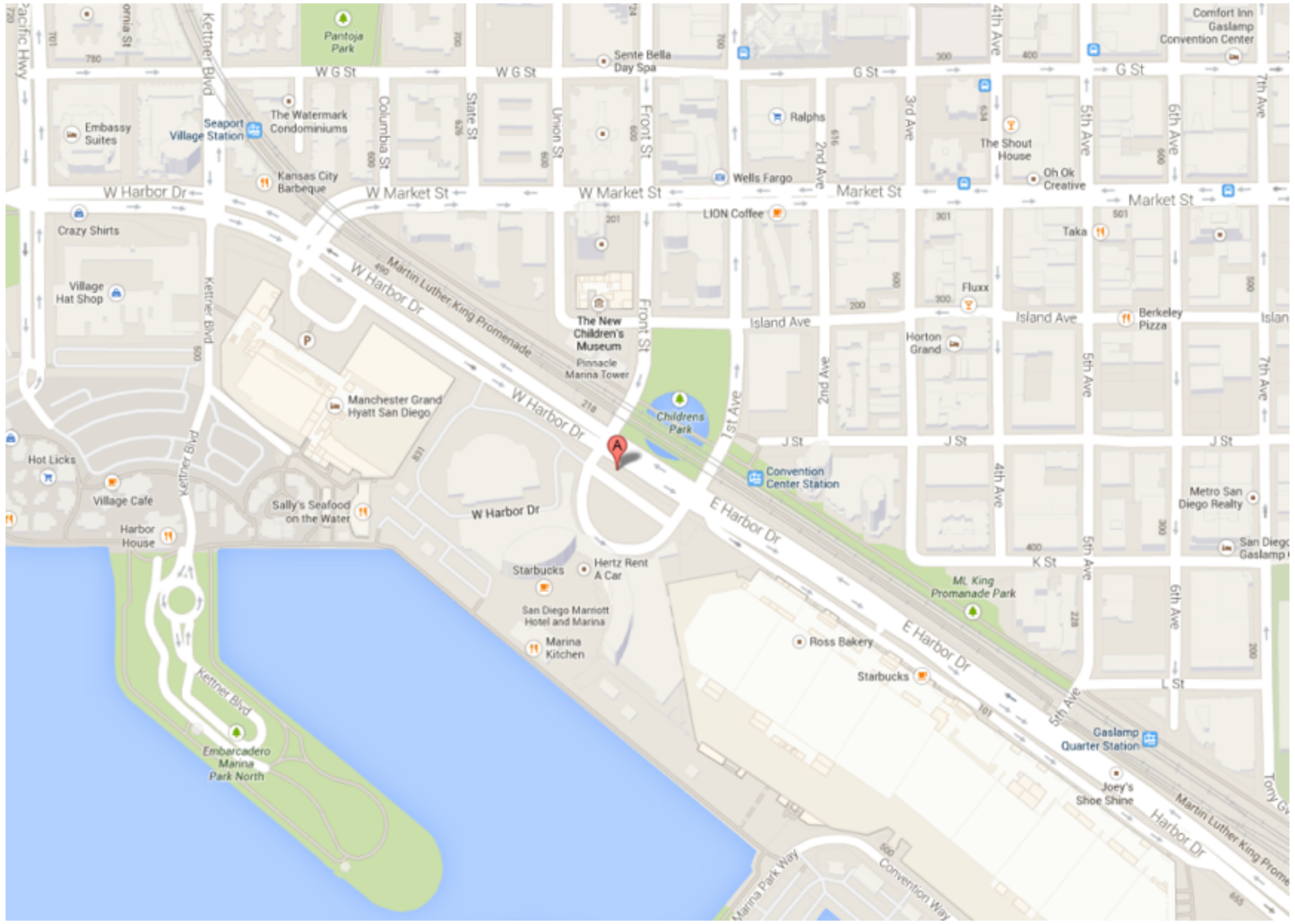
INTRO TO TEXT MINING: BAG OF WORDS

