Computational Communication Science 2 Week 7 - Lecture »Validation (in Supervised Machine Learning)«

Marthe Möller

a.m.moller@uva.nl

May 15, 2023

Digital Society Minor, University of Amsterdam

Today

Recap

Recap

Last week, we discussed:

- Supervised Machine Leaning (SML)
- The principles behind SML
- The steps of SML
- Some commonly used ML models

At home, you:

- Got some hands-on experience SML (week 6 exercises)

Recap

Today, we:

- Review your first SML experience
- Take a deep dive into validating SML-models

Recap

Last week, you practiced with code that:

- Read in some data (Q1)
- Split the data into a train and a test set (Q2)
- Set up a Count vectorizer (Q3)
- Trained a Naïve Bayes model with the count vectorizer (Q4)
- Requested some metrics for validation (Q5)

Recap Q1

```
import csv
from collections import Counter
import matplotlib.pyplot as plt

file = "hatespeech_text_label_vote_RESTRICTED_100K.csv"
tweets = []
labels = []

with open(file) as fi:
  data = csv.reader(fi, delimiter='\t')
  for row in data:
   tweets.append(row[0])
   labels.append(row[1])

print(len(tweets) == len(labels)) # there should be just as many tweets
     as there are labels

Counter(labels)
plt.bar(Counter(labels).keys(), Counter(labels).values())
```

Recap Q2

Split the dataset:

```
from sklearn.model_selection import train_test_split

tweets_train, tweets_test, y_train, y_test = train_test_split(tweets,
    labels, test_size=0.2, random_state=42)
```

Recap Q3

What happens here?

```
from sklearn.feature_extraction.text import (CountVectorizer)

countvectorizer = CountVectorizer(stop_words="english")
X_train = countvectorizer.fit_transform(tweets_train)
X_test = countvectorizer.transform(tweets_test)
```

Recap Q4

The actual SML part (yes, truly, it is three lines of code!):

```
nb = MultinomialNB()
nb.fit(X_train, y_train)
y_pred = nb.predict(X_test)
```

Recap Q5

You can check what was created:

```
nb = MultinomialNB()
nb.fit(X_train, y_train)
y_pred = nb.predict(X_test)

print(y_pred[:10])
```

```
['normal' 'normal' 'normal' 'normal' 'spam' 'normal' 'normal' '
    normal' 'abusive' 'normal']
```

Recap Q5

Classification report:

```
from sklearn.metrics import classification_report

print(classification_report(y_test, y_pred))
```

Up next

Classification report: validate your classifier.

More about this today!

Validating models

Validation

Validation: When we assess the performance of a classifier.

Or when we try to answer the question: "How well does the classifier work?"

Validation

Validation: When we assess the performance of a classifier.
Or when we try to answer the question: "How well does the classifier work?"

Validation

What criteria should we use to decide on this?

Entirely context specific!

Validation

What criteria should we use to decide on this?

Entirely context specific!

Validation

Compare different goals for using SML:

- To automatically decide what Instagram users should see an advertisement
- To automatically remove spam from Twitter feed

Would you use the same criterion in both cases to determine how well a classifier works? Why (not)?

Validation

There are various evaluation metrics available for machine learning.
In scikit-learn, they are presented by ways of a classification report!

Zooming out

So far, we:

- Reviewed the exercise and the basic steps of SML
- Talked about what validation is

Next, we will talk about:

- Some commonly used validation metrics
- Input for SML
- Finding the best classifier

Validation metrics

Precision

Precision quantifies the number of positive class predictions that actually belong to the positive cases.

OR: How much of what we found is actually correct?

Precision

Precision quantifies the number of positive class predictions that actually belong to the positive cases.
OR: How much of what we found is actually correct?

Precision

Precision quantifies the number of positive class predictions that actually belong to the positive cases.
OR: How much of what we found is actually correct?

Compare different goals for using SML in exercise 6:

- To automatically decide what Instagram users should see an advertisement
- To automatically remove spam from Twitter feed

Recall

Recall quantifies the number of positive class prediction made out of all positive examples in the dataset.

