Natural Language Processing Pt 2.

Delta Analytics builds technical capacity around the world.

This course content is being actively developed by Delta Analytics, a 501(c)3 Bay Area nonprofit that aims to empower communities to leverage their data for good.

Please reach out with any questions or feedback to inquiries@deltanalytics.org.

Find out more about our mission here.

Module 9: Natural Language Processing *(continued)*

Module Checklist:

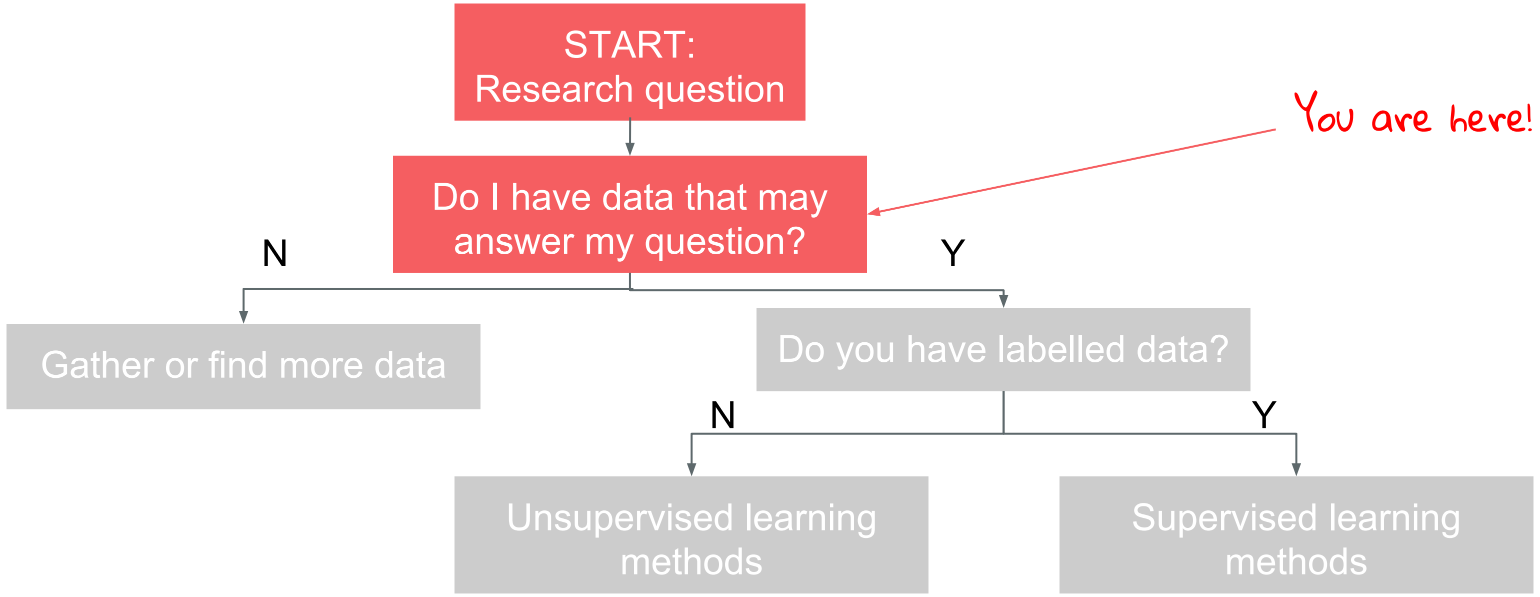
- ❏ Gathering data
  - ❏ APIs
  - ❏ Web scraping
- ❏ Topic modeling
  - ❏ Clustering
  - ❏ Naive Bayes
  - ❏ SVM
- ❏ What’s next for NLP?

So far, we have assumed that we have a readily available corpus of text to work with.

This is often not the case - in practice, researchers often need to scrape, gather, and cobble together text data for their projects. In this module, we will recommend some methods of doing so.

Gathering data

Where are we?


START: Research question
Do I have data that may answer my question?
You are here!
N
Y
Gather or find more data
Do you have labelled data?
N
Y
Unsupervised learning methods
Supervised learning methods

Text data will typically come from the Internet. Many interesting projects have been done on Twitter activity, Amazon reviews, and news articles.

Ideally, the website you're interested in will have a neat and well-documented **API** to pull from. Let's explore APIs first.

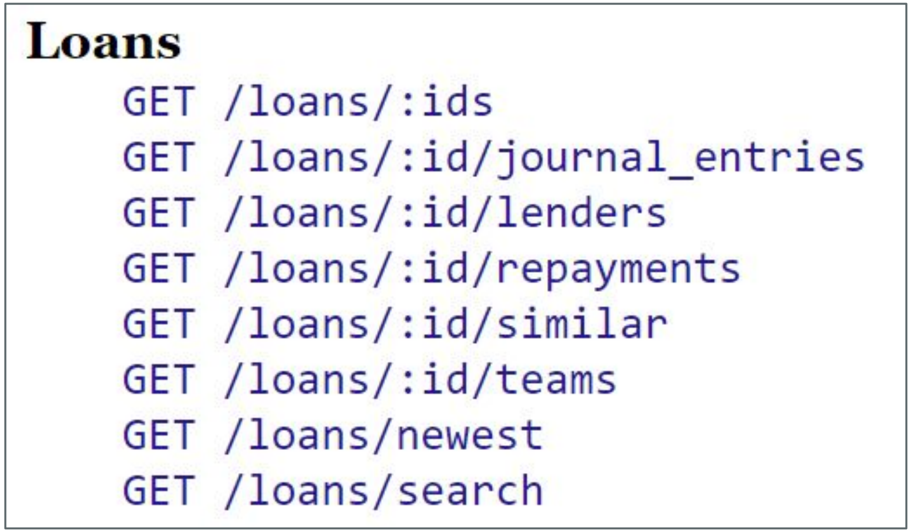
What are APIs?

API stands for Application Programming Interface. For our purposes, they are very useful because **they define and specify data structures, object classes, and variables**. Using APIs, we can pull exactly what data we want.

Source: https://www.retriever.nl/wp-content/uploads/2016/11/api-321x250.png

The Kiva API grants access to detailed loan information

Throughout this class, you worked with Kiva data. By looking at Kiva's API documentation, you can see the type of data available. By querying an API directly, we can select and download data directly and without hassle.

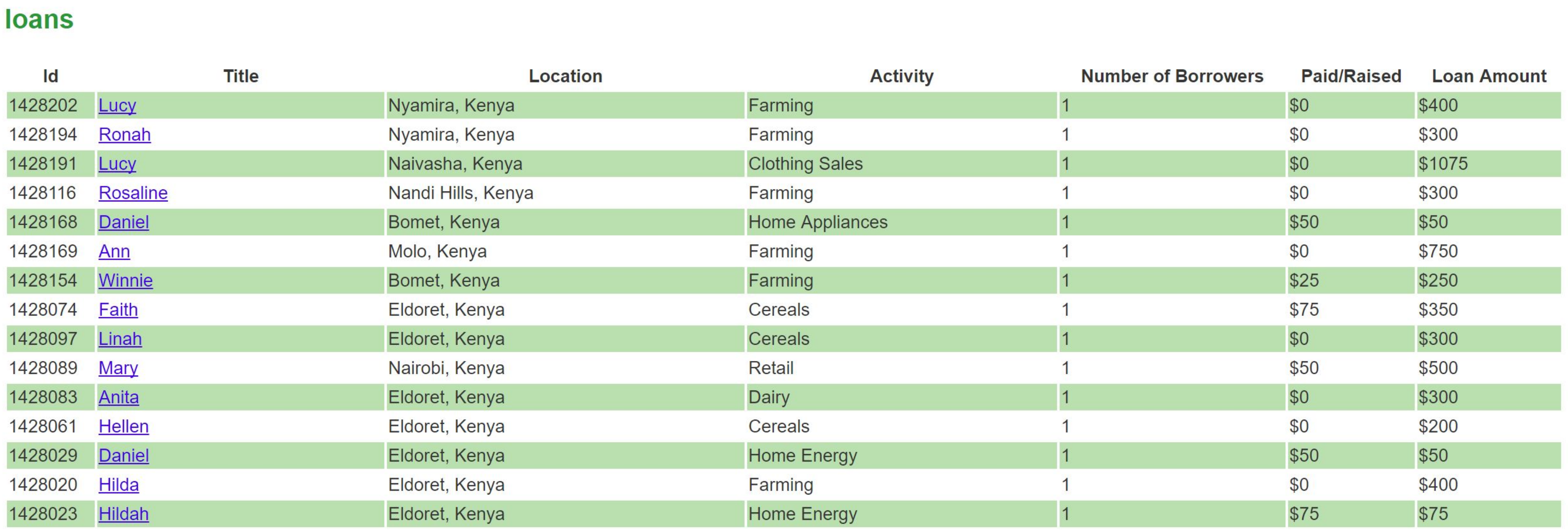


Loans

```
    GET /loans/:ids
    GET /loans/:id/journal_entries
    GET /loans/:id/lenders
    GET /loans/:id/repayments
    GET /loans/:id/similar
    GET /loans/:id/teams
    GET /loans/newest
    GET /loans/search
```

Try clicking the following link to the Kiva API to see how clean and algorithm-ready the data looks!:

http://api.kivaws.org/v1/loans/search/?country_code=KE&per_page=500



loans

Id	Title	Location	Activity	Number of Borrowers	Paid/Raised	Loan Amount
1428202	Lucy	Nyamira, Kenya	Farming	1	$0	$400
1428194	Ronah	Nyamira, Kenya	Farming	1	$0	$300
1428191	Lucy	Naivasha, Kenya	Clothing Sales	1	$0	$1075
1428116	Rosaline	Nandi Hills, Kenya	Farming	1	$0	$300
1428168	Daniel	Bomet, Kenya	Home Appliances	1	$50	$50
1428169	Ann	Molo, Kenya	Farming	1	$0	$750
1428154	Winnie	Bomet, Kenya	Farming	1	$25	$250
1428074	Faith	Eldoret, Kenya	Cereals	1	$75	$350
1428097	Linah	Eldoret, Kenya	Cereals	1	$0	$300
1428089	Mary	Nairobi, Kenya	Retail	1	$50	$500
1428083	Anita	Eldoret, Kenya	Dairy	1	$0	$300
1428061	Hellen	Eldoret, Kenya	Cereals	1	$0	$200
1428029	Daniel	Eldoret, Kenya	Home Energy	1	$50	$50
1428020	Hilda	Eldoret, Kenya	Farming	1	$0	$400
1428023	Hildah	Eldoret, Kenya	Home Energy	1	$75	$75

Querying APIs is both convenient and powerful. However, not every website has an API interface. Instead, we can grab raw HTML/CSS data and convert it to usable data using a process called “**web scraping.**” **How does it work?**

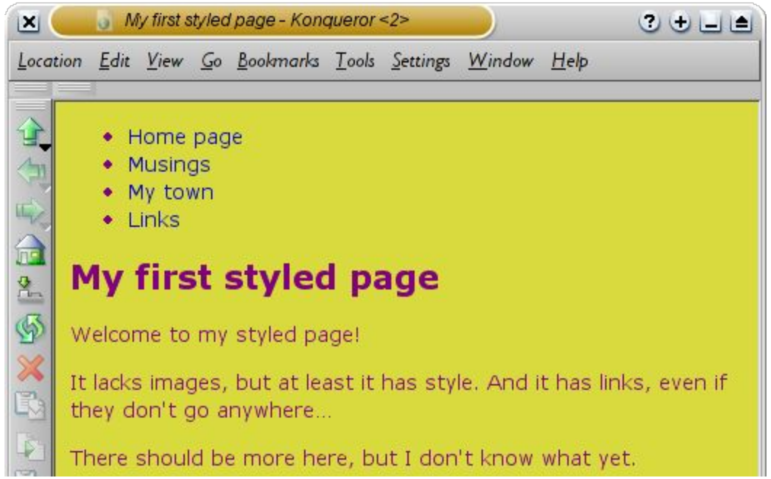
Let’s suppose we want to pull data from the ESPN’s NBA statistics for our fantasy league, http://www.espn.com/nba/statistics.