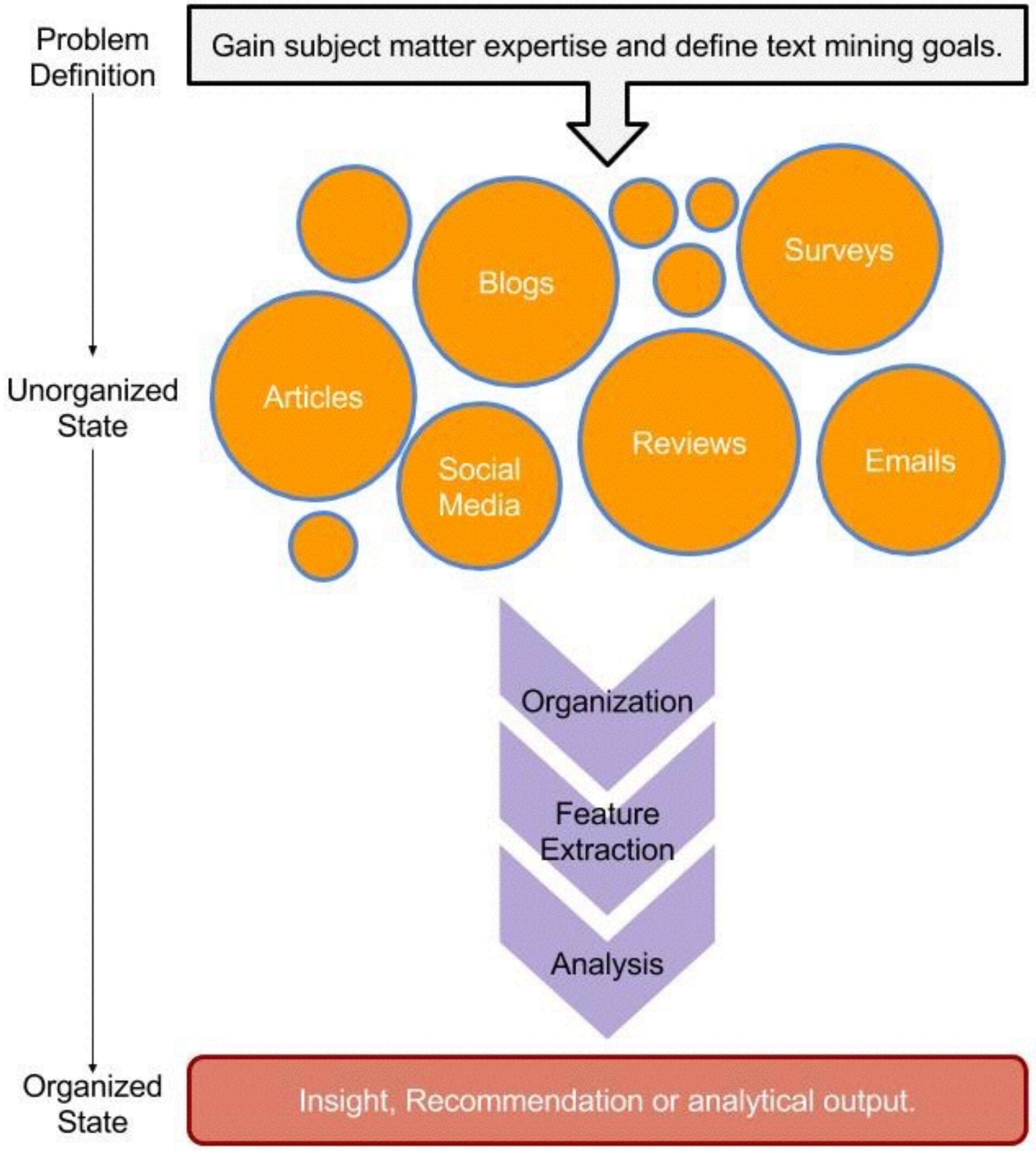
What is text mining?

What is text mining?

