UNIVERSITY OF AMSTERDAM

Python for Teachers:

Machine Learning

Day 5 – Morning Session

Dr. A. Marthe Möller

UNIVERSITY OF AMSTERDAM

Nice to meet you!

Dr. A. Marthe Möller

Assistant Professor of Entertainment Communication

My research:

- Entertainment experiences in response to online content
- Automated Content Analyses of YouTube

My teaching:

- Method courses: CCS-2, theses
- First time teaching this workshop – please bear with me! ☺

Agenda

Time:	Topic:
09:30 – 11:00	Basics of (supervised) machine learning
11:00 – 12:00	Exercise 1
12:00 – 13:00	Break
13:00 – 13:30	Recap on Exercise 1
13:30 – 14:30	Evaluating models
14:30 – 15:30	Exercise 2
15:30 – 16:00	Recap on Exercise 2
16:00 – 16:30	Closure

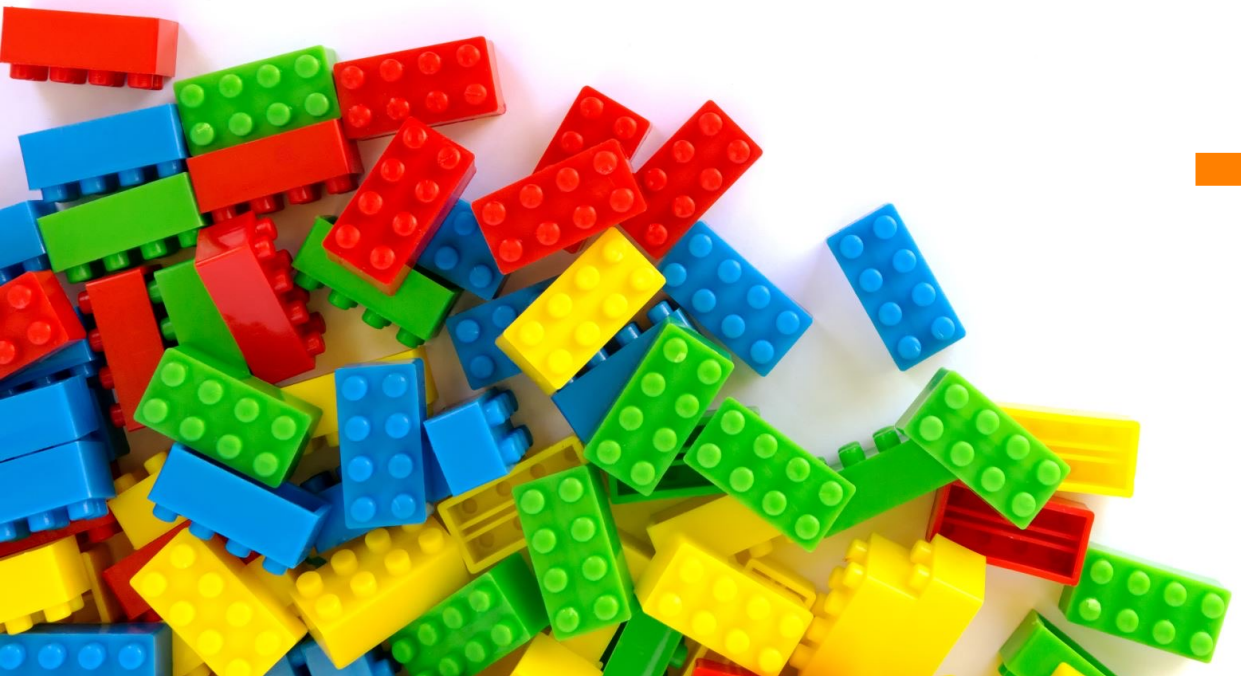
What are we talking about?!

- What you know already
- How today relates to the past weeks
- Rule-based text classifications
- Automated text classifications: SML
- The principles behind SML
- Practical applications of SML
- Some commonly used SML models

What you now know

- Basics of Python (different datatypes, functions, loops)
- Pandas
- Modules, packages, and libraries
- Getting data: web scraping, API's
- Working with text as data (NLP, Regular Expressions, BOW)

Today: Machine Learning



Start from something we know: Content analysis

- Manual analysis of text
- Coders use a codebook to analyze some text

Automated content analysis

- Bag of Words Approach
- Computer loops over words and counts each word

- Especially helpful when working with big data sets

Rule-based Text Classification

Text classification: To assign a label to a text

For example, we can distinguish between:

- Newspaper articles about sports vs. economics
- Reliable vs. unreliable information about vaccination
- Webpages about holding companies vs. financing companies
- Positive vs. negative movie reviews

RQ: How prevalent is flaming on Twitter?

A rule-based approach:

- Create a list of all the swearwords that exist
- For each tweet in the dataset, use the list to count the number of swearwords
- If a tweet contains X number of swearwords, label it as flaming

We can add nuance by adding rules

For example, in sentiment analyses, we can include a rule telling the computer what to do in case of negation of modifiers:

“This movie is really good!”

“This movie is really not good!”

Rule-based approaches

When you simply want to count the occurrence of specific words, a rule-based approach will be quick, cheap, easy, and transparent – perfect!

Challenges of rule-based approaches

- Not suitable to analyze latent or abstract variables
- You must know all the categories beforehand
- You must know and be able to express all the rules

When it is easy for humans to decide what class data belongs to, but we struggle to translate our decision process into straight-forward rules, we are likely to be better of using a form of automated text classification.

Van Atteveldt et al., 2022

What are we talking about?!

- ~~What you know already~~
- ~~How today relates to the past weeks~~
- ~~Rule-based text classifications~~
- Automated text classifications: SML
- The principles behind SML
- Practical applications of SML
- Some commonly used SML models

Hard for a computer, easy for us!

