UiS DAT305 Mandatory Project - News Article Classification

Gustavo Henrique Assunção Paiva

Abstract **– This report details the classification of German news articles into five different categories.The data, provided by the professor, were pre-processed, grouped into a single .csv file, and sanitized. Once sanitized, TF-IDF was applied to vectorize the texts. Finally, the models LinearSVC and MultinomialNB were trained and compared, resulting in a higher accuracy of 90.59% from LinearSVC.**

Keywords **– Text Classification, Machine Learning, TF-IDF, LinearSVC, German News, MultinomialNB, Comparison.**

1 Introduction

Text classification is a common challenge faced by Natural Language Processing (NLP), with the classification of news articles being a perfect example of this case study. This paper compares the results of two different classical AI models, LinearSVC and MultinomialNB, applied to this specific task.

2 Methodology

To address this task efficiently, a pipeline of methods had to be applied, which are data pre-processing, feature extraction, model selection and training.

2.1 Data Pre-processing

The raw data was acquired in a folder containing a .csv file with the labels and a folder with the articles in .rds format. The first step was to load the .rds files, so the library pyreadr was used to read all of the RDS files, which were then grouped in a list. Then, the library pandas was used to concatenate the list and merge it with the labels file, creating a single .csv file with all information consolidated.

Next, it was necessary to sanitize the raw text within the .csv file. So, the libraries re and unidecode were used inside the function clean_text to clean numbers, accents, special characters and lower all characters. These processes standardized the data to vectorization.

Finally, an exploratory analysis was performed on the consolidated dataset. As shown in Figure 1, the bar chart reveals that the dataset is imbalanced, with 'Politik' being the dominant class and 'Polizei' the minority. This imbalance reinforces the need to use the F1-Score, rather than just accuracy, for a fair model evaluation.

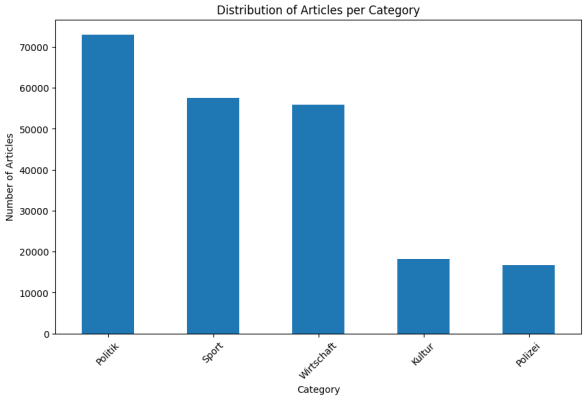

Figure 1: Distribution of articles per category.

2.2 Feature Extraction and Data Splitting

In order to achieve a stronger 'foundation', stop words were obtained using the nltk tool. These words were then cleaned, using the same clean_text function. Then, the sanitized data was split into the test and training group. Finally, the split data was vectorized using the TfidfVectorizer class from the scikit-learn library, with a limit of 15000 features, also using the cleaned stop words as a removal parameter. This resulted in two numerical matrices ready to be used on the model's training.

2.3 Model Selection and Training

LinearSVC and MultinomialNB were the chosen since they are classical for text classification cases, and have a very different approach to the task, while the MultinomialNB works with the probability of each word's appearance, LinearSVC maps them out and try to determine frontiers for each category. The models were then trained using the separated training data, learning the patterns of each category. Finally, the performance of both trained models was evaluated on the unseen test set, using metrics such as precision, recall, and F1-score.

3 Results

Following the training of LinearSVC and MultinomialNB models, this section presents the results achieved by each one of them on the test set. It is showcased with a direct comparison between the models, and then followed by a detailed analysis of the winning model.

3.1 Model Comparison

The following table presents the direct comparison between the two models As seen on the table, LinearSVC

Table 1: Performance comparison of the two models.

Model	Accuracy	F1-Score
LinearSVC	90.59%	0.90
MultinomialNB	89.17%	0.89

is the clear winner with a 90.59% accuracy and 0.90 F1-Score. The following section details LinearSVC performance in depth.

