Question3

June 5, 2020

1 IFN619 - Assignment 3 - Stream A

1.0.1 QUESTION 3:

Business concern: National conversation of top Australian news. What were the top Australian news topics over the last decade, and what can these say about the national conversation?

For the above question, we have to analyse the news headlines of Australia over ten years. And their understanding and visualization of national conversation.

1. Importing Libraries In this task, we have imported all the necessary python libraries to analyse the questions that have been raised.

[358]:
```
# import libraries
import numpy as np              # used for algebraic operations
import pandas as pd             # used for data manipulation and data analysis
import matplotlib.pyplot as plt # used for visualisations
import seaborn as sns           # used for visualisations
from IPython.display import display
from IPython.display import Markdown # used for visualisations
import plotly.graph_objs as go       # used for visualisations
import plotly.offline as py          # used for visualisations
from plotly.subplots import make_subplots # used for visualisations
import plotly.express as px          # used for visualisations
import json

from tqdm import tqdm
from collections import Counter
import ast

import matplotlib.mlab as mlab
import seaborn as sb

from sklearn.feature_extraction.text import CountVectorizer
import scipy.stats as stats

from sklearn.decomposition import TruncatedSVD
```

```
from sklearn.decomposition import LatentDirichletAllocation
from sklearn.manifold import TSNE

from bokeh.plotting import figure, output_file, show
from bokeh.models import Label
from bokeh.io import output_notebook
output_notebook()

from sklearn.model_selection import train_test_split
import pandas as pd
import numpy as np
from sklearn.utils import shuffle
import nltk



%matplotlib inline
```

1.0.2 2.Read the data

For analyses we have used news dataset of a noteworthy events in the globe from early 2003 to end 2019 published by the abcnews website. In this section we have loaded the data and checked the datasets. Link to the abcnews website: https://www.kaggle.com/therohk/million-headlines

[359]:
```
# read the CSV data into a dataframe variable (df)
df1 = pd.read_csv("abcnews-date-text-test.csv")
df1     #checking  dataset
```

[359]:

	PUBLISHED_DATE	HEADLINE_TEXT
0	19/2/03	aba decides against community broadcasting lic…
1	19/2/03	act fire witnesses must be aware of defamation
2	19/2/03	a g calls for infrastructure protection summit
3	19/2/03	air nz staff in aust strike for pay rise
4	19/2/03	air nz strike to affect australian travellers
…	…	…
1048570	18/9/16	bill de blasio no evidence of terror connection
1048571	18/9/16	boyer lectures michael marmot work harming you…
1048572	18/9/16	carnaby cockatoo revival program
1048573	18/9/16	collier convinced colin barnett has overwhelmi…
1048574	18/9/16	cowboys offer indigenous kids north queensland…

[1048575 rows x 2 columns]

[360]:
```
datafile = "abcnews-date-text-test.csv"
raw_data = pd.read_csv(datafile, parse_dates=[0], infer_datetime_format = True)
```

```
reindexed_data = raw_data['HEADLINE_TEXT']
reindexed_data.index = raw_data['PUBLISHED_DATE']

raw_data.head()
```

[360]:

	PUBLISHED_DATE	HEADLINE_TEXT
0	2003-02-19	aba decides against community broadcasting lic…
1	2003-02-19	act fire witnesses must be aware of defamation
2	2003-02-19	a g calls for infrastructure protection summit
3	2003-02-19	air nz staff in aust strike for pay rise
4	2003-02-19	air nz strike to affect australian travellers

3. General information extraction

[361]:
```
# Extracting general inforamtion

print("The dataset contains %d rows and %d columns" %(df1.shape[0], df1.
 ↪shape[1]))  #to view number of rows and columns
print(df1.dtypes)        #gives datatype of each column
```

```
The dataset contains 1048575 rows and 2 columns
PUBLISHED_DATE    object
HEADLINE_TEXT     object
dtype: object
```

[362]:
```
# extract the variables of this dataset
vars = df1.columns.tolist()
vars
```

[362]:
```
['PUBLISHED_DATE', 'HEADLINE_TEXT']
```

[363]:
```
df1.describe()           #Gives and genral overview of dataset
```

[363]:

	PUBLISHED_DATE	HEADLINE_TEXT
count	1048575	1048575
unique	4953	1021130
top	24/8/12	national rural news
freq	384	808

