1495 UNIVERSITY OF ABERDEEN | CELEBRATING 525 YEARS 1495 – 2020

ABERDEEN 2040

Text Mining – 1

Data Mining & Visualisation

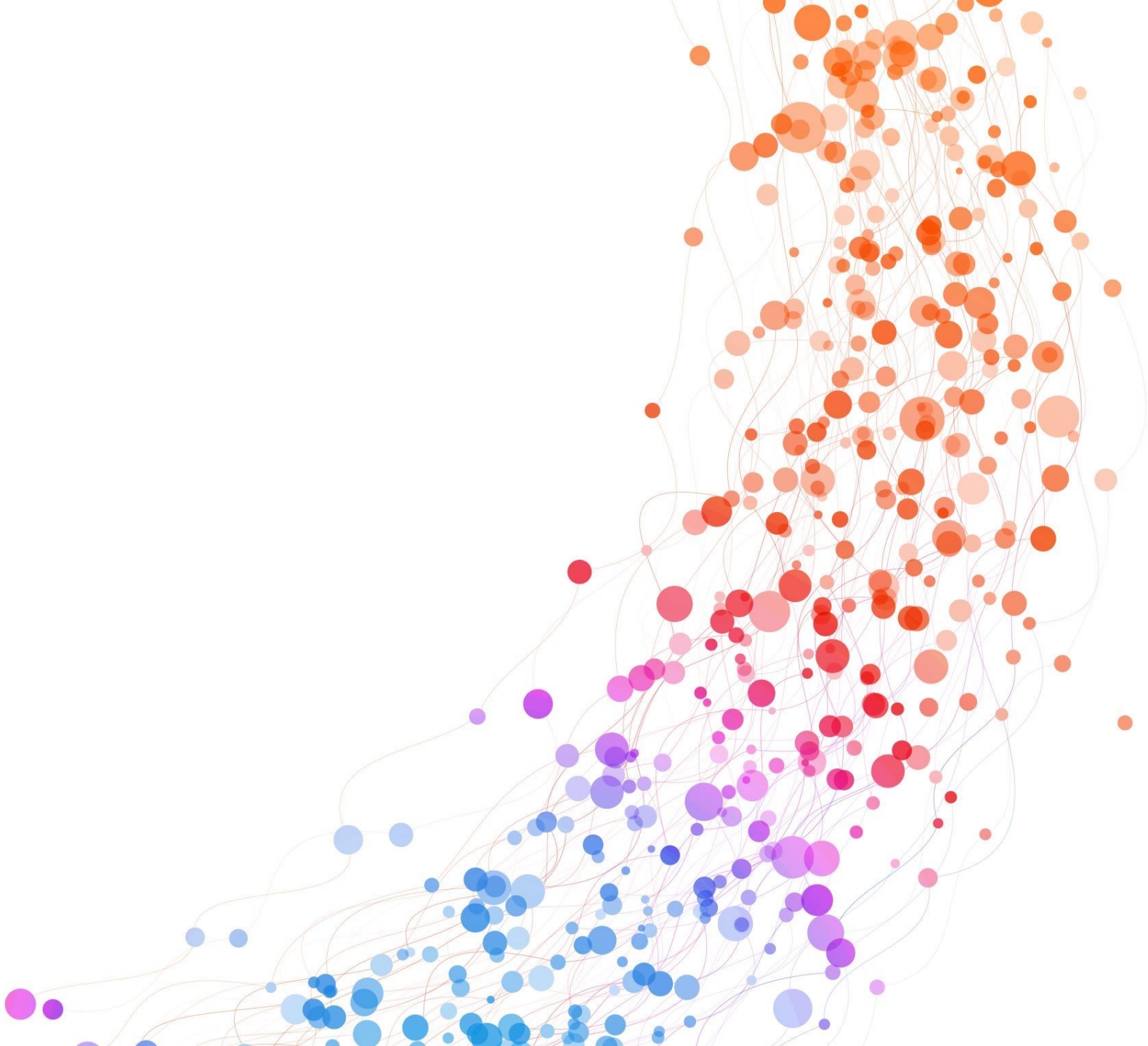
Lecture 27

2025

Today…

- Overview of Text Mining
- Data preprocessing for text
- Sentiment Analysis

Text Mining



Overview of Text Mining

Much like time series data, textual data appears everywhere.

