INFORMATION RETRIEVAL

LitCovid Track on Multi-label Topic Classification for COVID-19

Meltem Arslan
Gülsüm Tuba Çibuk Girgin

PREPROCESSING

- Pandas and nltk libraries are used.
  - In order to operate easily on the dataset, we chose to use dataframes of the Pandas library. We also used the natural language toolkit for preprocessing steps.
- Title, abstract, and keywords columns are chosen.
  - After examining the features of the dataset, we decided to include only three features. These features are title, abstract, and keywords.

NLTK

PREPROCESSING

- Applied preprocessing steps are shown in the table here.

Procedure	***Library***
Case folding	python standard library
Punctuation Removal	string library
Tokenization	nltk library
Stopword Removal	nltk library
Lemmatization	nltk library
Stemming	nltk library

STATISTICS

If we look at the number of documents in each class, we clearly see that the number of the documents in the prevention class is higher than others. Transmission and Epidemic Forecasting classes have less documents. In each document, token numbers are close to each other.

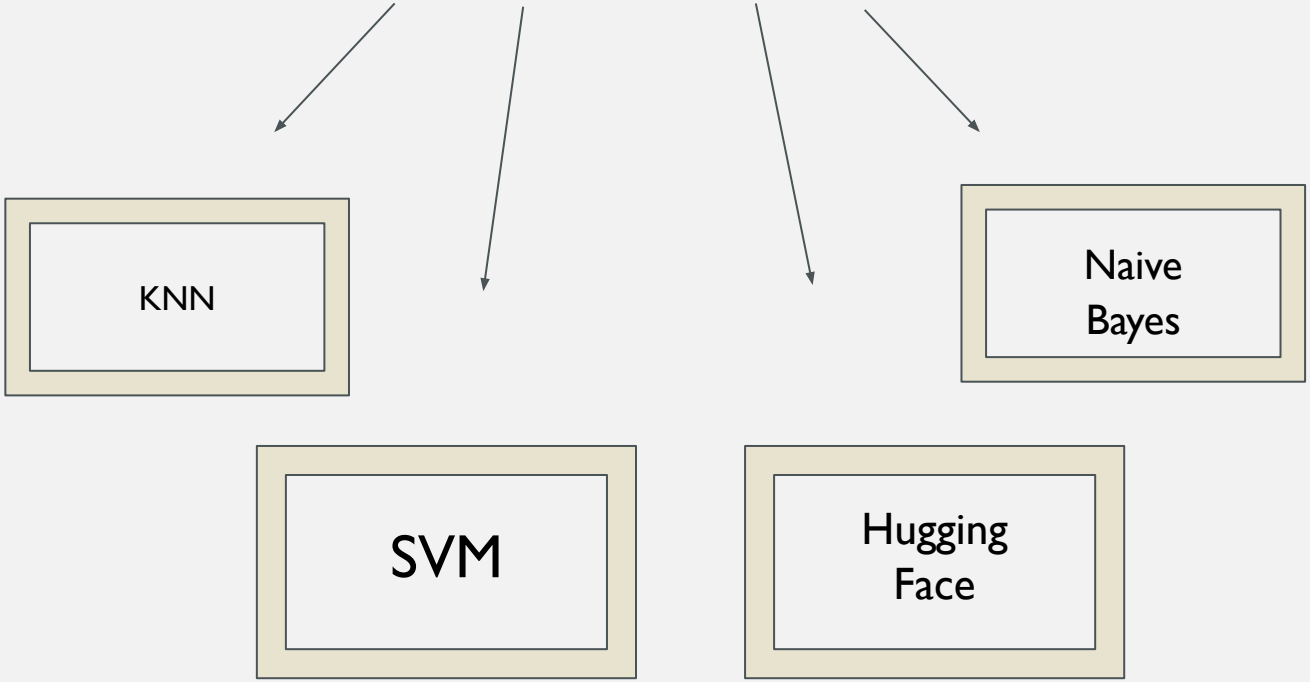
Classes	***# Documents***	***Mean # Tokens***
Treatment	8717	153
Diagnosis	6193	163
Prevention	11102	141
Mechanism	4438	139
Transmission	1088	145
Epidemic Forecasting	645	145
Case Report	2063	106

