In [1]:
```
# Import needed libraries

import findspark
findspark.init('/usr/hdp/2.6.5.0-292/spark2')

# Create a Spark Context which will be used for distributed data processing

import pyspark
sc = pyspark.SparkContext(appName="Twitter Topic Sentiment")

import string

import re as re

import nltk

import time

from pyspark.sql import SQLContext

from pyspark.sql.types import *

from pyspark.sql.functions import monotonically_increasing_id

from pyspark.mllib.util import MLUtils

from pyspark.ml.feature import RegexTokenizer, Tokenizer, StopWordsRemover, CountV

from pyspark.mllib.clustering import LDA, LDAModel

nltk.download('stopwords')

from nltk.corpus import stopwords

from pyspark.mllib.linalg import Vector as oldVector, Vectors as oldVectors

from pyspark.ml.linalg import Vector as newVector, Vectors as newVectors

from pyspark.ml.feature import IDF

import numpy as np

import matplotlib.pyplot as plt

import pyspark.sql.functions as func
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     /home/vagrant/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

In [2]:
```
# Create an SQL Context which will be used for sql like distriburted data processi

# As I get more familiar with what technology to use where I will be switching bet

# pyspark dataframes, and pandas dataframes

sqlContext = SQLContext(sc)
```

In [3]:
```
# Hadoop is the filesystem being used. This is a three node virtual cluster

# Read in data from Hadoop

ITData = sc.textFile("hdfs:////user/vagrant/practicum/input")
```

In [4]:
```
# Output sample of data

ITData.take(5)
```

Out[4]:
```
[u'timetext,tweetid,tweetsource,tweettruncated,tweettext,tweetuserscreenname,twee
tuserid,tweetuserlocation,tweetuserdescription,tweetuserfollowerscount,tweetusers
tatusescount,tweetusertimezone,tweetusergioenabled,tweetuserlang,tweetcoordinates
coordinates,tweetplacecountry,tweetplacecountrycode,tweetplacefullname,tweetplace
name,tweetplacetype',
 u'20180606021849,1004185815147163648,a hrefhttptwittercomdownloadandroid relnofo
llowTwitter for Androida,False,AgeroNews data science teams presentation at Spark
AISummit today httpstco7F2IAUvN4o,chyuck,139348324,Stoneham  MA,Software Engineer
Manchester United FC and FC Lokomotiv Moscow fan Interests soccer  programming  f
oosball  cross country running  chess  melodic death metal,57,1413,,True,en,,,,,,
',
 u'20180606021850,1004185819119259648,a hrefhttptwittercomdownloadiphone relnofol
lowTwitter for iPhonea,False,RT IAFsite When one door closes  another opens  The
IAF is planning to improve its equipment  replacing old aircraft and systems  whi
le,iBdGilmour,887623959688171520,Israel,MAGA  realDonaldTrump,169,10089,,False,en
,,,,,,',
 u'20180606021851,1004185820125908992,a hrefhttptwittercomdownloadiphone relnofol
lowTwitter for iPhonea,False,RT KremlinTrolls BREAKING Facebook has given at leas
t four Chinese companies access to user data One has been flagged by US intellige
,TheDubberman,574784811,Boston  MA,Believe nothing you hear and only half of what
you see,345,46476,,False,en,,,,,,',
 u'20180606021851,1004185821719613440,a hrefhttptwittercom relnofollowTwitter Web
Clienta,True,So I already didnt graduate with honors they didnt say my name at th
e graduation dinner  I was one of VERY FEW at the dinner whose name didnt get rea
d so after my Java professor gave us a lecture about how grades dont matter I SUP
ER dont care about errors in the HW,crescentlesl,199831004,,I post whatwouldisay
botgenerated statuses here instead of on Facebook Also subtweets Fewer subtweets
now that actual people read my twitter,20,1744,,False,en,,,,,,']
```

In [5]:
```
# Count number of records loaded to pyspark RDD

ITData.count()
```

Out[5]:
```
104783
```

In [6]:
```
# By default, data is partitioned based on the data size

# Check the number of partitions created

ITData.getNumPartitions()
```

Out[6]:
```
6
```

In [7]:
```
# Twitter data was collected and batched in files with each file having a file hea

