



# Portfolio

윤동진



# CONTENTS

**About Me**

---

**Realtime Trend Pipeline**

---

**Enhanced Index Tracking**

---

**Wind Power Forecasting**

---

**Court Decision Prediction**

---

# CONTENTS

---

About Me

# About Me



열정 있는 동료와 함께 성장하는 개발자입니다

- **Name** 윤동진(Dongjin Yoon)
- **Email** djyoon0223@gmail.com
- **Blog** <https://alchemine.github.io>
- **GitHub** <https://github.com/alchemine>
- **Docker Hub** <https://hub.docker.com/search?q=alchemine>
- **Skills**
  - Python (TensorFlow, PyTorch, scikit-learn, PyCaret, spaCy, seaborn, Dask, Numba, CUDA), R
  - Git, Docker, Poetry, SQL, Hadoop, Airflow, Spark, Hive, Kafka, Sphinx, Spotfire
- **Interests**
  - Refactoring, Parallel computing, Data visualization, Statistics
- **Work Experiences**
  - (주)디셈버앤파이낸셜 (2023/02 ~ 2023/06)
    - 금융데이터 빌더 운영 및 AI 투자로직 연구
  - (주)큐햇지 (2020/04 ~ 2022/02)
    - 머신러닝 투자 알고리즘 개발, Multi-node, multi-GPU distributed/parallel computing
- **Educations**
  - 인하대학교 전기컴퓨터공학과 석사 졸업 (2022/02)  
윤동진, 이주홍, 최범기, 송재원, “부분복제 지수 상향 추종을 위한 진화 알고리즘 기반 3단계 포트폴리오 선택 양상을 학습”,  
스마트미디어저널, 제10권, 제3호, 39-47쪽, 2021년 9월
  - 인하대학교 컴퓨터공학과 학사 졸업 (2019/02)

# About Me

## 동료 리뷰

### 동료 리뷰 – 이 부분은 계속 유지해주세요!

- 입사한지 얼마 되지 않음에도 불구하고 업무 내용을 빠르게 팔로업하여 인수인계에 많은 노력이 들지 않았습니다. 연구 진행 중에 모호한 점은 꼭 이해하려고 고민하고 끈질기게 리서치 하려는 자세가 연구자로서 강점이라고 생각됩니다. 기존에 관성적으로 진행되던 틀을 깨려는 시도 역시 연구자로서 좋은 태도로 보여 계속 유지하면 좋을 것으로 생각됩니다.
- 1. 맡은 업무를 책임감 있게 수행. 맡은 업무가 비교적 어려운 업무임에도 불구하고, 끝까지 책임감 있게 수행하려고 함. 도움이 필요하면, 팀 내 구성원들에게 적절하게 도움을 받고 업무를 잘 수행해 나감. 2. 타 팀과의 업무 능력. 이번 프로젝트에선 다른 팀과의 업무가 필수적임. 입사하지 얼마되지 않은 시점에서, 불편할 수도 있을 법한 타 팀과의 업무를 성공적으로 수행. 타 팀의 업무 결과를 잘 인수인계 받아 연구를 지속적으로 진행하고 있음.
- 금융 도메인 지식과 연구 경험이 많으셔서, 팀 멤버들의 연구에 대해 주시는 피드백 혹은 질문이 연구 방향 개선에 많은 도움이 되고 있습니다. 또한 맡으신 업무를 주도적으로 잘 진행하고 팀 회의 때 좋은 아이디어도 많이 내주셔서 팀에 많은 기여를 해주고 계신다고 생각합니다.
- 금융공학 관련한 배경지식이 탁월하시어 연구 중 보지 못했던 관점에서 항상 좋은 의견을 주십니다. 비단 업무적인 의견 이외에도 공학적인 솔루션 등에 관해서도 깊게 고민하고 계시며 원활하게 소통을 해주셔서 팀 전반이 함께 성장하는데 큰 도움이 되고 있습니다. 특히 현재 연구중이신 RIP모델의 데이터에 관해 고심하시며 기존 데이터에 관해서도 의견 주신 것들이 저의 팩터데이터 이해에도 큰 도움이 되었고 향후 연구에도 많은 참고가 될 것 같습니다.

### 동료 리뷰 – 이 부분은 더 노력이 필요해요!

- 대상자는 막 수습기간을 마친 관계로 평가할 부분이 적어, 아직까지는 개선할 부분을 파악하지 못 함. 타 팀과의 협업도 성공적으로 수행하였고, 이어서 자신의 업무도 별 다른 문제 없이 잘 수행하고 있음.
- 항상 많은 도움을 받고 있어 따로 말씀드릴 내용이 없습니다. 감사합니다.
- 잘 티는 못내고있지만 항상 공유주시는 정보나 의견들이 유익하고 도움이 많이 되었습니다.

---

# CONTENTS

---

Realtime Trend Pipeline



## Realtime Trend Pipeline

### 1. 프로젝트 소개

실시간 검색어를 크롤링하여 DB에 적재, 분석 후 분석 결과를 카카오톡 메시지로 전송하는 데이터 파이프라인

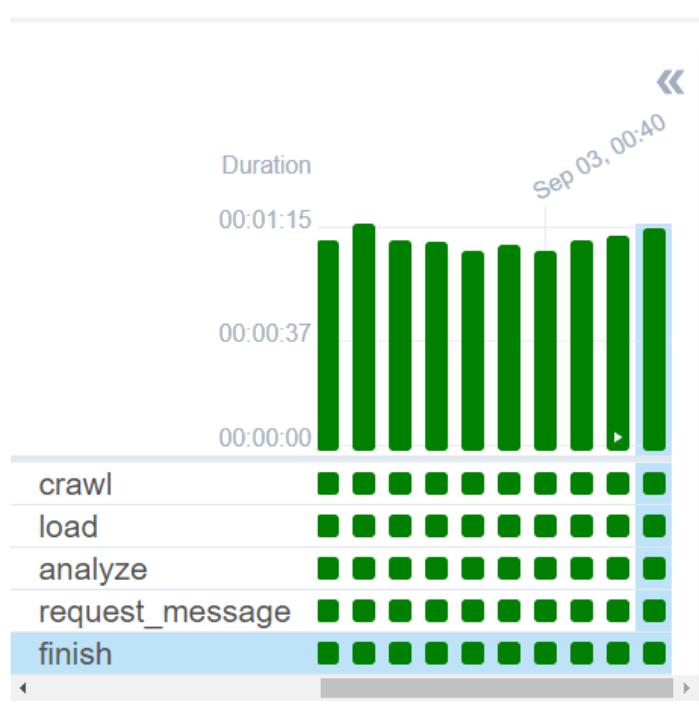
- GitHub [https://github.com/alchemine/realtime\\_trend\\_pipeline](https://github.com/alchemine/realtime_trend_pipeline)

### 2. 상세 설명

- Hadoop based Docker(<https://github.com/alchemine/hadoop-docker-cluster>) network 상에서 수행
- Components**
  - Workflow:** Airflow
  - Crawling:** Docker, Selenium
  - Loading:** Hadoop, Hive metastore
  - Analyzing:** Spark
  - Postprocessing:** Docker, Python
  - Message Request:** Kafka, KakaoAPI

## 1) Airflow Configuration

1. OS: Ubuntu22.04
2. Docker cluster: [alchemine/hadoop-docker-cluster](#)
3. Schedule: 10분 주기로 실행 (\*/10 \* \* \* \*)
4. Retries: 실패 시, 5분 후 1회 재시도 및 카카오톡 메시지 전송
5. Catchup: 과거 기간 동안 발생한 task instance를 backfill/catchup 하지 않음



```
default_args = {  
    'owner': 'alchemine',  
    'start_date': datetime(2023, 9, 2),  
    'retries': 1,  
    'retry_delay': timedelta(minutes=5),  
    'on_failure_callback': on_failure  
}  
  
with DAG(  
    dag_id='realtime_trend_pipeline',  
    schedule='*/10 * * * *',  
    # schedule='@once',  
    default_args=default_args,  
    tags=['realtime-trend'],  
    catchup=False  
) as dag:
```

## 2) Crawl

1. Data source: [시그널 실시간 검색어](#)
2. Selenium을 이용하여 실시간 검색어를 수집한 후, CSV format으로 local storage에 저장 (**DockerOperator**)

```
common_params = dict(
    task_id='crawl',
    container_name='realtime_trend_pipeline-crawl',
    api_version='auto',
    auto_remove='force',
    mounts=[
        Mount(source=os.getenv('HDFS_DIR'), target=PATH.hdfs, type='bind'), # source: host, target: container
        Mount(source=os.getenv('DFS_DIR'), target=PATH.dfs, type='bind'), # source: host, target: container
        Mount(source=os.getenv('DAGS_DIR'), target=PATH.dags, type='bind'), # source: host, target: container
    ],
    docker_url="tcp://docker-proxy:2375",
    mount_tmp_dir=False
)

cmd_install_python = """
    sudo apt update && apt install -yqq python3.8 python3.8-distutils
    curl https://bootstrap.pypa.io/get-pip.py -o /tmp/get-pip.py
    python3 /tmp/get-pip.py
    rm /tmp/get-pip.py
"""

cmd_install_python_deps = """
    python3 -m pip install pandas selenium
"""

cmd_execute_python = f"""
    cd {PATH.dags}
    python3 -m realtime_trend_pipeline.operators.crawl.topic_crawling {file_path}
"""
```

```
if image_type == 'unofficial':
    params = dict(
        image='joyzoursky/python-chromedriver:3.8-selenium',
        command=f"""
            bash -c "
                {cmd_install_python_deps}
                {cmd_execute_python}
            "
        """,
        **common_params
    )
elif image_type == 'official':
    params = dict(
        image='selenium/standalone-chrome:90.0.4430.85-chromedriver-90.0.4430.24',
        command=f"""
            bash -c "
                {cmd_install_python}
                {cmd_install_python_deps}
                {cmd_execute_python}
            "
        """,
        **common_params
    )
else:
    raise ValueError(image_type)

return DockerOperator(**params)
```

Time	Rank	Title
2023-09-02 17:09:00	1	부산 목욕탕 폭발
2023-09-02 17:09:00	2	연인 남궁민 안은진에
2023-09-02 17:09:00	3	1083회 로또 1등
2023-09-02 17:09:00	4	토트넘 벤리
2023-09-02 17:09:00	5	송가인 콘서트
2023-09-02 17:09:00	6	류현진
2023-09-02 17:09:00	7	영탁
2023-09-02 17:09:00	8	진짜가 나타났다
2023-09-02 17:09:00	9	손흥민
2023-09-02 17:09:00	10	달 태양 관측

# Realtime Trend Pipeline

## 3) Load

1. CSV file(local)을 hive metastore 내 external partitioned ORC table에 적재 (HiveOperator)

```
load = HiveOperator(  
    task_id='load',  
    hql=""">  
    -- 1. Create database  
    -- DROP DATABASE {PARAMS["db"]} CASCADE;  
    CREATE DATABASE IF NOT EXISTS {PARAMS["db"]}  
    LOCATION "{PARAMS['db_path']}";  
  
    -- 2. Create temporary CSV table  
    CREATE TEMPORARY TABLE {PARAMS["csv_table"]} (  
        `time` TIMESTAMP, rank INT, title STRING  
    )  
    ROW FORMAT DELIMITED  
    FIELDS TERMINATED BY ','  
    LINES TERMINATED BY '\n'  
    STORED AS TEXTFILE  
    TBLPROPERTIES('skip.header.line.count'=1');  
  
    LOAD DATA LOCAL INPATH "{PARAMS['local_csv_path']}"  
    OVERWRITE INTO TABLE {PARAMS["csv_table"]};  
  
    -- 3. Create ORC table  
    CREATE EXTERNAL TABLE IF NOT EXISTS {PARAMS["orc_table"]} (  
        `time` TIMESTAMP, rank INT, title STRING  
    )  
    PARTITIONED BY (hour STRING)  
    STORED AS ORC;  
  
    INSERT INTO TABLE {PARAMS["orc_table"]} PARTITION (hour="{{{ execution_date.strftime('%Y-%m-%d_%H-00-00') }}}")  
    SELECT * FROM {PARAMS["csv_table"]};  
  
    -- 4. Show table  
    SELECT * FROM {PARAMS["orc_table"]};  
    """,  
    hive_cli_conn_id='hive_cli_conn'  
)
```

partition
hour=2023-08-30_15-00-00
hour=2023-09-01_09-00-00
hour=2023-09-01_10-00-00
hour=2023-09-01_11-00-00
hour=2023-09-01_12-00-00
hour=2023-09-01_13-00-00