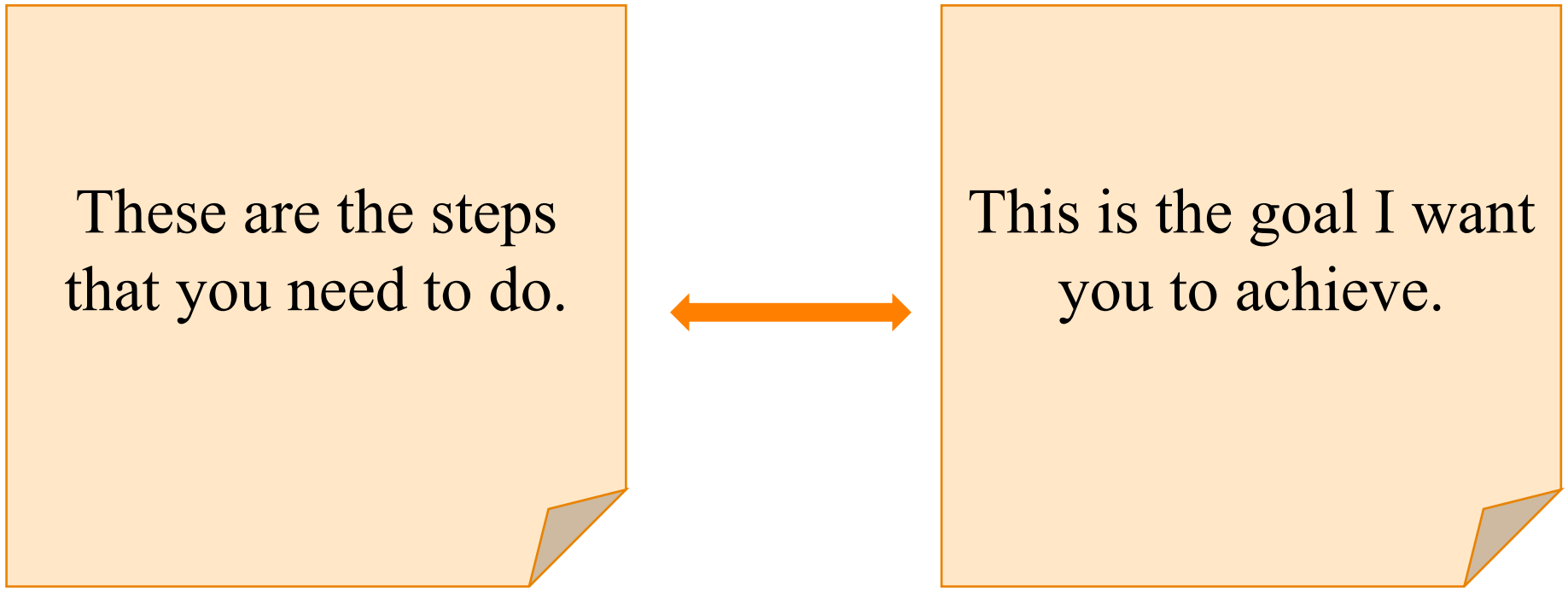
Yu, J., Ma, X., & Han, T. (2016). Four-Dimensional Usability Investigation of Image CAPTCHA. *arXiv preprint arXiv:1612.01067*.

What is Machine Learning?

Machine Learning: “a type of artificial intelligence in which computers use huge amounts of data to learn how to do tasks rather than being programmed to do them”

Oxford Dictionary

Machine Learning



What is Supervised Machine Learning?

Supervised Machine Learning: “a for of machine learning, where we aim to predict a variable that, for at least part of our data is known”

Van Atteveldt et al., 2022

Start with something you know: Regression!

$y = \text{constant} + b^1 * x^1 + b^2 * x^2$

y = Is this a dog (0 = definitely no, 1 = definitely yes)

x^1 = Does it bark? (0 = definitely no, 1 = definitely yes)

x^2 = Does it have a tail? (0 = definitely no, 1 = definitely yes)

Start with something you know: Regression!

$y = 0 + 0.8 * 1 + 0.2 * 0$

y = Is this a dog (0 = definitely no, 1 = definitely yes)

x^1 = Does it bark? (0 = definitely no, 1 = definitely yes)

x^2 = Does it have a tail? (0 = definitely no, 1 = definitely yes)

Start with something you know: Regression!

$$0.8 = 0 + 0.8 * 1 + 0.2 * 0$$

y = Is this a dog (0 = definitely no, 1 = definitely yes)

x^1 = Does it bark? (0 = definitely no, 1 = definitely yes)

x^2 = Does it have a tail? (0 = definitely no, 1 = definitely yes)

I'll start by something you know: Regression!

$0.8 = 0 + 0.8 * 1 + 0.2 * 0$

Classification: A predictive modeling problem where a class label is predicted for a given example of input data.

The basic idea behind SML

Traditional usage of models in CS: To explain.

Usage of models in ML: To predict.

Compare

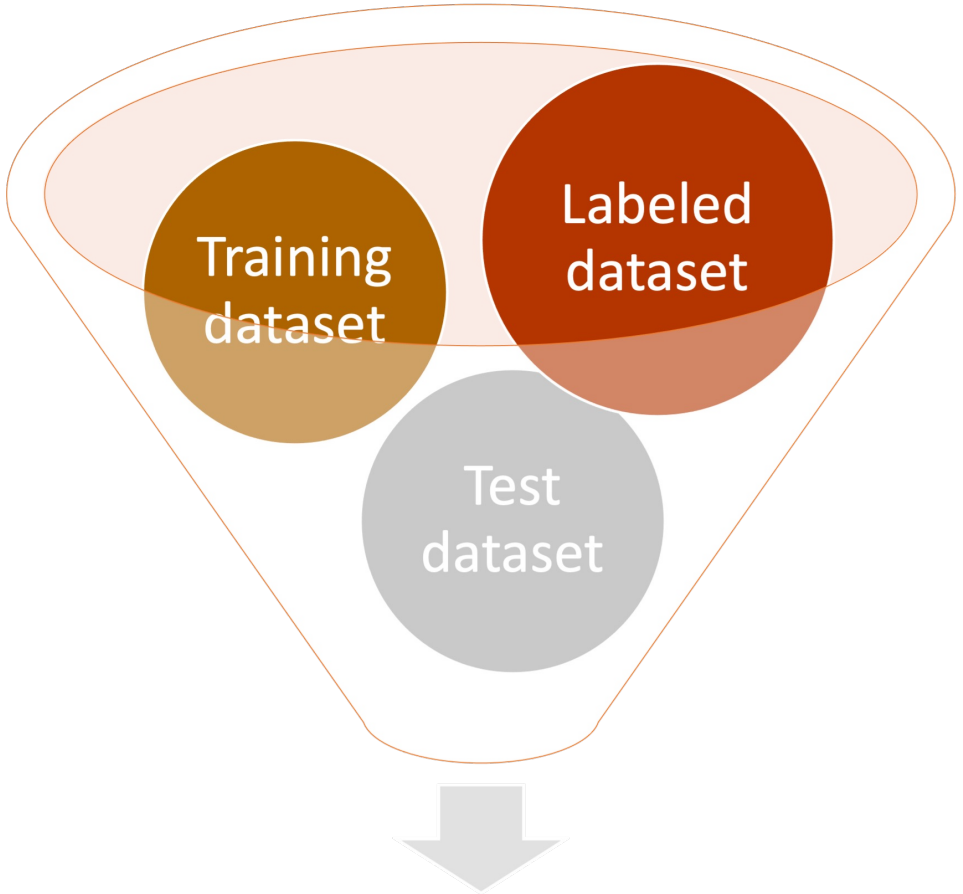
RQ_{trad}: To what extent does the number of hours spend playing video games predict aggressive behavior in individuals?

