

PA3c: Text classification (Part C)

Due 25 Apr by 23:59 **Points** 20 **Submitting** a file upload
File types ipynb, pdf, and html **Available** 12 Apr at 12:00 - 31 May at 0:30

This assignment was locked 31 May at 0:30.

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.

Background

Programming Assignment 3 (PA3) is divided into three parts

- PA3a (also called Part A) is about crowdsourcing data. This is to be done **individually**.
- PA3b (also called Part B) is about improving annotation quality. This is to be done **individually**.
- PA3c (also called Part C) is the programming task, i.e., implementing a text classifier that determines whether a given textual comment expresses an opinion that is positive or negative towards **restaurant reviews**. 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 hyperparameters to some other value?
 - Did you tune the hyperparameters, 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 of the annotators.)

- 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, now director of AI and Autopilot Vision at Tesla. The guide is for neural networks, while it probably can provide valuable insights also for machine learning projects that do not use NNs.

Practical information


The task

Write the code to implement a classifier that determines whether a given comment expresses a positive or negative review.


Data

We give you access to an initial training dataset and test dataset developed from the preceding year. Note that the training dataset has been annotated by two annotators, made up of previous course participants, and that you cannot be 100% certain that the annotations always are correct. Later on, you will receive the training dataset that has been annotated by three annotators (including your work on PA3b) that you should make use of. Note that your results may change when you switch datasets. Links to the data will be listed below as the assignment moves along.

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 positive or negative review). You may take some inspiration from the [document classification examples](#) listed under the 2023-03-30 Lecture. 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.

Code

The code should be a Jupyter notebook (submit both the ipynb and the html versions).

Submission

Please use the Canvas page to submit your solution (the code and the report).

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 mail questions to the responsible teaching assistant (Lovisa Hagström, lovhag@chalmers.se) 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 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 triple annotated data (from the PA3b data you collected)?
3. Your code. Is it well-implemented and easy to use for reproduction of your results?

Data

Please check this page regularly, since we will post the new datasets after they have been collected.

2023-04-12

Double annotated train data

[PA3_train.tsv](#) ↓

The data is stored in a text file consisting of tab-separated columns, where the first column contains the output labels (1 for positive review, 0 for negative review) and the second contains the example. The double annotations are separated by a slash (/). In some cases, as in the third example below, annotators may disagree.

```
1/1      Best ever yet and I think it will take time before I stumble upon anything better. Affordable V
ery tasty Perfect service Nice Interior.
1/1      Their sushi came beautifully presented & fresh as well.
0/-1     It has a bad placement and is very small. Even though the local is cosy I would have wanted to
see it a bit bigger.
0/0      Not so cousy restaurant, if you book a table you want to sit in the first part so you can look
at the chefs when making the food. A bit slow service at times.
```

Test data

[PA3_test.tsv](#) ↓

For this data we only have single annotations, as seen in the example below. As usual, the development of your system should not use the test set in any way, and you should only compute the test-set score after finalising your system.

```
1      Best ever yet and I think it will take time before I stumble upon anything better. Affordable Ver
y tasty Perfect service Nice Interior.
1      Their sushi came beautifully presented & fresh as well.
0      It has a bad placement and is very small. Even though the local is cosy I would have wanted to se
e it a bit bigger.
0      Not so cousy restaurant, if you book a table you want to sit in the first part so you can look at
the chefs when making the food. A bit slow service at times.
```

2023-04-20

Triple annotated train data

[PA3_train_final.tsv](#) ↓

Differences between this training dataset and the one shared previously:

- There are now three label annotations instead of two. The first value of the three annotated labels comes from the PA3b assignment, so for label "0/-1/1" you have that 0 is the more carefully made labelling based on the two already existing labels. You can thus see the first annotated label as the more correct one.
- The new dataset contains more samples (7698 instead of 7018).

With this data you now have the additional opportunity to further reason about data quality and its impact on your classifier. You keep the old test set.

No new datasets should be released from now on.