Partition unit: hour

```
hive> select * from realtime_trend_pipeline.topic limit 10;  
OK  
topic.time      topic.rank      topic.title      topic.hour  
2023-09-01 10:07:00      1      단식 농성 이재명에      2023-08-30_15-00-00  
2023-09-01 10:07:00      2      목욕탕 폭발 17명      2023-08-30_15-00-00  
2023-09-01 10:07:00      3      정동원      2023-08-30_15-00-00  
2023-09-01 10:07:00      4      성폭행 경찰 사이코패스      2023-08-30_15-00-00  
2023-09-01 10:07:00      5      미샤      2023-08-30_15-00-00  
2023-09-01 10:07:00      6      푸바오      2023-08-30_15-00-00  
2023-09-01 10:07:00      7      부동산 발표 위기설      2023-08-30_15-00-00  
2023-09-01 10:07:00      8      마이크로닷      2023-08-30_15-00-00  
2023-09-01 10:07:00      9      블루문      2023-08-30_15-00-00  
2023-09-01 10:07:00     10      10월2일 임시공휴일 확정      2023-08-30_15-00-00  
Time taken: 0.19 seconds, Fetched: 10 row(s)
```

## 4) Analyze

- 최근 1시간 내의 검색어들의 출현비율과 평균순위를 계산하여 CSV file(local)로 저장 (**SparkSubmitOperator**)

```

recent_topics = spark.sql(f"""
    SELECT
        rank,
        title
    FROM
        {table}
    WHERE
        hour IN (
            "{params['prev_hour']}",
            "{params['cur_hour']}"
        )
    ORDER BY
        time DESC
    LIMIT
        {params['n_iters_per_hour']}*{params['n_topics_per_iter']}
""")

raw_result = recent_topics \
    .groupby('title')\
    .agg(
        F.count('title').alias('count_title'),
        F.avg('rank').alias('avg_rank')
    )
    raw_result.createOrReplaceTempView("raw_result")

result = spark.sql(f"""
    SELECT
        title AS `최근 1시간 인기 검색어`,
        ROUND(count_title/{params['n_iters_per_hour'])*100, 1) AS `출현비율(%)`,
        ROUND(avg_rank, 1) AS `평균순위`
    FROM
        raw_result
    ORDER BY
        `출현비율(%)` DESC,
        `평균순위` ASC
    LIMIT
        {params['n_topics_per_iter']}
""")
    """

```

최근 1시간 인기 검색어	출현비율(%)	평균순위
부산 목욕탕 폭발	66.7	1
연인 남궁민 안은진에	66.7	2
1083회 로또 1등	66.7	3
송가인 콘서트	66.7	5
류현진	66.7	5.5
영탁	66.7	6.8
진짜가 나타났다	66.7	7.8
손흥민	66.7	8.8
토트넘 번리	50	4
달 태양 관측	50	10

### 5) Postprocess

- 분석 결과를 카카오톡 메시지로 보내기 적절한 형태로 처리하여 XCom에 push (**PythonOperator**)

```
def process_output_path_to_message(output_path: str):
    input_format = "%Y-%m-%d_%H-%M-%S"
    output_format = "%y/%m/%d %H:%M:%S"

    time      = datetime.strptime(basename(output_path), input_format).strftime(output_format)
    csv_path = glob(f"{output_path}/*.csv")[0] # *.csv is unique
    df       = pd.read_csv(csv_path)

    text = "[최근 1시간 인기 검색어] \n"
    text += f"기준시간: {time} \n"
    text += '\n'.join(f'{rank}: {title}' for rank, title in enumerate(df.iloc[:, 0], start=1))
    return text
```

#### XCom

Key	Value
return_value	[최근 1시간 인기 검색어] 기준시간: 23/09/03 16:30:00 1. 힙하게 2. 진짜가 나타났다 3. 돌싱글즈 4. 손흥민 5. 순항미사일 발사 훈련 6. 송가인 콘서트 7. 교사 숨진 채 8. 바이에른 뮌헨 9. 배준호 10. 연인

## 6) Request Message

1. 후처리된 메시지를 Kafka를 이용하여 카카오톡 메시지 전송 서버로 전송 (**ProduceToTopicOperator**)
2. 카카오톡 API가 메시지 전송 실패했을 경우를 대비하여, **group id를 지정하고 offset/commit을 관리하여 데이터 소실을 방지**

```
request_message = ProduceToTopicOperator(
    task_id='request_message',
    kafka_config_id='kafka_default',
    topic=PARAMS['topic_request_message'],
    producer_function=request_message_producer,
    producer_function_kwargs=dict(
        text="{{ ti.xcom_pull(task_ids='postprocess') }}",
        output_path=PARAMS['output_path']
    )
)
```

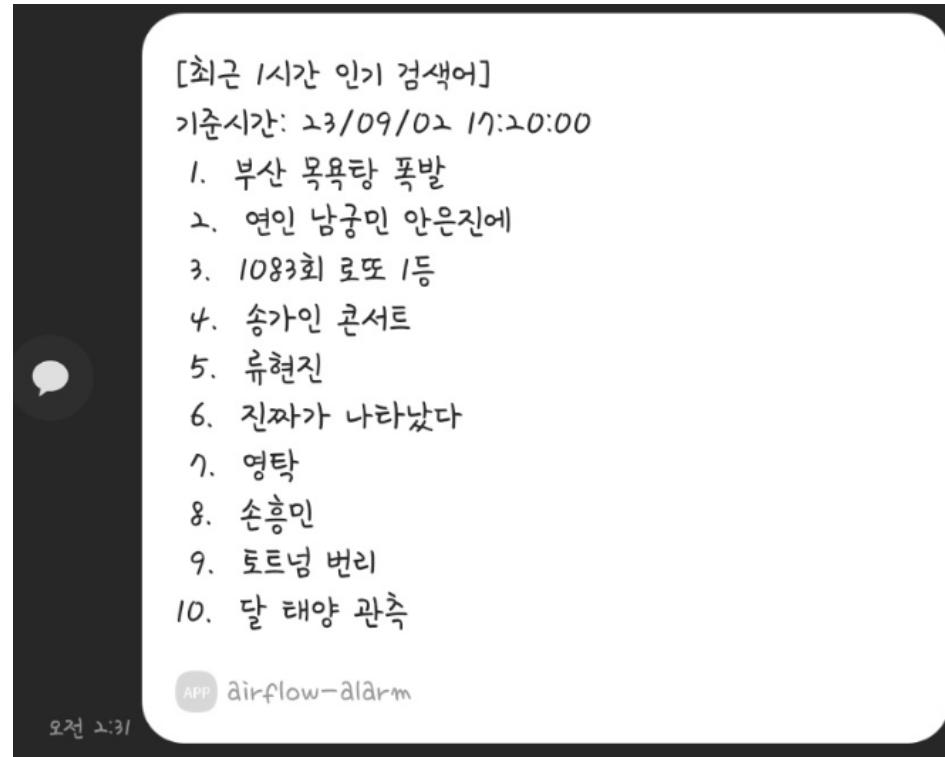
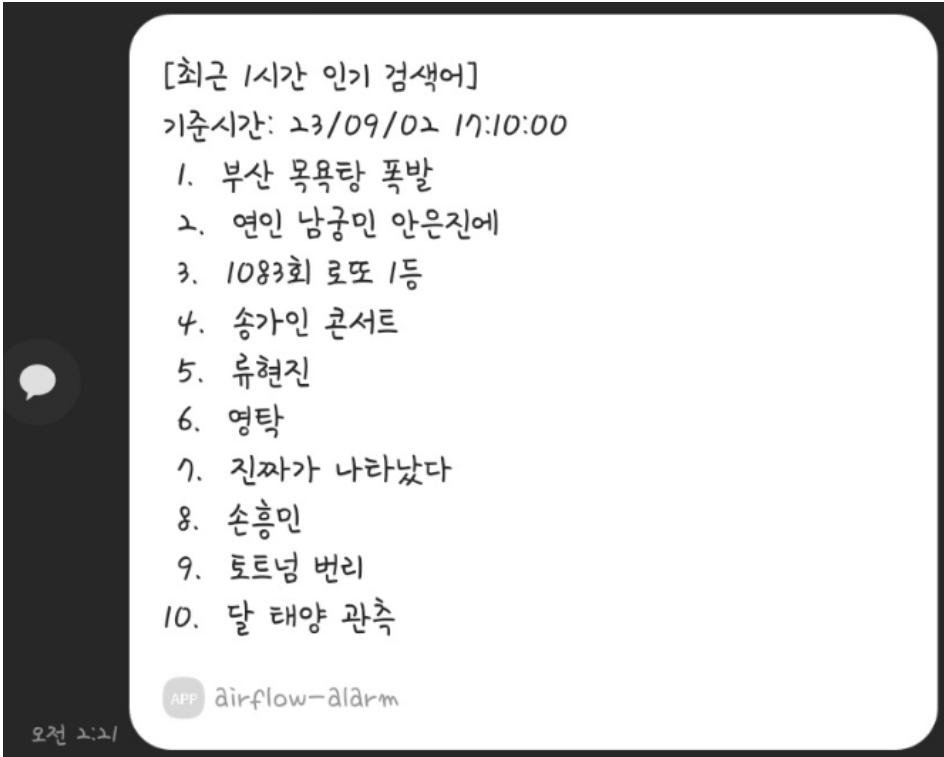
```
consumer = KafkaConsumer(
    'recent_topics',
    bootstrap_servers=['kafka-server:19092'],
    group_id='kakao_message_server',
    value_deserializer=lambda x: json.loads(x.decode('utf-8')),
    auto_offset_reset='earliest',
    enable_auto_commit=False
)

for msg in consumer:
    print("- Message info")
    print(f"topic={msg.topic}, partition={msg.partition}, offset={msg.offset}")
    print(f"key={msg.key}, value={msg.value}")

    text, output_path = msg.value['text'], msg.value['output_path']
    api.send_text(text=text, link={}, button_title="바로 확인")
    consumer.commit()
```

### 7) Result

- 10분마다 카카오톡으로 인기 검색어 분석 결과를 받을 수 있음



---

# CONTENTS

Enhanced Index Tracking

---

# Enhanced Index Tracking

## 1. 프로젝트 소개

벤치마크 지수를 구성하는 여러 종목들 중 지수 이상의 수익률을 내는 포트폴리오(종목과 편입비율)을 선택하는 프로젝트

본 프로젝트에 대한 논문은 스마트미디어저널(KCI)에 등재되었으며 석사 학위 논문으로 인정받았습니다.

자세한 내용은 아래의 논문과 상세설명 자료를 통해 확인하실 수 있습니다.