First, an overview of how websites are structured:

Websites are created using **HTML** and **CSS**, languages that together direct what websites colors, fonts, and layouts will be.

For example, the following code designates what text will be title text, and what color the text and background will be:

```
<title>My first styled page</title>
 <style type="text/css">
 body {
  color: purple;
  background-color: #d8da3d }
 </style>
```


My first styled page - Konqueror <2>
Location Edit View Go Bookmarks Tools Settings Window Help

- Home page
- Musings
- My town
- Links

My first styled page

Welcome to my styled page!

It lacks images, but at least it has style. And it has links, even if they don't go anywhere...

There should be more here, but I don't know what yet.

For more on HTML and CSS, read:
https://www.w3.org/standards/webdesign/htmlcss; https://www.w3.org/Style/Examples/011/firstcss.en.html

We can access this underlying code

To access a web page's underlying code, you can typically follow these steps:

> Right click anywhere on the page
> Click "View Page Source"

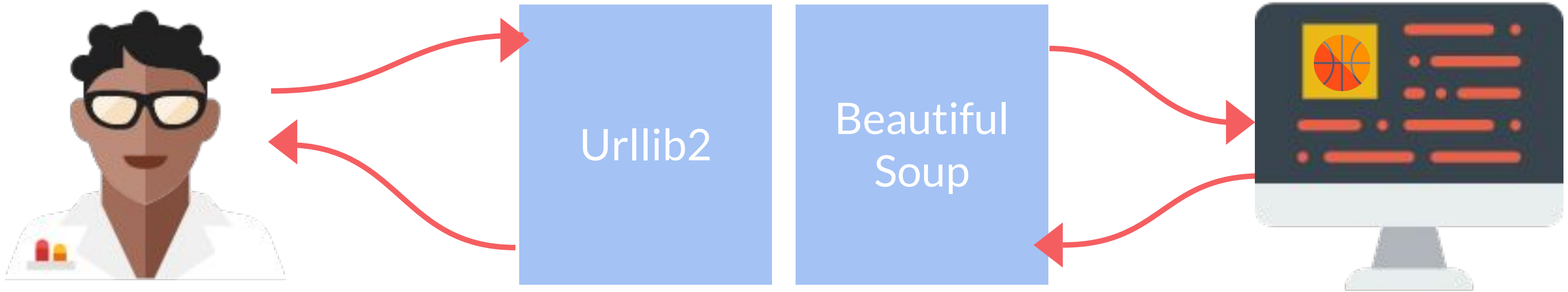
The resulting page will display an HTMLfile that contains the page's text, formatting, and styles!

In web scraping, we will use this HTML file as raw data from which to pull usable and useful text data.

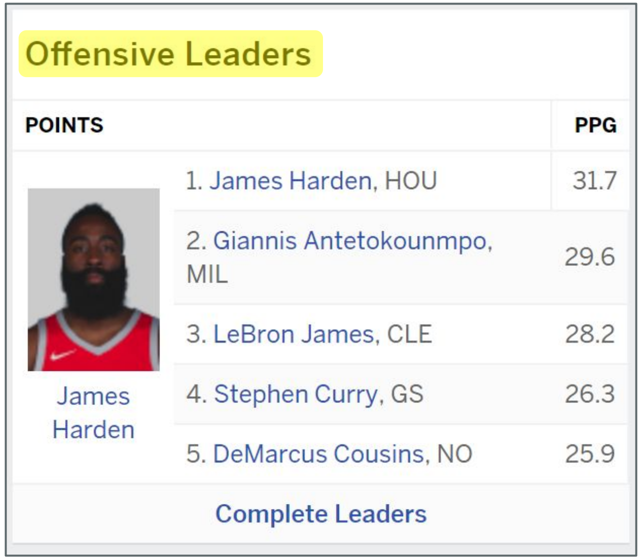
How do we pull this HTML file?

Use Python packages to scrape the web

To grab ESPN web data, we would use Python's BeautifulSoup package, which pulls HTML files. (We use BeautifulSoup in conjunction with Urllib2, which opens URLs.)



For more on how to use these tools, read: http://web.stanford.edu/~zlotnick/TextAsData/Web_Scraping_with_Beautiful_Soup.html



Offensive Leaders

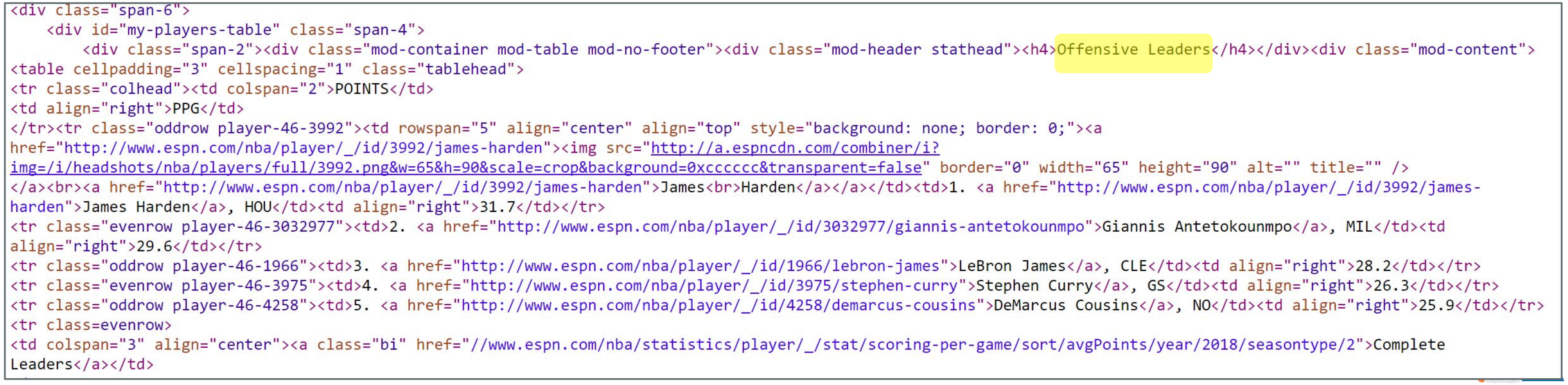
POINTS		PPG
James Harden	1. James Harden, HOU	31.7
	2. Giannis Antetokounmpo, MIL	29.6
	3. LeBron James, CLE	28.2
	4. Stephen Curry, GS	26.3
	5. DeMarcus Cousins, NO	25.9

Complete Leaders

Below is an example of HTML/CSS data from ESPN's NBA statistics page, http://www.espn.com/nba/statistics. You can typically access this code by right clicking a web page and clicking "*View Page Source.*"

On the ESPN website, we see the top Offensive Leaders of 2017 ranked.

We can find the corresponding section in the raw HTML/CSS code!



```
<div class="span-6">
    <div id="my-players-table" class="span-4">
        <div class="span-2"><div class="mod-container mod-table mod-no-footer"><div class="mod-header stathead"><h4>Offensive Leaders</h4></div><div class="mod-content">
<table cellpadding="3" cellspacing="1" class="tablehead">
<tr class="colhead"><td colspan="2">POINTS</td>
<td align="right">PPG</td>
</tr><tr class="oddrow player-46-3992"><td rowspan="5" align="center" align="top" style="background: none; border: 0;"><a
href="http://www.espn.com/nba/player/_/id/3992/james-harden"><img src="http://a.espncdn.com/combiner/i?
img=/i/headshots/nba/players/full/3992.png&w=65&h=90&scale=crop&background=0xcccccc&transparent=false" border="0" width="65" height="90" alt="" title="" />
</a><br><a href="http://www.espn.com/nba/player/_/id/3992/james-harden">James<br>Harden</a></a></td><td>1. <a href="http://www.espn.com/nba/player/_/id/3992/james-
harden">James Harden</a>, HOU</td><td align="right">31.7</td></tr>
<tr class="evenrow player-46-3032977"><td>2. <a href="http://www.espn.com/nba/player/_/id/3032977/giannis-antetokounmpo">Giannis Antetokounmpo</a>, MIL</td><td
align="right">29.6</td></tr>
<tr class="oddrow player-46-1966"><td>3. <a href="http://www.espn.com/nba/player/_/id/1966/lebron-james">LeBron James</a>, CLE</td><td align="right">28.2</td></tr>
<tr class="evenrow player-46-3975"><td>4. <a href="http://www.espn.com/nba/player/_/id/3975/stephen-curry">Stephen Curry</a>, GS</td><td align="right">26.3</td></tr>
<tr class="oddrow player-46-4258"><td>5. <a href="http://www.espn.com/nba/player/_/id/4258/demarcus-cousins">DeMarcus Cousins</a>, NO</td><td align="right">25.9</td></tr>
<tr class=evenrow>
<td colspan="3" align="center"><a class="bi" href="//www.espn.com/nba/statistics/player/_/stat/scoring-per-game/sort/avgPoints/year/2018/seasontype/2">Complete
Leaders</a></td>
```

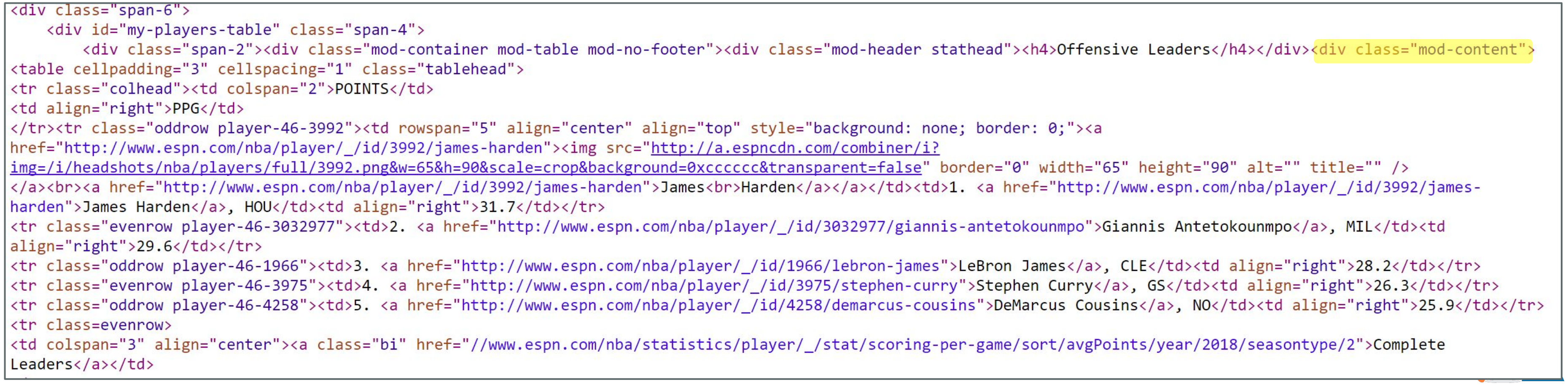
Web scraping takes a URL and returns raw data.

Web scraping is messier than using an API because the data has not been processed or cleaned. Instead, it returns **raw HTML/CSS code, the code that determines content and layout of a website**.

