Tài liệu phân tích thiết kế

*Đề tài: Phân loại cảm xúc trong văn bản: tích cực, tiêu cực hay trung lập*

1. **Phân tích**
2. *Phân tích :*

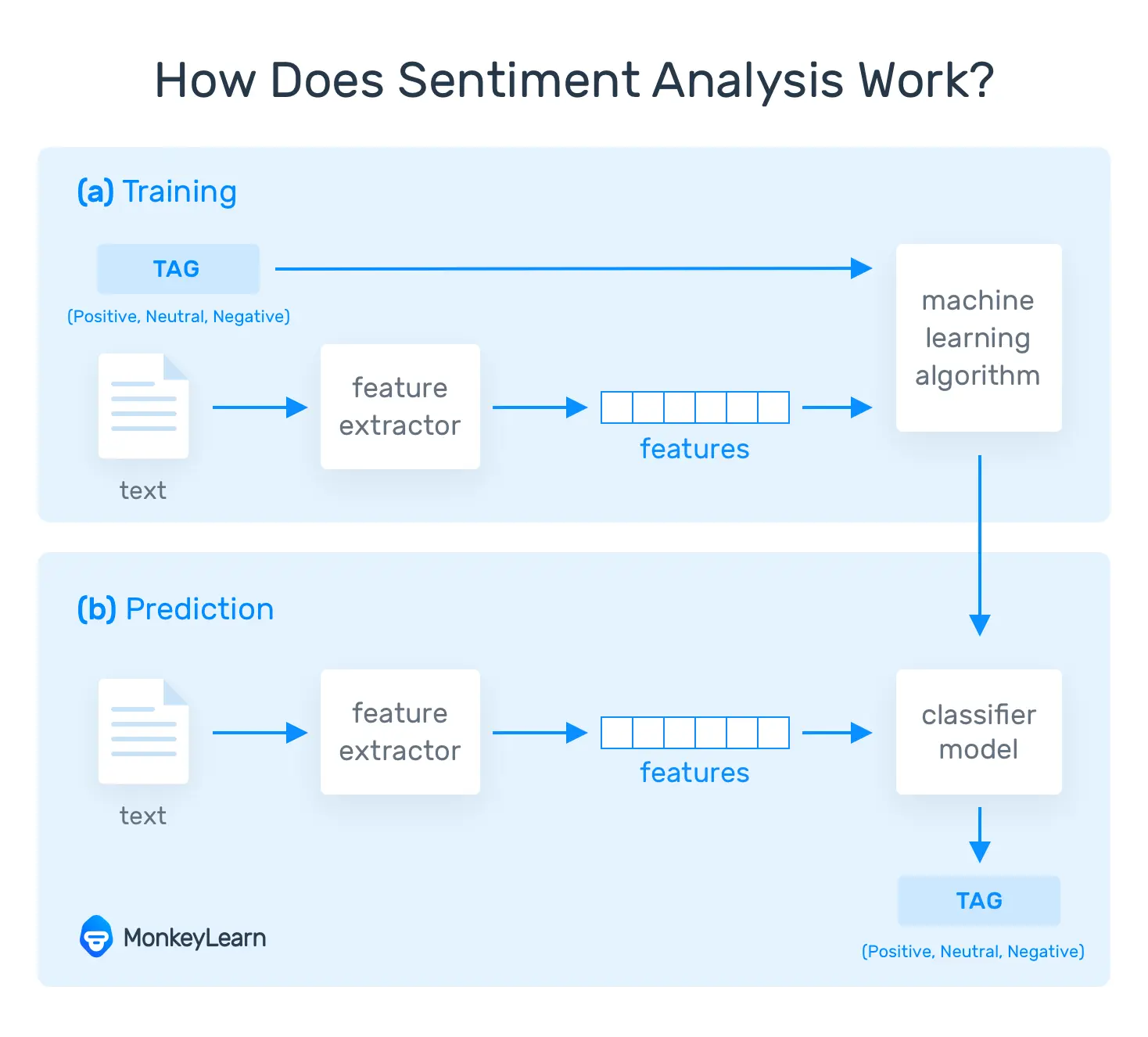
* Trong một đoạn văn bản, người viết sẽ đưa đến cho người đọc một cảm xúc tích cực, tiêu cực hay trung lập dựa vào từ ngữ sử dụng. Việc tiến hành phân loại cảm xúc có thể giúp doanh nghiệp nắm được thị hiếu của khách hàng đối với sản phẩm hoặc dịch vụ họ cung cấp để từ đó đưa ra chính sách phù hợp, đặc biệt trong thời điểm hiện nay, khi khách hàng bắt đầu bộc lộ suy nghĩ, cảm xúc và trải nghiệm của họ một cách thoải mái.
* Tuy nhiên việc tiến hành đọc từng phản hồi của một số lượng lớn khách hàng và tiến hành phân tích, chắt lọc thông tin cần rất nhiều thời gian và công sức của doanh nghiệp. Do đó, việc xây dựng một models có thể tự động hóa quá trình này sẽ giúp giảm thiểu được số lượng lớn thời gian và kinh phí cho các doanh nghiệp, khái niệm Data Sentiment Analysis (nghiên cứu phân loại thông tin) cũng được sinh ra từ đây.
* Sentiment Analysis sử dụng đa dạng các phương pháp phân tích ngôn ngữ (Natural Language Processing) và thuật toán khác nhau. Trong đó, các thuật toán được sử dụng chủ yếu bao gồm các bước cơ bản sau:
  + Chương trình được xây dựng cấu trúc chung: dữ liệu đưa vào để phân loại được dựa trên một cấu trúc chung (ví dụ: bỏ qua các dấu câu, các mạo từ, danh từ chỉ tên riêng hoặc địa điểm, các con số mang ý nghĩa về thời gian, địa điểm nơi chốn không mang lại hiệu quả khi đánh giá về cảm xúc…)
  + Chương trình tự động được xây dựng dựa trên phương pháp học máy (Machine Learning) từ những dữ liệu đưa vào trước đó
  + Chương trình lai: là sự kết hợp giữa những nguyên tắc có sẵn và tự động hóa

