Course: Intelligent Systems

Unit 4: Language Technologies

Language technologies
Part 2

Mariano Rico

2021

Technical University of Madrid

Table of Contents

1. **Part of Speech**
2. **Sparse Vector models**
3. **TF-IDF**
4. **Document classification**
5. **Hands-on 2**

PART OF SPEECH

Part-of-speech tagging

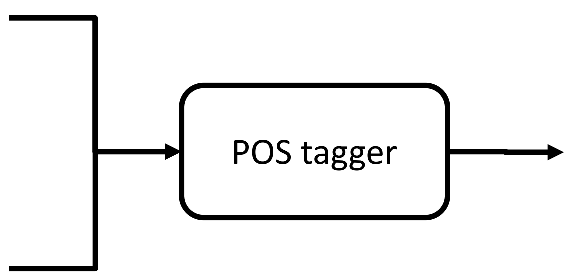
- **Part of speech (POS)**:
  - Noun, verb, pronoun, preposition, adverb, conjunction, participle, article, etc.
- **POS Tagging**
  - Automatic assignment of part-of-speech descriptors (tags) to input tokens

Tagset

/DT: determiner
/JJ: adjective
/NN: noun
/VBD: verb, past tense
…

Text

The grand jury commented on a number of other topics.



Tagged text

The/DT grand/JJ jury/NN commented/VBD on/IN a/DT number/NN of/IN other/JJ topics/NNS ./.

Lexical classes of English words

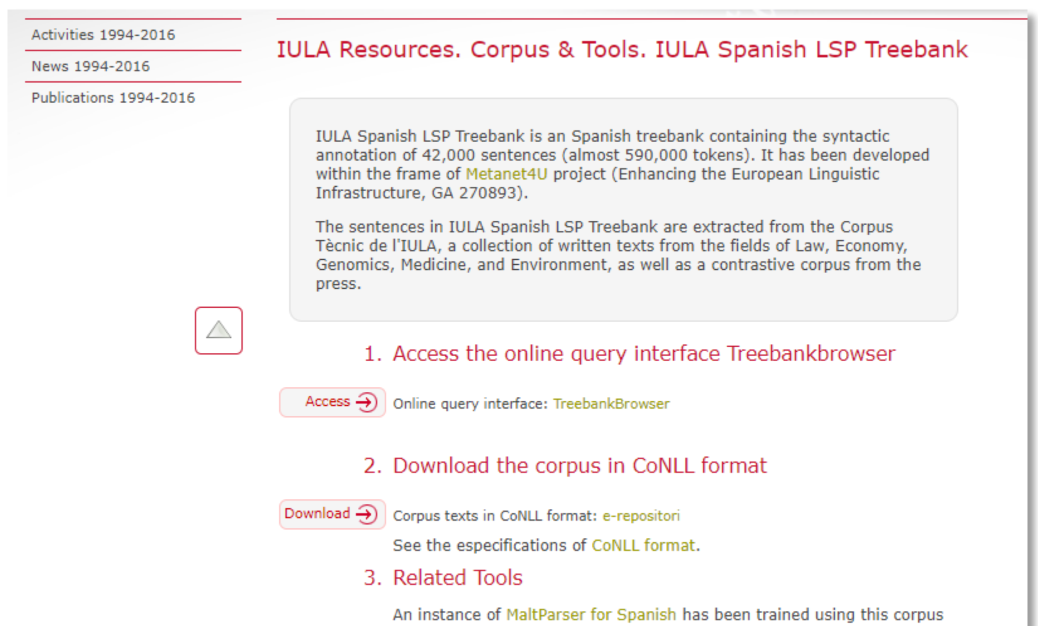
- Two broad categories
  - **Open class types**. Commonly accept the addition of new words
    - Nouns, verbs, adjectives, adverbs
  - **Closed class types**. New words are rarely added
    - Prepositions, determiners, pronouns, conjunctions, etc.
- Others
  - *Interjections* (oh, ah, hey, man, alas, uh, um)
  - *Negatives* (no, not)
  - *Politeness markers* (please, thank you)
  - *Greetings* (hello, goodbye)
  - *The existential there* (there are two on the table)
  - …

Universal Dependencies (UD)

- Framework for a coherent annotation of
  - POS
  - grammar trees
  - Syntactic dependencies
- Created by an open community
  - More than 300 collaborators
  - 200 *treebanks*
  - More than 100 languages
- UD annotations are the evolution of
  - Stanford Universal Dependencies. More info.
  - Google Universal POS. More info.
  - Interlingua from Interset for morph syntactic tagsets. More info.
- Even more info

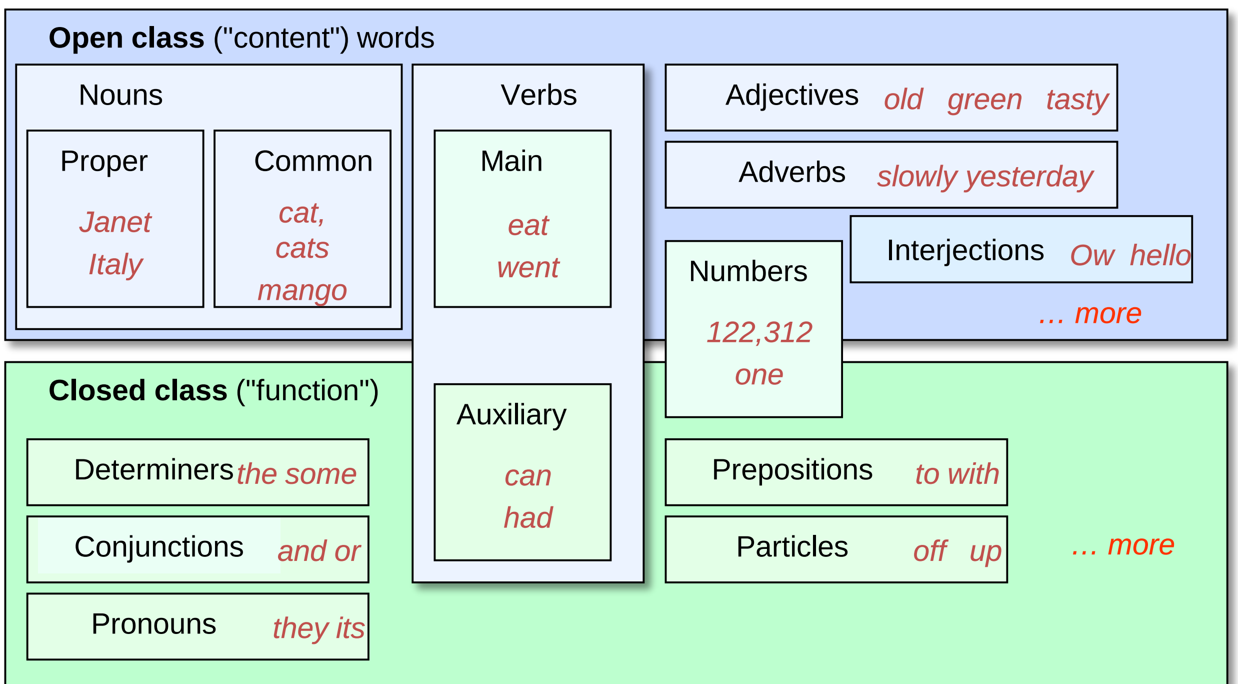
Treebanks for Spanish

- IULA Spanish LSP Treebank
  - Syntactic annotation of 42.000 phrases (590.000 tokens, 631.642 lines)
    - 41MB (uncompressed) in CONLL format (CONLL tagset)
    - Warning, it is NOT CONLL-U.
  - The corpus contains text from newspapers, and texts from areas like law, economy, medicine, genomics, etc.



Universal Dependencies (UD)

- *Tags* (labels) for *POS*
  - The most important (*core*)
    - More info
  - Additional properties



Open class words	Closed class words	Other
ADJ	ADP	PUNCT
ADV	AUX	SYM
INTJ	CCONJ	X
NOUN	DET	
PROPN	NUM	
VERB	PART	
	PRON	
	SCONJ	

Lexical features*	Inflectional features*	
	*Nominal**	*Verbal**
PronType	Gender	VerbForm
NumType	Animacy	Mood
Poss	NounClass	Tense
Reflex	Number	Aspect
Foreign	Case	Voice
Abbr	Definite	Evident
Typo	Degree	Polarity
		Person
		Polite
		Clusivity

Source: Jurafsky 3rd ed.

