# Motivation

In this modern era, multi label classification methods are of great necessity in many real -world problems. It is the association of each instance with a set of target labels simultaneously. For instance, a single book can be listed under multiple genres like mystery, sci-fi and thriller. Multi label classification is by its nature a challengeable problem due to many reasons such as the huge number of labels combinations, high dimensionality, unbalanced data, and many other reasons. In this project, we have explored Eur-Lex, the multilingual corpora which is a collection of legislative documents of the European Union.

# Problem Statement

The main challenge in multi label classification is that regular classification methods cannot be employed since several labels could be possibly assigned to a single document. Incorporating a model that could perform well in this scenario of large-scale legal domain was the major motive. Also, the model developed need to handle the problem of class imbalance in the data since lot of labels may appear a very few times in the documents. Finally, identifying the evaluation metrics was one of the problems to be addressed since normal evaluation metrics like accuracy, F1 measure etc. will not be a true reflection of the model’s performance on the data in multi label classification.

# Data Set

The dataset is crawled from EUR-Lex, which is a corpus of legislative documents of the European Union. It con­tains many dif­fer­ent types of doc­u­ments like treaties, leg­is­la­tion, case-law, leg­isla­tive pro­pos­als etc. which are in twenty-four official European languages. The dataset con­sti­tutes a very chal­leng­ing mul­ti­l­abel sce­nario due to the high num­ber of around 4000 labels and 20000 documents.

Majority of labels (3956) are on the various activities of the EU and its member states (EUROVOC descriptors). Each document is associated to at least 5.3 labels of this EUROVOC category. Other categories of labels are directory codes and subject matters which accounts for only 201 and 410 labels respectively. The mean averages about 2.21 labels per document of subject matters category and 1.29 labels per document of directory codes category.

A range of documents were excluded for the following reasons:

|  |  |
| --- | --- |
| **Reason for Excluding** | **Number of Documents** |
| Error Message | 214 |
| Non-English | 189 |
| Empty text field | 316 |
| Irrelevant Category | 50 |
| Non – English translations(Corrigendum) | 12 |
| **Remaining Total Documents** | **20000-781 = 19219** |

# Concept

**I) Pre-processing:**

The first step of pre-processing was to clean the data. Below tasks were performed on parsing the HTML files.

**A) Parsing of HTML files:**

1. HTML documents were parsed
2. Extraction of labels, text and ID’s of the documents
3. Cleaned the documents as below:

* Removed documents with no label
* Removed documents with no text
* Removed documents with non-English text

1. Transformation of the labels into binary column matrix of nearly 4000 columns

In order to analyse the dataset and to get maximum insight on the data, Exploratory Data Analysis was performed on the cleaned data as part of pre-processing as mentioned below.

**B) Exploratory Data Analysis:**

1. Calculated the average number of labels per document which was **5.3.**
2. Number of labels with respect to documents were explored:

Detailed numbers can be interpreted in the below graph:

* As visible in the graph, maximum documents of around 8000 had “6” labels each and around 5000 documents had around “5” labels
* Apart from this one document had maximum of “24” labels which is not shown in the graph

iii. Explored the frequencies of various labels to identify occurrence

* The graph depicts the ten most frequent labels of the documents that are featured
* The hundred most frequent labels of the documents are mentioned in the below sheet:

****

1. Correlation among labels were identified using binary variable association

* Threshold for high correlation was considered to be above 0.8 and filtered from the total correlated labels based on this threshold.
* The correlation details of the other top hundred frequent labels can be inferred from the below sheet attached:



* The below graph depicts the number of labels correlated with the top ten frequent labels in the dataset:

**Challenges faced in Pre-Processing:**

*i) Non – standard format of HTML files*

Faced difficulty in creation of uniform standard code for label and text extraction because the files did not follow a standard format.

*ii) Problems in Identical labels*

Because of spaces and different cases in identical labels, faced explosion of label numbers.

*iii) Huge size of Boolean Matrix*

Matrix was constructed considering each label as a column and each row a document. Binary values were assigned based on the presence and absence of a label in the document i.e. 1 if the label in the column value was present in the corresponding document of the row else 0. Since around 4000 labels and 19000 documents were there, the execution of code was difficult on such huge massive matrix size.

*iv) Difficulty in finding correlations*

Since the number of labels were too high and could not fit them all using binary association technique, were able to explore the correlations for only top hundred frequent labels.

**II) Extraction of Feature Vectors:**

**A) TF-IDF**

We have used the most popular term-weighting statistical measure, term frequency-inverse document frequency. With the resulting scores assigned to each word, we are able to identify the most important words in the corpus which are frequent in a document but not across documents in the whole corpus.

**B) TF-IDF N-grams**

Consecutive words might capture structure that isn’t present when one is just counting single words, and may provide context that makes tokens more understandable. Hence we have tokenized into consecutive sequences of words, called n-grams and then performed tf-idf on those grams. We used pairs of consecutive sequences of words considering n= 2.

