LECTURE 6
The Experimental Setup & Naive Bayes

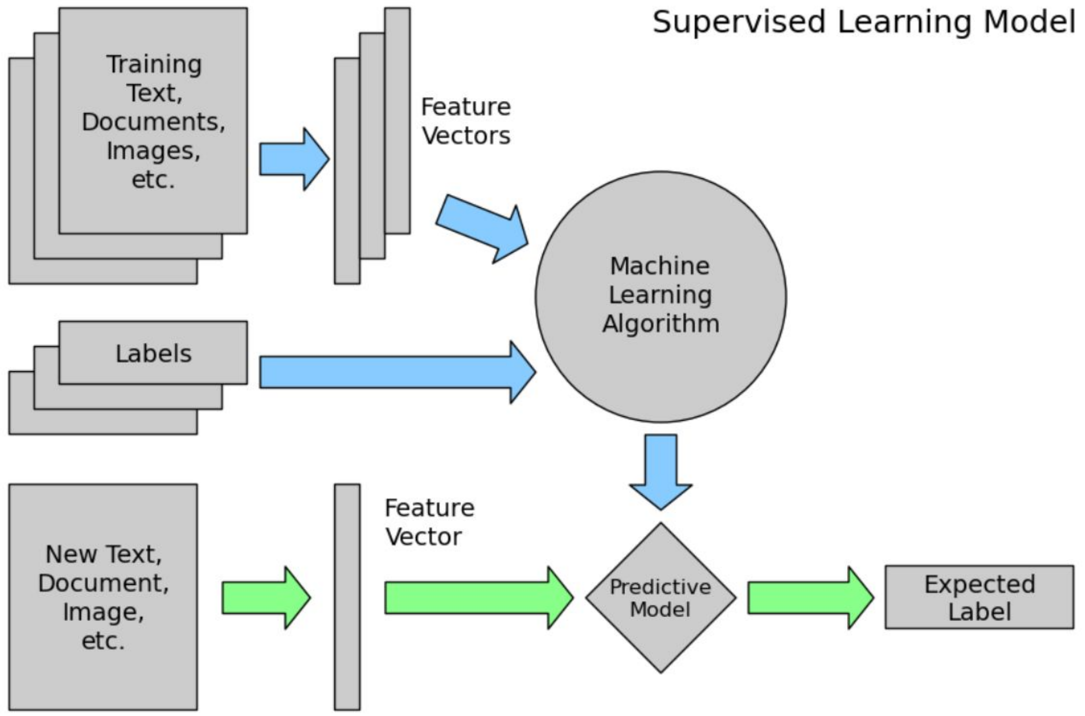
Data Preparation
Feature Representation
Naive Bayes

A reminder on **learning**

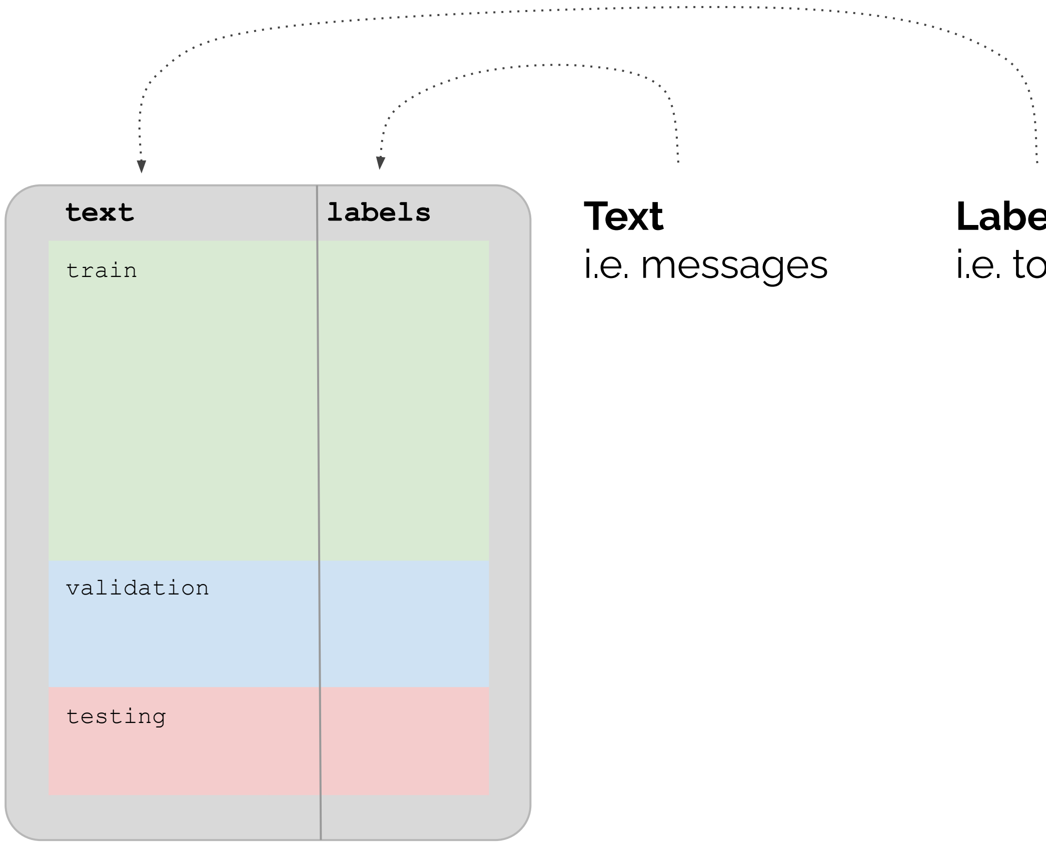
- we have a set of examples and we want to obtain an inference scheme to model our data: we want to **generalise**
- our model is **general enough** if it can describe yet unseen examples (with an acceptable error rate)

