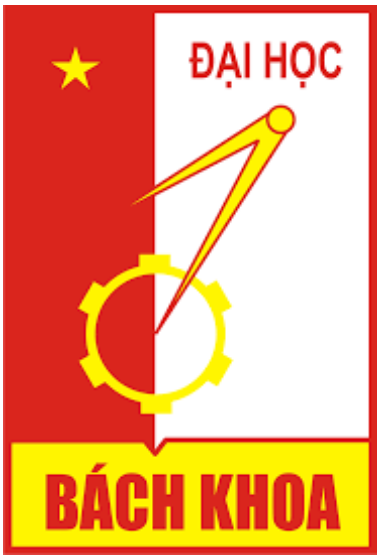
ĐẠI HỌC BÁCH KHOA HÀ NỘI

TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

———————— * ————————



BÁO CÁO PROJECT I

PHÂN LOẠI VĂN BẢN DỰA TRÊN HỌC MÁY SỬ DỤNG THƯ VIỆN SCIKITLEARN

Giảng viên hướng dẫn **:TS Nguyễn Kiêm Hiếu**

MSSV **:20204578**

Sinh viên thực hiện **:Vũ Xuân Lợi**

Hà Nội, ngày 30 tháng 01 năm 2023

MÔ TẢ YÊU CẦU BÀI TOÁN

Phân loại văn bản dựa trên học máy sử dụng thư viện scikitlearn (https://scikit-learn.org/stable/):

1/ 4 tuần: Biểu diễn văn bản dựa trên túi từ + phương pháp phân loại Multinomial Naive Bayes (MNB):

- Thu thập 1000 văn bản tiếng Việt thuộc 10 chủ đề, mỗi chủ đề 100 văn bản

- Áp dụng tách câu, tách từ, loại bỏ dấu câu, loại bỏ từ dừng cho văn bản

- Chia dữ liệu thành hai tập huấn luyện (80%) và kiểm thử (20%). Tìm hiểu và áp dụng phương pháp phân loại MNB

- Tìm hiểu các độ đo đánh giá precision, recall và F-score (micro và macro) và đánh giá bài toán.

2/ 4 tuần: Biểu diến văn bản dựa trên tf-idf + phương pháp phân loại k-nearest Neighbour (KNN)

- Tìm hiểu phương pháp đánh trọng số tf-idf

- Tìm hiểu và áp dụng KNN với các đô đo khoảng cách khác nhau (cosine, euclide...)

- Đánh giá trên trên tập dữ liệu đã thu thập

3/ 4 tuần: Biểu diễn văn bản dựa trên tf-idf + phương pháp phân loại Support Vector Machines (SVM)

- Tìm hiểu và áp dụng SVM cho tập dữ liệu đã thu thập

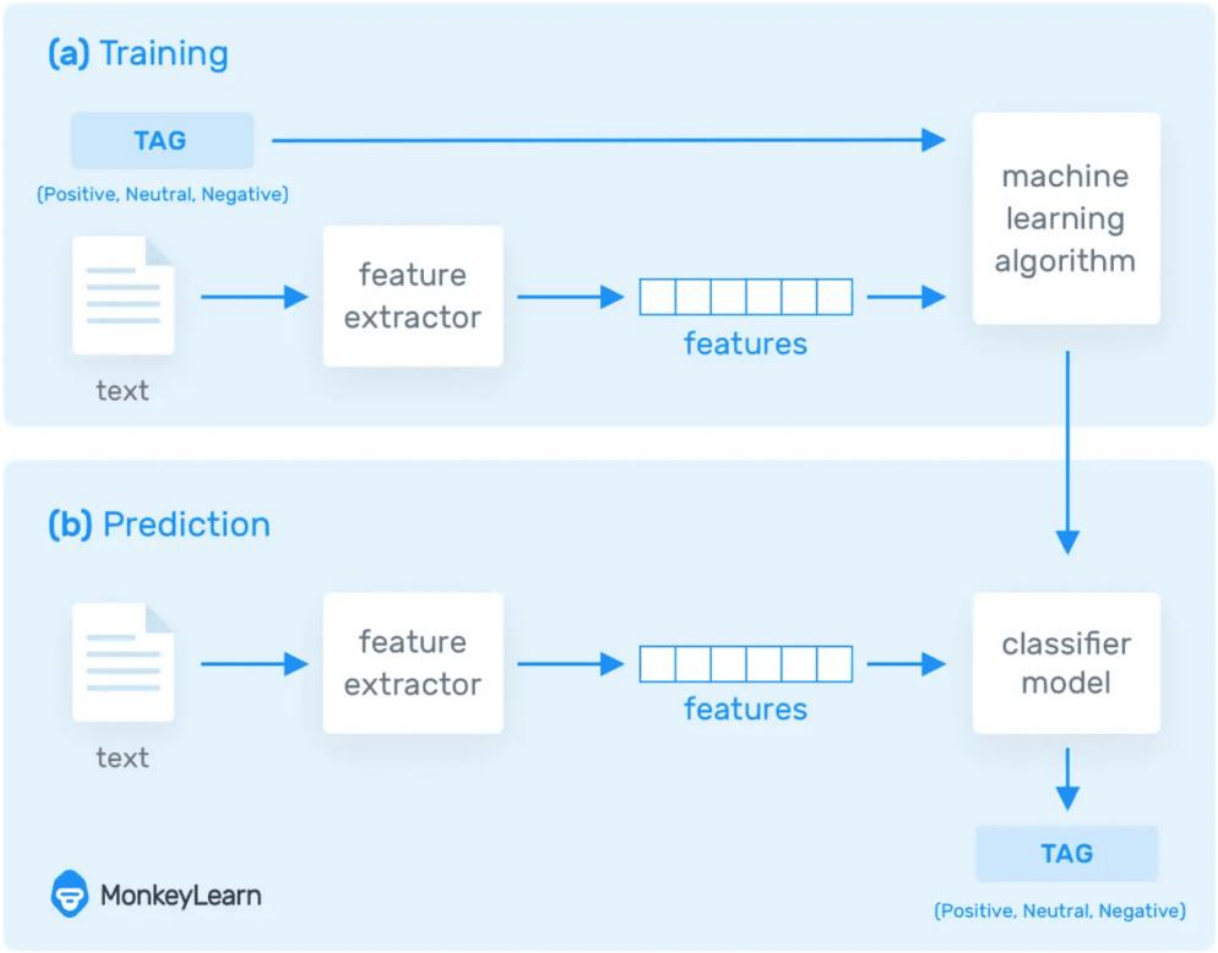
Mục Lục

Danh mục hình ảnh

PHÂN LOẠI VĂN BẢN DỰA TRÊN HỌC MÁY SỬ DỤNG THƯ VIỆN SCIKITLEARN

I. Bài toán phân loại văn bản

Phân loại văn bản (Text Classification) là bài toán thuộc nhóm học có giám sát (Supervised learning) trong học máy. Bài toán này yêu cầu dữ liệu cần có nhãn (label). Mô hình sẽ học từ dữ liệu có nhãn đó, sau đó được dùng để dự đoán nhãn cho các dữ liệu mới mà mô hình chưa gặp.



Hình 1: Text Classification

- *Giai đoạn 1*: Huấn luyện (training) là giai đoạn học tập của mô hình phân loại văn bản. Ở bước này, mô hình sẽ học từ dữ liệu có nhãn (trong ảnh trên nhãn là Possitive, Negative, Neutral). Dữ liệu văn bản sẽ được số hóa thông qua bộ

trích xuất đặc trưng (feature extractor) để mỗi mẫu dữ liệu trong tập huấn luyện trở thành 1 vector nhiều chiều (đặc trưng). Thuật toán máy học sẽ học và tối ưu các tham số để đạt được kết quả tốt trên tập dữ liệu này. Nhãn của dữ liệu được dùng để đánh giá việc mô hình học tốt không và dựa vào đó để tối ưu.

- *Giai đoạn 2*: Dự đoán (prediction), là giai đoạn sử dụng mô hình học máy sau khi nó đã học xong. Ở giai đoạn này, dữ liệu cần dự đoán cũng vẫn thực hiện các bước trích xuất đặc trưng. Mô hình đã học sau đó nhận đầu vào là đặc trưng đó và đưa ra kết quả dự đoán.

II. Chuẩn bị dữ liệu và tiền xử lý dữ liệu

Dữ liệu là yếu tố quan trọng nhất và cũng là vấn đề mà chúng ta cần quan tâm nhất. Trong quá trình xây dựng một hệ thống phân loại văn bản, bước chuẩn bị và tiền xử lý dữ liệu quyết định tới thành bại của hệ thống hơn cả.

Với bài toán phân loại văn bản tin tức tiếng Việt, dữ liệu bạn cần chuẩn bị là dữ liệu các bài báo tiếng Việt kèm theo chủ đề của bài báo đó.

1. Cài đặt một số hàm tiền xử lý văn bản cần thiết

```
!pip3 install --user underthesea
import regex as re
from underthesea import word_tokenize

uniChars = "àáảãạâầấẩẫậăằắẳẵặèéẻẽẹêềếểễệđìíỉĩịòóỏõọôồốổỗộơờớởỡợùúủũụưừứửữựỳýỷỹỵÀÁẢÃẠÂẦẤẨẪẬĂẰẮẲẴẶÈÉẺẼẸÊỀẾỂỄỆĐÌÍỈĨỊÒÓỎÕỌÔỒỐỔỖỘƠỜỚỞỠỢ
unsignChars = "aaaaaaaaaaaaaaaaaeeeeeeeeeeeediiiiiooooooooooooooooouuuuuuuuuuyyyyyAAAAAAAAAAAAAAAAAEEEEEEEEEEEDIIIOOOOOOOOOOOOOO
```