1. *Các chương trình con:*
2. Chương trình với đầu vào là các đoạn văn bản của người viết theo ngôn ngữ nói, viết thường ngày, đầu ra sẽ là đoạn văn bản còn lại các từ ngữ nguyên bản sau khi đã lược bỏ một số từ vựng không cần thiết cũng như dấu câu:

* Đối với dữ liệu đầu vào, nên là những bình luận hoặc ý kiến cá nhân của tập hợp nhiều cá thể, để từ đây xây dựng nên một vốn từ với độ phổ biến và thông dụng trong hàng ngày nhất, đặc biệt với các động từ, tính từ mang tính chất bộc lộ rõ cảm xúc của người viết như: good, bad, great, excellent, awful, exhausted, energentic,…
* Tuy nhiên đối với tiếng Anh nói riêng và các ngôn ngữ tự nhiên nói chung, ta nhận thấy có một điểm chung là cùng một nội dung có thể có nhiều cách đọc/viết/suy nghĩ khác nhau. Do đó, để máy có thể hiểu và học được những ngôn ngữ này, ta phải thực hiện các bước đơn giản hóa chúng về một cấu trúc chung để máy có thể phân tích và hiểu được. Đối với tiếng Anh, cách đơn giản nhất, ta có thể đưa về dạng từ nguyên thể (Infinitive Words) để phân tích, ví dụ như các động từ đuôi -ing, -ed có thể rút về dạng nguyên thể, hay một số trạng từ đánh vần tương tự như tính từ thêm đuôi -ly là những dạng phổ biến hay gặp nhất trong cuộc sống hàng ngày.
* Ngoài ra bên cạnh những đặc điểm về từ vựng, các yếu tố khác trong câu như dấu ngắt nghỉ câu như “ : ; , . & # “ ,… hay các danh từ chỉ tên riêng, địa điểm viết hoa, hoặc các con số mang tính chất chỉ thời gian địa điểm thường không mang lại yếu tố về thể hiện cảm xúc của người viết mà chúng ta đang cần phân loại. Do đó đây là một trong số những yếu tố nữa có thể đưa vào trong quá trình đơn giản hóa văn bản trước khi cho máy học.
* Khi mạng xã hội phát triển hơn, chúng ta có thêm một dạng ngôn ngữ, gọi là ngôn ngữ chat. Ở đây lại có một số đặc điểm khác hơn so với các văn bản thông thường đó là sự xuất hiện của một số từ ngữ mang ý nghĩa tượng thanh, một số ký hiệu mang ý nghĩa buộc lộ cảm xúc (hay còn gọi là emoji như thế này ☺) hay đôi khi ngay cả những ký hiệu ngắt câu như “! ? !?! ##” cũng mang yếu tố bộc lộ cảm xúc của người viết. Tuy nhiên, trong bài thiết kế này sẽ không phân tích sâu vào những đặc điểm này, các đặc tính như trên sẽ được lược bỏ nhưng các non-sense words (từ không có nghĩa) hoặc punctuations (dấu câu)

b. Chương trình tự động hóa xây dựng dựa trên phương pháp học máy:

* Đầu vào của hệ thống này sẽ là các đoạn văn bản đã được đơn giản hóa chỉ còn lại những từ khóa quan trọng được được vào với các chỉ số:
  + Negative: tương ứng với giá trị là -1
  + Neutral: tương ứng với giá trị 0
  + Positive: tương ứng với giá trị 1



* Khác với chương trình thiên về xử lý văn bản một cách thủ công như trên, chương trình tự động hóa này lại phụ thuộc vào phương pháp của học máy. Quá trình nghiên cứu về cảm xúc này thường được xây dựng dựa trên mô hình bài toán phân loại, nhờ đó đối tượng đi phân loại này được nạp vào những đoạn văn bản và trả về một phạm trù, ở đây là positive, negative và neutral
* Trong quá trình học (training) (a) ở trên, mô hình của chúng ta sẽ học liên kết các input tương ứng với các output dựa trên mẫu thử nghiệm được sử dụng cho training. Các đặc tính thu được từ đoạn văn bản sẽ được chuyển thành các vector đặc tính. Các cặp vector gồm đặc tính nêu trên cùng với các nhãn phân loại sẵn được nạp vào thuật toán học máy để tạo ra mô hình ta cần.
* Từ đây, trong quá trình dự đoán (prediction) (b), các đặc tính tiếp tục được sử dụng để chuyển các đoạn văn bản chưa có trước đó thành các vector đặc tính. Các vector này tiếp tục được đưa vào model, và tạo ra các nhãn dự đoán (positive, negative, neutral)
* Bước đầu tiên của phân loại văn bản theo học máy đó là chuyển văn bản thành các vector đặc tính, hay còn gọi là vector hóa, và một trong số những cách kinh điển nhất đó là bags-of-words hay còn gọi là bag-of-ngrams cùng với tần số của nó. Ở đây, ta có thể sử dụng sẵn CountVectorizer của sklearn là một module có sẵn trong Python, bên cạnh tính tiện dụng thì áp dụng CountVectorizer có sẵn này không thể giúp chúng ta lọc ra các từ đồng nghĩa với nhau, ví dụ như compete và contend đều mang nghĩa chung là cạnh tranh sẽ được chuyển thành 2 vector khác nhau, đó là xét trong bài toán hẹp khi ta chỉ phân loại ra 3 nhãn.
* Thuật toán phân loại: các thuật toán phân loại ở đây thường được kể đến là các thuật toán về xác suất như Naïve Bayes, Logistic Regression, Support Vector Machines hoặc Neural Networks. Tuy nhiên trong bài này ta sẽ chỉ để cập chủ yếu đếu Naïve Bayes, là thuật toán xác suất cơ bản nhất sử dụng định lý Bayes để dự đoán loại văn bản

1. Phương pháp lai (Hybird Approaches)

* Chương trình mô hình lai kết hợp các thành phần lý tưởng của 2 chương trình phân tích chọn lọc dữ liệu và tự động hóa trên thành một hệ thống.
* Ưu điểm dễ thấy của chương trình này chính là độ chính xác thường cao hơn.

1. Đặc tả kiểm thử (Test Case):

* Để tiến hành kiểm tra độ chính xác của mô hình, ta sẽ chia dữ liệu có sẵn thành 2 phần theo tỷ là 75% cho training và 25% cho testing.
* Sau khi tiến hành training mô hình 75% dữ liệu có sẵn, ta sẽ đưa 25% văn bản đầu vào còn lại chưa có nhãn phân loại để mô hình tiến hành dự đoán và đưa ra độ chính xác của mô hình để tiến hành đánh giá:

1. Kiểm thử (Kết quả Test):

* Đối với Naïve Bayes được áp dụng ở đây, độ chính xác xấp sỉ đạt 64.8%

1. **Thiết kế:**
2. Thu thập dữ liệu:

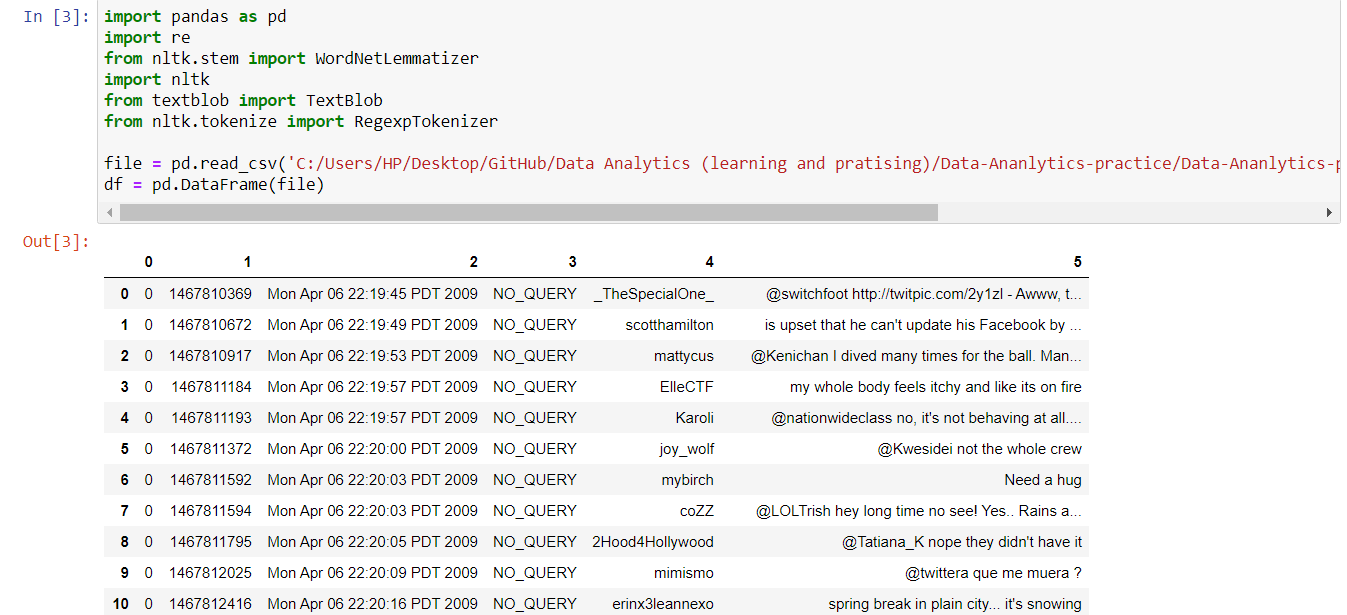
* Để thu thập dữ liệu, ta sẽ xây dựng một hàm với dữ liệu đầu vào là url của trang web mà ta sẽ lấy thông tin, đầu ra sẽ là các nội dung thông tin mà ta quan tâm như tên người viết, chủ đề bài viết, nội dung bài viết, và đánh giá có sẵn của người viết (nếu có)
* Để thực hiện được điều này, cần phải tự phân tích các thuộc tính trên html của trang web lựa chọn
* Trong các thư viên của Python, requests là thư viện cho phép ta gửi yêu cầu đến các HTTP. Lúc này các thông tin yêu cầu sẽ được lưu dưới dạng một response, tuy nhiên ở đây ta không nhất thiết phải lấy tất cả các thông tin của HTTP như coockie, encoding,… dữ liệu cần sử dụng ở đây là văn bản nên ta sẽ chỉ crawl response.text về
* Tiếp theo để trích xuất dữ liệu từ các tập tin HTML, ta sẽ sử dụng thư viên beautifulsoup4, cụ thể là đối tượng BeautifulSoup. Đối tượng này nhận hai đối số, đối số đầu tiên là markup thật sự, đối số thứ hai là parser muốn sử dụng. Ở đây ta sẽ để markup là response.text mà ta vừa crawl được ở trên, còn parser ở đây được lựa chọn là lxml bởi tốc độ phân tích HTML của nó rất nhanh.
* Đối với thư viện beautifulsoup4, BeautifulSoup phân tích HTML đưa về dạng một cây các đối tượng Python gồm: Tag, NavigableString, BeautifulSoup và Comment. Lúc này ta cần dựa vào các thuộc tính để đưa thông tin cần sử dụng về dạng array. Dưới đây là code minh họa:

from bs4 import BeautifulSoup  
import requests  
import numpy as np   
import pandas as pd  
#import csv  
  
urls = ['https://www.movieforums.com/reviews/search/any/higher',  
 'https://www.movieforums.com/reviews/search/any/higher/?page=2',  
 'https://www.movieforums.com/reviews/search/any/higher//?page=3',  
 'https://www.movieforums.com/reviews/search/any/higher//?page=4',  
 'https://www.movieforums.com/reviews/search/any/higher//?page=5',  
 'https://www.movieforums.com/reviews/search/any/higher//?page=6',  
 'https://www.movieforums.com/reviews/search/any/higher//?page=7',  
 'https://www.movieforums.com/reviews/search/any/higher//?page=8',  
 'https://www.movieforums.com/reviews/search/any/higher//?page=9',  
 'https://www.movieforums.com/reviews/search/any/higher//?page=10',  
 'https://www.movieforums.com/reviews/search/any/higher//?page=11',  
 'https://www.movieforums.com/reviews/search/any/higher//?page=12',  
 'https://www.movieforums.com/reviews/search/any/higher//?page=13',  
 'https://www.movieforums.com/reviews/search/any/higher//?page=14',  
 'https://www.movieforums.com/reviews/search/any/higher//?page=15']  
  