4. Clean the Data The data used is the government data hence there were no missing values and data was mostly in the structured format except few changes that had to be made

[364]:
```
df1.info() #checking for null values
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1048575 entries, 0 to 1048574
```

```
Data columns (total 2 columns):
PUBLISHED_DATE    1048575 non-null object
HEADLINE_TEXT     1048575 non-null object
dtypes: object(2)
memory usage: 16.0+ MB
```

[365]:
```
# we can remove all repeated elements of the columns by
# calling the unique() method

df1.HEADLINE_TEXT.unique()
```

[365]:
```
array(['aba decides against community broadcasting licence',
       'act fire witnesses must be aware of defamation',
       'a g calls for infrastructure protection summit', …,
       'carnaby cockatoo revival program',
       'collier convinced colin barnett has overwhelming support',
       'cowboys offer indigenous kids north queensland hope for future'],
      dtype=object)
```

[366]:
```
news_list = df1.groupby('HEADLINE_TEXT').count()
news_list
```

[366]:
```
                                                PUBLISHED_DATE
HEADLINE_TEXT
$1 billion darling harbour redevelopment plans …               1
$1 billion desalination claim frivolous walsh                  1
$1 billion wiped from nsw budget in accounting …               1
$1 million wall to shield coal train noise                     1
$1.3 billion to fund life saving drugs                         1
…                                                              …
zygier sabotaged mission to retrieve soldiers                  1
zylvester streaks field in alice cup                           1
zynga adoption pushes bitcoin back over 1000                   1
zyngier privatisation of schools                               1
zz top to tour australia                                       1

[1021130 rows x 1 columns]
```

[367]:
```
# the data is cummulative. We can group the information by country by getting
# the last value reported in the data, in order to get more realistic values
news_list = df1.iloc[:,0:].groupby(df1.index).last()

# let's take a look at the confirmed cases
news_list
```

[367]:
```
        PUBLISHED_DATE                                      HEADLINE_TEXT
0              19/2/03  aba decides against community broadcasting lic…
```

```
1              19/2/03     act fire witnesses must be aware of defamation
2              19/2/03     a g calls for infrastructure protection summit
3              19/2/03           air nz staff in aust strike for pay rise
4              19/2/03      air nz strike to affect australian travellers
…                …                                                    …
1048570        18/9/16    bill de blasio no evidence of terror connection
1048571        18/9/16  boyer lectures michael marmot work harming you…
1048572        18/9/16                    carnaby cockatoo revival program
1048573        18/9/16  collier convinced colin barnett has overwhelmi…
1048574        18/9/16  cowboys offer indigenous kids north queensland…

[1048575 rows x 2 columns]
```

[368]:
```
#Slicing out the data of last decade from the original dataset.
start_date = '1-1-2006'
end_date = '18-9-2016'
pd.period_range(start= start_date, end= end_date, freq='Y')
last_decade = df1[(df1['PUBLISHED_DATE']>= start_date) &␣
 ↪(df1['PUBLISHED_DATE']<= (end_date))]
reindexed_data = last_decade['HEADLINE_TEXT']
reindexed_data.index = last_decade['PUBLISHED_DATE']

print(last_decade)
```

```
       PUBLISHED_DATE                                     HEADLINE_TEXT
2180            1/3/03          30 million landmines destroyed worldwide
2181            1/3/03    adelaide international film festival kicks off
2182            1/3/03                 airpark planned for port douglas
2183            1/3/03            alinghi march delayed by postponement
2184            1/3/03                  alinghi poised for historic win
…                 …                                                  …
1048557        17/9/16  what the key players say about the cowboys bro…
1048558        17/9/16        winx claims george main stakes at randwick
1048559        17/9/16  woman stabbed in chest hand during sydney sout…
1048560        17/9/16  women afl players more receptive to coaching t…
1048561        17/9/16  young archies winning artworks show positive i…

[313140 rows x 2 columns]
```

[369]:
```
last_decade.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 313140 entries, 2180 to 1048561
Data columns (total 2 columns):
PUBLISHED_DATE    313140 non-null object
HEADLINE_TEXT     313140 non-null object
dtypes: object(2)
```