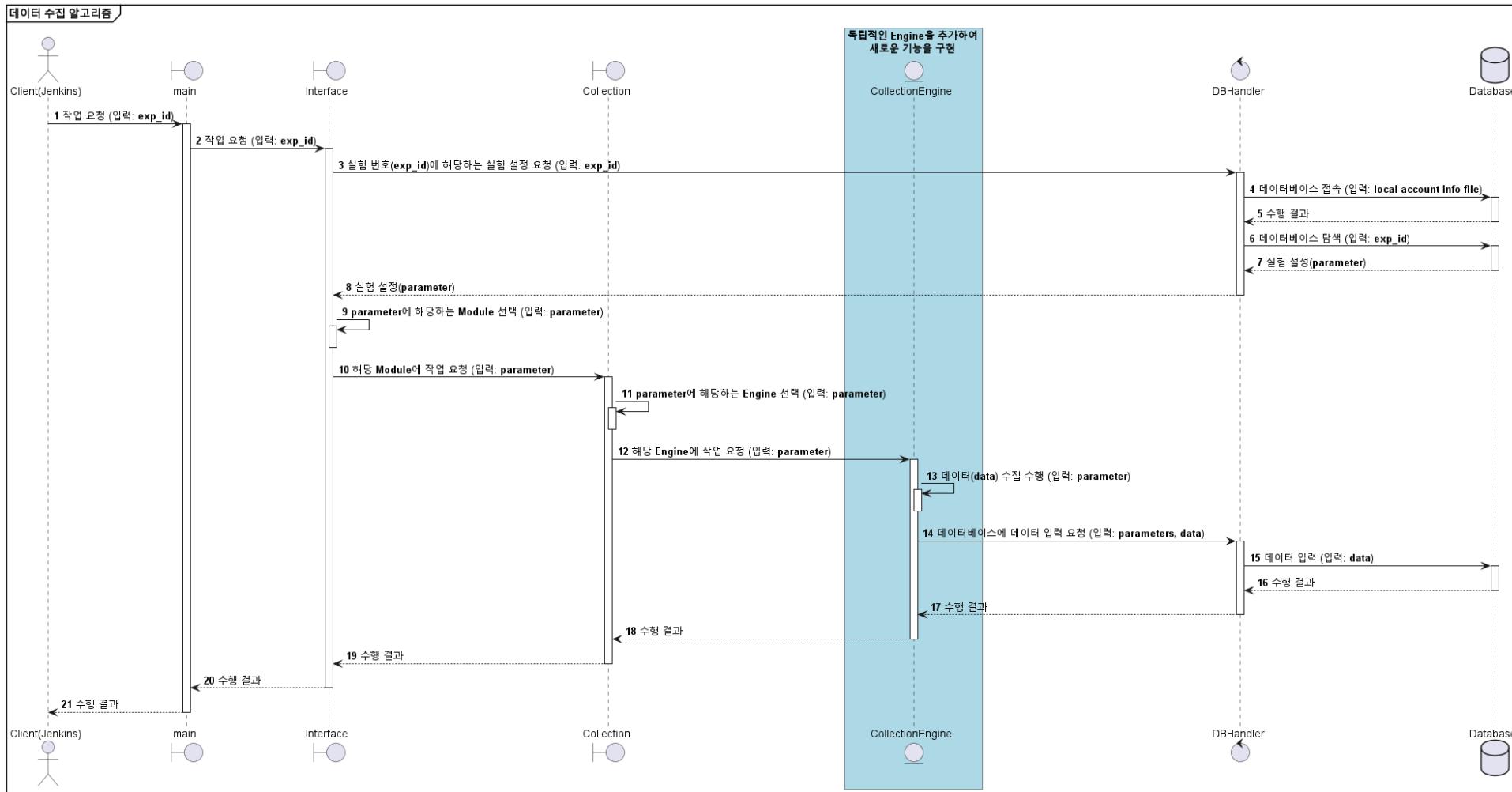
- GitHub <https://github.com/alchemine/enhanced-index-tracking>
- Paper [https://kism.or.kr/file/memoir/10\\_3\\_4.pdf](https://kism.or.kr/file/memoir/10_3_4.pdf)
- 상세설명 <https://shorturl.at/iDKO8>

## 2. 상세 설명

- 비정상적인(nonstationary) 금융 데이터를 사용하는 경우, 데이터의 변동성과 수익성의 trade-off를 제어하고 모델의 일반화 성능을 높이는 것 이 핵심 이를 해결하기 위해 3단계 포트폴리오 선택 알고리즘과 앙상블 학습 알고리즘을 제안하여 2016년 ~ 2020년 약 5년간 S&P500 지수에 적용한 결과, 평균적으로 18.9% 높은 샤프지수를 가진 포트폴리오를 선택할 수 있었음

## 2. 상세 설명(continued)

### Sequence Diagram (데이터 수집 알고리즘)

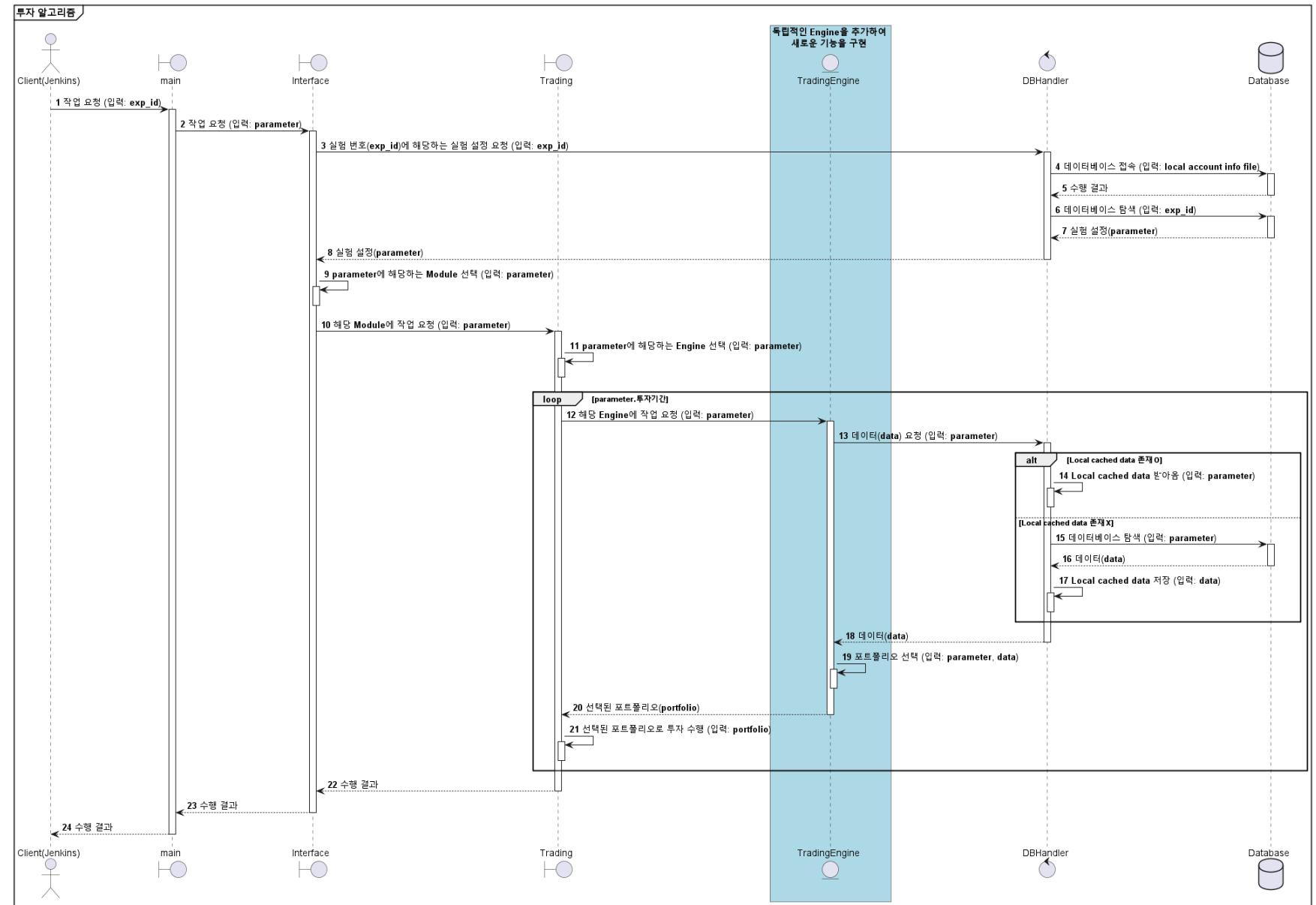


## Enhanced Index Tracking

### 2. 상세 설명(continued)

#### Sequence Diagram

##### (투자 알고리즘)

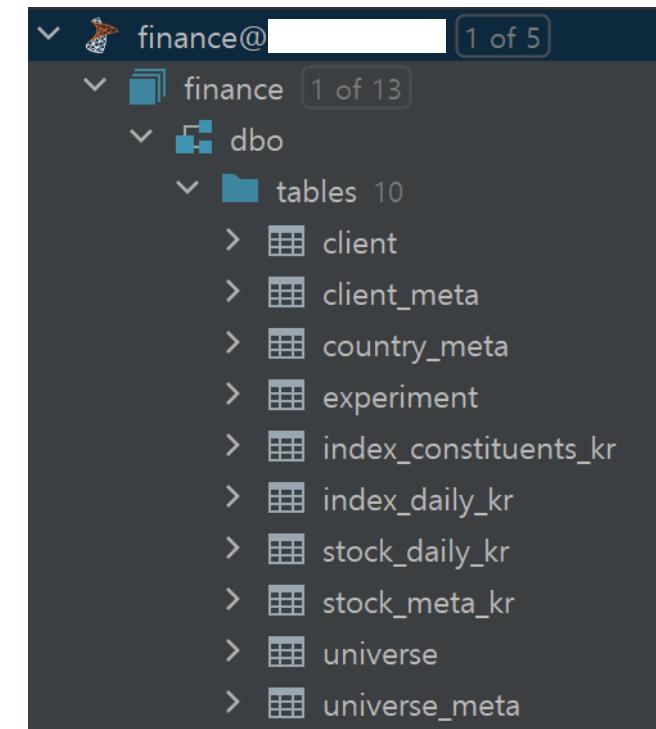


## 2. 상세 설명(continued)

### 스키마 정의

Table: stock\_daily\_kr

필드순번	필드명	데이터 TYPE	KEY	NULL값여부	자동여부	디폴트값	필드설명	OPTIONS	비고
1	index	INT	PK	NO	YES	NULL			
2	종목코드	VARCHAR(20)	FK(stock_meta_kr.종목코드)	NO	NO	NULL			
3	날짜	DATE		NO	NO	NULL			
4	시가	FLOAT		NO	NO	NULL			
5	고가	FLOAT		NO	NO	NULL			
6	저가	FLOAT		NO	NO	NULL			
7	종가	FLOAT		NO	NO	NULL			
8	거래량	FLOAT		NO	NO	NULL			
9	거래대금	FLOAT		NO	NO	NULL			
10	누적체결매도수량	FLOAT		NO	NO	NULL			
11	누적체결매수수량	FLOAT		NO	NO	NULL			
12	상장주식수	FLOAT		NO	NO	NULL			
13	시가총액	FLOAT		NO	NO	NULL			
14	외국인현보유비율	FLOAT		NO	NO	NULL			
15	기관순매수	FLOAT		NO	NO	NULL			
16	기관누적순매수	FLOAT		NO	NO	NULL			
17	대비부호	NVARCHAR(100)		NO	NO	NULL			
18	종목상태	NVARCHAR(100)		NO	NO	NULL			
19	종목상태_정상	BIT		NO	NO	NULL			
20	ETF	BIT		NO	NO	NULL			
21	수집날짜	DATE		NO	NO	NULL			