OR: How many of the cases that we wanted to find did we actually find?

Recall

Recall quantifies the number of positive class prediction made out of all positive examples in the dataset.
OR: How many of the cases that we wanted to find did we actually find?

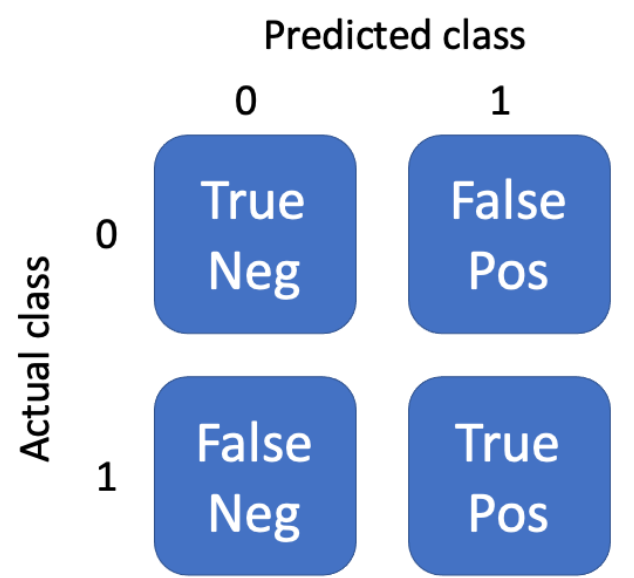
Recall

Recall quantifies the number of positive class prediction made out of all positive examples in the dataset.
OR: How many of the cases that we wanted to find did we actually find?

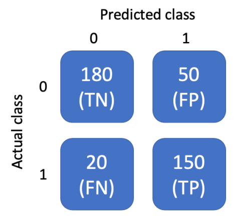
Compare different goals for using SML in exercise 6:

- To automatically decide what Instagram users should see an advertisement
- To automatically remove spam from Twitter feed

Precision and Recall



Precision and Recall



Precision is calculated as: $\frac{\text{TP}}{\text{TP+FP}}$
In this example $\frac{150}{150+50}$ which is 0.75
Recall is calculated as $\frac{\text{TP}}{\text{TP+FN}}$
In this example $\frac{150}{150+20}$ which is 0.88

What does this look like in code?

Let's ask for a confusion matrix:

```
from sklearn.metrics import confusion_matrix

y_test = [0, 1, 1, 1, 0]
y_pred = [0, 0, 1, 1, 1]

print(confusion_matrix(y_test, y_pred))
```

```
[[1  1 ]
[ 1  2]]
```

The classification report

Let's get some metrics for validation:

```
from sklearn.metrics import classification_report
print(classification_report(y_test, y_pred))
```

	precision	recall	f1-score	support
abusive	0.81	0.88	0.85	5369
hateful	0.83	0.05	0.10	966
normal	0.78	0.93	0.85	10848
spam	0.67	0.30	0.41	2817
accuracy			0.78	20000
macro avg	0.77	0.54	0.55	20000
weighted avg	0.78	0.78	0.75	20000

F_1-score

But wait...

Compare different goals for using SML:

- To automatically decide what Instagram users should see an advertisement
- To automatically remove spam from Twitter feed

Such information was not available in the exercise for week 6!

F_1-score

F_1-score: The harmonic mean of precision and recall. (Weighted average of precision and recall)

$$F_1\text{-score} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

Accuracy

	precision	recall	f1-score	support
abusive	0.81	0.88	0.85	5369
hateful	0.83	0.05	0.10	966
normal	0.78	0.93	0.85	10848
spam	0.67	0.30	0.41	2817
accuracy			0.78	20000
macro avg	0.77	0.54	0.55	20000
weighted avg	0.78	0.78	0.75	20000

