HUTECH UNIVERSITY

FACULTY of INFORMATION TECHNOLOGY

HUTECH

Đại học Công nghệ Tp.HCM

FINANCIAL SENTIMENT ANALYSIS on Vietnamese Stock Market Headlines

OPEN-SOURCE Tools for DATA SCIENCE

Module Code **CMP1044**

Instructor **Luu Toan Dinh**

Student **Nguyen Hoang Khang**

Student ID **2186400244**

Ho Chi Minh City, March 23, 2025

HUTECH UNIVERSITY
FACULTY of INFOMATION TECHNOLOGY

——————o0o——————

FINAL PROJECT

Module: **OPEN-SOURCE Tools for DATA SCIENCE**

FINANCIAL SENTIMENT ANALYSIS on Vietnamese Stock Market Headlines

Instructor: **Luu Toan Dinh**

Student: **Nguyễn Hoàng Khang**

Student ID: **2186400244**

Ho Chi Minh City, March 23, 2025

Contents

List of Figures

List of Tables

Acronyms & Abbreviations

Chapter 1

OVERVIEW

1.1 Problem

Financial news headlines are a fertile source of NLP data, especially when it comes to predicting how a stock will perform. Frequently, this is done via sentiment analysis, an NLP task that buckets phrases into positive, negative, and neutral.

In the field of Finance and Banking, language is not merely a means of communication but also a valuable resource for analysis and prediction activities. Language encapsulates emotional nuances, conveys the perspectives of information providers, and significantly influences the emotions, behaviors, and decisions of investors, thereby regulating market dynamics.

The nuances of financial language are not simply positive or negative but can vary depending on the context and the reader's perspective. A piece of news can convey various nuances to different entities such as investors, experts, and financial analysts. These nuances also depend on how information is conveyed, the choice of language used, and its impact on investment behavior and decisions.

1.2 Previous Works

Figure 1.1: Prevoous Works

Chapter 2 DATA PREPROCESSING

The data comprises 240 businesses from 2008 to 2023 containing information: Stock code (`Ma_CK`), Date (`Date`), Headlines related to each business's information updated every day and hour (`Title`), Opening (`Open`) and closing prices (`Close`), Volume (`Volume`), Highest and lowest prices (`High`,`Low`).

	_id		Datetime	Date	Time	Ma_CK	Title	Link
212	657af088d3e6d8c4382dd644	276	2023-02-02 09:28:00	2023-02-02	09:28:00	LQN	LQN: Báo cáo quản trị công ty năm 2022	https://s.cafef.vn/LQN-534329/lqn-bao-cao-quan...
44	657af088d3e6d8c4382dd5d5	165	2023-01-19 16:40:00	2023-01-19	16:40:00	HAS	HAS: Giải trình biến động lợi nhuận BCTC HN và...	https://s.cafef.vn/HAS-532148/has-giai-trinh-b...
164	657af088d3e6d8c4382dd6b2	386	2023-02-06 15:20:00	2023-02-06	15:20:00	NVT	Chủ chuỗi Resort cao cấp chính thức báo lỗ 12 ...	https://s.cafef.vn/NVT-534628/chu-chuoi-resort...
300	657af088d3e6d8c4382dd7e9	697	2023-02-07 17:17:00	2023-02-07	17:17:00	TGG	TGG: Thông báo họp và tài liệu họp ĐHĐCĐ bất t...	https://s.cafef.vn/TGG-534866/tgg-thong-bao-ho...
366	657af088d3e6d8c4382dd774	580	2023-02-23 14:01:00	2023-02-23	14:01:00	RDP	RDP: 10.3.2023, ngày GDKHQ tổ chức ĐHĐCĐ thườn...	https://s.cafef.vn/RDP-537715/rdp-1032023-ngay...

	_id		Date	maCK	San	Open	Close	High	Low	Volume	priceChange
5056	657b04e2d3e6d8c4382decd0	5226	31/01/2023	SDP	upcom	1.6	1.6	1.6	1.6	0	0.00(0.00%)
5999	657b04e2d3e6d8c4382defd5	5999	02/02/2023	TOP	upcom	1.6	1.6	1.6	1.6	0	0.00(0.00%)
2057	657b04e0d3e6d8c4382de118	2226	13/01/2023	LCM	upcom	2.4	2.4	2.4	2.3	312,840	0.00(0.00%)
4005	657b04e1d3e6d8c4382de83e	4056	12/01/2023	PVB	hnx	12.6	13.0	13.1	12.3	357,956	0.70(5.69%)
481	657b04ded3e6d8c4382dda1b	437	22/02/2023	BHT	upcom	16.9	16.9	16.9	16.9	0	0.00(0.00%)

Figure 2.1: The entire dataset

The data consists of two datasets:

a. `titles`: ≈ 670 headlines related to 240 businesses;

b. `price`: ≈ 12765 data observations of opening and closing prices each day within the timeframe of 2008 - 2023.

✍ **Note.** The number of observations for stock prices will be higher than the number of observations for news, even though they share the same research timeframe and scope. Several cases lead to this situation:

- Some companies only release a news bulletin every few days;
- Some companies face issues with delisting or no trading→ the absence of news during certain periods;
- Some companies may release 1-2 news bulletins in a day depending on the timing.

■

2.1 Connect to MongoDB Atlas by PyMongo & Import Libraries

```
from pymongo import MongoClient
from pprint import pprint
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
import numpy as np
from IPython.display import display
```

```
connect = "mongodb+srv://gausseuler159357:botvamatcuoi2272003@khang.pd0scur.mongodb.net/?retryWrites=true&w=majority"
```

```
client = MongoClient(connect)
```

```
client.list_database_names()
```

```
['THPTQG',
 'media',
 'new_db',
 'news_headlines',
 'sample',
 'seaborn_data',
 'admin',
 'local']
```

Figure 2.2: Connect to MongoDB Atlas by PyMongo

Import underthesea toolkit[1]

```
!pip install underthesea
from underthesea import word_tokenize, pos_tag, sent_tokenize

```

Listing 2.1: Import underthesea package

[1]underthesea is a suite of open source Python modules, data sets and tutorials supporting research and development in Vietnamese Natural Language Processing.

```
media = client.media
```

```
media.list_collection_names()
```

```
['media', 'price']
```

```
titles = media.media
price = media.price
```

```
pprint(price.find_one())
```

```
{'': 3,
 'Close': 10.8,
 'Date': '22/02/2023',
 'High': 10.8,
 'Low': 10.8,
 'Open': 10.8,
 'San': 'upcom',
 'Volume': 0,
 '_id': ObjectId('657b04ded3e6d8c4382dd869'),
 'maCK': 'ABC',
 'priceChange': '0.00(0.00%)'}
```

Figure 2.3: List collections that existing in database

```
# Perform a query and retrieve documents
query_result1 = titles.find({})
query_result2 = price.find({})
# Convert query result to a DataFrame
titles = pd.DataFrame(list(query_result1))
price = pd.DataFrame(list(query_result2))
# Display DataFrame
display(titles.sample(5))
display(price.sample(5))
```

	_id		Datetime	Date	Time	Ma_CK	Title	Link
167	657af088d3e6d8c4382dd6c5	405	2023-01-19 16:33:00	2023-01-19	16:33:00	PAC	PAC: Bổ nhiệm ông Nguyễn Duy Hưng giữ chức Phó...	https://s.cafef.vn/PAC-532126/pac-bo-nhiem-ong...
635	657af088d3e6d8c4382dd672	322	2023-01-30 18:32:00	2023-01-30	18:32:00	MSN	MSN: Báo cáo tình hình quản trị công ty năm 2022	https://s.cafef.vn/msn-533607/msn-bao-cao-tinh...
237	657af088d3e6d8c4382dd693	355	2023-02-09 17:14:00	2023-02-09	17:14:00	MWG	MWG: Báo cáo thay đổi về sở hữu của nhóm NĐTNN...	https://s.cafef.vn/MWG-535275/mwg-bao-cao-thay...
49	657af088d3e6d8c4382dd5ea	186	2023-02-23 14:20:00	2023-02-23	14:20:00	HLY	HLY: Nghị quyết HĐQT v/v chốt danh sách cổ đôn...	https://s.cafef.vn/HLY-537730/hly-nghi-quyet-h...
90	657af088d3e6d8c4382dd58d	93	2023-02-24 17:49:00	2023-02-24	17:49:00	DP2	DP2: Chốt danh sách cổ đông để tổ chức Đại hội...	https://s.cafef.vn/DP2-538064/dp2-chot-danh-sa...

	_id		Date	maCK	San	Open	Close	High	Low	Volume	priceChange
4960	657b04e1d3e6d8c4382deb4a	4836	24/02/2023	SAB	hose	187.1	189.00	189.6	185.00	98,800	1.90(1.02%)

Figure 2.4: The whole dataset

2.2 Import language-formed data ('titles')

```
#import data
# titles = pd.read_csv("sample_data/media.csv")
titles = titles[['Date','Time','Ma_CK','Title']]
display(titles.head(10))
```

1 to 10 of 10 entries

index	Date	Time	Ma_CK	Title
0	2023-02-24 00:00:00	17:51:00	ABC	ABC: Miễn nhiệm bà Dương Thị Thương - Kế toán trưởng từ 24/02/2023
1	2023-02-24 00:00:00	17:23:00	ABC	ABC: 3.3.2023, ngày GDKHQ Tham dự Đại hội đồng cổ đông thường niên năm 2023
2	2023-02-20 00:00:00	15:37:00	ABC	ABC: Bổ nhiệm bà Hồ Thị Hà giữ chức người công bố thông tin từ 22/02/2023
3	2023-01-31 00:00:00	17:29:00	ABC	ABC: Báo cáo quản trị công ty năm 2022
4	2023-01-30 00:00:00	15:59:00	ACM	ACM: Đính chính Báo cáo quản trị công ty năm 2022
5	2023-02-17 00:00:00	15:05:00	AGF	AGF: 14.3.2023, ngày GDKHQ Tham dự đại hội đồng cổ đông thường niên năm 2023
6	2023-02-13 00:00:00	16:52:00	AGF	AGF: Nghị quyết HĐQT thống nhất tổ chức ĐHĐCĐ TN 2023
7	2023-02-10 00:00:00	14:44:00	AGF	AGF: Nghị quyết HĐQT v/v tổ chức ĐHĐCĐ TN 2023
8	2023-01-17 00:00:00	14:50:00	AGF	AGF: Báo cáo quản trị công ty năm 2022
9	2023-02-20 00:00:00	16:15:00	AMD	AMD: Nghị quyết HĐQT v/v triệu tập ĐHĐCĐ bất thường năm 2023

Figure 2.5: Display the first 10 entries of the 'title' dataset

Retrieve a list of words that are `Ma_CK`.

```
#Retrieve a list of words that are "Ma_CK".
stockword = titles['Ma_CK'].unique().tolist()
#for instance
text = 'ABC: Miễn nhiệm bà Dương Thị Thương - Kế toán trưởng từ 24/02/2023'
```

Figure 2.6: Establish stock words list

2.3 Prepare language-formatted data

2.3.1 Standardize Vietnamese sentences

```
def loaddicchar():
    uniChars = "àáảãạâầấẩẫậăằắẳẵặèéẻẽẹêềếểễệđìíỉĩịòóỏõọôồốổỗộơờớởỡợùúủũụưừứửữựỳýỷỹỵÀÁẢÃẠÂẦẤẨẪẬĂẰẮẲẴẶÈÉẺẼẸÊỀẾỂỄỆĐÌÍỈĨỊÒÓỎÕỌÔỒỐỔỖỘƠỜỚỞỠỢ
    unsignChars = "aaaaaaaaaaaaaaaaaeeeeeeeeeeediiiiioooooooooooooooooouuuuuuuuuuuyyyyyAAAAAAAAAAAAAAAAAEEEEEEEEEEEDIIIIIOOOOOOOOO
    dic = {}
    char1252 = "à|á|ả|ã|ạ|â|ầ|ấ|ẩ|ẫ|ậ|ă|ằ|ắ|ẳ|ẵ|ặ|è|é|ẻ|ẽ|ẹ|ê|ề|ế|ể|ễ|ệ|đ|ì|í|ỉ|ĩ|ị|ò|ó|ỏ|õ|ọ|ô|ồ|ố|ổ|ỗ|ộ|ơ|ờ|ớ|ở|ỡ|ợ|ù|ú|ủ|ũ|ụ|ư
    charutf8 = "à|á|ả|ã|ạ|â|ầ|ấ|ẩ|ẫ|ậ|ă|ằ|ắ|ẳ|ẵ|ặ|è|é|ẻ|ẽ|ẹ|ê|ề|ế|ể|ễ|ệ|đ|ì|í|ỉ|ĩ|ị|ò|ó|ỏ|õ|ọ|ô|ồ|ố|ổ|ỗ|ộ|ơ|ờ|ớ|ở|ỡ|ợ|ù|ú|ủ|ũ|ụ|ư
    for i in range(len(char1252)):
        dic[char1252[i]] = charutf8[i]
    return dic
```

```
def convert_unicode(txt):
    dicchar = loaddicchar()
    return regex.sub(r'à|á|ả|ã|ạ|â|ầ|ấ|ẩ|ẫ|ậ|ă|ằ|ắ|ẳ|ẵ|ặ|è|é|ẻ|ẽ|ẹ|ê|ề|ế|ể|ễ|ệ|đ|ì|í|ỉ|ĩ|ị|ò|ó|ỏ|õ|ọ|ô|ồ|ố|ổ|ỗ|ộ|ơ|ờ|ớ|ở|ỡ|ợ|ù|ú|
                     lambda x: dicchar[x.group()], txt)

```

```
print("After unicode normalizing")
print(convert_unicode(text))
```

```
After unicode normalizing
ABC: Miễn nhiệm bà Dương Thị Thương - Kế toán trưởng từ 24/02/2023
```

Figure 2.7: Convert to standard unicode-form

2.3.2 The combination of Tokenization, Ngrams, and POS Tagging

Handle exceptional cases with the word 'không' If the word 'không' appears, it will be linked ('') with the word following it.

```
def process_special_word(text):
    new_text = ''
    text_lst = text.split()
    i = 0
    if 'không' in text_lst:
        while i<= len(text_lst)-1:
            word = text_lst[i]
            if word == 'không': # Handling exceptional cases with the word 'không'
                next_idx = i+1
                if next_idx <= len(text_lst) - 1:  #If the word 'không' appears, it will be linked with the word following it.
                    word = word + '_' + text_lst[next_idx]
                i = next_idx + 1
            else:
                i+=1
            new_text+=word+' '
    else:
        new_text = text
    return new_text.strip()
```

Figure 2.8: Handle exceptional cases with the word 'không' If the word 'không' appears, it will be linked ('_') with the word following it.