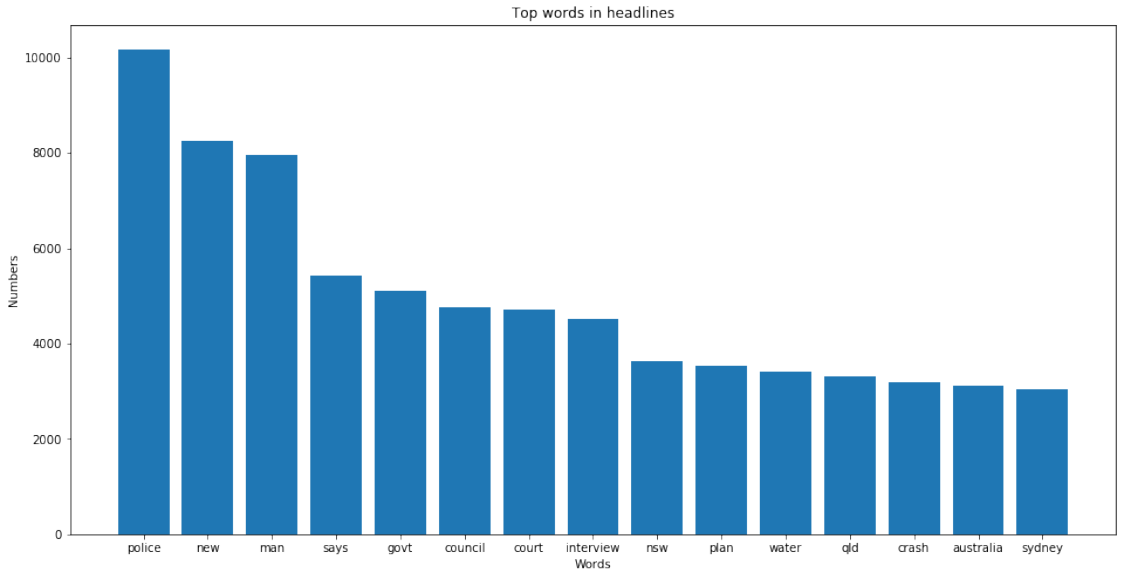
```
memory usage: 7.2+ MB
```

1.0.3 5.Analysis and visuaization.

Here we analyse the data using the headlines and the published dates of abcnews website, how many news were reported per day and their data. Visualization is shown by graphical data analysis and explains the patterns in the data represented by the stakeholders.

[370]:
```
# returns top words in a dataset
def get_top_n_words(n_top_words, count_vectorizer, text_data):

    vectorized_headlines = count_vectorizer.fit_transform(text_data.values)
    vectorized_total = np.sum(vectorized_headlines, axis=0)
    word_indices = np.flip(np.argsort(vectorized_total)[0,:], 1)
    word_values = np.flip(np.sort(vectorized_total)[0,:],1)

    word_vectors = np.zeros((n_top_words, vectorized_headlines.shape[1]))
    for i in range(n_top_words):
        word_vectors[i,word_indices[0,i]] = 1

    words = [word[0].encode('ascii').decode('utf-8') for
             word in count_vectorizer.inverse_transform(word_vectors)]

    return (words, word_values[0,:n_top_words].tolist()[0])
```

[371]:
```
count_vectorizer = CountVectorizer(stop_words='english')
words, word_values = get_top_n_words(n_top_words=15,
                                     count_vectorizer=count_vectorizer,
                                     text_data=reindexed_data)

fig, ax = plt.subplots(figsize=(16,8))
ax.bar(range(len(words)), word_values);
ax.set_xticks(range(len(words)));
ax.set_xticklabels(words, rotation='horizontal');
ax.set_title('Top words in headlines');
ax.set_xlabel('Words');
ax.set_ylabel('Numbers');
plt.show()
```



1.0.4 Sentimental Analysis

[372]:
```
SA = last_decade['HEADLINE_TEXT']
```

[373]:
```
all_reviews = last_decade['HEADLINE_TEXT']
all_sent_values = []
all_sentiments = []
```

[374]:
```
from nltk.sentiment.vader import SentimentIntensityAnalyzer
def sentiment_value(paragraph):
    analyser = SentimentIntensityAnalyzer()
    result = analyser.polarity_scores(paragraph)
    score = result['compound']
    return round(score,1)
```

[375]:
```
sample = last_decade['HEADLINE_TEXT'][109880]
print(sample)
print('Sentiment: ')
print(sentiment_value(sample))
```

```
ramos horta addresses public
Sentiment:
0.0
```

