3803ICT
Big Data Analysis

Lab 07 – Textual Data Analytics

Trimester 1 - 2020

Table of Contents

Complete the code with TODO tag in the Jupyter notebooks.

1. Feature Engineering

In this exercise we will understand the functioning of TF/IDF ranking. Implement the feature engineering and its application, based on the code framework provided below. Please read the file feature_engineering.ipynb

First, we use textual data from Twitter.

	id	created_at	text
0	849636868052275200	2017-04-05 14:56:29	b'And so the robots spared humanity ... https:...
1	848988730585096192	2017-04-03 20:01:01	b"@ForIn2020 @waltmossberg @mims @defcon_5 Exa...
2	848943072423497728	2017-04-03 16:59:35	b'@waltmossberg @mims @defcon_5 Et tu, Walt?'
3	848935705057280001	2017-04-03 16:30:19	b'Stormy weather in Shortville ...'
4	848416049573658624	2017-04-02 06:05:23	b"@DaveLeeBBC @verge Coal is dying due to nat ...

1.1. Text Normalization

Now we need to normalize text by stemming, tokenizing, and removing stopwords.

As you can see that the normalization is still not perfect. Please feel free to improve upon (OPTIONAL), e.g. https://marcobonzanini.com/2015/03/09/mining-twitter-data-with-python-part-2/

1.2. Implement TF-IDF

Now you need to implement TF-IDF, including creating the vocabulary, computing term frequency, and normalizing by tf-idf weights.

1.3. Compare the results with the reference implementation of scikit-learn library.

Now we use the scikit-learn library. As you can see that, the way we do text normalization affects the result. Feel free to further improve upon (OPTIONAL),
e.g. https://stackoverflow.com/questions/36182502/add-stemming-support-to-countvectorizer-sklearn

```
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
from sklearn.metrics.pairwise import linear_kernel

tfidf = TfidfVectorizer(analyzer='word', ngram_range=(1,1), min_df = 1, stop_words = 'english', max_features=500)

features = tfidf.fit(original_documents)
corpus_tf_idf = tfidf.transform(original_documents)

sum_words = corpus_tf_idf.sum(axis=0)
words_freq = [(word, sum_words[0, idx]) for word, idx in tfidf.vocabulary_.items()]
print(sorted(words_freq, key = lambda x: x[1], reverse=True)[:5])
print('testla', corpus_tf_idf[1, features.vocabulary_['tesla']])
```

```
[('http', 163.54366542841234), ('https', 151.85039944652075), ('rt', 112.61998731390989), ('tesla', 95.96401470715628
1), ('xe2', 88.209444863464768)]
testla 0.349524310066
```

1.4. Apply TF-IDF for information retrieval

We can use the vector representation of documents to implement an information retrieval system. We test with the query Q = "tesla nasa"

```
Top-5 documents
0 b'@ashwin7002 @NASA @faa @AFPAA We have not ruled that out.'
1 b'RT @NASA: Updated @SpaceX #Dragon #ISS rendezvous times: NASA TV coverage begins Sunday at 3:30amET: http://t.co/
qrm0Dz4jPE. Grapple at  ...'
2 b"Deeply appreciate @NASA's faith in @SpaceX. We will do whatever it takes to make NASA and the American people pro
ud."
3 b'Would also like to congratulate @Boeing, fellow winner of the @NASA commercial crew program'
4 b"@astrostephenson We're aiming for late 2015, but NASA needs to have overlapping capability to be safe. Would do t
he same"
```

We can also use the scikit-learn library to do the retrieval.

```
new_features = tfidf.transform([query])

cosine_similarities = linear_kernel(new_features, corpus_tf_idf).flatten()
related_docs_indices = cosine_similarities.argsort()[::-1]

topk = 5
print('Top-{0} documents'.format(topk))
for i in range(topk):
    print(i, original_documents[related_docs_indices[i]])
```

```
Top-5 documents
0 b'RT @NASA: Updated @SpaceX #Dragon #ISS rendezvous times: NASA TV coverage begins Sunday at 3:30amET: http://t.co/
qrm0Dz4jPE. Grapple at  ...'
1 b"Deeply appreciate @NASA's faith in @SpaceX. We will do whatever it takes to make NASA and the American people pro
ud."
2 b'@NASA Best of luck to the Cygnus launch'
3 b'RT @SpaceX: Success! Congrats @NASA on @MarsCuriosity!'
4 b'@ashwin7002 @NASA @faa @AFPAA We have not ruled that out.'
```

2. Sentiment Analysis

In this exercise, we will classify the sentiment of text documents. Complete the code with TODO tag.
References and Further Readings:

- http://www.nltk.org/howto/sentiment.html
- https://www.nltk.org/api/nltk.sentiment.html
- http://datameetsmedia.com/vader-sentiment-analysis-explained/
- https://github.com/cjhutto/vaderSentiment
- https://marcobonzanini.com/2015/05/17/mining-twitter-data-with-python-part-6-sentiment-analysis-basics/
- https://github.com/marrrcin/ml-twitter-sentiment-analysis

2.1. Classification approach

Classification approach looks at previously labeled data in order to determine the sentiment of never-before-seen sentences. It involves training a model using previously seen text to predict/classify the sentiment of some new input text. The nice thing is that, with a greater volume of data, we generally get better prediction or classification results. However, unlike the lexical approach, we need previously labeled data.

```
[nltk_data] Downloading package movie_reviews to
[nltk_data]     /Users/tnguyen/nltk_data...
[nltk_data]   Package movie_reviews is already up-to-date!

(1000, 1000)
```

Each document is represented by a tuple (sentence, label). The sentence is tokenized, so it is represented by a list of strings.