Think about the *apartments.csv* dataset on MyAberdeen, for instance.

This contains many advert listings for apartments in the US.

Overview of Text Mining

“Welcome home to an apartment community located in the heart of it all. This lush community offers one and two beds apartment homes as well as sheltered parking.

Located just off Campbell and Coit, this residential setting offers the finest shops, countless restaurants and Addison nightlife fingertips. So, come home to and discover the lifestyle you deserve.

DeluxeUnit, Fireplace, Select K/B upgrades, STORAGE, Stacked Washer-dryer, Pool, Cloths washer and drier Connections, Bar with sink. Ok for pets

Comments: Pet Charges and Deposits Vary by Property. Assistance animals are always welcome without deposit or fee. Restrictions: Pet Types Allowed: Dogs, Cats, Fish, Caged Birds. Price range: $898 - $1, 388. More units available: two Bd / 2 Bedrooms 1,000 sq. feet for $1,503/mo | two Bd / 1 Bedrooms 890 square ft for $1,563/mo | one Bd / 1 Bedrooms 630 square feet for $948/mo | two Bd / 2 Bedrooms 1,000 sq. feet for $1,308/mo | two Bd / 2 Bedrooms 1,000 sq-ft for $1,708/mo | two Bd / 1 Bedrooms 890 sq.”

Overview of Text Mining

Within *just one column* of the apartments data is an incredibly rich source of information…

…Now imagine the amount of information we might be able to extract from entire documents, books, websites, etc.

Overview of Text Mining

‘Text mining’ refers to approaches which look to make sense of textual data or collections of text.

This might involve specific algorithms to match, rank, or classify the information contained within…

…or, it might involve *broader* strategies for analysing and finding patterns in structured and unstructured text.

Text Mining – Applications

There are many applications of text mining. E.g.:

- Evaluating strength of sentiment of text (sentiment analysis)
- Querying within collections of text (information retrieval)
- Finding similar documents (document clustering)
- Extracting ‘themes’ from documents (topic modelling)
- Text comprehension, manipulation, or generation (NLP)

Text Mining – Challenges

However, note that text mining can suffer from many similar challenges to that faced by analysing time series data.

For one, think about the potential **complexity** and **dimensionality** of text mining e.g. a document or a book.

It is for this reason that, like time series, text mining is an ongoing area of active research, and could easily fill a course syllabus by itself.

Text Mining – Challenges

For another challenge, text data can be messy and complex!

Speech and written language has developed and evolved over many thousands of years.

Think about how *sarcasm* might impact text mining, how *ambiguity* can raise issues, and how *context* can matter.

Text Mining

To simplify things, we'll give a very broad overview of a few areas of text mining.

However, before we jump into these, let's take a broader look at the theory of text mining.

Text Mining

Text mining is all about extracting useful and valuable information from (at times, vast quantities of) text.

In the context of data mining, this will often be to be able to *do* something with that information.

What we're going to be thinking about is how we can *make use of* that information…

…to *understand more* about the underlying data.

Overview of Text Mining

“Welcome home to an apartment community located in the heart of it all. This lush community offers one and two beds apartment homes as well as sheltered parking.

Located just off Campbell and Coit, this residential setting offers the finest shops, countless restaurants and Addison nightlife fingertips. So, come home to and discover the lifestyle you deserve.