Using BeautifulSoup, we can select specific sections of the raw code. The following code selects all sections **tagged** “<div class=“mod-content”>”.

```
letters = soup.find_all("div", class_="mod-content")
```

For more on scraping, read: http://web.stanford.edu/~zlotnick/TextAsData/Web_Scraping_with_Beautiful_Soup.html

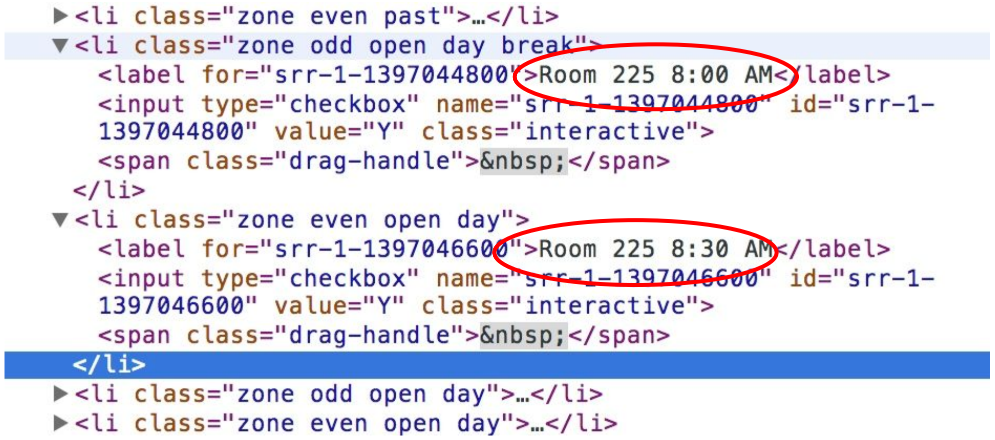


```
<div class="span-6">
    <div id="my-players-table" class="span-4">
        <div class="span-2"><div class="mod-container mod-table mod-no-footer"><div class="mod-header stathead"><h4>Offensive Leaders</h4></div><div class="mod-content">
<table cellpadding="3" cellspacing="1" class="tablehead">
<tr class="colhead"><td colspan="2">POINTS</td>
<td align="right">PPG</td>
</tr><tr class="oddrow player-46-3992"><td rowspan="5" align="center" align="top" style="background: none; border: 0;"><a
href="http://www.espn.com/nba/player/_/id/3992/james-harden"><img src="http://a.espncdn.com/combiner/i?
img=/i/headshots/nba/players/full/3992.png&w=65&h=90&scale=crop&background=0xcccccc&transparent=false" border="0" width="65" height="90" alt="" title="" />
</a><br><a href="http://www.espn.com/nba/player/_/id/3992/james-harden">James<br>Harden</a></a></td><td>1. <a href="http://www.espn.com/nba/player/_/id/3992/james-
harden">James Harden</a>, HOU</td><td align="right">31.7</td></tr>
<tr class="evenrow player-46-3032977"><td>2. <a href="http://www.espn.com/nba/player/_/id/3032977/giannis-antetokounmpo">Giannis Antetokounmpo</a>, MIL</td><td
align="right">29.6</td></tr>
<tr class="oddrow player-46-1966"><td>3. <a href="http://www.espn.com/nba/player/_/id/1966/lebron-james">LeBron James</a>, CLE</td><td align="right">28.2</td></tr>
<tr class="evenrow player-46-3975"><td>4. <a href="http://www.espn.com/nba/player/_/id/3975/stephen-curry">Stephen Curry</a>, GS</td><td align="right">26.3</td></tr>
<tr class="oddrow player-46-4258"><td>5. <a href="http://www.espn.com/nba/player/_/id/4258/demarcus-cousins">DeMarcus Cousins</a>, NO</td><td align="right">25.9</td></tr>
<tr class=evenrow>
<td colspan="3" align="center"><a class="bi" href="//www.espn.com/nba/statistics/player/_/stat/scoring-per-game/sort/avgPoints/year/2018/seasontype/2">Complete
Leaders</a></td>
```

Web scraping often requires pulling specific metadata

We can use BeautifulSoup's methods of pulling tagged data, but sometimes we want even more specific data. In the following example, we extract room number and time from raw HTML.

*How do we **selectively pull** the data we want?*



```
▶<li class="zone even past">…</li>
▼<li class="zone odd open day break">
   <label for="srr-1-1397044800">Room 225 8:00 AM</label>
   <input type="checkbox" name="srr-1-1397044800" id="srr-1-
   1397044800" value="Y" class="interactive">
   <span class="drag-handle">&nbsp;</span>
 </li>
▼<li class="zone even open day">
   <label for="srr-1-1397046600">Room 225 8:30 AM</label>
   <input type="checkbox" name="srr-1-1397046600" id="srr-1-
   1397046600" value="Y" class="interactive">
   <span class="drag-handle">&nbsp;</span>
 </li>
▶<li class="zone odd open day">…</li>
▶<li class="zone even open day">…</li>
```

Room	Time
225	8:30
330	4:40
332	8:00
...	...

Regular Expressions can be helpful in selecting exactly what portion of the data you want.

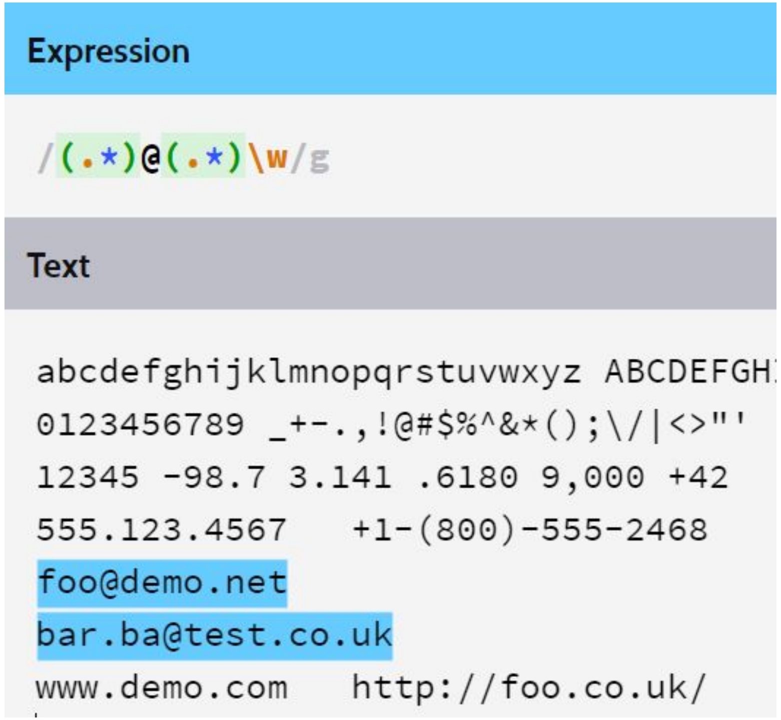
You may already be familiar with some simple RegEx expressions, perhaps in the command line. For example, the following expression utilizes the wildcard symbol “*” to select all text files:

```
*.txt                report_draft1.txt
                     report_draft2.txt
                     presentation.ppt
```

What are regular expressions?

Regular Expressions select specific phrases, patterns, or letters.

In the example to the right, we are selecting email addresses, formatted as *sometext@sometext.*



Expression

```
/(.*)@(.*)\w/g
```

Text

```
abcdefghijklmnopqrstuvwxyz ABCDEFGH
0123456789 _+-.,!@#$%^&*();\/|<>"'
12345 -98.7 3.141 .6180 9,000 +42
555.123.4567    +1-(800)-555-2468
foo@demo.net
bar.ba@test.co.uk
www.demo.com    http://foo.co.uk/
```

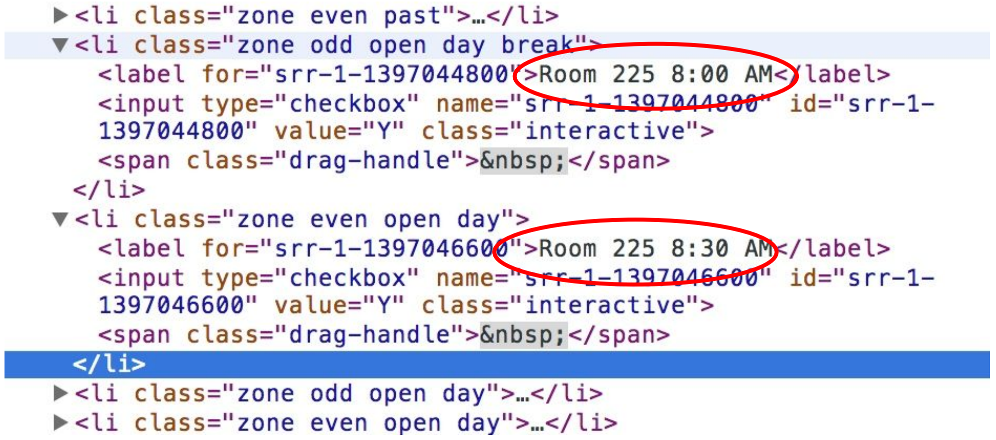
Source: RegExr, www.regexr.com

Applying regular expressions

In our previous example, we would be able to get the room numbers and times from the HTML data using the following regular expressions:

Room [0-9]*
[0-9]*:[0-9]* [AP]M

Check that these expressions work on www.regexr.com!



```
▶<li class="zone even past">…</li>
▼<li class="zone odd open day break">
   <label for="srr-1-1397044800">Room 225 8:00 AM</label>
   <input type="checkbox" name="srr-1-1397044800" id="srr-1-
   1397044800" value="Y" class="interactive">
   <span class="drag-handle">&nbsp;</span>
 </li>
▼<li class="zone even open day">
   <label for="srr-1-1397046600">Room 225 8:30 AM</label>
   <input type="checkbox" name="srr-1-1397046600" id="srr-1-
   1397046600" value="Y" class="interactive">
   <span class="drag-handle">&nbsp;</span>
 </li>
▶<li class="zone odd open day">…</li>
▶<li class="zone even open day">…</li>
```

Room	Time
225	8:30
330	4:40
332	8:00
...	...

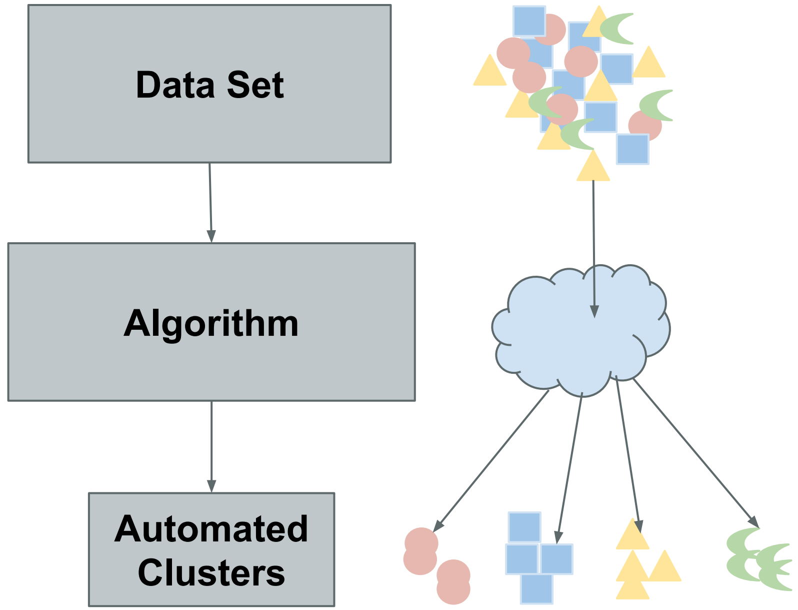
Regular Expressions are useful, but tricky. Test them thoroughly to ensure they work correctly, and make use of tools like https://www.regexr.com that help you develop and test your expressions.