Experimental Setup

Supervised Learning


Supervised Learning Model
Training Text, Documents, Images, etc.
Feature Vectors
Machine Learning Algorithm
Labels
New Text, Document, Image, etc.
Feature Vector
Predictive Model
Expected Label

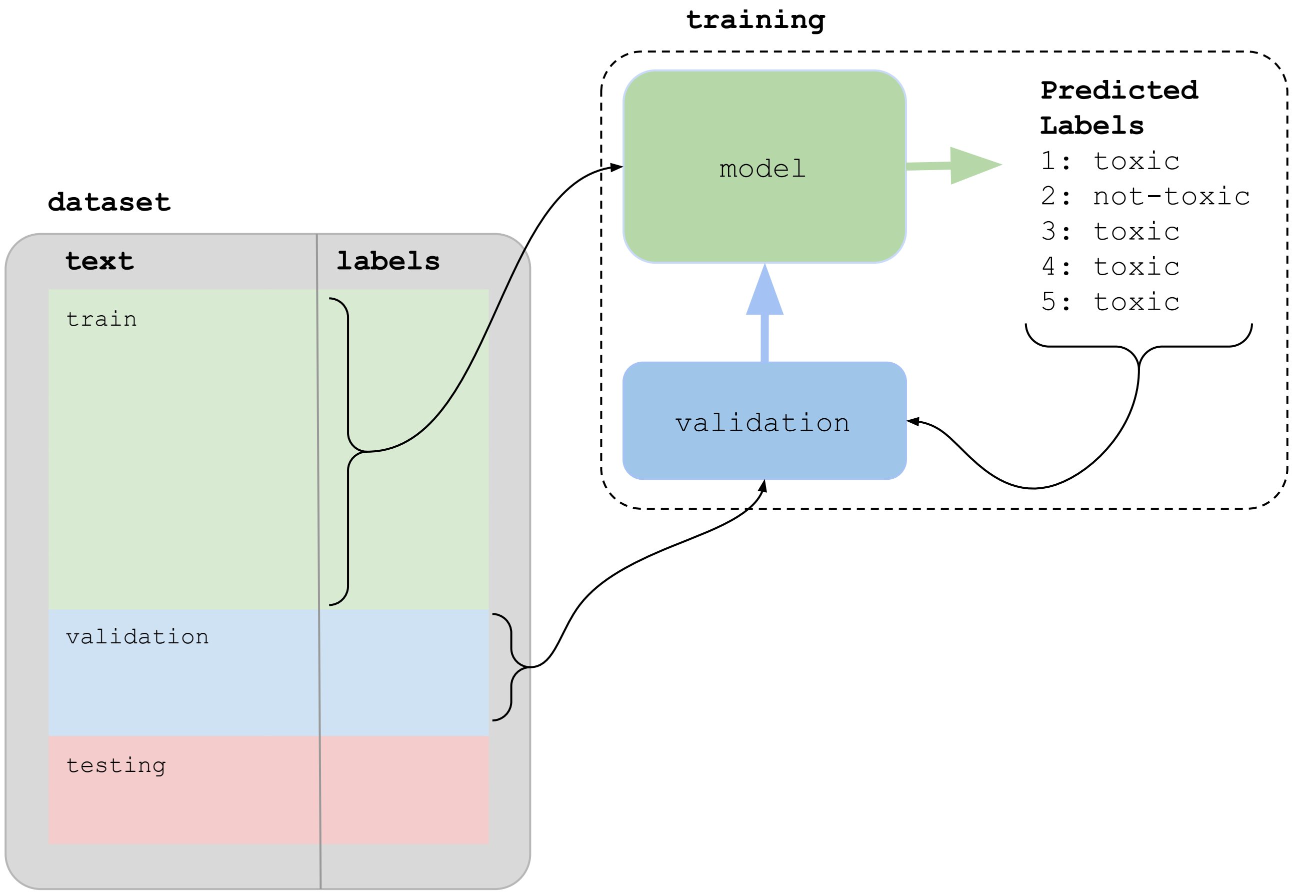
A dataset



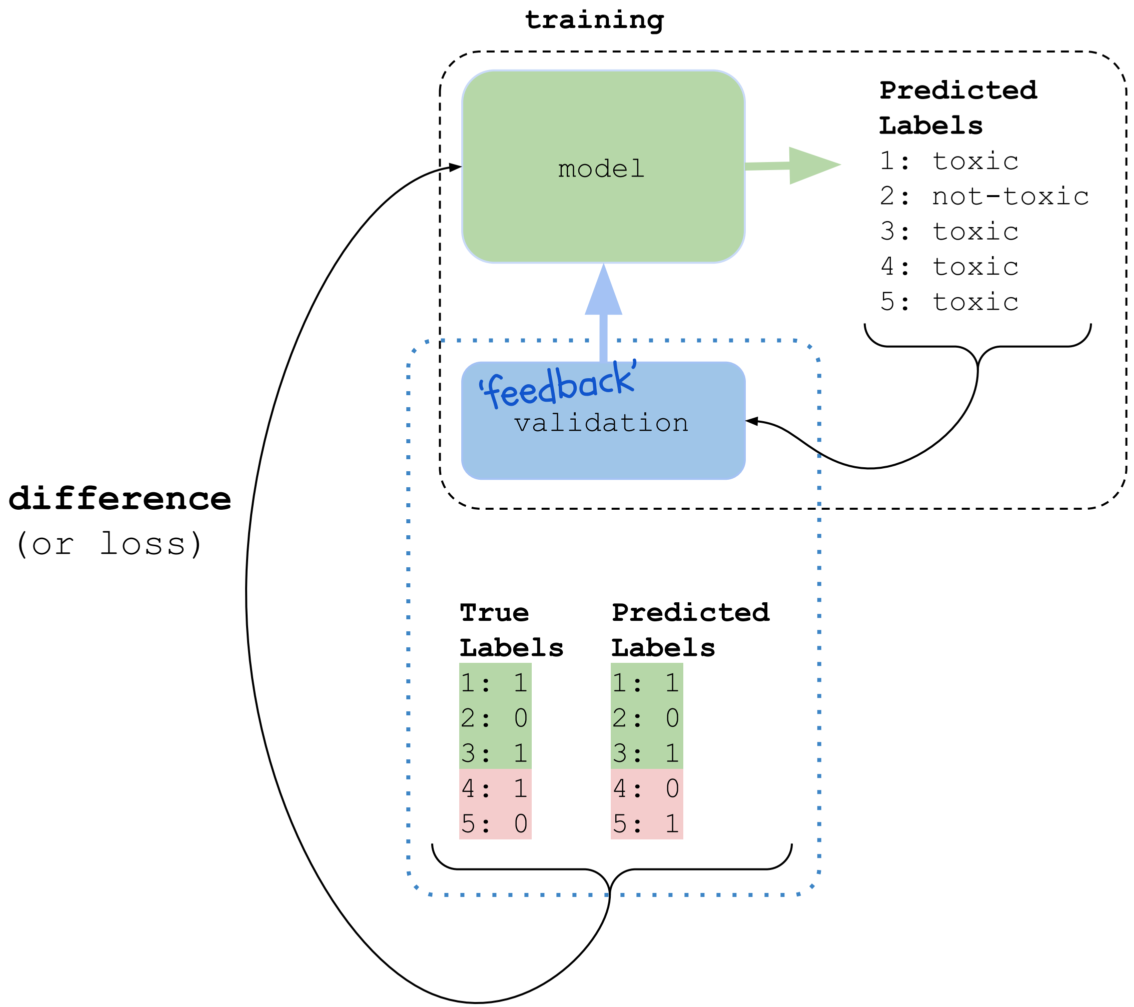
Text
i.e. messages

Labels
i.e. toxic or not-toxic.