DeluxeUnit, Fireplace, Select K/B upgrades, STORAGE, Stacked Washer-dryer, Pool, Cloths washer and drier Connections, Bar with sink. Ok for pets

Comments: Pet Charges and Deposits Vary by Property. Assistance animals are always welcome without deposit or fee. Restrictions: Pet Types Allowed: Dogs, Cats, Fish, Caged Birds. Price range: $898 - $1, 388. More units available: two Bd / 2 Bedrooms 1,000 sq. feet for $1,503/mo | two Bd / 1 Bedrooms 890 square ft for $1,563/mo | one Bd / 1 Bedrooms 630 square feet for $948/mo | two Bd / 2 Bedrooms 1,000 sq. feet for $1,308/mo | two Bd / 2 Bedrooms 1,000 sq-ft for $1,708/mo | two Bd / 1 Bedrooms 890 sq.”

What can we actually *do* with this information?

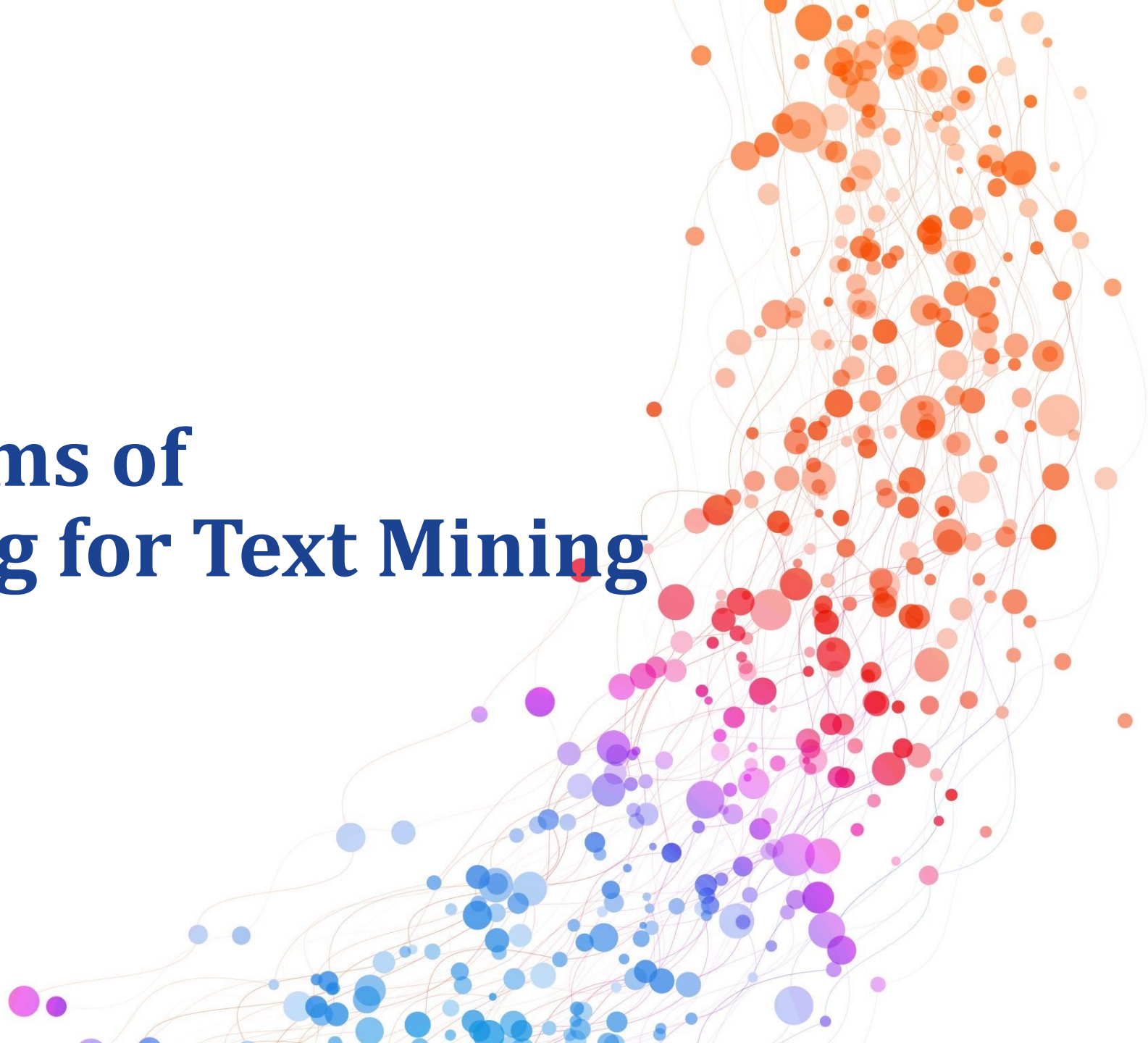
Text Mining

Let's think about text mining as a form of data preprocessing.

Maybe we want to parse the text to extract potential predictors, or undertake exploratory data analysis:

- Do apartments that allow dogs have higher rental prices? **(No)**
- Are particular words (e.g. 'spacious') more frequently associated with smaller apartments? **(No)**
- In which states are you more likely to find a property with a hot tub AND a fireplace? **(Texas)**

Common Forms of Preprocessing for Text Mining



Getting Started With Text Data

Let’s say we want to do some analysis on property listings.

We open up a dataset (e.g. *apartments.csv*), and are faced with a column comprising dense text data.

Where do we even begin?

Preprocessing Text Data

Part of the challenge you might come across is getting text data ready for further analysis.

Fortunately, there are a broad range of well-established techniques for preparing and preprocessing textual data.

We'll briefly mention some of these, but the possibilities are nearly endless…

Tokenization