Keep the parts of speech as verbs, adjectives, and nouns; and remove punctuation marks within the sentence

```
# Keep the parts of speech as verbs, adjectives, and nouns; and remove punctuation marks within the sentence
def process_postag_thesea(text):
    text = text + " a"
    new_document = ''
    for sentence in sent_tokenize(text):
        sentence = sentence.replace('.','').lower() #Remove the punctuation marks in the sentence
        #POS Tagging
        lst_word_type = ['V','N','A']
        sentence = ' '.join(word[0].lower() if word[1].upper() in lst_word_type else ''
                            for word in pos_tag(process_special_word(word_tokenize(sentence, format = "text"))))
        new_document = new_document + sentence + ' '
        new_document = regex.sub(r'\s+',' ', new_document).strip() #Remove unnecessary white spaces.
    return new_document

process_postag_thesea(text)
```

```
'xph bổ_nhiệm lê mạnh_cường giữ chức phụ_trách nhiệm_vụ kế_toán_trưởng 09/02/2023'
```

Figure 2.9: Keep the parts of speech as verbs, adjectives, and nouns; and remove punctuation marks within the sentence

Cleanse and standardize language data: remove stock words, times, time word, and digits in each sentence

```
def remove_stockword(text, stockword): #remove stock_words
    document = ' '.join('' if word.upper() in stockword  else word for word in text.split())
    document = regex.sub(r'\s',' ', document).strip()
    return document

def remove_time(text):
    document = ' '.join('' if word.find('/')!=-1 else word for word in text.split())
    document = regex.sub(r'\s+',' ',document).strip()
    return document

def remove_timeword(text):
    document = ' '.join('' if word in ['ngày','tháng','năm','quý','lần'] else word for word in text.split())
    document = regex.sub(r'\s+',' ',document).strip()
    return document

def remove_number(text):
    document = ' '.join('' if any(x.isdigit() for x in word) else word for word in text.split())
    document = regex.sub(r'\s+',' ',document).strip()
    return document
```

Figure 2.10: Cleanse and standardize language data: remove stock words, times, time word, and digits in sentences

```
print('Remove "Ma_CK')
print(remove_stockword(process_postag_thesea(text), stockword))
print("After removing Numbers")
print(remove_number(text))
print("After removing Time")
print(remove_time(text))
```

```
Remove "Ma_CK
bổ_nhiệm lê mạnh_cường giữ chức phụ_trách nhiệm_vụ kế_toán_trưởng 09/02/2023
After removing Numbers
XPH: Bổ nhiệm ông Lê Mạnh Cường giữ chức Phụ trách nhiệm vụ Kế toán trưởng từ
After removing Time
XPH: Bổ nhiệm ông Lê Mạnh Cường giữ chức Phụ trách nhiệm vụ Kế toán trưởng từ
```

Figure 2.11: The results after the first normalizing.

Comment. After standardizing Unicode and tokenizing into meaningful words, it's crucial to remove stock symbols, numbers, time words, and time in the preprocessing of natural language data because they create ambiguity during model training, adding noise and redundancy to the dataset sentences.

Normalize the time of the 'titles' data

```
titles['Time'] = titles['Time'].str[:4].str.replace(":",".").astype('float')
titles['Date'] = pd.DatetimeIndex(titles['Date'])
titles = titles.rename(columns = {'Date':'date','Time':"hour",'Ma_CK':"ma_ck",'Title':"title"})
titles = titles.dropna().reset_index(drop=True)
```

```
display(titles.tail(10))
```

1 to 10 of 10 entries

index	date	hour	ma_ck	title
811	2023-01-12 00:00:00	10.2	VTR	VTR: Bà Nguyễn Nguyệt Vân Khanh - Ủy viên HĐQT đăng ký mua 19.500 CP
812	2023-01-12 00:00:00	10.2	VTR	VTR: Ông Nguyễn Hà Trung - Phó Tổng Giám đốc đăng ký mua 108.100 CP
813	2023-01-12 00:00:00	10.2	VTR	VTR: Ông Huỳnh Phan Phương Hoàng - Phó Tổng Giám đốc đăng ký mua 92.100 CP
814	2023-01-12 00:00:00	10.2	VTR	VTR: Ông Đỗ Thanh Hùng - Giám đốc tài chính đăng ký mua 349.400 CP
815	2023-01-19 00:00:00	13.4	VVN	VVN: Báo cáo quản trị công ty năm 2022
816	2023-01-18 00:00:00	8.1	X77	X77: Báo cáo quản trị công ty năm 2022
817	2023-02-09 00:00:00	13.2	XPH	XPH: Bổ nhiệm ông Lê Mạnh Cường giữ chức Phụ trách nhiệm vụ Kế toán trưởng từ 09/02/2023
818	2023-02-09 00:00:00	13.2	XPH	XPH: Quyết định duy trì diện bị hạn chế giao dịch trên hệ thống giao dịch UPCoM
819	2023-02-03 00:00:00	8.5	XPH	XPH: Giải trình liên quan đến Báo cáo tài chính giữa niên độ năm 2022
820	2023-02-02 00:00:00	13.4	XPH	XPH: Báo cáo quản trị công ty năm 2022

Figure 2.12: Normalize the time of the 'titles' data

Handle the entire list of titles

```
full_content_new = []
for i in range(len(titles)):
    text = titles.loc[i]["title"]
    full_content_new.append(clean_text(text,stockword))
processed_data = pd.DataFrame.from_dict({"title":titles["title"],
                                         "title_new":full_content_new})
```

```
display(processed_data.head(10))
```

1 to 10 of 10 entries Filter

index	title	title_new
0	ABC: Miễn nhiệm bà Dương Thị Thương - Kế toán trưởng từ 24/02/2023	miễn_nhiệm dương_thị_thương kế_toán_trưởng
1	ABC: 3.3.2023, ngày GDKHQ Tham dự Đại hội đồng cổ đông thường niên năm 2023	tham_dự đại_hội_đồng cổ_đông
2	ABC: Bổ nhiệm bà Hồ Thị Hà giữ chức người công bố thông tin từ 22/02/2023	bổ_nhiệm hồ thị_hà giữ chức người công_bố thông_tin
3	ABC: Báo cáo quản trị công ty năm 2022	báo_cáo quản_trị công_ty
4	ACM: Đính chính Báo cáo quản trị công ty năm 2022	đính_chính báo_cáo quản_trị
5	AGF: 14.3.2023, ngày GDKHQ Tham dự đại hội đồng cổ đông thường niên năm 2023	tham_dự đại_hội_đồng cổ_đông
6	AGF: Nghị quyết HĐQT thống nhất tổ chức ĐHĐCĐ TN 2023	nghị_quyết hđqt thống_nhất tổ_chức đhđcđ tn
7	AGF: Nghị quyết HĐQT v/v tổ chức ĐHĐCĐ TN 2023	nghị_quyết v tổ_chức đhđcđ tn
8	AGF: Báo cáo quản trị công ty năm 2022	báo_cáo quản_trị công_ty
9	AMD: Nghị quyết HĐQT v/v triệu tập ĐHĐCĐ bất thường năm 2023	nghị_quyết v triệu_tập đhđcđ

Figure 2.13: Handle the entire list of titles

Handle the entire list of titles with timestamps

✍ **Note.** News released on Saturdays and Sundays will be moved to Friday evenings for convenient handling of price growth. Therefore, at this stage, Saturday and Sunday will be shifted to Friday, and the time will be changed to 7 p.m. ■

```
from datetime import datetime, timedelta
```

```
titles['weekday'] = titles['date'].dt.day_name()
titles['date'][titles['weekday']=='Sunday'] = titles[titles['weekday']=='Sunday']['date']-timedelta(days=2)
titles['date'][titles['weekday']=='Saturday'] = titles[titles['weekday']=='Saturday']['date']-timedelta(days=1)
titles['hour'][titles['weekday']=='Sunday']=19
titles['hour'][titles['weekday']=='Saturday']=19
```

```
df_delta = titles.copy()
df_delta['title_new'] = processed_data['title_new']
display(df_delta.sample(10))
```

1 to 10 of 10 entries Filter

index	date	hour	ma_ck	title	weekday	title_new
44	2023-01-10 00:00:00	13.1	BHT	BHT: 19.1.2023, ngày GDKHQ Tổ chức Đại hội đồng cổ đông thường niên năm 2023	Tuesday	tổ_chức đại_hội_đồng cổ_đông
176	2023-02-14 00:00:00	9.4	HLG	HLG: Thông báo về việc điều chỉnh tình trạng chứng khoán	Tuesday	thông_báo việc điều_chỉnh tình_trạng chứng_khoán
506	2023-01-31 00:00:00	23.4	PRT	PRT: Đính chính Báo cáo quản trị công ty năm 2022	Tuesday	đính_chính báo_cáo quản_trị
639	2023-01-27 00:00:00	16.4	SD5	SD5: Báo cáo quản trị công ty năm 2022	Friday	báo_cáo quản_trị công_ty
51	2023-01-27 00:00:00	15.5	CEG	CEG: Báo cáo quản trị công ty năm 2022	Friday	báo_cáo quản_trị công_ty

Figure 2.14: Handle the entire list of titles with timestamps

2.4 Establish a measurement scale to classify financial language based on stock returns

Using stock return values to create a rating scale for assessing the sentiment of financial language. If the stock return value is positive (+), it is labeled as 0, 1 in the remaining cases (i.e., negative label).

```
# price = pd.read_csv("sample_data/price.csv")
usecols = ['Date','maCK','Open','Close']
price = price[usecols]
```

```
display(price.sample(10))
```

	Date	maCK	Open	Close
4317	27/01/2023	S27	1.20	1.20
6283	09/01/2023	VST	2.70	2.70
4122	03/02/2023	PVH	1.90	2.00
3309	13/02/2023	POM	5.12	5.27
6803	22/02/2023	X77	0.30	0.30
997	23/02/2023	DLG	2.30	2.26
4752	30/01/2023	RCD	2.40	2.40
686	21/02/2023	DLG	2.42	2.36
3039	01/02/2023	PBP	13.60	13.40
2089	27/02/2023	KSH	0.60	0.60

Figure 2.15: The first 10 entries from the 'price' dataset.

Additionally, the variable `session` represents the return value before or during the trading session. The reason for calculating return within and before the session is because the news update timeframe impacts different stock market return periods.

- If the news appears before the market opens, the stock market's return is calculated based on the *pre-session threshold*.
- If the news appears while the market is open, the return for that day is calculated using the *intra-session threshold*.
- In the special case where information is released on Saturday or Sunday, the study calculates it within the *pre-session threshold of the following Monday, the start of the*

new week.

2.4.1 Renormalize 'price' table

```
price['Date'] = pd.DatetimeIndex(price['Date'],dayfirst=True)
price =  price.rename(columns = {'Date':'date','maCK':'ma_ck','Open':'open_price',
                                 'Close':'close_price'})
#Renormalize stock_price table
price = price.sort_values(['ma_ck','date'],ascending=True)
price['open_price'] = price['open_price'].astype('str')
price['open_price'] = price['open_price'].str.replace('00.0','000')
price['open_price'] = price['open_price'].str.replace(',','.')
price['open_price'] = price['open_price'].astype('float',errors = 'raise')
```

```
<ipython-input-146-ff8729940347>:7: FutureWarning: The default value of regex will change from True to False in a future version.
  price['open_price'] = price['open_price'].str.replace('00.0','000')
```

Figure 2.16: Renormalize 'price' table

```
display(price.sample(10))
```

	date	ma_ck	open_price	close_price
5254	2023-01-13	SMA	7.5	7.5
5354	2023-02-22	STT	1.8	1.8
1775	2023-02-02	HPI	33.5	33.5
4188	2023-02-06	PXI	2.1	2.1
6444	2023-02-23	VAT	0.8	0.8
5308	2023-01-30	TBH	8.5	8.5
451	2023-01-11	ATB	0.9	0.9
2934	2023-02-24	NHP	0.7	0.7

Figure 2.17: The result after renormalizing 'price' table.

