Big Data Project Report

Installation Manual

- **Google dataproc**

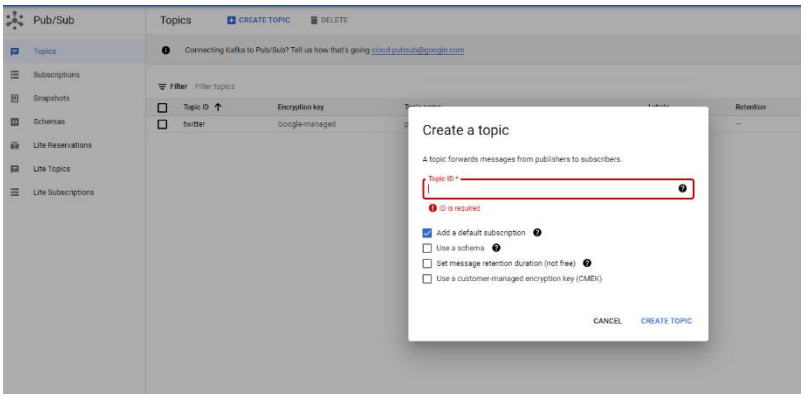
  สร้าง VM โดยเลือกใช้ Components ของ Jupyter และ ZooKeeper

  เพื่อให้เข้าถึงทุก Service ต้อง check Enable API

  (CentOS 8, Hadoop 3.2, Spark 3.1)

- **Google Pub/Sub**

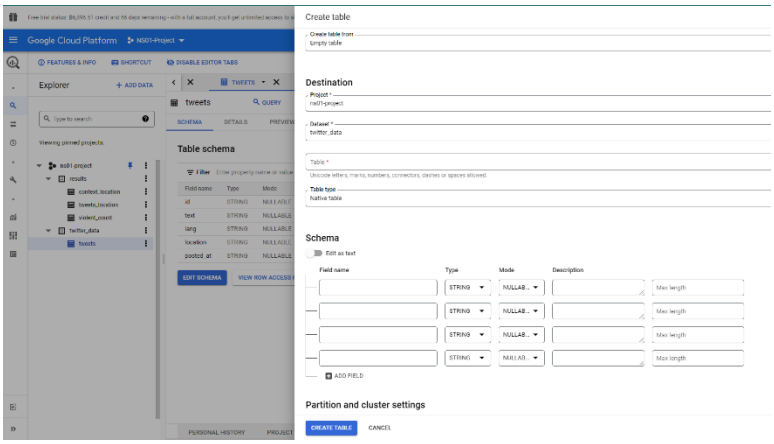
  ระบุชื่อ Topic ไว้สำหรับการ Subscription



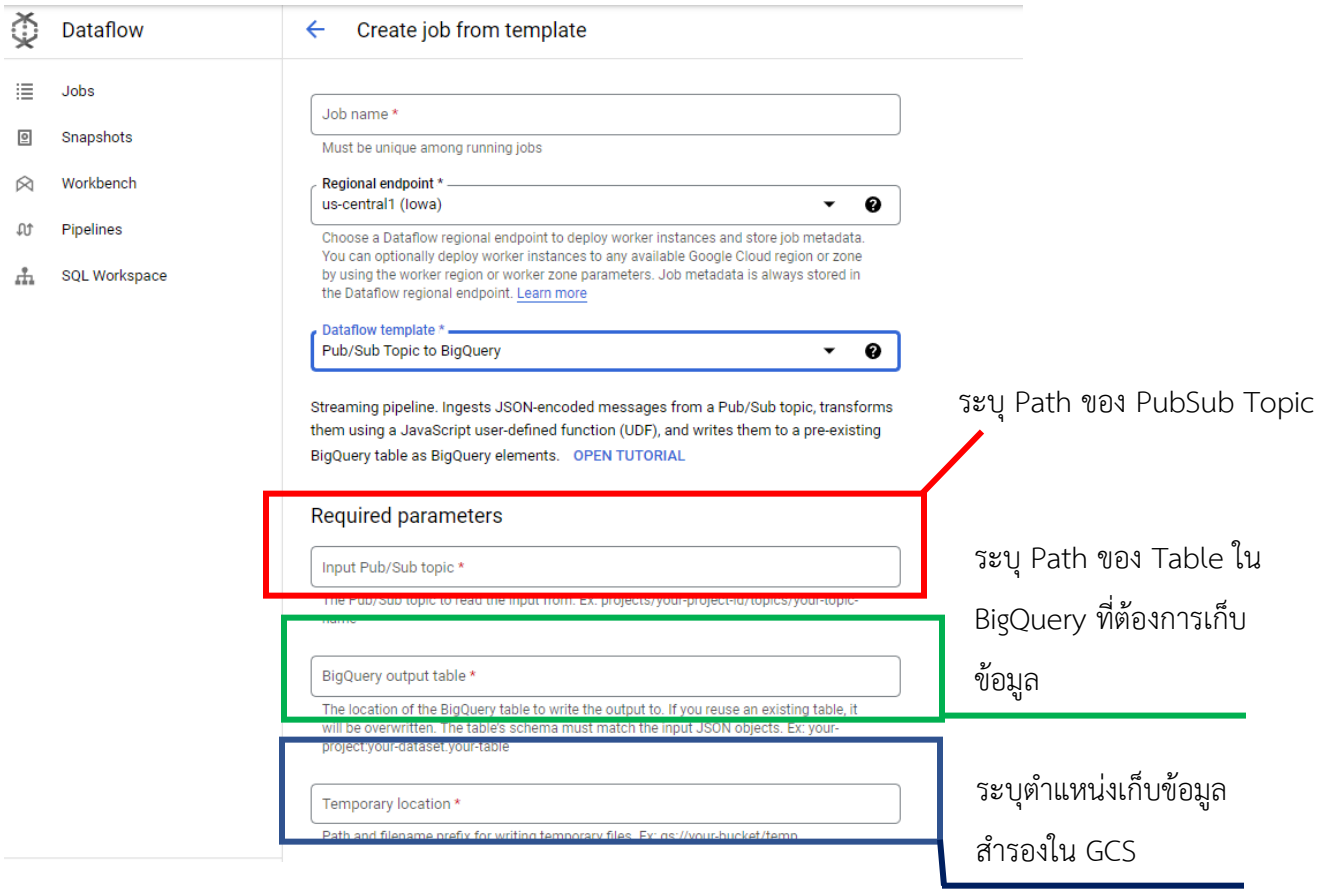
- **Google BigQuery**

  สร้าง Dataset และตาราง พร้อมกำหนด Schema ของตารางจาก Field ที่ต้องการใช้จาก

  ข้อมูล JSON



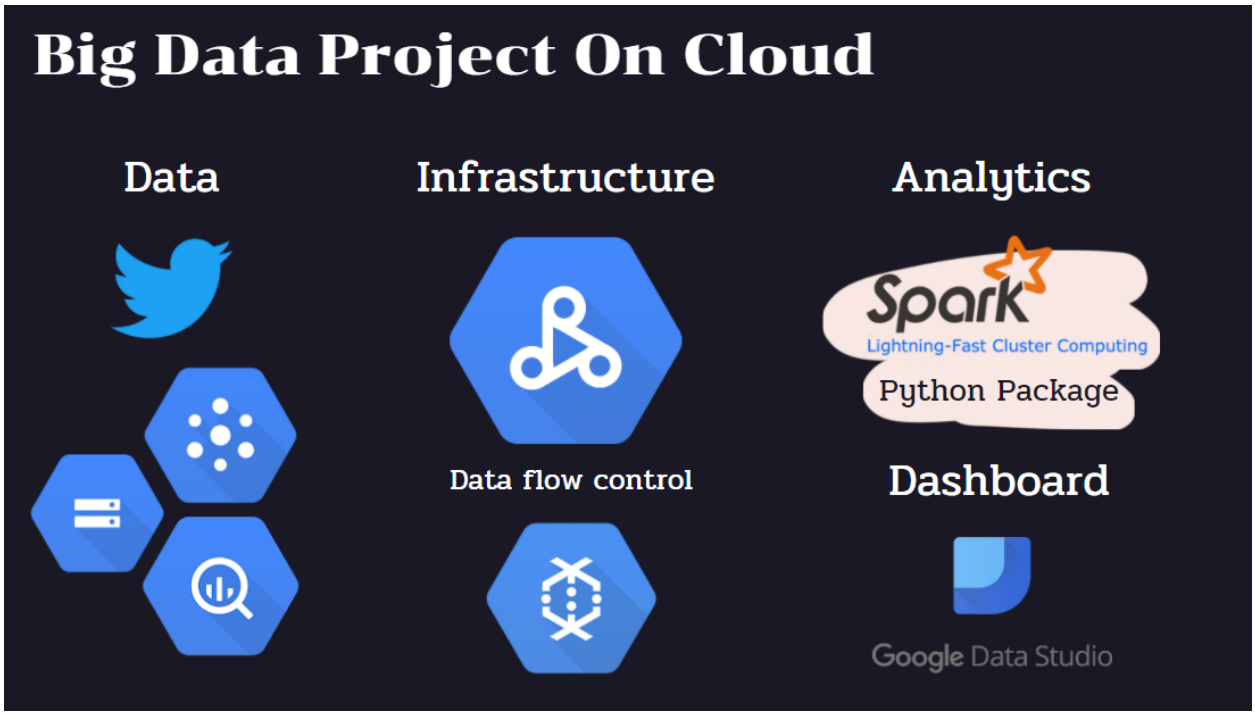
- **Data flow**

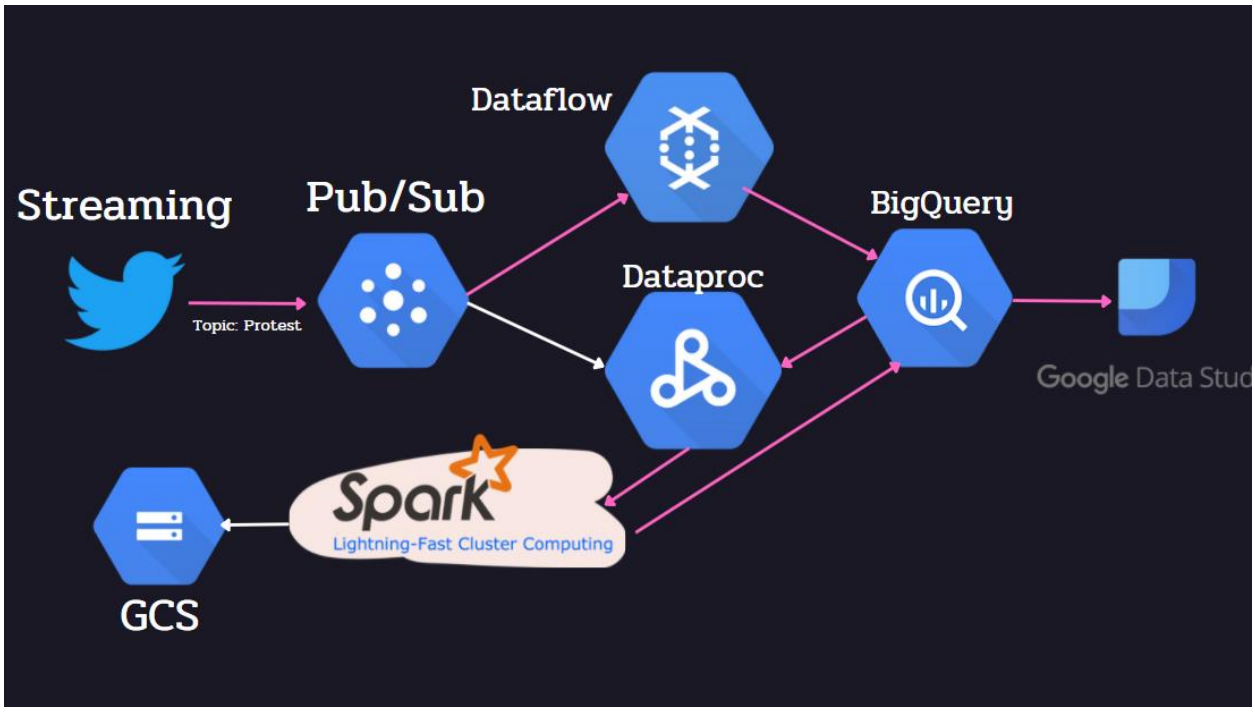


Project Objective

เป้าหมายของโปรเจคนี้คือการศึกษาเครื่องมือที่เกี่ยวข้องกับการจัดการข้อมูล Big data ซึ่งจะเลือกใช้ข้อมูล JSON Streaming ที่ถูกส่งเข้ามาเรื่อย ๆ จาก Twitter API ในบริบทที่มีความเกี่ยวข้องกับการประท้วง (Protest) ที่ถูกเอ่ยถึงใน Tweet ทั่วโลก และใช้เครื่องมือต่างๆในการวิเคราะห์ตำแหน่งที่ถูกเอ่ยถึงรวมถึงวิเคราะห์ความรุนแรงโดยอ้างอิงจากความรุนแรงในการแสดงออกผ่านข้อความ Tweet (Language Processing) ซึ่งข้อมูลเหล่านี้จะถูกเก็บและประมวลผลอยู่บน Cloud Platform ทั้งหมดและแสดงผลหลังการประมวลผลบน Dashboard

Diagram & Tools





Code & Detail Explanation

PART 1: Streaming & Publishing

Package ที่ใช้เชื่อมต่อกับ Pub/sub

```
In [3]: from google.cloud import pubsub_v1
```

```
In [4]: # config to pubsub
        publisher = pubsub_v1.PublisherClient()
        topic_path = publisher.topic_path("ns01-project", "twitter")
        print(topic_path)

        projects/ns01-project/topics/twitter
```

สร้าง Instance Client และ ระบุ Topic เพื่อเชื่อมต่อกับ pub/sub

```
In [5]: "projects/ns01-project/topics/twitter" == topic_path
Out[5]: True
```

```
In [6]: # Authenticate to twitter api config
        auth = OAuthHandler(CONSUMER_KEY, CONSUMER_SECRET)
        auth.set_access_token(ACCESS_TOKEN, ACCESS_TOKEN_SECRET)
```

Twitter API Authentication

```
In [7]: # Tag
        keyword = ["protest"]
```

ระบุ Keyword ที่ต้องการค้นหา

```
In [8]: def write_to_pubsub(data):
            try:
                publisher.publish(topic_path, data=json.dumps({
                    "text": data["text"],
                    "lang": data["lang"],
                    "location": data["location"],
                    "id": data["id"],
                    "posted_at": datetime.datetime.fromtimestamp(data["created_at"]).strftime('%Y-%m-%d %H:%M:%S')
                }).encode("utf-8"), tweet_id=str(data["id"]).encode("utf-8"))
            except Exception as e:
                raise
```

ฟังก์ชั่นสำหรับเขียนข้อมูลลง pub/sub ซึ่งระบุ topic path และ data, ในที่นี้รับข้อมูลเป็น JSON และต้อง encode เป็น byteString

```
class StdOutListener(StreamListener):
    """ A listener handles tweets that are received from the stream.
    This is a basic listener that just pushes tweets to pubsub
    """

    def __init__(self):
        super(StdOutListener, self).__init__()
        self._counter = 0

    def on_status(self, data):
        write_to_pubsub(reformat_tweet(data._json))
        print(reformat_tweet(data._json))
        self._counter += 1
        return True

    def on_error(self, status):
        if status == 420:
            print("rate limit active")
            return False
```

Class สำหรับรับข้อมูล Streaming ของ Twitter

Flow: นำข้อมูล json ที่ได้ไป preprocess → reformat_tweet จากนั้นส่งไปเขียนลง Pub/Sub → Write_to_pubsub

```
l = StdOutListener()
stream = tweepy.Stream(auth, l, tweet_mode='extended')
stream.filter(track=keyword)
```

เมื่อผ่าน Part 1 ข้อมูล Streaming จะถูก Dataflow จัดการโดยการดึงข้อมูลจาก Pub/Sub เขียนลงบน BigQuery ทันที, ณ ขั้นตอนนี้ข้อมูลจะอยู่บน BigQuery แล้ว

PART 2: Violent Detection (SparkML)

In [4]:
```
#3 - Setup SparkSession (SparkSQL)
spark = (SparkSession
        .builder
        .appName("DataFrameHandOn")
        .master("local[*]")
        .getOrCreate())
print(spark)
```

Initialize Spark

```
<pyspark.sql.session.SparkSession object at 0x7f5337f73a00>
```

In [5]:
```
df = spark.read.csv("gs://twitter_testtt/labeled_data.csv", header=True, inferSchema=True)
df.cache()
print("finish caching data")
```

```
[Stage 1:>
finish caching data
```

อ่านข้อมูล data ที่จะนำมา Train เก็บในรูปของ Spark data frame on top Spark RDD

In [6]:
```
df.show(5)
```

```
[Stage 2:>                                                  (0 + 1) / 1]
+---+-----+-----------+-----------------+-------+-----+--------------------+
|_c0|count|hate_speech|offensive_language|neither|class|               tweet|
+---+-----+-----------+-----------------+-------+-----+--------------------+
|  0|    3|          0|                0|      3|    2|!!! RT @mayasolov...|
|  1|    3|          0|                3|      0|    1|!!!!! RT @mleew17...|
|  2|    3|          0|                3|      0|    1|!!!!!!! RT @UrKin...|
|  3|    3|          0|                2|      1|    1|!!!!!!!!! RT @C_G...|
|  4|    6|          0|                6|      0|    1|!!!!!!!!!!!!! RT ...|
+---+-----+-----------+-----------------+-------+-----+--------------------+
only showing top 5 rows
```

ข้อมูลที่นำมา Train เป็น Dataset ที่เกี่ยวกับคำที่มีความรุนแรงหรือมีคำพูดที่มีการดูถูก ซึ่งน่าจะนำมาใช้กับ Protest analysis ได้เช่นเดียวกัน

In [11]:
```
from pyspark.sql.types import IntegerType
from pyspark.sql.functions import udf

def onlyTwoClass(x):
    return 1 if str(x)>str(1) else 0

my_udf = udf(onlyTwoClass, IntegerType())
```

ใช้ User define function ของ Spark เพื่อสร้างฟังก์ชั่นที่เปลี่ยน Label ของข้อมูลให้เหลือแค่ 0 และ 1 คือ ไม่มีความรุนแรง และ มีความรุนแรง ตามลำดับ

In [12]:
```
new_df = df.withColumn('class', my_udf('class'))
```

```
new_df.show(10)
```

```
[Stage 4:>                                                  (0 + 1) / 1]
```

```
+---+-----+-----------+-----------------+-------+-----+--------------------+
|_c0|count|hate_speech|offensive_language|neither|class|               tweet|
+---+-----+-----------+-----------------+-------+-----+--------------------+
|  0|    3|          0|                0|      3|    1|!!! RT @mayasolov...|
|  1|    3|          0|                3|      0|    0|!!!!! RT @mleew17...|
|  2|    3|          0|                3|      0|    0|!!!!!!! RT @UrKin...|
|  3|    3|          0|                2|      1|    0|!!!!!!!!! RT @C_G...|
|  4|    6|          0|                6|      0|    0|!!!!!!!!!!!!! RT ...|
|  5|    3|          1|                2|      0|    0|"!!!!!!!!!!!!!!!...|
|  6|    3|          0|                3|      0|    0|"!!!!!!""@__Brigh...|
|  7|    3|          0|                3|      0|    0|!!!!&#8220;@selfi...|
|  8|    3|          0|                3|      0|    0|""" &amp;amp; you mig...|
|  9|    3|          1|                2|      0|    0|""" @rhythmixx_ :...|
+---+-----+-----------+-----------------+-------+-----+--------------------+
only showing top 10 rows
```

ผลลัพธ์จะเปลี่ยน class ให้เหลือแค่ 0 และ 1 และจะ Drop column อื่นๆที่เหลือที่ไม่ได้ใช้งานแล้วออก (ใช้แค่ Class เป็น label และ tweet เป็น input)

:
```
import re

def preprocess(text_string):
    """
    Accepts a text string and replaces:
    1) urls with URLHERE
    2) lots of whitespace with one instance
    3) mentions with MENTIONHERE

    This allows us to get standardized counts of urls and mentions
    Without caring about specific people mentioned
    """
    space_pattern = '\s+'
    giant_url_regex = ('http[s]?://(?:[a-zA-Z]|[0-9]|[$-_@.&+]|'
        '[!*\(\),]|(?:%[0-9a-fA-F][0-9a-fA-F]))+')
    mention_regex = '@[\w\-]+'
    parsed_text = re.sub(space_pattern, ' ', str(text_string))
    parsed_text = re.sub(giant_url_regex, '', str(parsed_text))
    parsed_text = re.sub(mention_regex, '', str(parsed_text))
    parsed_text = re.sub("[^a-zA-Z:,]+", ' ', str(parsed_text))
    parsed_text = parsed_text.replace('RT', '')
    parsed_text = parsed_text.replace('!', '')
    parsed_text = parsed_text.replace(':', '')
    parsed_text = parsed_text.strip('\'"')
    parsed_text = parsed_text.lower()
    parsed_text = parsed_text.lstrip()

    return parsed_text
```

ฟังก์ชั่นสำหรับการเตรียมข้อมูล tweet เช่นการลบคำที่ไม่สำคัญ เช่น การ Retweet, ลิงก์ Url, Hashtag etc.

:
```
txt_process_udf = udf(preprocess, StringType())
new_df = new_df.withColumn('tweet', txt_process_udf('tweet'))
```

```
new_df.show(20)
```

```
+---+-----+--------------------+
|_c0|class|               tweet|
+---+-----+--------------------+
|  0|    1|as a woman you sh...|
|  1|    0|boy dats cold tyg...|
|  2|    0|dawg   you ever f...|
|  3|    0|she look like a t...|
|  4|    0|the shit you hear...|
|  5|    0|the shit just blo...|
|  6|    0|i can not just si...|
|  7|    0|cause i m tired o...|
|  8|    0|amp you might not...|
|  9|    0|hobbies include f...|
| 10|    0|keeks is a bitch ...|
| 11|    0|murda gang bitch ...|
| 12|    0|so hoes that smok...|
| 13|    0|bad bitches is th...|
| 14|    0|bitch get up off me |
| 15|    0|bitch nigga miss ...|
| 16|    0| bitch plz whatever |
| 17|    0|bitch who do you ...|
| 18|    0|bitches get cut o...|
| 19|    0|black bottle amp ...|
+---+-----+--------------------+
only showing top 20 rows
```

RESULT

Text Featurization

```
: from pyspark.ml.feature import Tokenizer, Word2Vec
  from pyspark.ml import Pipeline
```

```
: tokenizer = Tokenizer(inputCol="tweet", outputCol="words")
  w2v = Word2Vec(vectorSize=300, minCount=0, inputCol="words", outputCol="Features")

  #Create Pipeline
  w2v_pipeline = Pipeline(stages=[tokenizer, w2v])

  w2v_pipeline_model = w2v_pipeline.fit(train_df)
  train_df = w2v_pipeline_model.transform(train_df)
  test_df = w2v_pipeline_model.transform(test_df)
```

```
21/12/05 21:18:50 WARN com.github.fommil.netlib.BLAS
21/12/05 21:18:51 WARN com.github.fommil.netlib.BLAS
```

ทำ Featurization จาก Word2Vec (เปลี่ยน Text ให้อยู่ในรูปของ Embedding Vector) โดยทำการ Tokenize ข้อมูลก่อน แล้วเก็บใน column “words” จากนั้นจะนำ คำที่ถูก Tokenized มาทำ Embedding ใน Word2Vec โดยกำหนด Feature dimension = 300

```
: train_df.show(10)
```

```
[Stage 17:>                                                 (0 + 1) / 1]
```

Featurization Result

```
+-----+-----+--------------------+--------------------+--------------------+
| _c0|class|               tweet|               words|            Features|
+-----+-----+--------------------+--------------------+--------------------+
|    0|    1|as a woman you sh...|[as, a, woman, yo...|[2.21605230446742...|
|    1|    0|boy dats cold tyg...|[boy, dats, cold,...|[-0.0159893891707...|
|  100|    0|how bout them cow...|[how, bout, them,...|[-0.0075616843532...|
| 1000|    1|mike calls me t b...|[mike, calls, me,...|[-0.0012457987293...|
|10000|    0|he needs too we w...|[he, needs, too, ...|[-0.0080277135923...|
|10002|    0|he only favorites...|[he, only, favori...|[-0.0225900625093...|
|10003|    0|he prolly gone la...|[he, prolly, gone...|[-0.0201731702416...|
|10004|    0|he pussy whipped ...|[he, pussy, whipp...|[-0.0163165788762...|
|10005|    0|he run his mouth ...|[he, run, his, mo...|[-0.0348054950092...|
|10006|    0|  he said bitch boy |[he, said, bitch,...|[-0.0331203057139...|
+-----+-----+--------------------+--------------------+--------------------+
only showing top 10 rows
```

```
w2v.save("gs://twitter_testtt/w2v_model1")
```

Save Word2Vec model เก็บไว้ใน GCS เพื่อเรียกใช้สำหรับ Production

```
train_features = train_df.select("Features").collect()
train_lables = train_df.select("class").collect()
test_features = test_df.select("Features").collect()
test_labels = test_df.select("class").collect()


X_train = np.asarray([v[0].toArray() for v in train_features])
Y_train = np.asarray([v[0] for v in train_lables])
X_test =  np.asarray([v[0].toArray() for v in test_features])
Y_test = np.asarray([v[0] for v in test_labels])

xgbClassifier = xgb.XGBClassifier(max_depth=20, seed=18238, objective='multi:softmax',num_class = 2)
model = xgbClassifier.fit(X_train, Y_train)
pred = model.predict(X_test)

auc_score = accuracy_score(Y_test,pred)
print ("The accuracy score for XGboost model : ",auc_score)
```

Train model สำหรับ Classify violent โดยใช้โมเดลของ XGBoost ซึ่งได้ค่า Accuracy ของ dataset นี้อยู่ที่ 89.7%

```
/root/.local/lib/python3.8/site-packages/xgboost/sklearn.py:1224: UserWarning: The use of label encoder in XGB
in a future release. To remove this warning, do the following: 1) Pass option use_label_encoder=False when con
e your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)

[21:24:40] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric used with t
om 'merror' to 'mlogloss'. Explicitly set eval_metric if you'd like to restore the old behavior.
The accuracy score for XGboost model :  0.8976649746192893
```

SAVE MODEL PARAMS

```
model.save_model("model1.bst")
```

Save model parameters เก็บไว้บน GCS เพื่อใช้ทำนายผลใน Production data

```
!gsutil cp ./model.bst $MODEL_BUCKET
```

```
Copying file://./model.bst [Content-Type=application/octet-stream]...
- [1 files][  1.5 MiB/  1.5 MiB]
Operation completed over 1 objects/1.5 MiB.
```

PART 3: Query & Get final result

ในส่วนนี้จะทำการดึงข้อมูล Streaming ที่เก็บไว้ใน BigQuery โดย Pub/Sub และ DataFlow ผ่านการ Query โดยใช้ภาษา SQL และนำข้อมูลชุดนั้น (Batch) มาประมวลผล

```
In [2]: %load_ext google.cloud.bigquery
```

```
The google.cloud.bigquery extension is already loaded. To reload it, use:
  %reload_ext google.cloud.bigquery
```

```
In [5]: %%bigquery protest_df
SELECT *
FROM `ns01-project.twitter_data.tweets`
LIMIT 2000
```

```
Query complete after 0.00s: 100%|██████████| 2/2 [00:00<00:00, 852.33query/s]
Downloading: 100%|██████████| 300/300 [00:01<00:00, 299.32rows/s]
```

Magic command เพื่อเรียกใช้ Function ของ BigQuery และดึกข้อมูลมาอยู่ในรูปแบบ DataFrame ในชื่อ protest_Df

```
In [6]: protest_df
```

Out[6]:

	id	text	lang	location	posted_at
0	1468665145928867844	RT @NadiaWhittomeMP: The Downing Street party ...	en	the simulation	2021-12-08 19:33:51
1	1468665159304368128	RT @MintPressNews: "Media bears a responsibili...	en	None	2021-12-08 19:33:55
2	1468665170071281671	Wooooiiii folks in this space are saying they'...	en	London, England	2021-12-08 19:33:57
3	1468665186772869124	RT @live_Tripathi: ओमीक्रोन का खौफ: AKTU के छा...	hi	None	2021-12-08 19:34:01
4	1468665193391697940	Where and when do we get out on the streets to...	en	None	2021-12-08 19:34:03
...	...	...	...	...	...
295	1468666224292806662	@IAmCiele @thenixmin @doshinswitch @Evoltal_ @...	en	None	2021-12-08 19:38:09
296	1468666228088532994	RT @timeindawater1: 5.5.Donald Trump was not p...	en	None	2021-12-08 19:38:09
297	1468666229468667904	RT @newsbht1: London chaos: Capital gripped by...	en	None	2021-12-08 19:38:10
298	1468666233591681030	RT @NadiaWhittomeMP: The Downing Street party ...	en	None	2021-12-08 19:38:11
299	1468666238477864962	RT @AzamBaloshi: PPP District South Held Prote...	en	Karachi, Pakistan	2021-12-08 19:38:12

```
violent_df = pd.DataFrame(protest_df[protest_df['lang']=='en']['text'], columns=['text']) # Violent word detection
location_df = protest_df[['location', 'id']] # location of tweet
where_df = protest_df # find location in context
```

แยกข้อมูลเป็นส่วนๆ สำหรับการประมวลผล

1. Violent_df ดึงเฉพาะข้อมูลที่เป็นภาษาอังกฤษ มาวิเคราะห์ความรุนแรง (NLP)
2. Location_df นำข้อมูล location มา process
3. Where_df ข้อมูลสำหรับหาสถานที่ใน context

Location of tweet

```
def only_country(text):
    text = text.lower()
    text = text.replace(",", " ")
    spl_space_trigger = text.split(" ")

    if len(spl_space_trigger) > 1:
        text = spl_space_trigger[-1]
    else:
        return text

    if text == 'kingdom' or text == 'england':
        text = 'uk'
    elif text == 'states':
        text = 'usa'
    elif text == '':
        text = 'undefined'

    return text
```

ฟังก์ชั่นสำหรับเตรียมข้อมูลให้อยู่ในลักษณะเดียวกัน แล้วทำการ Groupby location เพื่อให้ได้ข้อมูลผลรวมที่เกิดขึ้นของแต่ละประเทศ

Result

location	
uk	33
undefined	13
usa	11
london	8
nigeria	6
pakistan	5
ca	4
deutschland	4
canada	4
serbia	3

```
location_df['location'] = location_df.apply(lambda row : only_country(row['location']) if(np.all(pd.notnull(row['location']))) else row['location'], axis=1)
```

```
/tmp/ipykernel_5234/1134729739.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  location_df['location'] = location_df.apply(lambda row : only_country(row['location']) if(np.all(pd.notnull(row['location']))) else row['location'], axis=1)
```

```
location_of_tweet = location_df.groupby("location").count().sort_values(by='id', ascending=False).head(10)
```

Location in context

Package spacy ใช้ในงานจำพวก name entity recognition (NER) ในที่นี้จะนำมาใช้เพื่อดึงชื่อประเทศใน Text

In [14]:
```
import spacy
from spacy import displacy
```

In [15]:
```
nlp = spacy.load("en_core_web_sm")
# Text with nlp
doc = nlp(" Multiple tornado warnings were issued for parts of New York on Sunday night.The first warning, which expired at 9 p.m
# Display Entities
displacy.render(doc, style="ent")
```

Multiple tornado warnings were issued for parts of New York GPE on Sunday DATE night TIME .The first ORDINAL wa

9 p.m. TIME , covered the Bronx GPE , Yonkers NORP and New Rochelle GPE . More than 2 million CARDINAL p

area.

In [16]:
```
def clean_line(text):
    text = re.sub(r"http\S+", "", text)
    text = re.sub(r"@[A-Za-z0-9]+", "", text)
    text = re.sub(r"#[A-Za-z0-9]+", "", text)
    text = text.replace(":","")
    text = text.lower()
    text = text.strip()
    return text

def loc_from_text(df):
    new_df = df.copy()
    new_df = new_df[new_df['lang']=='en']
    text_list = new_df['text']
    loc = []
    for text in text_list:
        doc = nlp(text)
        loc.extend(ent.text for ent in doc.ents if ent.label_ in ['GPE'])
    return loc
```

ฟังก์ชั่นสำหรับเตรียมข้อมูล และ ดึงชื่อประเทศออกจาก Text (ดึงเฉพาะ GPE เท่านั้น)

Result

location	count
london	7
canada	6
australia	5
uk	4
china	4
america	3
austria	3
lyari	3
chile	3
в|	2

Violent Detection

```
import findspark
findspark.init()

from pyspark.sql import SparkSession
from pyspark.sql.types import *
from pyspark.sql.functions import *
from pyspark.sql.functions import udf
```

```
spark_df = pandas_to_spark(violent_df)
spark_df.cache()
```

```
DataFrame[text: string]
```

```
spark_df.show(10)
```

```
[Stage 0:>
```

นำข้อมูลที่เตรียมไว้สำหรับ Violent detection มาแปลงอยู่ในรูปแบบของ Spark dataframe เพื่อเตรียมนำเข้าโมเดลและทำนายผล

```
+--------------------+
|                text|
+--------------------+
|RT @NadiaWhittome...|
|RT @MintPressNews...|
|Wooooiiii folks i...|
|Where and when do...|
|RT @Ysbryd5: Nurs...|
|RT @AndersonAfDMd...|
|RT @AlinejadMasih...|
|RT @TomPope695079...|
|The Palestinian p...|
|RT @GeorgeMonbiot...|
+--------------------+
only showing top 10 rows
```

ข้อมูลในรูปแบบ Spark Format

Preprocessing . . .

```
+--------------------+
|                text|
+--------------------+
|the downing stree...|
|media bears a res...|
|wooooiiii folks i...|
|where and when do...|
|nurse karen organ...|
|australia, austri...|
|this father who i...|
|hundreds of thous...|
|the palestinian p...|
|this should be al...|
+--------------------+
only showing top 10 rows
```

Prediction

```
import xgboost as xgb
from pyspark.ml.feature import Tokenizer, Word2Vec
from pyspark.ml import Pipeline
```

```
# Load trained parameters from GCS
w2v = Word2Vec.load("gs://twitter_testtt/w2v_model1")

tokenizer = Tokenizer(inputCol="text", outputCol="words")
w2v_pipeline = Pipeline(stages=[tokenizer, w2v])
w2v_pipeline_model = w2v_pipeline.fit(new_df)
train_df = w2v_pipeline_model.transform(new_df)
```

นำเข้าไป Tokenize และ W2V embedding จากโมเดลที่ train ไว้ โดยโหลด parameter จาก GCS

```
21/12/08 19:44:16 WARN com.github.fommil.netlib.BLA
21/12/08 19:44:16 WARN com.github.fommil.netlib.BLA
```

```
train_df.show(10)
```

```
+--------------------+--------------------+--------------------+
|                text|               words|            Features|
+--------------------+--------------------+--------------------+
|the downing stree...|[the, downing, st...|[-0.0105149733019...|
|media bears a res...|[media, bears, a,...|[-0.0083240476390...|
|wooooiiii folks i...|[wooooiiii, folks...|[-0.0110597727221...|
|where and when do...|[where, and, when...|[-0.0114352357632...|
|nurse karen organ...|[nurse, karen, or...|[-0.0118300816852...|
```

```
# Load XGBoost model parameters
model = xgb.Booster()
model.load_model("./model1.bst")
```

นำข้อมูล Feature ที่ได้จาก Word2Vec ไป Predict ในโมเดลของ XGBoost

```
train_features = train_df.select("Features").collect()
X_train = np.asarray([v[0].toArray() for v in train_features])
X_train = xgb.DMatrix(X_train)
pred = model.predict(X_train)
```

```
pred_df = pd.DataFrame(pred.astype(int), columns=["class"])
violent_result_df = pd.DataFrame(pred_df.value_counts(), columns=['count'])
```

```
violent_result_df
```

class	count
1	237
0	25

Result

Save result to BigQuery ---> BI

In [35]:
```
from google.cloud import bigquery

client = bigquery.Client()
table_id1 = 'ns01-project.results.tweets_location'
table_id2 = 'ns01-project.results.context_location'
table_id3 = 'ns01-project.results.violent_count'
```

ขั้นตอนนี้เป็นการเขียนข้อมูลทั้งหมดที่ได้ กลับลงไปใน BigQuery โดยการระบุ Schema และสร้างตารางบันทึก

In [240]:
```
job_config_table1 = bigquery.LoadJobConfig(schema=[
    bigquery.SchemaField("location", "STRING"),
    bigquery.SchemaField("id", "INT64")
])

job1 = client.load_table_from_dataframe(
    location_of_tweet, table_id1, job_config=job_config_table1
)
```

In [242]:
```
job_config_table2 = bigquery.LoadJobConfig(schema=[
    bigquery.SchemaField("location", "STRING"),
    bigquery.SchemaField("count", "INT64")
])

job2 = client.load_table_from_dataframe(
    loc_result_df, table_id2, job_config=job_config_table2
)
```

เพื่อใช้ในการ Visualization ต่อไป ซึ่งสามารถดึงจาก BigQuery ที่เป็น Data Warehouse ได้เลย

In [243]:
```
job_config_table3 = bigquery.LoadJobConfig(schema=[
    bigquery.SchemaField("class", "INT64"),
    bigquery.SchemaField("count", "INT64")
])

job3 = client.load_table_from_dataframe(
    violent_result_df, table_id3, job_config=job_config_table3
)
```

Visualization (BigQuery to Data studio)