# Extract the first file header from the dataset and display

# This will be used later to remove all headers from the dataset

header = ITData.first()
header
```

Out[7]:
```
u'timetext,tweetid,tweetsource,tweettruncated,tweettext,tweetuserscreenname,tweet
userid,tweetuserlocation,tweetuserdescription,tweetuserfollowerscount,tweetuserst
atusescount,tweetusertimezone,tweetusergioenabled,tweetuserlang,tweetcoordinatesc
oordinates,tweetplacecountry,tweetplacecountrycode,tweetplacefullname,tweetplacen
ame,tweetplacetype'
```

In [8]:
```
# Filter all of the headers from the data set

# Count the number of records remaining in the data set

# If 10 files were read from Hadoop, this count should be 10 less

ITData_NoHeader = ITData.filter(lambda row : row != header)
ITData_NoHeader.count()
```

Out[8]:
```
104777
```

In [9]:
```
# We now have an RDD with not header information

# In preparation for creating a dataframe from the RDD, create a schema based on t

schema = StructType([
    StructField('timetext', StringType(), nullable=True),
    StructField('tweet_id', StringType(), nullable=True),
    StructField('tweet_source', StringType(), nullable=True),
    StructField('tweet_truncated', StringType(), nullable=True),
    StructField('tweet_text', StringType(), nullable=True),
    StructField('tweet_user_screen_name', StringType(), nullable=True),
    StructField('tweet_user_id', StringType(), nullable=True),
    StructField('tweet_user_location', StringType(), nullable=True),
    StructField('tweet_user_description', StringType(), nullable=True),
    StructField('tweet_user_followers_count', StringType(), nullable=True),
    StructField('tweet_user_statuses_count', StringType(), nullable=True),
    StructField('tweet_user_time_zone', StringType(), nullable=True),
    StructField('tweet_user_geo_enabled', StringType(), nullable=True),
    StructField('tweet_user_lang', StringType(), nullable=True),
    StructField('tweet_coordinates_coordinates', StringType(), nullable=True),
    StructField('tweet_place_country', StringType(), nullable=True),
    StructField('tweet_place_country_code', StringType(), nullable=True),
    StructField('tweet_place_full_name', StringType(), nullable=True),
    StructField('tweet_place_name', StringType(), nullable=True),
    StructField('tweet_place_type', StringType(), nullable=True)
])

# Create a dataframe from the RDD with schema

ITData_df = sqlContext.createDataFrame(ITData_NoHeader.map(lambda s: s.split(","))

ITData_df.printSchema()
```

```
root
 |-- timetext: string (nullable = true)
 |-- tweet_id: string (nullable = true)
 |-- tweet_source: string (nullable = true)
 |-- tweet_truncated: string (nullable = true)
 |-- tweet_text: string (nullable = true)
 |-- tweet_user_screen_name: string (nullable = true)
 |-- tweet_user_id: string (nullable = true)
 |-- tweet_user_location: string (nullable = true)
 |-- tweet_user_description: string (nullable = true)
 |-- tweet_user_followers_count: string (nullable = true)
 |-- tweet_user_statuses_count: string (nullable = true)
 |-- tweet_user_time_zone: string (nullable = true)
 |-- tweet_user_geo_enabled: string (nullable = true)
 |-- tweet_user_lang: string (nullable = true)
 |-- tweet_coordinates_coordinates: string (nullable = true)
 |-- tweet_place_country: string (nullable = true)
 |-- tweet_place_country_code: string (nullable = true)
 |-- tweet_place_full_name: string (nullable = true)
 |-- tweet_place_name: string (nullable = true)
 |-- tweet_place_type: string (nullable = true)
```

In [10]:
```
# First convert dataframe to rdd

# Use map lambda to select the tweet_text column and filter out all empty records

tweet = ITData_df.rdd.map(lambda x: x['tweet_text']).filter(lambda x: x is not Non
```

In [11]:
```
# Retrieve stop words. Note we may need to add to the stop words list based on top