3.2 Detailed Results

LinearSVC proved to be the superior model in this case. The specific details are shown on the following full classification report and confusion matrix

```
 EVALUATION RESULTS - LinearSVC
Overall Accuracy: 90.59%

Detailed Classification Report:
              precision    recall  f1-score   support

      Kultur       0.90      0.81      0.85      3629
     Politik       0.87      0.91      0.89     14604
     Polizei       0.92      0.87      0.89      3333
      Sport        0.97      0.97      0.97     11494
  Wirtschaft       0.88      0.88      0.88     11158

    accuracy                           0.91     44218
   macro avg       0.91      0.89      0.90     44218
weighted avg       0.91      0.91      0.91     44218
```

Figure 2: Classification report for LinearSVC.

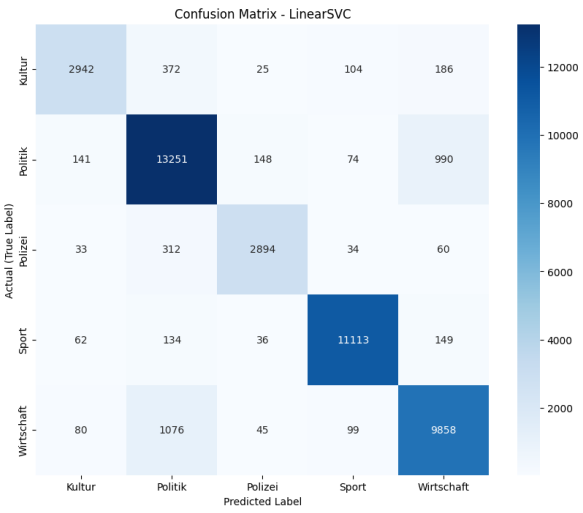

Figure 3: Confusion matrix for LinearSVC.

As shown in Figure 1, while the model performed very well in the Sport category(F1-score), it has a lesser success on the Kultur category. This category has a good Precision(0.9) but a low Recall (0.81). This means the model missed almost 20% of the truly 'Kultur' articles, a percentage that can not be ignored.

The F1-score, a harmonic mean, captures this imbalance. The Confusion Matrix confirms this data, revealing a great number of misclassifications going to Politik instead of Kultur.

4 Conclusion

The general idea of this project has shown successful results. The dominating victory from LinearSVC, with a 90.59% overall accuracy, over MultinomialNB, with only 89.17%. It is undeniable with an almost 1.5% difference in accuracy, that LinearSVC is the superior learning model for this case. However, a critical analysis of the model's limitations and techniques is fundamental to improve future tasks.

During the development, a logical error was neglected until the late stages, the stop words were not being cleaned, resulting in their permanence during the training and testing. Upon proper sanitation, there was a drop in accuracy, from 91% to 90.59%, revealing that those words were being used as a guide to the model. However, the outcome of this correction, despite the drop in accuracy, created a more robust model by operating without the assistance of unreliable data noise.

It is important to note that even with the immediate success, the LinearSVC model + TF-IDF is a 'Bag-of-words' method. It does not compute word order, resulting in no context, nuance, or irony understanding. As a consequence, it lacks the differentiation between 'team A won against team B' and 'team B won against team A', for example. Finally, it results in an insufficient model for tasks that require a deeper semantic and contextual understanding, making it unsuitable for more complex tasks.

For future work, to overcome this 'Bag-of-Words' limitation, the focus should shift to models that analyze sequential context. Architectures such as Transformers, OpenAI's GPT and Google's BERT for example, would represent a clear improvement. These models are designed to understand the semantic relationships and nuances of word order, directly addressing the primary weakness identified in the current pipeline.

Ultimately, while future work with Transformers promises deeper contextual understanding, the LinearSVC and TF-IDF pipeline developed in this report proved to be a robust, efficient, and highly accurate solution for the defined task, successfully achieving the project's primary objectives.

References

[1] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825-2830, 2011.

[2] W. McKinney, "Data Structures for Statistical Computing in Python," in *Proceedings of the 9th Python in Science Conference*, 2010, pp. 56-61.

[3] S. Bird, E. Loper, and E. Klein, *Natural Language Processing with Python*. O'Reilly Media Inc., 2009.

[4] T. De Smedt, "Unidecode: ASCII transliterations of Unicode text," *Python Software Foundation*, 2012. [Online]. Available: https://pypi.org/project/Unidecode/