def url\_to\_transcript(url):  
 req = requests.get(url)  
 soup = BeautifulSoup(req.text, 'lxml')  
 titles = [title.text for title in soup.find\_all('a', class\_='reviewtitle')]  
 contents = [content.text for content in soup.find\_all('div', class\_='reviewexcerpt')]  
 rates = [rate.div['title'].split(': '[1]) for rate in soup.find\_all('div', class\_='review')]  
 reviewers = [article.text.split(' ')[1] for article in soup.find\_all('a', class\_='authorlink light')]  
 #reviewers = list(dict.fromkeys(reviewers))  
 print(url)  
 return [titles, contents, rates, reviewers]  
  
data = [url\_to\_transcript(u) for u in urls]

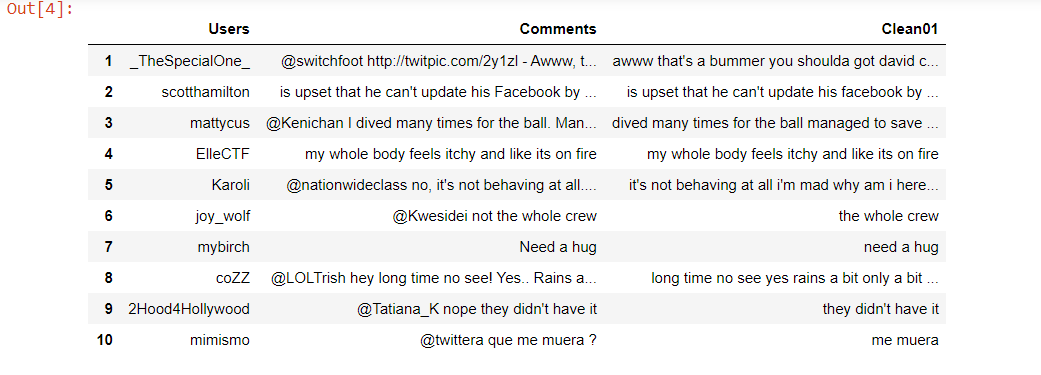
return [titles, contents, rates, reviewers]

* Tuy nhiên để đưa vào quá trình học máy cần một số lượng lớn dữ liệu và từ vựng thông dụng (ở đây thông tin crawl về là từ một website review phim, do đó vốn từ ở đây cũng chủ yếu xoanh quanh những phản hồi và cảm xúc, nhận xét về phim hơn là cảm xúc chung) nên việc triển khai thu thập dữ liệu như thế này cần rất nhiều thời gian. Ta sẽ hướng đến sử dụng một dataset có sẵn (<http://cs.stanford.edu/people/alecmgo/trainingandtestdata.zip>). Dataset này được thu thập bằng Twitter API kết hợp 160,000 tweets với các biểu tượng cảm xúc (emoji) đã được loại bỏ, gồm 6 phần: polarity, tweetID, date, query, user name và text of tweet.

1. Dọn dẹp dữ liệu:



* Từ dataset có sẵn, ở bước này các bước dọn dẹp dữ liệu được tiến hành khá thủ công và chủ yếu dựa trên quan sát của người lập trình
* Có thể thấy cái cột 0 1 2 3 tương ứng với polarity, tweetID, date, query ở đây không sử dụng đến, ta sẽ bỏ đi các cột này để dễ quan sát hơn
* Ngoài ra khi nhìn vào cột 5 tương ứng với text of tweet, các mentioned như “@switchfood” hay các link đính kèm như “<http://...> “ cũng không mang lại ý nghĩa cho việc phân loại văn bản. Ta sẽ xây dựng một hàm để loại bỏ những thông tin không dùng đến này kết hợp với những nguyên tắc chung như loại bỏ dấu câu, viết hoa, thời gian,...