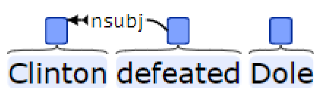
Universal Dependencies (UD)

- POS tags in detail (Nivre et al. 2016)

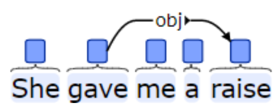
	Tag	Description	Example
Open Class	**ADJ**	Adjective: noun modifiers describing properties	*red*, *young*, *awesome*
	ADV	Adverb: verb modifiers of time, place, manner	*very*, *slowly*, *home*, *yesterday*
	NOUN	words for persons, places, things, etc.	*algorithm*, *cat*, *mango*, *beauty*
	VERB	words for actions and processes	*draw*, *provide*, *go*
	PROPN	Proper noun: name of a person, organization, place, etc..	*Regina*, *IBM*, *Colorado*
	INTJ	Interjection: exclamation, greeting, yes/no response, etc.	*oh*, *um*, *yes*, *hello*
Closed Class Words	**ADP**	Adposition (Preposition/Postposition): marks a noun's spacial, temporal, or other relation	*in, on, by under*
	AUX	Auxiliary: helping verb marking tense, aspect, mood, etc.,	*can, may, should, are*
	CCONJ	Coordinating Conjunction: joins two phrases/clauses	*and*, *or*, *but*
	DET	Determiner: marks noun phrase properties	*a, an, the, this*
	NUM	Numeral	*one, two, first, second*
	PART	Particle: a preposition-like form used together with a verb	*up, down, on, off, in, out, at, by*
	PRON	Pronoun: a shorthand for referring to an entity or event	*she, who, I, others*
	SCONJ	Subordinating Conjunction: joins a main clause with a subordinate clause such as a sentential complement	*that*, *which*
Other	**PUNCT**	Punctuation	; , ()
	SYM	Symbols like $ or emoji	$, %
	X	Other	asdf, qwfg

Universal Dependencies (UD)

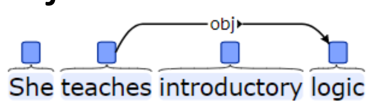
- T*ags* for **relations**
  - The most relevant:
    - nsubj: the subject



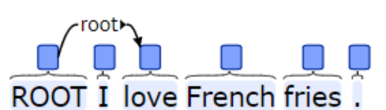
    - obj: the direct object



    - iobj: the indirect objet



    - root: the verb
      - Not represented explicitly in CoNLL-U



	Nominals	Clauses	Modifier words	Function Words
Core arguments	nsubj obj iobj	csubj ccomp xcomp		
Non-core dependents	obl vocative expl dislocated	advcl	advmod* discourse	aux cop mark
Nominal dependents	nmod appos nummod	acl	amod	det clf case
Coordination	**MWE**	**Loose**	**Special**	**Other**
conj cc	fixed flat compound	list parataxis	orphan goeswith reparandum	punct root dep

* The advmod relation is used for modifiers not only of predicates but also of other modifier words.

UD from R

```
library(udpipe)
model <- udpipe_download_model(language = "spanish-ancora") #Alternative: "spanish-gsd"
udmodel_es <- udpipe_load_model(file = model$file_model)

txt <- c("En un lugar de La Mancha, Don Quijote y Sancho esperaban a Cervantes.")
anno <- udpipe_annotate(udmodel_es, x = txt)
df <- as.data.frame(anno)
#Has 14 columns doc_id, paragraph_id, sentence_id, sentence, token_id, token,
#                 lemma,          upos,         xpos,    feats,   head_token_id,
#               dep_rel,          deps,         misc
df[,5:14]
```

```
   token_id     token     lemma  upos  xpos                                               feats head_token_id dep_rel deps           misc
1         1        En        en   ADP   ADP                                        AdpType=Prep             3    case <NA>           <NA>
2         2        un       uno   DET   DET    Definite=Ind|Gender=Masc|Number=Sing|PronType=Art             3     det <NA>           <NA>
3         3     lugar     lugar  NOUN  NOUN                               Gender=Masc|Number=Sing            12     obl <NA>           <NA>
4         4        de        de   ADP   ADP                                        AdpType=Prep             6    case <NA>           <NA>
5         5        La        el   DET   DET     Definite=Def|Gender=Fem|Number=Sing|PronType=Art             6     det <NA>           <NA>
6         6    Mancha    Mancha PROPN PROPN                                                <NA>             3    nmod <NA>   SpaceAfter=No
7         7         ,         , PUNCT PUNCT                                      PunctType=Comm             3   punct <NA>           <NA>
8         8       Don       Don PROPN PROPN                                                <NA>            12   nsubj <NA>           <NA>
9         9   Quijote   Quijote PROPN PROPN                                                <NA>             8    flat <NA>           <NA>
10       10         y         y CCONJ CCONJ                                                <NA>            11      cc <NA>           <NA>
11       11    Sancho    Sancho PROPN PROPN                                                <NA>             8    conj <NA>           <NA>
12       12 esperaban   esperar  VERB  VERB Mood=Ind|Number=Plur|Person=3|Tense=Imp|VerbForm=Fin             0    root <NA>           <NA>
13       13         a         a   ADP   ADP                                        AdpType=Prep            14    case <NA>           <NA>
14       14 Cervantes Cervantes PROPN PROPN                                                <NA>            12     obj <NA>   SpaceAfter=No
15       15         .         . PUNCT PUNCT                                      PunctType=Peri            12   punct <NA> SpacesAfter=\\n
```

¡Warn!, anno is a list containing 3 things (the last two thigs were lost when converted to dataframe):

1) x: The x character vector with text.
2) conllu: annotation in CONLL-U format
3) error: A vector with the same length of x containing possible errors when annotating x

```
cat(anno$conllu, file = "my_annotacion.conllu") #You can load it with udpipe_read_conllu()
```

CoNLL-U tools

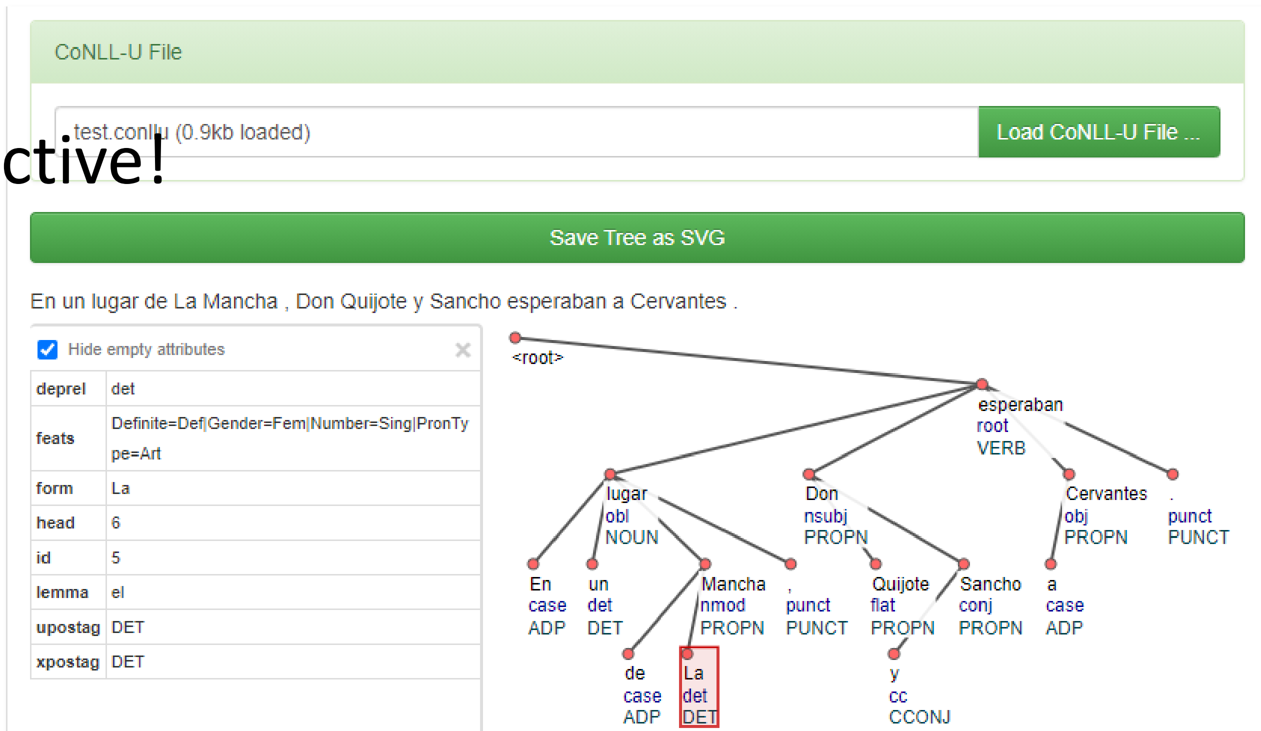
- [UniversalDependencies/Tools]()
  - Relevant command line tools
    - `validate.py` Verifies that a file is CoNLL-U
    - `normalize_Unicode.pl` Convierta UTF-8 to NFC format
    - `conllu_to_conllx.pl` Convierts from CoNLL-U to the previous format (CoNLL-X) that some tools still use
    - `restore_conlu_lines.pl` Joins a CoNLL-U file with a CoNLL-X, returning a CoNLL-U file

UD Tools

This repository contains various scripts in Perl and Python that can be used as tools for Universal Dependencies.

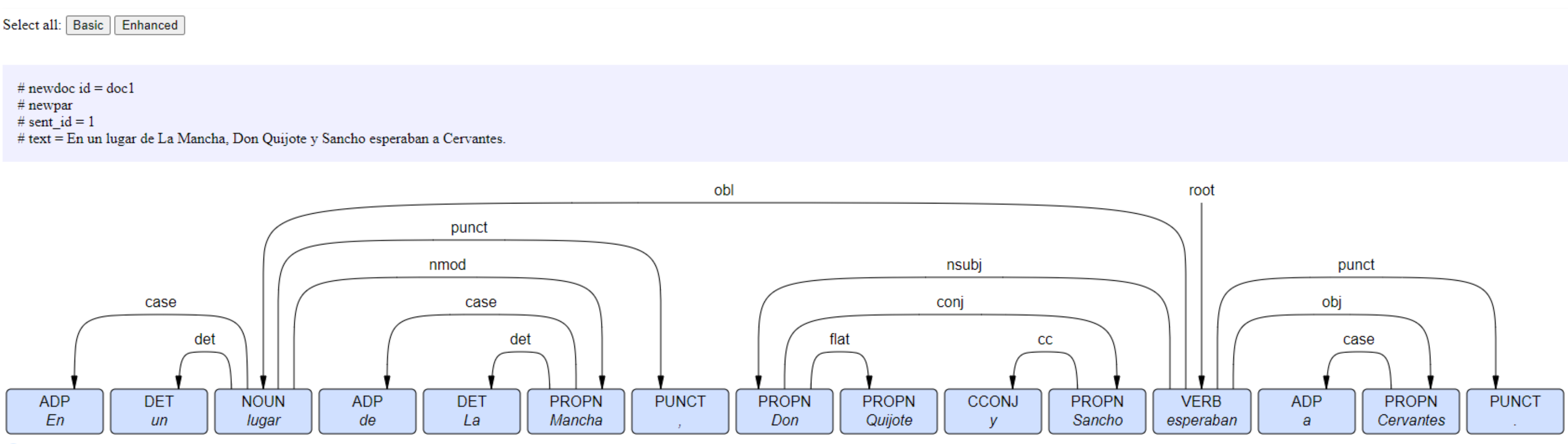
Playing with CoNLL-U files (1/2)

- CoNLL-U Viewer
  - One of the tools in UD
  - URL: https://universaldependencies.org/conllu_viewer.html
  - It is interactive!