Hình 2: Code cài đặt regex, underthesea

2. Chuyển Unicode dựng sẵn về Unicode tổ hợp

```
def convert_unicode(txt):
    return re.sub(
        r'à|á|ả|ã|ạ|ầ|ấ|ẩ|ẫ|ậ|ằ|ắ|ẳ|ẵ|ặ|è|é|ẻ|ẽ|ẹ|ề|ế|ể|ễ|ệ|ì|í|ỉ|ĩ|ị|ò|ó|ỏ|õ|ọ|ồ|ố|ổ|ỗ|ộ|ờ|ớ|ở|ỡ|ợ|ù|ú|ủ|ũ|ụ|ừ|ứ|ử|ữ|ự|ỳ|ý|ỷ|ỹ|y
        lambda x: dicchar[x.group()], txt)

bang_nguyen_am = [['a', 'à', 'á', 'ả', 'ã', 'ạ', 'a'],
                  ['ă', 'ằ', 'ắ', 'ẳ', 'ẵ', 'ặ', 'aw'],
                  ['â', 'ầ', 'ấ', 'ẩ', 'ẫ', 'ậ', 'aa'],
                  ['e', 'è', 'é', 'ẻ', 'ẽ', 'ẹ', 'e'],
                  ['ê', 'ề', 'ế', 'ể', 'ễ', 'ệ', 'ee'],
                  ['i', 'ì', 'í', 'ỉ', 'ĩ', 'ị', 'i'],
                  ['o', 'ò', 'ó', 'ỏ', 'õ', 'ọ', 'o'],
                  ['ô', 'ồ', 'ố', 'ổ', 'ỗ', 'ộ', 'oo'],
                  ['ơ', 'ờ', 'ớ', 'ở', 'ỡ', 'ợ', 'ow'],
                  ['u', 'ù', 'ú', 'ủ', 'ũ', 'ụ', 'u'],
                  ['ư', 'ừ', 'ứ', 'ử', 'ữ', 'ự', 'uw'],
                  ['y', 'ỳ', 'ý', 'ỷ', 'ỹ', 'ỵ', 'y']]
bang_ky_tu_dau = ['', 'f', 's', 'r', 'x', 'j']

nguyen_am_to_ids = {}

for i in range(len(bang_nguyen_am)):
    for j in range(len(bang_nguyen_am[i]) - 1):
        nguyen_am_to_ids[bang_nguyen_am[i][j]] = (i, j)
```

Hình 3: Chuyển đổi unicode

3. Chuẩn hóa kiểu gõ dấu

```
def chuan_hoa_dau_tu_tieng_viet(word):
    if not is_valid_vietnam_word(word):
        return word

    chars = list(word)
    dau_cau = 0
    nguyen_am_index = []
    qu_or_gi = False
    for index, char in enumerate(chars):
        x, y = nguyen_am_to_ids.get(char, (-1, -1))
        if x == -1:
            continue
        elif x == 9:  # check qu
            if index != 0 and chars[index - 1] == 'q':
                chars[index] = 'u'
                qu_or_gi = True
        elif x == 5:  # check gi
            if index != 0 and chars[index - 1] == 'g':
                chars[index] = 'i'
                qu_or_gi = True
        if y != 0:
            dau_cau = y
            chars[index] = bang_nguyen_am[x][0]
        if not qu_or_gi or index != 1:
            nguyen_am_index.append(index)
```

```
    if len(nguyen_am_index) < 2:
        if qu_or_gi:
            if len(chars) == 2:
                x, y = nguyen_am_to_ids.get(chars[1])
                chars[1] = bang_nguyen_am[x][dau_cau]
            else:
                x, y = nguyen_am_to_ids.get(chars[2], (-1, -1))
                if x != -1:
                    chars[2] = bang_nguyen_am[x][dau_cau]
                else:
                    chars[1] = bang_nguyen_am[5][dau_cau] if chars[1] == 'i' else bang_nguyen_am[9][dau_cau]
            return ''.join(chars)
        return word

    for index in nguyen_am_index:
        x, y = nguyen_am_to_ids[chars[index]]
        if x == 4 or x == 8:  # ê, ơ
            chars[index] = bang_nguyen_am[x][dau_cau]
            return ''.join(chars)

    if len(nguyen_am_index) == 2:
        if nguyen_am_index[-1] == len(chars) - 1:
            x, y = nguyen_am_to_ids[chars[nguyen_am_index[0]]]
            chars[nguyen_am_index[0]] = bang_nguyen_am[x][dau_cau]
        else:
            x, y = nguyen_am_to_ids[chars[nguyen_am_index[1]]]
            chars[nguyen_am_index[1]] = bang_nguyen_am[x][dau_cau]
    else:
        x, y = nguyen_am_to_ids[chars[nguyen_am_index[1]]]
        chars[nguyen_am_index[1]] = bang_nguyen_am[x][dau_cau]
    return ''.join(chars)
```

Hình 4: Chuẩn hóa kiểu gõ dấu

4. *Tách từ Tiếng Việt*

- Trong Tiếng Việt có rất nhiều từ đơn và từ ghép nên chúng ta cần huấn luyện cho mô hình biết đâu là từ đơn và đâu là từ ghép
- Hiện nay có khá nhiều thư viện mã nguồn mở cho bài toán này, trong mô hình lần này em áp dụng đó là UNDERTHESEA

```
from underthesea import word_tokenize
sentence = 'Tôi là sinh viên lớp khoa học máy tính 03'

word_tokenize(sentence)

word_tokenize(sentence, format="text")
```

Hình 5: Tách từ

5. *Đưa về chữ viết thường*

- Đưa về chữ viết thường giúp giảm số lượng đặc trưng và tăng độ chính xác cho mô hình
- Trong python việc đưa về chữ viết thường khá đơn giản chúng ta chỉ cần gọi hàm lower

```
document = document.lower()
```

6. *Xóa các ký tự không cần thiết*

- Để giúp cho mô hình được tốt việc xóa các ký tự không cần thiết là việc quan trọng, nó giúp làm giảm số chiều đặc trưng và tăng tốc độ học và xử lý

```
document = re.sub(r'[^\s\wáàảãạăắằẳẵặâấầẩẫậéèẻẽẹêếềểễệóòỏõọôốồổỗộơớờởỡợíìỉĩịúùủũụưứừửữựýỳỷỹỵđ_]',' ',document)
```

7. *Xóa khoảng trắng dư thừa*

- Trong dữ liệu khi mà lấy trên mạng. không tránh khỏi sẽ có những khoảng trắng dư thừa. Việc xóa các khoảng trắng dư thừa là cần thiết để mô hình được chính xác hơn

```
document = re.sub(r'\s+', ' ', document).strip()
```

8. *Xóa HTML code trong dữ liệu*

- Dữ liệu sau khi được tổng hợp từ các website đôi khi còn dính một số đoạn mã HTML, việc xóa các đoạn mã này sẽ giúp cho mô hình đạt được độ chính xác hơn

```
def remove_html(txt):
    return re.sub(r'<[^>]*>', '', txt)
```

9. *Tổng hợp các bước tiền xử lý*

```python
def text_preprocess(document):
    # xóa html code
    document = remove_html(document)
    # chuẩn hóa unicode
    document = convert_unicode(document)
    # chuẩn hóa cách gõ dấu tiếng Việt
    document = chuan_hoa_dau_cau_tieng_viet(document)
    # tách từ
    document = word_tokenize(document, format="text")
    # đưa về lower
    document = document.lower()
    # xóa các ký tự không cần thiết
    document = re.sub(r'[^\s\wáàảãạăắằẳẵặâấầẩẫậéèẻẽẹêếềểễệóòỏõọôốồổỗộơớờởỡợíìỉĩịúùủũụưứừửữựýỳỷỹỵđ_]',' ',document)
    # xóa khoảng trắng thừa
    document = re.sub(r'\s+', ' ', document).strip()
    return document
```

Hình 6: Tổng hợp tiền xử lý

10. Kết quả của văn bản sau các bước tiền xử lý

- Đầu vào:

  Với chiến thắng rất bất ngờ này, Nhật Bản không chỉ lọt vào vòng 1/8 mà còn giành luôn ngôi đầu bảng E. Trong khi đó, Tây Ban Nha do hơn Đức về hiệu số đã giành vé đi tiếp với ngôi nhì bảng E. Như vậy, Nhật Bản có lần thứ 2 liên tiếp vượt qua vòng bảng tại một kỳ World Cup.

  Ở giải đấu lần này, Nhật gây sốc thực sự khi đánh bại 2 nhà cựu vô địch thế giới là Đức và Tây Ban Nha cùng với tỷ số 2-1. Phía trước họ sẽ là á quân World Cup 2018, đội tuyển Croatia. Đây là thử thách khó cho Nhật Bản nhưng với những gì họ đã và đang làm được, có cơ sở chờ đợi Asano và các đồng đội sẽ tiếp tục tạo nên bất ngờ ở vòng knock-out sắp tới.

  Mặc dù chỉ đứng thứ 2 nhưng thầy trò Luis Enrique có thể hài lòng với kết quả thu lại bởi họ đã tránh được đối thủ được đánh giá khó nhằn là Croatia (nhì bảng F) ở vòng 1/8. Thay vào đó, La Roja sẽ tranh vé vào tứ kết với Morocco, đội bóng đã giành ngôi nhất bảng F.

  Đây là lần đầu tiên Morocco vượt qua vòng bảng World Cup sau 36 năm, kể từ năm 1986. Đặt cạnh Tây Ban Nha, Morocco hoàn toàn lép vế. Nếu chơi đúng sức, Tây Ban Nha có lẽ sẽ vượt qua đại diện châu Phi để tiến xa hơn nữa ở giải đấu tại Qatar."""

- Kết quả:

  với chiến_thắng rất bất_ngờ này nhật bản không chỉ lọt vào vòng 18 mà_còn giành luôn ngôi đầu_bảng e trong khi đó tây ban nha do hơn đức về hiệu_số đã giành vé đi tiếp với ngôi nhì bảng e như vậy nhật bản có lần thứ 2 liên_tiế

p vượt qua vòng bảng tại một kỳ world cup ở giải đấu lần này nhật gây sốc th ực_sự khi đánh_bại 2 nhà cựu vô_địch thế_giới là đức và tây ban nha cùng v ới tỷ_số 2 1 phía trước họ sẽ là á_quân world cup 2018 đội_tuyển croatia đây là thử_thách khó cho nhật bản nhưng với những gì họ đã và đang làm được có cơ_sở chờ_đợi asano và các đồng_đội sẽ tiếp_tục tạo nên bất_ngờ ở vòng knock out sắp tới mặc_dù chỉ đứng thứ 2 nhưng thầy_trò luis enrique có_thể hài_lòng với kết_quả thu lại bởi họ đã tránh được đối_thủ được đánh_giá kh ó_nhằn là croatia nhì bảng f ở vòng 18 thay vào đó la roja sẽ tranh vé vào tứ_ kết với morocco đội bóng đã giành ngôi nhất bảng f đây là lần đầu_tiên moro cco vượt qua vòng bảng world cup sau 36 năm kể từ năm 1986 đặt cạnh tây ban nha morocco hoàn_toàn lép_vế nếu chơi đúng sức tây ban nha có_lẽ sẽ v ượt qua đại_diện châu_phi để tiến xa hơn nữa ở giải đấu tại qatar

```
document = """
Với chiến thắng rất bất ngờ này, Nhật Bản không chỉ lọt vào vòng 1/8 mà còn giành luôn ngôi đầu bảng E. Trong khi đó, Tây Ban Nha

Ở giải đấu lần này, Nhật gây sốc thực sự khi đánh bại 2 nhà cựu vô địch thế giới là Đức và Tây Ban Nha cùng với tỷ số 2-1. Phía t


Mặc dù chỉ đứng thứ 2 nhưng thầy trò Luis Enrique có thể hài lòng với kết quả thu lại bởi họ đã tránh được đối thủ được đánh giá

Đây là lần đầu tiên Morocco vượt qua vòng bảng World Cup sau 36 năm, kể từ năm 1986. Đặt cạnh Tây Ban Nha, Morocco hoàn toàn lép
```