For example, you might want to start by breaking down a given body of text or document into discrete elements.

Tokenization gives us a way to break down text into ‘tokens’:

[This email is spam] → {‘This’, ‘email’, ‘is’, ‘spam’}

Tokenization – Dummy Coding

[This email is spam] → {‘This’, ‘email’, ‘is’, ‘spam’}

For this tokenized data, we could use dummy coding (also known as one-hot encoding) outlined in the EDA lecture.

This would represent each word as a binary column, i.e. {0, 1}, indicating its presence.

Tokenization – Dummy Coding

[This email is spam] → {‘This’, ‘email’, ‘is’, ‘spam’}

id	text	this	email	is	spam	new	not	iPhone
1	This email is spam	1	1	1	1	0	0	0
2	This new email is not spam	1	1	1	1	1	1	0
3	This iPhone is new	1	0	1	0	1	0	1

Tokenization – Bag of Words Model

Or, we could instead choose to use the ‘bag of words’ model that we discussed in the Naïve Bayes lecture.

In this case, rather than binary variables, we might choose to instead use counts (accounting for the frequency of words).

Tokenization – Bag of Words Model

Note that the Bag of Words models accounts for word frequency:

id	text	this	email	is	spam	new	not	an	iPhone
1	This email is spam	1	1	1	1	0	0	0	0
2	This new email is not spam	1	1	1	1	1	1	0	0
3	This iPhone is new	1	0	1	0	1	0	0	1
4	This email is not an email	1	**2**	1	0	0	1	1	0

Stop Word Removal

We might remove ‘stop words’, i.e. words or grammar that may offer little semantic value.

By doing so, we can look to reduce the dimensionality of our data, thereby making it easier to analyse, at the cost of losing some information.

Stop Word Removal – Example

{‘Are’; ‘the’; ‘reviews’; ‘for’; ‘the’; ‘latest’; ‘iPhone’; ‘good’; ‘?’ }

→

{‘Are’; ‘reviews’; ‘latest’; ‘iPhone’; ‘good’}

Stop Word Removal

There are many built in dictionaries and functions that handle this for you, including within scikit-learn.