[376]:
```
sample1 = last_decade['HEADLINE_TEXT'][50394]
print(sample1)
print('Sentiment: ')
```

```
print(sentiment_value(sample1))
```

```
children suffering on hospital waiting lists
Sentiment:
-0.5
```

[377]:
```
sample2 = last_decade['HEADLINE_TEXT'][100534]
print(sample2)
print('Sentiment: ')
print(sentiment_value(sample2))
```

```
grey nomads in tas not so grey
Sentiment:
0.1
```

[378]:
```
sample3 = last_decade['HEADLINE_TEXT'][10000]
print(sample3)
print('Sentiment: ')
print(sentiment_value(sample3))
```

```
tourism downturn hits jupiters profit
Sentiment:
0.4
```

[379]:
```
! pip install nltk
import nltk
nltk.download('stopwords')

nltk.download('vader_lexicon')
```

```
WARNING: pip is being invoked by an old script wrapper. This will fail in a
future version of pip.
Please see https://github.com/pypa/pip/issues/5599 for advice on fixing the
underlying issue.
To avoid this problem you can invoke Python with '-m pip' instead of running pip
directly.
Requirement already satisfied: nltk in /opt/conda/lib/python3.7/site-packages
(3.5)
Requirement already satisfied: tqdm in /opt/conda/lib/python3.7/site-packages
(from nltk) (4.42.0)
Requirement already satisfied: regex in /opt/conda/lib/python3.7/site-packages
(from nltk) (2020.5.14)
Requirement already satisfied: joblib in /opt/conda/lib/python3.7/site-packages
(from nltk) (0.14.1)
Requirement already satisfied: click in /opt/conda/lib/python3.7/site-packages
(from nltk) (7.0)

[nltk_data] Downloading package stopwords to /home/jovyan/nltk_data…
```

```
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package vader_lexicon to
[nltk_data]     /home/jovyan/nltk_data…
[nltk_data]   Package vader_lexicon is already up-to-date!
```

[379]:
```
True
```

[380]:
```
from nltk.sentiment.vader import SentimentIntensityAnalyzer as SIA
from nltk.corpus import stopwords
StopWords = stopwords.words("english")
from nltk.stem import PorterStemmer
import seaborn as sns
stemmer = PorterStemmer()
```

[381]:
```
sia = SIA()

pos_list = []
neg_list = []
neu_list = []
strpos_list =[]
strneg_list = []
for post in SA:
    post = " ".join([stemmer.stem(word) for word in str(post).lower().split()␣
 ↪if word not in set(StopWords)])
    res = sia.polarity_scores(post)
    if res['compound'] > 0.5:
        strpos_list.append(post)
    elif res['compound'] > 0.1:
        pos_list.append(post)
    elif res['compound'] < 0.0:
        neu_list.append(post)
    elif res['compound'] < 0.1:
        neg_list.append(post)
    else:
        str_neglist.append(post)
```

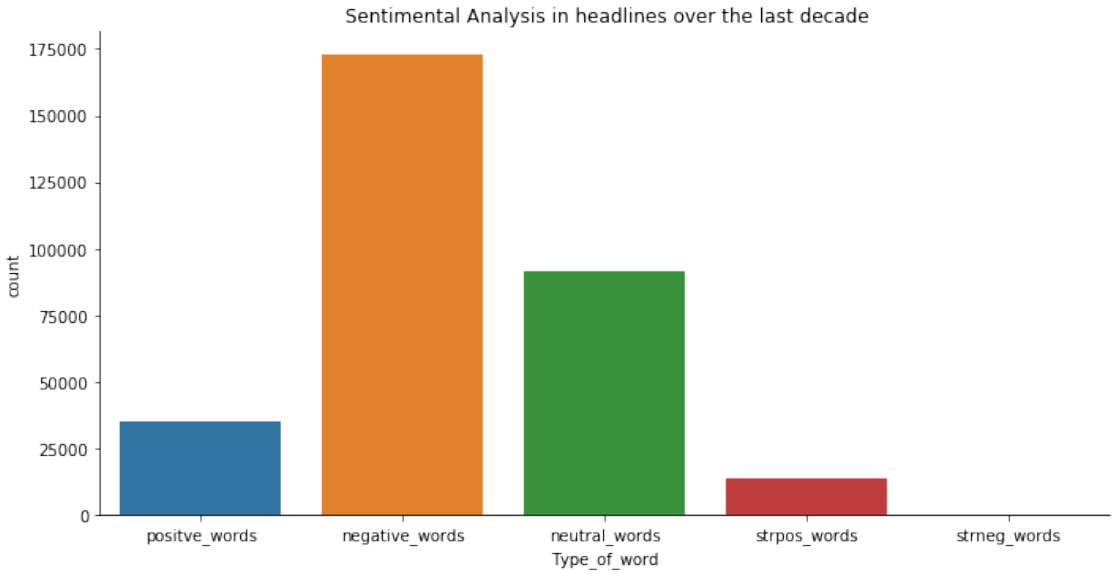
[382]:
```
v1 = []
v2 = []

v1 =␣
 ↪['positve_words','negative_words','neutral_words','strpos_words','strneg_words']
v2 = [len(pos_list),len(neg_list),len(neu_list), len(strpos_list),␣
 ↪len(strneg_list)]

u1 = pd.DataFrame(v1)
u2 = pd.DataFrame(v2)
```

```
titanic = pd.concat(objs = [u1, u2], axis = 1)
titanic.columns = ['Type_of_word','count']
print(titanic)
```

```
     Type_of_word   count
0   positve_words   35001
1  negative_words  172941
2   neutral_words   91439
3    strpos_words   13759
4    strneg_words       0
```

