NLP Assignment 2023

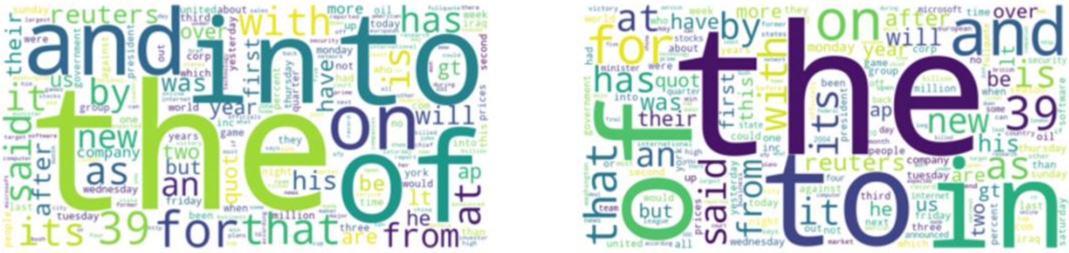
Group 10: Federico Colombo, Diego Sánchez, Adrià Termes, Nathaniel Thomas-Copeland, Ilias Trakas

30th June 2023

## 1. Packages + Dataset exploration

To carry out this project we used news text data to which each text was attributed one specific topic amongst the following: Science news, Sports News, World news, and Business news.

The first measure we took with the dataset was to break the different text entries into lists of tokens. This measure allowed us to count the total number of words within the *text* column, but most importantly count which words were the most frequently used. It was then that we discovered that out of the 58,831 words, the most popular were those such as "the", "to", etc. These words are called **stop words,** not conferring any meaning. TFIDF will be used to reduce their effect.



Source: Wordcloud for train and test set, respectively

## Training the models

After having explored the dataset, we then proceeded to training two classification models: MultinomialNB, and an SGDClassifier. Within the pipelines to train, two measures were introduced:

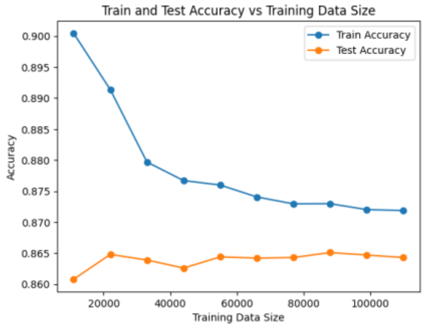
- Bag of Words vectorizer: this algorithm takes a text as input and transforms into a vector of the count of each word in the text column for a given text entry. This allowed our models to consider the frequency of certain words to attribute a given text to a topic.
- TF-IDF Transformer: TF-IDF is a metric used to evaluate the importance of a word contained in a document within a collection of documents. The Term Frequency (=TF) measures how often a given word occurs within a single document. The higher the TF, the more important a word is. The IDF measures the inverse of the frequency of a given word amongst all documents. Therefore, the more popular a word is, the less important it becomes, something achieved by a logarithmic multiplication of the number of instances / instances with word.

Although the performance of the models was not bad, we believed that these could clearly be improved. To do so, we proceeded by using GridSearchCV to find the best hyperparameter combination. One of the hyperparameters tested was the inclusion of N-grams, which trained or models using unigrams, bigrams, and trigrams. The model's performances have shown that the inclusion of bigrams and trigrams is generally optimal in terms of model performance (bigrams for the LinearSVC and trigrams for the SGDCLassifier), but presents certain drawbacks:

- Although n-grams might provide additional information to train the models on, their utility tends to dwindle as the training set sizes diminish
- Trigrams tend to overfit to the training set as the probability of trigrams being present in both train and test sets is lower (not to mention a brand new test dataset! We will test that…)

## 2. Evaluating model data-hungriness

To assess whether the models we train are data-hungry, we decided to assess the training and test scores for an SGDClassifier (which was chosen over the Naïve Bayes MultinomialNB classifier due to its higher performance and explainability). The method we used to assess it was checking a multiclass non-hyperparameterized SGDClassifier by chunks of 10% of training set data.



We could see that the accuracy score remained consistent other than for when the training set size was small (between 0 and 10% of the total training dataset). This means that increasing the training set beyond a certain point does not increase the model's predictive power as much as the computing. Therefore, 30% was our election. This leaves us with 32999 rows in the training set.