## Enhanced Index Tracking

## 2. 상세 설명(continued)

### WBS(Work Breakdown Structure) 작성

Task ID	작업	담당자	상태	프로그램 진행일수																	
				+1	+2	+3	+4	+5	+6	+7	+8	+9	+10	+11	+12	+13	+14	+15	+16	+17	+18
1	프로젝트 설계		운영진	완료	1																
1.1	요구사항 설정		운영진	완료	1																
1.2	Code structure 설계		운영진	완료	1																
1.3	Database structure 기초 설계		운영진	완료	1																
1.4	WBS 생성		운영진	완료	1																
1.5	진행상황 및 기술현황 문서화		운영진	진행		0															
2	프로젝트 빠대(input만, 기능구현 X) 개발																				
2.1	Submodule 불러오기(analysis-tools)		운영진	완료	1																
2.2	Main, Interface 구현		운영진	완료	1																
2.3	Collection, CollectionEngine, Trading, TradingEngine 선언		운영진	완료	1																
3	Collection 개발																				
3.1	Collection 개발		운영진	완료	1																
3.2	CollectionEngine 개발																				
3.2.1	Creon API 일데이터 수집기 (과거) 개발		운영진	완료	0	1															
3.2.2	데이터 수집 알고리즘 가속화		운영진	완료	1																
3.2.3	Creon API 일데이터 수집기 (실시간) 개발		운영진	대기	-1	-1															
3.2.4	지수 구성 종목 크롤러 개발		운영진	완료													1				
4	Database 구축																				
4.1	MSSQL 서버 구축		운영진	완료	0	1															
4.2	DBHandler 구현		운영진	완료	0	1	1														
4.3	테이블 설계 및 생성																			1	
4.3.1	experiments 테이블 생성		운영진	진행	0	0	0	0													
4.3.2	stock_meta_kr 테이블 생성		운영진	완료	-1				1												
4.3.3	stock_daily_kr 테이블 생성		운영진	완료	0	1															
4.3.4	index_daily_kr 테이블 생성		운영진	완료	-1																
4.3.5	country_info 테이블 생성		운영진	완료	-1	1															
4.3.6	universe 테이블 생성		운영진	완료			1														
4.3.7	universe_meta 테이블 생성		운영진	완료			1														
4.3.8	client 테이블 생성		운영진	완료				1													
4.3.9	client_meta 테이블 생성		운영진	완료				1													
4.3.10	index_constituents_kr 테이블 생성		운영진	완료					1											1	
5	Trading 개발																				
5.1	Trading 개발																				
5.1.1	Backtesting 개발		운영진	완료	-1	0	1														
5.1.2	Portfolio 정의		운영진	완료	-1	0	1														
5.1.3	Client 정의		운영진	완료	0	1															
5.1.4	Env 정의		운영진	완료	0	1															
5.1.3	투자 결과 시각화		운영진	진행	-1		-1	0	0	0	0	0	0	0	0	0	0	0	0	0	
5.1.4	증권사 API를 사용하여 모의/실투자 개발		운영진	대기														-1	-1		
5.2	TradingEngine 개발																				
5.2.1	테스트용 투자 알고리즘 개발		운영진	완료	-1	0	0			1											
5.2.2	Enhanced index tracking 알고리즘 가속화		운영진	완료	-1	0	0				0	0	0	0	0	0	0	1			
5.2.3	Enhanced index tracking 알고리즘 개선 및 개발		운영진	진행			-1	-1	0	0	0	0	0	0	0	0	0	0	0	0	
5.3	KPI 정의 및 구현		운영진	완료	-1		1														
6	Jenkins 구축																				
6.1	Deployment		운영진	대기														-1			
6.2	Start - Data Collection, Invest		운영진	대기														-1			
6.3	Stop - Data Collection, Invest		운영진	대기														-1			
현재 진행도				완료	29	7	1	6	3	7	0	1	1	0	0	0	0	2	0	2	0
				진행	4	0	5	2	6	3	1	1	3	3	3	3	3	2	2	2	0
				대기	5	0	4	6	1	0	2	1	0	0	0	0	1	1	3	0	0

## 2. 상세 설명(continued)

### 1) 문제 정의

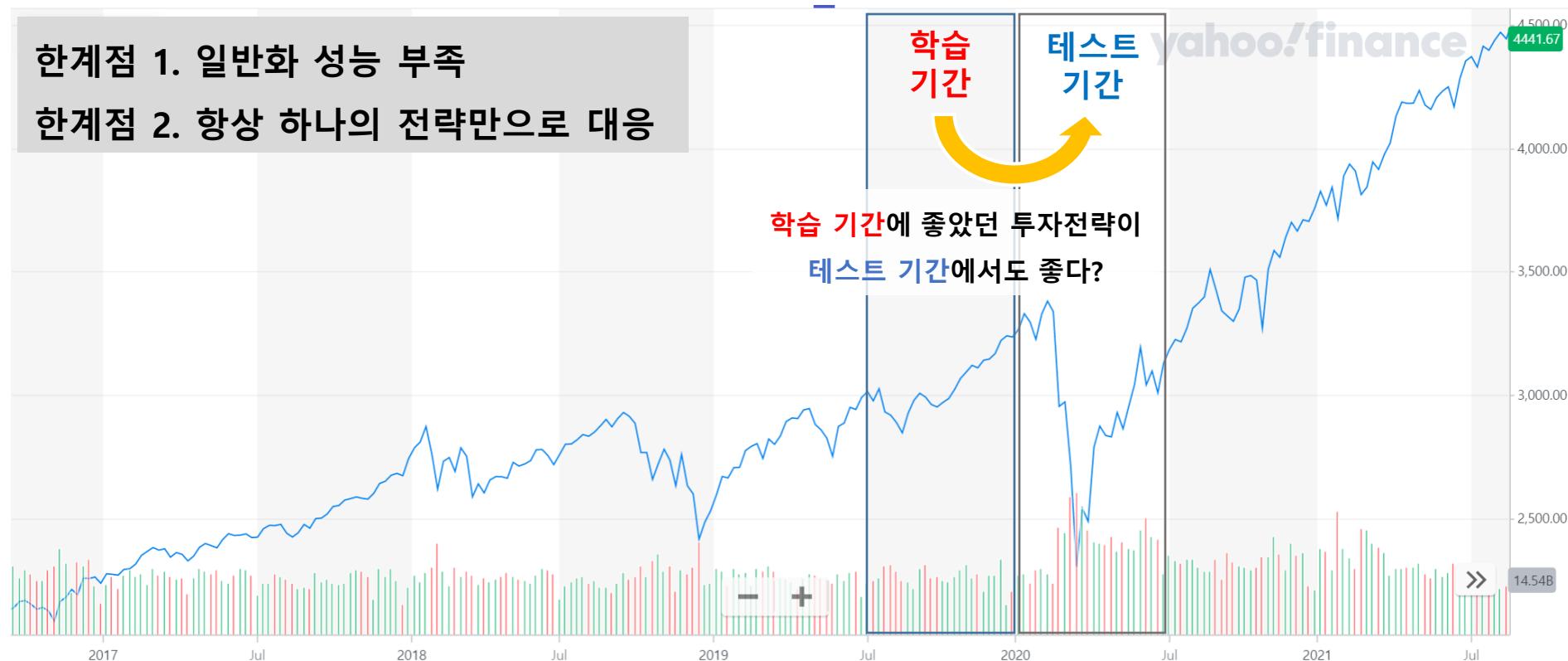
학습 기간의 종가(close) 데이터를 기반으로 학습하여 테스트 기간 동안 벤치마크 지수보다 (안정적으로) 높은 수익률을 얻고자 함



## 2. 상세 설명(continued)

### 2) 기존 연구의 한계점

- ① 테스트 기간에서 모델의 성능이 지속되기 어려움 (일반화 성능 부족)
- ② 항상 동일한 목적함수를 사용하여 문제를 해결하려고 함 (한 가지 전략만 고수)



## 2. 상세 설명(continued)

### 3) 기존 연구의 한계점에 대한 제안 방법

- ① 테스트 기간에서 모델의 성능이 지속되기 어려움 (일반화 성능 부족)

→ **3단계 포트폴리오 선택 알고리즘**: 목적함수와 별개의 지표를 regularization penalty로 고려하여 포트폴리오를 선택



- **포트폴리오 선택 방법**

Step 1 종목 선택: 몬테카를로-유전 알고리즘을 이용한 목적함수(e.g. downside risk) 최적화

Step 2 편입비율 할당: 차분 진화(differential evolution)를 이용한 목적함수 최적화

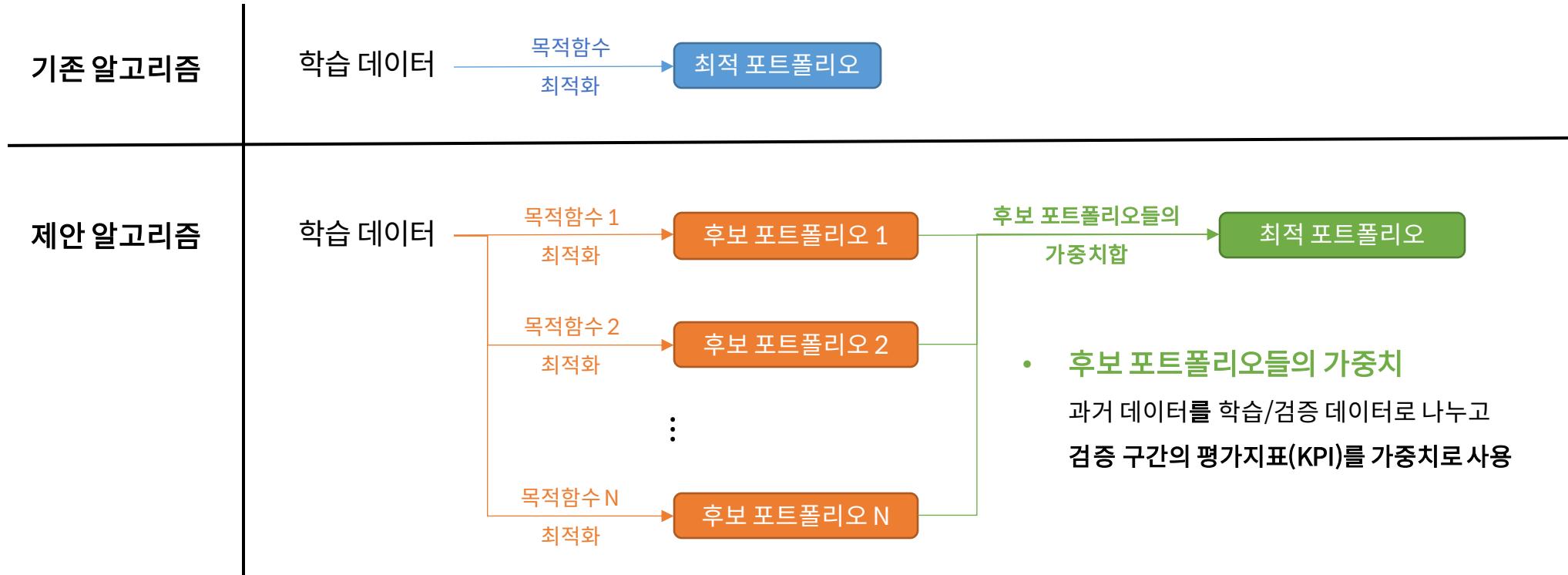
Step 3 포트폴리오 필터링: 후보 포트폴리오들 중 최적의 regularization penalty(e.g. 시가총액, 투자분산도, correlation)를 가진 포트폴리오를 선택

## 2. 상세 설명(continued)

### 3) 기존 연구의 한계점에 대한 제안 방법

- ② 항상 동일한 목적함수를 사용하여 문제를 해결하려고 함 (한 가지 전략만 고수)