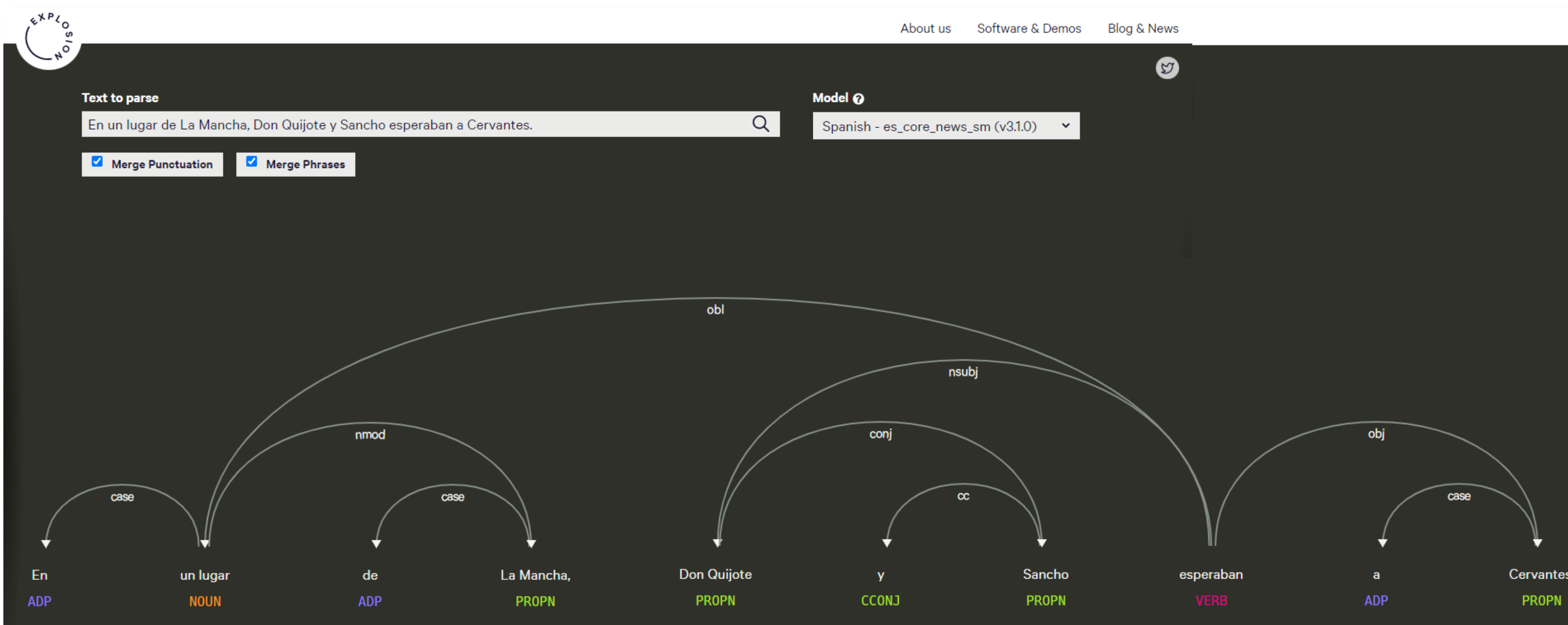
Playing with CoNLL-U files (2/2)

- The tool created by Kleiweg
  - Developed at Univ. Groningen
  - URL: https://urd2.let.rug.nl/~kleiweg/conllu
  - Load the file created previously



Dependencies with SpaCy

- SpaCy (spacy.io) now it is explosion.ai
- Web app to test dependencies
  - For Spanish only has the sm(all) model
- Spacy is faster than UD



Evaluating POS taggers

- **Tagset metrics**
  - *Informativeness*. Not easy to measure; rough measures:
    - Size of the tagset
    - Amount of ambiguity present in the input
  - *Specificability*. Degree to which different linguists uniformly use the tagset when independently tagging the same texts
- **Tagger metrics** (using a benchmark corpus)
  - *Precision/accuracy*
  - *Recall*
  - *Error rate*
  - *Ambiguity*. Average number of analyses in the tagger's output

POS tagging applications

- **Syntax parsing**
  - Basic unit for parsing
- **Information extraction**
  - Indication of names, relations
- **Machine translation**
  - The meaning of a particular word depends on its POS tag
- **Sentiment analysis**
  - Adjectives are the major opinion holders
    - Good vs Bad, Excellent vs Terrible
- **Linguistic studies**
  - Thanks to large tagged text corpora
- …

Table of Contents

1. Part of Speech
2. **Sparse Vector models**
3. TF-IDF
4. Document classification
5. Hands-on 2

SPARSE VECTOR MODELS

The term-document matrix

- Each row is a word (token) in the vocabulary
- Each columns is a document in the corpus
- The cell value is the number of occurrences of the word in the document
  - Example: 4 plays by Shakespeare

Occurrence table

	As You Like It	Twelfth Night	Julius Caesar	Henry V
battle	1	0	7	13
good	114	80	62	89
fool	36	58	1	4
wit	20	15	2	3

The term-document matrix

- Each row is a word (token) in the vocabulary
- Each columns is a document in the corpus
- The cell value is the number of occurrences of the word in the document
  - Example: 4 plays by Shakespeare

Occurrence table

	As You Like It	Twelfth Night	Julius Caesar	Henry V
battle	1	0	7	13
good	114	80	62	89
fool	36	58	1	4
wit	20	15	2	3

A document is characterized by a vector (4 dimensions in this case)

The term-document matrix

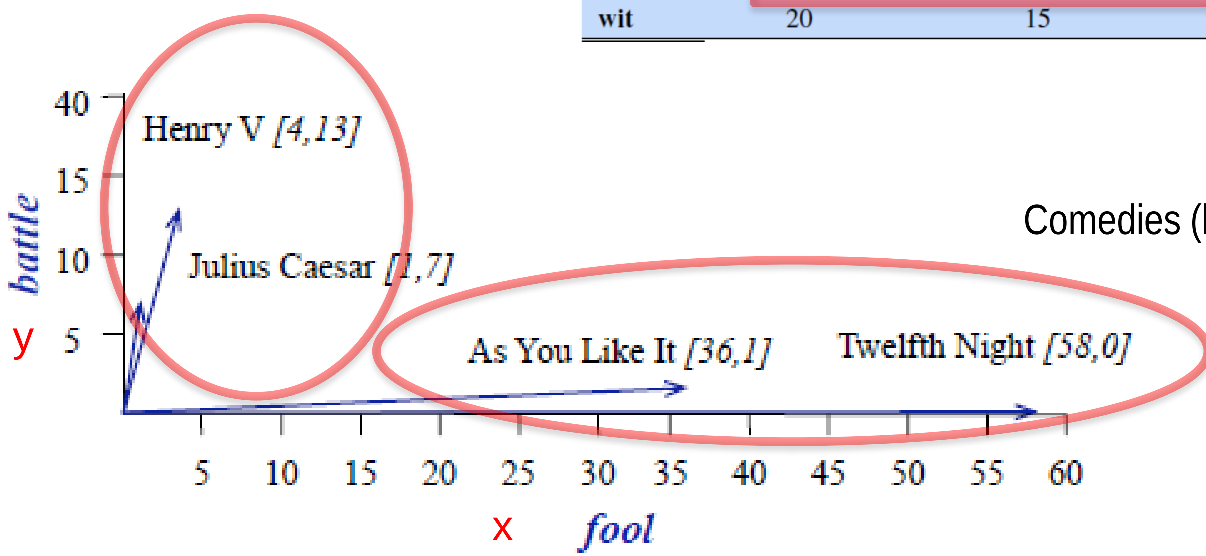
- Let us make a projection to 2 dimensions
  - Over any 2 axis in the space
  - Example: over axis fool and battle

Occurrence table

	As You Like It	Twelfth Night	Julius Caesar	Henry V	
battle	1	0	7	13	y
good	114	80	62	89	
fool	36	58	1	4	x
wit	20	15	2	3	

Epic plays (high values of *battle*)

Comedies (high values of *fool*)