2.4.2 Visualize Time Series Data

2.4.3 Calculate stock return and create 'session' variable.

```
#Create two columns for 'previous day's closing price' and 'next day's opening price.
price['close_price_lag'] = price.groupby('ma_ck')['close_price'].shift()
price['open_price_next'] = price.groupby('ma_ck')['open_price'].shift(-1)
price = price.dropna()
```

```
#Calculate stock profit obtained during the trading session
price['return_in_session'] = np.log(price['close_price']/price['open_price'])
#Calculate stock profit obtained between two trading sessions
price['return_before_session'] = np.log(price['open_price']/price['close_price_lag'])
```

```
#Standardize the price table to capture return within and between 2 sessions.
price_stack = price[['date','ma_ck','return_in_session','return_before_session']].set_index(['date',
                                                                                              "ma_ck"]).stack().reset_index()
price_stack.columns = ['date','ma_ck','session','return']
price_stack = price_stack.sort_values(['ma_ck','date','session'])
price_stack = price_stack[price_stack['return']!=np.inf]
price_stack = price_stack[price_stack['return']!=-np.inf]
```

```
display(price_stack.sample(10))
```

1 to 10 of 10 entries Filter

index	date	ma_ck	session	return
11595	2023-02-23 00:00:00	V15	return_before_session	0.0
5943	2023-02-06 00:00:00	PC1	return_before_session	0.0
12214	2023-02-13 00:00:00	VPC	return_in_session	0.0
7983	2023-02-13 00:00:00	PVS	return_before_session	0.0
3953	2023-01-17 00:00:00	KSH	return_before_session	0.0
6132	2023-02-16 00:00:00	PCT	return_in_session	0.0202027073175194 7
12586	2023-01-12 00:00:00	VVN	return_in_session	0.0
7133	2023-01-10 00:00:00	PPI	return_before_session	0.0
12056	2023-02-23 00:00:00	VLF	return_in_session	0.0
11849	2023-01-30 00:00:00	VFG	return_before_session	-0.012739025777429714

Figure 2.18: Calculate stock return and create 'session' variable.

2.4.4 Standardize the 'titles' table to match the 'title' variable with the data of the 'return' variable based on the intra or between 2 sessions

```
titles = df_delta.copy()
titles['new_date'] = titles['date']
titles['weekend'] = False
titles['weekend'][titles['weekday']=='Friday'] = titles[titles['weekday']=='Friday']['hour']>16
titles['new_date'][titles['hour']>16] = titles[titles['hour']>16]['new_date']+timedelta(days=1)
titles['new_date'][titles['weekend']==True] = titles[titles['weekend']==True]['new_date']+timedelta(days=2)
titles['session'] = ['return_in_session' if (x>8) and (x<16) else 'return_before_session' for
                     x in titles['hour']]
```

```
1  display(titles.head(10))
```

1 to 10 of 10 entries Filter

index	date	hour	ma_ck	title	weekday	title_new	new_date	weekend	session
0	2023-01-31 00:00:00	17.2	ABC	ABC: Báo cáo quản trị công ty năm 2022	Tuesday	báo_cáo quản_trị công_ty	2023-02-01 00:00:00	false	return_before_session
1	2023-01-30 00:00:00	15.5	ACM	ACM: Đính chính Báo cáo quản trị công ty năm 2022	Monday	đính_chính báo_cáo quản_trị	2023-01-30 00:00:00	false	return_in_session
2	2023-02-17 00:00:00	15.0	AGF	AGF: 14.3.2023, ngày GDKHQ Tham dự đại hội đồng cổ đông thường niên năm 2023	Friday	tham_dự đại_hội_đồng cổ_đông	2023-02-17 00:00:00	false	return_in_session
3	2023-02-10 00:00:00	14.4	AGF	AGF: Nghị quyết HĐQT v/v tổ chức ĐHĐCĐ TN 2023	Friday	nghị_quyết v tổ_chức đhđcđ tn	2023-02-10 00:00:00	false	return_in_session
4	2023-01-19 00:00:00	16.5	AMP	AMP: Báo cáo quản trị công ty năm 2022	Thursday	báo_cáo quản_trị công_ty	2023-01-20 00:00:00	false	return_before_session
5	2023-01-31 00:00:00	18.1	APT	APT: Báo cáo quản trị công ty năm 2022	Tuesday	báo_cáo quản_trị công_ty	2023-02-01 00:00:00	false	return_before_session
6	2023-01-30 00:00:00	18.0	AST	AST: Giải trình biến động KQKD quý 4/2022 so với cùng kỳ năm trước	Monday	giải_trình biến_động kqkd so cùng kỳ trước	2023-01-31 00:00:00	false	return_before_session
7	2023-02-01 00:00:00	16.1	ATB	ATB: Báo cáo quản trị công ty năm 2022	Wednesday	báo_cáo quản_trị công_ty	2023-02-02 00:00:00	false	return_before_session
8	2023-01-16 00:00:00	11.2	ATG	ATG: Báo cáo quản trị công ty năm 2022	Monday	báo_cáo quản_trị công_ty	2023-01-16 00:00:00	false	return_in_session

Figure 2.19: Standardize the 'titles' table to match the 'title' variable with the data of the 'return' variable based on the intra or between 2 sessions

2.4.5 Concatenate 'pricestack' and 'titles' data frame

```
1  price_stack = price_stack.rename(columns={'date':'new_date'})
2  df = titles.merge(price_stack, on = ['new_date','ma_ck','session'],how='left')
3  df_delta = df.drop_duplicates(ignore_index = True)
4  df_delta_sub = df_delta.dropna().reset_index(drop=True)
5  df_sub = df
6  df_sub['title_new'] = df_delta_sub['title_new']+' '
7  df = df_sub[['ma_ck','title_new','new_date','session',
8               'return']].groupby(['new_date','ma_ck','session','return']).sum().reset_index()
9  df['title_new'] = df['title_new'].str[:-1]
10
```

```
1  display(df.head(10))
```

1 to 10 of 10 entries Filter

index	new_date	ma_ck	session	return	title_new
0	2023-01-10 00:00:00	BHT	return_in_session	0.0	báo_cáo tình_hình quản_trị
1	2023-01-10 00:00:00	HAG	return_before_session	-0.008830079448272037	đính_chính kqkd a
2	2023-01-10 00:00:00	PET	return_in_session	-0.007623925110659259	thông_báo trái_phiếu có khả_năng bị hủy bỏ niêm_yết
3	2023-01-10 00:00:00	PLX	return_in_session	0.04865448816352538	huỳnh_phan_phương_hoàng tổng_giám_đốc đăng_ký mua cp
4	2023-01-10 00:00:00	PRT	return_in_session	-0.017544309650909393	báo_cáo thay_đổi sở_hữu nhóm ndtnn có liên_quan
5	2023-01-10 00:00:00	SCO	return_in_session	0.0	giải_trình liên_quan báo_cáo tài_chính a
6	2023-01-10 00:00:00	TDH	return_before_session	-0.06503247027085468	tham_dự đại_hội_đồng cổ_đông
7	2023-01-11 00:00:00	ATG	return_before_session	0.0	báo_cáo quản_trị công_ty
8	2023-01-11 00:00:00	BT6	return_before_session	0.0	thông_báo nộp tiền sử_dụng đất theo bản_án tòa_án
9	2023-01-11 00:00:00	HTT	return_in_session	0.0	báo_cáo tình_hình quản_trị

Figure 2.20: Concatenate 'price_stack' and 'titles' data frame

2.4.6 Drop Na values and save preprocessed data

```
df.shape
```

```
(543, 5)
```

```
df = df.dropna()
```

```
df.shape
```

```
(469, 5)
```

```
df.to_csv('input_table.csv')
```

Figure 2.21: Drop Na values and save preprocessed data

2.4.7 Convert language-based data into numerical value

Frequency-based Embedding

Just keep titles that change 'returns'

```
df = pd.read_csv('input_table.csv')
df = df[df['return']!=0]
```

```
df.shape
```

```
(245, 6)
```

Create dummy variables for titles that have a significant impact on stock prices

```
df['deep']= 0
df['deep'][df['return']>df['return'].quantile(0.999)]=100
df['deep'][df['return']<df['return'].quantile(0.001)]=100
df = df.reset_index(drop=True)
```

Create dummy variable for 'mack'

```
dummies = pd.get_dummies(df['ma_ck'])
```

Import CountVectorizer from sklearn

```
from sklearn.feature_extraction.text import CountVectorizer
```

```
df['title_new'].isna().sum()
```

```
0
```

Figure 2.22: Import "CountVectorizer" from sklearn

Use CountVectorizer()

```
X = df['title_new']
vectorizer = CountVectorizer()
vectorizer.fit(X)
X_vec = vectorizer.transform(X)
columns  = vectorizer.get_feature_names_out()
sentiments = pd.DataFrame(data = X_vec.toarray(),columns = columns)
sentiments[dummies.columns] = dummies
```

Filter out a list of words from the headlines of the day with significant changes in stock prices

```
sentiments['deep'] = df['deep']
deep = pd.DataFrame(data = sentiments[sentiments['deep']==100].sum(),columns=['deep'])
deep = deep[deep['deep']>0]
deep_lst = deep.index.tolist()
```

Filter out a list of words that are popular in the media

```
big = pd.DataFrame(data=np.count_nonzero(sentiments, axis=0), index = sentiments.columns, columns=['counts'])
big_lst = big[big['counts']>0.01*len(sentiments)].index.tolist()
```

Combine the above two lists to create a list of words to retain for the model

At this stage, it will be very helpful in the sparse matrix as it reduces many dimensions of the data.

```
for i in deep_lst:
    if i not in big_lst:
        big_lst.append(i)
```

The data table after being transformed into vector space

```
sentiments_new = sentiments[big_lst]
sentiments_new['title'] = df['title_new']
sentiments_new['return'] = df['return']
sentiments_new['ma_ck'] = df['ma_ck']
sentiments_new['Date'] = df['new_date']
sentiments_new['session'] = df['session']
```

Figure 2.23: The data table after being transformed into vector space

Eliminate the titles that do not contain or contain too few words from the chosen list

```
sentiments_new['sum'] = sentiments_new[big_lst].sum(axis=1)
sentiments_new = sentiments_new[sentiments_new['sum']>sentiments_new['sum'].quantile(0.2)]
```

Figure 2.24: Eliminate the titles that do not contain or contain too few words from the chosen list

Display 'Sentiments'

```
display(sentiments_new.sample(10))
```

	bctc	biến_động	bán	báo	báo_cáo	bên	bị	bổ_nhiệm	bổ_sung	cao	...	VTR	DP2	HLY	deep	title	return	ma_ck	Date
30	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	thông_báo việc chốt danh_sách cổ_đông dự họp đ...	0.135266	HU3	2023-01-18
132	0	0	0	0	1	1	0	0	0	0	...	0	0	0	0	quyết_định hđqt chủ_trương thực_hiện hợp_đồng ...	0.057158	AMD	2023-02-06
74	0	0	0	0	1	0	1	0	0	0	...	0	0	0	0	báo_cáo tiến_độ khắc_phục tình_trạng chứng_kho...	-0.016427	LHG	2023-01-31
107	0	0	1	0	0	0	0	0	0	0	...	0	0	0	0	quyết_định việc đưa diện cảnh_báo nguyễn_văn_c...	0.005141	POM	2023-02-01
171	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	giấy chứng_nhận đăng_ký chứng_khoán	-0.034253	SAB	2023-02-10

Figure 2.25: Display the 'Sentiments' table

Chapter 3

PROPOSED METHODS

3.1 Dimensionality Reduction

Dimensionality reduction is a fundamental technique in the field of machine learning and data analysis aimed at mitigating the "curse of dimensionality." As datasets grow in size and complexity, the number of features or variables often increases, posing challenges for computational efficiency and model performance.

Dimensionality reduction methods seek to transform high-dimensional data into a lower-dimensional representation while preserving essential information. Principal Component Analysis (PCA), Linear Discriminant Analysis (LDA), t-Distributed Stochastic Neighbor Embedding (t-SNE), and Uniform Manifold Approximation and Projection (UMAP) are among the popular techniques used for this purpose. By reducing the number of dimensions, these methods not only enhance computational efficiency but also aid in visualization and interpretation, ultimately improving the performance of machine learning models and facilitating a more meaningful analysis of complex datasets.

Among these techniques, PCA is most used for unsupervised data compression and LDA is a method used for supervised dimensionality reduction - which I'm going to discuss later in this post.

Principal component analysis (PCA) aims to transform high-dimensional data into a lower-dimensional space by identifying the principal components, which are orthogonal vectors capturing the maximum variance in the original dataset. These principal components are linear combinations of the original features and are arranged in descending order of their variance. By selecting a subset of the top principal components, researchers can reduce the dimensionality of the data while retaining the most significant information.

