

# PA3b: Sentiment Classification

4/26/2024

**11/20 Points**

Attempt 1



Review Feedback

4/26/2024

Attempt 1 Score:

**11/20**

View feedback

**Unlimited Attempts Allowed**

4/12/2024 to 4/27/2024

Anonymous grading: **no**

## Details

### To submit

- A Jupyter notebook (**both in .ipynb and .html**) containing the code of your solution as well as text cells describing what you have done. Make sure that the code is neat and the documentation readable.
- A report that describes your work (as a **PDF document**).

Please **remember to join a Canvas group before submitting the assignment**. You will need to repeat this procedure for every group assignment. Also please **remember to include the names of the group members** in the notebook and in the report.

### Background

Programming Assignment 3 (PA3) is divided into two parts

- PA3a is about annotating data. This is to be done **individually**.
- PA3b is the programming task, i.e., implementing a classifier that determines whether a given textual comment expresses an opinion that is neutral, positive, or negative. This is to be done in **groups of two**.

### Didactic purpose of this assignment

- to get some practical understanding of issues around crowdsourcing data and inter-annotator agreement.
- to practice several aspects of system development based on machine learning: collecting data, cleaning data, processing and selecting features, selecting and tuning a model, evaluating a model.
- to analyse results in a machine learning experiment.
- to describe the implementation, experiments, and results in a report.

# The assignment - Implement your text classification system

## The story

It is easiest to understand this assignment by considering the following scenario: You are working at a company named “Birch”, and they just decided that they want to develop a system capable of text classification in a specific domain. They already have some training data available for their domain of interest but need your expertise in developing the system.

Your group has been tasked with developing a performant system for text classification. Since the system is going to be used within Birch, it is important that you consider the requirements this puts on your work. A few examples of such requirements are:

- Is it possible to understand what you have done in the system, and why? Your colleagues should be able to understand what you have done, both to enable future developments of the system and help them trust it. Examples of things you probably do in your method and should explain are:
  - How do you represent your data as features?
  - Did you process the features in any way?
  - How did you select which learning algorithms to use?
  - Why did you make the method choices that you did?
  - Why did you not do certain things?
- For transparency you should also write a report for Birch that describes your system. Some things to think about for this are:
  - Since you have colleagues that are not as tech-savvy as you, you cannot assume that they will be able to open your Jupyter Notebooks. Therefore, you should make sure that all your important findings can be found in the report.
  - Also, you cannot assume that everyone already knows about your project beforehand, so make sure that your report is self-contained.
- Is the performance of your system optimal? It is very important for Birch that your system works well. You also have a group of colleagues who really prioritise performance, and they really like to ask questions such as “But what if you had done [something] instead, couldn’t that have worked better?”. To avoid awkward situations, it is probably best to ensure that you have an answer to such questions. Questions they typically like to ask are:
  - Could you have gained a better performance if you set the hyper-parameters to some other value?
  - Did you tune the hyper-parameters, and if so, how? Was it the best way?
  - How did you evaluate the quality of your system?
  - How well does your system compare to a trivial baseline? e. how do we know that it is actually doing something “smart”?
- Can we trust your system to always work? Your system will be deployed, so it is very important that it suddenly doesn’t break down or produces incorrect results for certain types of data. It

would at least be good to know when it works and when it does not work. Birch will be very interested in results on how risky it is to deploy your system. Examples of ways to investigate this are as follows.

- Is the training data for your system reliable? Did the annotators of the data perform their work well, or are some data samples of poor quality? You can evaluate this at large scale by e.g. looking at the consensus between the annotations.
- You will receive a dataset later on that has been annotated by three annotators, can you see any benefits of using "extra verified" datasets in your work?
- Can you say anything about the errors that the system makes? For a classification task, you may consider a [confusion matrix](#).
- What errors does the system make? What might have gone wrong in these cases?
- Does the system seem to act reasonably with respect to what features it considers to be important? (Whether this is easy to investigate depends on the type of classifier you are using.)
- Can other developers learn from and build on your work? Developers should also be able to verify themselves that your system works. Therefore, it is important that you release all the code necessary for reproducing your results. Some notes on this:
  - The code you release should be runnable.
  - It should be possible for an outsider to run your code, so instructions should be included at appropriate instances.
- Time management. You are working on this project for a limited amount of time, so you can't do everything. It is therefore important to prioritise. Ways to help Birch understand your prioritisations and learn from it are as follows:
  - Explain why you did not do some things that might have been fruitful.
  - Describe investigations that you did not have time for but would recommend the company to pursue in the future.