The process of distilling actionable insights from text



Text mining workflow



1 - Problem definition & specific goals

2 - Identify text to be collected

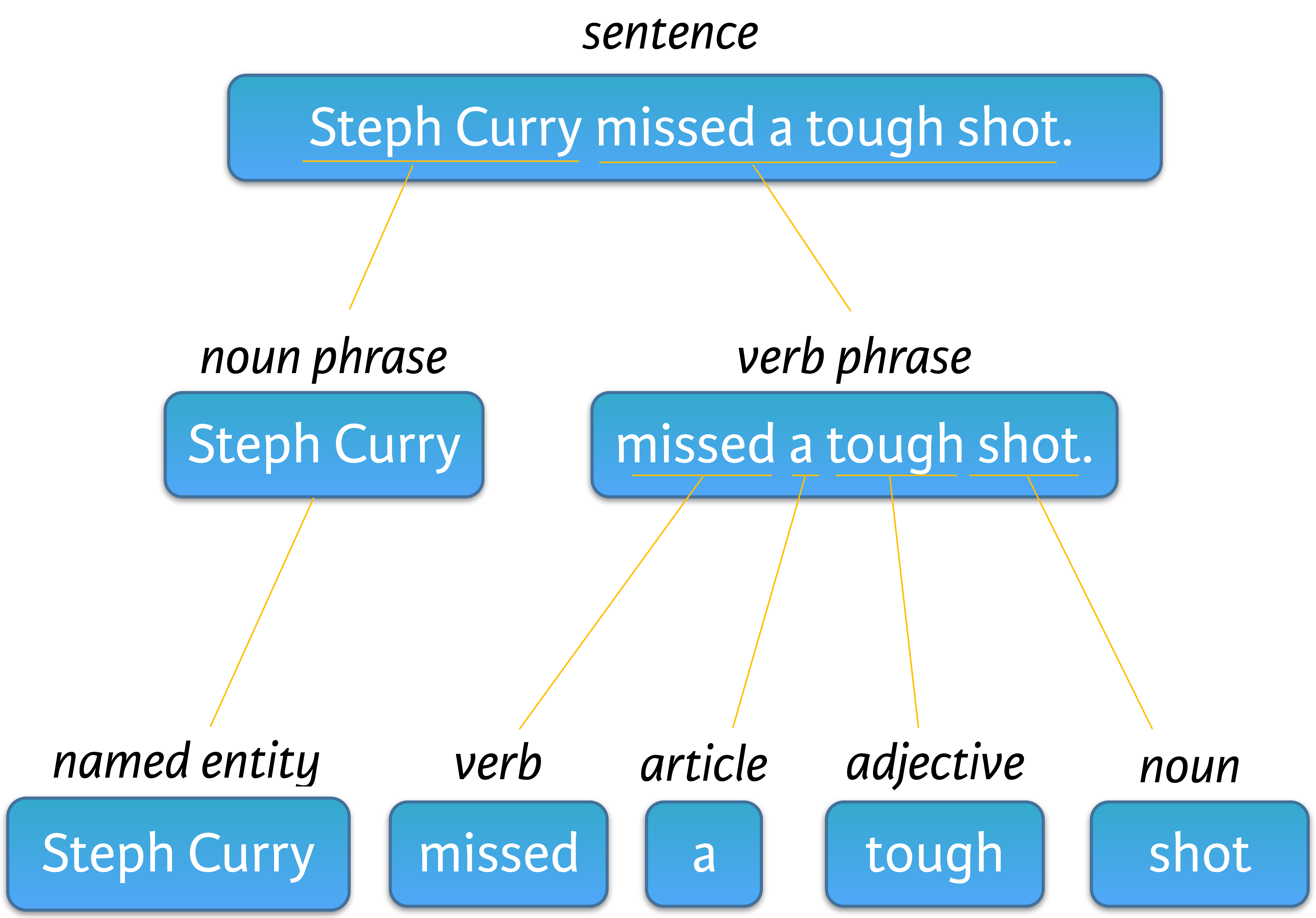
3 - Text organization

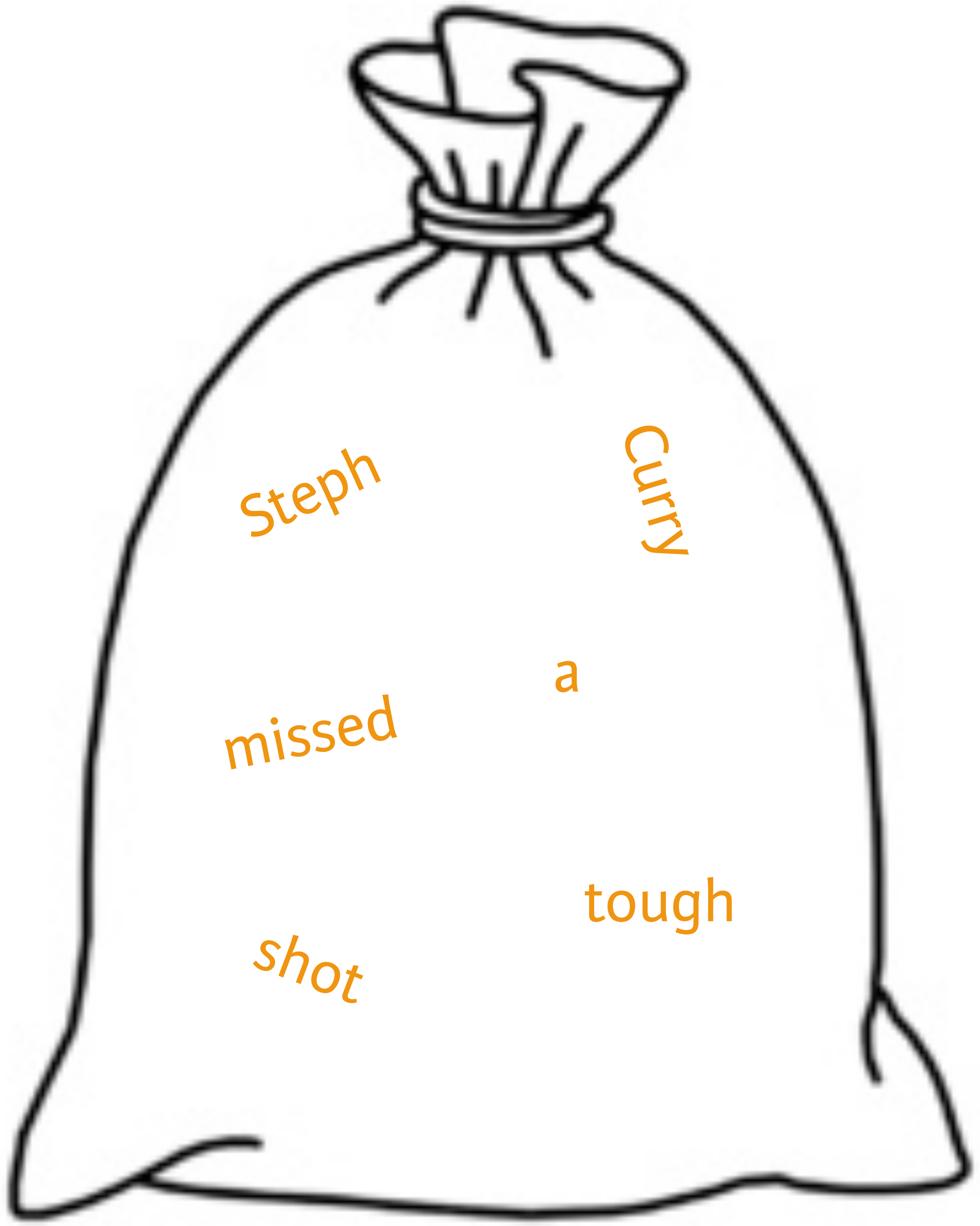
4 - Feature extraction

5 - Analysis

6 - Reach an insight,
recommendation or output

Semantic parsing vs. bag of words





Let’s practice!

Getting started

Building our first corpus

```
> # Load corpus
> coffee_tweets <- read.csv("coffee.csv", stringsAsFactors = FALSE)

> # Vector of tweets
> coffee_tweets <- coffee_tweets$text

> # View first 5 tweets
> head(coffee_tweets, 5)
[1] "@ayyytylerb that is so true drink lots of coffee"
[2] "RT @bryzy_brib: Senior March tmw morning at 7:25 A.M. in the
SENIOR lot. Get up early, make yo coffee/breakfast, cus this will
only happen ?"
[3] "If you believe in #gunsense tomorrow would be a very good day
to have your coffee any place BUT @Starbucks Guns+Coffee=#nosense
@MomsDemand"
[4] "My cute coffee mug. http://t.co/2udvMU6XIG"
[5] "RT @slaredo21: I wish we had Starbucks here... Cause coffee
dates in the morning sound perff!"
```

Let’s practice!