StopWords = stopwords.words("english")
```

In [12]:
```
# Further clean tweets, split them out into individual words, and number them by a

tokens = tweet.map(lambda document: document.strip().lower()) \
              .map(lambda document: re.split(" ", document)) \
              .map(lambda word: [x for x in word if x.isalpha()]) \
              .map(lambda word: [x for x in word if len(x) > 3]) \
              .map(lambda word: [x for x in word if x not in StopWords]) \
              .zipWithIndex()
```

In [13]:
```
# tokens is an RDD, display the first 5 records

tokens.take(5)
```

Out[13]:
```
[([u'ageronews',
   u'data',
   u'science',
   u'teams',
   u'presentation',
   u'sparkaisummit',
   u'today'],
  0),
 ([u'iafsite',
   u'door',
   u'closes',
   u'another',
   u'opens',
   u'planning',
   u'improve',
   u'equipment',
   u'replacing',
   u'aircraft',
   u'systems'],
  1),
 ([u'kremlintrolls',
   u'breaking',
   u'facebook',
   u'given',
   u'least',
   u'four',
   u'chinese',
   u'companies',
   u'access',
   u'user',
   u'data',
   u'flagged',
   u'intellige'],
  2),
 ([u'already',
   u'didnt',
   u'graduate',
   u'honors',
   u'didnt',
   u'name',
   u'graduation',
   u'dinner',
   u'dinner',
   u'whose',
   u'name',
   u'didnt',
   u'read',
   u'java',
   u'professor',
   u'gave',
   u'lecture',
   u'grades',
   u'dont',
   u'matter',
   u'super',
   u'dont',
   u'care',
   u'errors'],
  3),
 ([u'breaking',
```

In [14]:
```
# Create a new dataframe from the above RDD, adding column names

tweet_df = sqlContext.createDataFrame(tokens, ["tweet_words", "index"])
```

In [15]:
```
# Display the first 5 records of the dataframe

tweet_df.show(5)
```

```
+--------------------+-----+
|         tweet_words|index|
+--------------------+-----+
|[ageronews, data,...|    0|
|[iafsite, door, c...|    1|
|[kremlintrolls, b...|    2|
|[already, didnt, ...|    3|
|[breaking, facebo...|    4|
+--------------------+-----+
only showing top 5 rows
```

In [16]:
```
# Prepare for Topic Modeling

print(time.strftime('%m%d%Y %H:%M:%S'))
cv = CountVectorizer(inputCol="tweet_words", outputCol="raw_features", vocabSize=5
cvmodel = cv.fit(tweet_df)
print(time.strftime('%m%d%Y %H:%M:%S'))
```

```
06272018 21:19:35
06272018 21:20:32
```

In [17]:
```
print(time.strftime('%m%d%Y %H:%M:%S'))
result_cv = cvmodel.transform(tweet_df)
print(time.strftime('%m%d%Y %H:%M:%S'))
```

```
06272018 21:20:32
06272018 21:20:43
```

In [18]:
```
result_cv.show(1)
```

```
+--------------------+-----+--------------------+
|         tweet_words|index|        raw_features|
+--------------------+-----+--------------------+
|[ageronews, data,...|    0|(5000,[0,19,44,99...|
+--------------------+-----+--------------------+
only showing top 1 row
```

In [19]:
```
rs = result_cv.rdd.map(lambda (x, y, z): (x, y, oldVectors.fromML(z)))
```

In [20]:
```
rs_df = rs.toDF(['tweet_words', 'index', 'raw_features'])
```

In [21]:
```
rs.take(1)
```

Out[21]:
```
[([u'ageronews',
   u'data',
   u'science',
   u'teams',
   u'presentation',
   u'sparkaisummit',
   u'today'],
  0,
  SparseVector(5000, {0: 1.0, 19: 1.0, 44: 1.0, 993: 1.0, 2006: 1.0, 4829: 1.0}))
]
```

In [22]:
```
rs_df.show(1)
```

```
+--------------------+-----+--------------------+
|         tweet_words|index|        raw_features|
+--------------------+-----+--------------------+
|[ageronews, data,...|    0|(5000,[0,19,44,99...|
+--------------------+-----+--------------------+
only showing top 1 row
```

In [23]:
```
print(time.strftime('%m%d%Y %H:%M:%S'))
idf = IDF(inputCol="raw_features", outputCol="features")
idfModel = idf.fit(result_cv)
result_tfidf = idfModel.transform(result_cv)
print(time.strftime('%m%d%Y %H:%M:%S'))
```

```
06272018 21:21:11
06272018 21:21:51
```

In [24]:
```
# Run the LDA Topic Modeler