Training a Machine Learning Model

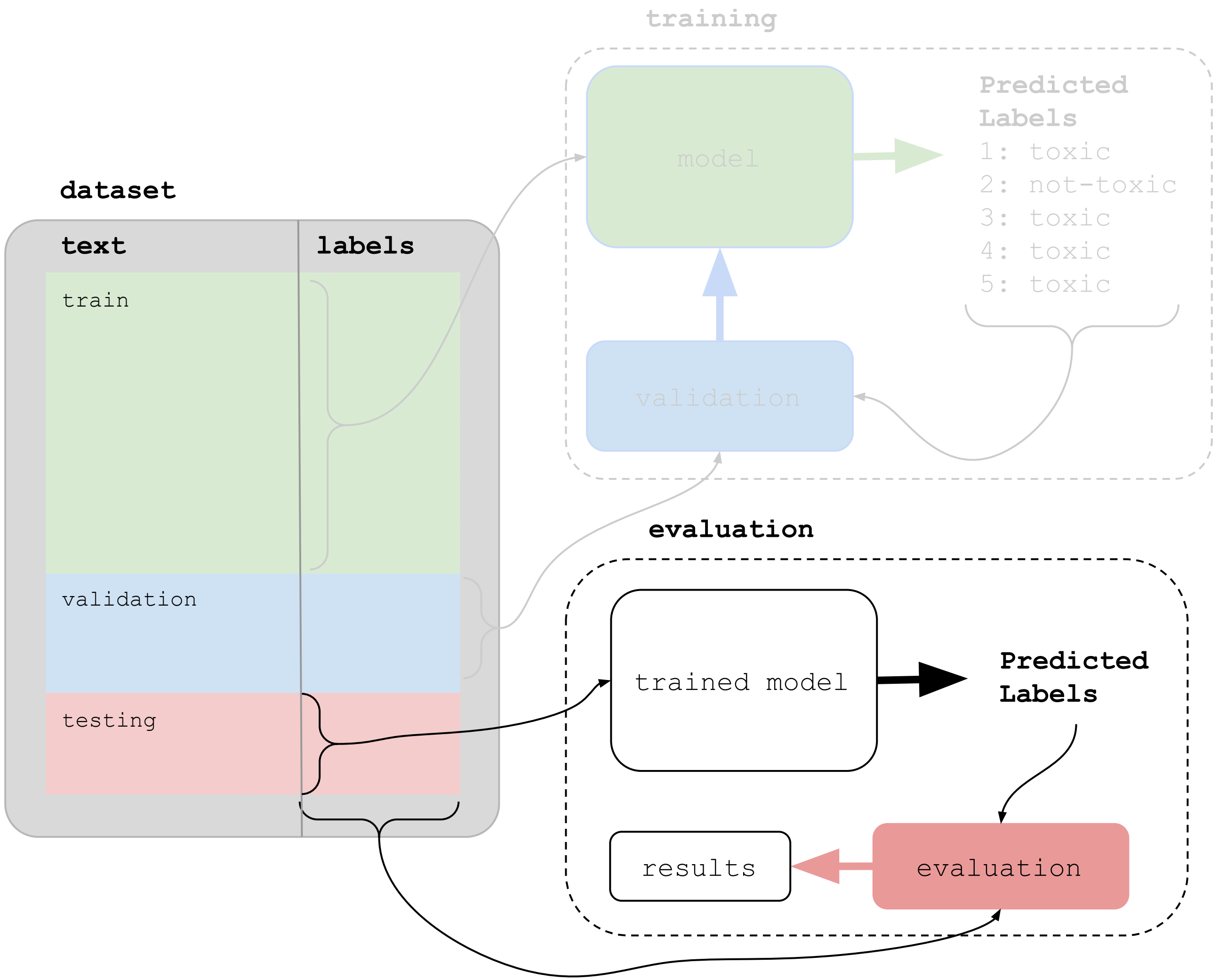


Validation



The model improves its predictions based on the difference between the **true labels** and **predicted labels.**

Evaluation


training
dataset
text
labels
train
validation
testing
model
Predicted Labels
1: toxic
2: not-toxic
3: toxic
4: toxic
5: toxic
validation
evaluation
trained model
Predicted Labels
results
evaluation

Feature Representation

The Inputs: **Feature Representation**

Can we just give the model text?

The Inputs: **Feature Representation**

Can we just give the model text?

- We need to transform examples into something a machine can understand.
- We need to tell the machine what to look for, what the relevant aspects of the phenomenon are.

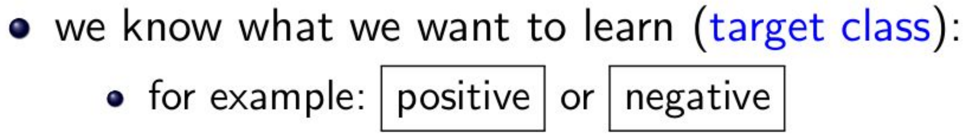
The Inputs: **Feature Representation**

in other words:

- we need to turn each example into some sort of machine-readable summary of itself (choosing relevant features)
- $\rightarrow$ our examples must become vectors of feature values

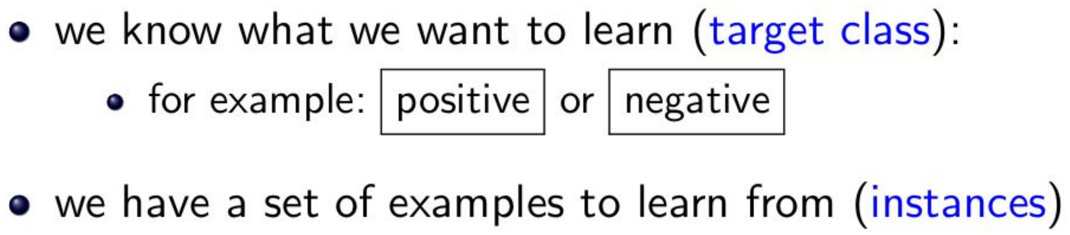
what *are* relevant features?