We use simple unigram word features, handling negation.

We separately split subjective and objective instances to keep a balanced uniform class distribution in both train and test sets.

We apply features to obtain a feature-value representation of our datasets.

We can now train our classifier on the training set, and subsequently output the evaluation results.

```
Training classifier
Evaluating NaiveBayesClassifier results...

{'Accuracy': 0.725,
 'F-measure [neg]': 0.717948717948718,
 'F-measure [pos]': 0.7317073170731707,
 'Precision [neg]': 0.7368421052631579,
 'Precision [pos]': 0.7142857142857143,
 'Recall [neg]': 0.7,
 'Recall [pos]': 0.75}
```

2.2. Lexical approach

Lexical approaches aim to map words to sentiment by building a lexicon or a 'dictionary of sentiment'. We can use this dictionary to assess the sentiment of phrases and sentences, without the need of looking at anything else. Sentiment can be categorical – such as {negative, neutral, positive} – or it can be numerical – like a range of intensities or scores. Lexical approaches look at the sentiment category or score of each word in the sentence and decide what the sentiment category or score of the whole sentence is. The power of lexical approaches lies in the fact that we do not need to train a model using labeled data, since we have everything we need to assess the sentiment of sentences in the dictionary of emotions. VADER is an example of a lexical method.

```
just how inseparable is the team of sgt . martin riggs ( mel gibson ) and sgt . roger murtaugh ( dan...
compound: 0.7656, neg: 0.103, neu: 0.791, pos: 0.106,
since 1990 , the dramatic picture has undergone a certain change of style . now , instead of emphasi...
compound: 0.9982, neg: 0.027, neu: 0.79, pos: 0.183,
a big surprise to me . the good trailer had hinted that they pulled the impossible off , but making ...
compound: 0.9779, neg: 0.145, neu: 0.65, pos: 0.205,
after having heard so many critics describe " return to me " as an old - fashioned hollywood romance...
compound: 0.9995, neg: 0.042, neu: 0.738, pos: 0.22,
wild things is a suspenseful thriller starring matt dillon , denise richards , and neve campbell tha...
compound: 0.9965, neg: 0.07, neu: 0.77, pos: 0.16,
* * * * * * minor plot spoilers in review * * * * * * * * * * * * no major spoilers are in review * ...
compound: 0.9904, neg: 0.073, neu: 0.82, pos: 0.107,
are you tired of all the hot new releases being gone by the time you get to the video store ? waffle...
compound: -0.9805, neg: 0.16, neu: 0.704, pos: 0.136,
```

2.3. Comparing the two approaches

First, we can transform the sentiment score by the lexical approach into label by the following rules:

- positive sentiment: compound score > 0
- negative sentiment: compound score <= 0

```
just how inseparable is the team of sgt . martin riggs ( mel gibson ) and sgt . roger murtaugh ( dan... pos
since 1990 , the dramatic picture has undergone a certain change of style . now , instead of emphasi... pos
a big surprise to me . the good trailer had hinted that they pulled the impossible off , but making ... pos
after having heard so many critics describe " return to me " as an old - fashioned hollywood romance... pos
wild things is a suspenseful thriller starring matt dillon , denise richards , and neve campbell tha... pos
* * * * * * minor plot spoilers in review * * * * * * * * * * * * no major spoilers are in review * ... pos
are you tired of all the hot new releases being gone by the time you get to the video store ? waffle... neg
many people dislike french films for their lack of closure . while possibly shallow , i ' ve often h... pos
the keen wisdom of an elderly bank robber , the naive ambitions of a sexy hospital nurse , and a par... pos
robert benton has assembled a stellar , mature cast for his latest feature , twilight , a film noir ... pos
warning : anyone offended by blatant , leering machismo had better avoid this film . or lots of bloo... neg
accepting his oscar as producer of this year ' s best picture winner , saul zaentz remarked that his... pos
a bleak look at how the boston underworld operates , a film which was based on the best - seller by ... neg
" footloose " has only one goal in mind : to reel in an audience with cheesy sentiment and feel - go... pos
as i walked out of crouching tiger , hidden dragon i thought to myself that i had had just seen a gr... pos
" crazy / beautiful " suffers from the damned - if - you - do , damned - if - you - don ' t syndrome... pos
everyone knows someone like giles de ' ath : stuffy , arrogant , set in his ways , and at war with a... pos
```

Now we evaluate the lexical approach by computing accuracy metrics

```
Accuracy: 0.6
F-measure [neg]: 0.5555555555555556
F-measure [pos]: 0.6363636363636364
Precision [neg]: 0.625
Precision [pos]: 0.5833333333333334
Recall [neg]: 0.5
Recall [pos]: 0.7
```