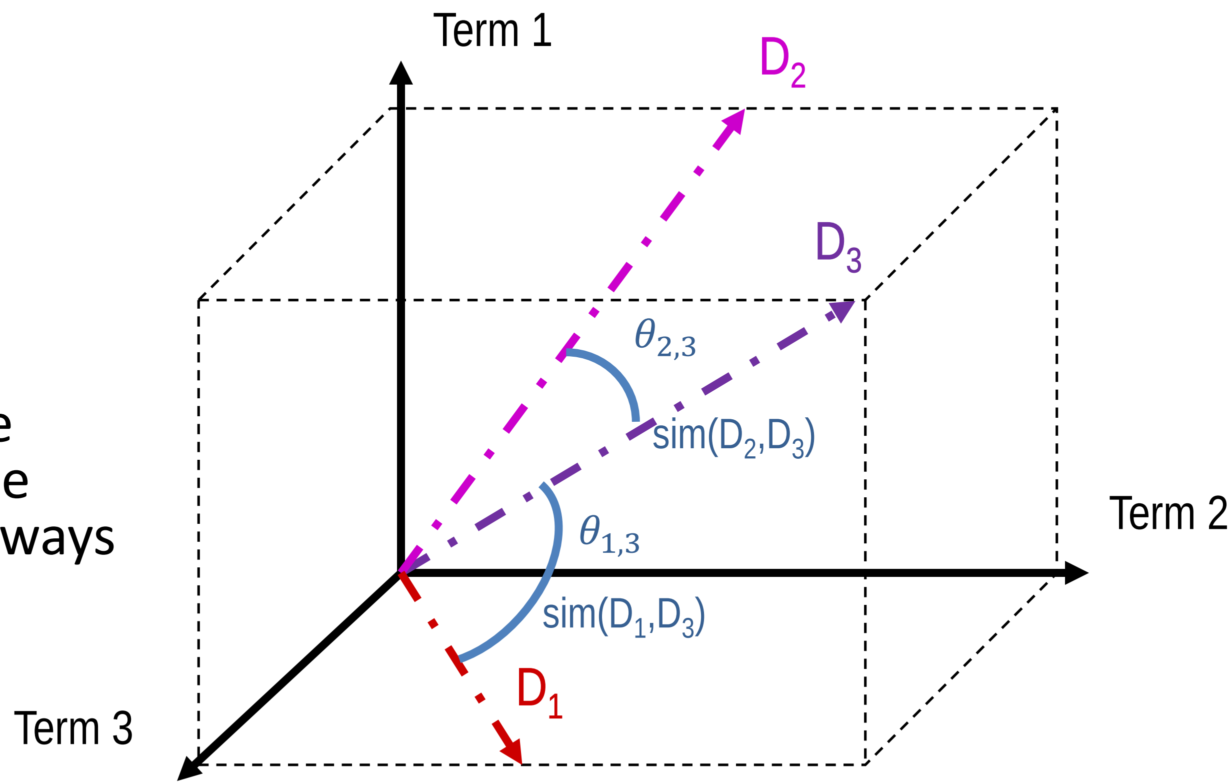


Semantic similarity
Similarity between words and vectors

- Operation with two vectors: dot product $a \cdot b$
  - We have to normalize the vectors (more words frequency do not implies more similarity)

$$\frac{a \cdot b}{|a|\ \ |b|}$$

  - It is the $\cos \theta$
  - As occurrences are always positive, the value of $\cos \theta$ is always between 0 and 1.



Semantic similarity

Similarity between words and vectors

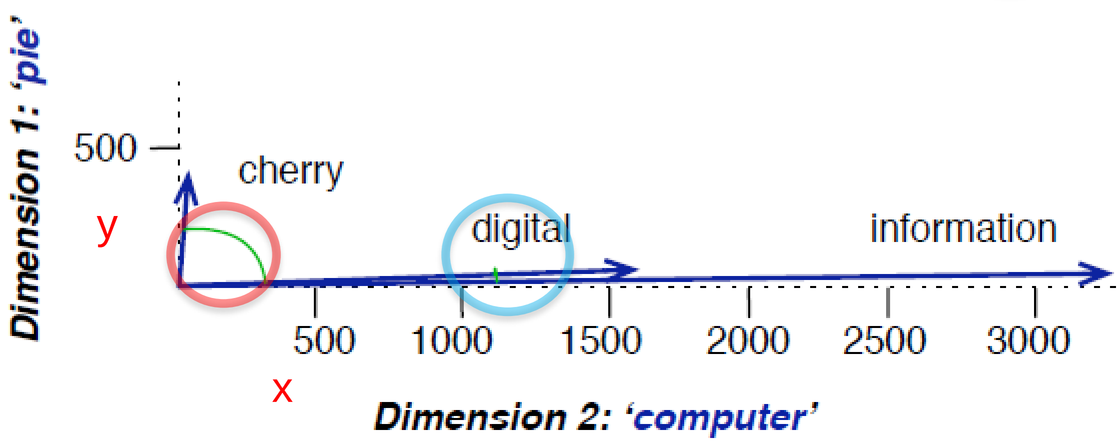
- An example (Jurafsky 2021)

Occurrence table over the dimensions of the columns

	y		x
	pie	**data**	**computer**
cherry	442	8	2
digital	5	1683	1670
information	5	3982	3325

$$\cos(\text{cherry}, \text{information}) = \frac{442 * 5 + 8 * 3982 + 2 * 3325}{\sqrt{442^2 + 8^2 + 2^2}\sqrt{5^2 + 3982^2 + 3325^2}} = .017$$

$$\cos(\text{digital}, \text{information}) = \frac{5 * 5 + 1683 * 3982 + 1670 * 3325}{\sqrt{5^2 + 1683^2 + 1670^2}\sqrt{5^2 + 3982^2 + 3325^2}} = .996$$



Semantic similarity
TF-IDF matrix

- TF from *term-frequency*
  - A Word occurring 100 times in a document is not 100 times more important than a word occurring only once
  - Calculate the **matrix** tf so:

    $$tf_{t,d} = \log_{10}(1 + occurrences(t, d))$$

    If $occurrences(t, d) = 0$ then $tf_{t,d} = 0$
- \- it's a hyphen, not a minus
- IDF from *inverse document frequency*
  - Gives a higher weight to words occurring only in some documents (valuable words for charactering)
  - Calculate the **vector** idf_t (it is not a matrix) so:

$$idf_t = \log_{10}\left(\frac{N}{df_t}\right), \text{ where } \begin{cases} N & \text{number of documents in the corpus} \\ df_t & \text{number of documents containing } t \end{cases}$$

Semantic similarity
TF-IDF matrix

Example with plays by Shakespeare

Matrix *term-frequency*
(the cell's value is the number of occurrences of the term (word in the row) in the document of the column). Let's compute: $tf_{t,d} = \log_{10}(1 + occurrencies(t,d))$

Occurrence table

	As You Like It	Twelfth Night	Julius Caesar	Henry V
battle	1	0	7	13
good	114	80	62	89
fool	36 *log(1+36) = 1,568*	58	1	4
wit	20 *log(1+20) = $\boxed{1.322}$*	15	2	3

Vector *df*
(number of documents containing the word)

Word	df	idf
Romeo	1	1.57
salad	2	1.27
Falstaff	4	0.967
forest	12	0.489
battle	21	0.246
wit	34	$\boxed{0.037}$
fool	36	0.012
good	37	0
sweet	37	0

log(N/df)

= log(37/1) = 1.57
= log(37/2) = 1.27
= log(37/4) = 0.967
etc…

$idf_t = \log_{10}\left(\frac{N}{df_t}\right)$

We know that N (number of plays) is 37

$$\boxed{\text{TF-IDF matrix: } w_{t,d} = tf_{t,d} * idf_t}$$

	As You Like It	Twelfth Night	Julius Caesar	Henry V
battle	0.074	0	0.22	0.28
good	0	0	0	0
fool	0.019	0.021	0.0036	0.0083
wit	$\boxed{0.049}$	0.044	0.018	0.022

$$w_{wit,As\,You\,Like\,It} = tf_{wit,As\,You\,Like\,It} * idf_{wit} = \boxed{1.322} * \boxed{0.037} = \boxed{0.049}$$