## 3. Multiclass models with reduced data to 30%: LinearSVC and SGDClassifier

Next, we compared the performance of a LinearSVC and SGDClassifier models when hyper parameterized and trained on the reduced training data. We explored two approaches: N-grams and hyper parameterization using GridSearchCV. For N-grams, we added pairs of words into the vector space, allowing the model to consider them in predictions. We tried 1, 2 and 3-grams.

We used the LinearSVC model within a pipeline. The parameter grid included the ngram_range and C (penalty parameter). We performed 5-fold cross-validation to ensure robustness. The best estimator obtained from the grid search was used to evaluate prediction metrics, and a cross-validation report data frame was generated to show the impact of different hyperparameters on model performance.

| | **Precision** | **Recall** | **F1-Score** | **Support** |
|---|---|---|---|---|
| **Science** | 0.87 | 0.88 | 0.88 | 2537 |
| **Sports** | 0.94 | 0.98 | 0.96 | 2458 |
| **World** | 0.92 | 0.89 | 0.90 | 2509 |
| **Business** | 0.87 | 0.86 | 0.87 | 2496 |
| **Accuracy** | | | 0.90 | 10000 |

Similarly, we used the SGDClassifier model and performed hyper parameterization using GridSearchCV. We defined a base pipeline with the necessary transformers and classifiers and specified the configurations to tune. The grid search helped us find the best hyperparameters for the model.

| | **Precision** | **Recall** | **F1-Score** | **Support** |
|---|---|---|---|---|
| **Science** | 0.87 | 0.87 | 0.87 | 2537 |
| **Sports** | 0.94 | 0.98 | 0.96 | 2458 |
| **World** | 0.91 | 0.89 | 0.90 | 2509 |
| **Business** | 0.87 | 0.86 | 0.87 | 2496 |
| **Accuracy** | | | 0.90 | 10000 |

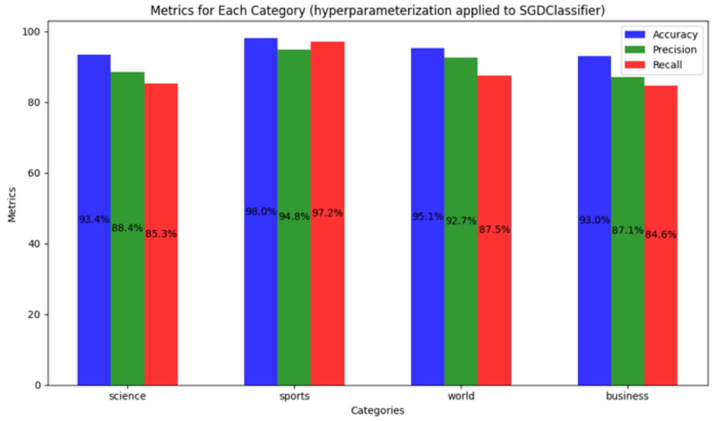
For the LinearSVC(), we found that the best parameters were C=1.0 and n-gram range=(1, 2). The model achieved an overall accuracy of 90%. The precision, recall, and F1-score varied for each category, with sports performing the best and business performing the worst. For the SGDClassifier() model the best parameters were found to be alpha=1e-06 and n-gram range=(1, 3). The model achieved an accuracy of 90%, with variations in precision, recall, and F1-score for each category.

In conclusion: Sports was the category best captured by the models, while business was the least accurately predicted category. The inclusion of trigrams in the feature space could improve the model's performance, but the overfitting of the training data should be considered. Regularization helped reduce overfitting and enhanced the model's ability to interpret unseen data. Reducing the training data size by 70% resulted in a slight decrease in performance, indicating a reasonable tradeoff considering the computational efficiency gained. Further exploration can be done by binarizing the problem for each category to extract more specific semantics.

## 4. **SGDClassifier for binary classification** (FEDE)

Transitioning from our previous sections, where we delved into data exploration, tested multiclass variants of Naive Bayes, LinearSVC and SGDClassifier models, and even experimented with reduced data models; we now direct our attention towards a deeper understanding of the linear Stochastic Gradient Descent (SGD) model. We delve into the binary variants of these models, evaluating their performance across the four distinct categories - Science, Sports, World, and Business. We will then examine feature importance in terms of words analyzing weight attributed by the model to unigram, bigrams and trigrams, if hyperparameters consider including all three.