RQ_{ML}: Given the number of hours that this individual spends playing video games, how likely is this person to show aggressive behavior?

What are we talking about?!

- ~~What you know already~~
- ~~How today relates to the past weeks~~
- ~~Rule-based text classifications~~
- ~~Automated text classifications: SML~~
- ~~The principles behind SML~~
- Practical applications of SML
- Some commonly used SML models

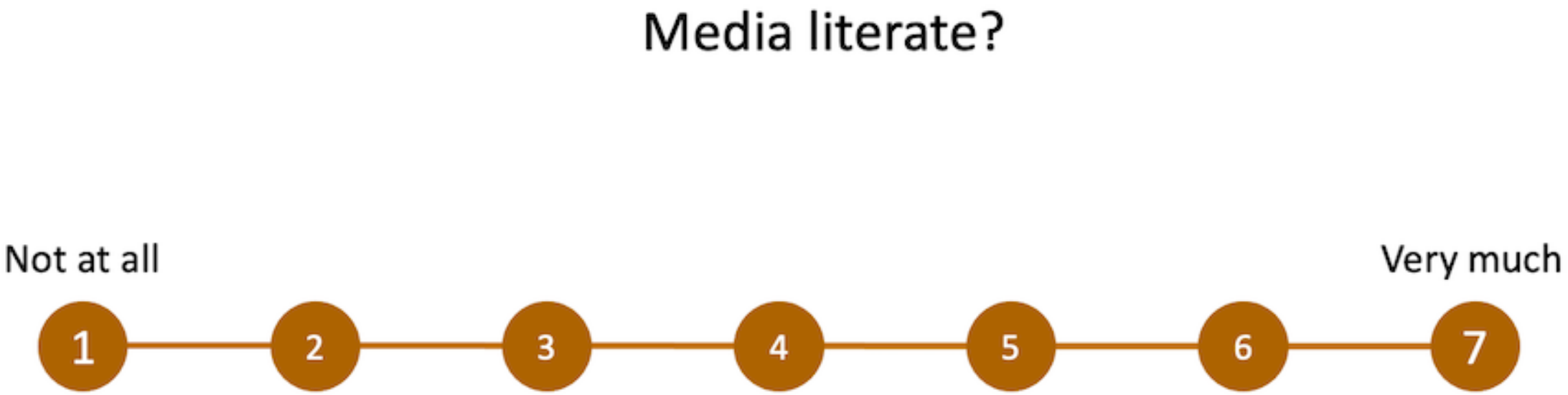
SML step by step



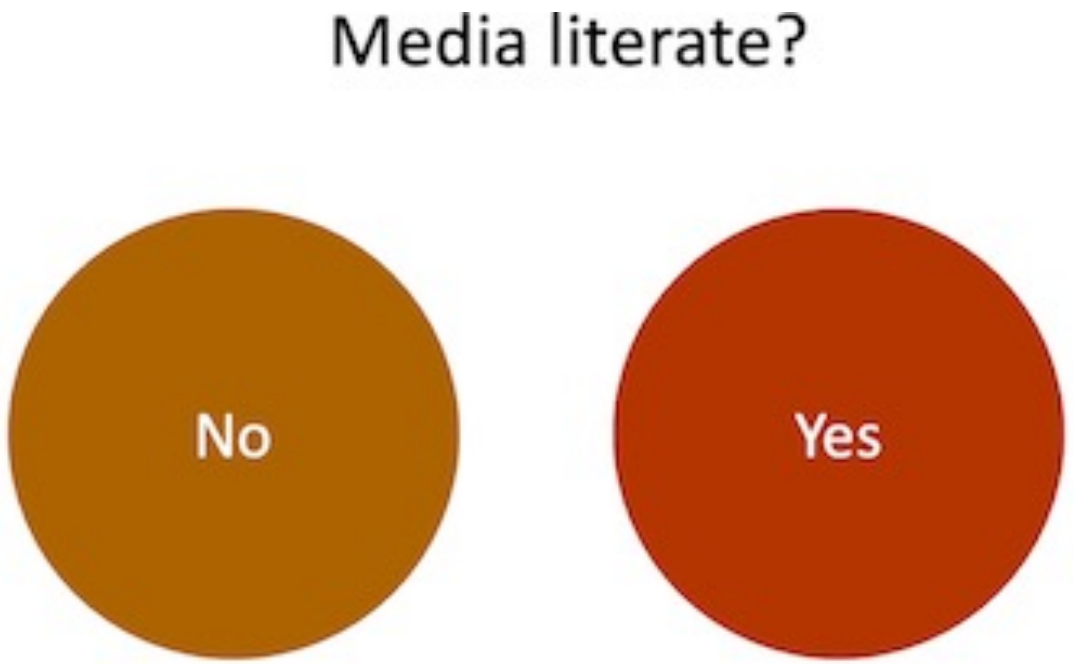
Machine Learning Process

SML step by step

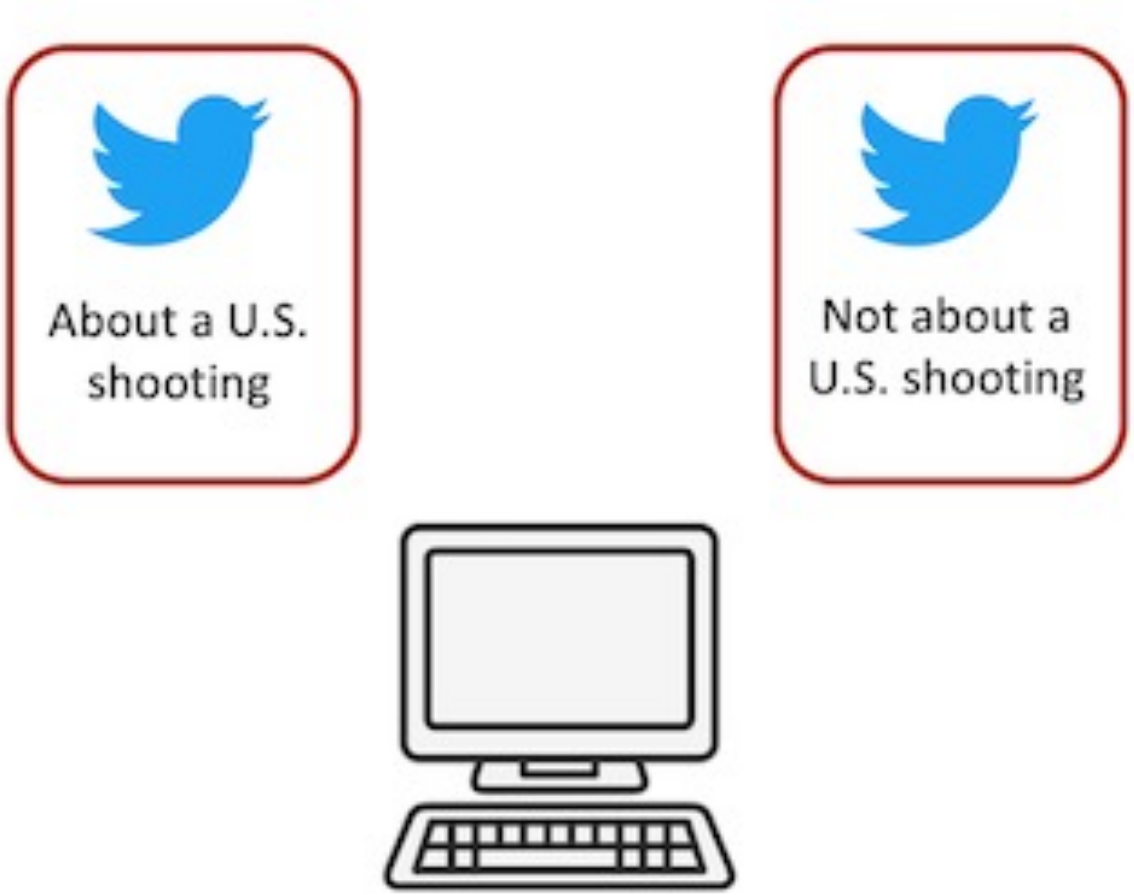
Regression



Logistic regression

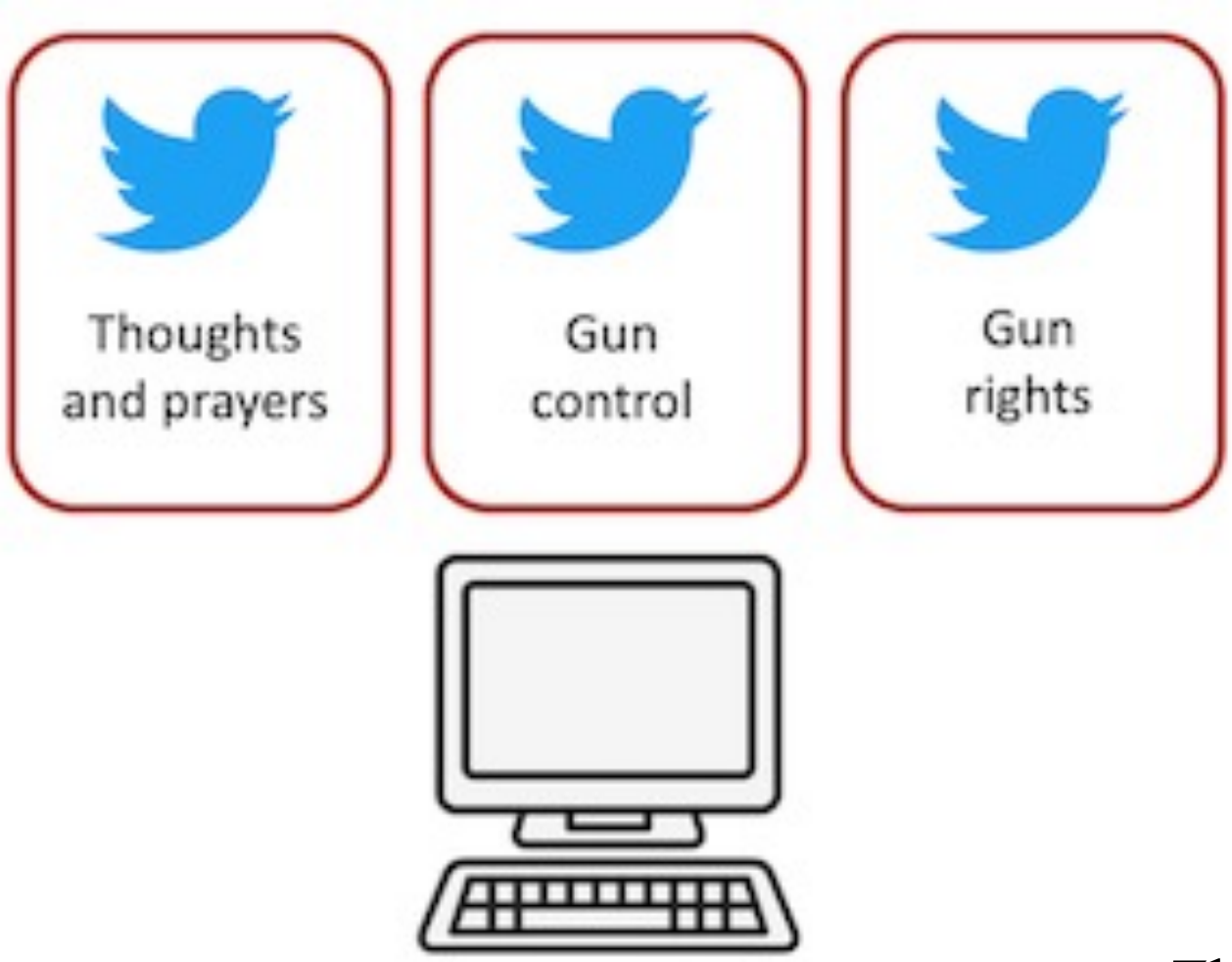


Logistic regression



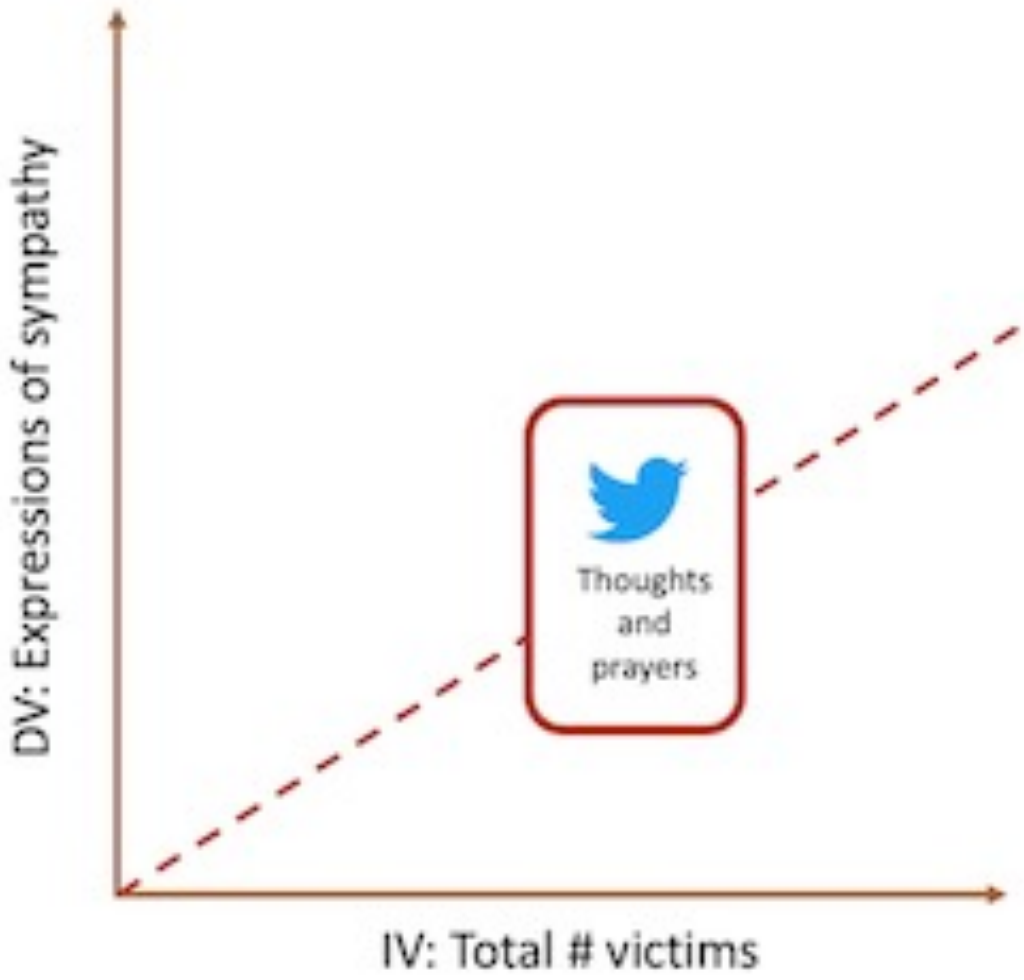
Zhang et al. (2019)

Logistic regression



Zhang et al. (2019)

Logistic regression