# Note the time before and after is printed in order to find out how much time it

print(time.strftime('%m%d%Y %H:%M:%S'))
num_topics = 10
max_iterations = 20
lda_model = LDA.train(rs_df['index', 'raw_features'].rdd.map(list), k=num_topics,
print(time.strftime('%m%d%Y %H:%M:%S'))
```

```
06272018 21:21:51
06272018 21:30:10
```

In [25]:
```
vocabArray = cvmodel.vocabulary
```

In [26]:
```
# Set the top number of topics to write to spark

wordNumbers = 20
topicIndices = sc.parallelize(lda_model.describeTopics(maxTermsPerTopic = wordNumb
```

In [27]:
```
def topic_render(topic):
    terms = topic[0]
    result = []
    for i in range(wordNumbers):
        term = vocabArray[terms[i]]
        result.append(term)
    return result
```

In [28]:
```
print(time.strftime('%m%d%Y %H:%M:%S'))
topics_final = topicIndices.map(lambda topic:
                                topic_render(topic)).collect()
print(time.strftime('%m%d%Y %H:%M:%S'))
```

```
06272018 21:30:17
06272018 21:30:24
```

In [29]:
```
# Display topics

for topic in range(len(topics_final)):
    print("Topic" + str(topic) + ":")
    for term in topics_final[topic]:
        print(term)
    print('\n')
```

```
Topic0:
data
technology
facebook
sure
computer
manage
saved
ithegiovanni
chinese
erased
cloud
robotics
access
systems
developer
blockchain
hybrid
companies
programming
```

In [30]:
```
# The above above relates topics to the terms I searched in Twitter

# For sentiment analysis, I would like to rate the actual search terms.

# For this I will build a python array with those search terms

search_terms = ["machine_learning", "computer_programmer", "database_engineer", "r
                "data_scientist", "systems_engineer", "data_analyst", "data_archit
                "web_programmer", "automation_engineer", "data_processing", "appli
                "software_engineer", "software_developer", "information_architect"
                "business_intelligence", "enterprise_architect", "solution_archite
                "information_technology", "data", "java", "iot", "computer", "syst
                "etl", "devops", "cloud", "developer", "programmer", "ai"]

search_terms
```

Out[30]:
```
['machine_learning',
 'computer_programmer',
 'database_engineer',
 'network_engineer',
 'data_scientist',
 'systems_engineer',
 'data_analyst',
 'data_architect',
 'etl_architect',
 'web_programmer',
 'automation_engineer',
 'data_processing',
 'application_engineer',
 'software_engineer',
 'software_developer',
 'information_architect',
 'security_analyst',
 'business_intelligence',
 'enterprise_architect',
 'solution_architect',
 'data_warehouse',
 'information_technology',
 'data',
 'java',
 'iot',
 'computer',
 'systems',
 'technology',
 'etl',
 'devops',
 'cloud',
 'developer',
 'programmer',
 'ai']
```

In [31]:
```
# Python function to search for topics within a tweet

# Function will return the topic and the related tweet or NA is no topic found and

def SearchTopics(topics, tweet_text):
    for term in topics:
        result = tweet_text.find(term)
        if result > -1:
            return term, tweet_text
    return 'NA', tweet_text
```

In [32]:
```
# While removing stopwords helps obtain valid topics it will not help with sentime

# With topics in hand, topics_final, we will use tweets where stop words have not

tweet.take(5)
```

Out[32]:
```
[u'AgeroNews data science teams presentation at SparkAISummit today httpstco7F2IA
UvN4o',
 u'RT IAFsite When one door closes  another opens  The IAF is planning to improve
its equipment  replacing old aircraft and systems  while',
 u'RT KremlinTrolls BREAKING Facebook has given at least four Chinese companies a
ccess to user data One has been flagged by US intellige',
 u'So I already didnt graduate with honors they didnt say my name at the graduati
on dinner  I was one of VERY FEW at the dinner whose name didnt get read so after
my Java professor gave us a lecture about how grades dont matter I SUPER dont car
e about errors in the HW',
 u'RT Breaking911 BREAKING Facebook has given at least 4 Chinese companies access
to user data One has been flagged by US intelligence a']
```

In [33]:
```
# Search each tweet for topics returning only tweets that match