[383]:
```
c = sns.catplot(x="Type_of_word", y="count", kind="bar", data=titanic, aspect=2)
c.set(title = ' Sentimental Analysis in headlines over the last decade', xlabel␣
 ↪= 'Type_of_word', ylabel = 'count')
```

[383]: <seaborn.axisgrid.FacetGrid at 0x7f2a2ba6cc90>



1.1 6. Preprocessing the data for analysis

It can be achieved using SKLearn's CountVectorizer model, which yields a document-term matrix where K is the number of unique words in our sample across the n headlines less stop words and with a max features limit

[384]:
```
small_count_vectorizer = CountVectorizer(stop_words='english',␣
 ↪max_features=40000)
small_text_sample = reindexed_data.sample(n=10000, random_state=0).values
```

```
print('Headline before vectorization: {}'.format(small_text_sample[123]))

small_document_term_matrix = small_count_vectorizer.
 ↪fit_transform(small_text_sample)

print('Headline after vectorization: {}'.
 ↪format(small_document_term_matrix[123]))
```

```
Headline before vectorization: australian students killer loses appeal to
overturn sentence
Headline after vectorization:   (0, 10402)      1
  (0, 688)      1
  (0, 912)      1
  (0, 5972)     1
  (0, 6443)     1
  (0, 7683)     1
  (0, 9652)     1
```

[385]:
```
n_topics = 15
```

1.1.1 Top news using LSA

[386]:
```
lsa_model = TruncatedSVD(n_components=n_topics)
lsa_topic_matrix = lsa_model.fit_transform(small_document_term_matrix)
```

[387]:
```
def get_keys(topic_matrix):
    '''
    returns an integer list of predicted topic
    categories for a given topic matrix
    '''
    keys = topic_matrix.argmax(axis=1).tolist()
    return keys

def keys_to_counts(keys):
    '''
    returns a tuple of topic categories and their
    accompanying magnitudes for a given list of keys
    '''
    count_pairs = Counter(keys).items()
    categories = [pair[0] for pair in count_pairs]
    counts = [pair[1] for pair in count_pairs]
    return (categories, counts)
```

[388]:
```
lsa_keys = get_keys(lsa_topic_matrix)
lsa_categories, lsa_counts = keys_to_counts(lsa_keys)
```