The better you follow these requirements, the better your work will be. Also, there are probably more things you can think of to ensure that your work provides as much value as possible to Birch.

For ideas on how to do a good project, you can also for example consider [this guide](#). It was written by Andrej Karpathy who used to be the senior director of AI at Tesla. Although the guide is for neural networks, it provides valuable insights for machine learning projects that do not use NNs as well.

## Practical information

### The Task

1. Reflect on the annotation task (PA3a). Briefly described how you experience the task. Suggests one or two way(s) to improve the task or the result of the task.
2. Explore the crowdsourced data.
  - Are the labels as expected? If so, proceed to the next task. If not, describe the issue(s) found, fix the issue(s), and describe what you did.

- Check the distribution of the data. Describe what you observed.
- Compare the crowdsourced data with the gold label data. Describe what you observed.

3. Write the code to implement a classifier that determines whether a given text expresses a neutral, positive, or negative sentiment.

- First, train the system with the crowdsourced data. Once you are happy, test your system with the test data. Report the performance.
- Next, train the system with the gold data. Test your system with the test data. Report the performance.
- Compare to the results you obtained when training the system with the crowdsourced data. Do you see any differences? If so, what could be the cause?

## The Data

You will use three datasets for this task:

- [crowdsourced\\_train.csv](#) ↓ - training dataset with crowdsourced labels from PA3a.
- [gold\\_train.csv](#) ↓ - training dataset with trusted labels (so-called "gold" labels). Contains the same tweets as in the other training set but with different labels.
- [test.csv](#) ↓ - test dataset with trusted labels. Contains different tweets than the training sets.

All three datasets are tab-separated CSV files with utf-8 encoding. They each have two columns:

- **sentiment** - the label. This will be one of "positive", "negative" or "neutral" for the datasets with gold labels.
- **text** - the tweet.

## Course restrictions

In your implementation, you are free to use any machine learning approach you think could be useful: the only restriction is that you are not allowed to use existing implementations that carry out exactly this task (that is: classifying whether a text contains a neutral, positive, or negative sentiment). You may take some inspiration from the [document classification example](#) ↓ . However, it is probably useful to try to improve over this solution. For instance, you may read more about the [TfidfVectorizer](#) and see what you can do with it.

## Report

The submitted report should be around 3–4 pages. Use [the following template](#) ↓ to write the report. It should be written as a typical technical report including sections for the introduction, method description, results, conclusion. The report should be a PDF document. Please include the names of all the students in the group.

The report should detail your implementation, your experiments and analysis. It should not be in the form of a bullet list that just goes through some discussion points. It should be a typical technical report, written in a clear and readable manner.

## Ask for help!







There will be a set of lab sessions dedicated to this assignment. We encourage you to use these sessions to help you learn as much as possible from the assignment. During the sessions you are free to ask about system issues, how to understand the task, inspiration on methods, the grading etc.

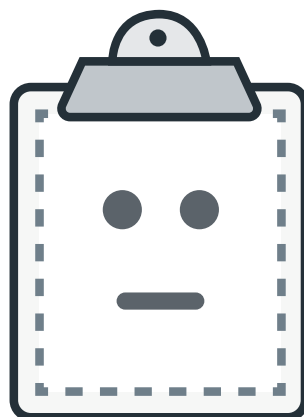
You can also email questions to Nicolas, Mehrdad, or Selpi if it is not possible to ask them during the lab sessions.

## Grading

Your grade will be based on mainly these three things:

1. Your report. Is it insightful and lives up to professional standards of technical writing, including decent clarity, spelling, grammar and structure? Also, does it describe all your work?
2. Your analysis on the data. Are you aware of the issues or characteristics related to the data that might affect the performance of your system?
3. Your method. Are your technical solutions justified and did you build an optimal system for text classification? Did you verify the performance of your system? How much did you follow the requirements listed above on transparency, performance, reliability, reproducibility and time management? Did you analyse the benefits of double annotated data?
4. Your code. Is it well-implemented and easy to use for reproduction of your results?

File name		Size	
	<a href="#">PA3b.html</a>	294 KB	
	<a href="#">PA3b.ipynb</a>	10.7 KB	
	<a href="#">PA3b__Sen...eport.pdf</a>	94 KB	



Preview Unavailable

PA3b.html



Download