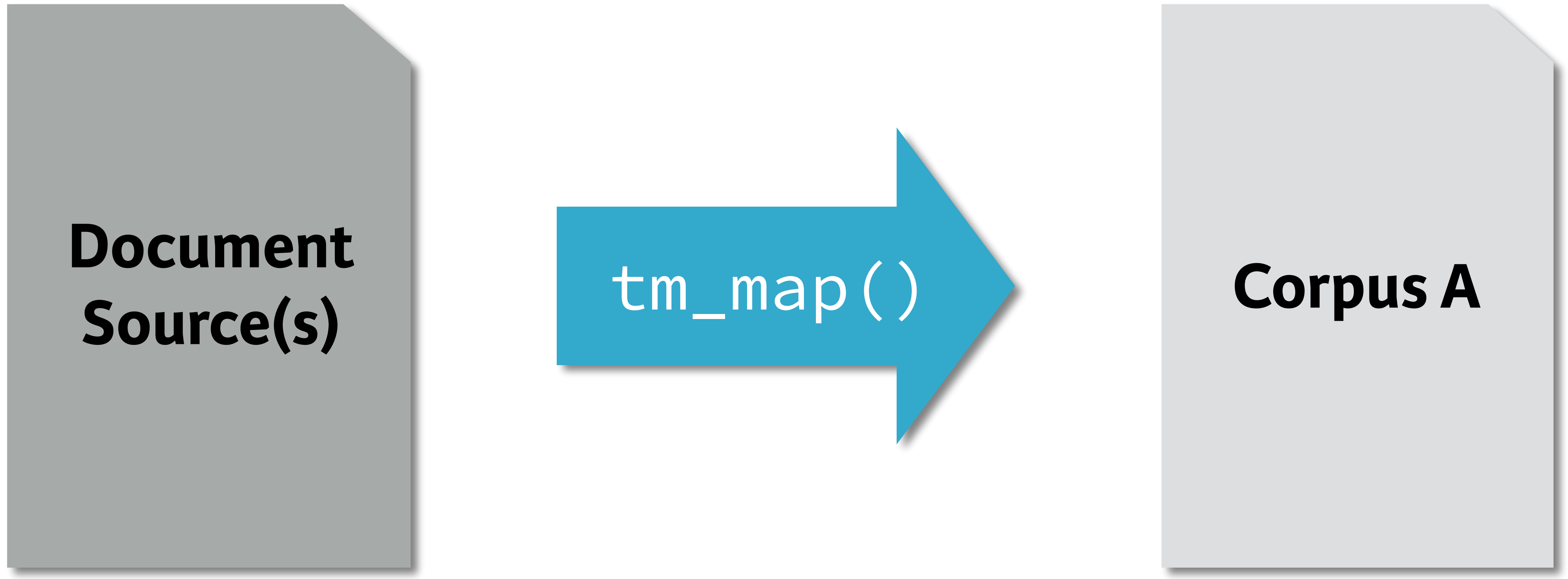
INTRO TO TEXT MINING: BAG OF WORDS

Cleaning and preprocessing text

Common preprocessing functions

TM Function	Description	Before	After
tolower()	Makes all text lowercase	Starbucks is from Seattle.	starbucks is from seattle.
removePunctuation()	Removes punctuation like periods and exclamation points	Watch out! That coffee is going to spill!	Watch out That coffee is going to spill
removeNumbers()	Removes numbers	I drank 4 cups of coffee 2 days	I drank cups of coffee days ago.
stripWhiteSpace()	Removes tabs and extra spaces	I like coffee.	I like coffee.
removeWords()	Removes specific words (e.g. "the", "of") defined by the data scientist	The coffee house and barista he visited were nice, she said hello.	The coffee house barista visited nice, said hello.

Preprocessing in practice



```
> # Make a vector source: coffee_source
> coffee_source <- VectorSource(coffee_tweets)

> # Make a volatile corpus: coffee_corpus
> coffee_corpus <- VCorpus(coffee_source)

> # Apply various preprocessing functions
> tm_map(coffee_corpus, removeNumbers)
> tm_map(coffee_corpus, removePunctuation)
> tm_map(coffee_corpus, content_transformer(replace_abbreviation))
```

Another preprocessing step: word stemming

```
> # Stem words
> stem_words <- stemDocument(c("complicatedly", "complicated",
                               "complication"))
> stem_words
[1] “complic” “complic” “complic”

> # Complete words using single word dictionary
> stemCompletion(stem_words, c("complicate"))
     complic      complic      complic
"complicate" "complicate" "complicate"

> # Complete words using entire corpus
> stemCompletion(stem_words, my_corpus)
```

Let’s practice!

The TDM & DTM

TDM vs. DTM

	Tweet 1	Tweet 2	Tweet 3	...	Tweet N
Term 1	0	0	0	0	0
Term 2	1	1	0	0	0
Term 3	1	0	0	0	0
...	0	0	3	1	1
Term M	0	0	0	1	0

Term Document Matrix (TDM)

	Term 1	Term 2	Term 3	...	Term M
Tweet 1	0	1	1	0	0
Tweet 2	0	1	0	0	0
Tweet 3	0	0	0	3	0
...	0	0	0	1	1
Tweet N	0	0	0	1	0

Document Term Matrix (DTM)

```
> # Generate TDM
> coffee_tdm <- TermDocumentMatrix(clean_corp)

> # Generate DTM
> coffee_dtm <- DocumentTermMatrix(clean_corp)
```

Word Frequency Matrix (WFM)

```
> # Load qdap package
> library(qdap)

> # Generate word frequency matrix
> coffee_wfm <- wfm(coffee_text$text)
```

	Tweet 1
Term 1	0
Term 2	1
Term 3	1
...	0
Term M	0

Word Frequency Matrix (WFM)

Let’s practice!