The Inputs:



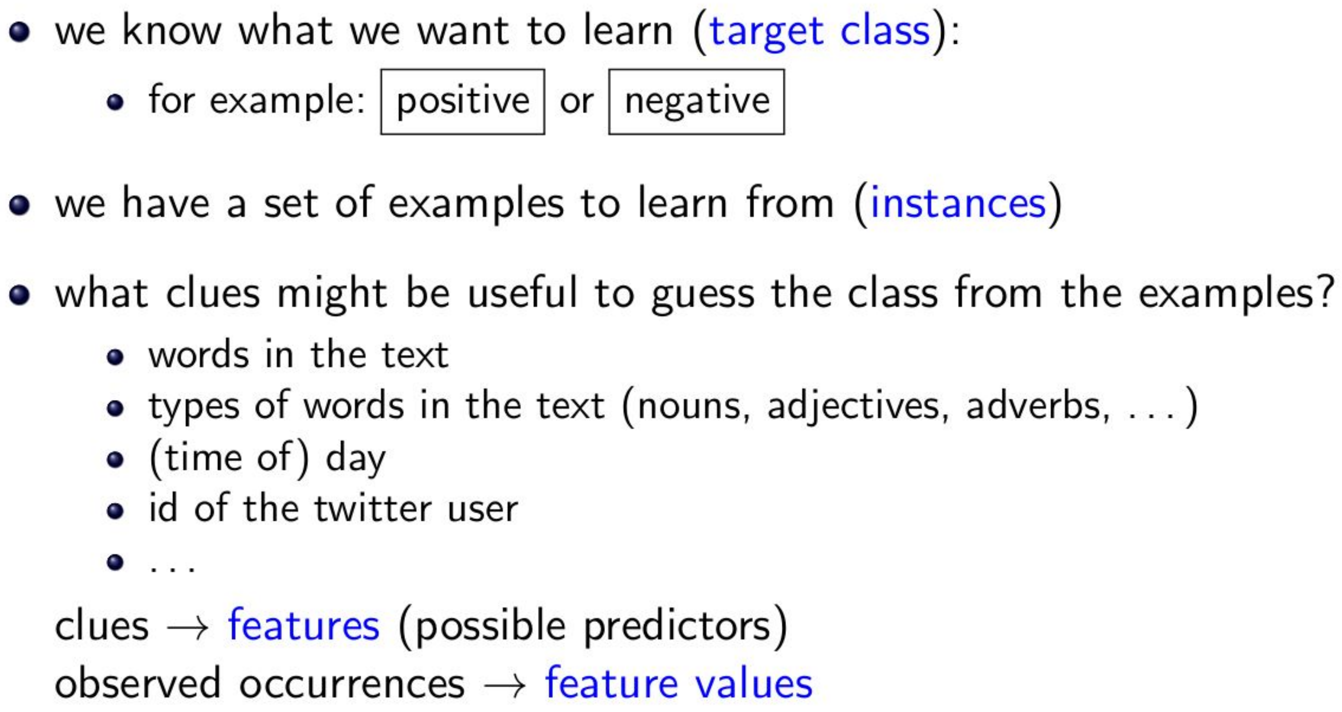
- we know what we want to learn (target class):
  - for example: $\boxed{\text{positive}}$ or $\boxed{\text{negative}}$

The Inputs:



- we know what we want to learn (target class):
    - for example: $\boxed{\text{positive}}$ or $\boxed{\text{negative}}$
- we have a set of examples to learn from (instances)

The Inputs:



- we know what we want to learn (target class):
  - for example: $\boxed{\text{positive}}$ or $\boxed{\text{negative}}$
- we have a set of examples to learn from (instances)
- what clues might be useful to guess the class from the examples?
  - words in the text
  - types of words in the text (nouns, adjectives, adverbs, ...)
  - (time of) day
  - id of the twitter user
  - ...