In [178]:
```
document = text_preprocess(document)
print(document)
```

```
với chiến_thắng rất bất_ngờ này nhật bản không chỉ lọt vào vòng 18 mà_còn giành luôn ngôi đầu_bảng e trong khi đó tây ban nha d
o hơn đức về hiệu_số đã giành vé đi tiếp với ngôi nhì bảng e như vậy nhật bản có lần thứ 2 liên_tiếp vượt qua vòng bảng tại một
kỳ world cup ở giải đấu lần này nhật gây sốc thực_sự khi đánh_bại 2 nhà cựu vô_địch thế_giới là đức và tây ban nha cùng với tỷ_
số 2 1 phía trước họ sẽ là á_quân world cup 2018 đội_tuyển croatia đây là thử_thách khó cho nhật bản nhưng với những gì họ đã v
à đang làm được có cơ_sở chờ_đợi asano và các đồng_đội sẽ tiếp_tục tạo nên bất_ngờ ở vòng knock out sắp tới mặc_dù chỉ đứng thứ
2 nhưng thầy_trò luis enrique có_thể hài_lòng với kết_quả thu lại bởi họ đã tránh được đối_thủ được đánh_giá khó_nhằn là croati
a nhì bảng f ở vòng 18 thay vào đó la roja sẽ tranh vé vào tứ_kết với morocco đội bóng đã giành ngôi nhất bảng f đây là lần đầu
_tiên morocco vượt qua vòng bảng world cup sau 36 năm kể từ năm 1986 đặt cạnh tây ban nha morocco hoàn_toàn lép_vế nếu chơi đún
g sức tây ban nha có_lẽ sẽ vượt qua đại_diện châu_phi để tiến xa hơn nữa ở giải đấu tại qatar
```

III. Thực hành

1. *Tải tập dữ liệu sau khi thu thập và tiền xử lý*

```
import pandas as pd
df = pd.read_csv('database.txt', header = None);
```

```
df
```

	0
0	__label__giải_trí vợ_chồng ốc_thanh_vân ngày_c...
1	__label__nhịp_sống ưu_đãi lớn khi mua galaxy s...
2	__label__thể_thao vụ berezovsky abramovich khi...
3	__label__thể_thao 5 bàn_thắng nhanh nhất europ...
4	__label__thể_thao suarez và neymar chửi trọng_...
...	...
996	__label__giải_trí yanbi bị chỉ_trích khi tuyển...
997	__label__thời_sự rạch mặt cướp iphone 5 ở hà_n...
998	__label__thời_sự đang đốt rác cụ bà 83 tuổi ng...
999	__label__thể_thao đội_bóng bầu hiển chê đồng_đ...
1000	__label__kinh_doanh hải_sản tươi ngon nhờ ngâm...

1001 rows × 1 columns

Hình 7: Tải file sau khi tiền xử lý

- Dữ liệu mà em đã chuẩn bị bao gồm 1000 văn bản Tiếng Việt gồm 10 chủ đề, được lưu vào file txt. Các chủ đề được ghi dưới 1 nhãn label

2. *Thống kê các word xuất hiện ở các nhãn*

```
vocab = {}
label_vocab = {}
for line in open("database.txt", encoding='utf-8'):
    try:
        words = line.split()
        # lưu ý từ đầu tiên là nhãn
        label = words[0]
        if label not in label_vocab:
            label_vocab[label] = {}
        for word in words[1:]:
            label_vocab[label][word] = label_vocab[label].get(word, 0) + 1
            if word not in vocab:
                vocab[word] = set()
            vocab[word].add(label)
    except IndexError:
        pass
```

```
count = {}
for word in vocab:
    if len(vocab[word]) == total_label:
        count[word] = min([label_vocab[x][word] for x in label_vocab])

sorted_count = sorted(count, key=count.get, reverse=True)
for word in sorted_count[:100]:
    print(word, count[word])
```

```
là 77
các 61
có 60
của 58
được 50
và 48
với 47
trong 45
```

Hình 8: Thống kê word

3. *Loại bỏ các stopword*

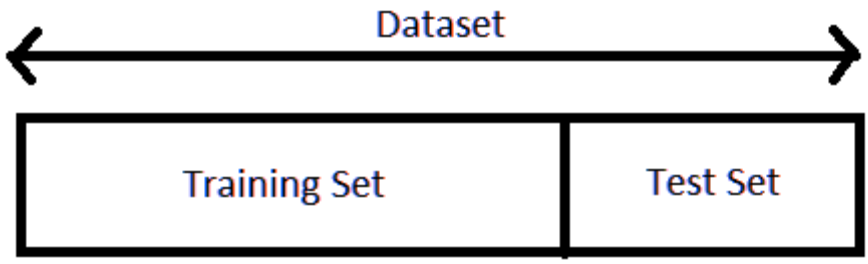
- Từ dừng (stop word) là một từ thường được sử dụng phổ biến trong ngôn ngữ (chẳng hạn như "là", "và", "của", "quá",….). Từ dừng là những từ không bổ sung nhiều ý nghĩa cho một câu. Chúng có thể được bỏ qua một cách an toàn mà không làm mất đi ý nghĩa của câu.
- Các từ dừng (stop word) thường được xóa khỏi văn bản trước khi huấn luyện (training) mô hình học sâu (deep learning model) và học máy (machine learning model) vì các từ dừng (stop word) xuất hiện rất nhiều, do đó cung cấp ít hoặc không có thông tin duy nhất có thể được sử dụng để phân loại (classification) hoặc phân cụm (clustering). Khi loại bỏ các từ dừng (stop word) , kích thước tập dữ liệu (dataset) giảm và thời gian huấn luyện mô hình cũng giảm mà không ảnh hưởng lớn đến độ chính xác (accuracy) của mô hình.

```
stopword = set()

def remove_stopwords(line):
    words = []
    for word in line.strip().split():
        if word not in stopword:
            words.append(word)
    return ' '.join(words)

with open('database.prep', 'w', encoding='utf-8') as fp, open('database.txt', encoding='utf-8') as f:
    for line in f:
        line = remove_stopwords(line)
        fp.write(line + '\n')
```

Hình 9: Loại bỏ từ dừng

4. *Chia tệp dữ liệu train test*

- Khi làm việc với dữ liệu, một thuật toán học máy làm việc theo 2 giai đoạn là huấn luyện và kiểm thử
- Trong bài toán phân loại văn bản này, em chia 80% để huấn luyện và 20% còn lại cho việc kiểm thử
- Sau đó lưu 2 tệp này lại và giữ nguyên tệp để so sánh các mô hình cho công bằng




```python
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
test_percent = 0.2

text = []
label = []

with open('database.prep', encoding='utf-8') as f:
    for line in f:
        words = line.strip().split()
        label.append(words[0])
        text.append(' '.join(words[1:]))

X_train, X_test, y_train, y_test = train_test_split(text, label, test_size=test_percent, random_state=42)

with open('train.txt', 'w', encoding='utf-8') as fp:
    for x, y in zip(X_train, y_train):
        fp.write('{} {}\n'.format(y, x))

with open('test.txt', 'w', encoding='utf-8') as fp:
    for x, y in zip(X_test, y_test):
        fp.write('{} {}\n'.format(y, x))

label_encoder = LabelEncoder()
label_encoder.fit(y_train)
print(list(label_encoder.classes_), '\n')
y_train = label_encoder.transform(y_train)
y_test = label_encoder.transform(y_test)

print(X_train[0], y_train[0], '\n')
print(X_test[0], y_test[0])
```