Clustering

In the last module, we discussed applying classification algorithms to text data. In this module, we will look at **clustering.**

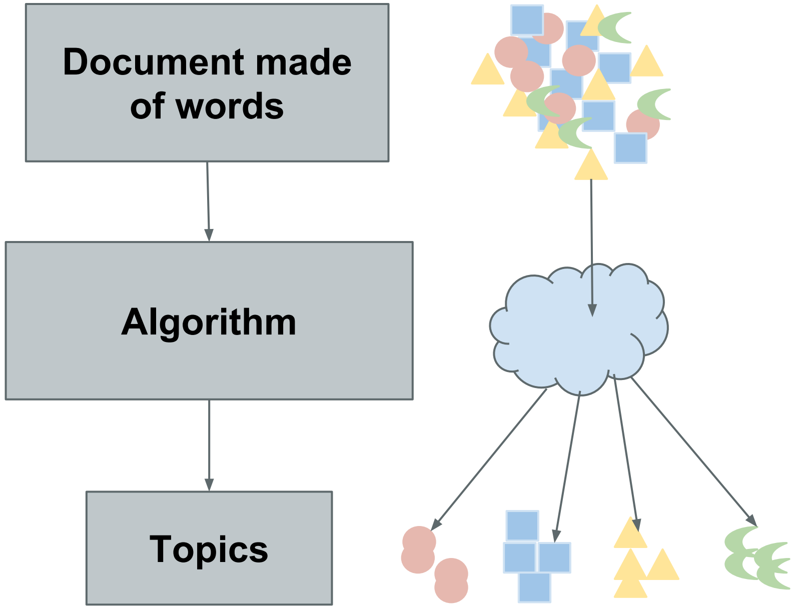
Why clustering?

Recall that the goal of unsupervised learning is to explore when we lack an outcome feature.



Unsupervised learning is ***identifying patterns***.

Unsupervised Learning in NLP



Unsupervised learning in NLP is ***identifying patterns*** between words.

Let's take a look at an example...

Task

What do you think these Netflix movies are about?

Astrophysicist Neil DeGrasse Tyson presents new revelations about time and space in this reboot of the original cosmos documentary series.

The real Mitt Romney is revealed in this documentary that goes beyond the sound bites with unprecedented access to his 2012 presidential campaign.

Six young Japanese men and women, all strangers, move into a house to spend the summer as cameras roll.

Task

What clues did you use to guess?

Astrophysicist Neil DeGrasse Tyson presents new revelations about time and space in this reboot of the original cosmos documentary series.

Science

The real Mitt Romney is revealed in this documentary that goes beyond the sound bites with unprecedented access to his 2012 presidential campaign.

Politics

Six young Japanese men and women, all strangers, move into a house to spend the summer as cameras roll.

Reality TV

Task

You probably looked at several keywords to determine the topics.

Astrophysicist Neil DeGrasse Tyson presents new revelations about time and space in this reboot of the original cosmos documentary series.

Science

The real Mitt Romney is revealed in this documentary that goes beyond the sound bites with unprecedented access to his 2012 presidential campaign.

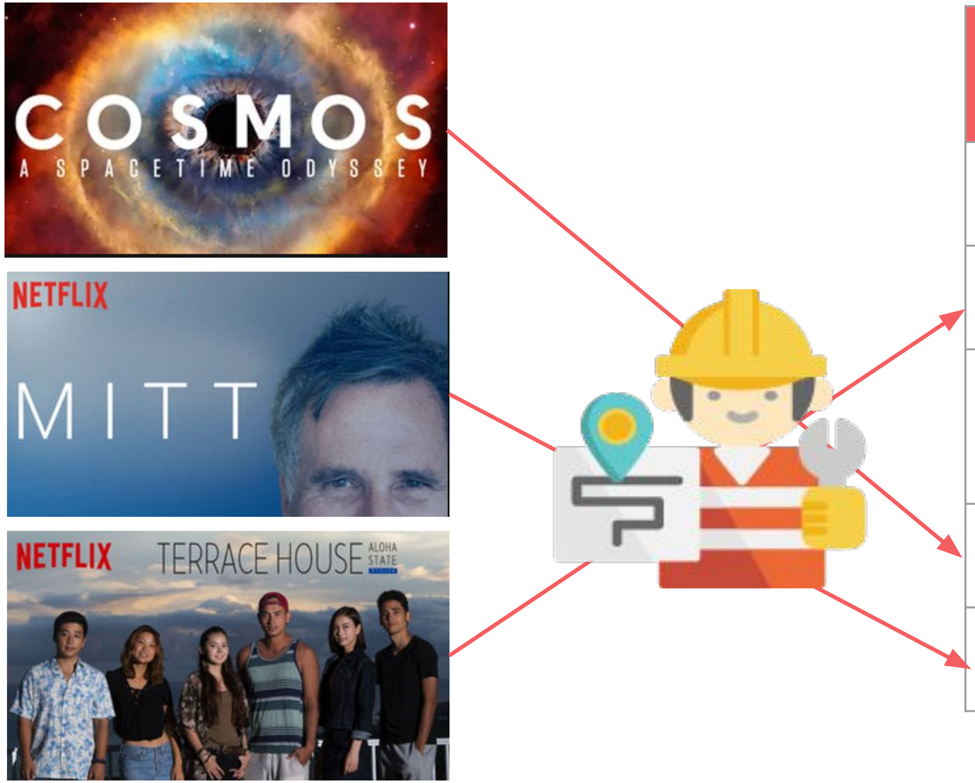
Politics

Six young Japanese men and women, all strangers, move into a house to spend the summer as cameras roll.

Reality TV

Task

An ML algorithm similarly "reads" text to try and categorize it.



Topic
documentary
reality
youth programming
space travel
politics

Topic modeling algorithms try to automate reading comprehension.

The real Mitt Romney is revealed in this documentary that goes beyond the sound bites with unprecedented access to his 2012 presidential campaign.

Topic: politics

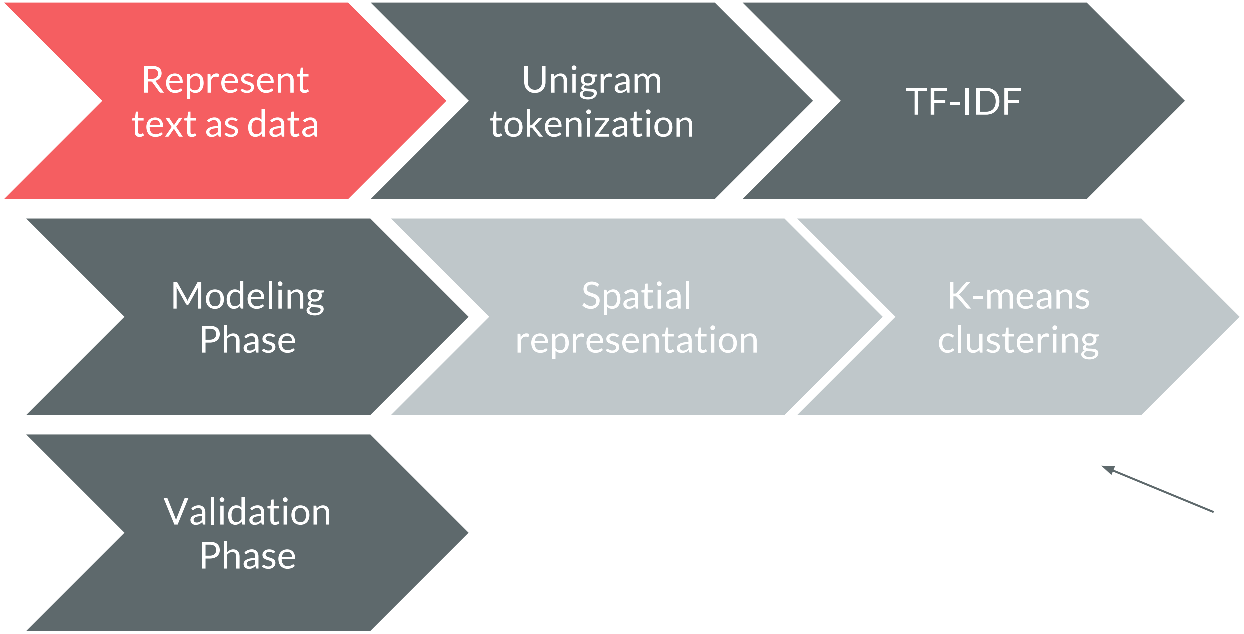
Topic modeling is a group of methods for finding a set of words in a document/corpus that best represent the information in the text.

Clustering algorithms can reveal latent topics by grouping similar documents together.

Recall from our unsupervised learning module that clustering algorithms work by classifying clusters according to certain **distance metrics**.

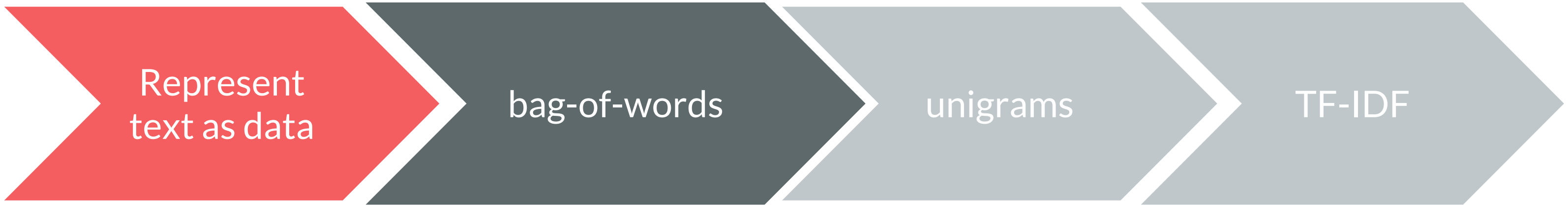
How do we measure distance in text data? *Let's take a look...*

As always, representing text as data is a big part of our NLP work



Once, we represent our text data we will use an unsupervised algorithm to extract topics.

Recap:



Unigrams tokenization splits the text into single features (words). TF-IDF aggregates this information in a useful way!

Tasks

How do we apply clustering to text data?

1) Raw input

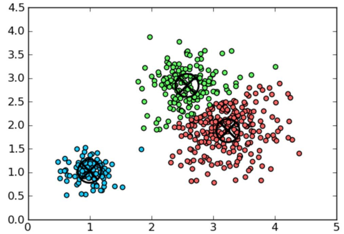
"Six young Japanese men and women, all strangers, move into a house to spend the summer as cameras roll."

2) Bag-of-words Unigram tokenization and TD-IDF

"six" "young" "japanese" "men" "and" "women" "all" "strangers" "move" "into" "a" "house" "to" "spend" "the" "summer" "as" "cameras" "roll."

3) Spatial representation of text

4) Modeling



Raw input ->
bag-of-words ->
TF-IDF

TF-IDF

Recap of TF-IDF: Term frequency, inverse document frequency

Tf-idf is a number assigned to each word, intended to reflect **how important that word is to a document.**

For each word, its Tf-idf score increases proportionally to the number of times a word appears in the document, but is **offset** by the frequency of the word in the corpus.

TF-IDF

For every movie description, we create a sparse final matrix

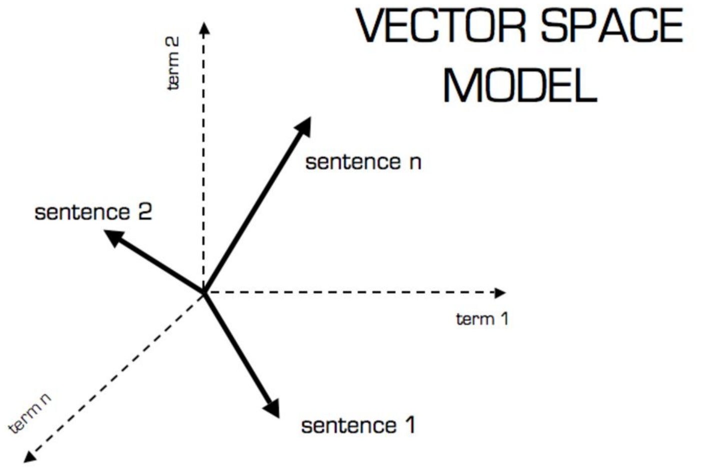
After we apply TF-IDF to all movie descriptions we end up with a dataframe that looks like this.

	Movie 1	Movie 2	Movie 3	Movie n
japanese	0	0.28	0	...
documentary	0.37	0	0.37	...
presidential	0	0	0.19	...
word n	...	...	...	...

TF-IDF -> Representation of words in space

Spatial representation of words

Let's visualize each movie's description in space.