Semantic similarity
TF-IDF matrix

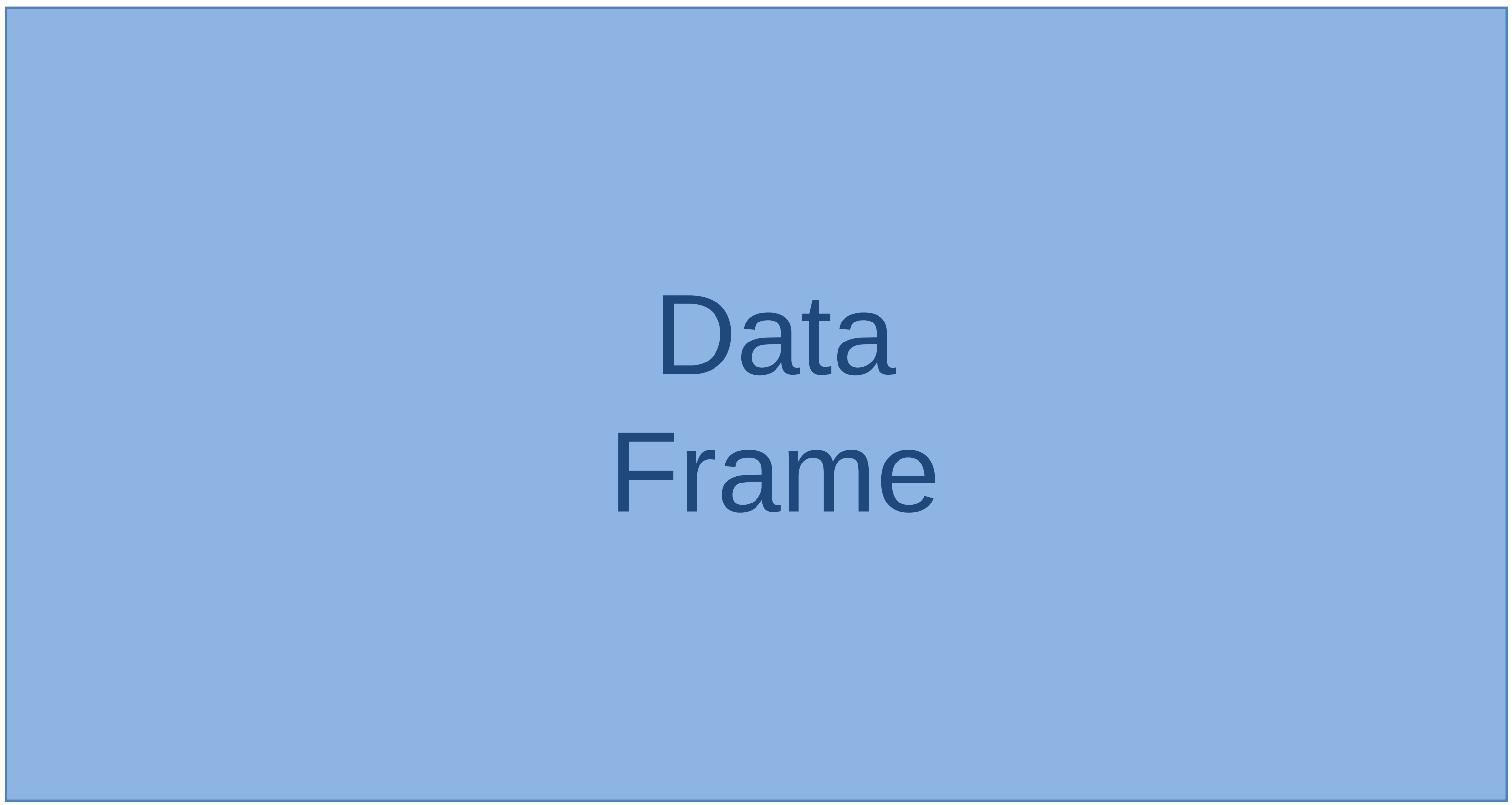
- From R
  - The `quanteda` package computes the tf-idf matrix from a given corpus
    - Function `textstat_simil()` returns a matrix of similarities
    - Function `textstat_dist()` returns a matrix of distances
      - With distances you can do dendrograms

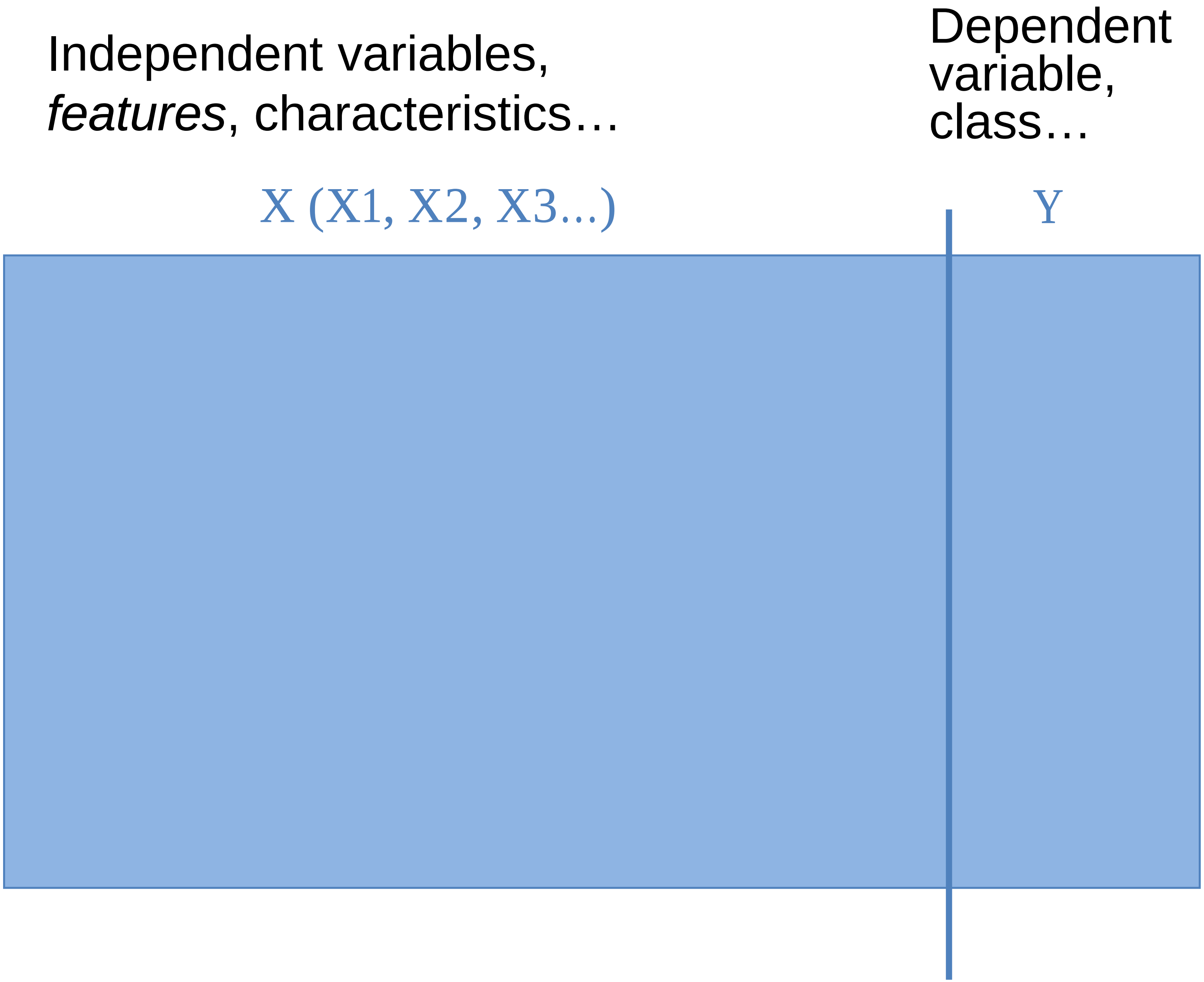
Table of Contents

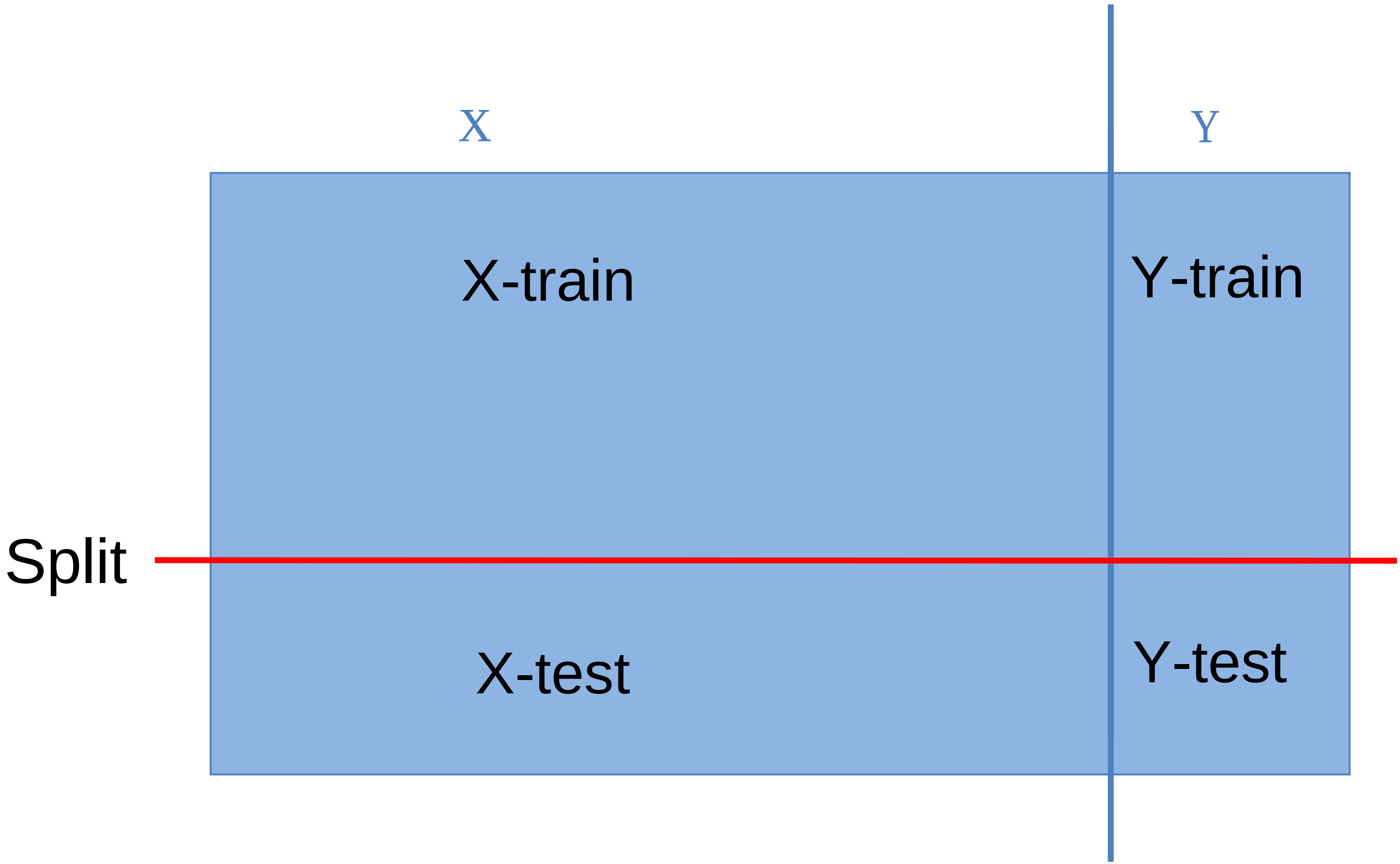
1. Part of Speech
2. Sparse Vector models
3. TF-IDF
4. **Document classification**
5. Hands-on 2

