Computational Communication Science 2
Week 8 - Lecture
»A Deep Dive into Supervised Machine Learning«

Marthe Möller
Anne Kroon

a.m.moller@uva.nl, @marthemoller
a.c.kroon@uva.nl, @annekroon

May, 2022

Digital Society Minor, University of Amsterdam

Today

Recap

More about validation

Cross-validation

SML: Strenghts and Challenges

Looking back and ahead

Recap

Recap

Last week, we talked about:

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

At home, you:

- Got some hands-on experience SML (homework assignment)

Recap

Today, we:

- Your first SML experience
- Take a deep dive into validating SML-models
- Take a critical look at SML

Recap

This week, you practiced with code that:

- Imported all required packages and modules needed to run the script (Q1)
- Read in some data (Q2)
- Set up a Count vectorizer (Q3)
- Trained a Naïve Bayes model with the count vectorizer and requested some metrics for validation (Q4)
- Set up a tf-idf vectorizer, used it in a LogRegression model and requested some metrics for validation (Q5)

Recap Q1

```
1  import csv
2  from collections import Counter
3  import matplotlib.pyplot as plt
4  from sklearn.feature_extraction.text import (CountVectorizer,
       TfidfVectorizer)
5  from sklearn.naive_bayes import MultinomialNB
6  from sklearn.metrics import accuracy_score
7  from sklearn.linear_model import (LogisticRegression)
8  from sklearn.metrics import confusion_matrix,classification_report
9  from sklearn.pipeline import (make_pipeline, Pipeline)
10 from sklearn.model_selection import GridSearchCV
11 import pickle
12 import joblib
```

Why import all modules at the start of a Python Script?

Recap Q2

Read in the ingredients for SML (dataset is already split):

```
test = "test.txt"
train = "train.txt"

texts_test = []
labels_test = []

with open(test) as fi:
    data = csv.reader(fi, delimiter=';')
    for row in data:
        texts_test.append(row[0])
        labels_test.append(row[1])
```

Mind: Choose the correct delimiter (e.g., ';', ',', '\t').

IndexError: list index out of range' often pops up if you choose the wrong one.

Recap Q2

What happens here?

```
len(texts_test) == len(labels_test)
len(texts_train) == len(labels_train)
```

```
TRUE
```

.

Recap Q2

Why would you do this?

```
1 plt.bar(Counter(labels_train).keys(), Counter(labels_train).values())
```

Recap Q3

Remember, the computer can't read words! We need to transform the texts and labels into vectorizers.

```
countvectorizer = CountVectorizer(stop_words="english")
X_train = countvectorizer.fit_transform(texts_train)
X_test = countvectorizer.transform(texts_test)
```

When fit_transform and when transform?

Recap Q4

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

```
1 nb = MultinomialNB()
2 nb.fit(X_train, labels_train)
3 y_pred = nb.predict(X_test)
```

No output?

Recap Q4

You can check what was created:

```
1 print(y_pred[:10])
```

```
1 ['sadness' 'sadness' 'sadness' 'joy' 'sadness' 'fear' 'sadness' '
      joy' 'joy' 'sadness']
```

Recap Q4

```
accuracy = accuracy_score(labels_test, y_pred)
print(accuracy)
```

```
0.7945
```

Recap Q5

Repeat with a different vectorizer and a different SML model!

```
tfidfvectorizer = TfidfVectorizer(stop_words="english")
X_train = tfidfvectorizer.fit_transform(texts_train)
X_test = tfidfvectorizer.transform(texts_test)

logres = LogisticRegression()
logres.fit(X_train, labels_train)
y_pred = logres.predict(X_test)

print(classification_report(labels_test, y_pred))
```

Up next

In the remainder of the exercise, you will:

- Write down code to compare different vectorizers/models in an effective way (Q6)
- Learn about hyperparameters in vectorizers (Q7)
- Practice with cross-validation and gridsearch (Q8 & Q9)
- Saving vectorizers and classifiers (Q10)

Cross-validation? Hyperparameters? Gridsearch?

More about validation

Finding the best performing model

We talked about validating models based on various metrics, such as accuracy, F1-score, precision, recall...

What is the most important metric when deciding what model is best?

Finding the best performing model

		Predicted	
		Negative	Positive
Actual	Negative	998	0
	Positive	1	1

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

Finding the best performing model

		Predicted	
		Relevant	Not relevant
Actual	Relevant	150	20
	Not relevant	50	180

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

Here: 0.88

Finding the best performing model

		Predicted	
		Relevant	Not relevant
Actual	Relevant	150	20
	Not relevant	50	180

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

Here: 0.75

Finding the best performing model

What if it is hard to say which one is most important and you want to find a balance between precision and recall?

F1- score!

Cross-validation

Cross-validation

Overfitting: When a model fits *exactly* against the data.

