IPSA

Ma 412 - Mathematical Foundations for Statistical Learning

2024

Multi-Label Classification of Scientific Literature Using the NASA SciX Corpus

Student :
Guillaume Faivre

Teacher :
Atilla Kaan Alkan

17 décembre 2024

Table des matières

1 Introduction

1.1 Task description

This project aims to develop a system that predicts relevant keywords (e.g., ”solar wind”, ”lunar composition”, etc.) by analyzing scientific papers’ titles and abstracts. You will train a machine learning model to identify associations between document content and specific keywords, enabling it to recognize and label topics based on linguistic patterns. This task involves multi-label text classification, meaning each document may have multiple labels to capture the diverse themes or topics within scientific texts, unlike single-label classification, where only one label is assigned. The annotated dataset for this task is provided by the NASA ADS/SciX team and is available through the HuggingFace repository. The SciX corpus, comprising titles and abstracts of published papers, is split into training and test sets with 18,677 and 3,025 documents, respectively.

1.2 Project organization

The project is available on GitHub and can be accessed at the following link :
https://github.com/Yomguy25/MA412_project

The repository is organized as follows :

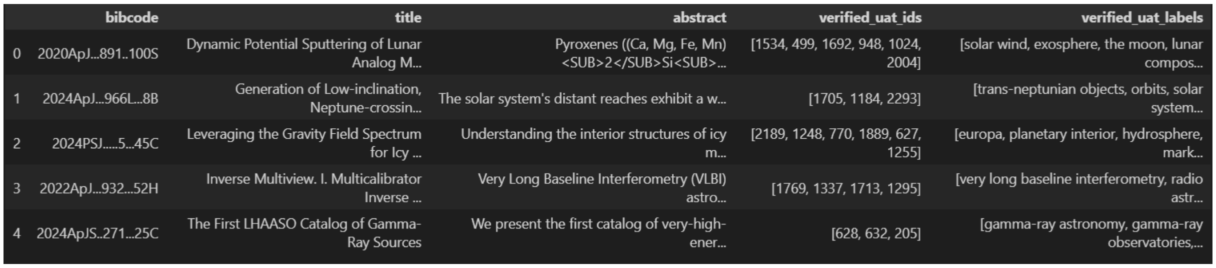
```
.
|-- data/                       # Directory containing datasets
|-- doc/                        # Documentation folder
|-- README.md                   # Project documentation
|-- data_processing.py          # Script containing utility functions
|-- hybrid.ipynb                # Notebook for a hybrid model combining rule-based
                                  and machine learning approaches
|-- model_tfidf.ipynb           # Notebook implementing TF-IDF vectorization for
                                  machine learning models
\-- rule_prediction.ipynb       # Notebook for rule-based keyword predictions
```

To observe the results, simply consult the provided notebooks and scroll to the end where the metrics are displayed.

2 Data exploration

2.1 Presentation of the dataset

Here are the 5 first rows of our dataset :



	bibcode	title	abstract	verified_uat_ids	verified_uat_labels
0	2020ApJ...891..100S	Dynamic Potential Sputtering of Lunar Analog M...	Pyroxenes ((Ca, Mg, Fe, Mn) ₂Si<SUB>...	[1534, 499, 1692, 948, 1024, 2004]	[solar wind, exosphere, the moon, lunar compos...
1	2024ApJ...966L...8B	Generation of Low-inclination, Neptune-crossin...	The solar system's distant reaches exhibit a w...	[1705, 1184, 2293]	[trans-neptunian objects, orbits, solar system...
2	2024PSJ.....5...45C	Leveraging the Gravity Field Spectrum for Icy ...	Understanding the interior structures of icy m...	[2189, 1248, 770, 1889, 627, 1255]	[europa, planetary interior, hydrosphere, mark...
3	2022ApJ...932...52H	Inverse Multiview. I. Multicalibrator Inverse ...	Very Long Baseline Interferometry (VLBI) astro...	[1769, 1337, 1713, 1295]	[very long baseline interferometry, radio astr...
4	2024ApJS..271...25C	The First LHAASO Catalog of Gamma-Ray Sources	We present the first catalog of very-high-ener...	[628, 632, 205]	[gamma-ray astronomy, gamma-ray observatories,...

Figure 1 – Head of the dataset

We have 5 columns :

- bitcode : unique id for each article (str)
- title : title of the article (str)
- abstract : a resume of the article (str)
- verified_uat_ids : id referring to label (keyword) (list of int)
- verified_uat_labels : keyword referring to the article (list of str)

2.2 Analysis of the labels

With 1864 distinct labels, the dataset presents an impressive diversity of themes. This richness is double-edged :

- Advantages : Extensive coverage means that the nuances of scientific articles are better represented.
- Challenges : The rarity of certain labels can make it difficult for the model to learn them.