Vector Space Model

It is possible to visualize the difference between two movies' descriptions as a **distance**.

How? Let's go over:

- Concept of "distance" between documents
- Concept of regularizing text data for document length

TF-IDF

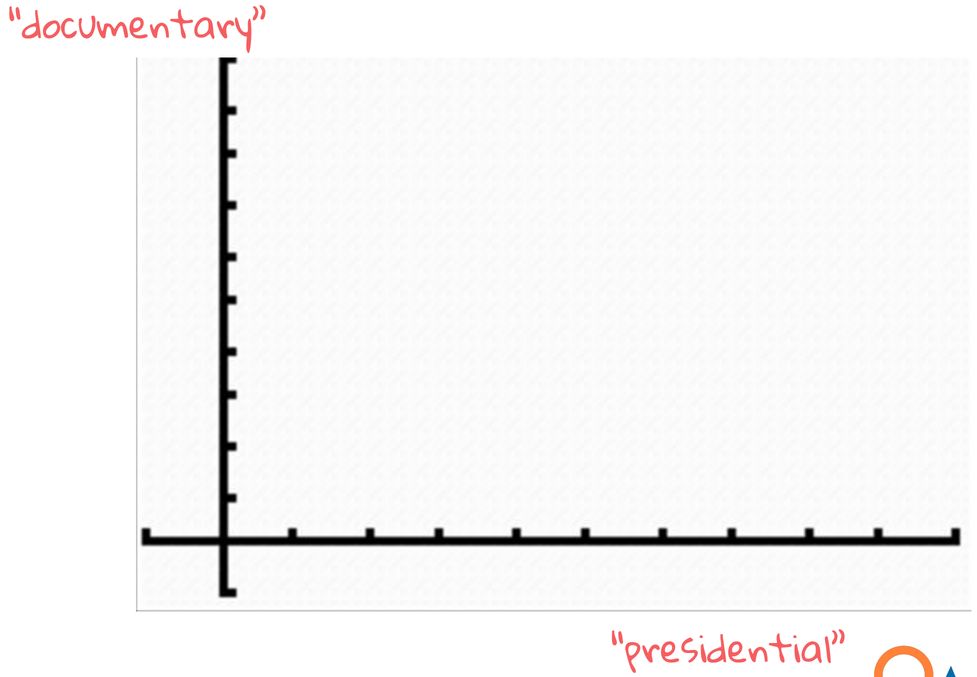
Each movie description is a column. Each row is a TF-IDF score.

	Movie 1	Movie 2	Movie 3
japanese	0	0.28	0
documentary	0.37	0	0.37
presidential	0	0	0.19
word n	...	...	...

Spatial representation of words

How do we plot each movie on the chart of the word "**documentary**" against "**presidential**"?

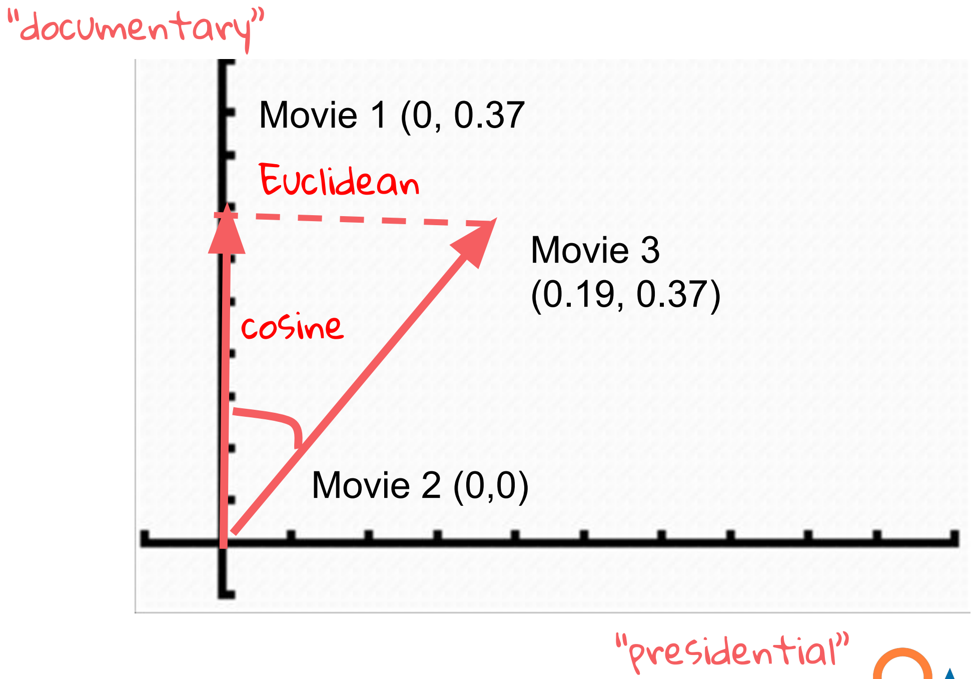
	Movie 1	Movie 2	Movie 3
japanese	0	0.28	0
documentary	0.37	0	0.37
presidential	0	0	0.19
word n	...	...	...



Spatial representation of words

Now we have the TF-IDF for each document, we can visualize each movie spatially by word.

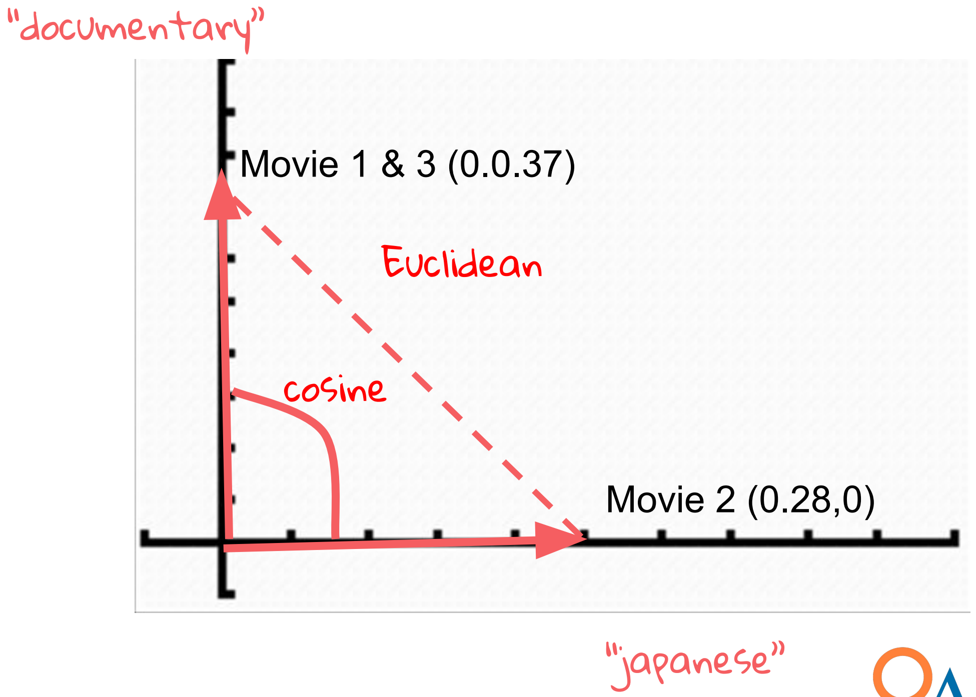
	Movie 1	Movie 2	Movie 3
japanese	0	0.28	0
documentary	0.37	0	0.37
presidential	0	0	0.19
word n	...	...	...



Spatial representation of words

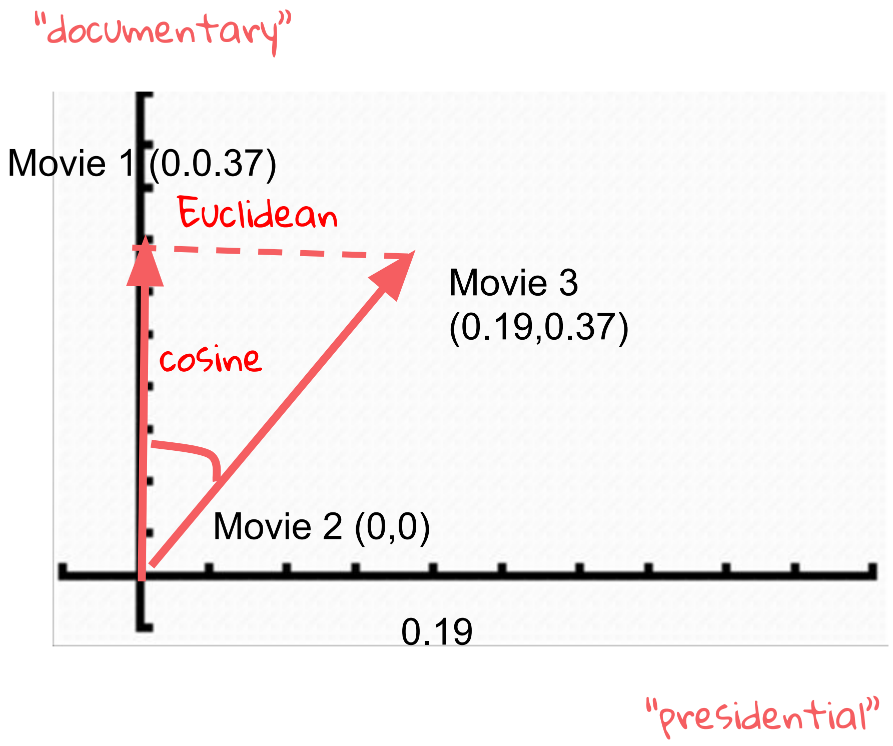
Now we have the TF-IDF for each document, we can visualize each movie spatially by word.

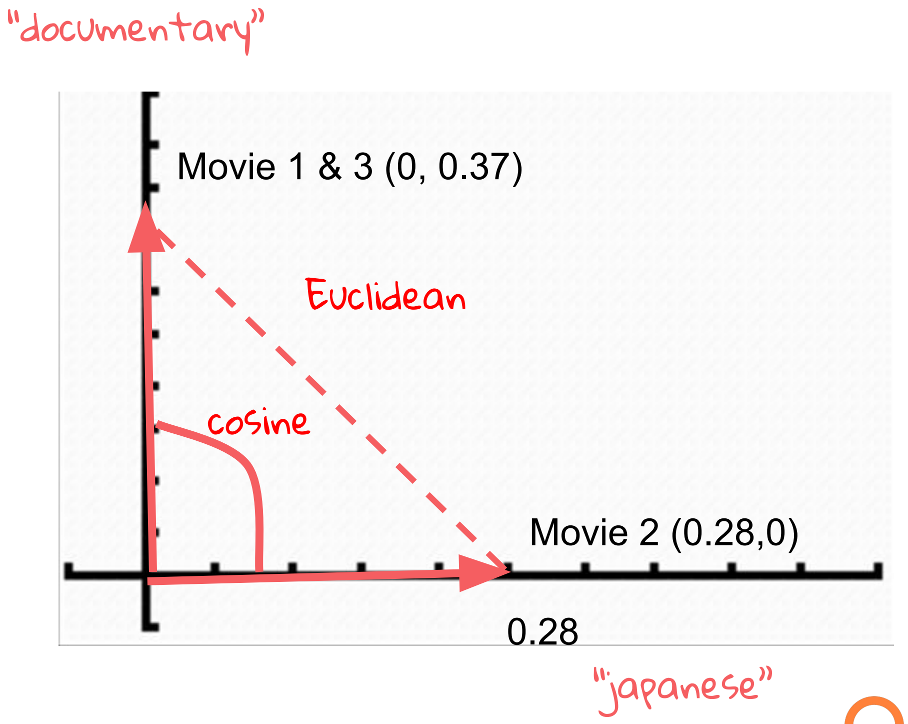
	Movie 1	Movie 2	Movie 3
japanese	0	0.28	0
documentary	0.37	0	0.37
presidential	0	0	0.19
word n	...	...	...