→ **양상블 학습 알고리즘**: 투자 시점마다 목적함수로 선택된 포트폴리오들의 양상을 적응적으로(adaptively) 선택



## 2. 상세 설명(continued)

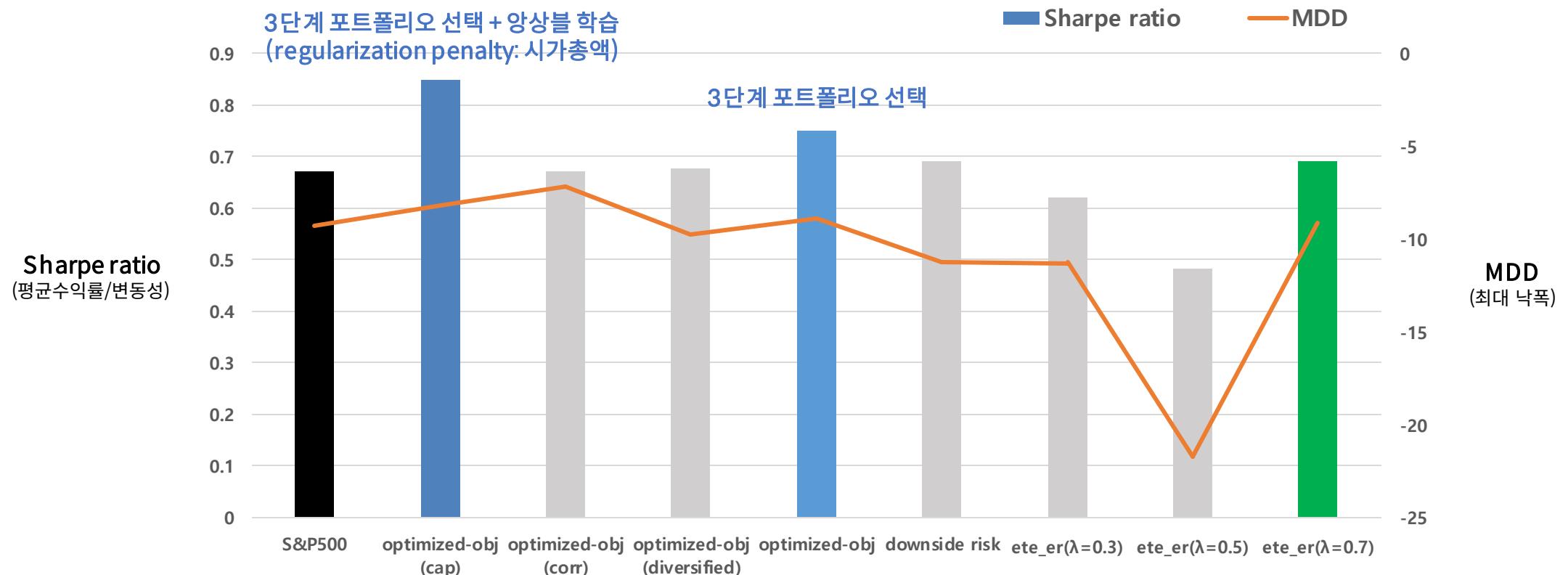
### 4) 실험 결과



## 2. 상세 설명(continued)

### 4) 실험 결과

S&P500 종가(2016/1/4 ~ 2020/10/7)에 대하여 **3단계 포트폴리오 선택 알고리즘과 양상블 학습 알고리즘**을 적용한 결과, 평균적으로 지수 대비 18.9% 더 높은 샤프지수(Sharpe ratio)를 얻을 수 있었습니다.



### 3. 결론

- 비정상적인(nonstationary) 금융 데이터의 변동성을 다루기 위해 새로운 regularization technique를 추가하여 2016년 ~ 2020년 약 5년간 S&P500 지수에 적용한 결과, 평균적으로 지수대비 18.9% 높은 샤프지수를 가진 포트폴리오를 선택할 수 있었음
- NN의 uncertainty modeling(MC Dropout 등)을 통해 데이터와 모델의 불확실성을 변동성으로 활용하고, 직접적으로 KPI를 최대화시키는 Safety Reinforcement Learning 기법을 사용한다면 더욱 활용성이 높아질 것으로 기대

---

# CONTENTS

Wind Power Forecasting

---

## Wind Power Forecasting

### 1. 프로젝트 소개

풍력 발전 터빈의 위치, 온도와 풍속 등의 시공간 데이터를 학습하여 미래의 유효전력을 예측하는 프로젝트

### 2. 특징

- 시계열의 특성을 유지하며 결측치와 에러값(전체 데이터의 23%)을 처리
- 데이터를 가공하기 위한 다양한 기법들을 사용
- 대학원생팀 1등으로 상금 200만원을 수상하였으며, imputing 알고리즘이 특히 좋은 평가를 받음

## 1) 문제 정의

### Feature 설명

TurbID	- Wind turbine ID, 발전기 ID
Day	- Day of the record, 날짜
Timestamp	- Created time of the record, 시간 (10분 단위)
Wspd	- The wind speed recorded by the anemometer, 풍속(m/s)
Wdir	- wind direction, 터빈이 바라보는 각도와 실제 바람 방향 각도 차이(°)
Etmp	- Temperature of the surrounding environment, 외부 온도(°C)
Itmp	- Temperature inside the turbine nacelle, 터빈 내부 온도(°C)
Ndir	- Nacelle direction, i.e., the yaw angle of the nacelle, 터빈이 바라보는 방향 각도(°)
Pab	- Pitch angle of blade, 터빈 당 3개의 날이 있으며 각각의 각도가 다름(°)
Prtv	- Reactive power, 무효전력 : 에너지원을 필요로 하지 않는 전력(kW)
Patv	- Active power(target variable), 유효전력 : 실제로 터빈을 돌리는 일을 하는 전력(kW)



Goal 134개의 터빈(TurbID)에 대한 유효전력(Patv) 예측

KPI 미래 2일 간 10분 단위로 sampling 된 유효전력에 대한 MSE와 MAE의 평균

### 1) 문제 정의 (continued)

Abnormal label 조건

1.  $Wspd > 2.5$  and  $Patv \leq 0$

바람은 불지만 발전량이 없는 경우

2.  $|Pab| > 89$

바람과 날개의 각도 차이가 커 바람의 영향을 받지 못하는 경우

3.  $|Wdir| > 180$  or  $|Ndir| > 720$

기기의 정상 범위를 넘어가는 경우

4.  $Patv$  is null

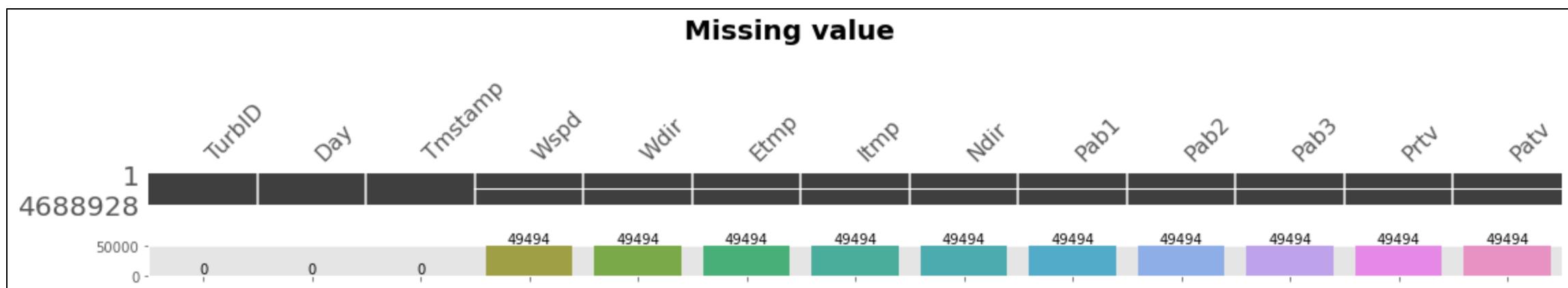
Target이 존재하지 않는 경우

## Wind Power Forecasting

### 2) Explanatory Data Analysis (Training data)

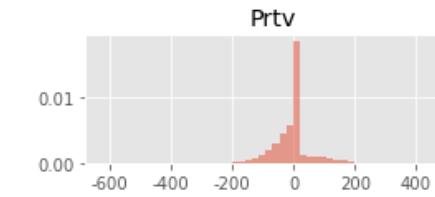
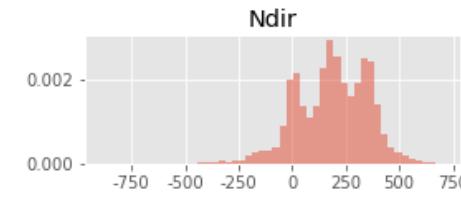
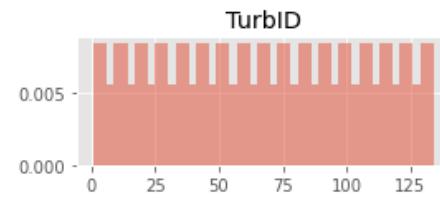
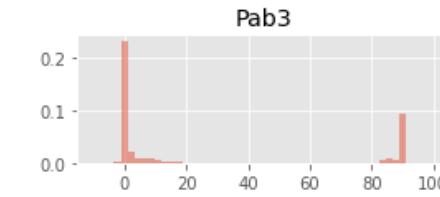
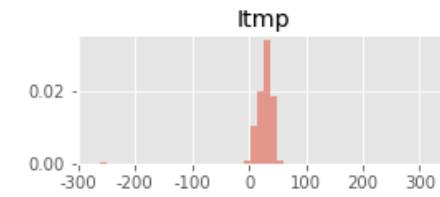
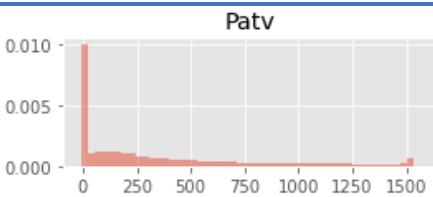
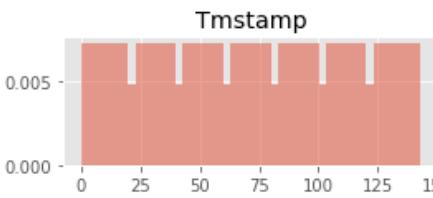
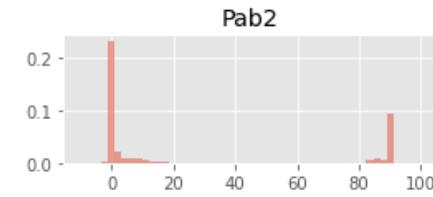
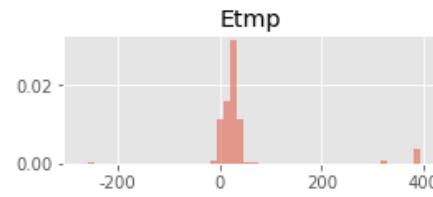
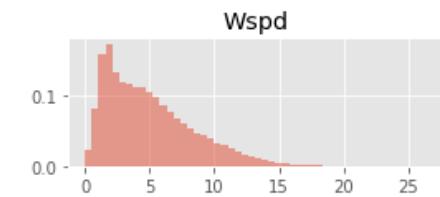
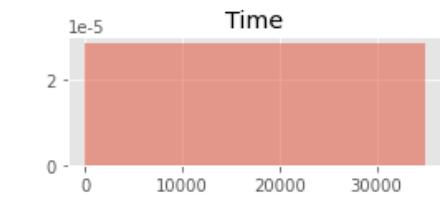
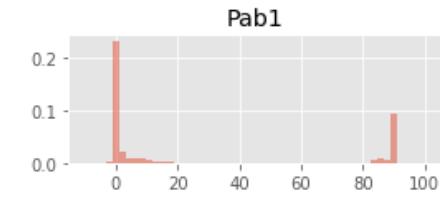
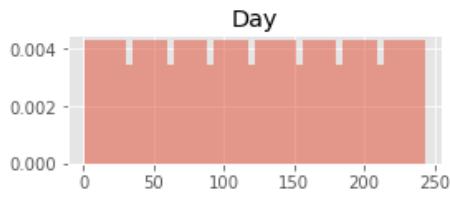
	TurbID	Day	Tmstamp	Wspd	Wdir	Etmp	Itmp	Ndir	Pab1	Pab2	Pab3	Prtv	Patv
0	1	1	00:00	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1	1	1	00:10	6.17	-3.99	30.73	41.80	25.92	1.00	1.00	1.00	-0.25	494.66
2	1	1	00:20	6.27	-2.18	30.60	41.63	20.91	1.00	1.00	1.00	-0.24	509.76
3	1	1	00:30	6.42	-0.73	30.52	41.52	20.91	1.00	1.00	1.00	-0.26	542.53
4	1	1	00:40	6.25	0.89	30.49	41.38	20.91	1.00	1.00	1.00	-0.23	509.36
...	...	...	...	...	...	...	...	...	...	...	...	...	...
4727227	134	243	23:10	10.98	-1.96	-5.11	-0.67	345.57	8.82	8.82	8.82	136.49	1152.60
4727228	134	243	23:20	11.82	-3.18	-5.46	-0.54	345.57	13.87	13.87	13.87	84.43	681.65
4727229	134	243	23:30	11.91	-1.42	-5.21	-0.42	345.57	10.69	10.69	10.69	145.72	1118.35
4727230	134	243	23:40	11.86	-0.95	-5.40	-0.38	345.57	13.94	13.94	13.94	89.56	683.49
4727231	134	243	23:50	11.72	0.04	-5.23	-0.37	345.57	10.90	10.90	10.90	120.18	1026.93