Accuracy

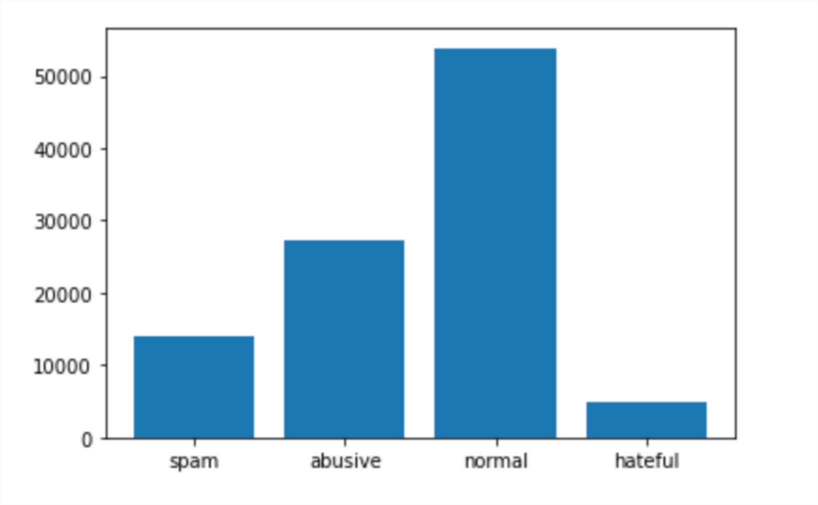
Accuracy: In which percentage of all cases was our classifier right?

Accuracy

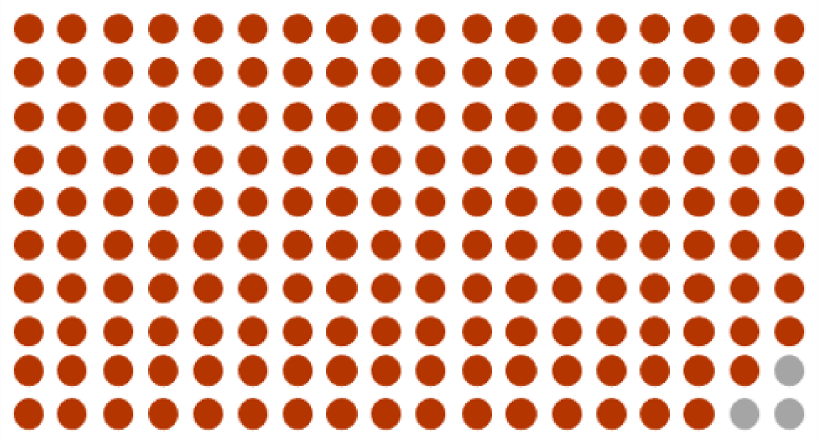
Class distribution: The number of examples that belong to each class.

Imbalanced classification: A predictive modeling problem where the distribution of examples across the classes within a training dataset is not equal.

Accuracy


50000
40000
30000
20000
10000
0
spam
abusive
normal
hateful

Accuracy



Majority class (red dots) vs. minority class (grey dots)

Accuracy

Always check how your cases are distributed across the labels.

Beyond validation metrics

Other ways to check the performance of classifiers:

- Compare the results of the automated coding to the hand coding results
- Check the identified features
- ...

Validation: Best practices

Decide what metric is best to use beforehand.

Consider:

- Is class imbalance an issue?
- What will the classifier be used for?

The latter can bring you back to the question: To SML or not to SML?

Validation: Best practices

SML suitability depends on:

- How hard/easy it is to translate the decision proces for classification into straight-forward rules
- How much data there are to classify
- How much room there is for errors

Consider these matters as you decide (a) whether or not to use SML, and (b) what performance metrics to use.

Validation: Best practices

In last week's lecture, we saw that you can train many different classifiers.

Amongst other, classifiers can differ based on:

- The vectorizer that is used on the data (i.e., count vectorizer or tf-idf vectorizer)
- The underlying model (e.g., Naïve Bayes, Logistic Regression, Decision Trees, SVM, etc.)

How do you know what classifier is best beforehand?

Validation: Best practices

In last week's lecture, we saw that you can train many different classifiers.

