}

Embedding

- GloVe
- Word2Vec
- Word-sense induction
- N-Gram vector
- Count-based token vector

Models

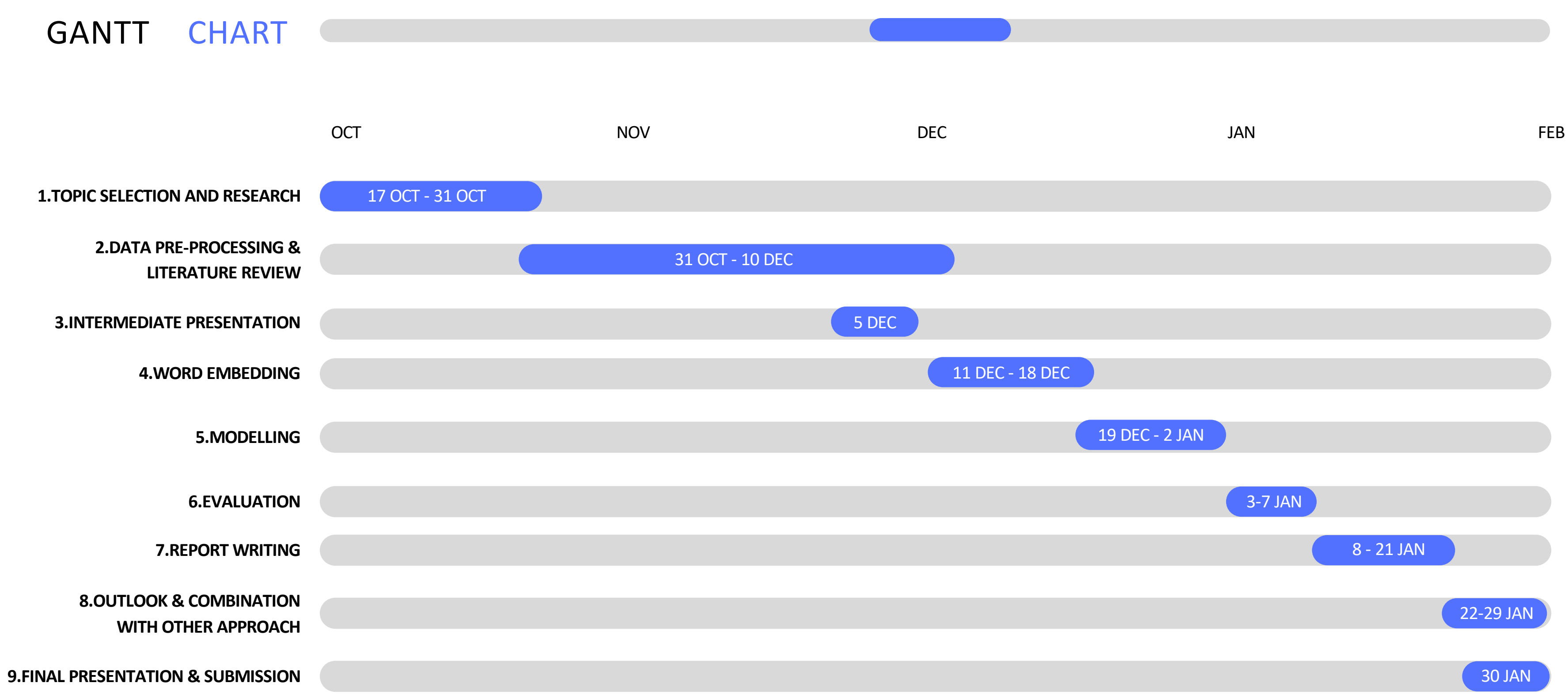
- Multinominal Naïve Bayes
- Support Vector Machine with SGD
- Multi-layer Perceptron
- Recurring Neural Network (RNN)
- Gated Recurrent Unit (GRU)
- Long Short-Term Memory (LSTM)

Evaluation

- Precision/Recall
- Micro aggregated F1 score

Source: Automated Business Process Discovery from Unstructured Natural-Language Documents 2020, https://link.springer.com/chapter/10.1007/978-3-030-66498-5_18

Currently still in the Data Pre-Processing Step



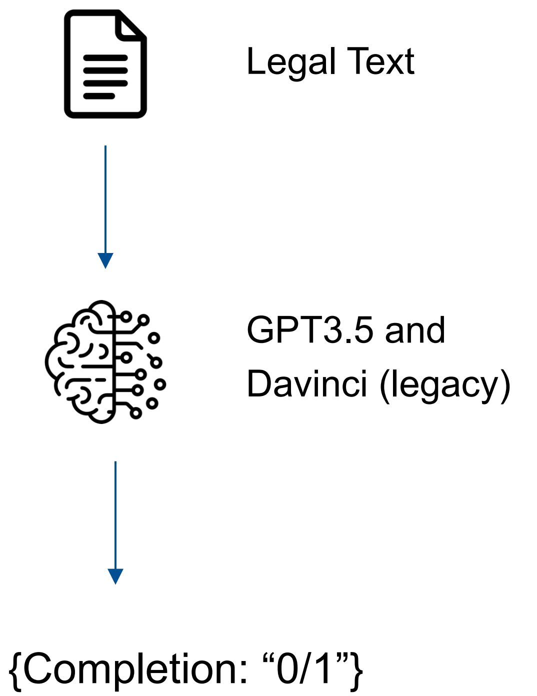
Australian Data + GDPR for the Data

	ratio 1:10 - input approaches				
	10% (group A)		45% (group B)	45% (group C)	100%
Processes	compliance relevant text paragraphs	informative relevant text paragraphs	non-relevant text paragraphs from relevant documents	non-relevant text paragraphs from non-relevant documents	total number of text paragraphs
1: travel insurance claim	21	28	220	220	489
2: know your customer	24	7	140	140	311
3: hire employee		9	40	41	90
4: GDPR 1-Data breach		8	36	36	80
5: GDPR 2-Consent to use the data		16	72	72	160
6: GDPR 3-Right to Access		11	50	49	110
7: GDPR 4-Right to Portability		4	18	18	40
8: GDPR 5-Right to Withdraw		4	18	18	40
9: GDPR 6-Right to Rectify		2	9	9	20
10: GDPR 7-Right to be forgotten		13	59	58	130
					1470

Data Labelling with LLM to Gather More Data

- **Davinci Fine-Tuning** : {
  prompt: “Process: <Process_Description> /n/n Text: <Legal_Passage> /n/n Relevant:"
  completion: “<0/1>###”}

- **GPT 3.5 Fine-Tuning** :
  { system_message: “Determine if the text is relevant to the process described” ,
  user_message: “Process Description : <Process_Description>/n/n Text to classify: <Legal_Passage>/n”,
  system_message: “<0/1>###” }



Cookie Cutter for the Repository Structure

```
├── Makefile           <- Makefile with commands like `make data` or `make train`
├── README.md          <- The top-level README for documentation and instruction on how to run the code.
├── data
│   ├── external       <- Data generated from the fine-tuned models for labeling.
│   ├── interim        <- Intermediate data that has been transformed.
│   ├── processed      <- The final data sets for modeling.
│   └── raw            <- The original, immutable data from the other repository which I decided to use.
│
│
├── models             <- Trained and serialized models, model predictions, or model summaries
│
├── notebooks          <- Jupyter notebooks. Contains for instance the fine-tuning of GPT for labeling notebooks.
│
├── references         <- Data dictionaries, manuals, and all other explanatory materials to understand the data bet
│
├── requirements.txt   <- The requirements file for reproducing the environment
│
│
├── setup.py           <- makes project pip installable (pip install -e .) so src can be imported
├── src                <- Source code for use in this project.
│   ├── __init__.py    <- Makes src a Python module
│   │
│   ├── data           <- Scripts to download or generate data
│   │   └── make_dataset.py
│   │
│   ├── features       <- Scripts to turn raw data into features for modeling and create word embeddings
│   │   └── build_features.py
│   │   └── build_word_embeddings.py
│   │
│   ├── models         <- Scripts to train models and then use trained models to make
│   │   │                 predictions
│   │   ├── predict_model.py
│   │   └── train_model.py
│   │
│   └── visualization  <- Scripts to create exploratory and results-oriented visualizations as well as word embeddin
│       └── visualize.py
└
```

Source: https://drivendata.github.io/cookiecutter-data-science/

Thanks!
Questions/Feedback?

Task

CLASSIFICATION OF TEXT

A) classify sentences into **„process relevant"** and **„process irrelevant"** (e.g. for process discovery from text)

--> Input: set of potentially relevant legal text; goldstandard of what is actually relevant (from insurance claims and banking use case)

1) No, I'm afraid a simplified multiclass classification like that would not serve the purpose of the task. We want the model to identify patterns between the process description and the regulatory text that lead to the correct label (class). With your proposed set-up the model could only learn patterns from the regulatory text to predict the label. But the label depends on the regulatory text in relation to the process description - just the regulatory text does not contain the necessary information to predict the label. This goes back to the basic understanding of the task: when thinking about the design of the approach we need to think about what information (features) are necessary in order to perform the prediction task. If the features are insufficient (only reg. texts) this can not lead to a satisfactory prediction/classification.