clues $\rightarrow$ features (possible predictors)

observed occurrences $\rightarrow$ feature values

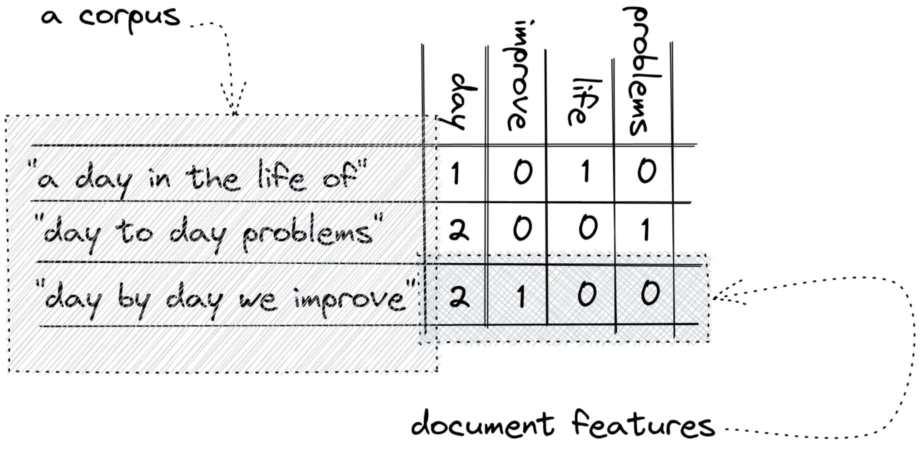
Bag of Words

I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun... It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet!

fairy
always
love
to
it
it
whimsical
it
I
and
seen
are
anyone
friend
happy
dialogue
recommend
adventure
satirical
who
sweet
of
movie
it
it
I
but
to
romantic
I
several
yet
humor
again
it
the
the
seen
would
to
scenes
I
the
manages
the
fun
I
and
times
and
about
whenever
while
have
conventions
with

it	6
I	5
the	4
to	3
and	3
seen	2
yet	1
would	1
whimsical	1
times	1
sweet	1
satirical	1
adventure	1
genre	1
fairy	1
humor	1
have	1
great	1
…	…

The Inputs: **Feature Representation**



	day	improve	life	problems
"a day in the life of"	1	0	1	0
"day to day problems"	2	0	0	1
"day by day we improve"	2	1	0	0

For a model to categorise text, we usually need to represent text documents numerically.

A rudimentary example of feature representation are word counts.

Bag of Words

- Vector representation doesn't consider the ordering of words in a document
- *John is quicker than Mary* and *Mary is quicker than John* have the same vectors
- This is called the bag of words model.

Term Frequency

- The term frequency $\mathrm{tf}_{t,d}$ of term t in document d is defined as the number of times that t occurs in d.
- We want to use tf when computing query-document match scores. But how?

Term Frequency

- The term frequency $\mathrm{tf}_{t,d}$ of term t in document d is defined as the number of times that t occurs in d.
- We want to use tf when computing query-document match scores. But how?
- Raw term frequency is not what we want:
  - A document with 10 occurrences of the term is more relevant than a document with 1 occurrence of the term.
  - But not 10 times more relevant.
- Relevance does not increase proportionally with term frequency.

Document Frequency

- Rare terms are more informative than frequent terms
  - Recall stop words
- Consider a term in the query that is rare in the collection (e.g., *arachnocentric*)
- A document containing this term is very likely to be relevant to the query *arachnocentric*
- → We want a high weight for rare terms like *arachnocentric*.

Tf-IDF

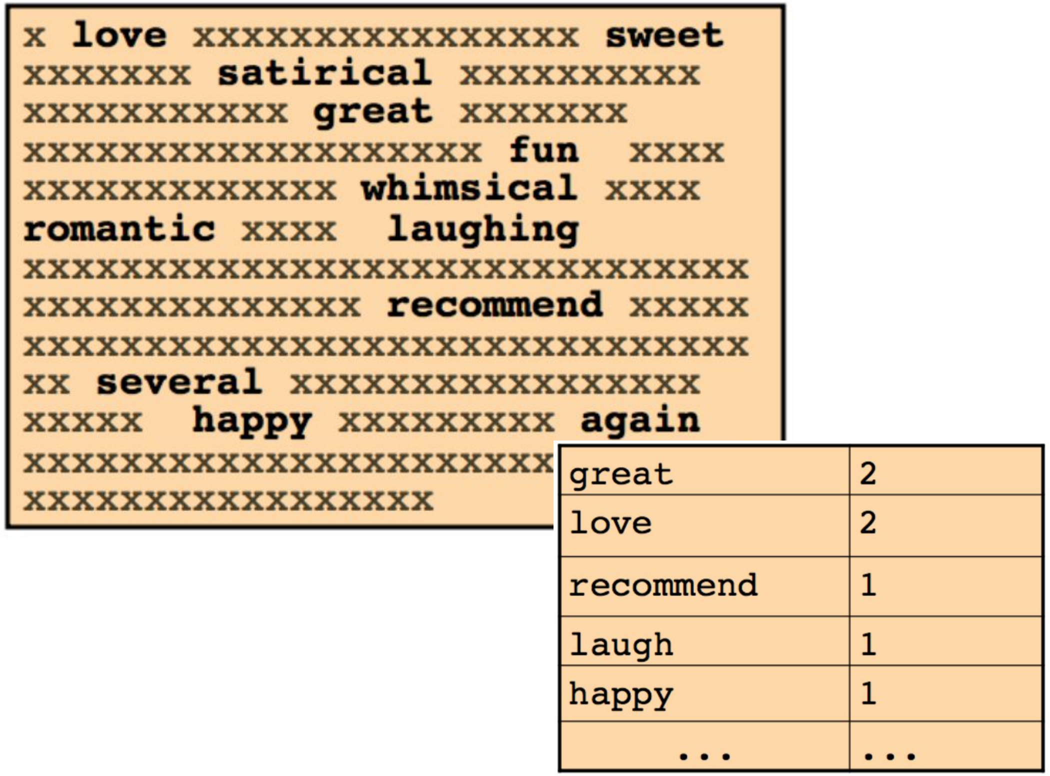
$$\mathrm{idf}(W) = \log \frac{\#(\mathrm{documents})}{\#(\mathrm{documents\ containing\ word\ W})}.$$

