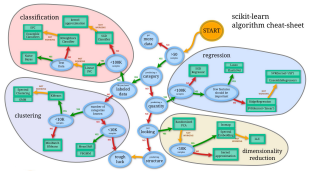
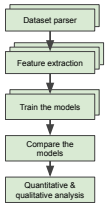
Assignment 2:

Building a Document Classification Pipeline





Fredrik Wahlberg
fredrik.wahlberg@lingfil.uu.se
5LN708: Machine Learning in Natural Language Processing

Summary

The scenario is as follows: You are given a text collection of news articles. Your task is to create a classifier for predicting the topics of unseen new articles. Design choices should be explained in a short report together with your final notebook.

Each document in the dataset is labeled with zero or more topic tags. You should implement both a probabilistic and a non-probabilistic classifier, explore feature encodings and tokenization schemes, and try to follow best practices for generalisability. Your results should be compared to some baseline and by using the 20 newsgroups dataset in sklearn.

While assignment one focussed on theoretical understanding, this assignment will focus on a more realistic task. Your main tools for constructing the pipeline should be numpy and sklearn. You can, however, use libraries like spacy and nltk for smaller tasks (e.g. tokenization, visualisation etc).

While you should explore combinations of pipeline components, combinatorial search should not replace reasoning about expected performance.

Timeline

Draft (c. five workdays)

Peer-review (c. two workdays)

Submission (c. three workdays)

Timeline

Write a draft of your report and pipeline

Everything should be in place before the draft submission for running the pipeline and critiquing your design choices. You are, however, expected to improve your report and pipeline after having received feedback.

Finish your pipeline

Read your reviews to find ways of improving your work.

April 22 — April 29 — 3 May — 6 May

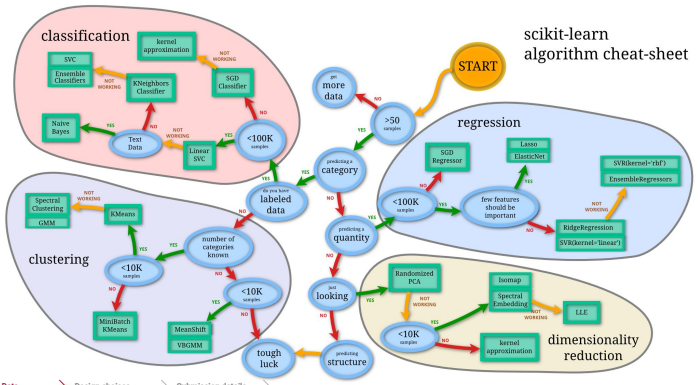
Peer-review

Each student will critique three submissions.

Final submission

This is your final work on which you will be graded.

Scikit Learn



Datasets

Reuters-21578


```
<!DOCTYPE lewis SYSTEM "lewis.dtd">
<REUTERS TOPICS="YES" LEWISSPLIT="TRAIN" CGISPLIT="TRAINING-SET"
OLDID="5544" NEWID="1">
<DATE>26-FEB-1987 15:01:01.79</DATE>
<TOPICS><D>cocoa</D></TOPICS>
<PLACES><D>el-salvador</D><D>usa</D><D>uruguay</D></PLACES>
<PEOPLE></PEOPLE>
<ORGS></ORGS>
<EXCHANGES></EXCHANGES>
<COMPANIES></COMPANIES>
<UNKNOWN>
&#5;&#5;&#5;C T
&#22;&#22;&#1;f0704&#31;reute
u f BC-BAHIA-COCOA-REVIEW   02-26 0105</UNKNOWN>
<TEXT>&#2;
<TITLE>BAHIA COCOA REVIEW</TITLE>
<DATELINE>    SALVADOR, Feb 26 - </DATELINE><BODY>Showers
continued throughout the week in
the Bahia cocoa zone, alleviating the drought since early
January and improving prospects for the coming temporao,
although normal humidity levels have not been restored,
Comissaria Smith said in its weekly review.
    The dry period means the temporao will be late this year.
    Arrivals for the week ended February 22 were 155,221 bags
of 60 kilos making a cumulative total for the season of 5.93
mln against 5.81 at the same stage last year.  [...]
```