# SearchTopics will return both the topic and the related tweet

# Sentiment will be done on these tweets

topic_tweet = tweet.map(lambda x: SearchTopics(search_terms, x)).filter(lambda x:
```

In [34]:
```
# Display 5 topic tweet combinations

topic_tweet.take(10)
```

Out[34]:
```
[('data',
  u'AgeroNews data science teams presentation at SparkAISummit today httpstco7F2I
AUvN4o'),
 ('systems',
  u'RT IAFsite When one door closes  another opens  The IAF is planning to improv
e its equipment  replacing old aircraft and systems  while'),
 ('data',
  u'RT KremlinTrolls BREAKING Facebook has given at least four Chinese companies
access to user data One has been flagged by US intellige'),
 ('data',
  u'RT Breaking911 BREAKING Facebook has given at least 4 Chinese companies acces
s to user data One has been flagged by US intelligence a'),
 ('data',
  u'RT KremlinTrolls BREAKING Facebook has given at least four Chinese companies
access to user data One has been flagged by US intellige'),
 ('data',
  u'RT PoliticalShort Facebook confirmed that 4 Chinese device makers were among
those that had broad access to customer data under a program'),
 ('iot',
  u'bruabreus thehorrorpics youlovemypugs pussyriotx AI QUE HORROR KKKKKKKKKKKKK
K'),
 ('data',
  u'RT Reuters Facebook confirms data sharing with Chinese companies httpstco7tnl
S6aTPe'),
 ('systems',
  u'Military bases have dangerous toxins children are especially stressed with fr
agile immune systemsThey have no right to subject these precious gifts from the C
reator to what equals to child torture'),
 ('iot',
  u'KeineMaster LITTLEM1B KEEMSTAR TTfue Lol you really think Im mad over an idio
t sitting behind his computer that started a tournament that streamers join and h
as his manager team do all the work  lmfao What a joke Keemstar is an idiot deal
with it Stop defending him for nothing')]
```

In [35]:
```
# Setup sentiment analysis

import nltk
from nltk.sentiment.vader import SentimentIntensityAnalyzer
nltk.download('vader_lexicon')
```

```
/home/vagrant/anaconda2/lib/python2.7/site-packages/nltk/twitter/__init__.py:20:
UserWarning: The twython library has not been installed. Some functionality from
the twitter package will not be available.
  warnings.warn("The twython library has not been installed. "

[nltk_data] Downloading package vader_lexicon to
[nltk_data]     /home/vagrant/nltk_data...
[nltk_data]   Package vader_lexicon is already up-to-date!
```

Out[35]:
```
True
```

In [36]:
```
# Python function to print the sentiment scores

# This function will have topic and related tweet as in put

# This function will perform sentiment analysis and output topic, tweet, and senti

# Also note this function will only return the compound portion of the sentiment

# Revert sigpipe to default behavior

def print_sentiment_scores(topic, sentence):
    snt = SentimentIntensityAnalyzer().polarity_scores(sentence)
    print("{:-<40} {}".format(sentence, str(snt)))
    print(str(snt))
    return(topic, sentence, str(snt.get('compound')))
```

In [37]:
```
# Retrieve sentiment for each topic, tweet

topic_tweet_sentiment = topic_tweet.map(lambda x: print_sentiment_scores(x[0], x[1
```

In [38]:
```
# Display sentiment

topic_tweet_sentiment.take(10)
```

Out[38]:
```
[('data',
  u'AgeroNews data science teams presentation at SparkAISummit today httpstco7F2I
AUvN4o',
  '0.0'),
 ('systems',
  u'RT IAFsite When one door closes  another opens  The IAF is planning to improv
e its equipment  replacing old aircraft and systems  while',
  '0.4404'),
 ('data',
  u'RT KremlinTrolls BREAKING Facebook has given at least four Chinese companies
access to user data One has been flagged by US intellige',
  '0.0'),
 ('data',
  u'RT Breaking911 BREAKING Facebook has given at least 4 Chinese companies acces
s to user data One has been flagged by US intelligence a',
  '0.4767'),
 ('data',
  u'RT KremlinTrolls BREAKING Facebook has given at least four Chinese companies
access to user data One has been flagged by US intellige',
  '0.0'),
 ('data',
  u'RT PoliticalShort Facebook confirmed that 4 Chinese device makers were among
those that had broad access to customer data under a program',
  '0.0'),
 ('iot',
  u'bruabreus thehorrorpics youlovemypugs pussyriotx AI QUE HORROR KKKKKKKKKKKKK
K',
  '-0.6633'),
 ('data',
  u'RT Reuters Facebook confirms data sharing with Chinese companies httpstco7tnl
S6aTPe',
  '0.4215'),
 ('systems',
  u'Military bases have dangerous toxins children are especially stressed with fr
agile immune systemsThey have no right to subject these precious gifts from the C
reator to what equals to child torture',
  '-0.7178'),
 ('iot',
  u'KeineMaster LITTLEM1B KEEMSTAR TTfue Lol you really think Im mad over an idio
t sitting behind his computer that started a tournament that streamers join and h
as his manager team do all the work  lmfao What a joke Keemstar is an idiot deal
with it Stop defending him for nothing',
  '-0.3744')]
```

In [39]:
```
# Assign the topic and sentiment only