Zhang et al. (2019)

What does this look like in code?

First, we need to read in all the stuff that we need:

```
import csv

from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import (TfidfVectorizer)
from sklearn.linear_model import (LogisticRegression)
```

We use Scikit learn to do machine learning.

Not on your computer yet? Pip install scikit learn before you start the exercise later!

What does this look like in code?

Let's create some fake data to practice and split that data into a test set and a training set:

```
tweets = ['This is a tweet', 'Tweet about shooting', 'Another tweet', 'Shooting!',
          'There was a gun', 'There is trouble in the state of Denmark', 'One more tweet',
          'Tweet gun']

labels = [0, 1, 0, 1, 1, 0, 0, 0]

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

What does this look like in code?

```
print(tweets_train)
print(tweets_test)
```

```
['This is a tweet', 'Tweet gun', 'Another tweet', 'There was a gun', 'Shooting!', 'One more tweet']
['Tweet about shooting', 'There is trouble in the state of Denmark']
```

```
print(y_train)
print(y_test)
```

```
[0, 0, 0, 1, 1, 0]
[1, 0]
```

What does this look like in code?

We need to vectorizer our data (input) before we give it to the computer to learn from:

```
tfidfvectorizer = TfidfVectorizer(stop_words="english")

X_train = tfidfvectorizer.fit_transform(tweets_train)
X_test = tfidfvectorizer.transform(tweets_test)
```

What does this look like in code?

Then, train a machine on the training data and test its performance:

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

What does this look like in code?

What did we now create?