3.1.1 Principal Component Analysis

Proposition 3.1

a PCA learns a representation that has lower dimensionality than the original input. $\mathbf{X} \approx \mathbf{ZW}$, where $\mathbf{Z} \in \mathbb{R}^{N \times p}$,$p < d$.
Optimization Problem:

$$\widehat{\mathbf{Z}}, \widehat{\mathbf{W}} = \arg\max_{\mathbf{Z},\mathbf{W}} \|\mathbf{X} - \mathbf{ZW}\|_{\mathcal{F}}^2$$

b It also learns a representation whose elements have no linear correlation with each other

Suppose we have a dataset $\mathcal{X} = \{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, ..., \mathbf{x}^{(N)}\}$ with $\mathbf{x}^{(i)} \in \mathbb{R}^d$. If d is sufficiently large, we aim to compress $\mathcal{X}$ into data points with a dimension p such that $p << d$.
Let $\mathcal{B} = \{\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_d\}$ be an orthonormal basis of $\mathbb{R}^d$ (the right singular vectors in the decomposition $\mathbf{X} = \mathbf{U\Sigma V}^\top$). For each $\mathbf{x}^{(i)}$, there exists a unique linear representation as follows:

$$\mathbf{x}^{(i)} = \sum_{j=1}^{d} z_j^{(i)} \mathbf{v}_j \tag{3.1}$$

We seek an approximation:

$$\mathbf{x}^{(i)} \approx \sum_{j=1}^{p} z_j^{(i)} \mathbf{v}_j \tag{3.2}$$

Or

$$\mathbf{X} \approx \mathbf{ZV}^\top \tag{3.3}$$

where $\mathbf{Z} \in \mathbb{R}^{N \times p}$ and $\mathbf{V} \in \mathbb{R}^{d \times p}$.
Note that each column of $\mathbf{V}$ is a $\mathbf{v}_k$, hence $\mathbf{V}^\top\mathbf{V} = \mathbf{I}_d$.
We minimize the approximation error:

$$(\widehat{\mathbf{V}}, \widehat{\mathbf{A}}) = \arg\min_{\mathbf{V},\mathbf{Z}} \left\{ \left\| \mathbf{ZV}^\top - \mathbf{X} \right\|_{\mathcal{F}}^2 \right\} \tag{3.4}$$

We have: $\nabla_{\mathbf{Z}} \left\| \mathbf{ZV}^\top - \mathbf{X} \right\|_{\mathcal{F}}^2 = 2\mathbf{V}^\top(\mathbf{VZ}^\top - \mathbf{X}^\top) = \mathbf{0}_{p \times N} \Leftrightarrow \mathbf{Z} = \mathbf{XV}$
Substituting into (3.4), we obtain the optimization problem for a matrix variable $\mathbf{V}$ as follows:

$$\begin{aligned}
\widehat{\mathbf{V}} &= \arg\min_{\mathbf{V}} \left\{ \left\| \mathbf{X}\mathbf{V}\mathbf{V}^\top - \mathbf{X} \right\|_{\mathcal{F}}^2 \right\} \\
&= \arg\min_{\mathbf{V}} \left\{ \mathrm{Tr}((\mathbf{V}\mathbf{V}^\top\mathbf{X}^\top - \mathbf{X}^\top)(\mathbf{X}\mathbf{V}\mathbf{V}^\top - \mathbf{X})) \right\} \\
&= \arg\min_{\mathbf{V}} \left\{ \mathrm{Tr}\left[\mathbf{X}^\top\mathbf{X} - \mathbf{X}^\top\mathbf{X}\mathbf{V}\mathbf{V}^\top - \mathbf{V}\mathbf{V}^\top\mathbf{X}^\top\mathbf{X} + \mathbf{V}\mathbf{V}^\top\mathbf{X}^\top\mathbf{X}\mathbf{V}\mathbf{V}^\top \right] \right\} \\
&= \arg\max_{\mathbf{V}} \left\{ \mathrm{Tr}(\mathbf{V}^\top\mathbf{X}^\top\mathbf{X}\mathbf{V}) \right\} \\
&= \arg\max_{\mathbf{V}} \{ \mathrm{Tr}(\mathbf{V}^\top \mathbb{E}\left[\mathbf{X}^\top\mathbf{X} \right] \mathbf{V}) \} \\
&= \arg\max_{\mathbf{V}} \left\{ \mathrm{Tr}\left[\mathbf{\Sigma}^2 \right] \right\}
\end{aligned}$$

Note that the final equality occurs because we can always assume $\mathbb{E}(\mathbf{X}) = 0$.

Since the covariance matrix $\mathrm{Var}[\mathbf{X}]$ is symmetric and positive semi-definite, it has a unique orthogonal diagonalization:

$$\mathrm{Var}[\mathbf{X}] = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^\top$$

, where $\mathbf{\Lambda} = \mathrm{diag}\{\lambda_1, ..., \lambda_d\}$ with $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_d \geq 0$.

The solution to the main optimization problem is the top p columns of $\mathbf{V}$ (corresponding to the p largest eigenvalues of $\mathrm{Var}[\mathbf{X}]$).

Singular Value Decomposition

Proposition 3.2

KHANG Suppose $\{\mathbf{v}_1, .., \mathbf{v}_n\}$ is an orthonormal basis $\mathbb{R}^n$ consisting eigenvectors of $\mathbf{A}^\top\mathbf{A}$, arranged so that the corresponding eigenvalues of $\mathbf{A}^\top\mathbf{A}$ satisfy $\lambda_1 \geqslant ... \geqslant \lambda_n \geqslant 0$, and suppose $\mathbf{A}$ has r nonzero singular values.

Then $\{\mathbf{A}\mathbf{v}_1, ...\mathbf{A}\mathbf{v}_r\}$ is an orthogonal basis for $\mathrm{Col}\{\mathbf{A}\}$, and $\mathrm{Rank}(\mathbf{A}) = \dim(\mathrm{Col}(A)) = r$.

Proof. Firstly, we have to demonstrate that $\{\mathbf{A}\mathbf{v}_1, ...\mathbf{A}\mathbf{v}_r\}$ is orthogonal. Indeed, due to $\mathbf{A}\mathbf{v}_i \cdot \mathbf{A}\mathbf{v}_j = \mathbf{v}_i^\top\mathbf{A}^\top\mathbf{A}\mathbf{v}_j = \lambda_j\mathbf{v}_i \cdot \mathbf{v}_j = 0, \forall 1 \leq i \neq j \leq r$.

Finally, it's easy to realize that $\mathrm{Span}\{\mathbf{A}\mathbf{v}_1, ...\mathbf{A}\mathbf{v}_r\} = \mathrm{Col}(\mathbf{A})$. ■

Proposition 3.3

Singular Value Decomposition[a]

$$\mathbf{A} = \mathbf{U}_{m\times m}\mathbf{\Sigma}_{m\times n}\mathbf{V}_{n\times n}^\top$$

[a] $\mathbf{\Sigma} := \begin{bmatrix} \mathrm{diag}\{\sigma_1, .., \sigma_r\} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}$

Proof. For $1 \leq i \leq r$, we set: $\mathbf{u}_i = \frac{1}{\sigma_i}\mathbf{A}\mathbf{v}_i = \frac{1}{\|\mathbf{A}\mathbf{v}_i\|}$. Then $\{\mathbf{u}_1, .., \mathbf{u}_r\}$ is a orthonormal basis for Col($\mathbf{A}$). We can add $\mathbf{u}_{r+1}, .., \mathbf{u}_m$ such that $\{\mathbf{u}_1, .., \mathbf{u}_m\}$ is orthonormal basis for $\mathbb{R}^m$.
Build $\mathbf{U} := [\mathbf{u}_1|\mathbf{u}_2|...|\mathbf{u}_m]$, $\mathbf{V} := [\mathbf{v}_1|\mathbf{u}_2|...|\mathbf{u}_n] \Rightarrow \mathbf{A}\mathbf{V} = \mathbf{U}\mathbf{\Sigma}$. ■

3.2 Naive Bayes

Naive Bayes (NB) is a classification algorithm based on computing probabilities applying the Bayes theorem. This algorithm belongs to the group of supervised learning. This is the classification approach according to the probabilistic model. It predicts the probability that a new object belongs to a member of the class in question. Naive Bayes can be trained very effectively. With one training over the data, it computes the conditional probability distribution of each feature for each label. For prediction, it applies Bayes theorem to calculate the conditional probability distribution of each label for an observation. According to Bayes theorem, we have

$$\mathbb{P}(y|x) = \frac{\mathbb{P}(x|y)\mathbb{P}(y)}{\mathbb{P}(x)} \tag{4}$$

Suppose we split an event x into d different components $x_1, x_2, \ldots, x_d$. Naive Bayes, as the name suggests, relies on the naive assumption that $x_1, x_2, \ldots, x_d$ are independent and identically distributed from the given probability distribution components. From there we can calculate as in Equation (5):

$$\mathbb{P}(\mathbf{x}|y) = \mathbb{P}(x_1|y)\mathbb{P}(x_2|y)\ldots\mathbb{P}(x_n|y) \tag{5}$$

Hence, we have Equation (6):

In practice, it is rare to find data whose components are completely independent of each other. However

(3.5)

3.3 Logistic Regression

Logistic Regression (LR) is a process of modeling the probability of a discrete outcome for an input variable. Logistic regression is a popular method for predicting a classification problem. This is a special case of Generalized Linear models which are used to predict the probability of the outcomes.

The most important difference between naive Bayes and logistic regression is that logistic regression is a discriminative classifier while naive Bayes is a generative classifier. These are two very different frameworks for how to build a machine learning model. Consider a visual metaphor: imagine we're trying to distinguish dog images from cat images. A generative model would have the goal of understanding what dogs look like and what cats look like. You might literally ask such a model to 'generate', i.e., draw, a dog. Given a test image, the system then

asks whether it's the cat model or the dog model that better fits (is less surprised by) the image, and chooses that as its label. A discriminative model, by contrast, is only trying to learn to distinguish the classes (perhaps without learning much about them). So maybe all the dogs in the training data are wearing collars and the cats aren't. If that one feature neatly separates the classes, the model is satisfied. If you ask such a model what it knows about cats all it can say is that they don't wear collars [JM09].

a. A feature representation of the input. For each input observation $x^{(i)}$, this will be a vector of features $[x_1; x_2; \ldots; x_n]$. We will generally refer to feature i for input $x^{(j)}$ as $x_i^{(j)}$, sometimes simplified as x_i, but we will also see the notation f_i, $f_i(x)$, or, for multiclass classification, $f_i(c; x)$.

b. A classification function that computes $\hat{y}$, the estimated class, via $p(y|x)$. In the next section, we will introduce the sigmoid and softmax tools for classification.

c. An objective function for learning, usually involving minimizing error on training examples. We will introduce the cross-entropy loss function.

d. An algorithm for optimizing the objective function. We introduce the stochastic gradient descent algorithm.

Logistic regression has two phases: training, where we train the system (specifically the weights w and b) using stochastic gradient descent and the cross-entropy loss, and test, where given a test example x, we compute $p(y|x)$ and return the higher probability label $y = 1$ or $y = 0$.

3.4 K-Nearest Neighbour

Lazy Learning

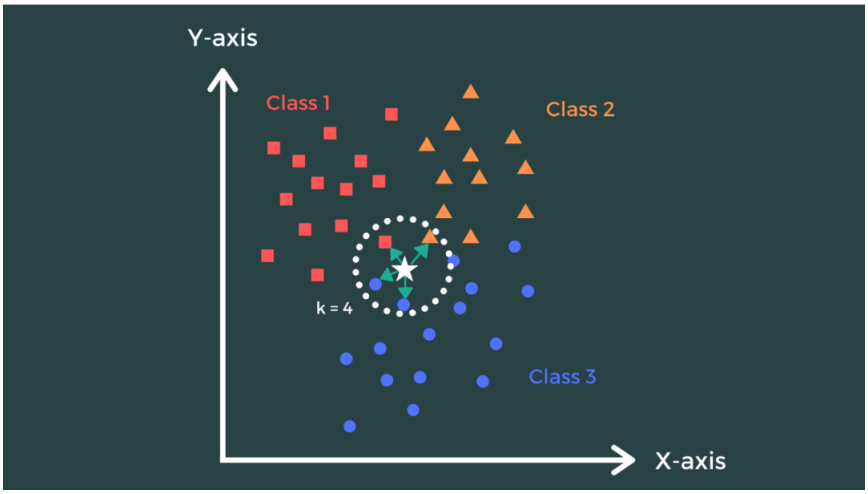


Figure 3.1: Enter Caption

Sure, K-Nearest Neighbors (K-NN) is a simple and intuitive algorithm used for both classification and regression tasks in machine learning.

How K-NN Works:

For classification:

a. **Training Phase**: K-NN stores all available cases and their class labels.

b. **Prediction Phase**: Given a new, unlabeled observation, the algorithm identifies the K closest labeled data points (nearest neighbors) based on a chosen distance metric (such as Euclidean, Manhattan, etc.).

Mathematical Basis:

The prediction or classification for a new data point is based on the majority class among its K nearest neighbors.

Mathematical Explanation:

a. **Distance Calculation**: For each new data point, the algorithm calculates its distance to all other points in the dataset.

- Euclidean distance between two points ($p1$ and $p2$) in a 2-dimensional space:

$$\text{Euclidean Distance} = \sqrt{(p1_x - p2_x)^2 + (p1_y - p2_y)^2}$$

- This formula can be extended to higher dimensions.

b. **Voting Mechanism**: Once the K nearest neighbors are identified, the algorithm counts the occurrences of each class among these neighbors.

- In classification, the class that occurs most frequently among the K neighbors is assigned to the new data point.

c. **Regression**: In regression tasks, the algorithm predicts the average of the K nearest neighbors' target values.