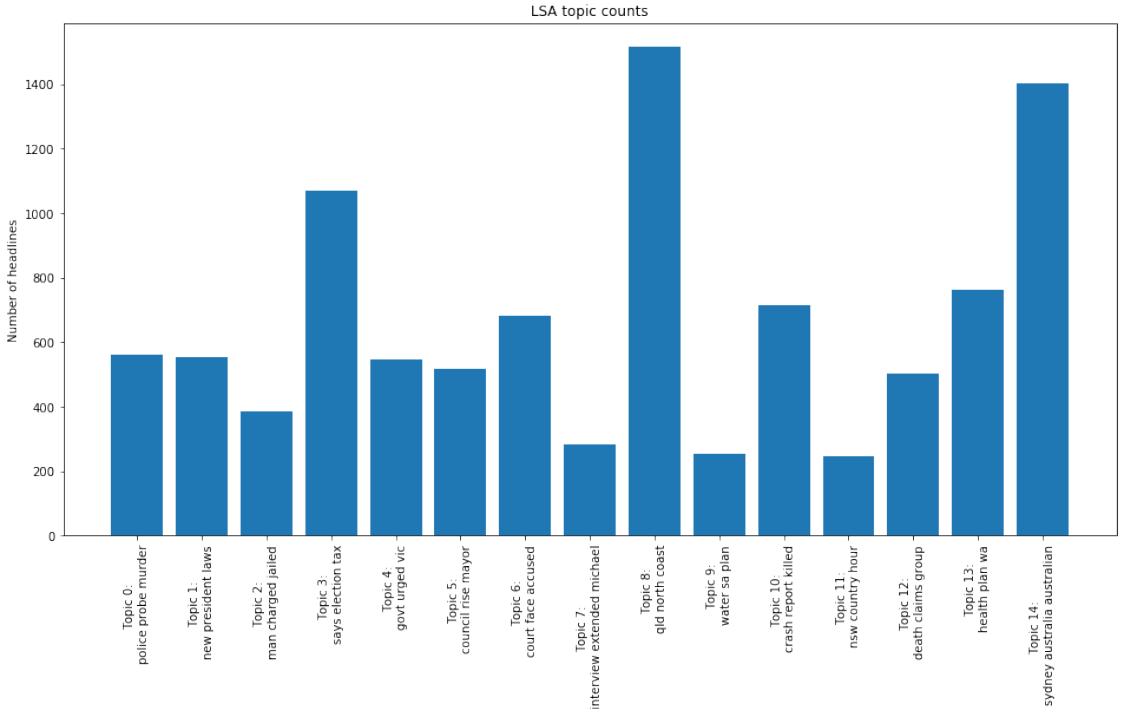
[389]:
```
def get_top_n_words(n, keys, document_term_matrix, count_vectorizer):
    '''
    returns a list of n_topic strings, where each string contains the n most␣
 ↪common
    words in a predicted category, in order
    '''
    top_word_indices = []
    for topic in range(n_topics):
        temp_vector_sum = 0
        for i in range(len(keys)):
            if keys[i] == topic:
                temp_vector_sum += document_term_matrix[i]
        temp_vector_sum = temp_vector_sum.toarray()
        top_n_word_indices = np.flip(np.argsort(temp_vector_sum)[0][-n:],0)
        top_word_indices.append(top_n_word_indices)
    top_words = []
    for topic in top_word_indices:
        topic_words = []
        for index in topic:
            temp_word_vector = np.zeros((1,document_term_matrix.shape[1]))
            temp_word_vector[:,index] = 1
            the_word = count_vectorizer.
 ↪inverse_transform(temp_word_vector)[0][0]
            topic_words.append(the_word.encode('ascii').decode('utf-8'))
        top_words.append(" ".join(topic_words))
    return top_words
```

[390]:
```
top_n_words_lsa = get_top_n_words(10, lsa_keys, small_document_term_matrix,␣
 ↪small_count_vectorizer)

for i in range(len(top_n_words_lsa)):
    print("Topic {}: ".format(i+1), top_n_words_lsa[i])
```

```
Topic 1:  police probe murder man hunt investigate shooting missing suspect
perth
Topic 2:  new president laws security party opens school record england closer
Topic 3:  man charged jailed murder guilty court years jail child sex
Topic 4:  says election tax minister changes ban opposition jobs wont fears
Topic 5:  govt urged vic defends urges public local scheme funds port
Topic 6:  council rise mayor gets rates city considers rate brisbane close
Topic 7:  court face accused high home case drug charges told trial
Topic 8:  interview extended michael peter nrl john speaks paul ben kevin
Topic 9:  qld north coast market farmers industry road residents business
queensland
Topic 10:  water sa plan canberra murray restrictions school act dam supply
Topic 11:  crash report killed car attack woman dies dead hospital abc
Topic 12:  nsw country hour rural friday 2014 nrn drum 13 tas
```

```
Topic 13:  death claims group open deal government china rises toll probe
Topic 14:  health plan wa calls work power indigenous act mp workers
Topic 15:  sydney australia australian world cup set win south west day
```

