Funded by the European Union

Topic Modelling

Vitor Rocio

(vitor.rocio@uab.pt)

What is topic modelling?

- To discover hidden topics/themes in text
- To classify documents according to the discovered themes
- To help organize/summarize/search the documents
- Topic modelling methods:
  - Clustering techniques: k-means, PCA, NNMF
  - LDA (Latent Dirichlet Allocation)

Latent Dirichlet Allocation (LDA)

Main idea: each document can be describe by a statistical distribution of topics and each topic can be described by a distribution of words

```
Doc1: word1, word3, word5, word45, word11, word 62, word88 ...
Doc2: word9, word77, word31, word58, word83, word 92, word49 ...
Doc3: word44, word18, word52, word36, word64, word 11, word20 ...
Doc4: word85, word62, word19, word4, word30, word 94, word67 ...
Doc5: word19, word53, word74, word79, word45, word 39, word54 ...
```

	Word1	word2	word3	word4	.....
Topic1	0.01	0.23	0.19	0.03	
Topic2	0.21	0.07	0.48	0.02	
Topic3	0.53	0.01	0.17	0.04	

LDA Algorithm

1. Randomly assign each word of each document to a topic;
2. For each document *d* and word *w*, calculate:
   - p(topic *t* | document *d*) - # words in *d* with *t* / # words in *d*
   - p(word *w* | topic *t*) - # assignments of *t* to *w* / # assignments to *t*
3. Update p(word *w* with topic *t*) = p(topic *t* | document *d*) * p(word *w* | topic *t*)
4. Repeat!

LDA in Python

- Pre-processing — `nltk`
  - stop word removal
  - stemming/lemmatization
- Dictionary building
- Creation of a bag-of-words
- Generating the LDA model — `gensim`
- Results
- Evaluation
- Visualization — `WordCloud, pyLDAvis`

LDA in Python - Pre-processing

```
In [4]: en_stop = set(nltk.corpus.stopwords.words('english'))

In [5]: en_stop.add("student")
        en_stop.add("online")
        en_stop.add("university")
```

LDA in Python - Lemmatization

In [7]:

```
import re
from nltk.stem import WordNetLemmatizer

stemmer = WordNetLemmatizer()

def preprocess_text(document):
        # Remove all the special characters
        document = re.sub(r'\W', ' ', str(document))

        # remove all single characters
        document = re.sub(r'\s+[a-zA-Z]\s+', ' ', document)

        # Remove single characters from the start
        document = re.sub(r'\^[a-zA-Z]\s+', ' ', document)

        # Substituting multiple spaces with single space
        document = re.sub(r'\s+', ' ', document, flags=re.I)

        # Removing prefixed 'b'
        document = re.sub(r'^b\s+', '', document)

        # Converting to Lowercase
        document = document.lower()

        # Lemmatization
        tokens = document.split()
        tokens = [stemmer.lemmatize(word) for word in tokens]
        tokens = [word for word in tokens if word not in en_stop]
        tokens = [word for word in tokens if len(word)  > 5]

        return tokens
```