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 4688928 entries, 0 to 4727231
Data columns (total 13 columns):
 #   Column      Dtype  
 --- 
 0   TurbID     int64  
 1   Day         int64  
 2   Tmstamp    object  
 3   Wspd        float64 
 4   Wdir        float64 
 5   Etmp        float64 
 6   Itmp        float64 
 7   Ndir        float64 
 8   Pab1        float64 
 9   Pab2        float64 
 10  Pab3        float64 
 11  Prtv        float64 
 12  Patv        float64 
dtypes: float64(10), int64(2), object(1)
memory usage: 500.8+ MB
```



# Wind Power Forecasting

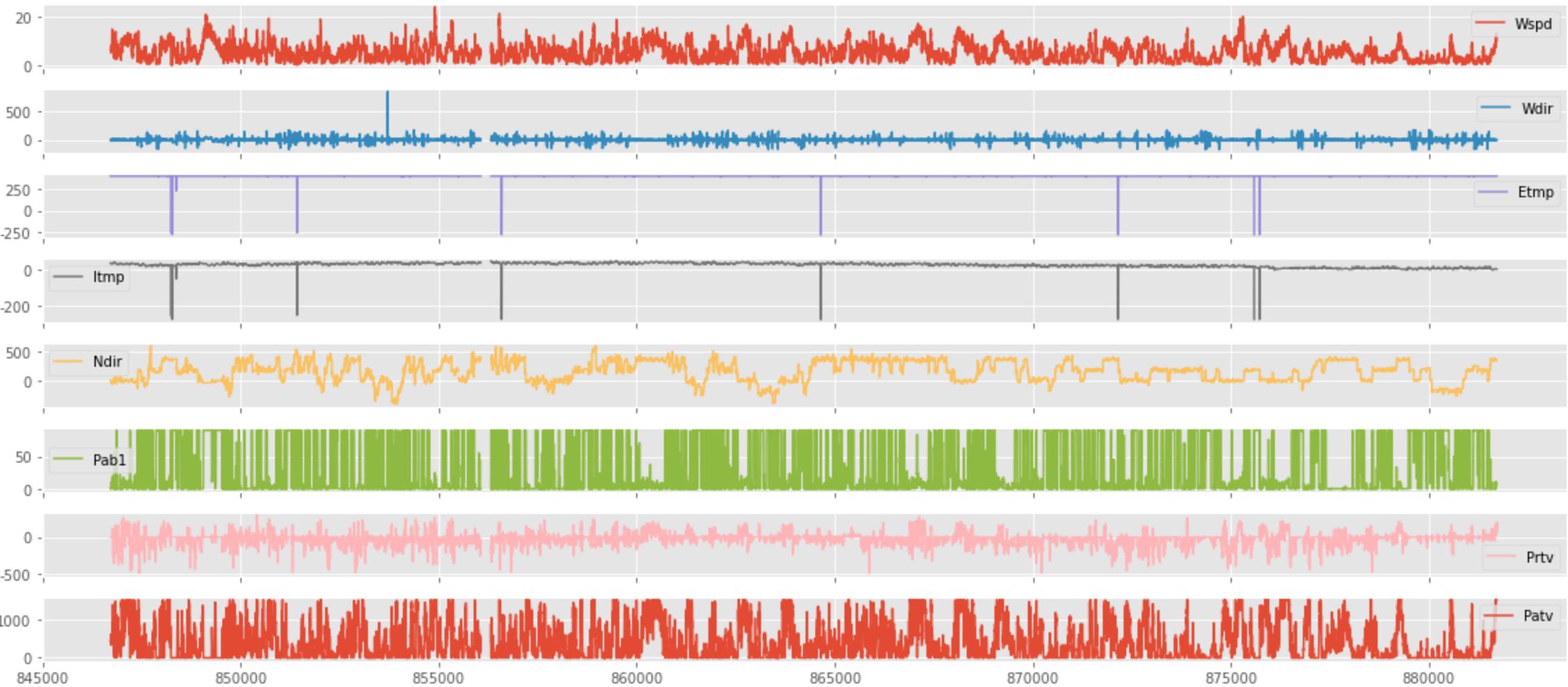
## Features



Target

# Wind Power Forecasting

## Features of TurbID=25

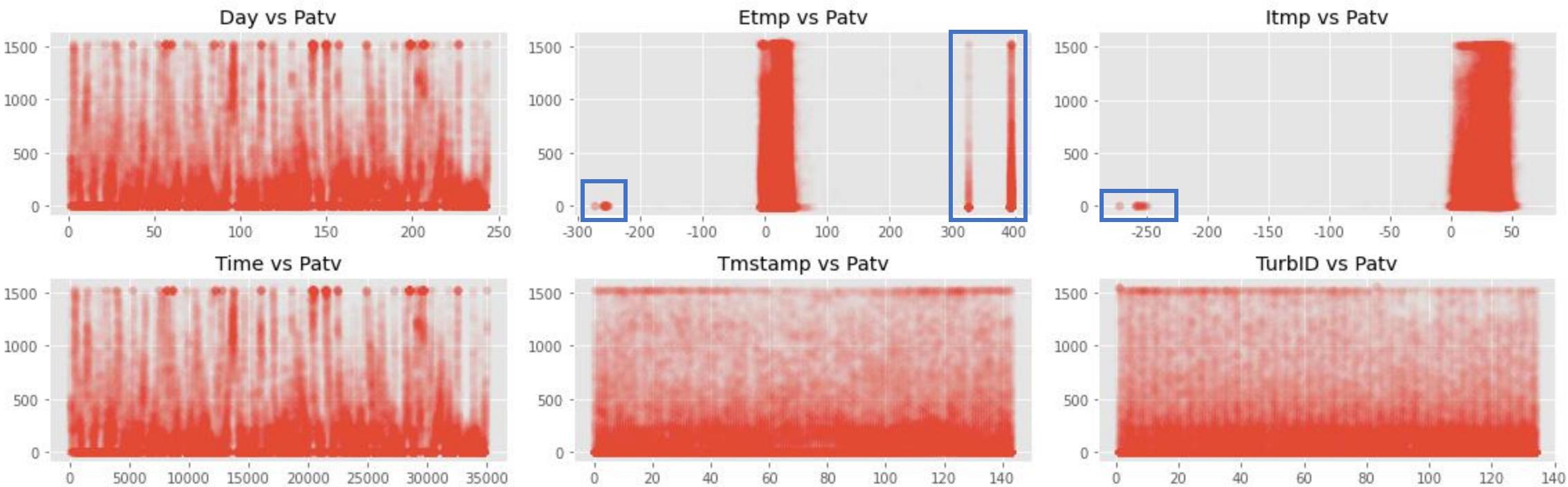


## Wind Power Forecasting

### ① Day, Ettmp, Itmp, Time, Tmstamp, TurbID

- 뚜렷한 관계가 보이지 않음
- Ettmp, Itmp: 이상치 처리가 필요

### Features vs Target



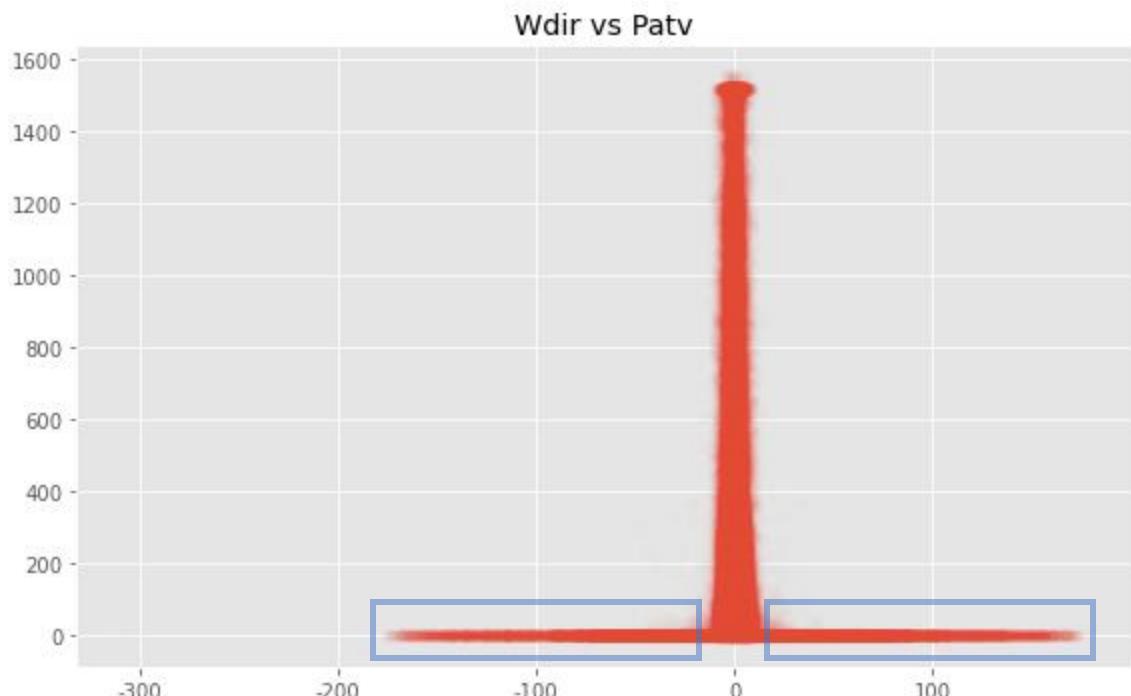
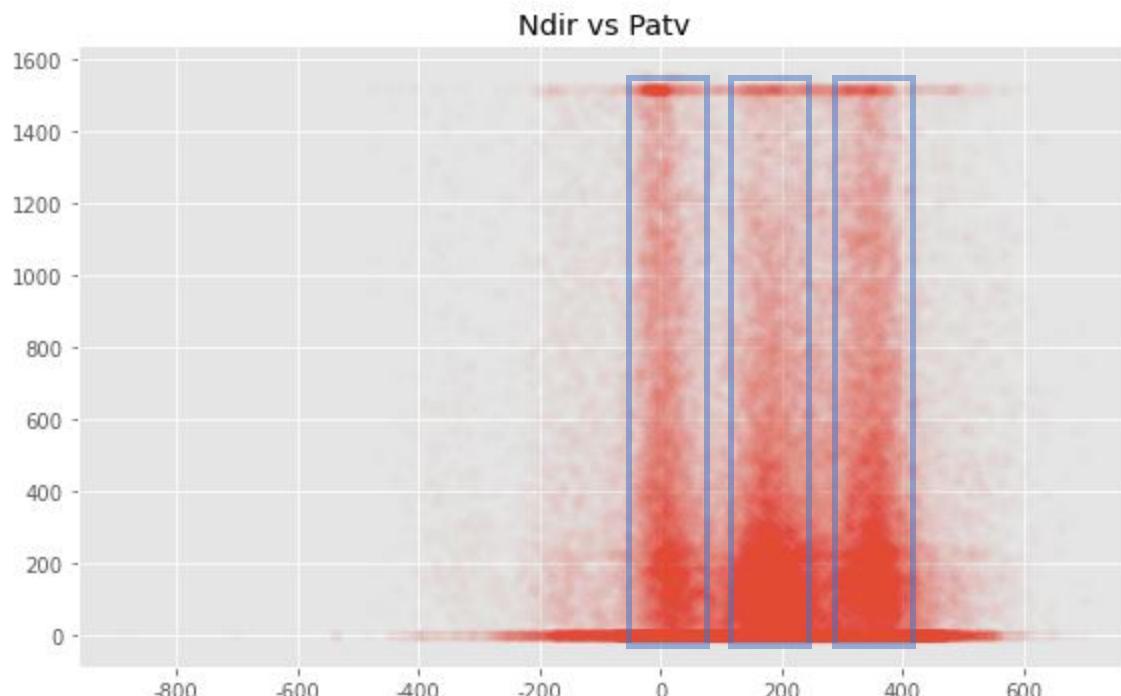
### ② Ndir

- 0도, 180도, 360도에서 기둥이 보임 (180도의 배수인 것처럼 보이나, -180도는 기둥이 없기 때문에 음수에 대한 구분이 필요)

### ③ Wdir

- 절댓값이 10도 이상이면, Patv  $\approx 0$