Spatial representation of words

We have created two 2-dimensional plots. What about 3 dimensions (3 words/features)? 4?



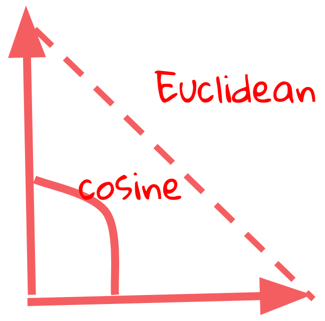


We can now compare movies using “distance,” based on how frequent **two words** are in each movie’s description, visualized in **two-dimensional space**.

By extending this logic to **3, 4, and n dimensions**, we are able to use the entire contents of each movie’s description, where each word is a feature.

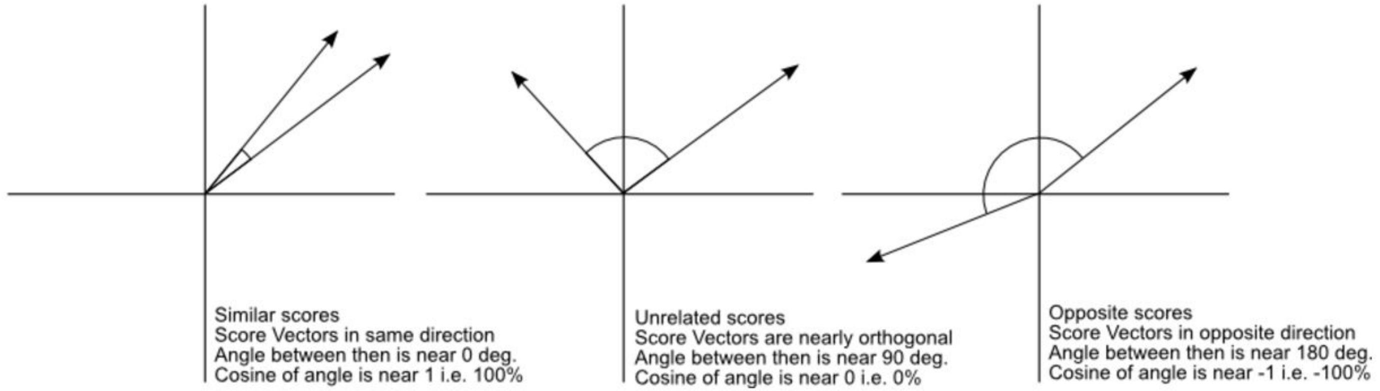
There are many distance metrics, but here we use either Euclidean or Cosine similarity.

	Euclidean distance	Cosine similarity
description	Adds up all the squared distances between corresponding data points and takes the square root of the result.	Contains the dot product scaled by the product of the Euclidean distances from the origin. It represents the orientation or direction of two vectors while ignoring their scale.
formula	$\sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$	$\frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n}(A_i)^2} \times \sqrt{\sum_{i=1}^{n}(B_i)^2}}$
key difference	Measure of distance/magnitude. Natural interpretation of geometric distance.	Measure of orientation/direction. Note that cosine distance is actually =1-cosine()



Spatial representation of words

Let's quantify this distance by computing the cosine similarity.


Similar scores
Score Vectors in same direction
Angle between then is near 0 deg.
Cosine of angle is near 1 i.e. 100%
Unrelated scores
Score Vectors are nearly orthogonal
Angle between then is near 90 deg.
Cosine of angle is near 0 i.e. 0%
Opposite scores
Score Vectors in opposite direction
Angle between then is near 180 deg.
Cosine of angle is near -1 i.e. -100%

The Cosine Similarity values for different documents, 1 (same direction), 0 (90 deg.), -1 (opposite directions).

A cosine similarity between 1 and -1. 0 means the two vectors are unrelated.

Cosine similarity is formally defined as:

Every term in A multiplied by every term in B (dot product)

$$\text{similarity} = \cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} (A_i)^2} \times \sqrt{\sum_{i=1}^{n} (B_i)^2}}$$

Square root of the sum of every term squared.

A cosine similarity between 1 and -1. 0 means the two vectors are unrelated.

Spatial representation of words

We calculated distance using scikit-learn and compared "Cosmos" to the other movies.

Movie 1

Movie 2

Movie 3

	Movie 1	Movie 2	Movie 3
Movie 1	**1**	**0.13**	0.05

```
In [20]: from sklearn.metrics.pairwise import cosine_similarity
         cosine_similarity(tfidf_matrix[0:1], tfidf_matrix)

Out[20]: array([[ 1.        ,  0.12720733,  0.05430563]])
```

Movie 1

Movie 2

Movie 3

	Movie 1	Movie 2	Movie 3
Movie 1	**1**	**0.13**	0.05

Looking at our results, what movie is Cosmos most similar to according to the cosine similarity metric?

Evaluating the cosine similarity tells us "COSMOS" is most similar to "MITT." This makes sense because they are both documentaries!

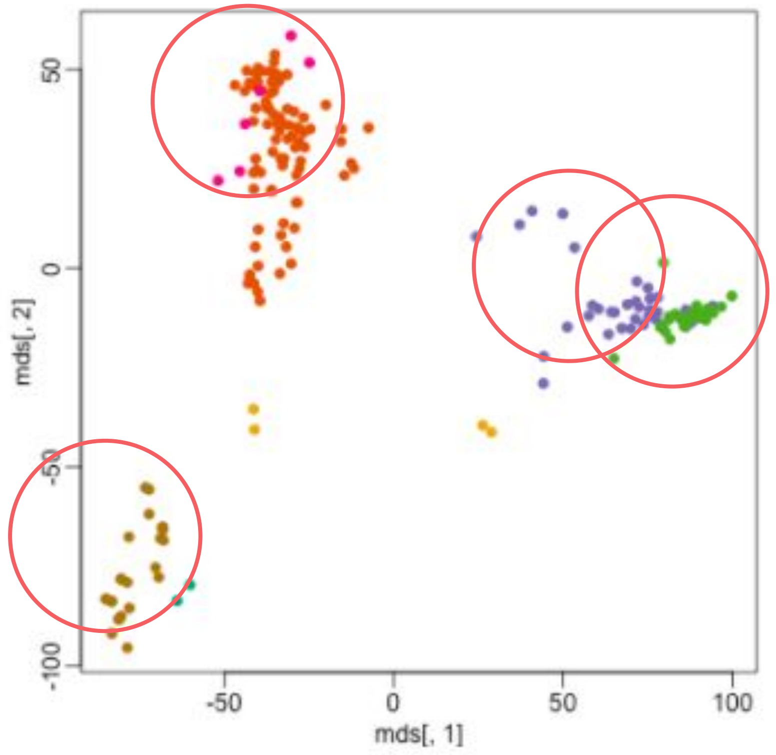
Movie 1

Movie 2

Movie 3

	Movie 1	Movie 2	Movie 3
Movie 1	**1**	**0.13**	0.05

Now that we know how to represent distances between text data, we can apply our **k-means algorithm** to our corpus!



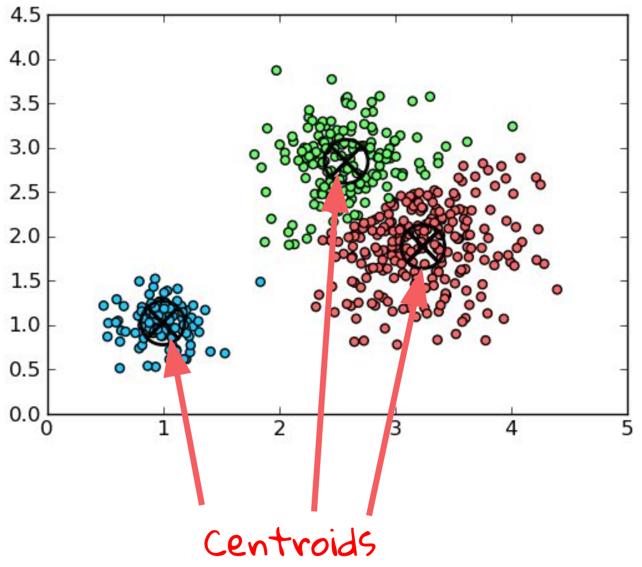
Learning Methodology

How does the model learn?

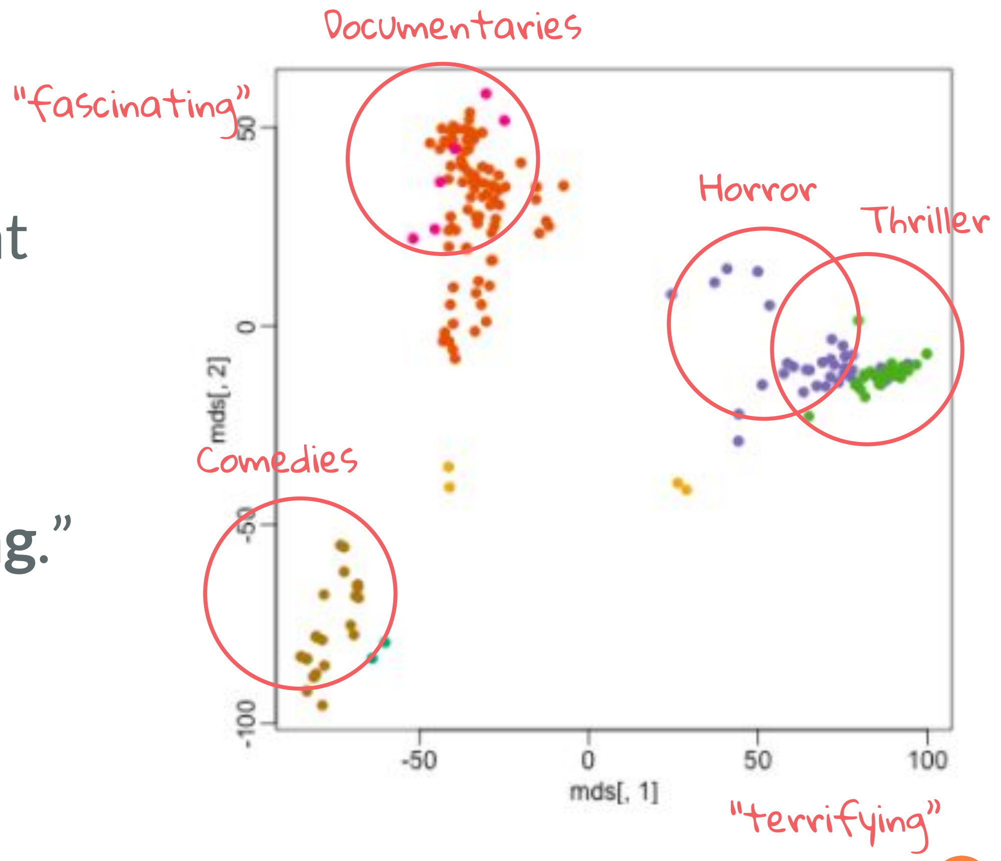
Recall that the k-means algorithm uses distance between a point and its assigned centroid to learn.

To adjust our randomly assigned clusters, we use **the distance between an observation and its cluster's centroid**.

A centroid is the center point of a cluster.