$$\text{tf-idf} = \textit{tf} * \textit{idf}$$

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	5.25	3.18	0	0	0	0.35
Brutus	1.21	6.1	0	1	0	0
Caesar	8.59	2.54	0	1.51	0.25	0
Calpurnia	0	1.54	0	0	0	0
Cleopatra	2.85	0	0	0	0	0
mercy	1.51	0	1.9	0.12	5.25	0.88
worser	1.37	0	0.11	4.15	0.25	1.95

Naive Bayes

- simple classification method based on **Bayes rule**
- relies on a simple representation of documents: **bag of words**



great	2
love	2
recommend	1
laugh	1
happy	1
...	...

Conditional probability

- conditional probability of an event: a probability obtained with the **additional information** that **some other event** has already occurred
    - new information is used to revise the probability of the initial event
- **prior** vs **posterior probability**
    - prior: probability obtained “as things stand”, before any additional information is acquired
    - posterior: probability value which has been revised by using additional information

An Example

in a corpus: happy tweets = 45% ; sad tweets = 55%

- I select one instance, how probable is it to be happy?
- The tweet contains the word “cheerful”.
  - “cheerful” occurs in 70% of happy tweets, and in 25% of sad tweets.
  - Does the probability now change?
- which one is prior and which one posterior?

An Example

in a corpus: happy tweets = 45% ; sad tweets = 55%

- I select one instance, how probable is it to be happy?
- The tweet contains the word “cheerful”.
  - “cheerful” occurs in 70% of happy tweets, and in 25% of sad tweets.
  - Does the probability now change?
- which one is prior and which one posterior?

An Example

in a corpus: happy tweets = 45% ; sad tweets = 55%

- I select one instance, how probable is it to be happy?
- The tweet contains the word “cheerful”.
    - “cheerful” occurs in 70% of happy tweets, and in 25% of sad tweets.
    - Does the probability now change?
- which one is prior and which one posterior?

Bayes Theorem

$$p(c_j|i) = \frac{p(i|c_j)p(c_j)}{[p(c_j)p(i|c_j)] + [p(\neg c_j)p(i|\neg c_j)]}$$

c_j : a given class (happy)

i : a given instance (“I always feel cheerful on Friday evening”)

Bayes Theorem

$$\boxed{p(c_j|i)} = \frac{p(i|c_j)p(c_j)}{[p(c_j)p(i|c_j)] + [p(\neg c_j)p(i|\neg c_j)]}$$

$p(c_j|i)$ = probability of instance i being in class c j

how likely is it this given tweet from the corpus is happy?

Bayes Theorem

$$p(c_j|i) = \frac{p(i|c_j)p(c_j)}{[p(c_j)p(i|c_j)] + [p(\neg c_j)p(i|\neg c_j)]}$$

$p(i|c_j)$ = (likelihood function) = pr of generating instance i given

class c_j (happy) given class c_j (happy) how likely is it to get i?

TRUE POSITIVE: 70%

Bayes Theorem

$$p(c_j|i) = \frac{p(i|c_j)p(c_j)}{[p(c_j)p(i|c_j)] + [p(\neg c_j)p(i|\neg c_j)]}$$

p(i|¬c j) = pr of generating instance i given ¬c j (sad) given ¬c j (sad) how likely is it to get i?

FALSE POSITIVE: 25%

Working Example

p “cheerful” in of happy tweets

p Happy

$$p(c_j|i) = \frac{0.70 \cdot 0.45}{[0.45 \cdot 0.70] + [0.55 \cdot 0.25]}$$

p Sad

p “cheerful” in of sad tweets

- c_j : happy
- i : “I always feel cheerful on Friday evening”

- p(i|c_j) = given class c_j (happy) how likely is it to get i?
- p(i|¬c_j) = given class ¬c_j (sad) how likely is it to get i?