3.5 Tree-based Algorithms

3.5.1 Decision Tree

Decision Tree (DT) is a popular method of classification and regression methods. Decision trees are widely used because they are easy to interpret, handle categorical features, extend

to multiclass classification implementations, do not require object scaling, and can capture object interactions and nonlinear. A decision tree is a greedy algorithm that performs recursive binary partitioning of the feature space. The tree predicts the same label for each bottom (leaf) partition. Each partition is chosen greedily by choosing the best split from a set of possible splits, to maximize information gain at a tree node. In other words, the split is selected at each tree node selected from the set as shown in (3): arg max x IG(D, s) (3) Where IG(D, s) is the information gain when a division s is applied to the data set D Entropy: $\mathbf{E}(S) = -\mathbb{E}_{X \sim p_{data}}(\log \mathbb{P}(X))$
Information Gain: $\mathrm{IG}(S|d) = \mathbf{E}(S) - \mathbb{E}_{s \sim S|d}(\mathbf{E}(s))$
GINI: $\mathbf{GINI}(S) = 1 - \sum_{i=1}^{C} \mathbb{P}_i <= 0.5$

3.5.2 Random Forest

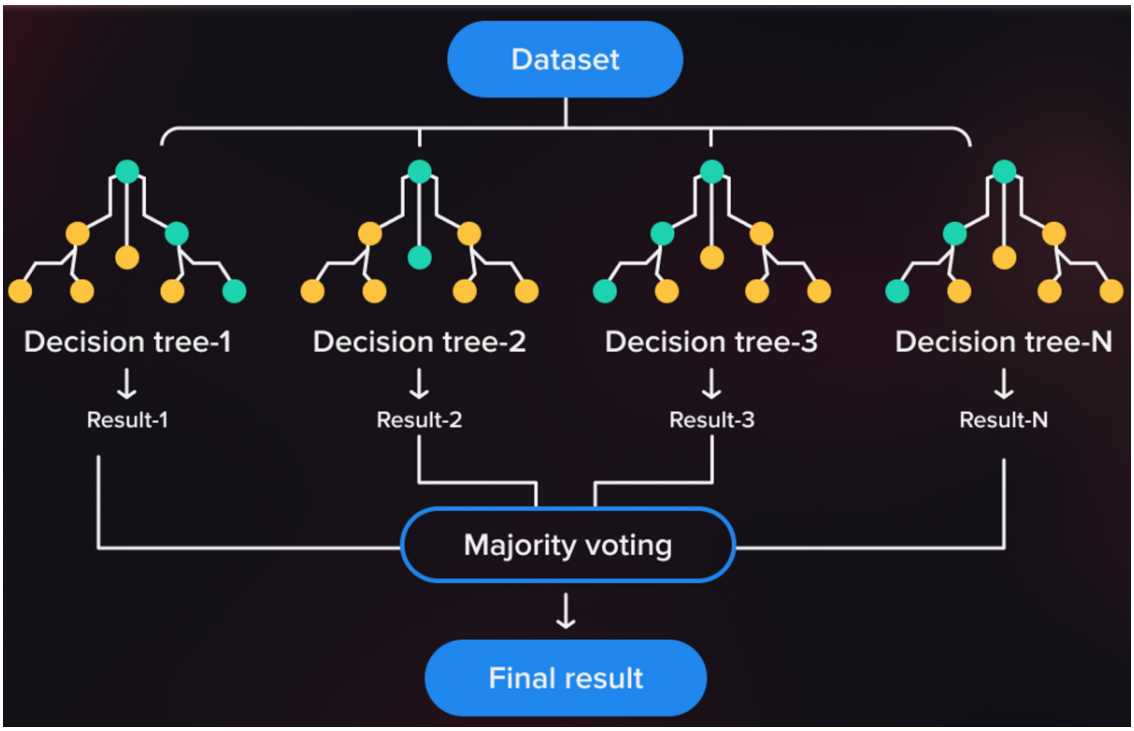


Figure 3.2: Enter Caption

Bagging **H**
Random Forest (RF) is a popular group of classification and regression methods. Random forest combines many decision trees, but each decision tree will be different (with random factor). The prediction results are then aggregated from the decision trees, the output's label is the most predicted (majority) label of all decision trees to reduce the risk of overfitting. The spark.ml implementation supports random forests for binary and multiclass classification and for regression, using both numerical and categorical features.

3.5.3 Adaptive Boost

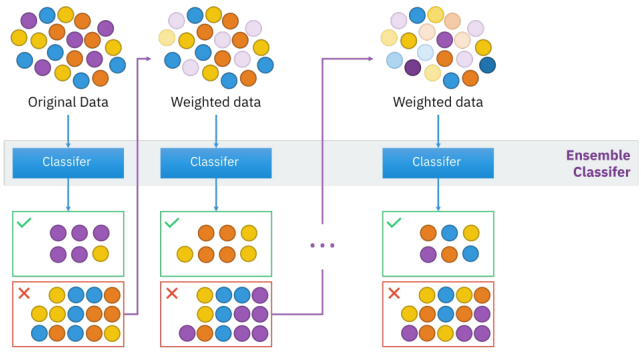


Figure 3.3: Enter Caption

AdaBoost Algorithm [FS95]

Proposition 3.4

a Initialize the observation weights $w_i = \frac{1}{N}, \forall i = \overline{1, N}$

b For $m = 1$ to M:

(a) Fit a classifier $\mathcal{T}_m(\mathbf{x}^{(i)})$ to the training data using weight $w_i, \forall i = \overline{1, N}$

(b) Compute *error*:

$$\text{Err}_m = \frac{\sum_{i=1}^{N} w_i \mathbb{I}(y^{(i)} \neq \mathcal{T}_m(\mathbf{x}^{(i)}))}{\sum_{i=1}^{N} w_i}$$

(c) Compute *amount of say for each Tree*:

$$\alpha_m = \log \frac{1 - \text{Err}_m}{\text{Err}_m}$$

(d) Update:

$$w_i \leftarrow w_i \exp \left[\alpha_m \mathbb{I}(y^{(i)} \neq \mathcal{T}_m(\mathbf{x}^{(i)})\right]$$

(e) Re-normalize w_i

c Oupute:

$$C(\mathbf{x}) = \arg\max_c \sum_{m=1}^{M} \alpha_m \mathbb{I}\left[\mathcal{T}_m(\mathbf{x}) = c\right]$$

SAMME Algorithm

Stagewise Additive Modeling using a Multi-class Exponential loss function

a Initialize the observation weights $w_i = \frac{1}{N}, \forall i = \overline{1, N}$

b For $m = 1$ to M:

(a) Fit a classifier $\mathcal{T}_m(\mathbf{x}^{(i)})$ to the training data using weight w_i

(b) Compute *error*:

$$\text{Err}_m = \frac{\sum_{i=1}^{N} w_i \mathbb{I}(y^{(i)} \neq \mathcal{T}_m(\mathbf{x}^{(i)}))}{\sum_{i=1}^{N} w_i}$$

(c) Compute *amount of say for each Tree*:

$$\alpha_m = \log \frac{1 - \text{Err}_m}{\text{Err}_m} + \log (C - 1)$$

(d) Update:

$$w_i \leftarrow w_i \exp \left[\alpha_m \mathbb{I}(y^{(i)} \neq \mathcal{T}_m(\mathbf{x}^{(i)})\right]$$

(e) Re-normalize w_i

c Oupute:

$$C(\mathbf{x}) = \arg\max_c \sum_{m=1}^{M} \alpha_m \mathbb{I}\left[\mathcal{T}_m(\mathbf{x}) = c\right]$$

Formularizing α_m

The hypothesis function: $f(\mathbf{x}) = \sum_{m=1}^{M} \alpha_m \mathcal{T}(\mathbf{x})$.
Overall exponential loss function: $\ell(f) = \mathbb{E}_{(\mathbf{x},y)\sim\hat{p}_{\text{data}}} \exp\{-yf(\mathbf{x})\}$
Expand the error function for the m^{th} tree: $\ell_m = \sum_{i=1}^{N} \exp\left\{y^{(i)} f_{m-1}(\mathbf{x}^{(i)})\right\} \exp\left\{-y^{(i)}\alpha_m \mathcal{T}_m(\mathbf{x}^{(i)})\right\}$
Set $w_i = \exp\left\{-y^{(i)} f_{m-1}(\mathbf{x}^{(i)})\right\}$. Note that w_i not depend on α_m and $\mathcal{T}_m$
$\Rightarrow \ell_m = \mathrm{e}^{-\alpha_m} \sum_{y^{(i)}=\mathcal{T}_m(\mathbf{x}^{(i)})} w_i + \mathrm{e}^{\alpha_m} \sum_{y^{(i)}\neq\mathcal{T}_m(\mathbf{x}^{(i)})} w_i$
Set $T_w = \sum_{i=1}^{N} w_i$ and $E_w = \sum_{y^{(i)}\neq\mathcal{T}_m(\mathbf{x}^{(i)})} w_i \Rightarrow \ell_m = \mathrm{e}^{-\alpha_m}(T_w - E_w) + \mathrm{e}^{\alpha_m} E_w$
$\Rightarrow \frac{\mathrm{d}}{\mathrm{d}\alpha_m} \text{Err}_m = -\alpha_m \mathrm{e}^{-\alpha_m}(T_w - E_w) + \alpha_m \mathrm{e}^{\alpha} E_w = 0 \Rightarrow \mathrm{e}^{2\alpha_m} = \frac{1 - \frac{E_w}{T_w}}{\frac{E_w}{T_w}}$

$$\Rightarrow \alpha_m = \frac{1}{2} \log \frac{1 - \text{Err}_m}{\text{Err}_m}$$

3.5.4 Gradient Boost

Gradient Boost: Least-Square Regression

a Initialize random prediction: $\mathcal{T}_0(\mathbf{x}) = \overline{y}$

b For $m = 1$ to M do:

(a) Compute pseudo-residual error: $\widetilde{y}_i = y^{(i)} - \mathcal{T}_{m-1}(\mathbf{x}^{(i)}), \forall i = \overline{1, N}$

(b) Find:

$$(\widehat{\alpha_m}, \widehat{w^{(m)}}) = \arg\min_{\alpha, w} \sum_{i=1}^{N} \left[\widetilde{y}_i - \alpha^{(m)} \mathcal{T}(\mathbf{x}^{(i)}; w^{(m)})\right]^2$$

(c) Accumulate:

$$f_m(\mathbf{x}) = f_{m-1}(\mathbf{x}) + \alpha^{(m)}\mathcal{T}^{(m)}(\mathbf{x}; w^{(m)})$$

End For

End Algorithm

3.5.5 Extreme Gradient Boost (xG Boost)

Overall Loss: $\ell = \mathbb{E}_{(\mathbf{x},y)\sim\sim\hat{p}_{\text{data}}}\left[\mathcal{L}(y, \widetilde{y} + v) + \frac{1}{2}\lambda v^2\right] + \gamma T$

Taylor Approximation: $\ell \approx \mathbb{E}\left[\underbrace{g}_{\frac{\partial}{\partial \widetilde{y}}\mathcal{L}(y,\widetilde{y})} v + \frac{1}{2}\underbrace{h}_{\frac{\partial^2}{\partial \widetilde{y}^2}\mathcal{L}(y,\widetilde{y})} v^2 + \frac{1}{2}\lambda v^2\right] + \gamma T$

$\Rightarrow \frac{\mathrm{d}}{\mathrm{d}v}\ell = \mathbb{E}\left[g + hv + \lambda v\right] = 0$

$$\Rightarrow v = \frac{-\mathbb{E}\left[g\right]}{\mathbb{E}\left[h\right] + \lambda}$$

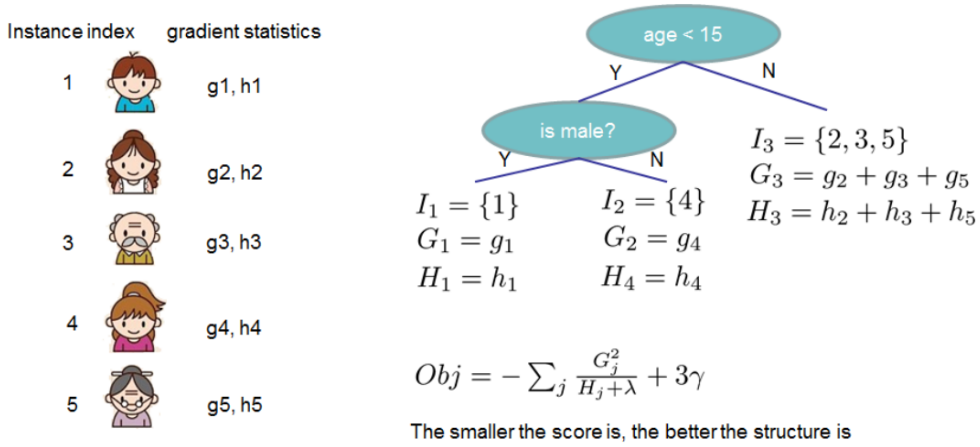


Figure 3.4: Enter Caption

For Classification Problem

Negative log-likelihood loss: $\mathcal{L}(y, \widetilde{y}) = -\left[y \log\left(\widetilde{y}\right) + (1-y)\log\left(1-\widetilde{y}\right)\right]$

$\Rightarrow \mathcal{L}(y, \log\left(\text{odds}\right)) = -y\log\left(\text{odds}\right) + \log\left[1 + \exp\left(\log\left(\text{odds}\right)\right)\right]$