Hình 10: Chia tệp train/test

800 __label__sống_trẻ honda vn tặng người_dân sơn_la 1 000 mũ_bảo_hiểm đạt chuẩn sự_kiện zing vn ngày 5 8 nhằm hưởng_ứng chương_trình hỗ_trợ mũ_bảo_hiểm trọn nghĩa đồng_bào ấm tình cha_mẹ ủy_ban atgt quốc_gia và honda việt_nam trao_tặng 1 000 mũ_bảo_hiểm tại sơn_la tham_dự chương_trình có sự hiện_diện của bà tòng_thị_phóng ủy_viên bct phó chủ_tịch quốc_hội ông trương_quang_nghĩa ủy_viên bch tw đảng bộ_trưởng bộ gtvt phó chủ_tịch thường_trực ủy_ban atgt quốc_gia ông khuất_việt_hùng phó chủ_tịch chuyên_trách ủy_ban atgt quốc_gia lãnh_đạo tỉnh sơn_la cùng đại_diện công_ty honda việt_nam đây cũng là hoạt_động nằm trong chiến_dịch trao_tặng 20 000 mũ_bảo_hiểm mang tên cùng honda chắp_cánh tương_lai trong năm 2016 của honda việt_nam chương_trình nhằm tăng tỷ_lệ người tham_gia giao_thông đội mũ_bảo_hiểm đạt chuẩn chất_lượng góp_phần nâng_cao ý_thức khi tham_gia giao_thông của người_dân cả nước tiếp nối thành_công của chương_trình trao_tặng 30 000 mũ_bảo_hiểm cho nhân_dân tại 63 tỉnh thành trên cả nước trong năm 2015 honda việt_nam tiếp_tục triển_khai chiến_dịch này với chủ_đề cùng honda chắp_cánh tương_lai trong năm 2016 để nhân rộng hơn_nữa số_lượng người được đội mũ_bảo_hiểm đạt chuẩn tăng tính an_toàn cho người sử_dụng môtô xe_gắn_máy tại việt_nam với hoạt_động trao_tặng mũ_bảo_hiểm tại sơn_la trong năm 2016 đã có gần 5 000 mũ_bảo_hiểm được ủy_ban atgt quốc_gia và honda việt_nam phối_hợp trao_tặng cho người_dân trên cả nước ông vũ_quang_tâm trao_tặng 1 000 mũ_bảo_hiểm đạt chuẩn cho nhân_dân và trẻ_em sơn_la tại buổi lễ ông vũ_quang_tâm phó tổng_giám_đốc thứ nhất công_ty honda việt_nam đã trao_tặng biển tượng_trưng 1 000 mũ_bảo_hiểm cho ông trịnh_xuân_hùng giám_đốc sở gtvt phó trưởng_ban atgt tỉnh sơn_la lễ trao_tặng có sự chứng_kiến của đồng_chí khuất_việt_hùng phó chủ_tịch chuyên_trách ủy_ban atgt quốc_gia cũng tại buổi lễ bà tòng_thị_phóng ủy_viên bct phó chủ_tịch quốc_hội ông trương_quang_nghĩa ủy_viên bch tw đảng bộ_trưởng bộ gtvt phó chủ_tịch thường_trực ủy_ban atgt quốc_gia và ông hoàng_văn_chất bí_thư tỉnh_ủy chủ_tịch hđnd tỉnh sơn_la đã trao_tặng và đội mũ_bảo_hiểm tượng_trưng cho 10 học_sinh tiểu_học trên địa_bàn tỉnh bộ_trưởng bộ gtvt trương_quang_nghĩa đội mũ_bảo_hiểm cho các em nhỏ tỉnh sơn_la trao_tặng mũ_bảo_hiểm cho trẻ_em và nhân_dân tỉnh sơn_la phát_biểu tại buổi lễ bà tòng_thị_phóng đã ghi_nhận sự đóng_góp chung tay vào hoạt_động atgt tại việt_nam của các đơn_vị đồng_hành trong đó có công_ty honda việt_nam cũng trong buổi lễ bên cạnh niềm_vui khi nhận được những phần quà ý_nghĩa các em nhỏ và người_dân địa_phương tỏ ra hào_hứng khi được hướng_dẫn đội mũ_bảo_hiểm đúng cách người_dân cũng mong_muốn sẽ có thêm nhiều chương_trình đào_tạo kiến_thức về atgt trên địa_bàn đặc_biệt là tại những nơi có địa_hình phức_tạp và đi_lại khó_khăn sau chương_trình này ủy_ban atgt quốc_gia sẽ tiếp_tục cùng công_ty honda việt_nam triển_khai các hoạt_động trao_tặng mũ_bảo_hiểm cho đồng_bào các dân_tộc tây_bắc và trẻ_em trên cả nước hoạt_động nhằm tăng tỷ_lệ người tham_gia giao_thông đội mũ_bảo_hiểm đạt chuẩn đặc_biệt là trẻ_em giảm_thiểu thiệt_hại do hậu_quả tai_nạn giao_thông đối_với người tham_gia giao_thông bằng môtô xe_máy và xe_đạp điện
801

Hình 11: Train

test.txt

201 __label__thời_sự sập vận_thăng 3 người uống nước vỉa_hè trọng_thương thời_sự zing vn chiếc vận thăng sập xuống từ tòa nhà 6 tầng khối bê_tông đối_trọng cùng thanh đỡ rơi xuống quán nước bên dưới khiến 3 nam thanh_niên chấn_thương nặng vụ_việc xảy ra lúc 10h30 ngày 7 5 tại ngõ 178 thái_hà phường trung_liệt quận đống_đa hà_nội khi vận_thăng của công_trình đang thi_công lắp_đặt cửa_kính bất_ngờ rơi xuống ba thanh_niên chấn_thương nặng và được đưa đi cấp_cứu ngay sau đó khối bê_tông đối_trọng cùng thanh đỡ rơi xuống quán nước bên dưới khiến 3 nam thanh_niên chấn_thương ảnh h b anh thái một người_dân sinh_sống gần đó cho_biết thời_điểm trên anh nghe tiếng_động lớn ra hiện_trường anh thấy chiếc vận thăng rụng xuống toàn_bộ khối bê_tông đối_trọng và thanh đỡ rơi xuống quán trà đá trong ngõ chiếc vận thăng được cho là đã tuột vít chịu lực đứt_cáp và sập thẳng xuống từ tòa nhà bên trong ngõ 178 thái_hà khối bê_tông đối_trọng cùng thanh đỡ rơi xuống quán nước bên dưới khiến 3 nam thanh_niên chấn_thương nặng phải cấp_cứu lúc này trên vận_thăng không có công_nhân làm_việc chiếc vận thăng rơi xuống từ tòa nhà 6 tầng 1 tum ảnh quang_huy tại hiện_trường nhiều cành cây dây_điện và cáp viễn_thông bị kéo xuống đường khiến việc lưu_thông qua ngõ rất nguy_hiểm anh dũng 27 tuổi nhân_viên tiệm cắt tóc gần hiện_trường cho_biết cả ba nam thanh_niên gặp nạn đều là nhân_viên của tiệm và đều sinh năm 1991 vị_trí khối bê_tông đối_trọng cùng thanh đỡ rơi xuống trước đó là quán nước ảnh quang_huy sự_việc xảy ra thu_hút nhiều người hiếu_kỳ tập_trung lại xem khiến giao_thông qua đoạn 178 thái_hà ùn ứ lực_lượng công_an phường trung_liệt phải tiến_hành điều_tiết giao_thông lực_lượng chức_năng và công_an phường trung_liệt đã phong_tỏa hiện_trường chặn lối vào ngõ 178 từ phía thái_hà và lấy lời khai từ những công_nhân đang thi_công tại công_trình để làm rõ trách_nhiệm các bên liên_quan ngõ 178 thái_hà nơi xảy ra vụ_việc ảnh google_maps

Hình 12: Test

IV. Phương pháp phân loại Multinomial Naïve Bayes (MNB)

1. Cơ sở lý thuyết

- Phân loại Naive Bayes là tạo ra các giả thiết độc lập về các đặc trưng đầu vào và độc lập có điều kiện với mỗi một lớp đã cho. Sự độc lập của phân loại Naive Bayes chính là thể hiện của mô hình mạng tin cậy (belief network) trong trường hợp đặc biệt, và phân loại là chỉ dựa trên một nút cha duy nhất của mỗi một đặc trưng đầu vào. Mạng tin cậy này đề cập tới xác suất phân tán P(Y) đối với mỗi một đặc trưng đích Y và P(Xi |Y) đối với mỗi một đặc trưng đầu vào Xi . Với mỗi một đối tượng, dự đoán bằng cách tính toán dựa trên các xác suất điều kiện của các đặc trưng quan sát được cho mỗi đặc trưng đầu vào.
- Định lý Bayes: Giả sử A và B là hai sự kiện đã xảy ra. Xác suất có điều kiện A khi biết trước điều kiện B được cho bởi:
  P(A|B) = P(B|A).P(A)/P(B)
  + P(A): Xác suất của sự kiện A xảy ra.
  + P(B): Xác suất của sự kiện B xảy ra.
  + P(B|A): Xác suất (có điều kiện) của sự kiện B xảy ra, nếu biết rằng sự kiện A đã xảy ra.
  + P(A|B): Xác suất (có điều kiện) của sự kiện A xảy ra, nếu biết rằng sự kiện B đã xảy ra.
- Mô hình xác suất
  + Một cách trừu tượng, mô hình xác suất cho phân loại là một mô hình điều kiện ρ(C|F1 ,.., Fn) Trên một lớp biến C với số lượng nhỏ các đầu ra hoặc các lớp. Điều kiện trên một vài biến đặc trưng F1đến F2 . Vấn đề chính trong bài toán này là nếu số đặc trưng n là lớp hoặc một đặc trưng có thể có số lượng lớn các giá trị, thì một mô hình được tạo ra dựa trên các bảng xác suất là phù hợp trong điều kiện này. Lý thuyết Bayes có thể viết thành:

$$\rho(C|F_1,\ldots,F_n = \frac{\rho(C)\,\rho(F_1,\ldots,F_n|C)}{\rho(F_1,\ldots,F_n|C)}$$

  + Một cách mô tả đơn giản cho công thức trên như sau:

$$\text{Hậu nghiệm} = \frac{\text{nghiệm trước} \times \text{khả năng}}{\text{Bằng chứng}}$$

- Trên thực tế, chỉ cần quan tâm tới số các phân mảnh (fraction), bởi có một số đặc trưng không phụ thuộc vào C và các giá trị Fi đã cho, mô hình ρ(C|F1 ,.., Fn) có thể được viết lại như sau, sử dụng luật xích để lặp lại định nghĩa của xác suất điều kiện:

$$\begin{aligned}\rho(C,F_1,\ldots,F_n) &= \rho(C)\,\rho(F_1,\ldots,F_n|C)\\ &= \rho(C)\,\rho(F_1|C)\,\rho(F_2,\ldots,F_n|C,F_1)\\ &= \rho(C)\,\rho(F_1|C)\,\rho(F_2|C,F_1)\,\rho(F_3,\ldots,F_n|C,F_1,F_2)\\ &= \rho(C)\,\rho(F_1|C)\,\rho(F_2|C,F_1)\ldots\rho(F_n|C,F_1,F_2,F_3,\ldots F_{n-1})\end{aligned}$$

- Xây dựng phân lớp từ mô hình xác suất: Phân lớp Bayes kết hợp với luật quyết định tạo ra phân loại Naive Bayes. Một luật thông thường đưa ra giả thuyết về khả năng nhất hay còn được xem như là cực đại hóa xác suất hậu nghiệm (maximum a posteriori). Bộ phân loại Bayes là một hàm phân loại được định nghĩa:

$$classify(f_1,\ldots,f_n) = \operatorname{argmax} p(C=c)\prod_{i=1}^{n} p(F_i=f_i|(C=c)$$

2. *Xây dựng mô hình*

- Giảm chiều đặc trưng: Như ở phần trước em đã nêu một số cách giúp giảm đặc trưng như loại bỏ các stopword,….
- Xây dựng mô hình

```
import numpy as np
from sklearn.naive_bayes import MultinomialNB

X = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])
y = np.array([1, 2, 3])

clf = MultinomialNB()
clf.fit(X, y)
```

```
MultinomialNB()
```

3. *Đánh giá precition, recall và F-score*

- Đánh giá precition: Thể hiện sự chuẩn xác của việc phát hiện các điểm Positive. Số này càng cao thì model nhận các điểm Positive càng chuẩn.

$$Precision = \frac{TruePositive}{TruePositive + FalsePositive}$$

- Đánh giá recall: Thể hiện khả năng phát hiện tất cả các postivie, tỷ lệ này càng cao thì cho thấy khả năng bỏ sót các điểm Positive là thấp

$$Recall = \frac{TruePositive}{TruePositive + FalseNegative}$$

- Đánh giá F-score: Là số dung hòa Recall và Precision giúp ta có căn cứ để lựa chọn model. F1 càng cao càng tốt

$$F1 = 2\frac{Precision * Recall}{Precision + Recall}$$

4. *Đánh giá vào mô hình bằng precision, recall và F-score:*

```
# calculate precision, recall, and F-score
precision = metrics.precision_score(y_true, y_pred)
recall = metrics.recall_score(y_true, y_pred)
f1_score = metrics.f1_score(y_true, y_pred)

print('Precision:', precision)
print('Recall:', recall)
print('F-score:', f1_score)
```

```
Precision: 0.6666666666666666
Recall: 0.8
F-score: 0.7272727272727272
```

Hình 13: precision, recall và F-score

5. *Thực hành*

```
import pickle
import time
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.naive_bayes import MultinomialNB
from sklearn.pipeline import Pipeline

start_time = time.time()
text_clf = Pipeline([('vect', CountVectorizer(ngram_range=(1,1),
                                              max_df=0.8,
                                              max_features=None)),
                     ('tfidf', TfidfTransformer()),
                     ('clf', MultinomialNB())
                    ])
text_clf = text_clf.fit(X_train, y_train)

train_time = time.time() - start_time
print('MNB: ', train_time, 'seconds.')

pickle.dump(text_clf, open(os.path.join(MODEL_PATH, "naive_bayes.pkl"), 'wb'))
```

```
MNB:  0.5414910316467285 seconds.
```

Hình 14: Thực hành MNB

V. Phương pháp đánh trọng số TF-TDF

- TF-IDF là viết tắt của Term frequencey inverse document frequency. Nó có thể được xác định là mức độ liên quan của một từ trong chuỗi các đoạn text.