A Classifier

- How do we make a classifier out of this?

$$p(c_j|i) = \frac{p(i|c_j)p(c_j)}{p(i)}$$

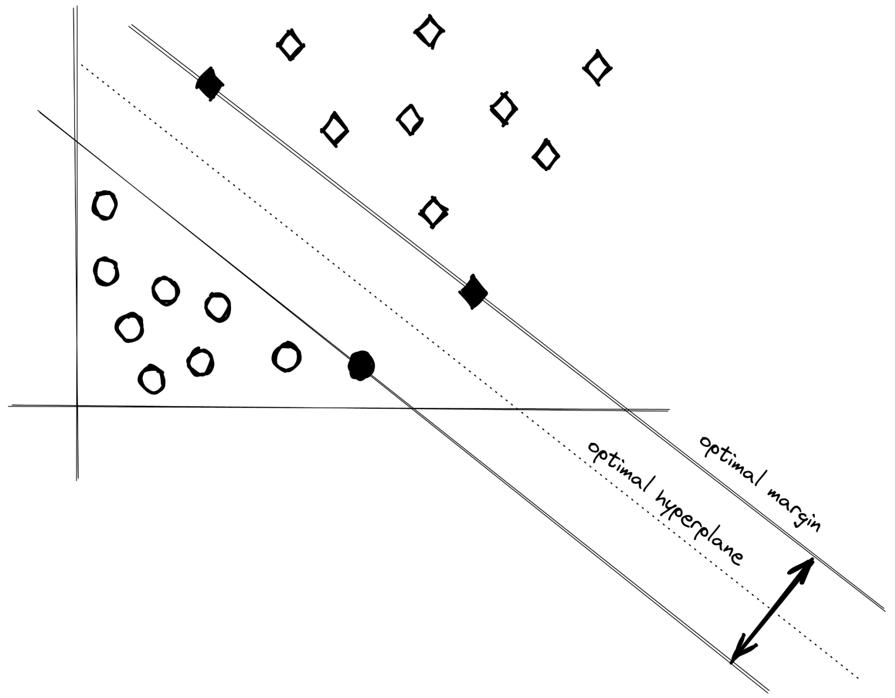
A Naive Bayes Classifier

- How do we make a classifier out of this?
- We add a decision rule: maximum a posteriori (map)

$$p(c_j|i) = \frac{p(i|c_j)p(c_j)}{p(i)}$$

$$c_{map} = \arg\max_{c \in C} \frac{p(i|c) \cdot p(c)}{p(i)}$$

A model:
The Support Vector Machine



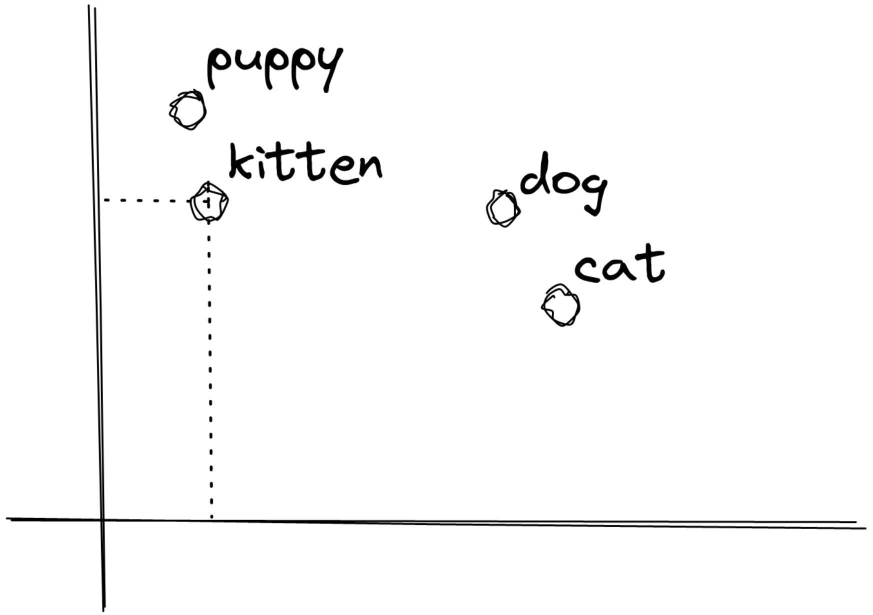
A Support Vector Machine calculates a hyperplane that separates the categories of the problem.

—

The Inputs:
Feature Representation as embeddings

Features can also be represented as spatial embeddings within a vector space, or **word vectors**.

Word vectors are the spatial coordinates of the word within a semantic space.


puppy
kitten
dog
cat