topic_tweet_sentiment_pair = topic_tweet_sentiment.map(lambda x: (x[0], x[2]))
```

In [40]:
```
# Display topic, sentiment combination

topic_tweet_sentiment_pair.take(10)
```

Out[40]:
```
[('data', '0.0'),
 ('systems', '0.4404'),
 ('data', '0.0'),
 ('data', '0.4767'),
 ('data', '0.0'),
 ('data', '0.0'),
 ('iot', '-0.6633'),
 ('data', '0.4215'),
 ('systems', '-0.7178'),
 ('iot', '-0.3744')]
```

In [41]:
```
# Convert to dataframe naming columns

topic_tweet_sentiment_pair_df = topic_tweet_sentiment_pair.toDF(['topic', 'sentiment
```

In [42]:
```
# Display dataframe

topic_tweet_sentiment_pair_df.show(5)
```

```
+-------+---------+
|  topic|sentiment|
+-------+---------+
|   data|      0.0|
|systems|   0.4404|
|   data|      0.0|
|   data|   0.4767|
|   data|      0.0|
+-------+---------+
only showing top 5 rows
```

In [43]:
```
# Count sentiment records

topic_tweet_sentiment_pair_df.count()
```

Out[43]:
```
61624
```

In [44]:
```
# Create panda dataframe based on topic, sentiment dataframe

# This dataframe will enable us to plot highs, lows, and means

pdf1 = topic_tweet_sentiment_pair_df.toPandas()
```

In [45]:
```
# Check new dataframe types

pdf1.dtypes
```

Out[45]:
```
topic        object
sentiment    object
dtype: object
```

In [46]:
```
# Sentiment is currently of type object, needs to be float

# Convert sentiment datatype to float

pdf1['sentiment'] = pdf1.sentiment.astype(float)

# Check datatypes

pdf1.dtypes

# list new panda dataframe

pdf1
```

Out[46]:

	topic	sentiment
0	data	0.0000
1	systems	0.4404
2	data	0.0000
3	data	0.4767
4	data	0.0000
5	data	0.0000
6	iot	-0.6633
7	data	0.4215
8	systems	-0.7178
9	iot	-0.3744
10	ai	0.0000
11	ai	0.0000
12	computer	0.6045
13	ai	0.0000
14	ai	-0.5267
15	data	0.0000
16	data	0.0000
17	data	0.0000
18	ai	-0.5267
19	data	0.0000
20	computer	-0.6705
21	data	0.6486
22	ai	0.0000
23	ai	0.0000
24	ai	0.0000
25	developer	0.0772
26	data	0.6249
27	machine_learning	0.4588
28	systems	0.0000

In [47]:
```
# Describe data