20 newsgroups

- Loader in sklearn
- Text from 18000 newsgroups posts
- 20 topics

```
['alt.atheism',  'comp.graphics',
'comp.os.ms-windows.misc',  'comp.sys.ibm.pc.hardware',
'comp.sys.mac.hardware',  'comp.windows.x',
'misc.forsale',  'rec.autos',  'rec.motorcycles',
'rec.sport.baseball',  'rec.sport.hockey',  'sci.crypt',
'sci.electronics',  'sci.med',  'sci.space',
'soc.religion.christian',  'talk.politics.guns',
'talk.politics.mideast',  'talk.politics.misc',
'talk.religion.misc']
```

You will likely have issues with memory, depending on choice of classifier.

Tools

Implementation recommendations

Regular expressions can parse xml fast and easy reuters. There are great web tools for testing your regexes.

Red thread testing is when you test a pipeline from beginning to end while not focusing on the quality of each component. You can then go back and improve.

Write smaller pieces of code and test these with asserts. This is called *unit testing*.

Also, remember that ocular inspection is your friend.

Console commands

Anything that is a standard linux/unix terminal command (e.g. tar, gunzip, head, grep).

External packages

standard packages
numpy
sklearn
nltk
spacy

The manuals for the above are filled with code examples.

Some design choices

Data

- How to parse the two datasets for classification *(hint: use sklearn and re)*
- Tokenization, stopwords

Feature extraction

- Encoding (BOW, tf-idf, ...)
- Grouping (n-gram, ...)

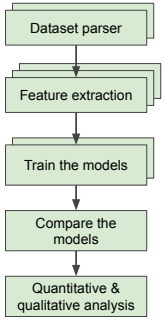
Classifier

- Classifiers to use (one probabilistic, one non-probabilistic)
- Hyperparameters
- Data splits / Generalizability
- Training feasibility (memory, time etc)

Evaluation

- Metrics

Another hint: Keep things simple if you don't hate time off. Think before you code, but not too much.



Submission details

G requirements

Implement pipelines comparing the performance of two classifiers (probabilistic, non-probabilistic) on the two given datasets.

Take extra care thinking about

- Model inductive bias
- Hyperparameter choice
- Training and Test sets
- Evaluation metrics

You should hand in a report (c. 500-2500 words), with your design choices and results together with a notebook reproducing your principal results.

VG requirements

In addition, you should work more on:

- Generalisability
- Statistically significant differences in performance
- Some visualisation of your results
- Inductive biases for all parts of the pipelines (not only the model)
- Overfitting
- Hyperparameter search

A VG report is not necessarily longer, but necessarily more informative.

Note that no one likes reading obvious filler text. I'd rather read a short informative report than a long literary masterpiece.

Double blind peer review

Your assignments are handed in individually and should be implemented as such. However, we will use peer-review in this assignment where you can give anonymous criticism (just like we do for academic papers).

A good review gives constructive criticism while leaving room for the author to make their own choices. You should point out flaws, good practice, well thought out solutions, unclear explanations etc.

For double *blind* peer-review, it is a hard requirement that no identifying information is left in your final submission.

Examples

The classifiers in your pipelines are well implement, but perhaps you should look into tokenization more. I believe that you haven't removed stopwords (as we did in NLP). Do you have a good reason for not doing this?

Paragraph 5 feels like filler text. What is it that you want to say?

You are using a train/test split but your code is still training your hyperparameters on the training set. Maybe your should use a validation set as the lecturer is always going on about them. Might give you a higher grade.

I think you forgot to comment on the inductive biases of your naive bayes classifier.

I believe you are confusing kNN with decision trees in paragraph 3. I think that lecture x and book chapter y deals with this issue.

As the grading scheme in this course is not relative (i.e. no grading curve), you have nothing to lose by giving a productive review.