  - **Term frequency**: trong document d, frequency (tần số) biểu diễn số lần xuất hiện của từ t. Trọng số của từ xuất hiện trong document

$$tf(t,d) = \frac{\text{count of t in d}}{\text{number of words in d}}$$

  - **Document frequency** Ở đây chúng ta sẽ xét đến corpus (tập hợp của nhiều documents). Ở đây chúng ta quan tâm đến số lần xuất hiện của từ trong corpus.

$$df(t) = \text{occurence of t in documents}$$

  - **Inverse Document Frequency**
  - **Computation**: Tf-idf là một trong những metric tốt nhất để xác định độ quan trọng của từ trong một đoạn text (document) trong một corpus. tf-idf là hệ thống trọng số cái mà gán trọng số cho mỗi từ trong document dựa trên *term frequency (tf)* và *document frequency (idf)*. Từ có weight cao hơn sẽ có ý nghĩa nhiều hơn.

- Thông thường tf-idf weight chứa 2 thành phần:

  - Normalized term frequency (tf)
  - Inverse document frequency (idf)

$$\text{tf-idf}(t,d) = \text{tf}(t,d) * \text{idf}(t)$$

- Python giá trị tf-idf có thể được tính bằng cách sử dụng TfidfVectorizer() method trong scikit-learn.
- **TF-IDF** (term frequency-inverse document frequency) là trọng số của một từ trong một văn bản thông qua thống kê
- **TF (Term frequency)** - tần suất xuất hiện của một từ trong document.

$$tf(t,d) = \frac{\text{count of t in d}}{\text{number of words in d}}$$

- Nói cách khác *tf(t,d)* chính là tỉ số giữa số lần xuất hiện của từ t trong văn bản d so với độ dài của văn bản d.

- **IDF (invert document frequency)** dùng để đánh giá múc độ quan trọng của 1 từ trong băn bản. Khi tính tf mức độ quan trọng của các từ coi là như nhau. Tuy nhiên trong văn bản thường xuất hiện nhiều từ không quan trọng xuất hiện với tần suất cao:
  - Từ nối: và, hoặc,… (đối với tiếng Việt)
  - Giới từ: ở, trong, của, để…
  - Từ chỉ định: ấy, đó, nhỉ

➢ Do đó chúng ta cần giảm mức độ quan trọng của những từ đó bằng **IDF**

$$idf(t, D) = \log \frac{|D|}{|d \in D : t \in d|}$$

➢ **TF-IDF**

$$\text{TF-IDF}(t, d, D) = TF(t, d) \times IDF(t, D)$$

➢ Những từ có giá trị TF-IDF cao là những từ
  - Xuất hiện nhiều trong văn bản d (có *tf(t,d)* cao)
  - Xuất hiện nhiều trong các văn (có *idf(t, D)* cao)

➢ Chính điều này giúp lọc ra những từ phổ biến và giữ lại những giá trị cao (coi là từ khóa của văn bản).

VI. Phương pháp phân loại k-nearest Neighbour (KNN)

1. Tìm hiểu các độ đo khoảng cách

1.1. Khoảng cách Euclide:

Đây là một biến đo lường phổ biến nhất trong KNN. Khoảng cách Euclide giữa hai điểm được tính bằng căn bậc hai của tổng bình phương của sự khác biệt giữa các thuộc tính của hai điểm đó. Công thức tính khoảng cách Euclide giữa hai điểm a và b như sau:

$$d_{Euclide}(a,b) = \sqrt{\sum_{i=1}^{n}(a_i - b_i)^2}$$

Hình 15: Euclide

1.2. Khoảng cách cosine

là một khoảng cách đo lường dựa trên góc giữa hai vectơ. Nó được tính bằng cosin của góc giữa hai vectơ trong không gian nhiều chiều. Khoảng cách Cosine giữa hai điểm a và b được tính bằng cách chia tích vô hướng của hai vector của hai điểm đó cho tích độ dài của chúng. Công thức tính khoảng cách Cosine giữa hai điểm a và b như sau:

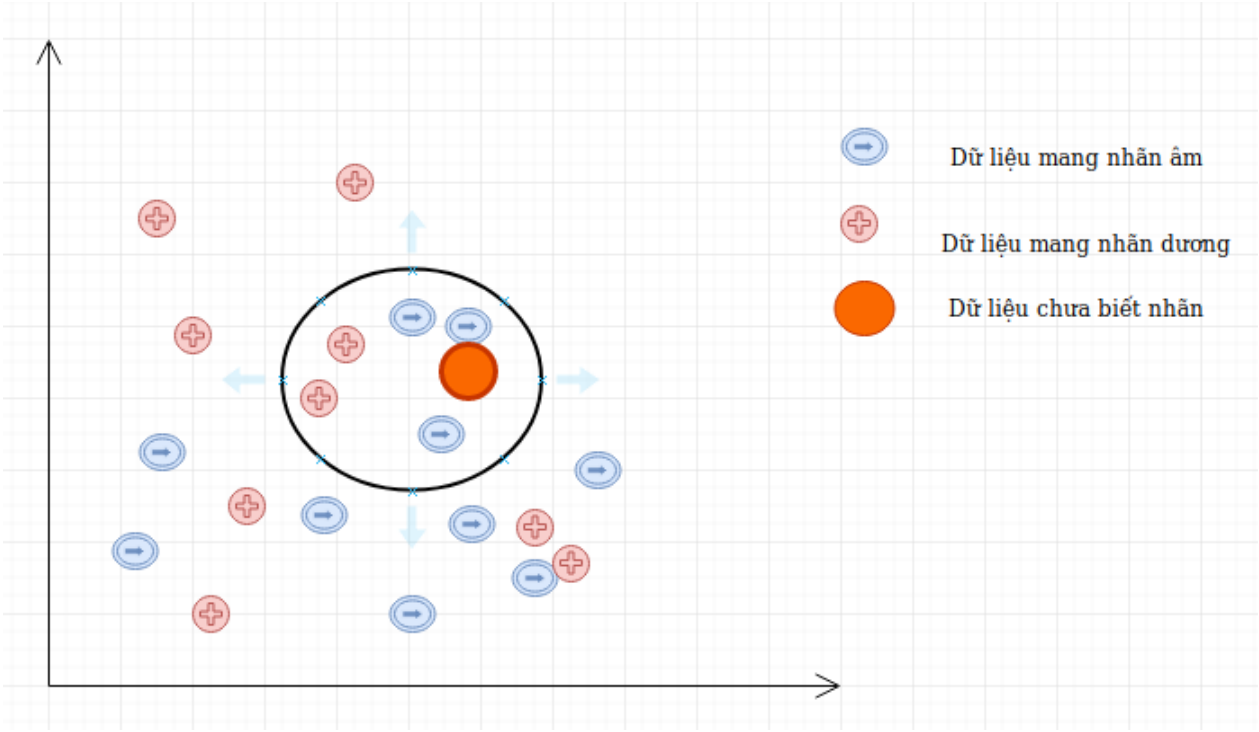
$$d_{Cosine}(a,b) = 1 - \frac{\sum_{i=1}^{n} a_i \times b_i}{\sqrt{\sum_{i=1}^{n} a_i^2} \times \sqrt{\sum_{i=1}^{n} b_i^2}}$$

Hình 16: Cosine

2. Phân loại văn bản bằng k-nearest Neighbour (KNN)

2.1. Cơ sở lý thuyết

- KNN (K-Nearest Neighbors) là một trong những thuật toán học có giám sát. Ý tưởng của thuật toán này là nó không học một điều gì từ tập dữ liệu học (nên KNN được xếp vào loại lazy learning), mọi tính toán được thực hiện khi nó cần dự đoán nhãn của dữ liệu mới. Lớp (nhãn) của một đối tượng dữ liệu mới có thể dự đoán từ các lớp (nhãn) của k hàng xóm gần nó nhất.