But: $\begin{cases} g = \dfrac{\partial}{\partial \log\left(\text{odds}\right)}\mathcal{L}(y, \log\left(\text{odds}\right)) = -y + \dfrac{\exp\left\{\log\left(\text{odds}\right)\right\}}{1 + \exp\left\{\log\left(\text{odds}\right)\right\}} = -(y - \widetilde{y}) \\ h = \dfrac{\partial^2}{\partial\left\{\log\left(\text{odds}\right)\right\}^2}\mathcal{L}(y, \log\left(\text{odds}\right)) = \dfrac{\exp\left\{\log\left(\text{odds}\right)\right\}}{1 + \exp\left\{\text{odds}\right\}} \cdot \dfrac{1}{1 + \exp\left\{\log\left(\text{odds}\right)\right\}} = \widetilde{y}(1-\widetilde{y}) \end{cases}$

$$\Longrightarrow v = \frac{-\mathbb{E}\left[g\right]}{\mathbb{E}\left[h\right] + \lambda} \approx \frac{\#\sum \text{Residual}}{\#\sum \text{previous\_prob}(1 - \text{previous\_prob}) + \lambda}$$

3.6 Support Vector Machine

3.6.1 Hard-Margin Support Vector Machine

Denote the example in the dataset that is closest to the hyperplane by $\mathbf{x}_a$

Margin

$y^{(i)}(\mathbf{w} \cdot \mathbf{x}^{(i)} + b) \geq 1, \forall i = \overline{1, N}$

Definition 1. *Margin is the distance of the separating hyperplane to the closest examples in the dataset (assuming that the dataset is linearly separable).*

Hay $Margin := \min_i \left\{ \frac{y^{(i)}(\mathbf{w} \cdot \mathbf{x}^{(i)} + b)}{\|\mathbf{w}\|^2} \right\} = \|\mathbf{w}\|^{-2}$

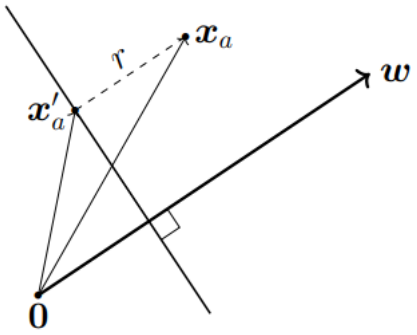


Figure 3.5: Vector addition to express distance to hyperplane: $\mathbf{x}_a = \mathbf{x}'_a + r\frac{\mathbf{w}}{\|\mathbf{w}\|}$

$$\mathbf{x}_a = \mathbf{x}'_a + r\frac{\mathbf{w}}{\|\mathbf{w}\|} \tag{3.6}$$

Refer to 3.6

Require

$$y^{(i)}(\mathbf{w} \cdot \mathbf{x}^{(i)} + b) \geq r, \forall i = \overline{1, N} \tag{3.7}$$

Proposition 3.5

Optimization Problem 1 $\max_{\mathbf{w},b,r} \underbrace{r}_{\text{margin}}$

subject to $\underbrace{y^{(i)}(\mathbf{w} \cdot \mathbf{x}^{(i)} + b) \geq r}_{\text{data fitting}}$, $\underbrace{\|w\| = 1}_{\text{normalization}}$, r>0

Traditional derivation of the Margin $r = \dfrac{1}{\|\mathbf{w}\|}$

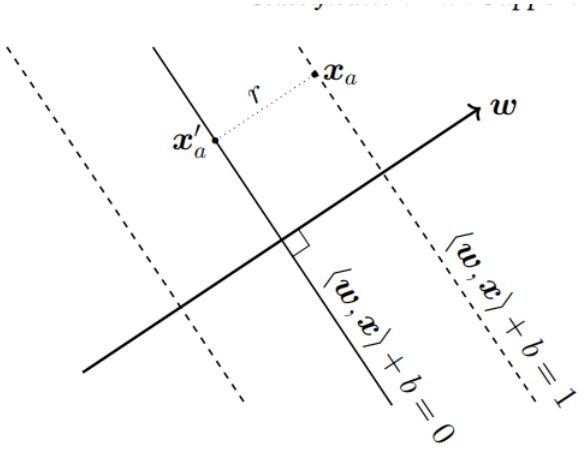


Figure 3.6: Derivation of the margin: $r = \dfrac{1}{\|\mathbf{w}\|}$

Of course, $\mathbf{w} \cdot \mathbf{x}'_a + b = 0 \Rightarrow \mathbf{w} \cdot \left(\mathbf{x}_a - r\dfrac{\mathbf{w}}{\|\mathbf{w}\|} \right) + b = 0 \Rightarrow \mathbf{w} \cdot \mathbf{x}_a + b - r\dfrac{\mathbf{w} \cdot \mathbf{w}}{\|w\|} = 0$

$\Rightarrow r = \dfrac{1}{\|\mathbf{w}\|}$

Why we can set the Margin $r = 1$**???**

Lemma 1. *3.6.1 is equivalent to scaling data, such that the margin is unity:*

$$\min_{\mathbf{w},b} \underbrace{\frac{1}{2}\|\mathbf{w}\|^2}_{margin} \text{ subject to } \underbrace{y^{(i)}(\mathbf{w} \cdot \mathbf{x}^{(i)} + b) \geq 1.}_{data\ fitting} \tag{3.8}$$

Proof. Kjak ■

Proposition 3.6

Optimization Problem 1

$$\left(\hat{\mathbf{w}}, \hat{b}\right) = \arg\min_{\mathbf{w},b} \left\{ \frac{1}{2}\|\mathbf{w}\|^2 \right\}$$

thỏa mãn: $1 - y^{(i)}(\mathbf{w} \cdot \mathbf{x}^{(i)} + b) \leq 0, \forall i = \overline{1, N}$

Chú ý: Bài toán 1 là **Quadratic Programing** và điều kiện Slater dễ dàng được kiểm tra thỏa mãn ⇒ Nghiệm của hệ điều kiện **Karush - Kuhn - Tucker** sẽ là nghiệm cho bài toán 1.

Lagrangian & Dual function

Lagrangian: $\mathcal{L}(\mathbf{w}, b, \mathbf{\Lambda}) := \frac{1}{2}\|\mathbf{w}\|^2 + \sum_{i=1}^{N} \lambda_i(1 - y^{(i)}(\mathbf{w} \cdot \mathbf{x}^{(i)} + b))$
Dual function: $\mathcal{D}(\mathbf{\Lambda}) := \min_{\mathbf{w},b} \mathcal{L}(\mathbf{w}, b, \mathbf{\Lambda})$
Để ý rằng $\mathcal{L}$ là một hàm convex.

Ta có: $\begin{cases} \nabla_{\mathbf{w}}\mathcal{L} = \mathbf{w} - \sum_{i=1}^{N} \lambda_i y^{(i)}\mathbf{x}^{(i)} = 0 \\ \nabla_b \mathcal{L} = -\sum_{i=1}^{N} \lambda_i y^{(i)} = 0 \end{cases} \Rightarrow \begin{cases} \mathbf{w} = \sum_{i=1}^{N} \lambda_i y^{(i)}\mathbf{x}^{(i)} \\ \sum_{i=1}^{N} \lambda_i y^{(i)} = 0 \end{cases}$

Thay vào, ta được công thức hàm đối ngẫu tường minh như sau:

$$\mathcal{D}(\mathbf{\Lambda}) = \sum_{i=1}^{N} \lambda_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{i=1}^{N} \lambda_i\lambda_j y^{(i)} y_j \mathbf{x}^{(i)} \cdot \mathbf{x}_j$$

Proposition 3.7

Dual problem cho hard-margin S.V.M

$$\hat{\mathbf{\Lambda}} = \arg\max_{\mathbf{\Lambda}} \mathcal{D}(\mathbf{\Lambda}) = \arg\max_{\mathbf{\Lambda}} \left\{ \sum_{i=1}^{N} \lambda_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{i=1}^{N} \lambda_i\lambda_j y^{(i)} y_j \mathbf{x}^{(i)} \cdot \mathbf{x}_j \right\}$$

thỏa mãn: $\begin{cases} \mathbf{\Lambda} \succeq \mathbf{0} \\ \sum_{i=1}^{N} \lambda_i y^{(i)} = 0 \end{cases}$

Hệ điều kiện Karush-Kunh-Tucker cho Hard-Margin S.V.M

Bộ $(\mathbf{w}, b)$ thỏa mãn hệ điều kiện sau chính là nghiệm của **bài toán 1** (mục 3.6.1):

- Stationary : $\mathbf{w} = \sum_{i=1}^{N} \lambda_i y^{(i)}\mathbf{x}^{(i)}; \sum_{i=1}^{N} \lambda_i y^{(i)} = 0$
- **Complementary Slackness**: $\lambda_i(1 - y^{(i)}(\mathbf{w} \cdot \mathbf{x}^{(i)} + b)) = 0, \forall i = \overline{1, N}$
- Primal Feasibility: $1 - y^{(i)}(\mathbf{w} \cdot \mathbf{x}^{(i)} + b) \leq 0, \forall i = \overline{1, N}$
- Dual Feasibility: $\lambda_i \geq 0, \forall i = \overline{1, N}$

Support Vector Set

Từ complementary slackness, kéo theo:

$$\forall i = \overline{1, N}, \lambda_i = 0 \quad \vee \quad \mathbf{w} \cdot \mathbf{x}^{(i)} + b = y^{(i)}$$

Tập $\{\mathbf{x}^{(i)} \mid \mathbf{w} \cdot \mathbf{x}^{(i)} + b = y^{(i)}\}$ được gọi là *support vector set*.
Đặt $\mathcal{R} := \{j \mid \lambda_j > 0\}$

Proposition 3.8

Nghiệm cho bài toán 1

$\mathbf{w} = \sum_{i \in \mathcal{R}} \lambda_i y^{(i)} \mathbf{x}^{(i)}$

$b = \frac{1}{|\mathcal{R}|} \sum_{i \in \mathcal{R}} \left(y^{(i)} - \mathbf{w} \cdot \mathbf{x}^{(i)} \right) = \frac{1}{|\mathcal{R}|} \sum_{i \in \mathcal{R}} \left(y^{(i)} - \sum_{j \in \mathcal{R}} \lambda_j y_j \mathbf{x}_j \cdot \mathbf{x}^{(i)} \right)$

$$\Rightarrow \text{class}(\mathbf{x}) = \text{sign} \left(\sum_{i \in \mathcal{R}} \lambda_i y^{(i)} \mathbf{x}^{(i)} \cdot \mathbf{x} - \frac{1}{|\mathcal{R}|} \sum_{i \in \mathcal{R}} \left(y^{(i)} - \sum_{j \in \mathcal{R}} \lambda_j y_j \mathbf{x}_j \cdot \mathbf{x}^{(i)} \right) \right).$$

3.6.2 Soft-Margin Support Vector Machine

Khi dataset là gần linearly seperable hoặc chứa noise thì hard-margin S.V.M sẽ hoạt động không hiệu quả.

Ta cần 1 slack variable (biến bù) cho mỗi mẫu: $\gamma^{(i)} \geq 0$, đo lường mức độ vi phạm cho phép của quan sát thứ i. Chúng ta có thể giả sử $\gamma^{(i)} := |\mathbf{w} \cdot \mathbf{x}^{(i)} + b - y^{(i)}|$

Soft constraints: $y^{(i)}(\mathbf{w} \cdot \mathbf{x}^{(i)} + b) \geq 1 - \gamma^{(i)}, \forall i = \overline{1, N}$

Từ đó, ta được bài toán tối ưu cho soft-margin S.V.M

Proposition 3.9

Optimization Problem 2

$$(\hat{\mathbf{w}}, \hat{b}, \hat{\mathbf{\Gamma}}) = \arg \min_{\mathbf{w}, \mathbf{b}, \mathbf{\Gamma}} \left\{ \frac{1}{2} \|\mathbf{w}\|^2 + \beta \sum_{i=1}^{N} \gamma^{(i)} \right\}$$

$$\text{subject to: } \begin{cases} 1 - y^{(i)}(\mathbf{w} \cdot \mathbf{x}^{(i)} + b) - \gamma^{(i)} \leq 0, \forall i = \overline{1, N} \\ -\gamma^{(i)} \leq 0, \forall i = \overline{1, N} \end{cases}$$

Lagrangian: $\mathcal{L}(\mathbf{w}, b, \mathbf{\Lambda}) := \frac{1}{2} \|\mathbf{w}\|^2 + \beta \sum_{i=1}^{N} \gamma^{(i)} + \sum_{i=1}^{N} \lambda_i (1 - y_i(\mathbf{w} \cdot \mathbf{x}_i + b) - \gamma^{(i)}) - \sum_{i=1}^{N} \mu_i \gamma^{(i)}$ với $\mathbf{\Lambda}, \boldsymbol{\mu} \succeq \mathbf{0}$

Ta có: $\begin{cases} \nabla_{\mathbf{w}} \mathcal{L} = \mathbf{w} - \sum_{i=1}^{N} \lambda_i y_i \mathbf{x}_i = 0 \\ \nabla_b \mathcal{L} = -\sum_{i=1}^{N} \lambda_i y_i = 0 \\ \nabla_{\gamma^{(i)}} = \beta - \lambda_i - \mu_i \end{cases} \Rightarrow \begin{cases} \mathbf{w} = \sum_{i=1}^{N} \lambda_i y_i \mathbf{x}_i \\ \sum_{i=1}^{N} \lambda_i y_i = 0 \\ \lambda_i = \beta - \mu_i \end{cases}$

$\Rightarrow$ Dual function: $\mathcal{D}(\mathbf{\Lambda}) = \sum_{i=1}^{N} \lambda_i - \frac{1}{2} \sum_{j=1}^{N} \sum_{i=1}^{N} \lambda_i \lambda_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j$

