Feature Extraction and Embeddings

Data preparation

We load and preprocess the dataset that we got from the previous tp, encoding the *author* column using One-hot Encoding and get new column named *author_encoded*

```
label_encoder = LabelEncoder()
df['author_encoded'] = label_encoder.fit_transform(df['author'])
```

Construction of training/test set

We split the dataset into training and test sets (30% for the test) using *train_test_split* and since the dataset is imbalanced, we use the *stratify* parameter to ensure that the distribution of classes in training/test is similar

```
x_train, x_test, y_train, y_test = train_test_split(df['text'], df['author_encoded'], test_size=0.3, random_state=0, stratify = df['author_encoded'].values)
```

we get

	train set	**test set**
0	5529	2369
2	4230	1813
1	3944	1691

train_set = test_set * 2.33

Vectorization methods

we use two vectorization methods: *frequency-based* from *CountVectorizer* and *TF-IDF (Term Frequency-Inverse Document Frequency)* in order to convert raw text data into numerical resresentation that machine learning models can process, *frequency-based* will split text into individual words/tokens, count the frequency of each token in the document and generate a matrix where each row refers to a document, each column refers to unique token in the corpus and the valus represented are the frequency of each token in that document, for *TF-IDF* used to evaluate the importance of token is the document relative to the curpos with

TF(token, document)=(how many time this token appears in the document)/(total tokens in the deocument)

```
count_vectorizer = CountVectorizer(binary=False,analyzer= 'word', stop_words='english')
tfidf_vectorizer = TfidfVectorizer()
```

Train the models and predict

after training the model, as execution time we get
```

*count vectorizer : 0.023411035537719727 seconds*

*tfidf vectorizer : 0.025000572204589844 seconds*

*One-Hot Encoding : 0.020104408264160156 seconds*

```

one-hot-encoding is faster because it represents words as binary vectors, it doesn’t provide complex computation like counting and frequency computation