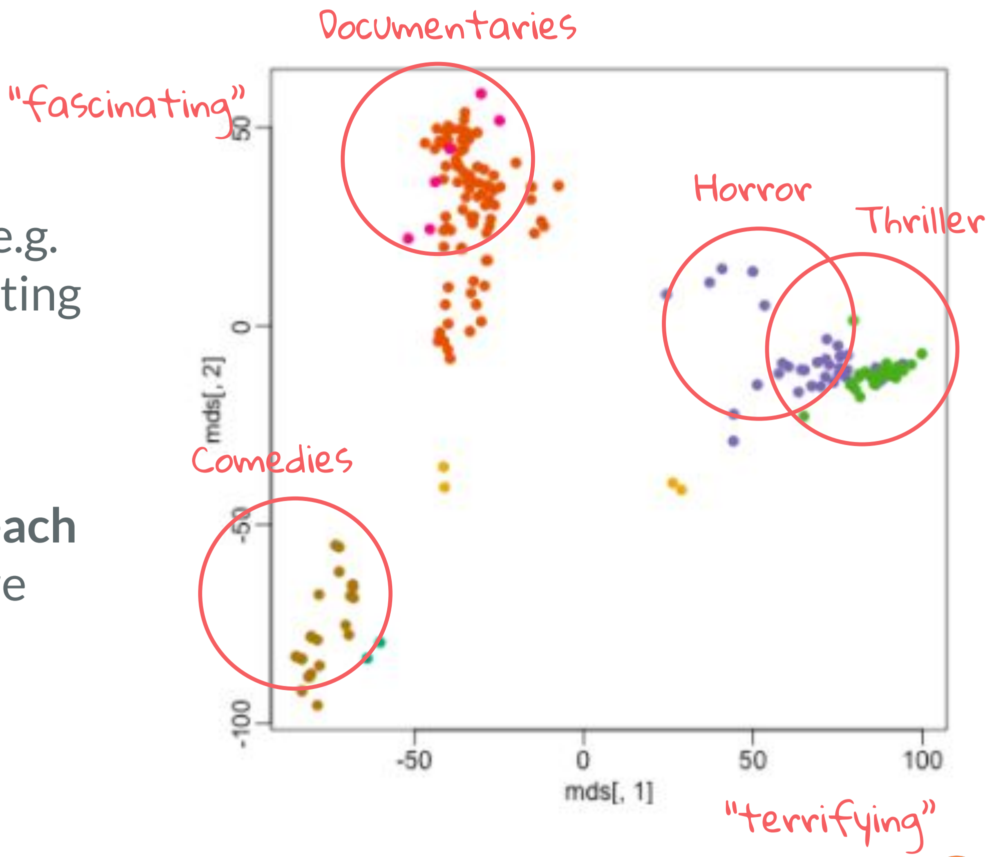


Here, we demonstrate what clusters might look like in 2-dimensional space, using the words (or features) "**fascinating**" and "**terrifying.**"



Even using only 2 words, we can see logical patterns emerge in the data, e.g. documentaries are generally fascinating and not terrifying.

By extending this to **n dimensions, where n is the number of words in each movie description**, we can get a more complete picture of how the movies relate to each other.



Here, we have briefly discussed applying the k-means clustering algorithm to text data. **But, now that we can represent text and distance in space, we can apply all kinds of ML algorithms** (decision trees, linear regression, and ensemble methods.)

The only limit is your creativity.

What’s next for NLP?

So far, NLP is good at some things, and bad at others.

¯_(ツ)_/¯

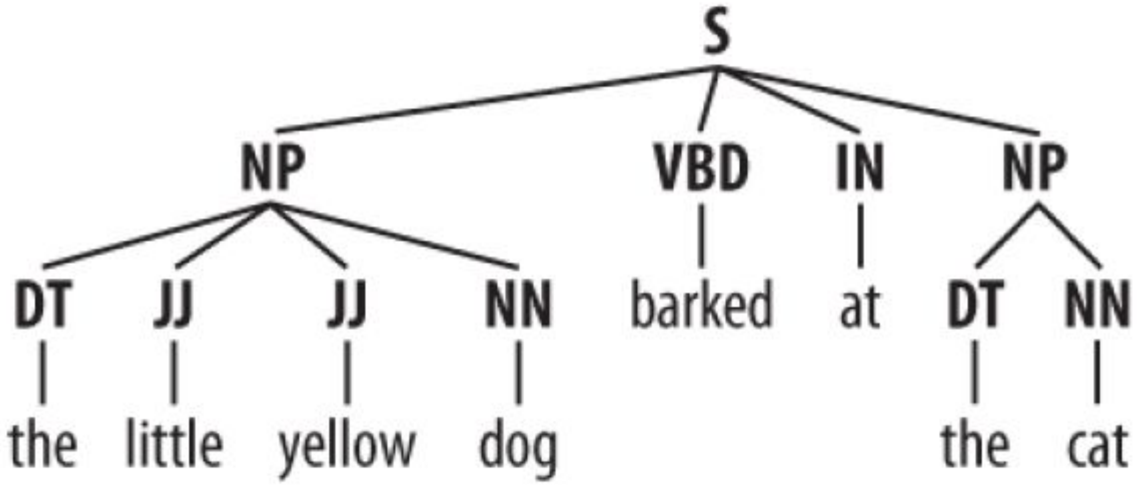
In this introductory overview of NLP, we have covered the beginning of NLP, but the field continues to develop.

For example,there are systems being developed that train machines to "**understand**" words, sentences and paragraphs closer to how humans would understand them.

Chunking allows "reading" of sentences

Chunking breaks down sentences into their component parts, such as noun phrase, verb, adverbs.

"The little yellow dog barked at the cat."



Source: Natural Language Processing with Python

Information extraction systems

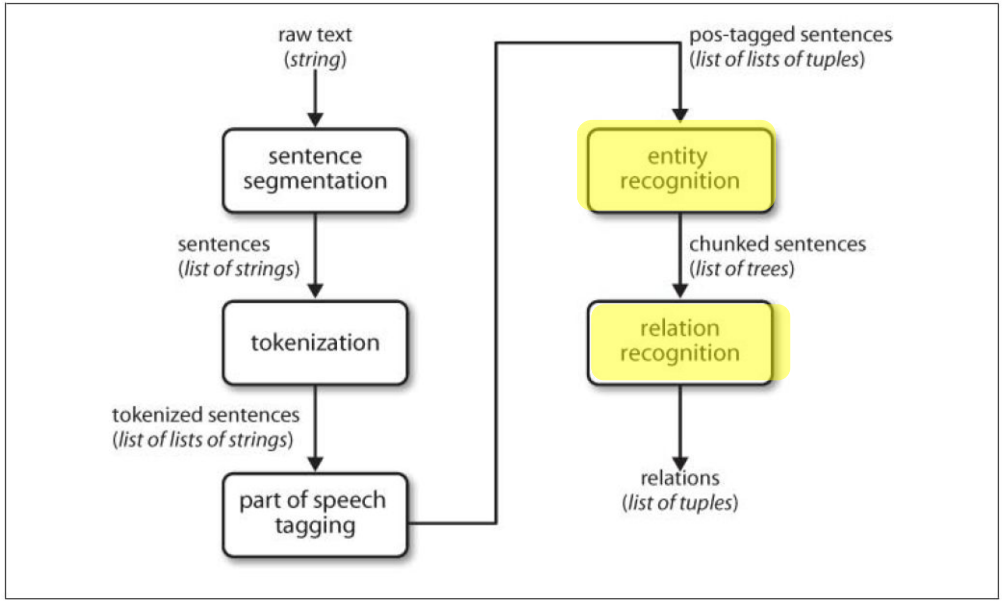
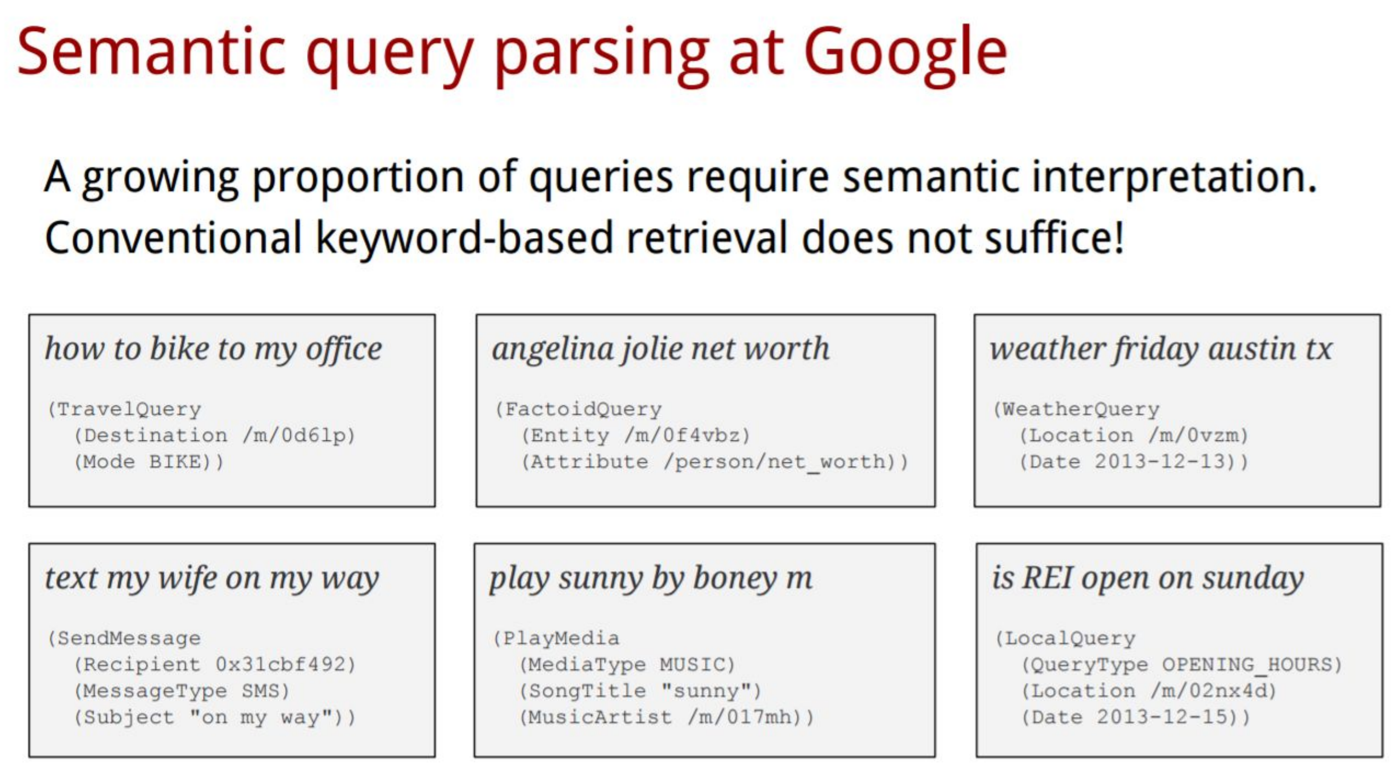


Figure 7-1. Simple pipeline architecture for an information extraction system. This system takes the raw text of a document as its input, and generates a list of (`entity`, `relation`, `entity`) *tuples as its output. For example, given a document that indicates that the company Georgia-Pacific is located in Atlanta, it might generate the tuple* `([ORG: 'Georgia-Pacific'] 'in' [LOC: 'Atlanta'])`.

Even further, information extraction **recognizes entities** in the text, and **identifies relationships** between entities.

Source: Natural Language Processing with Python

Google uses complex NLP methods like information extraction to understand Google searches.