### Binary Classification



For the binary classification we defined a function for the display of the classification visuals and storage of the different metrics, we used the previously created grid search to optimize hyperparameters, finally we compared precision, accuracy and recall across the four different categories. The four different models show different levels of metrics, with Sports still being the best in terms of recall and precision, and science and Business the worst ones. Although our results are good there is still a risk that most of the sentences may overfit not only to the train dataset, but also to the test (this is way later we will present how our model perform on a different dataset).

### Unigrams, Bigrams, Trigrams

**Science:** For the category of science, we found that unigrams such as 'space', 'nasa', 'software', and 'linux' play a vital role in identifying news belonging to this category. Simultaneously, words related to the economy, sports, and general weekdays like 'Sunday' are negatively indicative of science news. The bigrams and trigrams painted a more nuanced picture, revealing that names of cities associated with economic activities (e.g., 'New York', 'Seattle Microsoft') and specific wordings like 'world largest chipmaker' bear substantial weight in negating a piece of news from being classified as science.

**Sports:** The sports category was distinctly identifiable with unigrams like 'coach', 'football', 'basketball', whereas 'internet', 'minister', 'software', and 'Microsoft' were inversely influential. Bigrams helped identify top stars and competitions, such as 'Kobe Bryant', 'world cup', 'Manchester United', 'us open'... Interestingly, our model found that trigrams added little to no value in improving categorization accuracy for sports news, signifying the effectiveness of unigrams and bigrams.

**World:** The 'World' category presented a complex scenario. Specific city names, media outlets, or a combination of both played a vital role in the identification process. The value of bigrams emerged significantly, revealing correlations with entities like 'United Nations', 'Canadian Press', and 'Prime Minister'. However, counterintuitive results such as 'Tokyo Reuters' being negatively weighted indicate that our model may have room for improvement in this category.

**Business:** In the business category, terms like 'Airlines', 'tax', 'banks', 'oil' were indicative. However, we found peculiar classifications, where 'Reuters', 'New York Press' were deemed as non-business features. Phrases like 'Intel Corp' (bigram) and 'New York Stocks' (trigram) were also misclassified, raising questions about the model's robustness in this category, although prediction metrics are good.

## Taking a closer look at n-grams

- For all categories, Unigrams and bigrams are particularly useful, as the words used to predict these topics demonstrate a good-catching model with meaningful combinations of words. However, the same cannot be said for trigrams, with doubtful classifications always.
- When it comes to business news lots of unigrams, bigrams, and trigrams have no meaning, which shows that for this topic, our models are struggling to capture the meaning of the words.

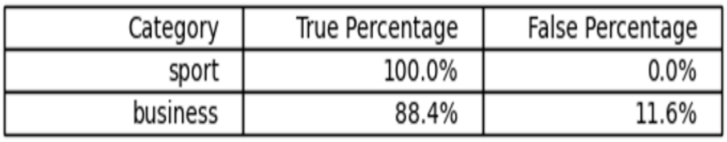
| | **Unigrams** | **Bigrams** | **Trigrams** |
|---|---|---|---|
| **Science** | space, software, nasa, linux... | *washingtonpost com, ft news, open source, video game... | *reuters intl business, lt gt san, 36 400 million, oracle corp 36... |
| **Sports** | coach, football, cup, team, olympic, season, ap... | kobe bryant, manchester utd., world cup, us open... | *greece michael phelps, game to win, elite sports the... |
| **World** | *afp, iraq, president, peace, elections, terrorism, baghdad... | *canadian press, ap tokyo, north korea, united nations... | *new york stocks, ap president bush, brasilia brazil reuters... |
| **Business** | ap, afp, team, game, linux, nasa, washingtonpost, reuters... | intel corp, reuters microsoft, manchester united, ap federal... | *reuters verizon wireless, new york stocks, corp said on... |

Source: Own creation. Green denotes if it helps more by identifying relationship with the topic (by inclusion), red if it helps by identifying words not related (by exclusion). * Denotes doubtful world classifications in the other side (inclusion or exclusion), not included here.