These will remove many common words, such as:

{‘a’, ‘am’, ‘an’, ‘and’, ‘are’, ‘as’, ‘at’, ‘be’, ‘by’, ‘do’, ‘eg’, ‘else, ‘etc’, …, }

 Full list: https://github.com/scikit-learn/scikit-learn/blob/main/sklearn/feature_extraction/_stop_words.py

Stemming

Stemming is the process of removing prefixes or suffixes from words, to reduce them to some ‘base’ form.

By doing so, we can get the underlying meaning of a particular word, regardless of how it is used grammatically.

Stemming – Example

For example:

{ ‘investing’, ‘invests’, ‘invested’, ‘investment’, ‘investments’, … }

→

{ ‘invest’ }

Lemmatization

Lemmatization is similar, but rather than focusing on prefixes or suffixes, lemmatizing uses a more sophisticated, semantic understanding of words to revert them to their ‘base form’.

It typically does this using a pre-defined dictionary.

Lemmatization – Examples

{ ‘Went’ } → { ‘Go’ }

{ ‘Better’ } → { ‘Good’ }

{ ‘Are’ } → { ‘Be’ }

Normalization

We might also want to ‘normalize’ words based on some criteria, e.g. case sensitivity.

By doing so, words that are the same, but capitalized in different ways, will not be treated differently.

Normalization – Example

{‘Are’; ‘the’; ‘reviews’; ‘for’; ‘the’; ‘latest’; ‘iPhone’; ‘good’; ‘?’ }

→

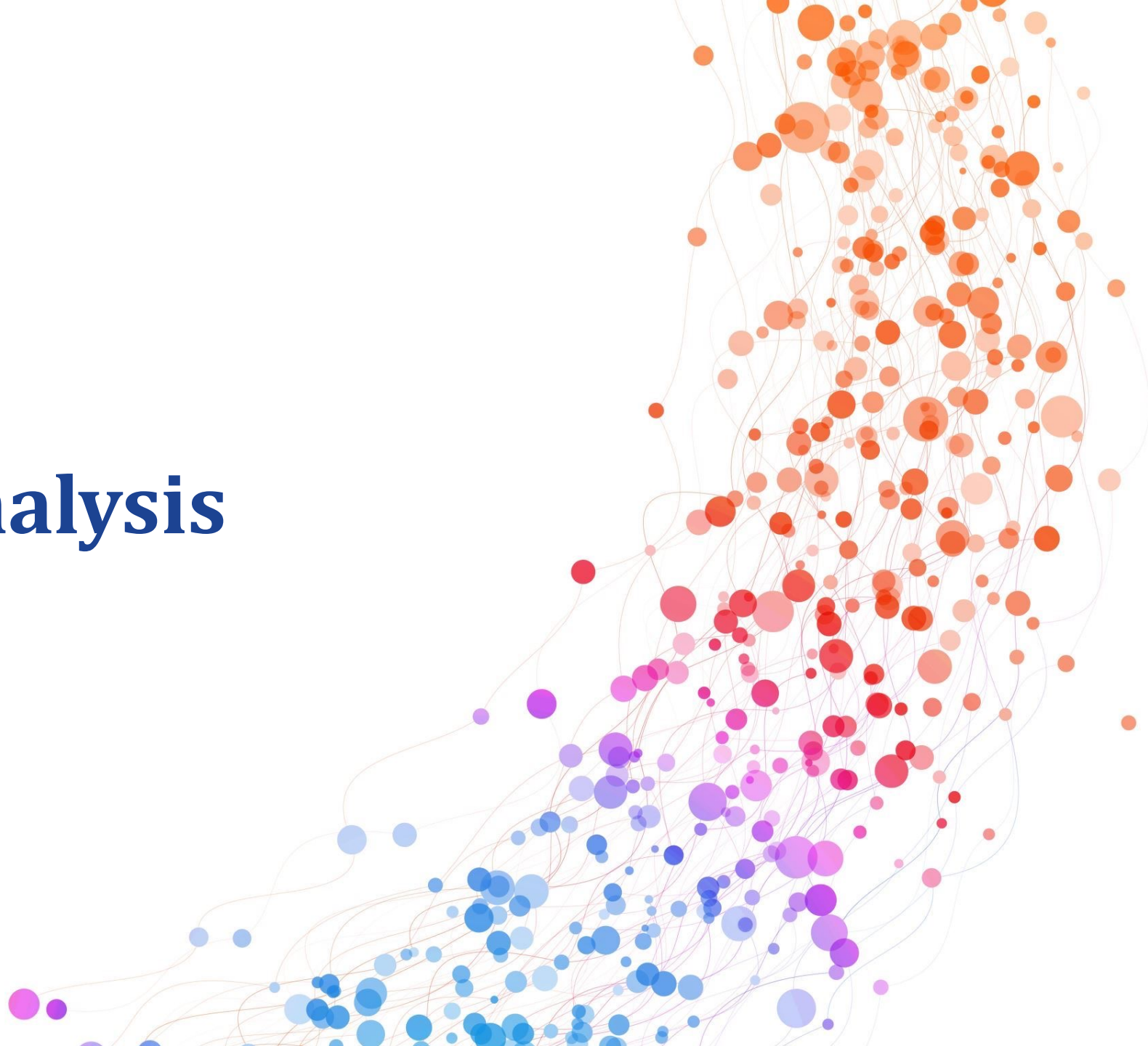
{‘are’; ‘the’; ‘reviews’; ‘for’; ‘the’; ‘latest’; ‘iphone’; ‘good’; ‘?’ }