```
print(y_pred)
print(y_test)
```

```
[0 0]
[1, 0]
```

How well did the machine do?

SML models

Logistic Regression is only one of the commonly used models to train classifiers.

Let’s have a look at the other models as well!

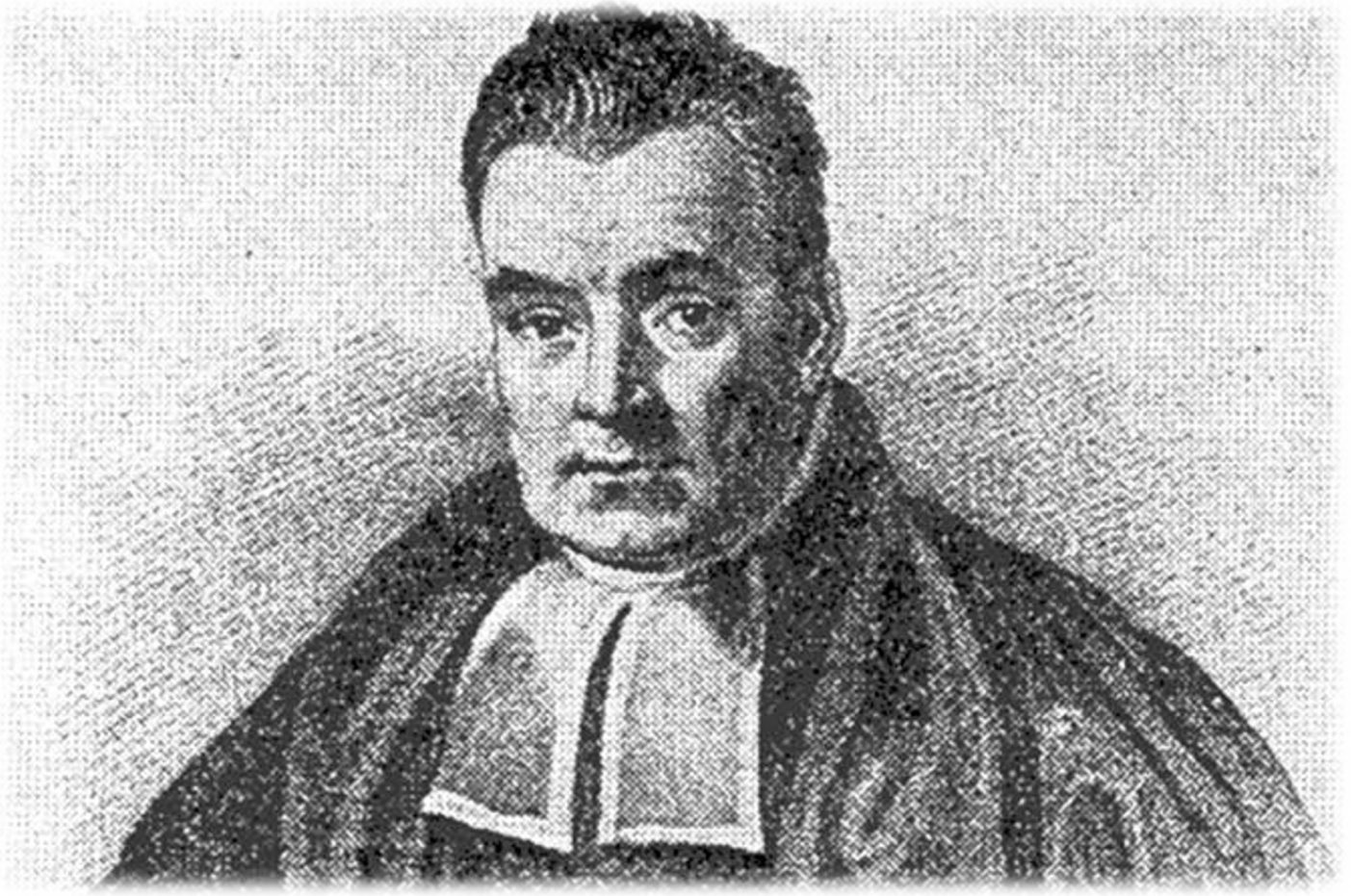
What are we talking about?!

- ~~What you know already~~
- ~~How today relates to the past weeks~~
- ~~Rule-based text classifications~~
- ~~Automated text classifications: SML~~
- ~~The principles behind SML~~
- ~~Practical applications of SML~~
- Some commonly used SML models

Naïve Bayes

Possibly* Thomas Bayes

(1702 – 1761)



*Possibly, because it is unclear if this guy is actually him, but there is no other (claimed) portrait of him.

Naïve Bayes

$$P(\text{A}|\text{B}) = \frac{\text{P}(\text{B}|\text{A}) \times \text{P}(\text{A})}{\text{P}(\text{B})}$$

Mathematicians' language for: The probability of A if B is being the case/present/true

$$P(\text{label}|\text{features}) = \frac{\text{P}(\text{features}|\text{label}) \times \text{P}(\text{label})}{\text{P}(\text{features})}$$

What does this look like in code?

Let's train a model based on a count vectorizer and Naïve Bayes:

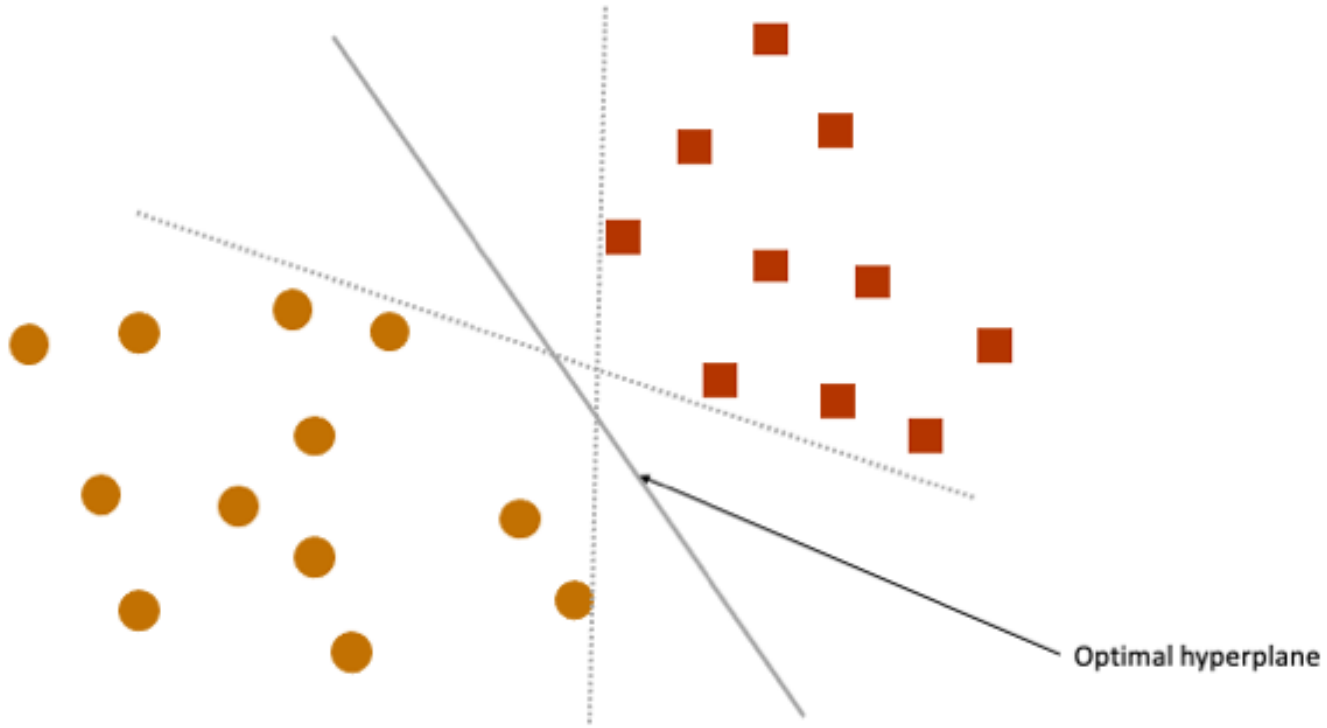
```
from sklearn.feature_extraction.text import (CountVectorizer)
from sklearn.naive_bayes import MultinomialNB