Proposition 3.10

Dual problem 2

$$\hat{\mathbf{\Lambda}} = \arg\max_{\mathbf{\Lambda}} \left\{ \sum_{i=1}^{N} \lambda_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \lambda_i \lambda_j y^{(i)} y^{(j)} \mathbf{x}^{(i)} \cdot \mathbf{x}_{(j)} \right\}$$

$$\text{subject to: } \begin{cases} \mathbf{0} \preceq \mathbf{\Lambda} \preceq \boldsymbol{\beta} \\ \sum_{i=1}^{N} \lambda_i y^{(i)} = 0 \end{cases}$$

Bộ $(\mathbf{w}, b, \mathbf{\Gamma})$ thỏa mãn bảng 2.1 (với mọi $i = \overline{1, N}$) chính là nghiệm của **bài toán 2** (xem mục

Table 3.1: Bảng hệ điều kiện K.K.T cho dual problem của soft-margin S.V.M

Name of condition	Equation and Inequation
Stationary	$\mathbf{w} = \sum_{i=1}^{N} \lambda_i y_i \mathbf{x}_i$
	$\sum_{i=1}^{N} \lambda_i y_i = 0$
Complementary Slackness	$\lambda_i (1 - y_i(\mathbf{w} \cdot \mathbf{x}_i + b) - \gamma^{(i)}) = 0$
	$\mu_i \gamma^{(i)} = 0$
Primal Feasibility	$1 - y_i(\mathbf{w} \cdot \mathbf{x}_i + b) - \gamma^{(i)} \leq 0$
	$-\gamma^{(i)} \leq 0$
Dual Feasibility	$\lambda_i \geq 0$
	$\mu_i \geq 0$

Chú ý (*box constraint*): $\lambda_i \in [0; \beta]$

Support vector set cho Soft-margin S.V.M

$$\text{Đặt: } \begin{cases} \mathcal{R} := & \{i \mid \lambda_i > 0\} \\ \mathcal{S} := & \{\mathbf{x}_i \mid y_i(\mathbf{w} \cdot \mathbf{x}_i + b) = 1 - \gamma^{(i)}\} \\ \mathcal{R}_1 := & \{i \mid 0 < \lambda_i < \beta\} \\ \mathcal{S}_1 := & \{\mathbf{x}_i \mid \mathbf{w} \cdot \mathbf{x}_i + b = y_i\} \\ \mathcal{R}_2 := & \{i \mid \lambda_i = \beta\} \\ \mathcal{S}_2 := & \{\mathbf{x}_i \mid y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \leq 1\} \end{cases} \Rightarrow \begin{cases} \mathcal{S} = \{\mathbf{x}_i \mid i \in \mathcal{R}\} \\ \mathcal{S}_1 = \{\mathbf{x}_i \mid i \in \mathcal{R}_1\} \\ \mathcal{S}_2 = \{\mathbf{x}_i \mid i \in \mathcal{R}_2\} \end{cases}$$

Trong đó:

- $\mathcal{S}$ là support vector set, tập chứa các vector đóng góp vào quá trình tối ưu của loss function.
- $\mathcal{S}_1$ là tập các vector "nằm kề" lên lên rìa của margin.
- $\mathcal{S}_2$ là tập các vector nằm trong hoặc nằm trên hai boudaries.

Proposition 3.11

Solution for the problem 2

$$\mathbf{w} = \sum_{i \in \mathcal{R}} \lambda_i y^{(i)} \mathbf{x}^{(i)}$$

$$b = \frac{1}{|\mathcal{R}_1|} \sum_{i \in \mathcal{R}_1} \left(y^{(i)} - \mathbf{w} \cdot \mathbf{x}^{(i)} \right) = \frac{1}{|\mathcal{R}_1|} \sum_{i \in \mathcal{R}_1} \left(y^{(i)} - \sum_{j \in \mathcal{R}} \lambda_j y_j \mathbf{x}_j \cdot \mathbf{x}^{(i)} \right)$$

$$\Rightarrow \text{class}(\mathbf{x}) = \text{sign} \left(\sum_{i \in \mathcal{R}} \lambda_i y_i \mathbf{x}_i \cdot \mathbf{x} - \frac{1}{|\mathcal{R}_1|} \sum_{i \in \mathcal{R}_1} \left(y_i - \sum_{j \in \mathcal{R}} \lambda_j y_j \mathbf{x}_j \cdot \mathbf{x}_i \right) \right).$$

Non-constraint Optimization Problem cho Soft-margin S.V.M

Dễ thấy, **bài toán tối ưu 2** có thể có thể được đưa về dạng tối ưu không ràng buộc như sau:

$$(\hat{\mathbf{w}}, \hat{b}) = \arg\min_{\mathbf{w}, \mathbf{b}} \left\{ \beta \sum_{i=1}^{N} \max(0, 1 - y_i(\mathbf{w} \cdot \mathbf{x}_i + b)) + \frac{1}{2} \|\mathbf{w}\|^2 \right\}$$

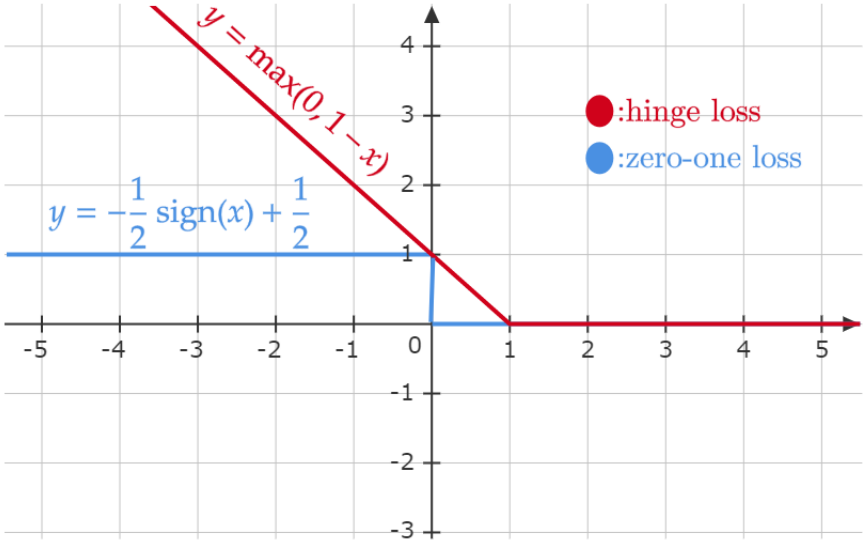


Figure 3.7: So sánh giữa hàm hinge-loss và hàm zero-one loss

3.7 Stacking

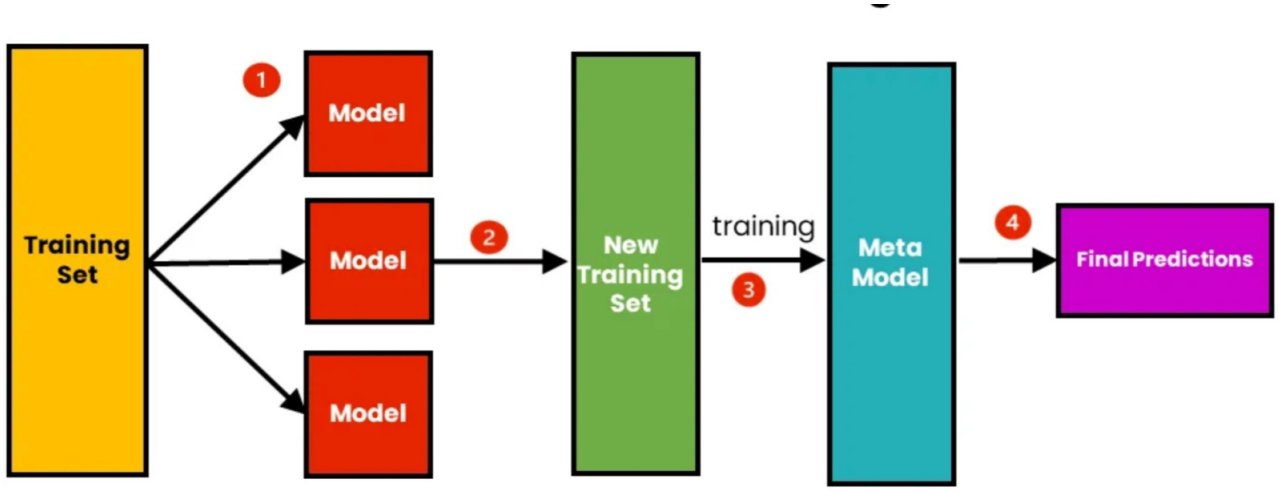


Figure 3.8: Enter Caption

Stacking, in the context of machine learning and deep learning, is an ensemble learning technique that involves combining multiple individual models (often referred to as base learners or base models) to create a more robust and accurate predictive model. The fundamental idea behind stacking is to leverage the diverse perspectives and strengths of different models to improve overall predictive performance.

3.7.1 Stacking Process of Stacking

1. **Base Models:** Initially, various base models are trained on the same dataset. These models can belong to different families (e.g., decision trees, support vector machines, neural networks) or might be different configurations of the same type of model.
2. **Meta Model:** After training the base models, a meta-model or a meta-learner is employed to learn from the predictions made by these base models. The predictions of the base models serve as the input features for the meta-model.
3. **Training the Meta Model:** The meta-model is trained on these predictions, learning how to combine or weigh them optimally to make the final prediction. It learns the relationships between the base model predictions and the actual target values to create a more accurate and robust predictive model.
4. **Making Predictions:** During the prediction phase, the trained base models are used to generate predictions on new, unseen data. These predictions are then fed into the trained meta-model to produce the final prediction.

3.8 Pretrained Language Representation Learning Models

3.9 BERT

BERT, which stands for Bidirectional Encoder Representations from Transformers, is a state-of-the-art natural language processing (NLP) model introduced by Google AI in 2018 [Dev+18]. It is designed to capture contextualized representations of words and sentences in a way that enables it to understand the meaning of language more effectively.

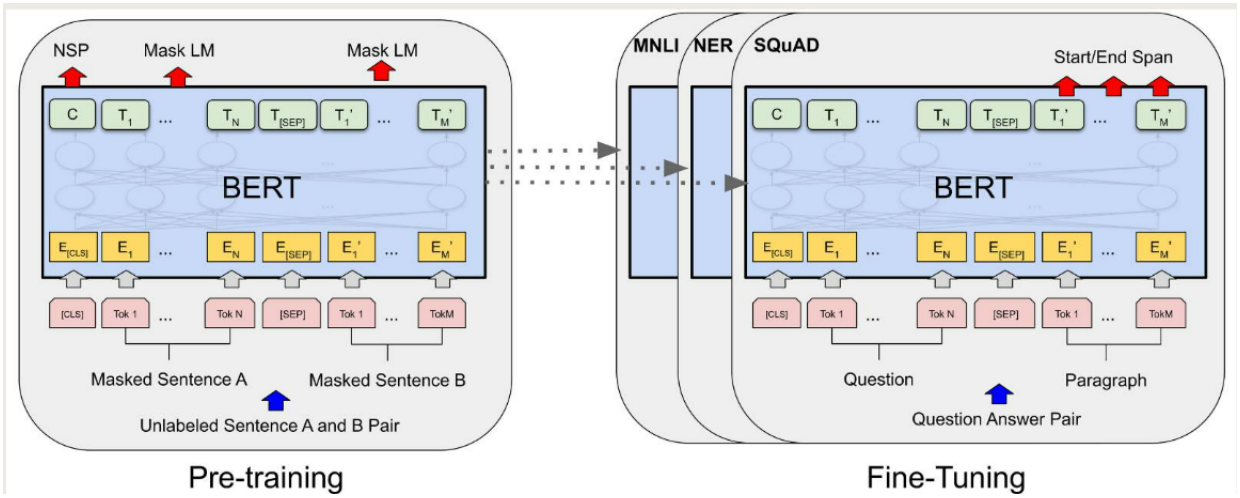


Figure 3.9: Enter Caption

3.9.1 Architecture

- BERT is built on the Transformer architecture, which is a neural network architecture introduced by Vaswani et al. in the paper "Attention Is All You Need." [Vas+17]. The Transformer architecture utilizes a mechanism called self-attention to weigh the importance of different words in a sentence relative to each other.

- The core idea behind BERT is that it learns contextualized word representations by considering both the left and right context of each word in a sentence. It does so by training on a masked language model (MLM) objective and a next sentence prediction (NSP) objective.

- Architecture: BERT consists of a stack of Transformer layers. Each layer contains a multi-head self-attention mechanism and position-wise feedforward networks. The self-attention mechanism allows BERT to weigh the importance of different words in a sentence based on their relationships, enabling it to capture context effectively.

3.9.2 How BERT works?

Tokenization

BERT tokenizes input text into smaller units such as words or subwords. These tokens are then mapped to fixed-size vectors through an embedding layer.

Pretraining

BERT is pre-trained on a massive amount of text data using the MLM objective. During this pretraining phase, a certain percentage of the tokens in each input sentence are randomly selected and replaced with a special [MASK] token. The model's goal is to predict the original identity of the masked words based on the context provided by the surrounding words. This bidirectional approach allows BERT to capture rich contextual information.

Fine-Tuning

After pretraining, BERT can be fine-tuned on specific downstream NLP tasks such as text classification, named entity recognition, question answering, and more. During fine-tuning, task-specific layers are added on top of the pre-trained BERT model, and the entire model is fine-tuned on the task's labeled data.