pdf1.describe()
```

Out[47]:

	sentiment
count	61624.000000
mean	0.158525
std	0.385764
min	-0.983200
25%	0.000000
50%	0.000000
75%	0.476700
max	0.983000

In [59]:
```
pdf1.groupby('topic').groups.keys()
```

Out[59]:
```
[u'data_architect',
 u'ai',
 u'security_analyst',
 u'data_processing',
 u'machine_learning',
 u'network_engineer',
 u'computer',
 u'automation_engineer',
 u'information_technology',
 u'programmer',
 u'technology',
 u'cloud',
 u'developer',
 u'data_warehouse',
 u'java',
 u'business_intelligence',
 u'devops',
 u'software_developer',
 u'systems',
 u'solution_architect',
 u'information_architect',
 u'iot',
 u'data_analyst',
 u'software_engineer',
 u'data',
 u'computer_programmer',
 u'systems_engineer',
 u'etl',
 u'data_scientist']
```

In [61]:
```
pdf1_group_counts = pdf1.groupby(['topic'])[['sentiment']].count()
pdf1_group_counts
```

Out[61]:

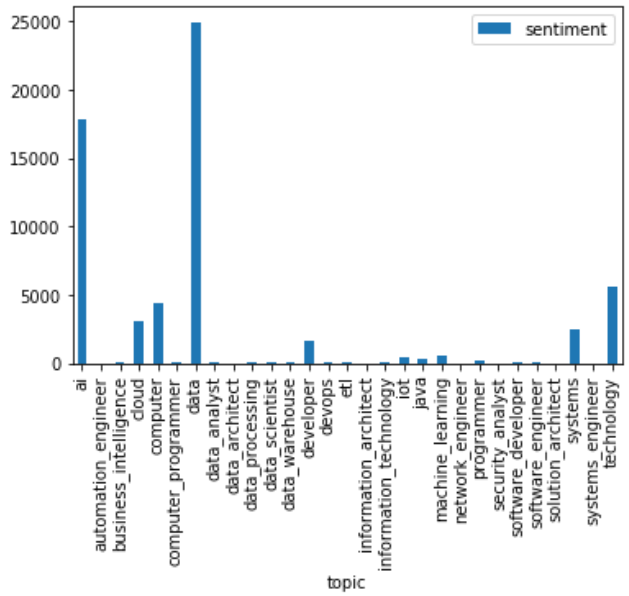
topic	sentiment
ai	17865
automation_engineer	2
business_intelligence	9
cloud	3049
computer	4384
computer_programmer	10
data	24880
data_analyst	26
data_architect	1
data_processing	12
data_scientist	48
data_warehouse	9
developer	1569
devops	123
etl	50
information_architect	1
information_technology	23
iot	486
java	360
machine_learning	499
network_engineer	4
programmer	152
security_analyst	3
software_developer	26
software_engineer	53
solution_architect	1
systems	2427
systems_engineer	6
technology	5546

In [62]:
```
pdf1_mean = pdf1.groupby('topic', as_index=False).agg({"sentiment": "mean"})
pdf1_mean
```

Out[62]:

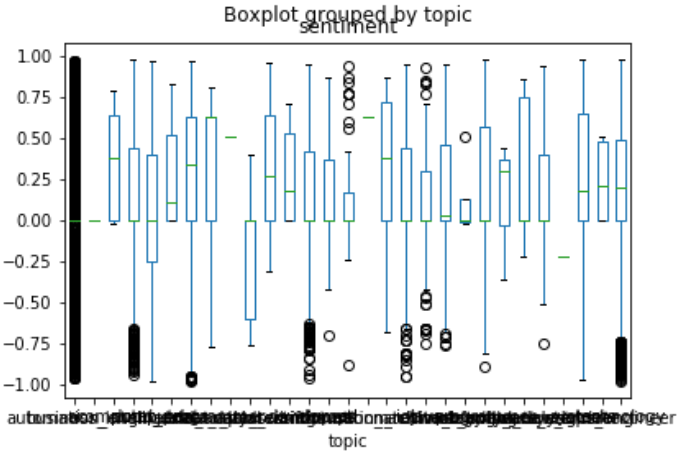
	topic	sentiment
0	ai	0.034751
1	automation_engineer	0.000000
2	business_intelligence	0.351122
3	cloud	0.168504
4	computer	0.045772
5	computer_programmer	0.255450
6	data	0.254000
7	data_analyst	0.363973
8	data_architect	0.510600
9	data_processing	-0.159783
10	data_scientist	0.304515
11	data_warehouse	0.258478
12	developer	0.187582
13	devops	0.171636
14	etl	0.142446
15	information_architect	0.624900
16	information_technology	0.318122
17	iot	0.151902
18	java	0.122913
19	machine_learning	0.188162
20	network_engineer	0.121200
21	programmer	0.206038
22	security_analyst	0.125067
23	software_developer	0.261092
24	software_engineer	0.194768
25	solution_architect	-0.226300
26	systems	0.192733
27	systems_engineer	0.237467
28	technology	0.184405

In [63]:
```
# Barchart