GENERAL APPROACH

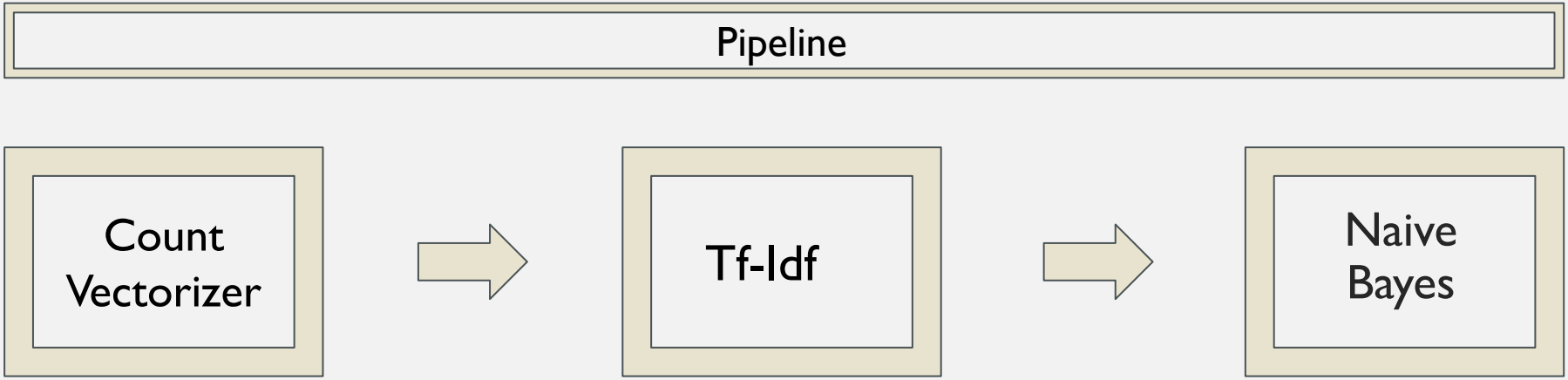
After preprocessing, the data needed be converted into required format. Since the problem is multiclass and multilabel problem. The data could not directly be given to the classifiers. So, we have followed a **binary approach** for each class and looked out their existences in each sample. Then, the results were combined in a csv file as requested for evaluation.

Classes	***Sample x***
Treatment	1
Diagnosis	1
Prevention	0
Mechanism	0
Transmission	0
Epidemic Forecasting	1
Case Report	0

APPLIED METHODS



Naive Bayes



NAIVE BAYES

- Bernoulli Naive Bayes from scikit-learn library.
  - We applied Bernoulli Naive Bayes classification method from scikit-learn library to our preprocessed and vectorized dataset. The reason for choosing Bernoulli Naive Bayes instead of other types of naive bayes is that its classification accuracy was higher.

NAIVE BAYES

Evaluation Results

- Tested with provided evaluation script.

	Precision	***Recall***	***F1-score***	***Support***
Treatment	0.8261	0.8115	0.8187	2207
Diagnosis	0.7257	0.6947	0.7098	1546
Prevention	0.8897	0.9298	0.9093	2750
Mechanism	0.7921	0.8416	0.8161	1073
Transmission	0.7500	0.0352	0.0672	256
Epidemic Forecasting	1.0000	0.0052	0.0104	192
Case Report	0.8733	0.5290	0.6589	482

NAIVE BAYES

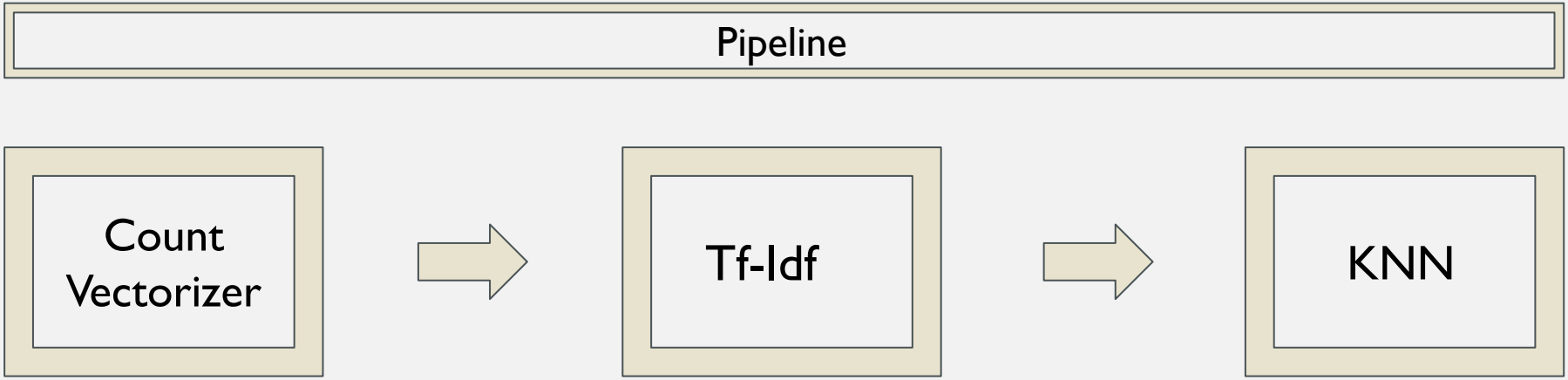
	Precision	*Recall*	*F1-score*	*Support*
micro avg	0.8272	0.7747	0.8001	8506
macro avg	0.8367	0.5496	0.5701	8506
weighted avg	0.8284	0.7747	0.7780	8506
samples avg	0.7979	0.8008	0.7807	8506