The figures show that certain labels are largely dominant, such as those illustrated in the Top 20 most frequent keywords (Figure 2). These frequent labels, although easier to predict, run the risk of biasing the model by masking the rarer labels.

In contrast, Figure 3 shows the least represented labels. These labels, often specific to niche topics, will require special attention to ensure that they are correctly accounted for.

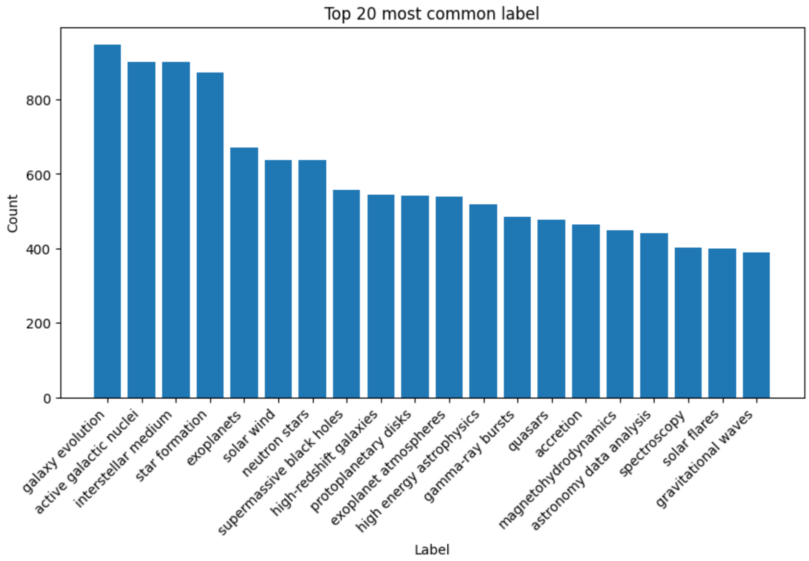


Figure 2 – Top 20 most common label

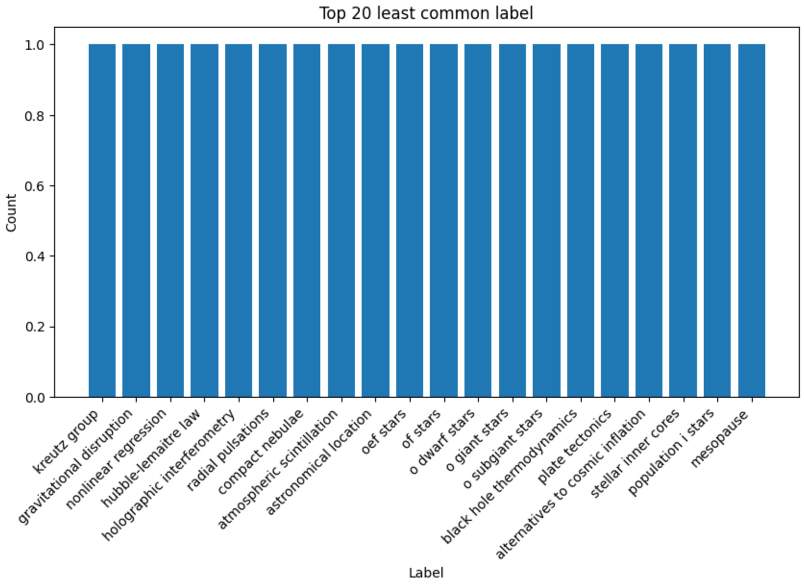


Figure 3 – Top 20 least common label

Visualising label co-occurrences (Figure 4) highlights frequent associations between certain labels. For example, an item labelled with 'Galaxy evolution' is often also associated with terms such as 'High-redshift galaxies'.

In the same way as above, these relationships could introduce bias if the model over-learns certain frequent associations, to the detriment of independent or rare labels.