Hình 17: Sơ đồ KNN

2.2. *Các bước trong KNN*

- B1: Ta có D là tập dữ liệu được gán nhãn và A là dữ liệu chưa được phân loại
- B2: Đo khoảng cách (euclide, cosin, manhattan,…..) từ dữ liệu A đến tất cả các dữ liệu khác trong D
- B3: Chọn K là khoảng cách nhỏ nhất của các láng giềng
- B4: Kiểm tra danh sách các lớp có khoảng cách ngắn nhất và đếm số lượng của mỗi lớp xuất hiện
- B5: Lấy đúng lớp

2.3. *Ưu điểm của KNN*

- Thuật toán đơn giản, dễ dàng triển khai
- Độ phức tạp tính toán nhỏ
- Xử lý tốt với tập dữ liệu nhiễu

2.4. *Nhược điểm của KNN*

- Với K nhỏ dễ gặp kết quả không chính xác
- Cần thời gian để thực hiện do phải tính khoảng cách với tất cả các đối tượng trong tập dữ liệu
- Cần chuyển đổi kiểu dữ liệu thành các yếu tố định tính

2.5. *Thực hành*

- Dưới đây là phân loại văn bản bằng KNN với 5 láng giềng gần và bằng phương pháp tính khoảng cách cosine. Kết quả thu được sau khi huấn luyện như trên

```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.pipeline import Pipeline
from sklearn.feature_extraction.text import CountVectorizer, TfidfTransformer
from sklearn.metrics import accuracy_score

# Tạo pipeline gộp CountVectorizer, TfidfTransformer và KNN với độ đo cosine
text_clf = Pipeline([
    ('vect', CountVectorizer()),
    ('tfidf', TfidfTransformer()),
    ('clf', KNeighborsClassifier(n_neighbors=5, metric='cosine'))
])

# Huấn luyện mô hình trên tập huấn luyện
text_clf.fit(X_train, y_train)

# Đánh giá mô hình trên tập kiểm tra
y_pred = text_clf.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print('KNN cosine:', accuracy)

pickle.dump(text_clf, open(os.path.join(MODEL_PATH, "knn_cosine.pkl"), 'wb'))
```

```
KNN cosine: 0.88
```

Hình 18: Thực hành KNN_cosine

- Tiếp sau đây là phương pháp phân loại bằng KNN với 4 láng giềng gần và bằng phương pháp tính khoảng cách Euclide. Kết quả thu được như trên. Và cả 2 mô hình đều được xây dựng dựa trên cách đánh trọng số TF-IDF

```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.pipeline import Pipeline
from sklearn.feature_extraction.text import CountVectorizer, TfidfTransformer
from sklearn.metrics import accuracy_score

# Tạo pipeline gộp CountVectorizer, TfidfTransformer và KNN với độ đo euclide
text_clf = Pipeline([
    ('vect', CountVectorizer()),
    ('tfidf', TfidfTransformer()),
    ('clf', KNeighborsClassifier(n_neighbors=4, metric='euclidean'))
])

# Huấn luyện mô hình trên tập huấn luyện
text_clf.fit(X_train, y_train)

# Đánh giá mô hình trên tập kiểm tra
y_pred = text_clf.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print('KNN euclide:', accuracy)

pickle.dump(text_clf, open(os.path.join(MODEL_PATH, "knn_euclide.pkl"), 'wb'))
```

```
KNN euclide: 0.885
```

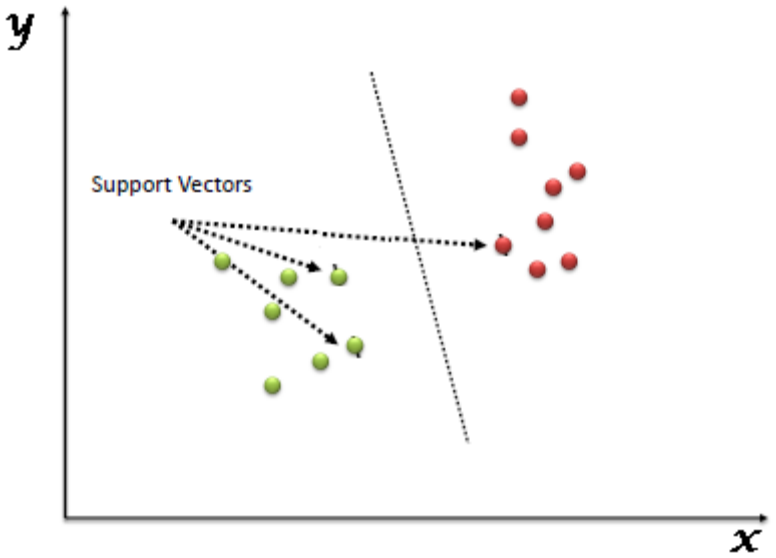
Hình 19: Thực hành KNN_euclide

- Do tệp dữ liệu không được lớn lắm nên kết quả của 2 phương pháp gần như không có quá nhiều sai lệch

VII. Phân loại văn bản bằng Support Vector Machines (SVM)

1. *Cơ sở lý thuyết SVM*

- **SVM** là một thuật toán giám sát, nó có thể sử dụng cho cả việc phân loại hoặc đệ quy. Tuy nhiên nó được sử dụng chủ yếu cho việc phân loại. Trong thuật toán này, chúng ta vẽ đồi thị dữ liệu là các điểm trong n chiều (ở đây n là số lượng các tính năng bạn có) với giá trị của mỗi tính năng sẽ là một phần liên kết. Sau đó chúng ta thực hiện tìm "đường bay" (*hyper-plane*) phân chia các lớp. Hyper-plane nó chỉ hiểu đơn giản là 1 đường thẳng có thể phân chia các lớp ra thành hai phần riêng biệt.



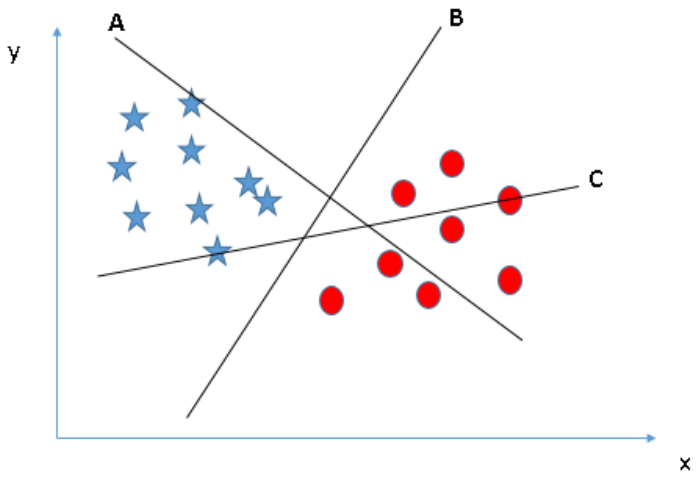
Hình 20: Sơ đồ SVM tổng quát

- *Support Vectors* hiểu một cách đơn giản là các đối tượng trên đồ thị tọa độ quan sát, *Support Vector Machine* là một biên giới để chia hai lớp tốt nhất.

2. *Cách làm việc của SVM*

- "Làm sao để vẽ-xác định đúng hyper-plane". Chúng ta sẽ theo các tiêu chí sau:

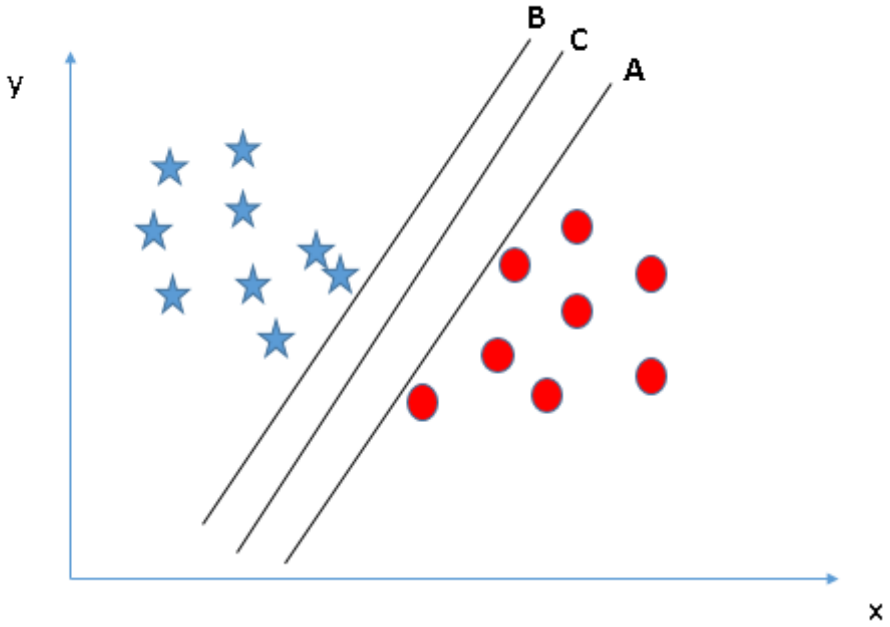
2.1. *Identify the right hyper-plane (Scenario-1):*



Hình 21: Scenario-1

Quy tắc số một để chọn 1 hyper-lane, chọn một hyper-plane để phân chia hai lớp tốt nhất. Trong ví dụ này chính là đường B.

2.2. Identify the right hyper-plane (Scenario-2):

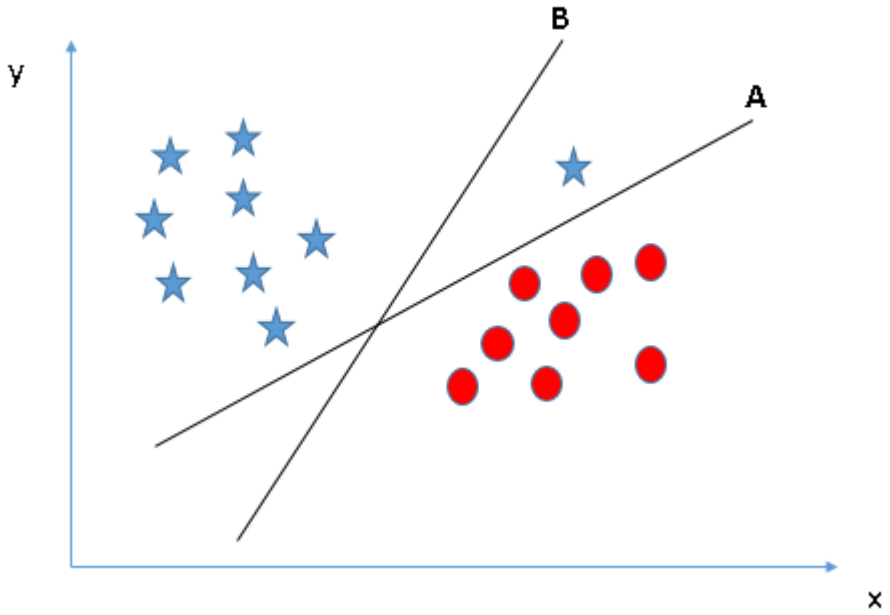


Hình 22: Scenario-2

Quy tắc thứ hai chính là xác định khoảng cách lớn nhất từ điểu gần nhất của một lớp nào đó đến đường hyper-plane. Khoảng cách này được gọi là "Margin", Hãy nhìn hình bên dưới, trong đấy có thể nhìn thấy khoảng cách margin lớn nhất đấy là

đường C. Cần nhớ nếu chọn lầm hyper-lane có margin thấp hơn thì sau này khi dữ liệu tăng lên thì sẽ sinh ra nguy cơ cao về việc xác định nhầm lớp cho dữ liệu.

