June 12, 2025

Natural Language Processing

Project 1.6: Sentiment analysis (Twitter Data)

Organization

- Please read the project description carefully. All the details are presented in this document.
- Submit your solution no later than June 26th, 23:55. Submit your solution using the corresponding ISIS section, where you have downloaded this document.
- You can make unlimited re-submissions until the deadline, but the latest received submission will be graded only.
- Submit your solution as a zip file with the following name format: 'NLP_project1_6_[Your Names separated by underscores].zip' containing your codes and report files.
- For those who decided to work on the project in a group, one submission of the solution would be enough.
- For each task, you see a percentage that shows the task's score in the whole project.
- Please use Python 3 for coding.
    - You are free to use any package/module/library for different tasks (except when it's clearly mentioned in a task to not use available packages to solve the task).
- Please put comments on your code, wherever you think it would help to improve the readability and understandability of your code.
    - You can submit your codes as a .py file or Jupyter notebook
- You should submit a short report, describing your approach to solving different tasks and also provide the obtained results (e.g., the evaluation result of your models).
    - For the report please use the ACM proceedings template from here (https://www.acm.org/publications/proceedings-template). The Office Word and latex templates are provided on the page.
    - The report should be between 4 to 6 pages.
- There is a bonus task at the end of the document, as the name implies it is not a mandatory task, but you can take your time and solve it to get more scores.

Plagiarism Statement

Your project, including the report and the code, will be checked against other submissions in the class. We trust you all to submit your own work only. Copying someone else's code and report and submitting it with minor changes, will be treated as plagiarism.

If you have any further questions, please write to: salar.mohtaj@tu-berlin.de

Introduction:

Sentiment analysis could be formulated as a text classification task (like spam filtering), where each sentiment is presented as a category.

Sentiment classification is the task of predicting the polarity of a piece of text (e.g., positive, negative, or neutral) using text classification models. In this project, we will investigate sentiments in tweets. We define our problem statement as follows: we seek to detect the polarity of a tweet (being positive, neutral, and negative).

Data:

In this project, you receive a zip file containing a CSV file. There are a number of columns in the files including textID, text, selected_text, sentiment, Time of Tweet, and so on. Please keep text and sentiment as the input and output features and ignore the other columns.

Moreover, you can sample a part of the dataset, if it's too big to train a model on using your machine. Otherwise, you can use the whole dataset to train your models.

Tasks:

Task 1: Extract insights from data (15%)

For this task, you should extract some insights (i.e., some statistics and graphs) from the provided data. They should help someone who has no access to the dataset to get some ideas about the dataset and the distribution of terms, and different classes. Here are some ideas:

- It could be a graph comparing the length of positive, negative, and neutral news.
- Number of unique words in each category and also in the whole dataset
- Word cloud for each class.
- …

Please don't limit yourself to these two examples and try to summarize the data with numbers and graphs. Please highlight some of the most important findings in your report.

Task 2: Pre-processing (10%)

In this task, first, apply all the necessary pre-processing steps that you think would help to better prepare your data for the next steps. You don't have to apply all the pre-processing tasks which are covered in the course. Regarding the report, you should briefly mention all the steps you have applied to the data and a brief description of why you've decided to apply the chosen pre-processing steps (and why not the others).

Task 3: Text classification (60%)

In this task you should do the following sub-tasks:

- Split data into train and test sets. Use 20% of the data as the test set. Make sure to under or over-sample in case of imbalance in classes.
- Train a naïve Bayes model on the training part and test it, using the test set.
    - Compare the impact of different vectorization models (e.g., count vectorizer, TF-IDF, and …) on the final performance of your naïve Bayes model.

    - Compare the impact of different pre-processing pipelines (e.g., with and without stop words, stemming, and …) on the final performance of your naïve Bayes model.
    - Perform error analysis on the model's prediction. In other words, analyze errors that have been made by the model and describe why your model couldn't work well in case of these errors.

- Train a feed-forward neural network model and report its performance (F1 score) in detecting sentiments on test data.
    - Again, compare the impact of different vectorization approaches on the final performance of your model.
    - Again, Compare the impact of different pre-processing pipelines (e.g., with and without stop words, stemming, and …) on the final performance of your model.
    - Perform error analysis on the model's prediction.
- Compare the performance of your naïve Bayes model with the achieved results from the feed-forward model. What can you conclude from the differences between the performance of the two models?
- Train a binary classification model by ignoring the instances from the neutral class (only use positive and negative instances). How does the performance differ from the multi-class classification model?

Please report all the achieved results with either model in your report document. Moreover, describe the hyper-parameters of your neural network model in the report.

Task 4: Textual similarity (15%)

In this task, you should choose 15 random negative instances, and compute the semantic textual similarity between them. Please use the average of word vectors as a distributional semantics approach at the sentence level to measure the similarity between messages. Please report the cosine similarity between randomly selected sentences in your report. (Please don't use available packages to solve the task)

Bonus Task: Transfer Learning (+10%)

As the bonus task, you should fine-tune two pre-trained models (in binary and multiclass setup) on the provided tweet dataset. You can choose between a wide range of pre-trained models (e.g., BERT, RoBERTa, and so on). It is very important to be careful about the hyper-parameters and fine-tune the models as accurately as possible without too much overwriting the current weights. Please describe your selected models, training process, and performance comparison in detail in the report.