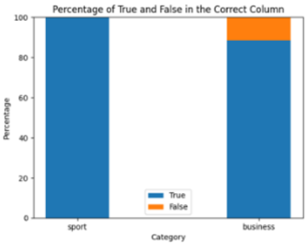
# 5. Testing a New Dataset on Known Categories for Inference Beyond Our Dataset

Continuing from our previous tests, we decided to use a different set of data. We wanted to see if our SGDClassifier, which is used to categorize text, could work well with different but related data, not just the news text data we originally used. For this, we used a free data source from Kaggle: BBC News Archive dataset (https://www.kaggle.com/datasets/hgultekin/bbcnewsarchive). This new dataset also had two categories, 'business' and 'sports', that were in our original training data. These were important to us because they were the best and worst performing categories in our earlier tests. So, we could see how well our model could predict both types of categories. Once the new data was prepared and adjusted to match the structure, we reintroduced our star performer, the SGDClassifier, to this new challenger.

**The results were surprisingly positive!**. The overall accuracy reached a whopping 94% with the external dataset! When it came to category-specific performance, the model maintained its perfect track record for 'sports' with a 100% success rate, while 'business' was accurately predicted 88.4% of the time. These great results show that our model and the words it learned to recognize can work well with other similar text datasets. What's even more impressive is that we achieved these results using only 30% of our original training data. So, yes this shows that our SGDClassifier is not that bad!



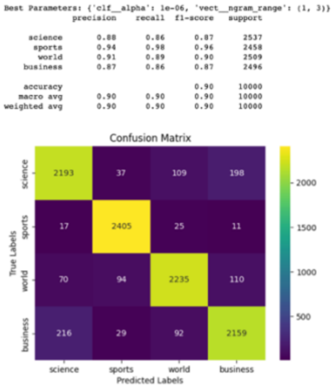
| Category | True Percentage | False Percentage |
|---:|---:|---:|
| sport | 100.0% | 0.0% |
| business | 88.4% | 11.6% |



## 6. The Role of Lemmatization in Prediction Metrics

We also inspected another part of getting text ready for analysis, called lemmatization. This means changing words to their simplest or base form to reduce repeated and different forms of the same word in the text. We wanted to see if doing this would give us better results overall. In pursuit of this, we used the Lemmatization functionality from Spacy to transform our text data into a 'bag-of-lemmas'. We then retrained using the same hyperparameters: {'clf__alpha': 1e-06, 'vect__ngram_range': (1, 3)}.

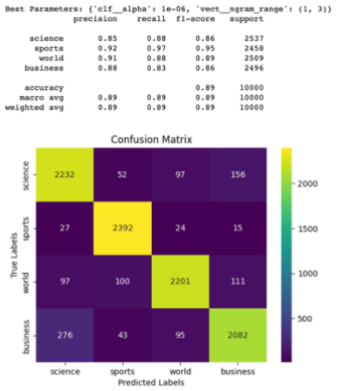
The model's performance was not improved; in fact, it was slightly worse! The overall accuracy dropped to 89%, and the precision, recall, and F1-scores for each category were not as high as when we used the original text. Our model, it seemed, was better able to categorize the news articles when using the original, unaltered words rather than their lemmatized counterparts.



```
Best Parameters: {'clf__alpha': 1e-06, 'vect__ngram_range': (1, 3)}
              precision    recall  f1-score   support

     science       0.88      0.86      0.87      2537
      sports       0.94      0.98      0.96      2458
       world       0.91      0.89      0.90      2509
    business       0.87      0.86      0.87      2496

    accuracy                           0.90     10000
   macro avg       0.90      0.90      0.90     10000
weighted avg       0.90      0.90      0.90     10000
```



```
Best Parameters: {'clf__alpha': 1e-06, 'vect__ngram_range': (1, 3)}
              precision    recall  f1-score   support

     science       0.85      0.88      0.86      2537
      sports       0.92      0.97      0.95      2458
       world       0.91      0.88      0.89      2509
    business       0.88      0.83      0.86      2496

    accuracy                           0.89     10000
   macro avg       0.89      0.89      0.89     10000
weighted avg       0.89      0.89      0.89     10000
```

**These findings support the idea that the exact way words are used and appear in text can have important meaning that's useful for predicting things**. Basically, the real-world use of words can sometimes be more helpful for making predictions than their basic definitions. While using lemmatization can be helpful in some cases, it might not always be the best for jobs like ours where we categorize text. This shows how complex language is.