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

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

Support Vector Machines

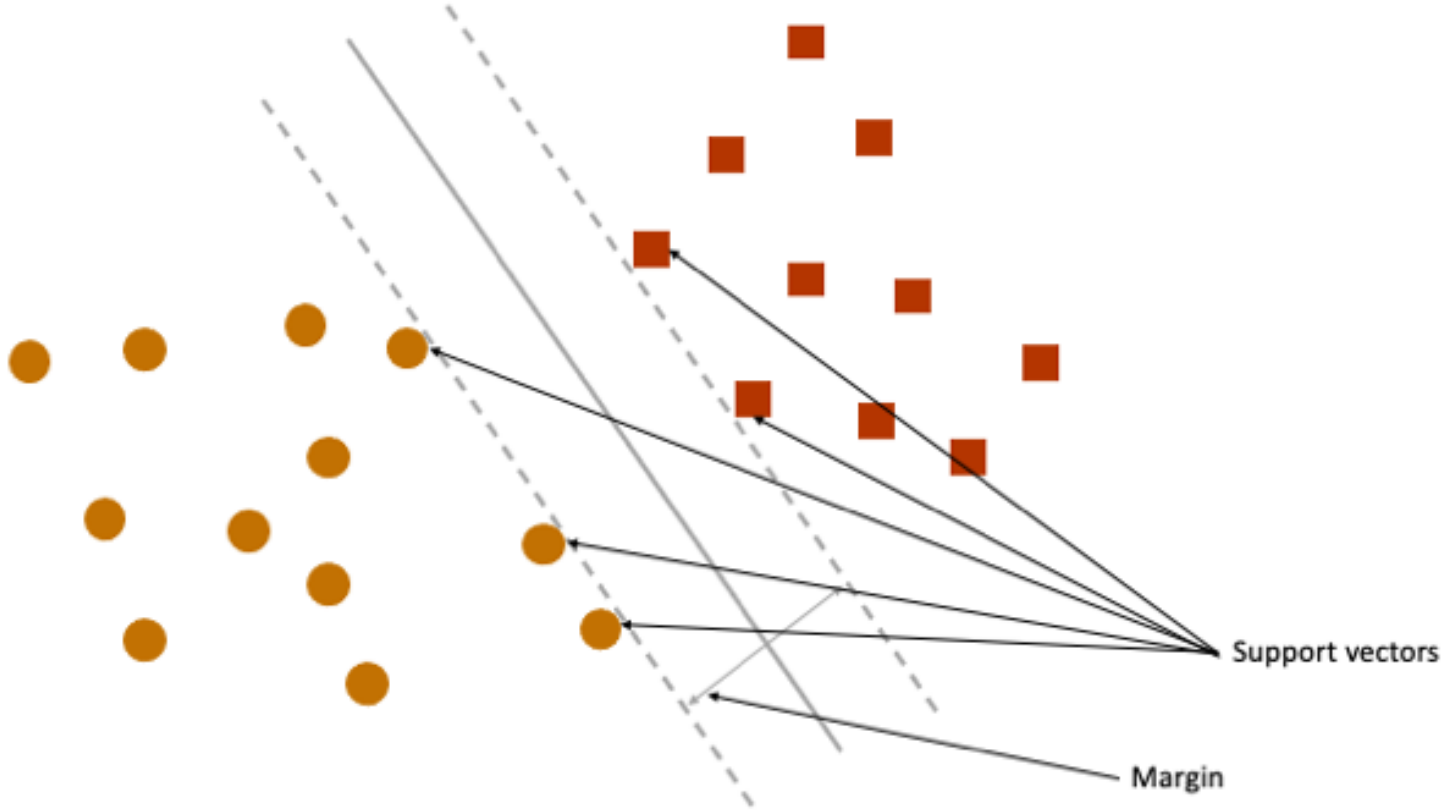
Aim to find a hyperplane in an *N*-dimensional space that distinctly classifies the datapoints.

The best hyperplane is the one that has the maximum margin (distance) between the datapoints of both classes.