BERT is a groundbreaking NLP model that leverages a bidirectional Transformer architecture to understand and represent language context more effectively. Its pretrained representations have won many competitions with several downstream tasks in recent years.

3.10 PhoBERT

Pre-trained Language Model for Vietnamese PhoBERT, standing for Vietnamese BERT, is a state-of-the-art language model specifically tailored for the Vietnamese language [NT20]. Introduced by VinAI Research, PhoBERT aims to capture the nuances and intricacies of Vietnamese, a language with complex syntax and semantics.

3.10.1 Architecture

Much like BERT [Dev+18], PhoBERT is built upon the Transformer architecture. It employs the same self-attention mechanism to weigh the importance of different words in a sentence relative to one another, tailored for Vietnamese. PhoBERT is pre-trained on a massive Vietnamese corpus, utilizing both masked language modeling (MLM) and next sentence prediction (NSP) objectives, similar to BERT. However, the specific tokenization strategy and pretraining techniques are adjusted for the Vietnamese language.

Figure 3.10: Enter Caption

PhoBERT also consists of a stack of Transformer layers, each containing a multi-head self-attention mechanism and position-wise feed-forward networks, specifically optimized for Vietnamese.

3.10.2 How PhoBERT works?

Tokenization

One of the distinguishing features of PhoBERT is its tokenization process. Vietnamese is a language where spaces can occur inside a word, making traditional tokenization methods less effective. PhoBERT employs a specialized tokenization approach tailored for Vietnamese (RDRSegmentor from VnCoreNLP).

Pretraining

Similar to BERT, PhoBERT is pre-trained on a massive corpus of Vietnamese text. A percentage of tokens are masked, and the model is trained to predict these masked words using the surrounding context.

Fine-Tuning

After the pretraining phase, PhoBERT can be fine-tuned on specific Vietnamese NLP tasks like text classification, sentiment analysis, and named entity recognition. During this phase, task-specific layers are integrated with the pre-trained PhoBERT model, and the entire model is further trained on task-specific labeled data.

PhoBERT represents a significant advancement in Vietnamese NLP, leveraging the power of the Transformer architecture to capture the essence and context of the Vietnamese language

effectively. PhoBERT has shown state-of-the-art performance on various Vietnamese NLP benchmarks, especially on Sentiment Analysis and Hate Speech Detection tasks.

3.11 FinBERT

Chapter 4 EXPERIMENTS & EVALUATION

4.1 Evaluation Metrics

4.1.1 The precision-recall(PR) Curve

Confusion matrix: 1 bảng chéo 2x2,

		Ground Truth	
		Positive	Negative
Model Claim	Positive	TP	FP
	Negative	FN	TN

$$\begin{bmatrix} TP & FP \\ FN & TN \end{bmatrix}$$

Proposition 4.1

Accuracy: $A \stackrel{\text{def}}{=} \dfrac{TP + TN}{TP + TN + FP + FN}$

Recall: $R \stackrel{\text{def}}{=} \dfrac{TP}{TP + FN}$

Precision: $P \stackrel{\text{def}}{=} \dfrac{TP}{TP + FP}$

False Alarm Rate $p_F = R\dfrac{1-P}{P}$

Miss Rate Rate $p_M = 1 - R$

F-Score[a]: $F_\beta \stackrel{\text{def}}{=} \dfrac{(\beta^2+1)PR}{\beta^2 P + R}$. For $\alpha = 1$, $F_1 = \frac{2PR}{P+R}$

[a] $F = \dfrac{1}{\frac{\alpha}{P} + \frac{1-\alpha}{R}}$, where $\beta^2 = \dfrac{1-\alpha}{\alpha}$

Precision & Recall: Two metrics are used to measure the performance of a retrieval system:

precision and recall.Precision is defined as the ratio of correctly classified positive samples (True Positive) to a total number of classified positive samples, see 7. The recall is calculated as the ratio between the numbers of Positive samples correctly classified as Positive to the total number of Positive samples. The recall measures the model's ability to detect positive samples. The higher the recall, the more positive samples detected as shown in (7) and (8) [2]

F1–score: F–score is defined as the weighted mean of precision and recall depending on the weight mean between precision and recall, see (10), when it is written as F–score, it usually means F1–score. F– score is also known as F–Measure. F1–score can have different metrics that give different weights for precision and recall.

Accuracy: Another metric defined as the proportion of true cases retrieved, both positive and negative, out of all retrieved cases. Accuracy is a weighted average of inverse precision and precision. The higher the Accuracy, the more efficient the model. However, this is not true for problems with unbalanced datasets.

4.1.2 Experimental Results

Experimental Results: RANDOM FOREST

	precision	recall	f1-score	support
0	0.79	0.69	0.74	55
1	0.72	0.81	0.77	54

Figure 4.1: Random Forest's results on y_train

	precision	recall	f1-score	support
0	0.67	0.47	0.55	30
1	0.41	0.61	0.49	18

Figure 4.2: random forest's results on y_test

Experimental Results: STACKING

	precision	recall	f1-score	support
0	0.35	0.38	0.37	55
1	0.31	0.28	0.29	54

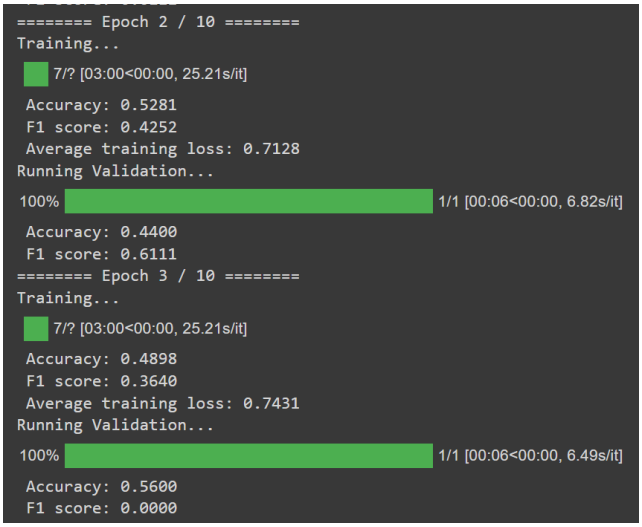
Figure 4.3: Stack_results on y_train

H

	precision	recall	f1-score	support
0	0.65	0.67	0.66	30
1	0.41	0.39	0.40	18

Figure 4.4: Stack_result on y_test

Experimental Results: Fine-tuned PhoBERT



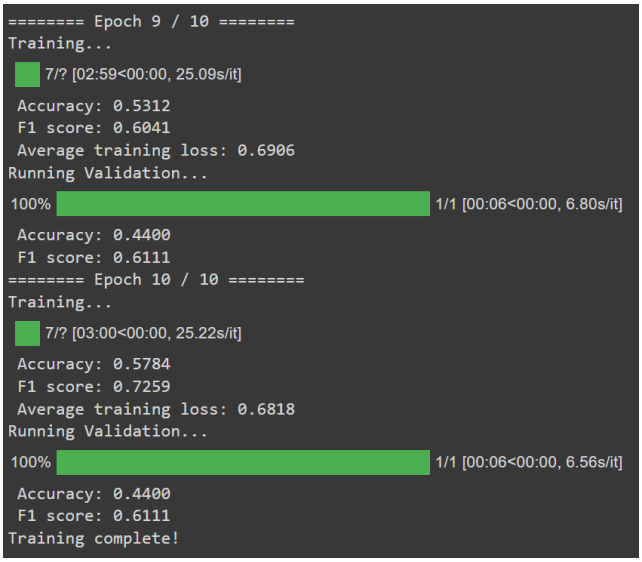


Figure 4.5: phoBERT_results after 10 epochs

Comment. PhoBERT>Stacking>RandomForest

Chapter 5 CONCLUSION

Proposition 5.1

a. In this work, we applied traditional ML methods and fine-tuned PhoBERT to Finacial S.A.

b. The ML methods suffer from the inability to represent the semantic information that results from a particular sequence of words, while the latter is often deemed as too "data-hungry" as it learns a much higher number of parameters.

c. But due to data-hungry and specialized language in the financial sector, so Finet-tuned PhoBERT just overcomes ML methods.

Appendix

A Source Code

✍ **Note.** Source Code ■

BIBLIOGRAPHY

[Cox58] David R Cox. "The regression analysis of binary sequences." In: *Journal of the Royal Statistical Society: Series B (Methodological)* 20.2 (1958), pp. 215–232.

[Alt92] Naomi S Altman. "An introduction to kernel and nearest-neighbor nonparametric regression." In: *The American Statistician* 46.3 (1992), pp. 175–185.

[FS95] Yoav Freund and Robert E. Schapire. "A desicion-theoretic generalization of on-line learning and an application to boosting." In: *Computational Learning Theory*. Ed. by Paul Vitányi. Berlin, Heidelberg: Springer Berlin Heidelberg, 1995, pp. 23–37. ISBN: 978-3-540-49195-8.

[Ho95] Tin Kam Ho. "Random decision forests." In: *Proceedings of 3rd international conference on document analysis and recognition*. Vol. 1. IEEE. 1995, pp. 278–282.

[Bre96] Leo Breiman. "Bagging predictors." In: *Machine learning* 24.2 (1996), pp. 123–140.

[Fri01] Jerome H Friedman. "Greedy function approximation: a gradient boosting machine." In: *Annals of statistics* (2001), pp. 1189–1232.

[Wu+08] Xindong Wu et al. "Top 10 algorithms in data mining." In: *Knowledge and information systems* 14.1 (2008), pp. 1–37.

[JM09] Dan Jurafsky and James H. Martin. *Speech and language processing : an introduction to natural language processing, computational linguistics, and speech recognition*. Upper Saddle River, N.J.: Pearson Prentice Hall, 2009. ISBN: 9780131873216 0131873210. URL: http://www.amazon.com/Speech-Language-Processing-2nd-Edition/dp/0131873210/ref=pd_bxgy_b_img_y.

[RK12] K.H. Rosen and K. Krithivasan. *Discrete Mathematics and Its Applications: With Combinatorics and Graph Theory*. McGraw-Hill Companies, 2012. ISBN: 9780070681880. URL: https://books.google.com.tr/books?id=IQ2jAgAAQBAJ.

[CG16] Tianqi Chen and Carlos Guestrin. "XGBoost: A Scalable Tree Boosting System." In: *CoRR* abs/1603.02754 (2016). arXiv: 1603.02754. URL: http://arxiv.org/abs/1603.02754.

[GBC16] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. http://www.deeplearningbook.org. MIT Press, 2016.

[Gér17] Aurélien Géron. *Hands-on machine learning with Scikit-Learn and TensorFlow: concepts, tools, and techniques to build intelligent systems*. Sebastopol, CA: O'Reilly Media, 2017. ISBN: 978-1491962299.

[Vas+17] Ashish Vaswani et al. "Attention Is All You Need." In: *CoRR* abs/1706.03762 (2017). arXiv: 1706.03762. URL: http://arxiv.org/abs/1706.03762.

[Dev+18] Jacob Devlin et al. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." In: *CoRR* abs/1810.04805 (2018). arXiv: 1810.04805. URL: http://arxiv.org/abs/1810.04805.

[Ara19] Dogu Araci. "FinBERT: Financial Sentiment Analysis with Pre-trained Language Models." In: *CoRR* abs/1908.10063 (2019). arXiv: 1908.10063. URL: http://arxiv.org/abs/1908.10063.

[FB19] L. Favero and P. Belfiore. *Data Science for Business and Decision Making*. Elsevier Science, 2019. ISBN: 9780128112168. URL: https://books.google.com.vn/books?id=ToC%5C_swEACAAJ.

[BBG20] P. Bruce, A. Bruce, and P. Gedeck. *Practical Statistics for Data Scientists: 50+ Essential Concepts Using R and Python*. O'Reilly Media, 2020. ISBN: 9781492072898. URL: https://books.google.com.vn/books?id=F2bcDwAAQBAJ.

[DFO20] Marc Peter Deisenroth, A. Aldo Faisal, and Cheng Soon Ong. *Mathematics for Machine Learning*. Cambridge University Press, 2020.

[NT20] Dat Quoc Nguyen and Anh Tuan Nguyen. "PhoBERT: Pre-trained language models for Vietnamese." In: *Findings of the Association for Computational Linguistics: EMNLP 2020*. Ed. by Trevor Cohn, Yulan He, and Yang Liu. Online: Association for Computational Linguistics, Nov. 2020, pp. 1037–1042. DOI: 10.18653/v1/2020.findings-emnlp.92. URL: https://aclanthology.org/2020.findings-emnlp.92.

[YUH20] Yi Yang, Mark Christopher Siy Uy, and Allen Huang. "FinBERT: A Pre-trained Language Model for Financial Communications." In: *CoRR* abs/2006.08097 (2020). arXiv: 2006.08097. URL: https://arxiv.org/abs/2006.08097.

[Cha21] S.H. Chan. *Introduction to Probability for Data Science*. Michigan Publishing, 2021. ISBN: 9781607857464. URL: https://books.google.com.vn/books?id=TKKkzgEACAAJ.

[Liu+21] Zhuang Liu et al. "FINBERT: A PRE-TRAINED FINANCIAL LANGUAGE REPRESENTATION MODEL FOR FINANCIAL TEXT MINING." In: *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*. IJCAI'20. Yokohama, Yokohama, Japan, 2021. ISBN: 9780999241165.