* Sau khi dọn dẹp lần thứ 1, ta được cột Clean01 như trên, ở lần thứ 2, nhận thấy sự xuất hiện của các từ viết tắt khi tokenizer bị chia thành các vector khác nhau, lúc này việc viết thêm một hàm để đưa các từ viết tắt của nó về dạng nguyên thể để loại bỏ các stop words (từ ngắt: ví dụ như các đại từ hay mạo từ như I, he, she, a, an, the, be,…) sẽ giúp máy học hiệu quả hơn
* Ngoài ra khi viết, đối với tiếng Anh, ta có dạng danh động từ hoặc các động từ chia theo các thì hoặc hoàn cảnh cụ thể, do đó việc đưa các từ về dạng nguyên thể sẽ giúp giảm bớt tính phức tạp của văn bản khi đưa vào mô hình máy học. Có 2 phương pháp chính của NLP (Natural Language Proccessing) thường được sử dụng trong trường hợp này là: Stemming và Lemmtization. Stemming thì xử lý văn bản theo cách thô, nếu như văn bản là dạng động từ đuôi chia theo quy tắc thông thường thì Stemming có thể thực hiện tốt, tuy nhiên các trường hợp chia khác với quy tắc thông thường thì Stemming lại xử lý hơi máy móc (ví dụ: denied -> deni thay vì là deny). Ngược lại với Stemming, Lemmatization hướng đến mục đích xa hơn là cắt bớt đuôi của từ một cách khá thô như stemming, đó là đưa vào một ngôn ngữ với một vốn từ đầy đủ để áp dụng cho nghiên cứu hình thái của từ. Có thể thấy sử dụng Lemmatization sẽ phù hợp hơn với tiêu chí đơn giản hóa text mà vẫn giữ được ý nghĩa của chúng cho quá trình đưa vào học máy sau này.

1. Phân loại các nhãn cho văn bản, chuẩn bị đưa vào học máy:

* Đối với dataset có sẵn mà chúng ta sử dụng ở đây thì chưa bao gồm các nhãn phân loại, mặc dù trên dataset có cột polarity có thể giúp phân loại nhãn như mong muốn. Tuy nhiên tất cả các giá trị thu được ở cột này đều bằng 0, điều này không mang lại ý nghĩ cho quá trình phân loại.
* Do đó ở đây ngoài cách tiến hành một cách thủ công tự phân loại 160,000 tweets, ta có áp dụng một module có sẵn của Python để tham khảo thêm là TextBlob trong textblob.
* Hàm TextBlob.sentiment(polarity) của thư viện sẽ trả về cho chúng điểm của đoạn văn dựa trên các từ vựng đưa vào (nằm trong khoảng từ -1 đến 1), dựa trên số điểm có sẵn ta sẽ gán các nhãn riêng cho từng văn bản:
  + Polarity < - 0.01 : gán nhãn Negative
  + -0.01 < Polarity < 0.01: gán nhãn Neutral
  + Polarity > 0.01: gán nhãn Positive
* Sau khi có được dữ liệu ở dạng đơn giản nhất (X) cùng các nhãn tương ứng (y), ta sẽ tiến hành vector hóa văn bản thành các vector đặc tính để phân loại. Các stop words cũng sẽ được đồng thời loại bỏ ở bước này, ngoài ra trong Vectorizer của Scikit-learn mà chúng ta sử dụng còn có một thuộc tính là ngram\_range:
  + Với cùng 1 câu: ‘various singer perform full mix concert’
  + Nếu ngram\_range = (1, 1), ta sẽ có các vector thuộc tính dưới dạng: {‘various’ : 0, ‘singer’ : 1, ‘perform’ : 2, ‘full’ : 3, ‘mix’ : 4, ‘concert’: 5}
  + Nếu ngram\_range = (1, 2), ta sẽ có các vector thuộc tính dưới dạng: {‘various’ : 0, ‘various singer’ :1, ‘singer’: 2, ‘singer perform’: 3, ‘perform’: 4, ‘perform full’, …}
  + Trong quá trình máy học, ta có thể thay đổi thuộc tính này để tăng độ chính xác của mô hình nếu cần

1. Chia dữ liệu và thực hiện thuật toán học máy:

* Bằng thực nghiệm, các nhà nghiên cứu đã chứng minh rằng so với việc đưa toàn bộ dữ liệu vào mô hình máy học để training có độ tin cậy không cao bằng việc thực hiện quá trình training – testing – training.
* Khi thực hiện thuật toán học máy, ta có thể thử nghiệm trên nhiều phương pháp khác nhau để đưa ra được tỷ lệ chính xác cao nhất (fit – mode), các phương pháp phân loại có thể kể đến rất nhiều, tuy nhiên trong bài báo cáo này ta chỉ đề cập đến phương pháp đơn giản nhất là Naïve Bayes:

**Naive Bayes Classification** (NBC) là một thuật toán phân loại dựa trên tính toán xác suất áp dụng định lý Bayes. Thuật toán này thuộc nhóm [Supervised Learning](https://1upnote.me/tags/supervised-learning/) (Học có giám sát). Theo định lý Bayes, ta có công thức tính xác suất ngẫu nhiên của sự kiện y khi biết x như sau:

Giả sử ta phân chia 1 sự kiện xx thành nn thành phần khác nhau

**Naive Bayes** theo đúng như tên gọi dựa vào một giả thiết **ngây thơ** rằng ​ là các thành phần độc lập với nhau. Từ đó ta có thể tính được:

Từ đó ta có:

Naïve Bayes có 2 mô hình thuật toán là: Bernoulli và Multinomial. Ở đây ta sẽ tiến hành thử trên mô hình Multinomial có công thức:

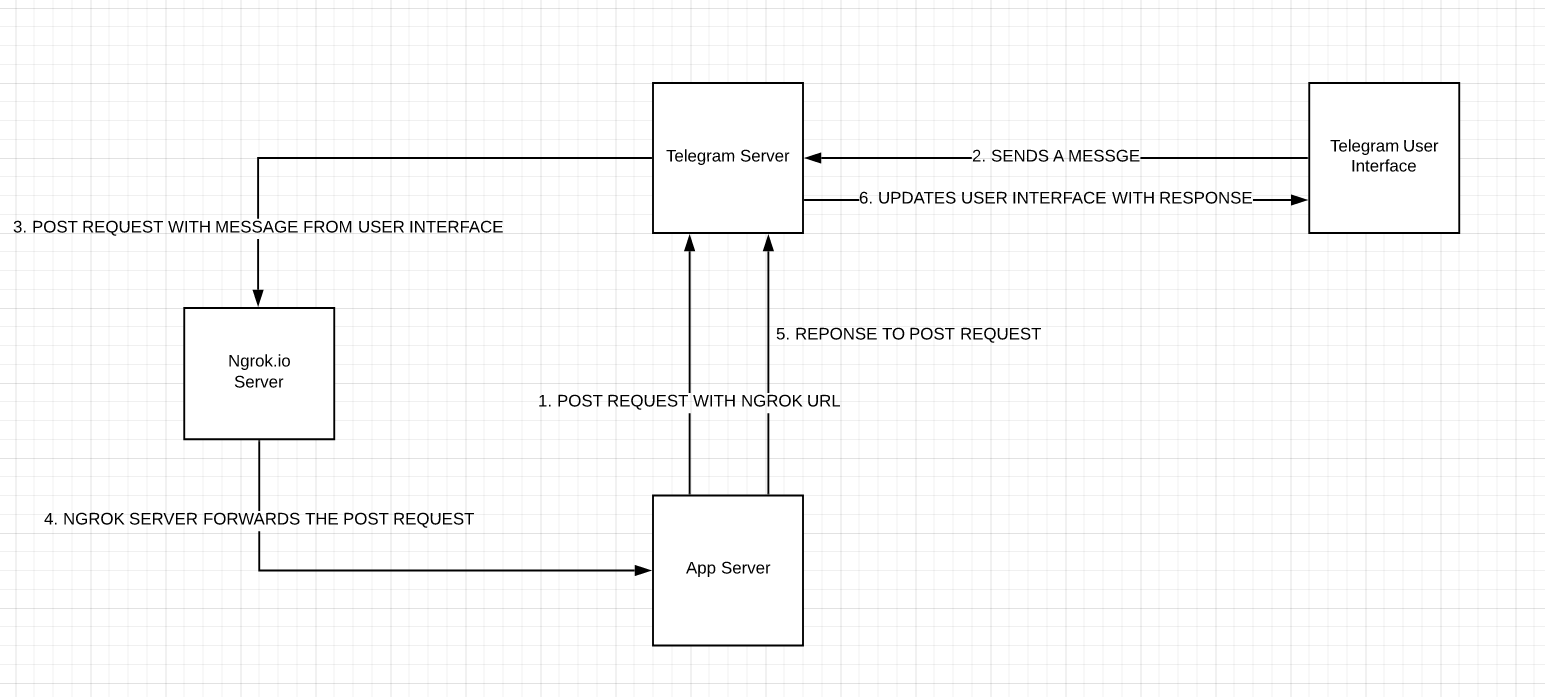
trong đó là số lần từ xuất hiện trong văn bản và là số lần các từ xuất hiện trong văn bản

Từ đây ta sẽ có được xác suất một từ Positive, Negative hay Neutral, và khi predict, máy sẽ dựa trên những xác suất có sẵn và tính toán để đưa điểm polarity score và từ đó dán nhãn dự đoán cho text bất kỳ