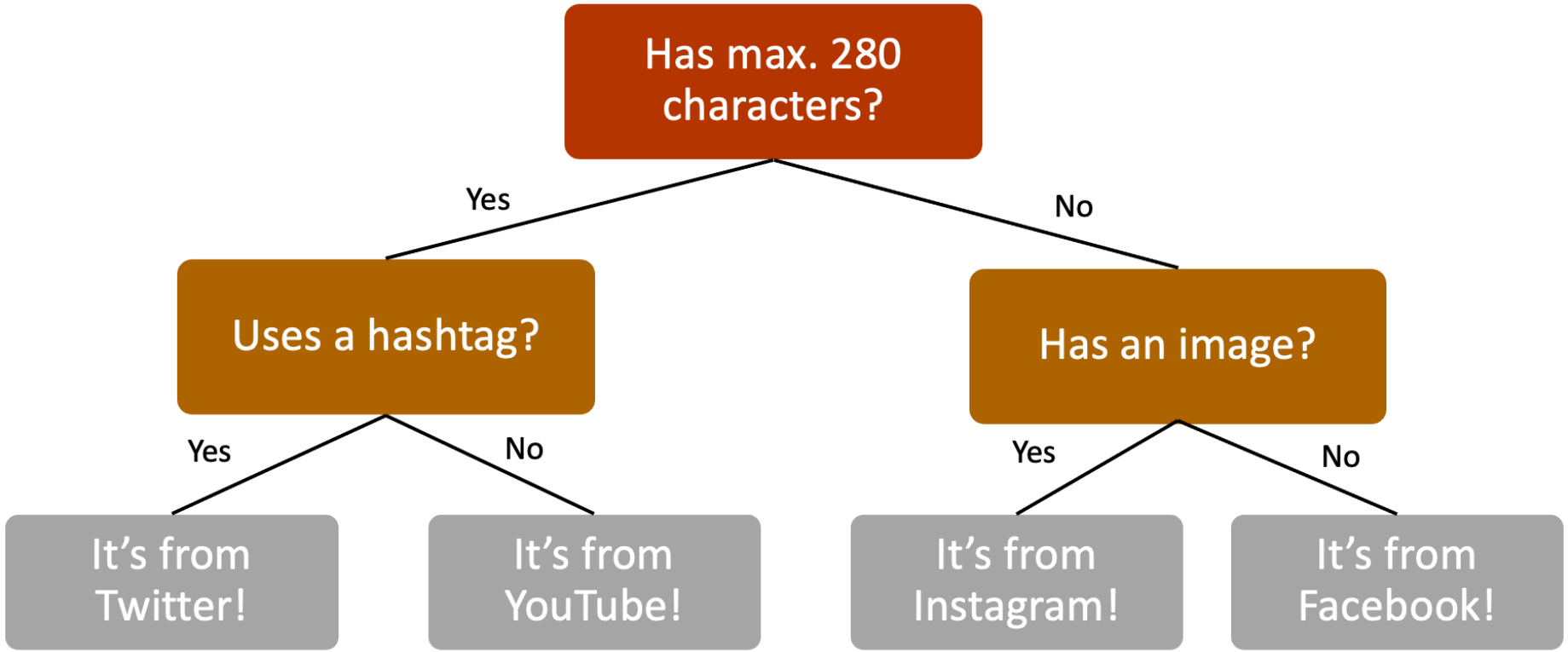
Support Vector Machines



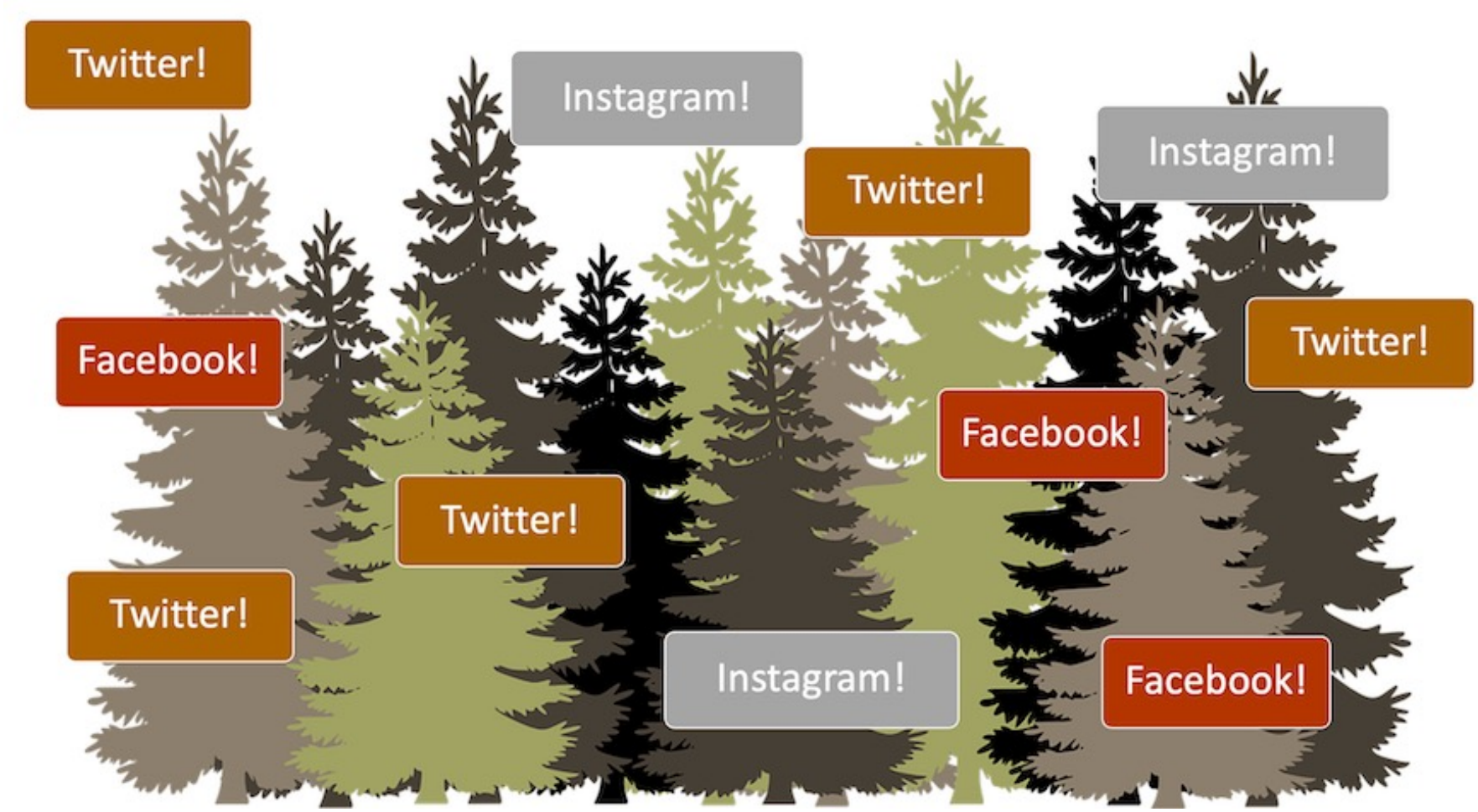
Support Vector Machines



Decision Trees and Random Forests



Decision Trees and Random Forests



More models available!

There are many different models that you can use.

How do you know which is the best for your project?

Try it out and validate the results!

What are we talking about?!

- ~~What you know already~~
- ~~How today relates to the past weeks~~
- ~~Rule-based text classifications~~
- ~~Automated text classifications: SML~~
- ~~The principles behind SML~~
- ~~Practical applications of SML~~
- ~~Some commonly used SML models~~

Enough talking – let's get cracking!

Find Exercise 1 on Github (Jupyter Notebook)

The exercise takes you through the steps to train one machine

Upon completion, your computer spits out something…

…we talk about this something together after the break

The cracking stops at: 12:00 hrs (break!)

Up next

Time:	**Topic:**
~~09:30 – 11:00~~	~~Basics of (supervised) machine learning~~
~~11:00 – 12:00~~	~~Exercise 1~~
12:00 – 13:00	Break
13:00 – 13:30	Recap on Exercise 1
13:30 – 14:30	Evaluating models
14:30 – 15:30	Exercise 2
15:30 – 16:00	Recap on Exercise 2
16:00 – 16:30	Closure

Image: https://pixabay.com/nl/vectors/koffie-kop-koffie-beker-cafe%c3%afne-5009730/