2.3. Identify the right hyper-plane (Scenario-3):



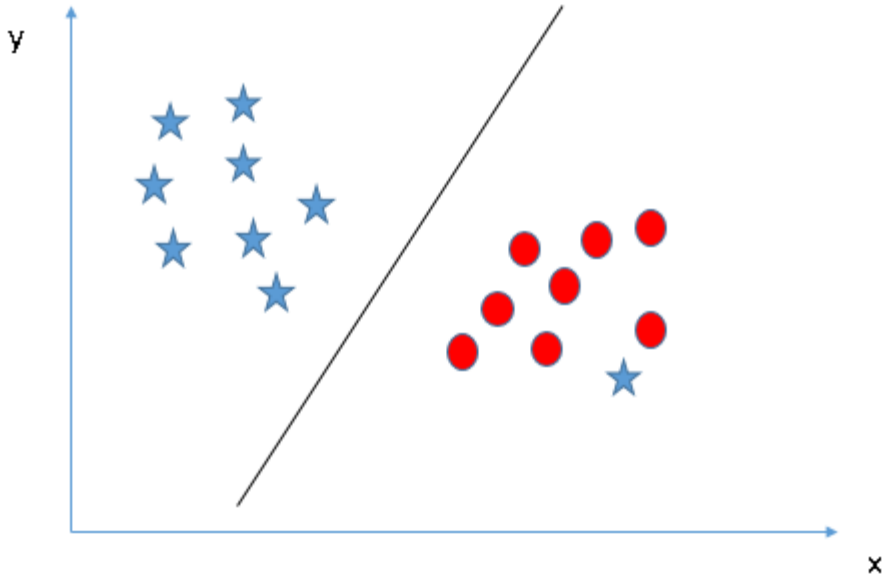
Hình 23: Scenario-3

Có thể có một vài người sẽ chọn đường B bởi vì nó có margin cao hơn đường A, nhưng đấy sẽ không đúng bởi vì nguyên tắc đầu tiên sẽ là nguyên tắc số 1, chúng ta cần chọn hyper-plane để phân chia các lớp thành riêng biệt. Vì vậy đường A mới là lựa chọn chính xác.

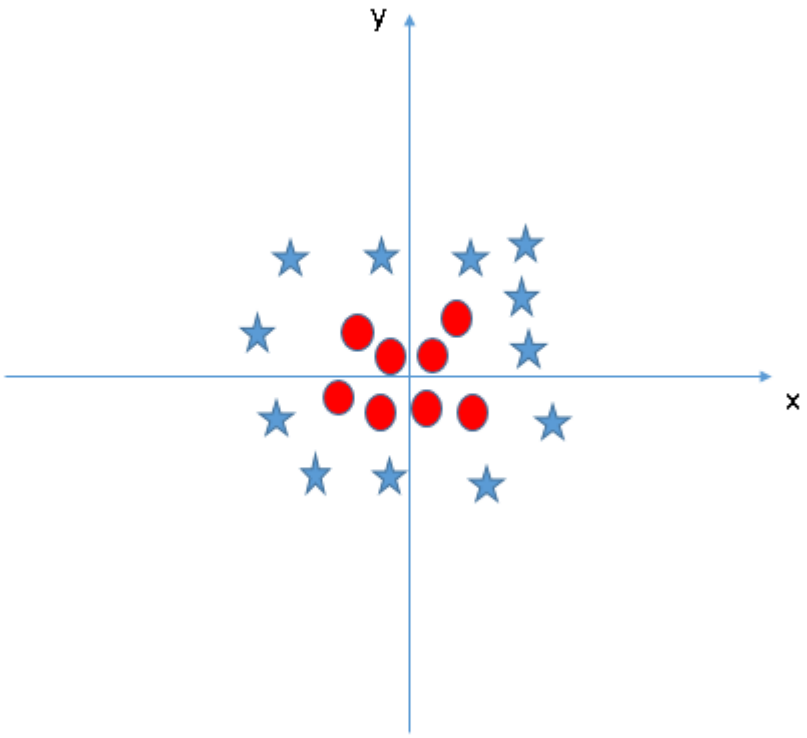
2.4. Can we classify two classes (Scenario-4)?

Hình 24: Scenario-4

Ở đây sẽ chấp nhận, một ngôi sao ở bên ngoài cuối được xem như một ngôi sao phía ngoài hơn, SVM có tính năng cho phép bỏ qua các ngoại lệ và tìm ra hyper-plane có biên giới tối đa . Do đó có thể nói, SVM có khả năng mạnh trong việc chấp nhận ngoại lệ.

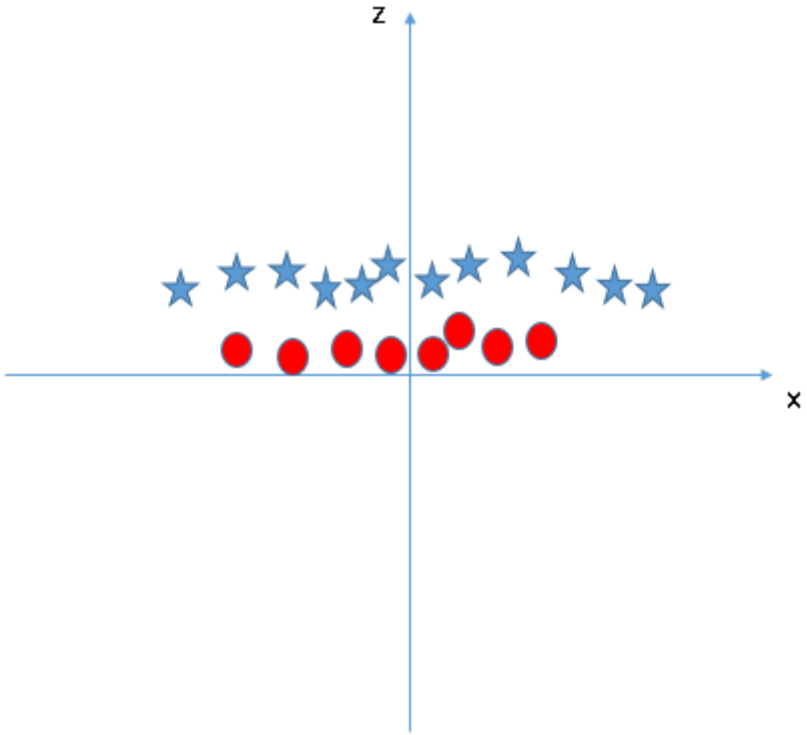


2.5. *Find the hyper-plane to segregate to classes (Scenario-5)*

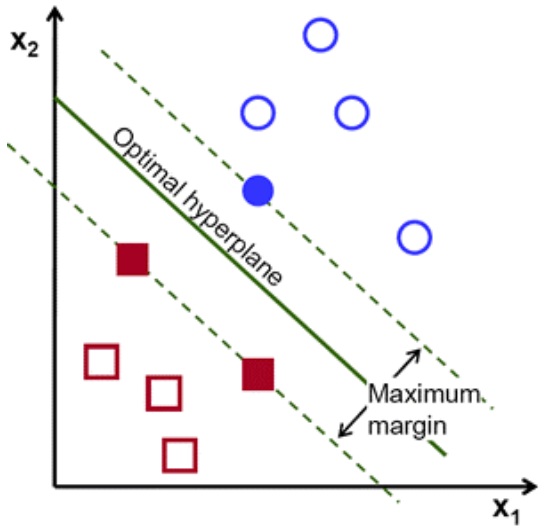


Hình 25: Scenario-5

SVM có thể giải quyết vấn đề này, Khá đơn giản, nó sẽ được giải quyết bằng việc thêm một tính năng, Ở đây chúng ta sẽ thêm tính năng z = x^2+ y^2. Bây giờ dữ liệu sẽ được biến đổi theo trục x và z như sau



3. *Margin trong SVM*



Hình 26: Margin to SVM

- Margin là khoảng cách giữa siêu phẳng đến 2 điểm dữ liệu gần nhất tương ứng với các phân lớp. Trong ví dụ quả táo quả lê đặt trên mặt bán, margin chính là khoảng cách giữa cây que và hai quả táo và lê gần nó nhất. Điều quan trọng ở đây đó là phương pháp SVM luôn cố gắng cực đại hóa margin này, từ đó thu được một siêu phẳng tạo khoảng cách xa nhất so với 2 quả táo và lê. Nhờ vậy, SVM có thể giảm thiểu việc phân lớp sai (misclassification) đối với điểm dữ liệu mới đưa vào

4. *Ưu điểm của SVM*

- Xử lý trên không gian số chiều cao: SVM là một công cụ tính toán hiệu quả trong không gian chiều cao, trong đó đặc biệt áp dụng cho các bài toán phân loại văn bản và phân tích quan điểm nơi chiều có thể cực kỳ lớn.
- Tiết kiệm bộ nhớ: Do chỉ có một tập hợp con của các điểm được sử dụng trong quá trình huấn luyện và ra quyết định thực tế cho các điểm dữ liệu mới nên chỉ có những điểm cần thiết mới được lưu trữ trong bộ nhớ khi ra quyết định.
- Tính linh hoạt - phân lớp thường là phi tuyến tính. Khả năng áp dụng Kernel mới cho phép linh động giữa các phương pháp tuyến tính và phi tuyến tính từ đó khiến cho hiệu suất phân loại lớn hơn.

5. *Nhược điểm của SVM*

- Bài toán số chiều cao: Trong trường hợp số lượng thuộc tính (p) của tập dữ liệu lớn hơn rất nhiều so với số lượng dữ liệu (n) thì SVM cho kết quả khá tồi.

- Chưa thể hiện rõ tính xác suất: Việc phân lớp của SVM chỉ là việc cố gắng tách các đối tượng vào hai lớp được phân tách bởi siêu phẳng SVM. Điều này chưa giải thích được xác suất xuất hiện của một thành viên trong một nhóm là như thế nào. Tuy nhiên hiệu quả của việc phân lớp có thể được xác định dựa vào khái niệm margin từ điểm dữ liệu mới đến siêu phẳng phân lớp mà chúng ta đã bàn luận ở trên.