Preprocessing Text Data

There are many ways to preprocess text data to prepare it for some further analysis.

Once we have done so, we might go on to apply some other interesting data mining techniques, such as sentiment analysis.

Sentiment Analysis



Sentiment Analysis

Sentiment analysis refers to a family of techniques for quantifying the ‘strength of feeling’ of text.

E.g., which do you think is a more *positive* review?

- “This apartment was great. I loved it and it made me happy.”
- “Worst apartment ever! I hated it! It should be condemned!”

Sentiment Analysis

Sentiment analysis assigns some score to each word.

‘Positive’ words result in a more positive score, whereas ‘negative’ words result in a more negative score.

- “great”; “loved”; “happy” +3
- “Worst”; “hated”; “condemned” -3

Sentiment Analysis

SA typically involves some kind of pre-defined dictionary, where each word is assigned a sentiment score.

Different words can have different (positive or negative) scores, indicating their ‘strength’.

…and sentences can have both positive and negative words, resulting in a summed total.

(illustrative values)

Word	Sentiment Score
Good	+1
Beautiful	+2
Amazing	+3
Bad	-1
Sad	-1
Abhorrent	-4
…	…

Sentiment Analysis

"This film had good and bad parts" [score: 0]

+1 -1

"The film was beautiful, but sad" [score: 1]

+2 -1

"This amazing film was really good" [score: 4]

+3 +1

(illustrative values)

Word	Sentiment Score
Good	+1
Beautiful	+2
Amazing	+3
Bad	-1
Sad	-1
Abhorrent	-4
…	…

Sentiment Analysis

Furthermore, some SA libraries can quantify other ‘kinds’ of emotions (not just positive/negative), with their own scores:

Word	Positivity	Negativity	Fear	Anger	Surprise	…
…						
…						
…						

Sentiment Analysis

Sentiment analysis can be a very useful preprocessing technique in itself, where the sentiment score of a given body of text might be used in some downstream analysis.

Hopefully you can see where this might have a wide range of cool applications in EDA/Data Mining.

Sentiment Analysis – Examples

Are my product's written reviews more positive than my competitor's

What are the themes of the most angry reviews?

Can we cluster the negative reviews together, based on some other characteristics of the customer?

Sentiment Analysis – Examples

How positive/negative are social media posts during the world cup?

Can we use this to predict which countries are winning and losing at any given point?

What is the common theme between the angriest posts?