pdf1_plot = pdf1_group_counts_plot(kind='bar')
```



In [64]:
```
# Boxplot sentiments by topic

pdf1.boxplot(by='topic', column=['sentiment'], grid=False)
```

Out[64]: <matplotlib.axes._subplots.AxesSubplot at 0x7f15c8e0e690>



In [49]:
```
sentiment_terms1 = ['ai', 'data', 'technology', 'cloud']
```

In [50]:
```
pdf2 = pdf1[pdf1.topic.isin(sentiment_terms1)]
pdf2
```

Out[50]:

	topic	sentiment
0	data	0.0000
2	data	0.0000
3	data	0.4767
4	data	0.0000
5	data	0.0000
7	data	0.4215
10	ai	0.0000
11	ai	0.0000
13	ai	0.0000
14	ai	-0.5267
15	data	0.0000
16	data	0.0000
17	data	0.0000
18	ai	-0.5267
19	data	0.0000
21	data	0.6486
22	ai	0.0000
23	ai	0.0000
24	ai	0.0000
26	data	0.6249
29	cloud	0.0000
31	cloud	-0.5267
33	data	0.6249
34	data	0.4767
35	ai	-0.3818
36	data	-0.8834
37	ai	0.0000
38	data	0.6249
39	data	0.4588
42	ai	0.0000
...	...	...
61579	ai	0.0000
61581	ai	0.0000
61583	data	0.0000
61584	data	0.6249
61586	ai	0.0000

In [51]:
```
pdf2.groupby('topic').groups.keys()
```

Out[51]:
```
[u'ai', u'data', u'cloud']
```

In [53]:
```
pdf2_group_counts = pdf2.groupby(['topic'])[['sentiment']].count()
```

Out[53]:

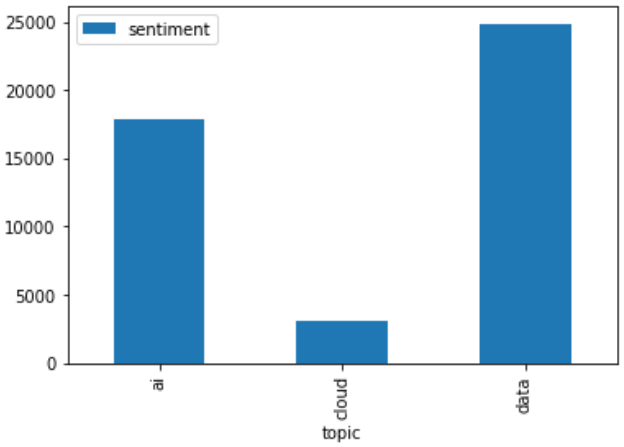
topic	sentiment
ai	17865
cloud	3049
data	24880

In [56]:
```
pdf2_mean = pdf2.groupby('topic', as_index=False).agg({"sentiment": "mean"})
```

Out[56]:

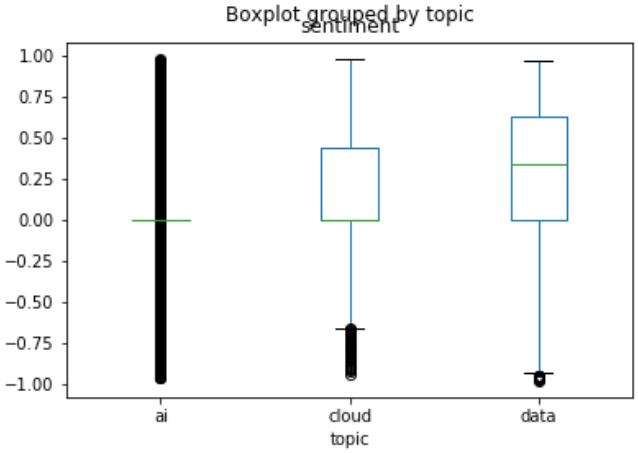
	topic	sentiment
0	ai	0.034751
1	cloud	0.168504
2	data	0.254000

In [57]:
```
# Barchart

pdf2_plot = pdf2_group_counts.plot(kind='bar')
```



In [58]:
```
# Boxplot sentiments by topic

pdf2.boxplot(by='topic', column=['sentiment'], grid=False)
```

Out[58]: <matplotlib.axes._subplots.AxesSubplot at 0x7f15ca428c10>



In [ ]: