ASSIGMENT 3

GROUP 5: 吳桑墨, 佳樂恩, 阮氏惠, 裴春全.

Hillary Clinton

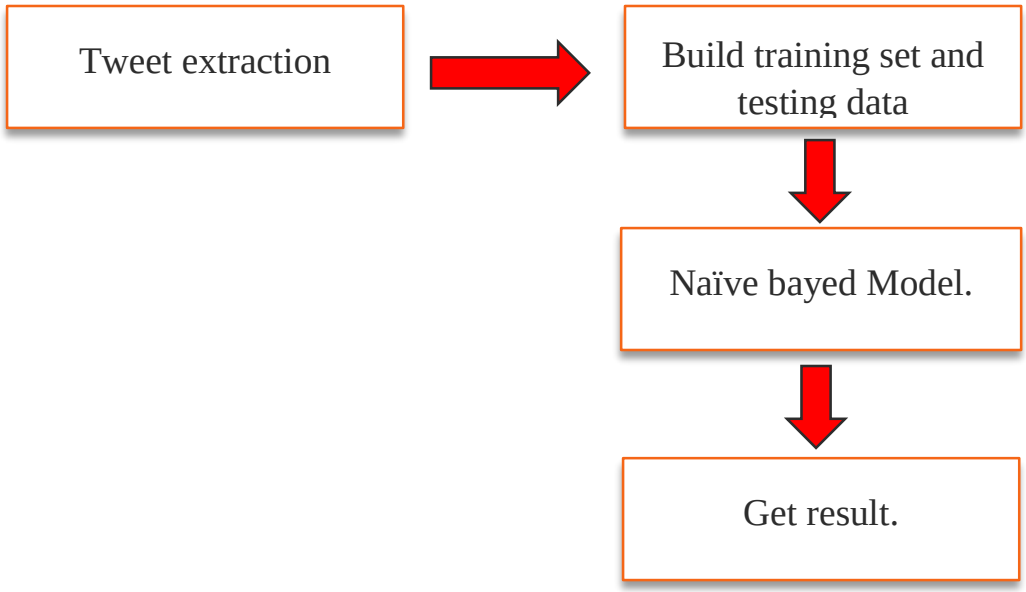
58.56%

Group 5 Prediction Result

Donald J. Trump

41.44%

System process



Tweet extraction

1. Twitter API and Tweet extraction

A tweet is structured from 7 dimensions, which are “id” “content” “user” “reply to” …. And for this assignment, we only get the tweet “content” and “id”. Following is tweet extraction API function which provided by Twitter.

```
public interface TwitterApiService {
  @GET(ApiConstants.TWITTER_HASHTAG_SEARCH_CODE )
  void getTweetList(
      @Header("Authorization") String authorization,
      @Query("keywords") String hashtag,
        @Query("count") int count,
      Callback<TweetList> callback
  );
}
```

Where: **keywords** are the input keywords for enquiring the tweet. For example: “Clinton Trump”; “vote Trump”; “Vote Hillary” …. “**count**” is set to maximum 100. (means we can obtain 100 tweet per query). After extracting tweet list base on input keywords, we save the list into cvs format.

```
File file = new File(path, "tweets.csv");
for (Tweet tweet : tweetsList.tweets) {
   s = s.append(tweet.text + "\"\r\n");
}
writeToFile(s.toString(), file);
```

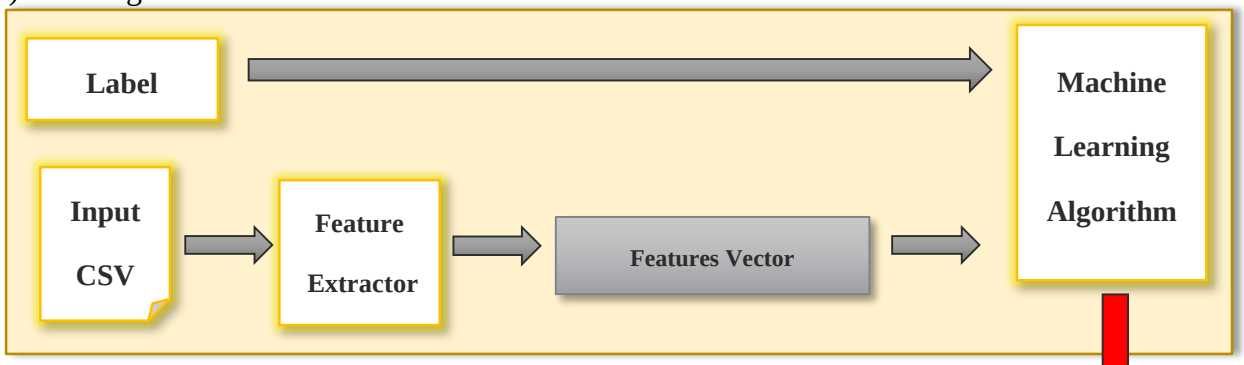
2. **Building Training data and Testing data**.
   - For testing data, we obtained 8000 tweets from following keywords “Clinton Trump”, “vote trump” “vote Hillary” “2016 election”
   - For training data, we manually collected 1000 tweets from testing data. Training tweets will be collected if it express obviously either “vote for Hillary” or “vote for trump”. These tweets then were labeled as “Positive Hillary” and “Positive Trump”

- For example:

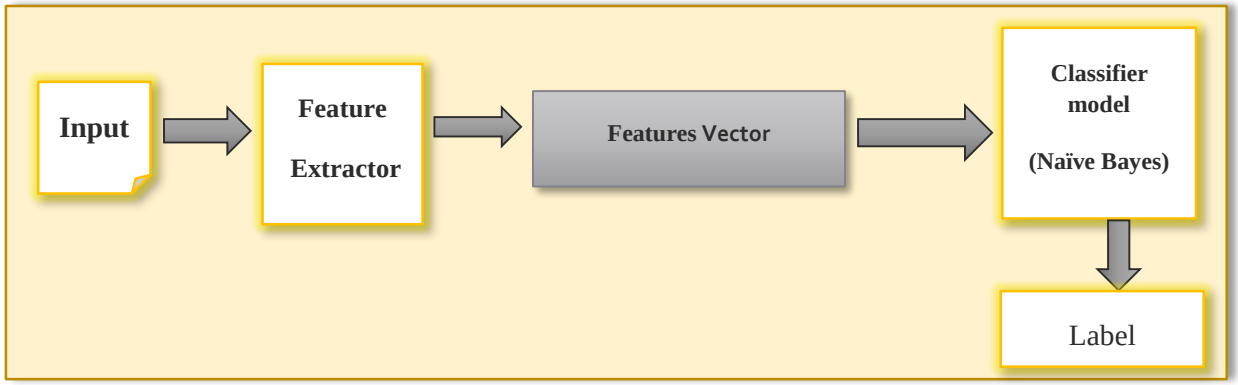
PositiveHillary	*RT @AdamParkhomenko: Trump's unfit. Get out that vote for Hillary and Democrats up and down the ballot and make generations to come proud hâ€¦*
PositiveHillary	*Less than a week until history is made. So proud to have voted for @HillaryClinton . Proud to get out the vote @ChelseaClinton*
PositiveHillary	*RT @abctweet100: Trump is Dangerous bc He is normalizing Racism. Is this what we want? For kids? STOP Him VOTE @HillaryClinton #USA https:/â€¦*
PositiveTrump	*@nathanTbernard @HillaryClinton *Are you out of you're mind? Or just stupid to vote HRC*
PositiveTrump	*@mindyfinn @Evan_McMullin This is who will be America's Miracle. Vote Trump!!! https://t.co/vKeTt1MFJv*
PositiveTrump	*@realDonaldTrump Face it, America's in a mess Hillary will be under pressure to Cover It up. America appoint a Clean Sweep Vote TrumpðŸ‘□ðŸ□»*

Work flow diagram

(a) Training Set



(b) Testing Set



Ordering training the classifier we construct the training set via manual (Training_set) over than 1000 records. The tweets can have some valuable info about it's sentiment and some word without meaning. Therefore, we need to remove stop word and preprocess the tweets.

Preprocess tweets (processTweet())

1. Convert the tweets to lower case.
2. Replace URLs with generic word URL.
3. Eliminate "@username" via regex matching or replace it with generic word AT_USER.
4. Replace #hashtag with exact same word without the hash.
5. Remove punctuation at the start and ending of tweets.

```
def processTweet(tweet):
    #Convert to Lower case
    tweet = tweet.lower()
    #Convert www. * or https?://* to URL
    tweet = re.sub('((www\.[^\s]+)|(https?://[^\s]+))','URL',tweet)
    #Convert @username to AT_USER
    tweet = re.sub('@[^\s]+','AT_USER',tweet)
    #Remove additional white spaces
    tweet = re.sub('[\s]+', ' ', tweet)
    #Replace #word with word
    tweet = re.sub(r'#([^\s]+)', r'\1', tweet)
    #trim
    tweet = tweet.strip('\'"')
    return tweet

#end
```

Feature Vector (getFeatureVector())

Feature vector is the most important concept in implementing a classifier. A good feature vector directly determines how successful your classifier will be. The feature vector is used to build a model which the classifier learns from the training data and further can be used to classify previously unseen data. In tweets, we can use the presence/absence of words that appear in tweet as feature. In the training data, consisting of vote Hillary Clinton and vote Donald Trump, we can split each tweet into words and add each word to the feature vector. Some of the words might not have any say in indicating the sentiment of a tweet and hence we can filter them out. Before adding the words to the feature vector, we need to preprocess them in order to filter, otherwise, the feature vector will explode.

Filtering tweet words

1. Remove stop word (is, a, the, with etc.) You can look in the detail of stop word at Stopword.txt. (getStopWordList())
2. Remove repeating letters (replaceTwoOrMore())
3. Remove word which don't start with an alphabet.

```
#start getStopWordList
def getStopWordList():
    #read the stopwords file and build a list
    stopWords = []
    stopWords.append('AT_USER')
    stopWords.append('URL')
    fp = open('data/stopwords.txt', 'r')
    line = fp.readline()
    while line:
        word = line.strip()
        stopWords.append(word)
        line = fp.readline()
    fp.close()
    return stopWords
#end
```

```
#start replace TwoOrore
def replaceTwoOrMore(s):
    #look for 2 or more repetitions of character and replace with the character itself
    pattern = re.compile(r"(.)\1{1,}",re.DOTALL)
    return pattern.sub(r"\1\1", s)
#end
```

```
def getFeatureVector(tweet):
    stopWords1= getStopWordList()
    featureVector = []
    #split tweet into words
    words = tweet.split()
    for w in words:
        #replace two or more with two occurrences
        w = replaceTwoOrMore(w)
        #strip punctuation
        w = w.strip('\'",.!?')
        #check if the word stats with an alphabet
        val = re.search(r"^[a-zA-Z][a-zA-Z0-9]*$", w)
        #ignore if it is a stop word
        if(w in stopWords1 or val is None):
            continue
        else:
            #print w
            featureVector.append(w.lower())
    return featureVector
#end
```

Supposing you want to know the detail of above source code, you can go to filtering_tweet_word.py. The main process is read_tweet.py ,including : Reading CSV file , Calling preprocess function, feature extraction (extract_features()). We

use National Language ToolKit (NLTK) library for extract features. NLTK has a neat feature which enables to extract features as above in bulk for all the tweets and can be done using the below code snippet. The line of interest is "**nltk.classify.apply_features(extract_features, tweets)**" where you pass in the tweets variable to the extract_features method.

```
inpTweets = csv.reader(open('data/New_Traning_set.csv', 'rb'), delimiter=',', quotechar='|')
featureList = []
tweets = []

for row in inpTweets:
    sentiment = row[0]
    tweet = row[1]
    processedTweet = processTweet(tweet)
    featureVector = getFeatureVector(processedTweet)
    featureList.extend(featureVector)
    tweets.append((featureVector,sentiment));
    #The sentiment string is also called label
#end loop
```

```
def extract_features(tweet):
    tweet_words = set(tweet)
    features = {}

    for word in featureList:
        features['contains(%s)' % word] = (word in tweet_words)
    return features
#end
```

Naive Bayes Classifier

In machine learning, naive Bayes classifiers are a family of simple probabilistic classifiers based on applying Bayes' theorem with strong (naive) independence assumptions between the features. Naive Bayes classifiers are highly scalable, requiring a number of parameters linear in the number of variables (features/predictors) in a learning problem. In this work, we use Naïve Bayes Classifier that is provided by NLTK library for creating the classifier model shown as the following figure.

```
training_set = nltk.classify.util.apply_features(extract_features, tweets)
# Train the classifier
NBClassifier = nltk.NaiveBayesClassifier.train(training_set)
```

Prediction the election result.

We use the tweet over than 8000 tweets for predict the election result (New_Testing_Set.csv). The source code for prediction the election result is shown as the following figure:

```
# Test the classifier
voteHillary = 0
voteTrump = 0
i=0
testFile = csv.reader(open('data/New_Testing_set.csv', 'rb'))
for row in testFile:
    i = i+1
    print i
    testTweet = row[0]
    processedTestTweet = processTweet(testTweet)
    test_set = getFeatureVector(processedTestTweet)
    #print 'The result is '+ NBClassifier.classify(extract_features(test_set))+'\n'
    if (NBClassifier.classify(extract_features(test_set)) == 'PositiveHillary'):
        voteHillary = voteHillary+1
    if (NBClassifier.classify(extract_features(test_set)) == 'PositiveTrump'):
        voteTrump = voteTrump+1
print "Hillary = " + str(format((voteHillary*100.0/8070),'.2f') ) + '%   Trump = ' + str(format((voteTrump*100.0/8070),'.2f')) + '% '
```

Result

The election result from our system is:

Hillary = 58.56%   Trump = 41.44%

=::=

Group Member

1) 103584601 Joy Nguyen joynguyen7@gmail.com
2) 104426602 Bui Xuani Toani toanbui1991@gmail.com
3) 104582602 Worapot Sommool title.wo@gmail.com
4) 104582604 Chalothon Chootong chalothon.cs@gmail.com