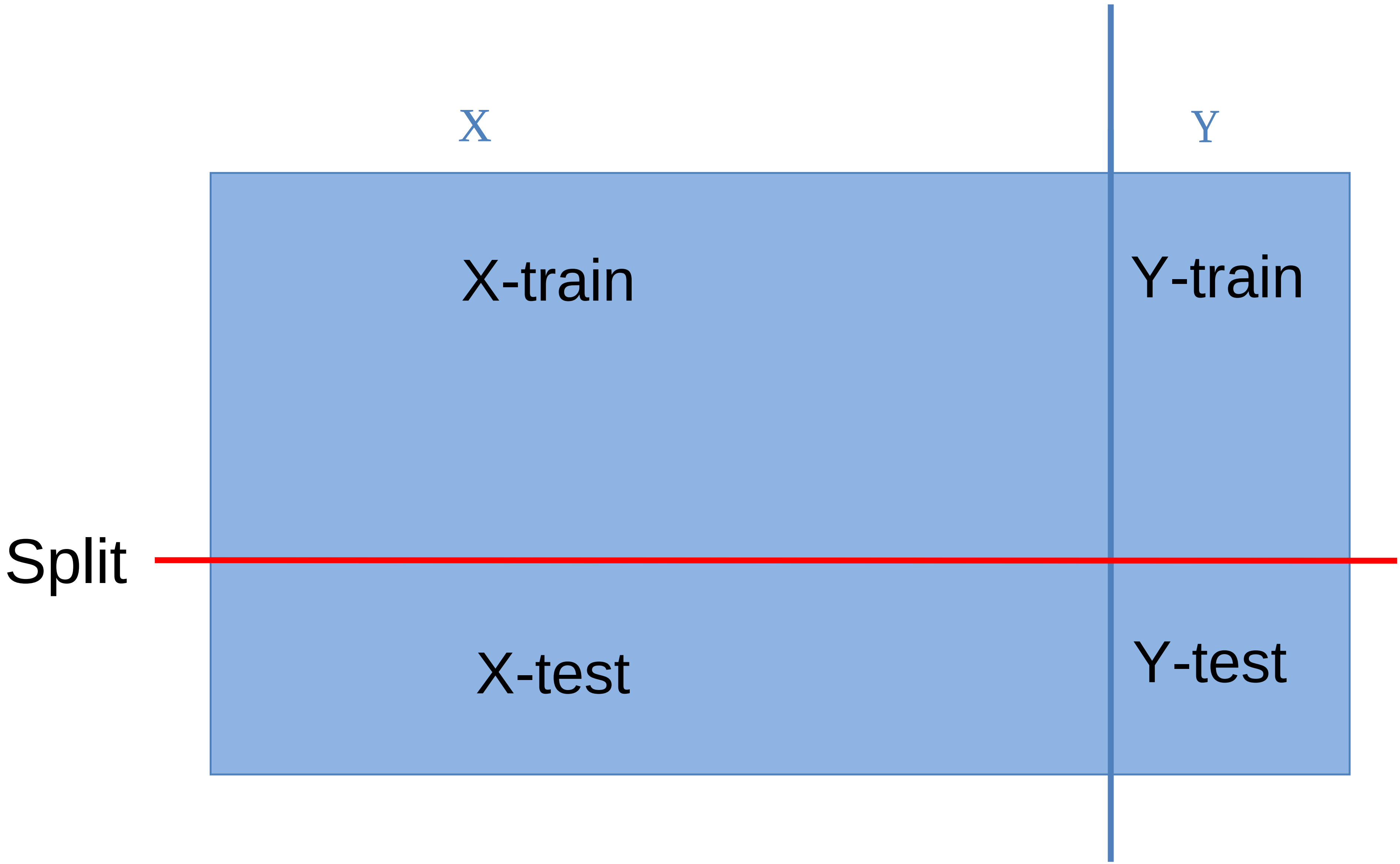
DOCUMENT CLASSIFICATION

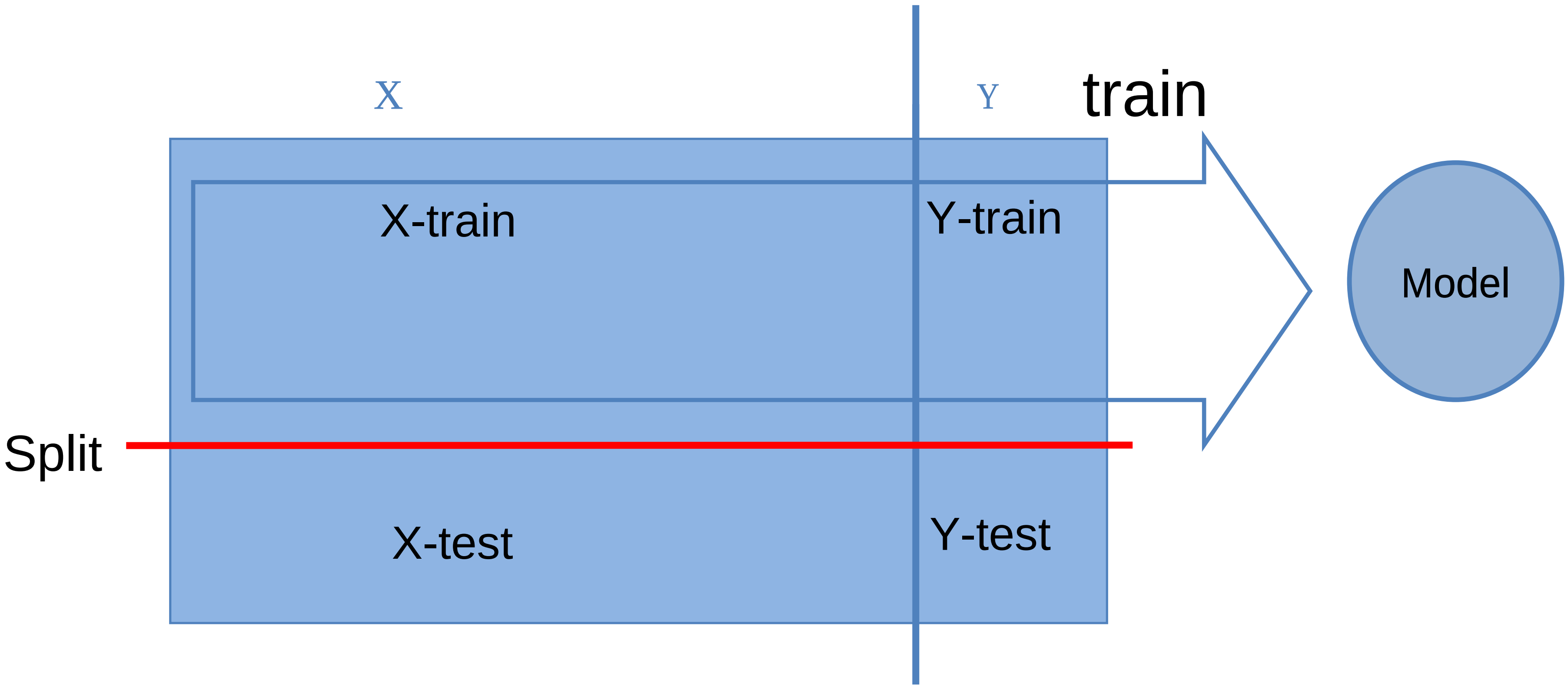
Dataset

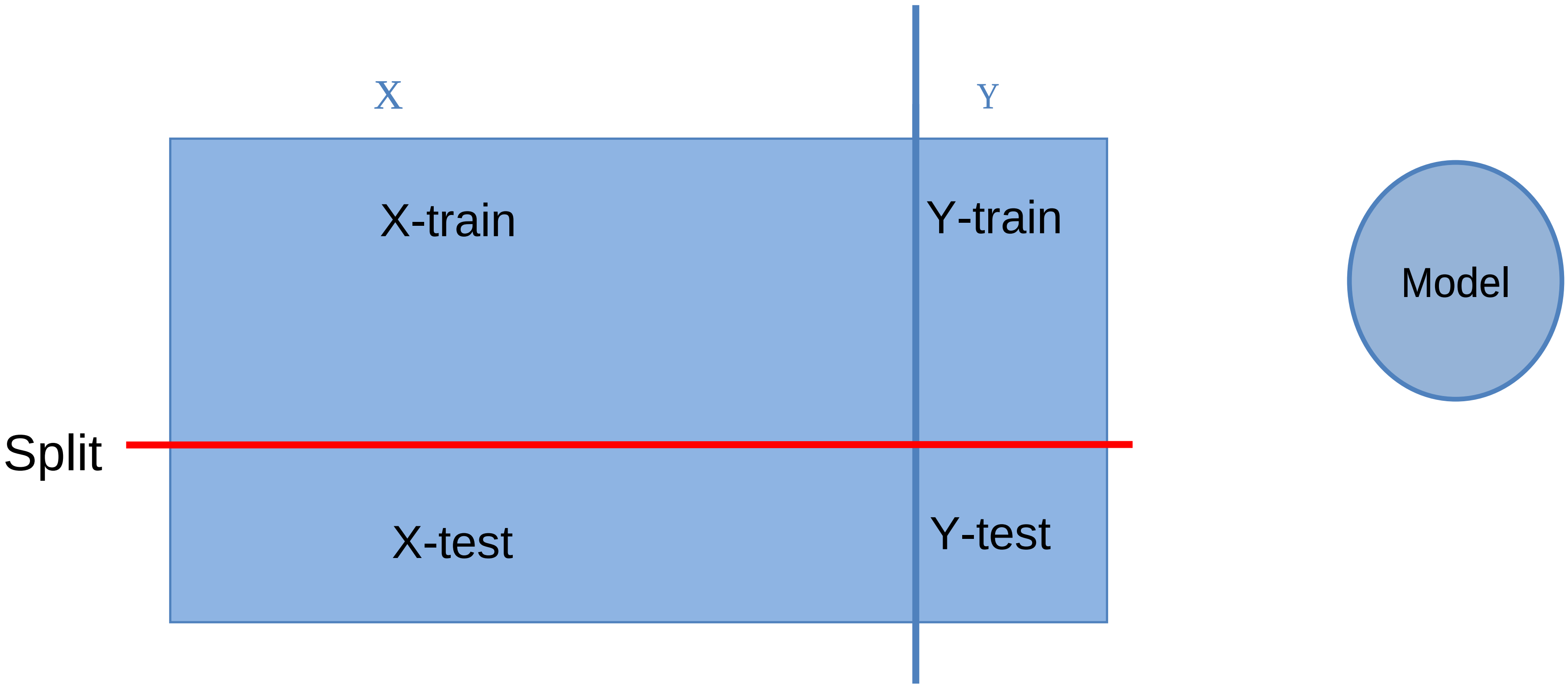