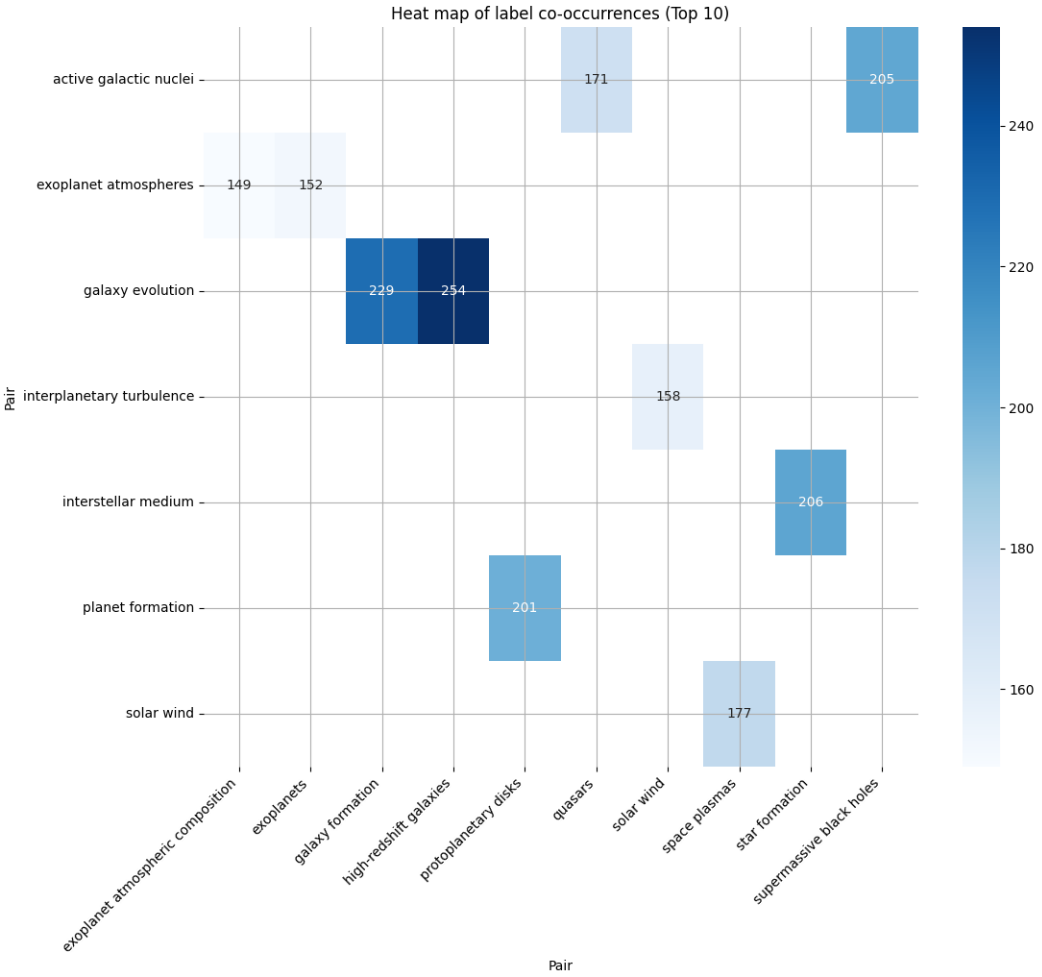


Figure 4 – Heat-map of label co-occurrences (top 10)

2.3 Analysis of the Title and Abstract

Analysis of the lengths of the abstracts in the dataset reveals the following information :

— Minimal word present in an abstract : 18
— Maximal word present in an abstract : 428
— Mean of word present in abstract : 210

By doing this research, I made some discovery, some abstract are empty (In fact they are considered as = 'None'). It could be better to drop the samples where the abstract are empty or an other solution is to keep these empty abstract and considering only the title to predict the label.

We can do the same analysis for the title, we obtain the following result :

— Minimal word present in an title : 1

— Maximal word present in an title : 36

— Mean of word present in titles : 13

Here, i didn't discover any title empty. So, for the development of my model I'm going to consider a new column called 'text' which will be the concatenation of 'title' and 'abstract'.

3 Metrics

In this section, the main metrics used to evaluate the performance of multi-label classification models are presented. Each metric provides a different perspective on the predictions, allowing a comprehensive and rigorous evaluation of the results.

3.1 F1-score

The F1-score is a metric that combines precision and recall into a single measure. It is particularly useful in cases where class imbalance is present. The F1-score is calculated as the harmonic mean of precision and recall :

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Where :

— Precision : The proportion of correctly predicted positive labels among all predicted positive labels.

— Recall : The proportion of correctly predicted positive labels compared to all actual positive labels.

In multi-label classification, the F1-score can be calculated for each label and then averaged using samples averages. It ranges between 0 and 1, where 1 indicates perfect prediction.

3.2 Hamming Loss

The Hamming loss measures the fraction of incorrectly predicted labels out of all the predictions made. It considers both false positives and false negatives in a multi-label context :

$$\text{Hamming Loss} = \frac{1}{N \times L} \sum_{i=1}^{N} \sum_{j=1}^{L} [y_{ij} \neq \hat{y}_{ij}]$$

Where :

— N is the number of samples.

— L is the total number of labels.