### Features vs Target

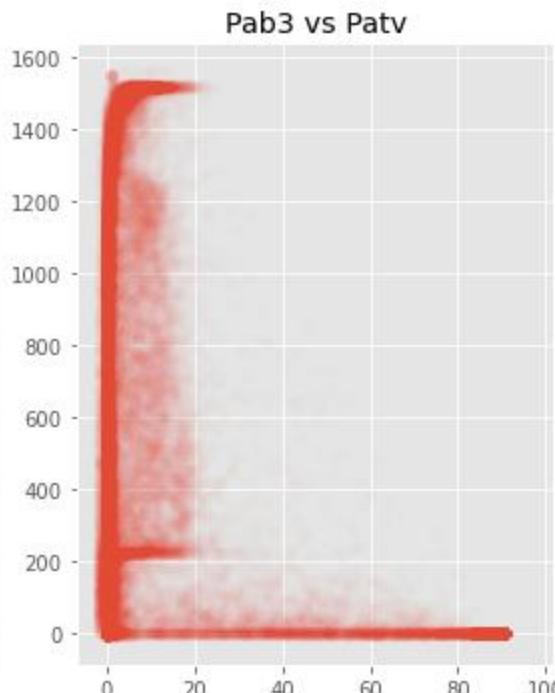
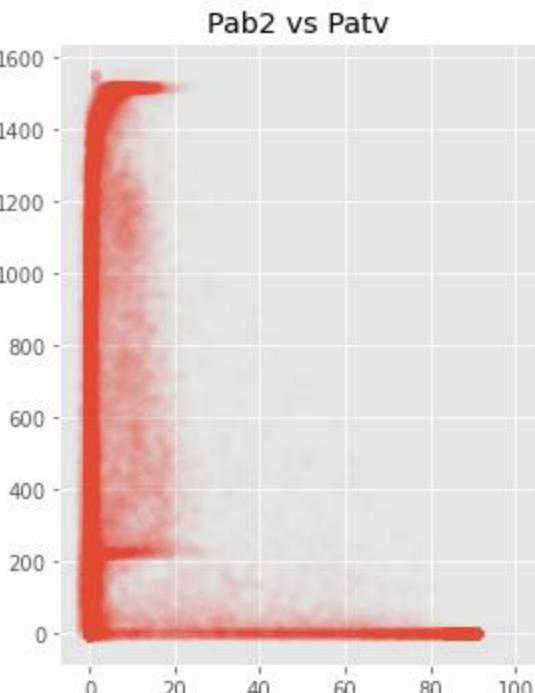
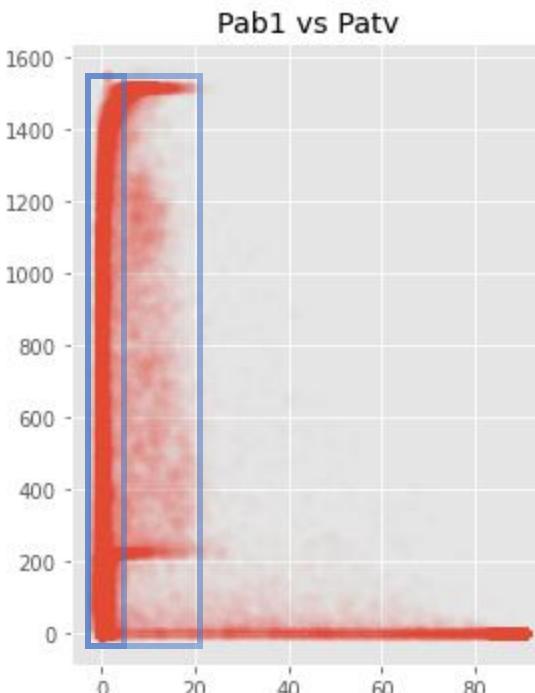


## ④ Pab1, Pab2, Pab3

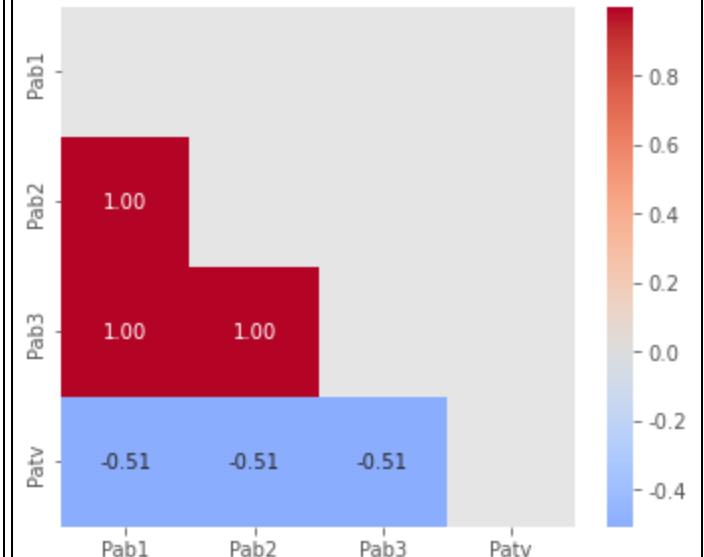
- 날개 3개의 각도로 서로 간의 correlation이 10이기 때문에 평균값 Pab를 사용
- 0.03도, 20도를 기준으로 층이 보임



## Features vs Target



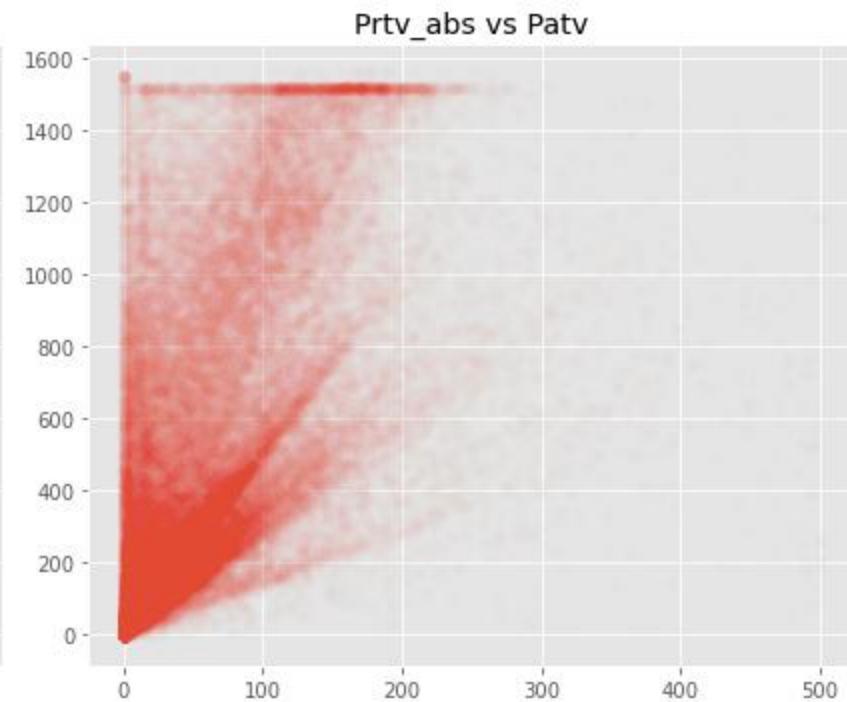
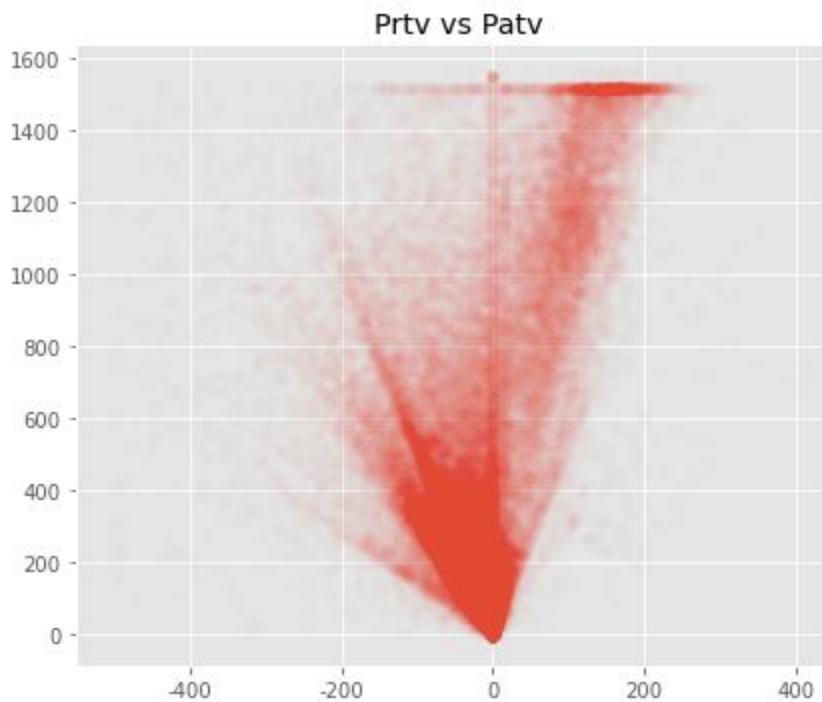
## Correlation matrix



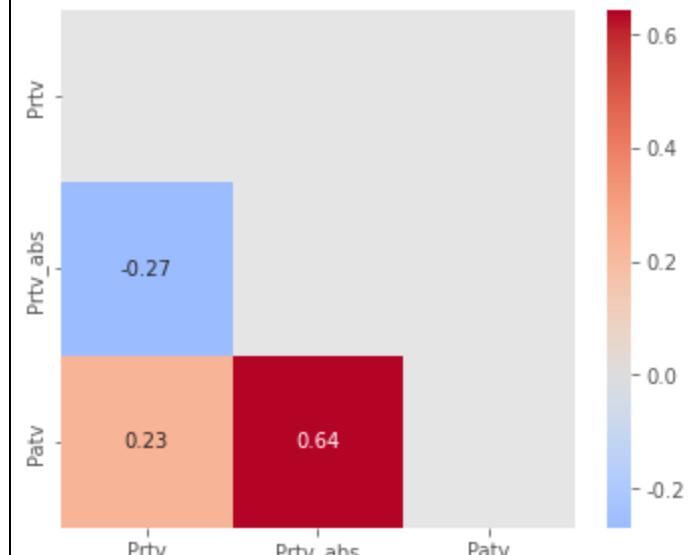
## ⑤ Prtv

- 음수인 경우와 양수인 경우에 Patv와 다른 관계를 가지는 것 같음
- 절대값이 Patv와 강한 양의 상관관계를 가짐(0.64)

### Features vs Target

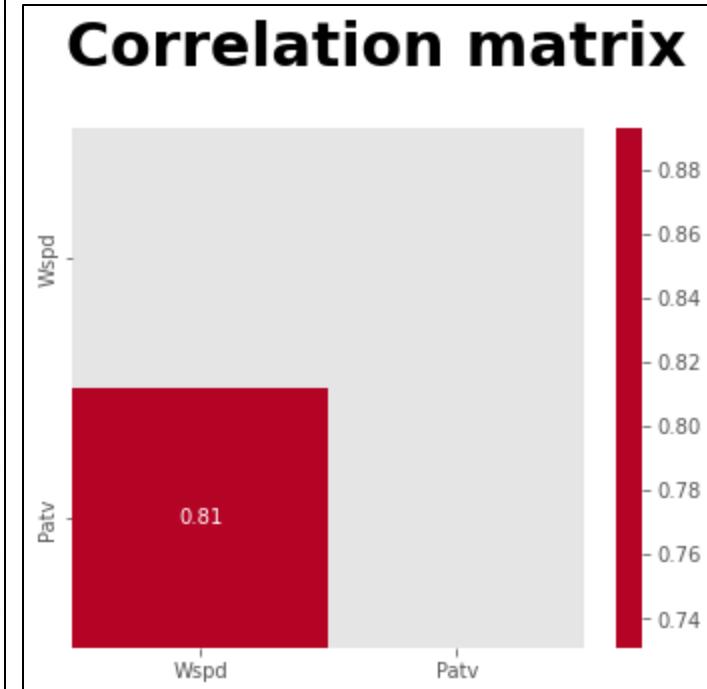
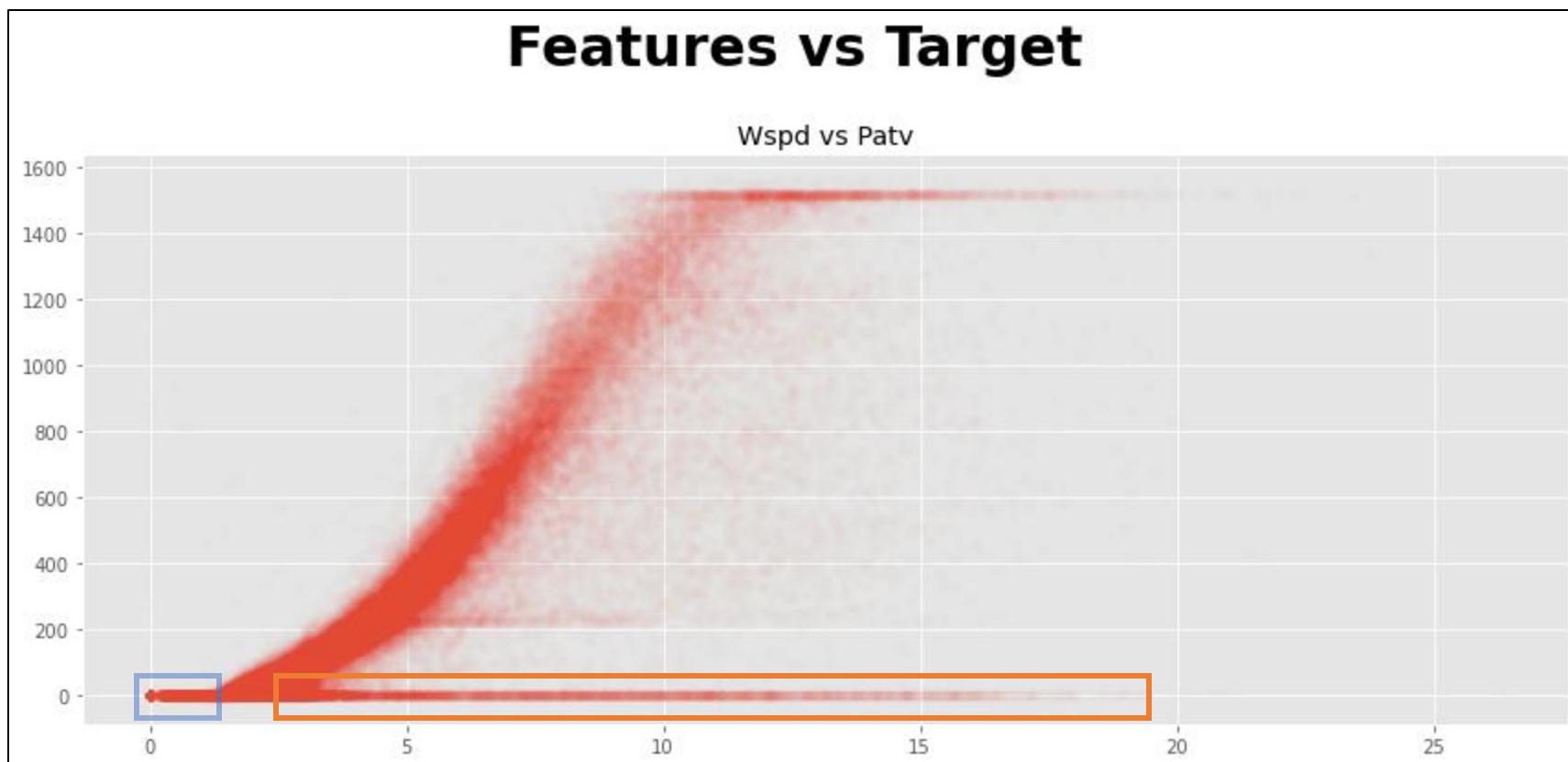


### Correlation matrix



## ⑥ Wspd

- Patv와 강한 양의 상관관계를 가짐(0.81)
- 1m/s 이하:  $\text{Patv} = 0$
- 1m/s 이상: 다른 요인으로 인해  $\text{Patv} = 0$  인 데이터를 구분할 수 있다면 Wspd의 correlation을 더 높일 수 있음



### 3) Preprocessing

#### ① Mark abnormal Patv

조건 1. Wspd > 2.5 and Patv <= 0

조건 2. Pab > 89

조건 3. |Wdir| > 180 or |Ndir| > 720

조건 4. Patv is null

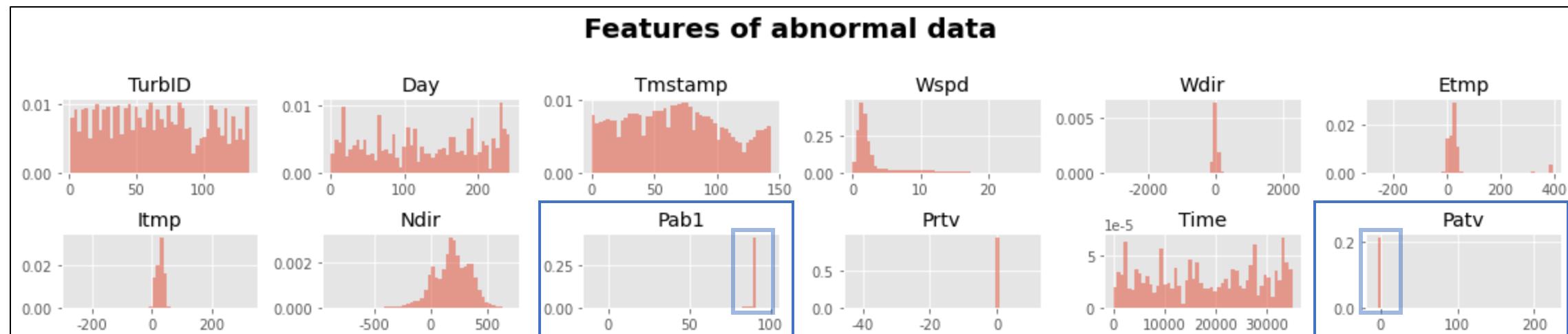
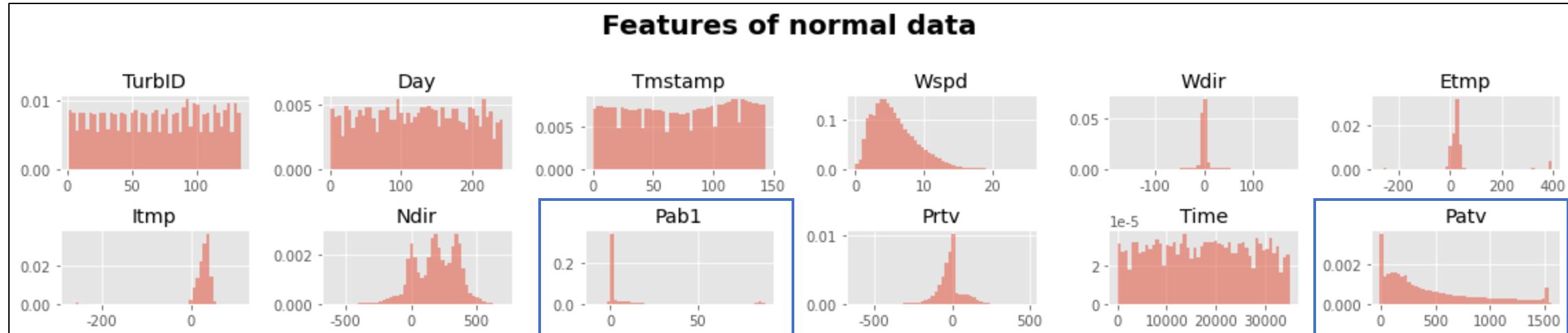