Amongst other, classifiers can differ based on:

- The vectorizer that is used on the data (i.e., count vectorizer or tf-idf vectorizer)
- The underlying model (e.g., Naïve Bayes, Logistic Regression, Decision Trees, SVM, etc.)

How do you know what classifier is best beforehand?

Selecting a classifier

You don't!

Typically, various classifiers are trained and their performance is compared.

The best performing classifier is then selected and used to annotate more/new data.

Selecting a classifier

You don't!

Typically, various classifiers are trained and their performance is compared.

The best performing classifier is then selected and used to annotate more/new data.

Selecting a classifier

You don't!

Typically, various classifiers are trained and their performance is compared.

The best performing classifier is then selected and used to annotate more/new data.

Zooming out

So far, we:

- Reviewed the exercise and the basic steps of SML
- Talked about what validation is
- Discussed some commonly used validation metrics

Next, we will talk about:

- Input for SML

About input for SML

Bad input

Poor quality input can cause:

- Your classifier not being able to identify specific categories
- Your classifier to perform badly overall
- Your classifer to pick up implicit bias

Bad input

What makes input (training data) bad input?

Bad input

What can be done to solve these issues?

Poor quality input can cause:

- Your classifier not being able to identify specific categories
- Your classifier to perform badly overall
- Your classifer to pick up implicit bias

Creating good input

You can create good input:

- Avoid class imbalance: increase your sample (add rare cases), or random selection
- Make sure the hand coding is reliable (check Krippendorf's Alpha, add coder training)
- Make sure to develop a clear and objective codebook

The limits of your input

User case*:

A classifier was trained to distinguish between YouTube comments that address the video that they accompany, and YouTube comments that do not. The classifier was trained on a training set consisting of $>$ 16.000 comments written in response to music videos. The classifier performed well, with high scores on all evaluation metrics. The researchers now want to use their classifier to automatically label a new set of 10.000 YouTube comments written in response to vlogs. Can they?

*Based on: Möller et al., in press

The limits of your input

User case*:

A classifier was trained to distinguish between YouTube comments that address the video that they accompany, and YouTube comments that do not. The classifier was trained on a training set consisting of > 16.000 comments written in response to music videos. The classifier performed well, with high scores on all evaluation metrics. The researchers now want to use their classifier to automatically label a new set of 10.000 YouTube comments written in response to vlogs. Can they?

*Based on: Möller et al., in press

The limits of your input

User case*:

A classifier was trained to distinguish between YouTube comments that address the video that they accompany, and YouTube comments that do not. The classifier was trained on a training set consisting of > 16.000 comments written in response to music videos. The classifier performed well, with high scores on all evaluation metrics. The researchers now want to use their classifier to automatically label a new set of 10.000 YouTube comments written in response to vlogs. Can they?

*Based on: Möller et al., in press

The limits of your input

User case*:

A classifier was trained to distinguish between YouTube comments that address the video that they accompany, and YouTube comments that do not. The classifier was trained on a training set consisting of > 16.000 comments written in response to music videos. The classifier performed well, with high scores on all evaluation metrics. The researchers now want to use their classifier to automatically label a new set of 10.000 YouTube comments written in response to vlogs. Can they?

*Based on: Möller et al., in press

The limits of your input

User case*:
A classifier was trained to distinguish between YouTube comments that address the video that they accompany, and YouTube comments that do not. The classifier was trained on a training set consisting of > 16.000 comments written in response to music videos. The classifier performed well, with high scores on all evaluation metrics. The researchers now want to use their classifier to automatically label a new set of 10.000 YouTube comments written in response to vlogs. Can they?

*Based on: Möller et al., in press

The limits of your input

Classifiers can only be used to automatically annotate data from the same population as the data that they were trained on.

Zooming out

Today, we:

- Reviewed the exercise and the basic steps of SML
- Talked about what validation is
- Discussed some commonly used validation metrics
- Talked about input for SML

Zooming out

Tomorrow and this week, you will:

- Set up multiple different classifiers
- Validate those classifiers
- Select the best performing classifier

Work on the tutorial exercises for this week.