— y_{ij} represents the true labels.

— $\hat{y}_{ij}$ represents the predicted labels.

The Hamming loss ranges between 0 and 1, where 0 indicates perfect predictions.

3.3 Precision Score

The precision score evaluates the quality of positive predictions by measuring how many of them are correct :

$$\text{Precision} = \frac{\text{Number of True Positives}}{\text{Number of True Positives} + \text{Number of False Positives}}$$

It ranges between 0 and 1, where 1 indicates perfect prediction.

3.4 Accuracy Score

The accuracy score is a simple metric that measures the proportion of exactly correct predicted labels out of the total labels :

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Labels}}$$

It ranges between 0 and 1, where 1 indicates perfect prediction.

4 Models

In this section, we present three different approaches to keyword prediction : a rule-based method, a machine learning-based method with TF-IDF vectorization, and a hybrid approach combining rules and machine learning. The aim here is to compare the results obtained with the different approaches.

4.1 Rule-Based Model

In this subsection, we detail the implementation of a simple rule for predicting keywords based on the presence of specific words in the text. The rule used is based on searching for keywords directly in the titles and abstracts of scientific articles.

How it works :

— Pre-processing : Each text is converted to lower case to ensure that the search is not case sensitive.

— Direct match : The function observes the presence of each of the 1864 labels of the training dataset in the text being the concatenation of the title and abstract. If a match is found, the label associated with the keyword is activated.

— Output : A binary matrix is generated, where each row corresponds to a document and each column represents a label. A value of 1 indicates that the label has been predicted for the text.

To evaluate this rule-based method, tests were carried out using metrics such as F1-score, Hamming loss, precision_score and accuracy_score on the entire dataset train. The results are as follows :

Metrics	**Result**
F1-score	0.0121
Hamming Loss	0.0052
Precision Score	0.0118
Accuracy Score	0.0

Table 1 – Metrics and Results for rule-based model

The F1-score of 0.0121 is relatively low, indicating that the model struggles to correctly predict the relevant labels. This observation is further reinforced by the Accuracy Score of 0.0, which reveals that no document was entirely predicted with all correct labels.

However, the model does not perform completely at random, as evidenced by the Hamming Loss of 0.0052, which remains quite low. This suggests that the model correctly avoids assigning incorrect labels in many cases but fails to identify the true labels effectively.

The reason for this performance lies in the imbalance between the number of present labels (those appearing in documents) and the total number of possible labels in the dataset. The rule-based approach seems to favour identifying certain labels not truly relevant to the text while struggling to match the appropriate labels accurately. This imbalance creates a situation where the model reduces the number of mistakes overall but fails to capture the actual associations between the text and its intended keywords.

4.2 Machine-learning models with TF-IDF vectorisation

Definition :

The TF-IDF (Term Frequency-Inverse Document Frequency) vectorisation is a widely used method in natural language processing to transform text data into numerical representations that machine learning models can utilise. It measures the importance of a term in a document relative to the entire corpus, allowing it to highlight relevant words while minimising the influence of common, non-informative terms. The method consists of two main components : Term Frequency (TF) and Inverse Document Frequency (IDF).

Term Frequency (TF) :
The Term Frequency measures the frequency of a term t in a document d. It is calculated as :

$$\mathrm{TF}(t, d) = \frac{\text{Number of occurrences of } t \text{ in } d}{\text{Total number of words in } d}.$$

This metric gives higher importance to words that appear more frequently within a document.

Inverse Document Frequency (IDF) :
The Inverse Document Frequency reduces the weight of terms that occur frequently across the entire corpus, as these are less informative. The IDF is computed as :

$$\mathrm{IDF}(t) = \log\left(\frac{\text{Total number of documents}}{\text{Number of documents containing } t}\right).$$

Words that appear in many documents (e.g., “the”, “is”) receive a low IDF score, while rarer words have a higher score, highlighting their importance.

TF-IDF Score :
The final TF-IDF score combines the Term Frequency and Inverse Document Frequency as follows :

$$\text{TF-IDF}(t, d) = \mathrm{TF}(t, d) \times \mathrm{IDF}(t).$$

This score assigns a weight to each term, reflecting its significance within the document and across the corpus.

Models :

The models was built using three different vectorisation as follow :

- **Model 1** : Default vectorisation using `TfidfVectorizer()`.
- **Model 2** : Vectorisation with a limit on the number of features : `TfidfVectorizer(max_features=10000)`.
- **Model 3** : Vectorisation with additional parameters to control document frequency : `TfidfVectorizer(max_features=10000, max_df=0.8, min_df=0.01)`.

These configurations were designed to analyse the impact of different parameters on vectorisation performance.