LDA in Python - Model generation

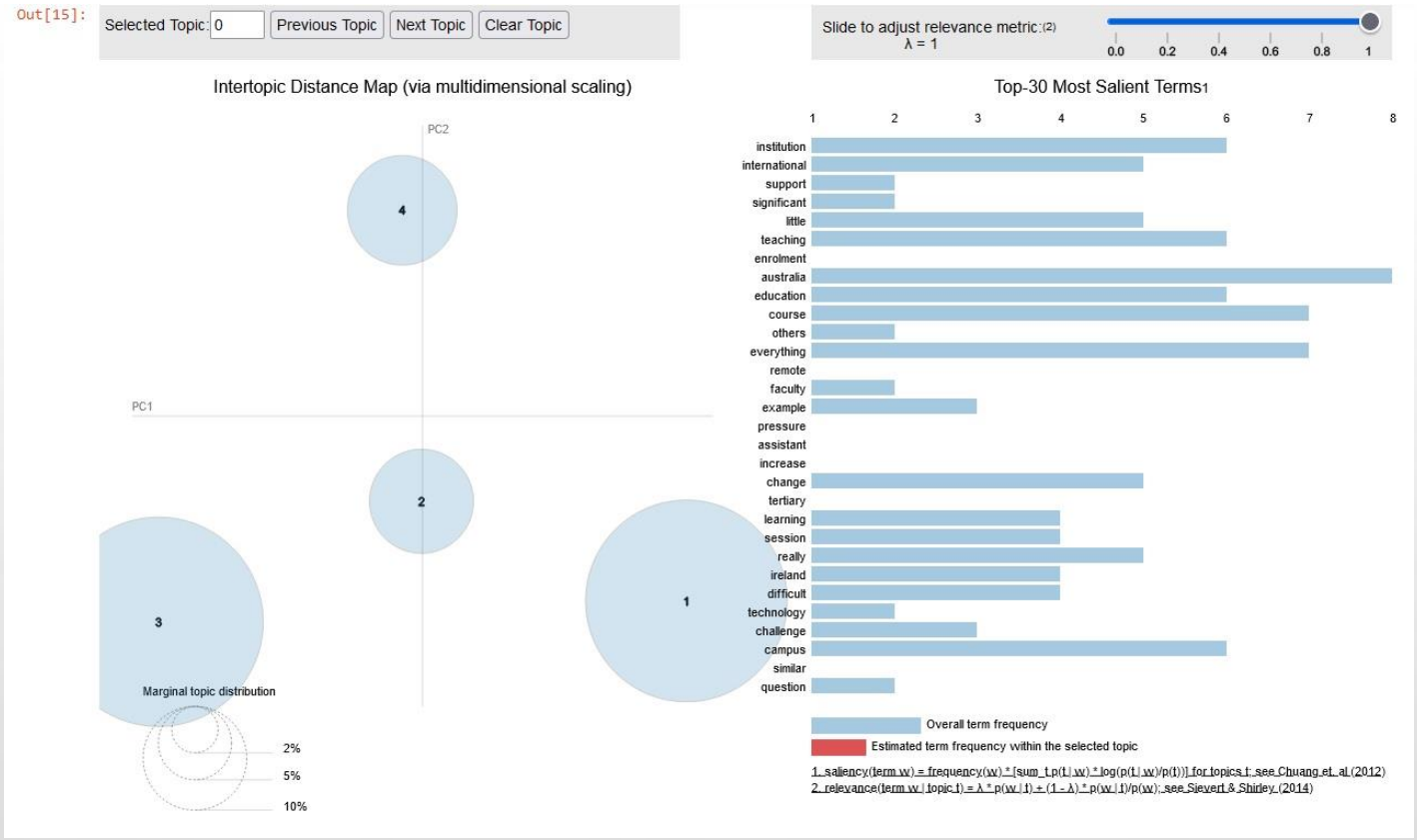
In [11]:
```
import gensim

lda_model = gensim.models.ldamodel.LdaModel(gensim_corpus, num_topics=4, id2word=gensim_dictionary, passes=5)
lda_model.save('gensim_model.gensim')
```

In [12]:
```
topics = lda_model.print_topics(num_words=10)
for topic in topics:
    print(topic)
```

```
(0, '0.026*"australia" + 0.019*"education" + 0.015*"change" + 0.015*"changed" + 0.013*"ireland" + 0.013*"international" + 0.012*"different" + 0.011*"little" + 0.011*"
academic" + 0.011*"melbourne"')
(1, '0.016*"course" + 0.014*"little" + 0.014*"remote" + 0.012*"teaching" + 0.012*"assistant" + 0.011*"really" + 0.011*"session" + 0.010*"example" + 0.010*"change" +
0.008*"technology"')
(2, '0.023*"everything" + 0.018*"different" + 0.018*"campus" + 0.018*"teaching" + 0.015*"learning" + 0.013*"course" + 0.013*"difficult" + 0.013*"experience" + 0.011*"
teacher" + 0.011*"institution"')
(3, '0.028*"institution" + 0.023*"support" + 0.023*"significant" + 0.018*"international" + 0.017*"enrolment" + 0.012*"little" + 0.012*"faculty" + 0.012*"others" + 0.0
12*"pressure" + 0.012*"increase"')
```

Visualization of results



Funded by
the European Union

Thank you for your attention