	mean precision	*mean recall*	*F1*
Instance-based measures	0.7979	0.8008	0.7993

NAIVE BAYES

- Tested with provided evaluation script.
    - Looking at the results, we can say that the classification method could not correctly classify the samples whose categories are transmission and epidemic forecasting. Furthermore, it was able to classify the samples of the prevention category most. The reason for that may be the difference between the numbers of documents in these categories.

K-NEAREST NEIGHBORS



K-NEAREST NEIGHBORS

Evaluation Results

- Tested with provided evaluation script.

	Precision	***Recall***	***F1-score***	***Support***
Treatment	0.8043	0.8174	0.8108	2207
Diagnosis	0.7565	0.7477	0.7521	1546
Prevention	0.8934	0.8687	0.8809	2750
Mechanism	0.8262	0.7176	0.7681	1073
Transmission	0.5549	0.3555	0.4333	256
Epidemic Forecasting	0.7284	0.6146	0.6667	192
Case Report	0.7467	0.3548	0.4810	482

K-NEAREST NEIGHBORS

	Precision	*Recall*	*F1-score*	*Support*
micro avg	0.8193	0.7640	0.7907	8506
macro avg	0.7586	0.6395	0.6847	8506
weighted avg	0.8147	0.7640	0.7841	8506
samples avg	0.8126	0.7922	0.7857	8506

	mean precision	*mean recall*	*F1*
Instance-based measures	0.8126	0.7922	0.8023

HuggingFace

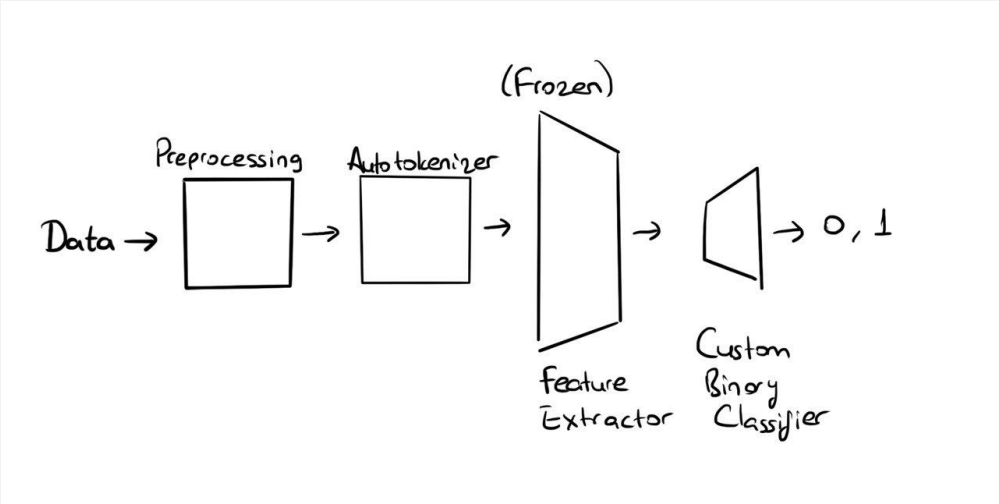
- Using pretrained models from HuggingFace as feature extractors
- AutoNLP Bert Covid
  - Binary Classification
  - Covid Related Dataset

- AutoNLP BBC News Classifier
  - Topic Classification

- DMIS LAB BIOBERT Feature Extractor
  - Covid Related Dataset
  - 180,849 downloads last month

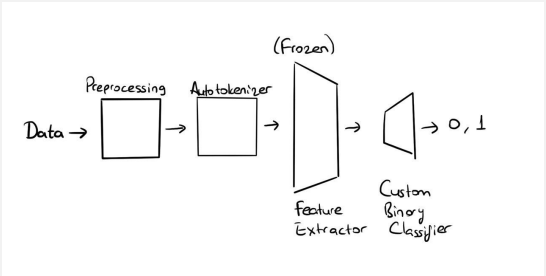
HuggingFace Models - Pipeline

- Weights of the feature extractor is frozen.
- Custom classifier is implemented.
- Different number of epochs



HuggingFace Models - Challenges

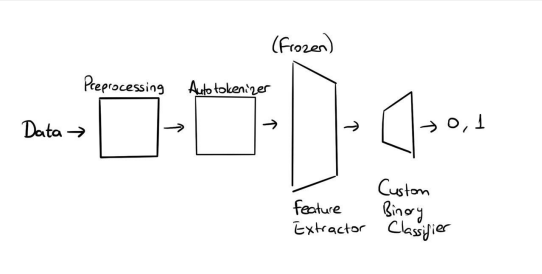
- Bert models are huge. They do not fit in memory (Local & Colab)
  - Solution: Freezing the weights of the feature extractor part.
- Training a model for 50 epochs take 30 minutes for each class.
  - Loss decreases, however accuracy does not change.
- Underfitting Problem
  - While binary classifying, it only chooses the class that has more data.
- Dataset is not evenly distributed.



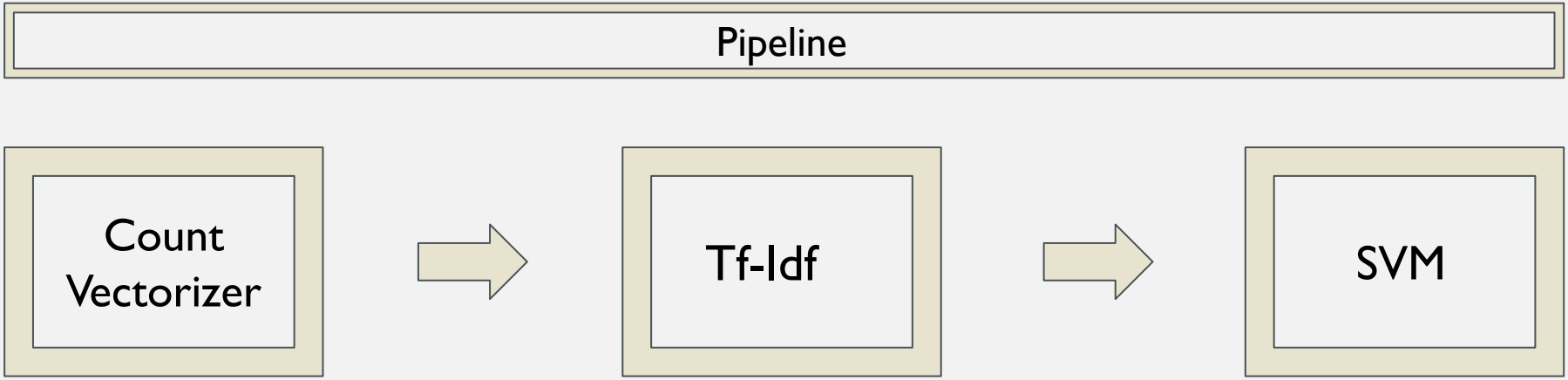
Step	Training Loss
500	0.717400
1000	0.710300
1500	0.698900
2000	0.698700
2500	0.696700
3000	0.692700
3500	0.691000
4000	0.692300
4500	0.689600
5000	0.688800
5500	0.688700
6000	0.690900

HuggingFace Models - Conclusion

- DMIS LAB BIOBERT Feature Extractor
  - While other models are giving 0.55 accuracy, this model gave 0.53 accuracy on Prevention Class. (0.55 accuracy means predicting only one label). So that this model made a different prediction.
  - With further training, 0.63 accuracy is obtained. (200 Epoch, 1 hour, Prevention Class)
  - Deeper classifier, more epochs, promising!



SUPPORT VECTOR MACHINES - I



SUPPORT VECTOR MACHINES - I

Evaluation Results

- Tested with provided evaluation script.