1. Kiểm tra độ chính xác của mô hình học máy:

* Áp dụng điều này đưa vào mô hình học máy, ta sẽ chia dataset có sẵn thành 2 phần với tỷ lệ: 75% văn bản và nhãn để training và đưa 25% văn bản còn lại vào để dự đoán predict, sử dụng kết quả predict của mô hình để so sánh với các nhãn có sẵn, ta sẽ có xác suất về độ chính xác của mô hình.
* Đối với phân loại văn bản, đặc biệt là cảm xúc, có rất nhiều cách thể hiện khác nhau trong đó có cả những phạm trù như nói mỉa mai một cách tích cực, những trường hợp này máy không thể dự đoán chính xác hoàn toàn. Do đó thông thường độ chính xác của mô hình chỉ khoảng 70-80% là khá cao

1. Mô tả quá trình hoạt động của Telegram bot:



* Khi người dùng giao tiếp với Telegram server thông qua Telegram User Interface, yêu cầu gửi tin nhắn cùng với nội dung của tin nhắn người gửi sẽ được đưa đến ngrok.io server, vai trò của nó sẽ gần như báo cho ta biết mỗi khi người dùng gửi tin nhắn. Sau đó ngrok sẽ đưa yêu cầu gửi tin nhắn đến App Server và tại đây sẽ nơi xử lý dữ liệu để đưa ra phản hồi lại cho người dùng. Sau khi xử lý và đưa ra đoạn tin nhắn hồi đáp với tin nhắn gửi trước đó, yêu cầu gửi phản hồi sẽ được đưa đến Telegram server để từ đây đưa đến phản hồi cho người dùng thông qua Telegram User Interface.
* Ở đây cấu tạo từ 2 phần chính, 1 phần để cập nhật các sự kiện đến (tin nhắn gửi đi) và 1 phần xử lý tin nhắn (hay còn gọi lại webhook – có dạng như một vòng lặp để thực hiện quá trình xử lý và phản hồi tin nhắn)
* Áp dụng điều này vào chủ đề phân loại của chúng ta, từ mô hình học máy đã có ở trên, ta sẽ thực hiện dự đoán nhãn của tin nhắn do người dùng gửi đến và phản hồi lại thông qua hành động send\_message() trong mô hình Telegram\_Bot xây dựng:
  + Telegram\_Bot sẽ bao gồm 2 thông tin cần sử dụng:
    - Chat\_ID của người dùng: để thu thập tin nhắn dạng văn bản của họ và gửi phản hồi trở lại.
    - Text: Để đưa vào chương trình vector hóa và dự đoán nhãn cảm xúc tương ứng.
  + Đầu tiên khi người dùng gửi một đoạn tin nhắn, đoạn tin nhắn này sẽ được gửi lên ngrok server và được trích xuất để lấy được các thông tin cần cho đối tượng thông qua hàm parse\_webhook\_data. Cách chọn tag và thuộc tính khá tương đồng với cách crawl dữ liệu từ web
  + Lúc này, hệ thống sẽ gửi yêu cầu post phản hồi đến telegram server sau khi đoạn text được chuyển hóa thành các vector đặc tính và dự đoán nhãn theo model xây dựng trước đó. Hoạt động action này bao gồm 2 phần:
    - Nhận văn bản đến, đưa vào mô hình và trả về phản hồi
    - Gửi phản hồi đi thông qua requests với các thông tin: outgoing\_message\_text và chat\_ID của người dùng

1. Hướng cải tiến độ chính xác của mô hình:

* Đối với mô hình ở trên, dữ liệu vẫn chưa “sạch” hoàn toàn do vẫn còn sự xuất hiện của các từ tượng thanh được kéo dài âm cuối hoặc âm giữa chưa bị loại bỏ hoàn toàn cùng các một số trạng từ vẫn có nghĩa với Lemmatization nên vẫn được chia thành các vector đặc tính khác nhau
* Trong quá trình vector hóa văn bản, ở đây ta mới chỉ áp dụng ngram\_range là (1, 1) là cách đơn giản nhất. Ngoài ra có thể thử ngram\_range là (1, 2) và (1, 3) để kiểm tra độ chính xác của mô hình so sánh với các mô hình khác nhằm nâng cao độ chính xác (đặc biệt với các phrase như: try, try hard,…)
* Bên cạnh Multinomial Naïve Bayes, các mô hình thuật toán như Logistic Regression, Support Vector Machines hoặc Neural Networks cũng hoàn toàn có thể mang lại độ chính xác cao hơn.