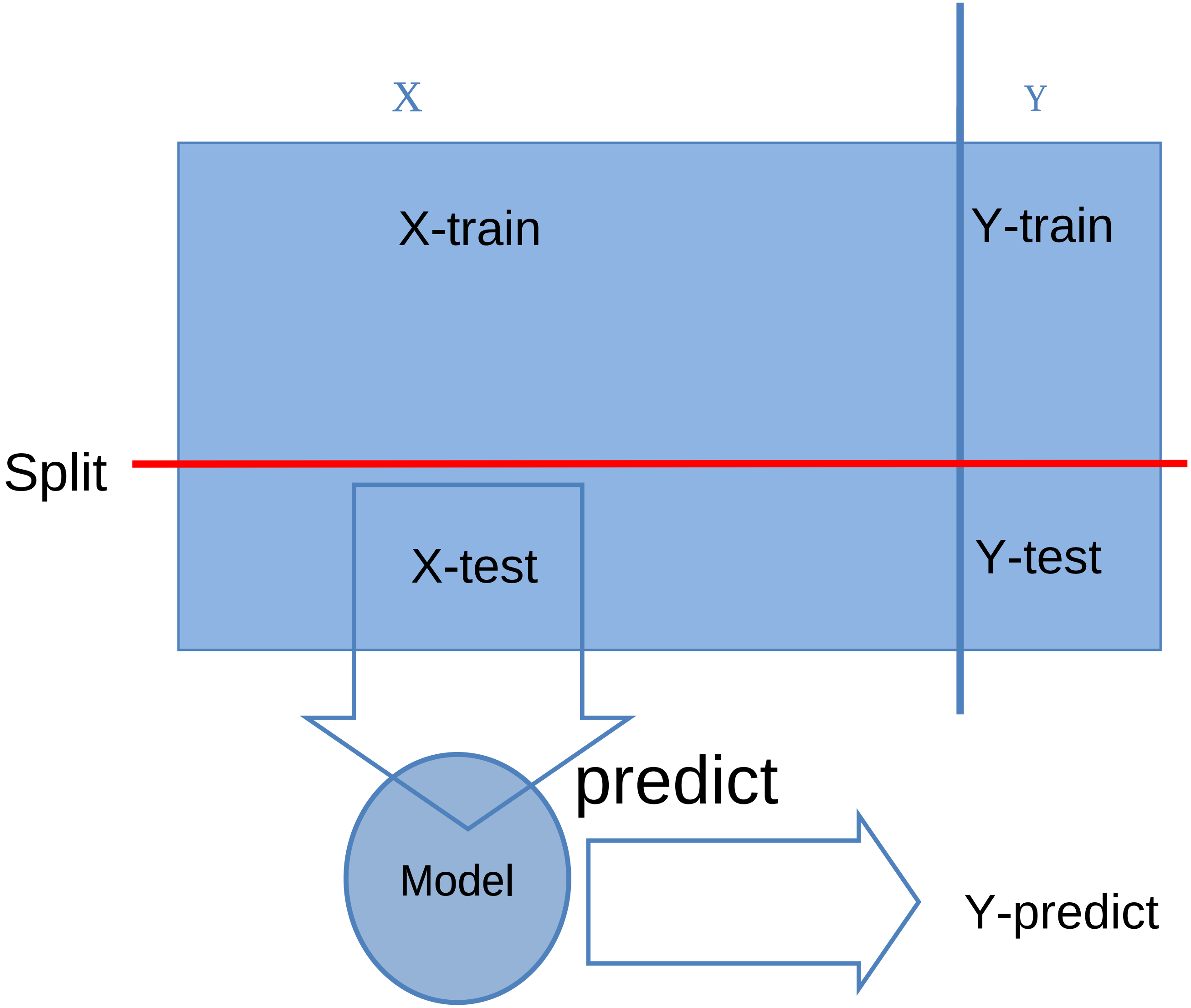

Independent variables,
features, characteristics…
Dependent variable, class…
X (X1, X2, X3…)
Y