Also, before the vectorisation, a series of text preprocessing steps were applied to standardise the input data :

— Conversion of text to lowercase,
— Removal of special characters,
— Removal of stop words,
— Lemmatization of the text.

Then, the different vectorisation was fit using the same method as `OneVsRestClassifier` combinated with `LogisticRegression`. The results has been evaluated on the test split part of the train dataset. The results are as follows :

Models	F1-score
Model 1	0.0993
Model 2	0.1081
Model 3	0.1217

Table 2 – F1-score Results for different Vectorisation

As observed, the addition of certain parameters influences the prediction results. To further improve performance, additional parameters could have been explored or fine-tuned for optimal results. A grid-search could have been useful for finding the optimum parameters. However, this requires a huge amount of computation and time, which cannot be achieved within the scope of this project.

Furthermore, alternative methods to logistic regression could have been considered for prediction, such as Binary Relevance or SVM.

4.3 Hybrid model

The hybrid method combines the predictions from the rule-based model and the most performant model obtained (Model 3), which uses TF-IDF vectorisation . The rationale behind this approach is to leverage the strengths of both methods : the rule-based model ensures that specific keywords explicitly present in the text are captured, while Model 3, powered by machine learning, identifies more complex associations between text content and labels. By merging these predictions, the hybrid method aims to improve overall performance.

The predictions from the two methods were combined, and the resulting union was evaluated on the validation dataset. The results are as follows :

Metrics \Models	TFIDF model 3	Rule-based model	Hybrid
F1-score	0.1287	0.0	0.1977
Hamming Loss	0.0022	0.0030	0.0043
Precision score	0.2691	0.0	0.1836

Table 3 – Metrics and results for Hybrid Model, Independant model with TF-IDF, independant Rule-based model

The zero values for the F1-score and Precision score of the rule-based model should not be considered as valid, as they likely result from an error in the evaluation process. However, a meaningful comparison can still be made between the TF-IDF model 3 and the hybrid model, which combines predictions from the rule-based approach and the TF-IDF model.

The results demonstrate a clear improvement in the F1-score for the hybrid model (0.1977) compared to TF-IDF model 3 (0.1287), indicating that the hybrid method effectively leverages the strengths of both approaches. This improvement suggests that the rule-based model captures certain keywords that the machine learning model might overlook, particularly when those keywords are explicitly present in the text.

However, this increase in the F1-score comes at the cost of a slight reduction in Precision (from 0.2691 to 0.1836). This trade-off can be explained by the hybrid model's broader coverage : the inclusion of predictions from the rule-based model introduces additional labels, some of which may be false positives. While this reduces the precision, it improves the overall recall by ensuring that more true labels are identified, leading to a higher F1-score.

In summary, the hybrid approach achieves a better balance between precision and recall, which is reflected in the significant improvement of the F1-score. This highlights the added value of combining rule-based methods with machine learning for multi-label classification tasks.

5 Conclusion

This project developed a multi-label classification system based on the titles and abstracts of scientific articles. Three main approaches were used : a rule-based model, a model using TF-IDF vectorisation with Logistic Regression, and a hybrid method combining the previous two. Performance was evaluated using metrics such as F1-score, Hamming Loss and accuracy.

Several avenues of improvement can be considered to optimise the multi-label classification model. First of all, optimising the parameters using a systematic search using GridSearch could refine the TF-IDF vectorisation and the classification model. In addition, exploring other vectorisation techniques such as Word2Vec, GloVe or BERT embeddings would help to better capture the semantic relationships between words.

In terms of models, it would be relevant to test alternatives to logistic regression, such as Random Forest, SVM or neural network approaches.

In conclusion, although the results obtained already demonstrate the effectiveness of a hybrid approach, many avenues remain open for refining the model, improving its generalisation capacity and meeting the challenges inherent in multi-label classification.

External Ressources

- **An introduction to multi-label text classification** :
  `https://medium.com/analytics-vidhya/an-introduction-to-multi-label-text-classification-b1bcb7c7364c`
- **Hierarchical Multi-Label Classification of Scientific Documents**, by Sadat and Caragea (2022).
- **Extreme Multi-Label Legal Text Classification : A Case Study in EU Legislation**, by Chalkidis et al. (2019).
- **Evaluating Extreme Hierarchical Multi-label Classification**, by Amigo and Delgado (2022).
- **A Survey on Recent Advances in Hierarchical Multi-label Text Classification**, by Liu et al. (2023).
- **Scikit-learn Documentation for TfidfVectorizer and Multi-label Classification** :
  `https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html`
  `https://scikit-learn.org/stable/modules/multiclass.html`