6. *Thực hành SVM*

```
from sklearn.svm import SVC

start_time = time.time()
text_clf = Pipeline([('vect', CountVectorizer(ngram_range=(1,1),
                                              max_df=0.8,
                                              max_features=None)),
                     ('tfidf', TfidfTransformer()),
                     ('clf', SVC(gamma='scale'))
                    ])
text_clf = text_clf.fit(X_train, y_train)

train_time = time.time() - start_time
print('SVM: ', train_time, 'seconds.')

# Save model
pickle.dump(text_clf, open(os.path.join(MODEL_PATH, "svm.pkl"), 'wb'))
```

```
SVM:  18.983959913253784 seconds.
```

Hình 27: Thực hành SVM

VIII. Đánh giá các mô hình

1. *So sánh các mô hình*

```
import numpy as np
# Naive Bayes
model = pickle.load(open(os.path.join(MODEL_PATH,"naive_bayes.pkl"), 'rb'))
y_pred = model.predict(X_test)
print('Naive Bayes, Accuracy =', np.mean(y_pred == y_test))

# KNN
model = pickle.load(open(os.path.join(MODEL_PATH,"knn_cosine.pkl"), 'rb'))
y_pred = model.predict(X_test)
print('KNN cosine, Accuracy =', np.mean(y_pred == y_test))

# KNN
model = pickle.load(open(os.path.join(MODEL_PATH,"knn_euclide.pkl"), 'rb'))
y_pred = model.predict(X_test)
print('KNN euclide, Accuracy =', np.mean(y_pred == y_test))

# SVM
model = pickle.load(open(os.path.join(MODEL_PATH,"svm.pkl"), 'rb'))
y_pred = model.predict(X_test)
print('SVM, Accuracy =', np.mean(y_pred == y_test))

model = fasttext.load_model(os.path.join(MODEL_PATH,"fasttext.ftz"))
print_results(*model.test('test.txt'))
```

```
Naive Bayes, Accuracy = 0.5572139303482587
KNN cosine, Accuracy = 0.6766169154228856
KNN euclide, Accuracy = 0.6616915422885572
SVM, Accuracy = 0.7014925373134329
```

Hình 28: Đánh giá các mô hình

2. *Xem kết quả của từng nhãn trên từng mô hình*

2.1.Mô hình MNB

```
from sklearn.metrics import classification_report

nb_model = pickle.load(open(os.path.join(MODEL_PATH,"naive_bayes.pkl"), 'rb'))
y_pred = nb_model.predict(X_test)
print(classification_report(y_test, y_pred, target_names=list(label_encoder.classes_)))
```

	precision	recall	f1-score	support
__label__chính_trị_xã_hội	0.83	0.96	0.89	47
__label__công_nghệ	0.89	0.97	0.93	34
__label__giáo_dục	0.84	0.97	0.90	33
__label__khoa_học	1.00	0.67	0.81	46
__label__phim_ảnh	1.00	0.81	0.90	48
__label__pháp_luật	0.98	1.00	0.99	44
__label__thể_thao	0.97	1.00	0.99	36
__label__thời_trang	0.90	0.82	0.86	34
__label__âm_nhạc	0.78	1.00	0.88	36
__label__ẩm_thực	0.95	0.95	0.95	42
accuracy			0.91	400
macro avg	0.92	0.92	0.91	400
weighted avg	0.92	0.91	0.91	400

Hình 29: Đánh giá MNB

2.2.Mô hình KNN_euclide

```
from sklearn.metrics import classification_report

nb_model = pickle.load(open(os.path.join(MODEL_PATH,"knn_euclide.pkl"), 'rb'))
y_pred = nb_model.predict(X_test)
print(classification_report(y_test, y_pred, target_names=list(label_encoder.classes_)))
```

	precision	recall	f1-score	support
__label__chính_trị_xã_hội	0.79	0.89	0.84	47
__label__công_nghệ	0.94	0.91	0.93	34
__label__giáo_dục	0.86	0.91	0.88	33
__label__khoa_học	0.92	0.72	0.80	46
__label__phim_ảnh	0.95	0.81	0.88	48
__label__pháp_luật	0.93	0.95	0.94	44
__label__thể_thao	1.00	0.94	0.97	36
__label__thời_trang	0.85	0.85	0.85	34
__label__âm_nhạc	0.74	0.94	0.83	36
__label__ẩm_thực	0.93	0.95	0.94	42
accuracy			0.89	400
macro avg	0.89	0.89	0.89	400
weighted avg	0.89	0.89	0.89	400

Hình 30: Đánh giá KNN

2.3.Mô hình KNN_cosine

```
from sklearn.metrics import classification_report

nb_model = pickle.load(open(os.path.join(MODEL_PATH,"knn_cosine.pkl"), 'rb'))
y_pred = nb_model.predict(X_test)
print(classification_report(y_test, y_pred, target_names=list(label_encoder.classes_)
```

```
                          precision    recall  f1-score   support

__label__chính_trị_xã_hội       0.80      0.87      0.84        47
       __label__công_nghệ       0.91      0.91      0.91        34
        __label__giáo_dục       0.81      0.91      0.86        33
        __label__khoa_học       0.89      0.72      0.80        46
        __label__phim_ảnh       0.93      0.81      0.87        48
       __label__pháp_luật       0.93      0.95      0.94        44
        __label__thể_thao       1.00      0.94      0.97        36
      __label__thời_trang       0.84      0.79      0.82        34
         __label__âm_nhạc       0.77      0.94      0.85        36
         __label__ẩm_thực       0.93      0.98      0.95        42

                 accuracy                           0.88       400
                macro avg       0.88      0.88      0.88       400
             weighted avg       0.88      0.88      0.88       400
```

Hình 31: Đánh giá KNN

2.4.Mô hình SVM

```
from sklearn.metrics import classification_report

nb_model = pickle.load(open(os.path.join(MODEL_PATH,"svm.pkl"), 'rb'))
y_pred = nb_model.predict(X_test)
print(classification_report(y_test, y_pred, target_names=list(label_encoder.classes_)))
```

```
                          precision    recall  f1-score   support

__label__chính_trị_xã_hội       0.88      0.94      0.91        47
       __label__công_nghệ       0.94      0.94      0.94        34
        __label__giáo_dục       0.88      0.91      0.90        33
        __label__khoa_học       0.81      0.85      0.83        46
        __label__phim_ảnh       1.00      0.90      0.95        48
       __label__pháp_luật       1.00      1.00      1.00        44
        __label__thể_thao       1.00      0.89      0.94        36
      __label__thời_trang       0.94      0.94      0.94        34
         __label__âm_nhạc       0.90      1.00      0.95        36
         __label__ẩm_thực       0.98      0.95      0.96        42

                 accuracy                           0.93       400
                macro avg       0.93      0.93      0.93       400
             weighted avg       0.93      0.93      0.93       400
```

Hình 32: Đánh giá SVM

3. Nhận xét

- Sau khi làm xong, em đưa ra kết luận mô hình MNB và SVM đưa ra kết quả chính xác vượt trội hơn mô hình KNN với % chính xác như trên
- Một số label có độ đo đang còn ở tầm khoảng hơn 80% thậm chí là chỉ hơn 70% có lẽ là do độ dài ngắn của dữ liệu chuẩn bị ban đầu. Em sẽ cố gắng hoàn thiện hơn
- Dữ liệu sau khi em chuẩn bị lại có phần trực quan hơn nên đưa ra kết quả khá chính xác. Và sau đây là phần demo lại cho bài tập lớn lần này.

IX. Kết quả demo thực hiện

```
document = "Đại học bách khoa hà nội"

document = text_preprocess(document)
document = remove_stopwords(document)

label = nb_model.predict([document])
print('Predict label:', label_encoder.inverse_transform(label))
```

```
Predict label: ['__label__giáo_dục']
```

```
document = "Project1"

document = text_preprocess(document)
document = remove_stopwords(document)

label = nb_model.predict([document])
print('Predict label:', label_encoder.inverse_transform(label))
```

```
Predict label: ['__label__giáo_dục']
```

```
document = "Chủ tịch nước"

document = text_preprocess(document)
document = remove_stopwords(document)

label = nb_model.predict([document])
print('Predict label:', label_encoder.inverse_transform(label))
```

```
Predict label: ['__label__chính_trị_xã_hội']
```

```
document = "Thần điêu đại hiệp"

document = text_preprocess(document)
document = remove_stopwords(document)

label = nb_model.predict([document])
print('Predict label:', label_encoder.inverse_transform(label))
```

```
Predict label: ['__label__phim_ảnh']
```

```
document = "Chelsea"

document = text_preprocess(document)
document = remove_stopwords(document)

label = nb_model.predict([document])
print('Predict label:', label_encoder.inverse_transform(label))
```

```
Predict label: ['__label__thể_thao']
```

```
document = "đàn piano"

document = text_preprocess(document)
document = remove_stopwords(document)

label = nb_model.predict([document])
print('Predict label:', label_encoder.inverse_transform(label))
```

```
Predict label: ['__label__âm_nhạc']
```

```
document = "công an"

document = text_preprocess(document)
document = remove_stopwords(document)

label = nb_model.predict([document])
print('Predict label:', label_encoder.inverse_transform(label))
```

```
Predict label: ['__label__pháp_luật']
```

```
document = "samsung"

document = text_preprocess(document)
document = remove_stopwords(document)

label = nb_model.predict([document])
print('Predict label:', label_encoder.inverse_transform(label))
```

```
Predict label: ['__label__công_nghệ']
```

Hình 33: Một số kết quả thực hiện

- Sau khi thực hiện lại phần dữ liệu cho trực quan nhất có thể em thấy mô hình đã hoàn thiện hơn lúc ban đầu khá nhiều, cụ thể các nhãn được phân loại một cách khá chính xác, giường như sai số là khá nhỏ và nếu em thực hiện trên 1 đoạn văn bản dài thì mức độ chính xác lên tới hơn 90%

X. Tài liệu tham khảo

Thư viện scikit_learn: https://scikit-learn.org/

https://machinelearningcoban.com/

https://github.com/nguyenvanhieuvn/text-classification-tutorial

https://viblo.asia/p/gioi-thieu-ve-support-vector-machine-svm-6J3ZgPVElmB

https://viblo.asia/p/thuat-toan-phan-lop-naive-bayes-924lJWPm5PM

https://viblo.asia/p/knn-k-nearest-neighbors-1-djeZ14ejKWz

Phương pháp đánh trọng số tf-idf trong thuật toán naïve bayes

$$\mathrm{tf}(t,d) = \frac{\mathrm{f}(t,d)}{\max\{\mathrm{f}(w,d) : w \in d\}}$$

Trong đó:

- tf(t, d): tần suất xuất hiện của từ t trong văn bản d
- f(t, d): Số lần xuất hiện của từ t trong văn bản d
- max({f(w, d) : w ∈ d}): Số lần xuất hiện của từ có số lần xuất hiện nhiều nhất trong văn bản d

$$\mathrm{idf}(t,D) = \log \frac{|D|}{|\{d \in D : t \in d\}|}$$

Trong đó:

- idf(t, D): giá trị idf của từ t trong tập văn bản
- |D|: Tổng số văn bản trong tập D
- |{d ∈ D : t ∈ d}|: thể hiện số văn bản trong tập D có chứa từ t.

➢ **tfidf(t, d, D) = tf(t, d) x idf(t, D)**