When we calculate the metrics discussed above for multiple models on the same test dataset, we run the risk of overfitting on the test data.

Cross-validation

Potential solution: Split the dataset into three smaller sets. A training dataset, a validation dataset and a test dataset.

However, this requires us to have a very large labeled dataset. In reality, this is not always the case!

Cross-validation

Cross-validation: A resampling procedure to evaluate ML models on a limited data sample.

k-fold cross-validation, where *k* refers to the number of groups or folds in which a sample is split.

k-fold cross-validation

k-fold cross-validation step by step:

1. Shuffle the data
2. Split the data into k folds (groups)
3. For each unique group
   - 3.1 Take the group as a test dataset
   - 3.2 Take the remaining groups as one training dataset
   - 3.3 Fit a model on the training set and evaluate it on the test set
   - 3.4 Retain the evaluation score and discard the model
4. Summarize the evaluation scores to assess the model

Cross-validation

Cross-validation is often used to compare many different model specifications, for example to find the best hyperparameters.

Hyperparameters: Parameters of the model that are not estimated from the data.

To do this, the Grid Search algorithm is often used.

Grid Search

The GridSearchCV module (scikit-learn) searches for the best values of specified parameters using cross-validation.

The outcome is the optimal combination of one or more parameters.

Zooming out

We talked about:

- Your experience with SML so far
- Cross-validation and grid search

Next, we will talk about:

- Strengths and challenges associated to SML

SML: Strenghts and Challenges

Strengths and Challenges

Strengths?

Challenges?

Strengths and Challenges

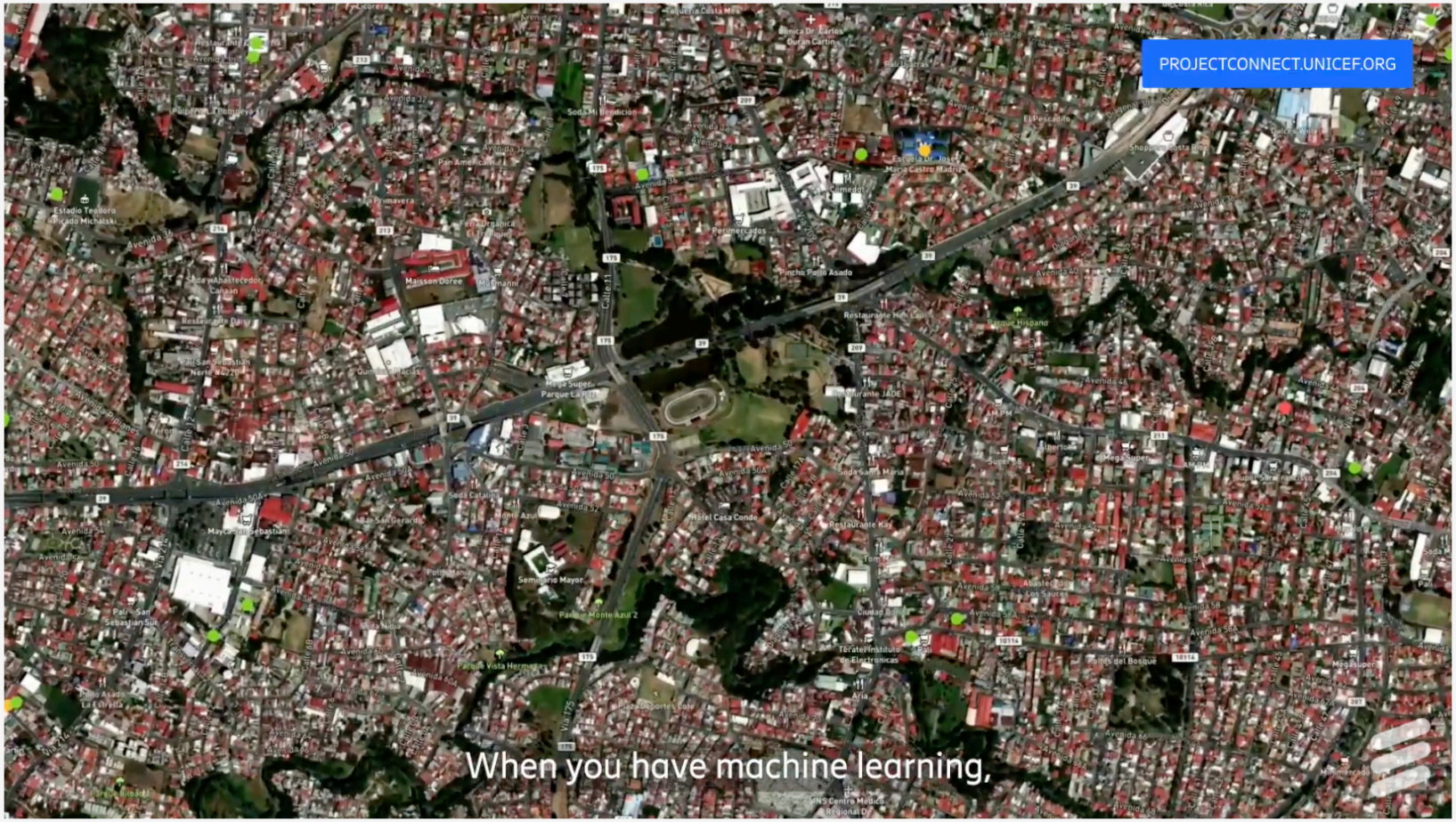
Strengths:

- Easier to code large datasets
- Enhances replicable research
- Easier to study "natural" human behavior

Challenges:

- Resource constraints
- Ethical considerations
- Criticism required (see next slide)

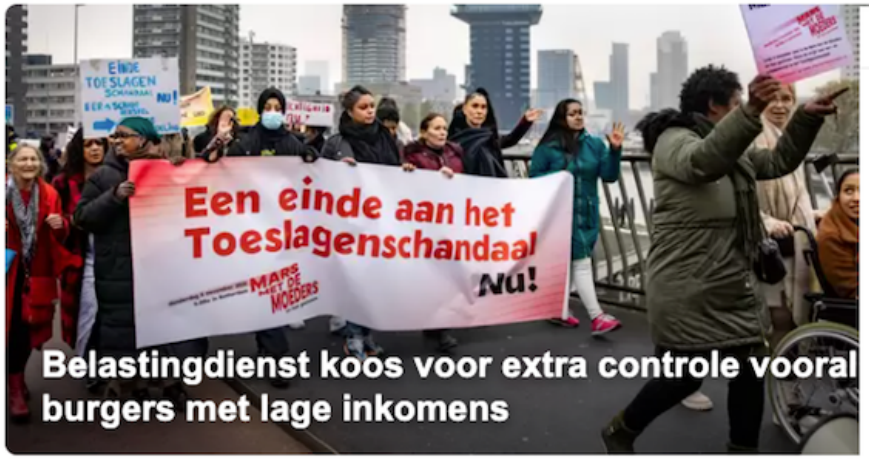
Strengths and Challenges



#ITU #Giga #GigaConnect

Technology to map schools and their connectivity

Strengths and Challenges



Belastingdienst koos voor extra controle vooral burgers met lage inkomens

23 november 2021 00:47
Laatste update: 1 dag geleden

761 NUjij-reacties

Het systeem van de Belastingdienst koos ervoor om de kinderopvangtoeslag vooral bij mensen met een laag inkomen extra te controleren. Dat heeft de fiscus toegegeven in antwoord op vragen van *Trouw* en *RTL Nieuws*.

Trouw

Toeslagen

Belastingdienst ging vooral achter lage inkomens aan

Om toeslagen te controleren op fouten en fraude gebruikte de Belastingdienst een zelflerend algoritme. Dat selecteerde vooral lage inkomens voor controle.

Jan Kleinnijenhuis 22 november 2021

De Belastingdienst heeft jarenlang specifiek burgers met een laag inkomen geselecteerd voor extra controle op

Fraudejacht bij Toeslagen

Belastingdienst controleerde extra bij lage inkomens in jacht op fraude

22 november 2021 22:59

Looking back and ahead

Zooming out

We talked about:

- The principles behind SML
- Some frequently used SML models
- Validating SML classifiers
- SML in practice
- Cross-validation and grid search
- Strengths and challenges associated to SML

Zooming out

This week's tutorial:

- Hands-on approach to take a further look into the machine learning process
- Questions about prior tutorial exercises

Looking back

You started with the basics (e.g., what is a list, how to save data, write a loop).
In two months, you learned:

- How to read in data
- How to preprocess data
- How transform text into data that a computer can understand
- How to compare texts to provide a recommendation
- How to analyze text to classify it automatically

Looking at the very near future

One last grade for this course, the take-home exam.

- The take-home exam will be published on 30 May (9 am)
- The deadline for submitting the take-home exam is on 2 June (9 am)
- The exam covers materials discussed in weeks 1 through 7 (but you may of course also use anything you learned in week 8, this is not required, however)

Looking at the near future

Final course of the minor: the research project!

- You will combine your programming skills with your skills as a researcher
- Run a research project about ComScience using the materials you created for the group assignment in this course
- More information follows in the first meeting of the research project

Looking at the future

CCS-1 and CCS-2: An introduction to coding. You can continue to learn and work with Python.
No course materials and instructors to help you out, but there are a lot of resources online!
Our tips:

- Error message? Google is your best friend!
- Check out the documentation of any module that you use to learn how it works
- Check out pubs using the method you are interested in. Often, they publish their python scripts (e.g., Meppelink et al., 2021)

Looking at the future

Thank you for the past weeks and enjoy the research project!