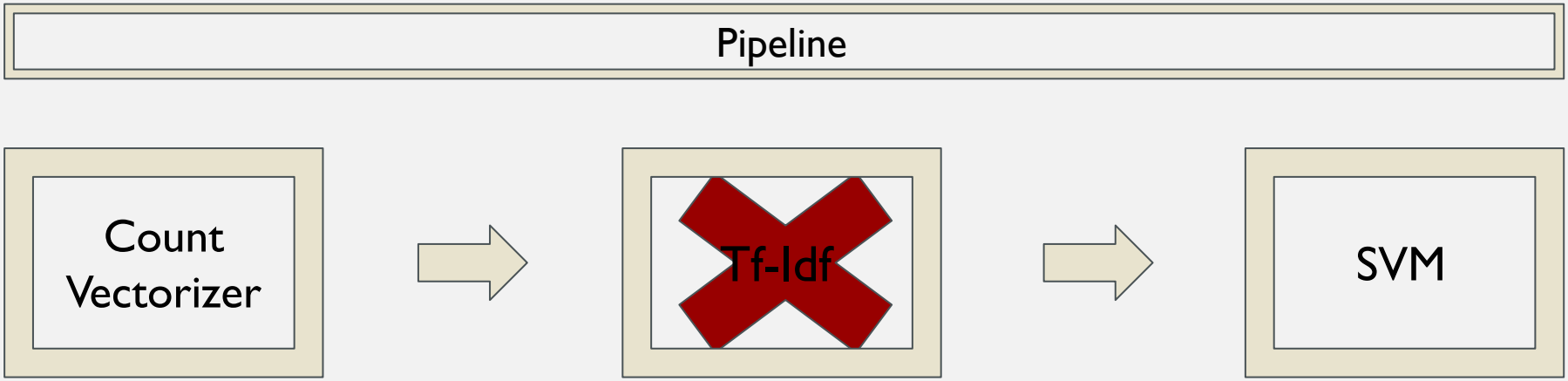
	Precision	*Recall*	*F1-score*	*Support*
Treatment	0.8902	0.8632	0.8765	2207
Diagnosis	0.8875	0.8060	0.8447	1546
Prevention	0.9340	0.9320	0.9330	2750
Mechanism	0.9000	0.8136	0.8546	1073
Transmission	0.8222	0.4336	0.5678	256
Epidemic Forecasting	0.8295	0.5573	0.6667	192
Case Report	0.9241	0.7075	0.8014	482

SUPPORT VECTOR MACHINES - I

	Precision	*Recall*	*F1-score*	*Support*
micro avg	0.9056	0.8401	0.8716	8506
macro avg	0.8839	0.7304	0.7921	8506
weighted avg	0.9036	0.8401	0.8679	8506
samples avg	0.8885	0.8701	0.8653	8506

	mean precision	*mean recall*	*F1*
Instance-based measures	0.8885	0.8701	0.8792

SUPPORT VECTOR MACHINES - 11



SUPPORT VECTOR MACHINES - II

Evaluation Results

- Tested with provided evaluation script.

	Precision	*Recall*	*F1-score*	*Support*
Treatment	0.8935	0.8518	0.8722	2207
Diagnosis	0.8913	0.7846	0.8345	1546
Prevention	0.9387	0.9295	0.9340	2750
Mechanism	0.8999	0.7959	0.8447	1073
Transmission	0.8015	0.4102	0.5426	256
Epidemic Forecasting	0.8333	0.5208	0.6410	192
Case Report	0.9160	0.7469	0.8229	482

SUPPORT VECTOR MACHINES - II

	Precision	*Recall*	*F1-score*	*Support*
micro avg	0.9084	0.8309	0.8679	8506
macro avg	0.8820	0.7200	0.7846	8506
weighted avg	0.9057	0.8309	0.8639	8506
samples avg	0.8821	0.8630	0.8583	8506

	mean precision	*mean recall*	*F1*
Instance-based measures	0.8821	0.863	0.8724

MODEL COMPARISON

	mean precision	***mean recall***	***F1***
NB	0.7979	0.8008	0.7993
KNN	0.8126	0.7922	0.8023
SVM-I	0.8885	0.8701	0.8792
SVM-II	0.8821	0.863	0.8724

ERROR ANALYSIS & POSSIBLE DIRECTIONS

- Error Analysis:
  - The models gave worse results in classes with fewer samples.
  - SVM gave better results on Epidemic Forecasting compared to Naive Bayes however, it gave worse results compared to KNN.
- Possible Directions
  - Tf-idf - > Word2Vec
  - Implementing Tf-idf + SVM without countvectorizer.
  - Samples can be selected according to their labels. In a class there will be equal number of different labels. (0-1)
  - Deep models can be trained with complexer classifiers and with more epochs. However, because of hardware limitations, it is hard to train.
  - Giving dataset to the deep models without preprocessing.

REFERENCES

- pandas https://pandas.pydata.org/
- nltk https://www.nltk.org/
- scikit-learn https://scikit-learn.org/stable/
- HuggingFace https://huggingface.co/
- AutoNLP Bert Covid (nurkayevaa/autonlp-bert-covid-407910467)
- AutoNLP BBC News C. (abhishek/autonlp-bbc-news-classification-37229289)
- DMIS LAB BIOBERT Feature Extractor(dmis-lab/biobert-v1.1)