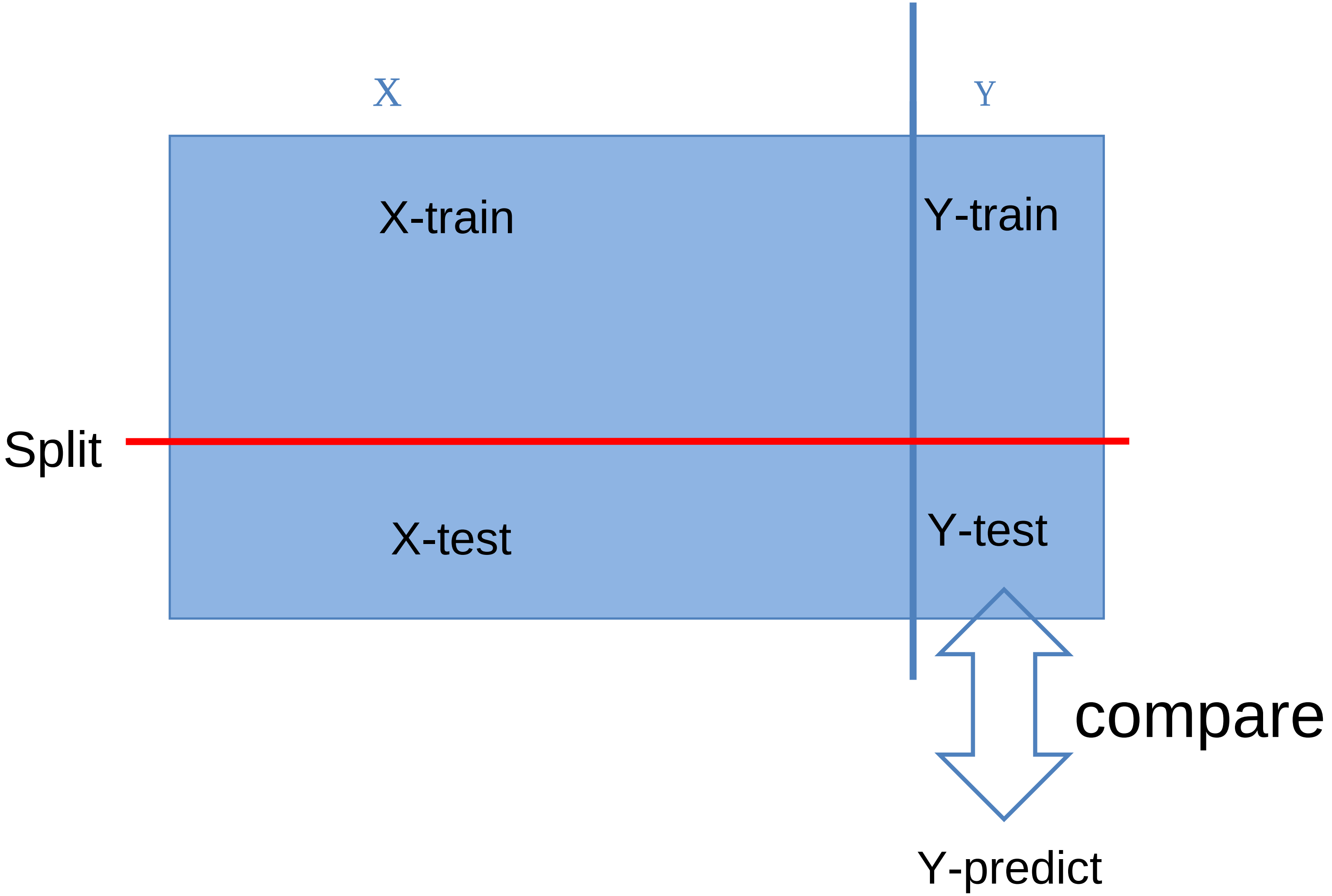

X
Y
X-train
Y-train
Split
X-test
Y-test


X
Y
X-train
Y-train
Split
X-test
Y-test


X
Y
train
X-train
Y-train
Model
Split
X-test
Y-test

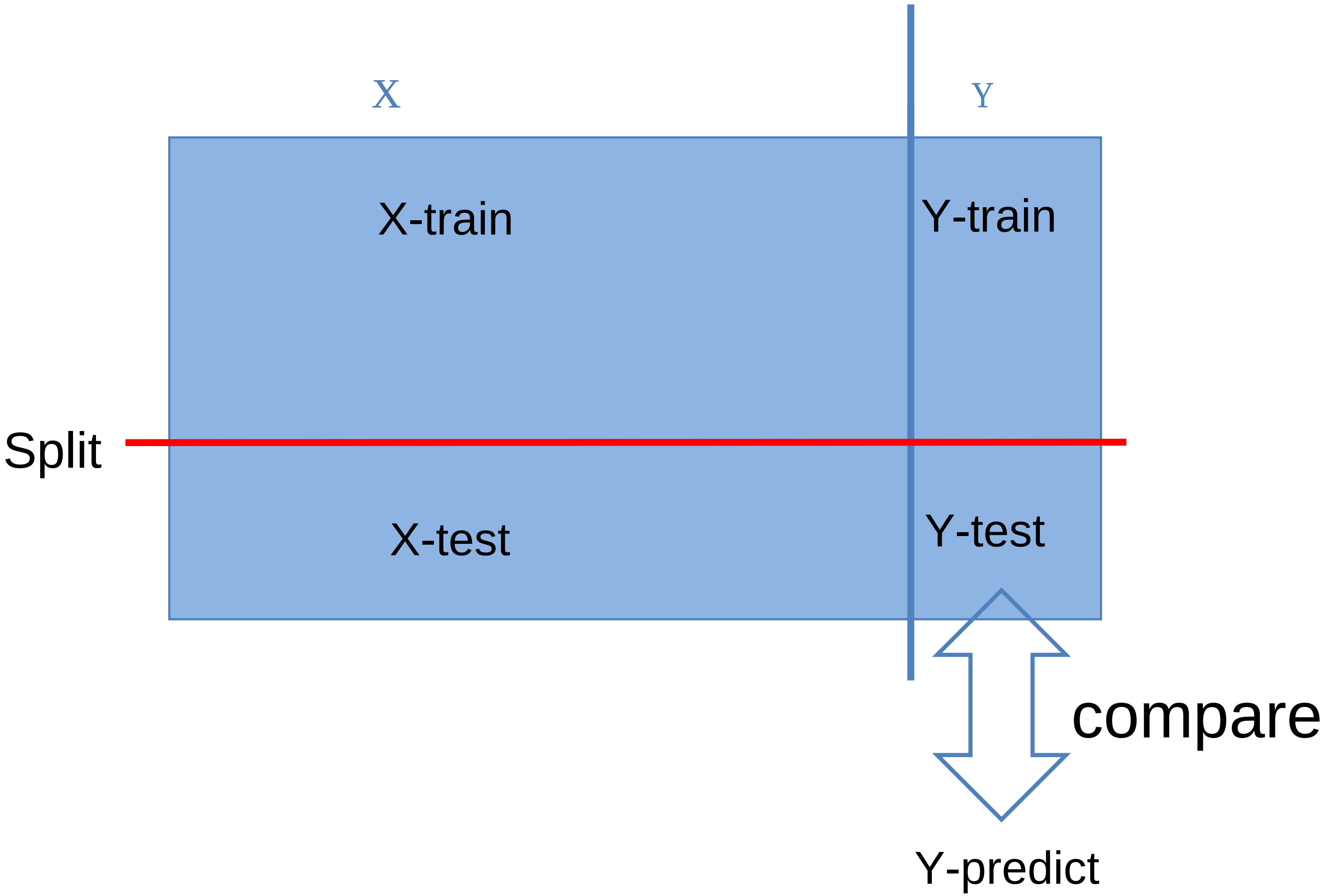

X
Y
X-train
Y-train
Model
Split
X-test
Y-test


X
Y
X-train
Y-train
Split
X-test
Y-test
predict
Model
Y-predict


X
Y
X-train
Y-train
Split
X-test
Y-test
compare
Y-predict

Classification

MODELS: EVALUATION


X
Y
X-train
Y-train
Split
X-test
Y-test
compare
Y-predict

How do we measure if prediction (**binary**) is good?

		Y-test	
		pos	neg
Y-predict	pos		
	neg		

Y-test

compare

Y-predict

How do we measure if prediction (**binary**) is good?

We use the **confusión matrix**

		Y-test	
		pos	neg
Y-predict	pos	True positives (*TP*)	False positives (*FP*)
	neg	False negativos (*FN*)	True negatives (*TN*)

Y-test

compare

Y-predict

How do we measure if prediction (**binary**) is good?

We use the **confusión matrix**,
and **calcule *p* (precision) and *r* (*recall*)**

		Y-test	
		pos	neg
Y-predict	pos	True positives (*TP*)	False positives (*FP*)
	neg	False negatives (*FN*)	True negatives (*TN*)

$$p = \frac{TP}{TP + FP}$$

$$r = \frac{TP}{TP + FN}$$

An example with spam detection:

Each email is classified as ***spam*** or ***normal***
The confusion matrix is this:

Total emails: 60+50+30+200 = 340

We know (Y-test) than 60+30=90 are normal

50+200=250 are spam

Our predictor (Y-predict) says that 60+50 = 110 are normal

30+200 = 230 are spam

		Y-test	
		pos (normal)	neg (spam)
Y-predict	pos (normal)	True positives (*TP*) = 60	False positives (*FP*) = 50
	neg (spam)	False negatives (*FN*) = 30	True negatives (*VN*)=200

$$p = \frac{TP}{TP + FP} = \frac{60}{60 + 50} = 0.54\ (54\%)$$

$$r = \frac{TP}{TP + FN} = \frac{60}{60 + 30} = 0.66 = 66\%$$

If we have more than two classes (is not binary):

Example: each email is classified as ***spam***, ***normal***, or as ***urgent***.

		Y-test		
		urgent	normal	spam
Y-predict	urgent	8	10	1
	normal	5	60	50
	spam	3	30	200

$$p_{urgent} = \frac{8}{8+10+1}$$

$$p_{normal} = \frac{60}{5+60+50}$$

$$p_{spam} = \frac{200}{3+30+200}$$

$$r_u = \frac{8}{8+5+3} \quad r_n = \frac{60}{10+60+30} \quad r_s = \frac{200}{1+50+200}$$

Example from Jurafsky 2021

Hands-on 2

1.- to learn how to perform different annotations (word, sentence, part-of-speech) over text documents

Materials:

R script at: http://rpubs.com/rgcmme/IS-HO2

Tasks:

Annotate corpus

2.- work with dictionaries using aspell

Materials:

Aspell manual

Post by Typethinker (usage of aspell and affixes)

Deliverable ideas:

For a given language (e.g. English, Spanish), provide:

- *List of nouns (masculine, singular)*
- *List of verbs (infinitive)*

3.- More in Moodle (to appear)

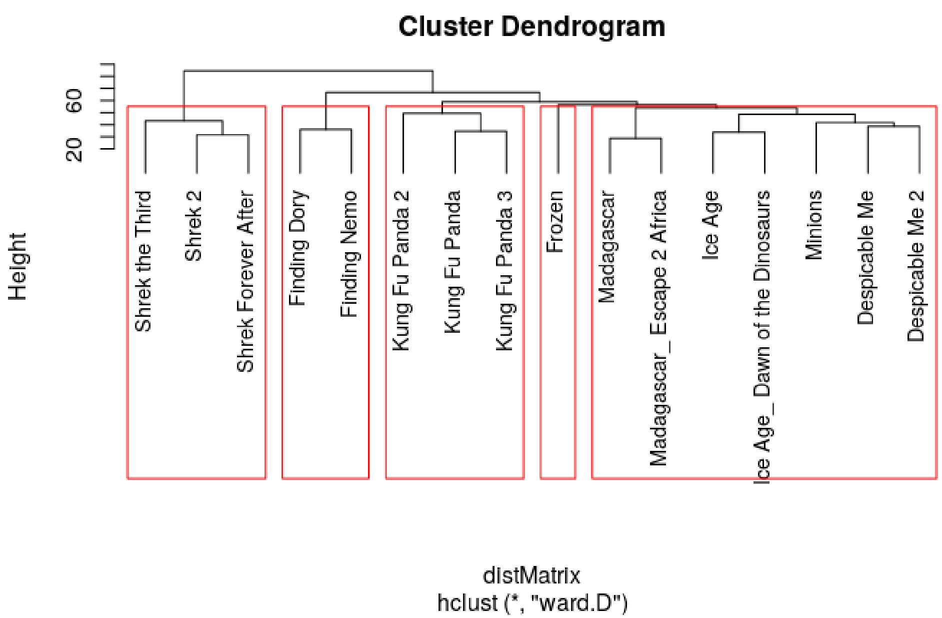
Text classification

From R

- Dendrograms with `hclust()`

```
m  <- as.matrix(dtm)
distMatrix <- dist(m, method="euclidean")

groups <- hclust(distMatrix, method="ward.D")
plot(groups, cex=0.9, hang=-1)
rect.hclust(groups, k=5)
```



Source: here

Text classification

From R

- Package quanteda.textmodels (in CRAN). Has 8 basic models for quanteda corpora
  - The simplest is the Naive Bayes classifier
    - Function `textmodel_nb()`.With 2 types of distributions:
      - » Multinomial
      - » Bernoulli
  - A more advanced (SVM)
    - Function `textmodel_svm()`
- Package quanteda.classifiers (**no** in CRAN). Advanced models for quanteda corpora
  - Two classifiers (using neuronal networks)
    - Multilevel perceptron network
    - Convolutional neural network + LSTM model fitted to word embeddings

Questions?

Course: Intelligent Systems

Unit 4: Language Technologies

Language technologies
Part 2

Mariano Rico
2021
Technical University of Madrid