[401]:
```
top_3_words = get_top_n_words(3, lsa_keys, small_document_term_matrix,␣
 ↪small_count_vectorizer)
labels = ['Topic {}: \n'.format(i) + top_3_words[i] for i in lsa_categories]

fig, ax = plt.subplots(figsize=(16,8))
ax.bar(lsa_categories, lsa_counts);
ax.set_xticks(lsa_categories);
ax.set_xticklabels(labels,rotation='vertical');
ax.set_ylabel('Number of headlines');
ax.set_title('LSA topic counts');
plt.show()
```



1.1.2 Top news using LDA

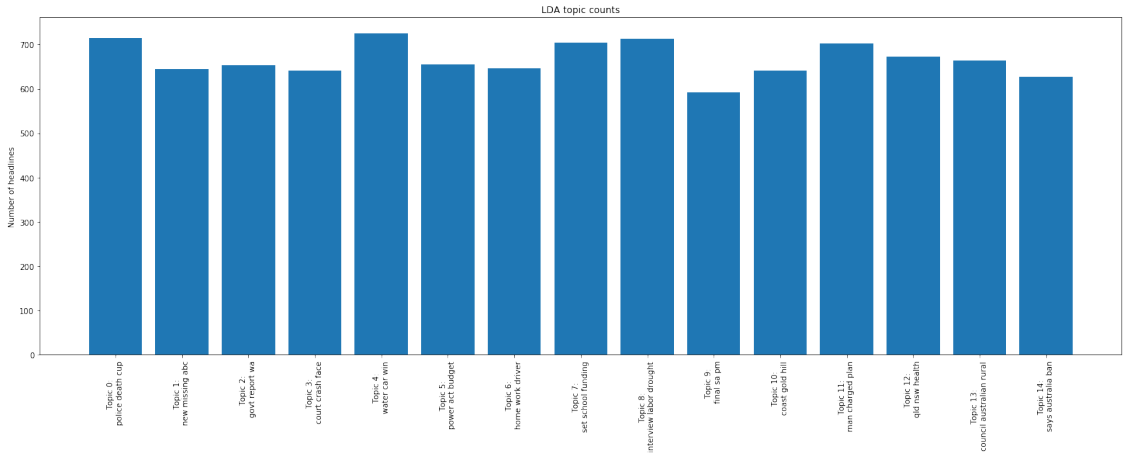
[392]:
```
lda_model = LatentDirichletAllocation(n_components=n_topics,␣
 ↪learning_method='online',
                                      random_state=0, verbose=0)
lda_topic_matrix = lda_model.fit_transform(small_document_term_matrix)
```

[393]:
```
lda_keys = get_keys(lda_topic_matrix)
lda_categories, lda_counts = keys_to_counts(lda_keys)
```

[394]:
```
top_n_words_lda = get_top_n_words(10, lda_keys, small_document_term_matrix,␣
 ↪small_count_vectorizer)

for i in range(len(top_n_words_lda)):
    print("Topic {}: ".format(i+1), top_n_words_lda[i])
```

```
Topic 1:  police death cup world hit market guilty warning public drug
Topic 2:  new missing abc country business day hour industry search wins
Topic 3:  govt report wa attack government workers asylum victory rail closer
Topic 4:  court crash face hospital man claims sex funds finds accused
Topic 5:  water car win woman murder man big police obama opposition
Topic 6:  power act budget residents dead end station queensland park rain
Topic 7:  home work driver urges sri aussie case parliament cyclone concerns
Topic 8:  set school funding indigenous urged open boost new backs future
Topic 9:  interview labor drought help abuse police rejects vote nrl shooting
Topic 10:  final sa pm road calls canberra jail bank deal murray
Topic 11:  coast gold hill media arrested broken flood drum tour afl
Topic 12:  man charged plan fears group police mayor child fight court
Topic 13:  qld nsw health north election china killed minister test year
Topic 14:  council australian rural sydney national changes festival black talks
gets
Topic 15:  says australia ban record high south change accused new farmers
```

[400]:
```
top_3_words = get_top_n_words(3, lda_keys, small_document_term_matrix,␣
 ↪small_count_vectorizer)
labels = ['Topic {}: \n'.format(i) + top_3_words[i] for i in lda_categories]

fig, ax = plt.subplots(figsize=(25,8))
ax.bar(lda_categories, lda_counts);
ax.set_xticks(lda_categories);
ax.set_xticklabels(labels,rotation='vertical' );
ax.set_title('LDA topic counts');
ax.set_ylabel('Number of headlines');
```