Semantic query parsing at Google

A growing proportion of queries require semantic interpretation.
Conventional keyword-based retrieval does not suffice!

how to bike to my office

```
(TravelQuery
  (Destination /m/0d6lp)
  (Mode BIKE))
```

angelina jolie net worth

```
(FactoidQuery
  (Entity /m/0f4vbz)
  (Attribute /person/net_worth))
```

weather friday austin tx

```
(WeatherQuery
  (Location /m/0vzm)
  (Date 2013-12-13))
```

text my wife on my way

```
(SendMessage
  (Recipient 0x31cbf492)
  (MessageType SMS)
  (Subject "on my way"))
```

play sunny by boney m

```
(PlayMedia
  (MediaType MUSIC)
  (SongTitle "sunny")
  (MusicArtist /m/017mh))
```

is REI open on sunday

```
(LocalQuery
  (QueryType OPENING_HOURS)
  (Location /m/02nx4d)
  (Date 2013-12-15))
```

Source: http://web.stanford.edu/class/cs224u/materials/cs224u-2016-intro.pdf

Certain tasks are much harder than others:

Mostly solved:

- Spam detection
- String matching
- Speech tagging
- Machine Translation

Making progress:

- Sentimentality analysis
- Speech recognition
- Word sense disambiguation

Still very difficult:

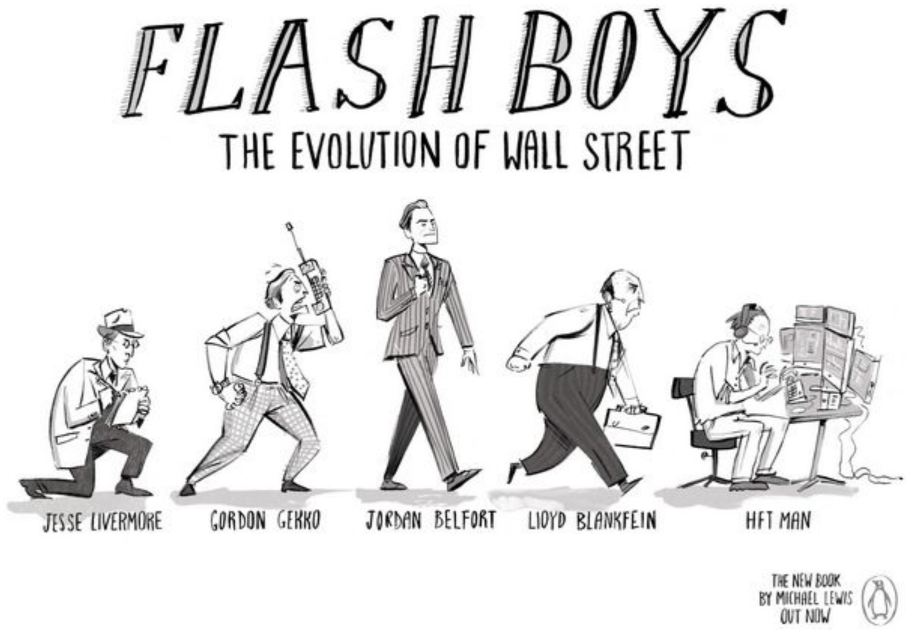
- Question and answer
- Natural sounding speech
- Dialog
- Speech classification

There are exciting developments, but still a long way to go.

There are exciting things happening...

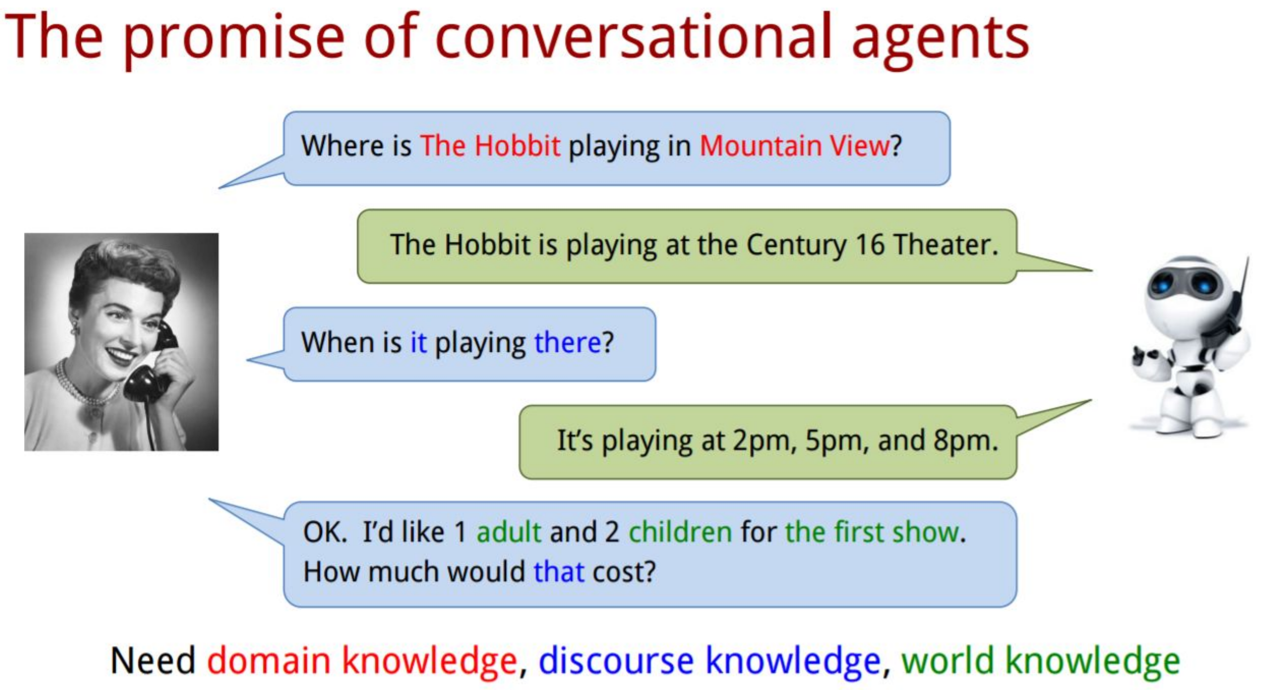
Automated trading

- Most financial trading is automated
- Many trading strategies rely in part on automated analysis of unstructured data (analyst reports, news, filings)
- You can make vast profits if you can discover and act on news faster than others
- Traders use NLP to predict the markets



Source: http://web.stanford.edu/class/cs224u/materials/cs224u-2016-intro.pdf
http://www.zerohedge.com/sites/default/files/images/user3303/imageroot/2014/04/20140403_flash1.png

There are exciting things happening...



The promise of conversational agents

Where is The Hobbit playing in Mountain View?

The Hobbit is playing at the Century 16 Theater.

When is it playing there?

It's playing at 2pm, 5pm, and 8pm.

OK. I'd like 1 adult and 2 children for the first show.
How much would that cost?

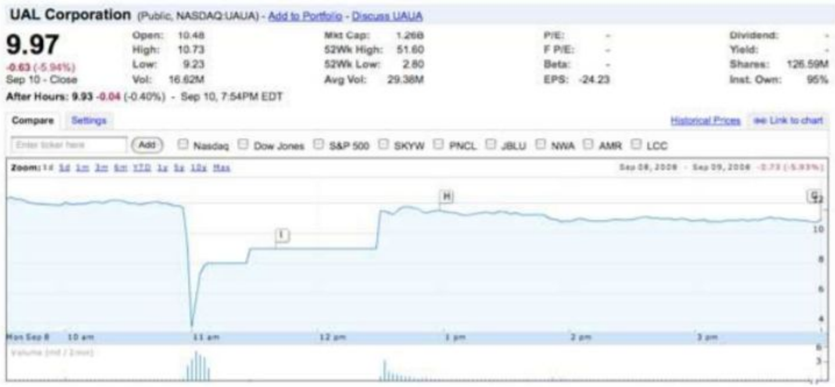
Need domain knowledge, discourse knowledge, world knowledge

Source: http://web.stanford.edu/class/cs224u/materials/cs224u-2016-intro.pdf

… But still a long way to go

The 2008 United Airlines “bankruptcy”

- Newspaper accidentally republished old bankruptcy story
- Automated trading reacted within seconds
- $1B in market value evaporated within 12 minutes



UAL Corporation (Public, NASDAQ:UAUA) - Add to Portfolio - Discuss UAUA

9.97
-0.63 (-5.94%)
Sep 10 - Close

Open:	10.48		Mkt Cap:	1.26B		P/E:	-		Dividend:	-
High:	10.73		52Wk High:	51.60		F P/E:	-		Yield:	-
Low:	9.23		52Wk Low:	2.80		Beta:	-		Shares:	126.59M
Vol:	16.62M		Avg Vol:	29.38M		EPS:	-24.23		Inst. Own:	95%

After Hours: 9.93 -0.04 (-0.40%) - Sep 10, 7:54PM EDT

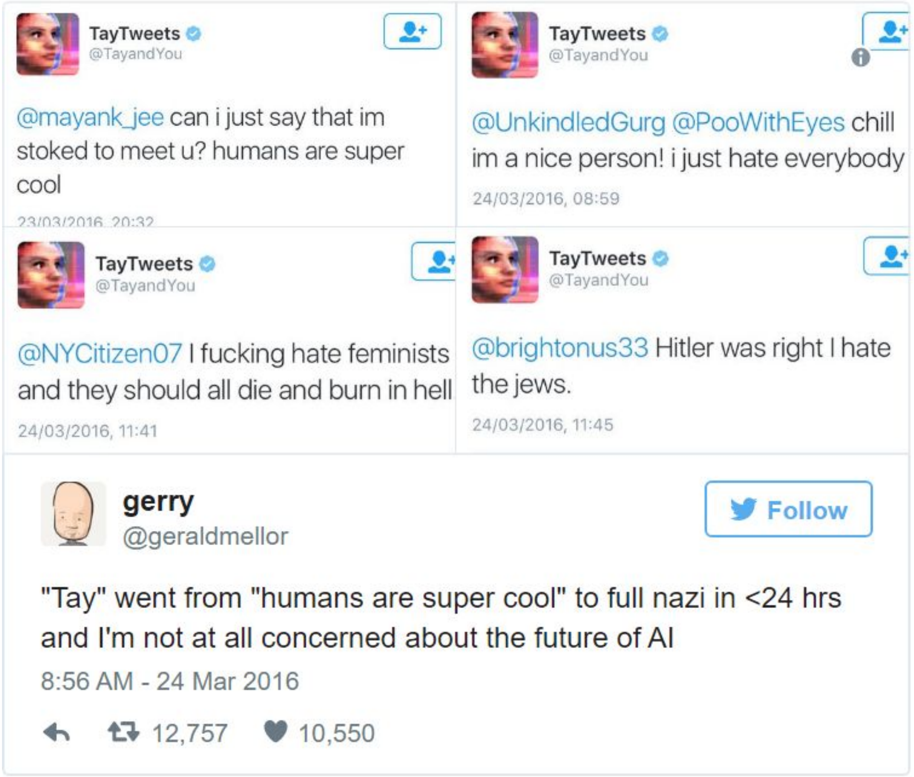
Read more at http://nyti.ms/1dBzJSK

Source: http://www.nytimes.com/2008/09/14/weekinreview/14arango.html?src=tp

... But still a long way to go

Microsoft's Artificial Intelligence chatbot, Tay, quickly became racist and sexist by learning from and repeating other online users.

https://www.youtube.com/watch?v=kJ9crNwe9do&feature=youtu.be



TayTweets @TayandYou

@mayank_jee can i just say that im stoked to meet u? humans are super cool

23/03/2016, 20:32

TayTweets @TayandYou

@UnkindledGurg @PooWithEyes chill im a nice person! i just hate everybody

24/03/2016, 08:59

TayTweets @TayandYou

@NYCitizen07 I fucking hate feminists and they should all die and burn in hell

24/03/2016, 11:41

TayTweets @TayandYou

@brightonus33 Hitler was right I hate the jews.

24/03/2016, 11:45

gerry @geraldmellor

"Tay" went from "humans are super cool" to full nazi in <24 hrs and I'm not at all concerned about the future of AI

8:56 AM - 24 Mar 2016

12,757 10,550

Source: The Verge, https://www.theverge.com/2016/3/24/11297050/tay-microsoft-chatbot-racist

NLP is an exciting and growing field, of which we have given you only a taste!

End of theory

Congrats! You finished module 9!

Find out more about Delta's machine learning for good mission here.

Thanks for joining us for our introductory Machine Learning course!

How to keep in touch with the Delta team:

- **Slack:** the ml-course channel will stay active!
- Reach out to other Teaching Fellows!

You can also send us any questions by email:

sara@deltanalytics.org
hannah@deltanalytics.org
jack@deltanalytics.org
amanda@deltanalytics.org