**Challenges Faced:**  
We were not able to run the feature extractor on higher values of n greater than 2.

**III) Classifier Models:**

**A) Binary Relevance – Problem transformation**

* Binary relevance method transforms multi-label classification problem into multiple “binary classification problem”.
* The binary relevance classifier independently trains one binary classifier for each label (ensemble of single-label binary classifiers), ignoring the existence of other labels.
* Each classifier predicts either the membership or the non-membership of one class.
* Then the union of all classes predicted were considered as the multi-label output.

**Reasons for selecting this method:**

Apart from the fact that Binary Relevance is simple baseline and efficient model, it has been proven (Ó Luaces Rodríguez, 2012) to exhibit competitive performance with respect to a more complex method in difficult problems with many labels.

**B) ML – kNN - Algorithm Adaptation**

* This method is a multi-label lazy learning, derived from traditional K-nearest neighbour algorithm.
* For each unseen instance, its K nearest neighbours in the training set are identified.
* Based on statistical information gained from the label sets of these neighbouring instances, the maximum a posteriori (MAP) principle is utilized to determine the label set for the unseen instance.

**Reasons for selecting this method:**

**C) Ensemble of binary relevance and ML-kNN**

* Since both the methods have drawbacks, we considered ensemble model of binary relevance and ML-kNN.
* BR basically works by decomposing the multi-label learning task into a number of ***independent***binary learning tasks, it will not consider the correlations between the labels and would be ignored.
* ML-kNN has disadvantage that it considers only one label every time.
* Equal weightage was considered for both the models in the ensemble.

# Implementation

* Pre-processing of the dataset were implemented using NTLK library in python.
* Classifier models were implemented using scikit-multilearn library in python.
* TF-IDF calculation implemented using scikit-learn’s TfidfVectorizer in python.

# Evaluation

The evaluation method used was Hold-Out method in which the entire Eur-Lex dataset was portioned into training set and test set. About seventy-five percent of the data was used for training the model and the remaining twenty five percent were used as test set for evaluation of the trained models. Due to the limited system resources the k hyperparameter tuning on the dataset was not successful and could not proceed with further on that.

**Evaluation Measures:**

Since the predictions for an instance is not a single label but a set of labels, predictions were “fully correct” or “partially correct” (with different levels of correctness) or “fully incorrect”. We have employed the following label-based evaluation metrics on the classifiers.  For each label the metrics (precision, recall, F1) are computed and then these label-wise metrics are aggregated.

**A) Macro – average:**

* The performance of each individual class (precision, recall, F1 score) is computed and then averaged.
* All classes are given equal weightage in macro-average methods.

**Metrics Employed:**

* Macro precision
* Macro recall
* Macro F1

**B) Micro – average:**

* We have used this metric since micro averaged results have proven to be an effective measure when applied on a large number of classes.
* In this method we sum up the individual true positives, false positives, and false negatives for different samples separately and then applied to compute the results.
* A micro-average will thus aggregate the contributions of all classes to compute the average metric.

**Metrics Employed:**

* Micro precision
* Micro recall
* Micro F1

C) **Hamming Loss:**

* By this metric, we ignore partially correct classifications scenario and considers those as incorrect classifications.
* We have computed the fraction of wrong labels to the total number of labels by this metric.

**Results:**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Micro Precision** | **Micro Recall** | **Micro F1 Score** | **Accuracy** | **Hamming Loss** | **Macro Precision** | **Macro Recall** | **Macro Average** | **Macro F1** |
| **Binary Relevance - TF\_IDF** | 16.14 | 25.93 | 19.89 | 99.72 | 0.28 | 7.70 | 6.07 | 99.72 | 6.79 |
| **ML kNN - TF\_IDF** | 66.49 | 36.93 | 47.49 | 99.89 | 0.11 | 60.61 | 17.51 | 99.89 | 27.17 |
| **Ensemble of BR & MLkNN TF\_IDF** | 24.14 | 20.17 | 31.97 | 99.73 | 0.27 | 18.32 | 20.17 | 99.73 | 19.20 |
| **ML kNN - TF\_IDF NGram** | 65.68 | 35.56 | 46.14 | 99.89 | 0.11 | 58.49 | 16.23 | 99.89 | 25.40 |

**Challenges faced:**

* In multi-label classification, existing evaluation metrics like accuracy, precision, recall did not help since there is no perfect answer as correct classification or misclassification for an instance.
* There were no libraries or functions available for evaluation metrics of multi label classification problems, self-coding was done.

# Conclusion

In the initial phase of the project Binary Relevance model was considered but after exploring the correlations between labels, proceeded with ML-kNN since binary relevance does not consider the correlations. ML-kNN had also a drawback of local predictions that it considers only one label at a time, hence hypothesis was considered that better method would be an ensemble of both the models. On the contrary, the ensemble did not complement each other model and the performance was not better than the two models. Thus to conclude the best performance was delivered by the Multi-label KNN model outperforming the other models based on the evaluation results that was performed on the Eur-Lex data.

# References

Ó Luaces Rodríguez, J. D. (2012). Binary relevance efficacy for multilabel classification. *Progress in Artificial Intelligence*.