However, with the data points clustering algorithm can be implemented using either Latent Dirichilet Allocation. This analysis will take the term matrix as input and output of n*N topic matrix, where N is the parameter for number of topic categories

[396]:
```
tsne_lda_model = TSNE(n_components=2, perplexity=50, learning_rate=100,
                      n_iter=2000, verbose=1, random_state=0, angle=0.75)
tsne_lda_vectors = tsne_lda_model.fit_transform(lda_topic_matrix)
```

```
[t-SNE] Computing 151 nearest neighbors…
[t-SNE] Indexed 10000 samples in 0.016s…
[t-SNE] Computed neighbors for 10000 samples in 2.913s…
[t-SNE] Computed conditional probabilities for sample 1000 / 10000
[t-SNE] Computed conditional probabilities for sample 2000 / 10000
[t-SNE] Computed conditional probabilities for sample 3000 / 10000
[t-SNE] Computed conditional probabilities for sample 4000 / 10000
[t-SNE] Computed conditional probabilities for sample 5000 / 10000
[t-SNE] Computed conditional probabilities for sample 6000 / 10000
[t-SNE] Computed conditional probabilities for sample 7000 / 10000
[t-SNE] Computed conditional probabilities for sample 8000 / 10000
[t-SNE] Computed conditional probabilities for sample 9000 / 10000
[t-SNE] Computed conditional probabilities for sample 10000 / 10000
[t-SNE] Mean sigma: 0.000000
[t-SNE] KL divergence after 250 iterations with early exaggeration: 88.116547
[t-SNE] KL divergence after 2000 iterations: 1.861940
```

Remaining data points is to plot the clustered headlines. The top three words in each cluster are also included, which are placed at the centre of this topic

[397]:
```
def get_mean_topic_vectors(keys, two_dim_vectors):
    '''
    returns a list of centroid vectors from each predicted topic category
    '''
```

```
    mean_topic_vectors = []
    for t in range(n_topics):
        articles_in_that_topic = []
        for i in range(len(keys)):
            if keys[i] == t:
                articles_in_that_topic.append(two_dim_vectors[i])

        articles_in_that_topic = np.vstack(articles_in_that_topic)
        mean_article_in_that_topic = np.mean(articles_in_that_topic, axis=0)
        mean_topic_vectors.append(mean_article_in_that_topic)
    return mean_topic_vectors
```

[398]:
```
colormap = np.array([
    "#1f77b4", "#aec7e8", "#ff7f0e", "#ffbb78", "#2ca02c",
    "#98df8a", "#d62728", "#ff9896", "#9467bd", "#c5b0d5",
    "#8c564b", "#c49c94", "#e377c2", "#f7b6d2", "#7f7f7f",
    "#c7c7c7", "#bcbd22", "#dbdb8d", "#17becf", "#9edae5" ])
colormap = colormap[:n_topics]
```

[399]:
```
top_3_words_lda = get_top_n_words(3, lda_keys, small_document_term_matrix,␣
 ↪small_count_vectorizer)
lda_mean_topic_vectors = get_mean_topic_vectors(lda_keys, tsne_lda_vectors)

plot = figure(title="t-SNE Clustering of {} LDA Topics".format(n_topics),␣
 ↪plot_width=700, plot_height=700)
plot.scatter(x=tsne_lda_vectors[:,0], y=tsne_lda_vectors[:,1],␣
 ↪color=colormap[lda_keys])

for t in range(n_topics):
    label = Label(x=lda_mean_topic_vectors[t][0],␣
 ↪y=lda_mean_topic_vectors[t][1],
                  text=top_3_words_lda[t], text_color=colormap[t])
    plot.add_layout(label)

show(plot)
```

6.Insights

- The abcnews have been reported approx to 1 million and since the year 2011 more news are published.
- The Top words were sorted and police cases have been reported more compared with other cases.
- From the analysis the death cases are reported low that is 10 thousand.
- From the semantic analysis the count for strong negative shows zero

- From the LSA technique the top 15 topics shows police cases high and australia world cup as low
- From the LDA technique the top 15 topic shows the police case as high and the australian ban list as low