하나라도 해당된다면 Abnormal=1 o.w. 0

	TurbID	Day	Tmstamp	Wspd	Wdir	Etmp	Itmp	Ndir	Pab1	Pab2	Pab3	Prtv	Time	Patv	Abnormal
0	1	1	0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	1	NaN	1
1	1	1	1	6.17	-3.99	30.73	41.80	25.92	1.00	1.00	1.00	-0.25	2	494.66	0
2	1	1	2	6.27	-2.18	30.60	41.63	20.91	1.00	1.00	1.00	-0.24	3	509.76	0
3	1	1	3	6.42	-0.73	30.52	41.52	20.91	1.00	1.00	1.00	-0.26	4	542.53	0
4	1	1	4	6.25	0.89	30.49	41.38	20.91	1.00	1.00	1.00	-0.23	5	509.36	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
4688923	134	243	139	10.98	-1.96	-5.11	-0.67	345.57	8.82	8.82	8.82	136.49	34988	1152.60	0
4688924	134	243	140	11.82	-3.18	-5.46	-0.54	345.57	13.87	13.87	13.87	84.43	34989	681.65	0
4688925	134	243	141	11.91	-1.42	-5.21	-0.42	345.57	10.69	10.69	10.69	145.72	34990	1118.35	0
4688926	134	243	142	11.86	-0.95	-5.40	-0.38	345.57	13.94	13.94	13.94	89.56	34991	683.49	0
4688927	134	243	143	11.72	0.04	-5.23	-0.37	345.57	10.90	10.90	10.90	120.18	34992	1026.93	0

## Wind Power Forecasting

### ① Mark abnormal Patv (continued)

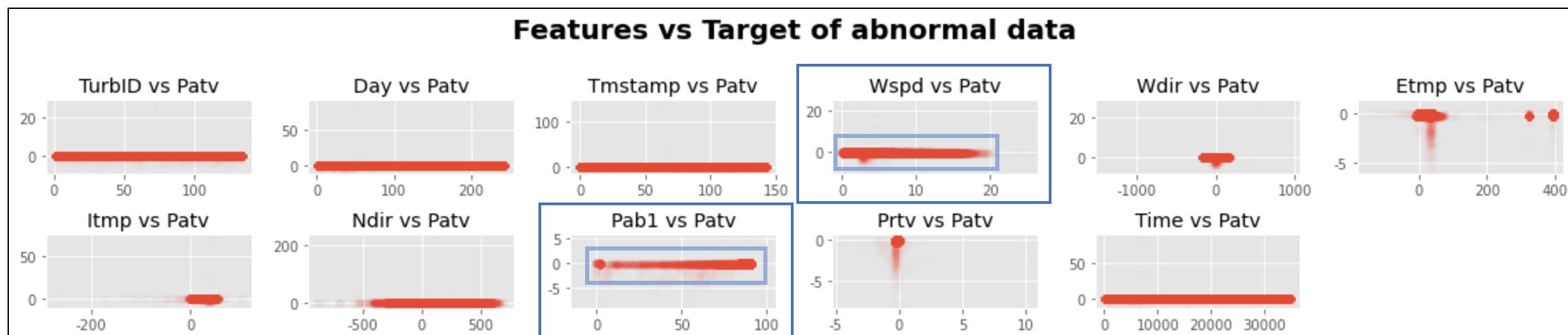
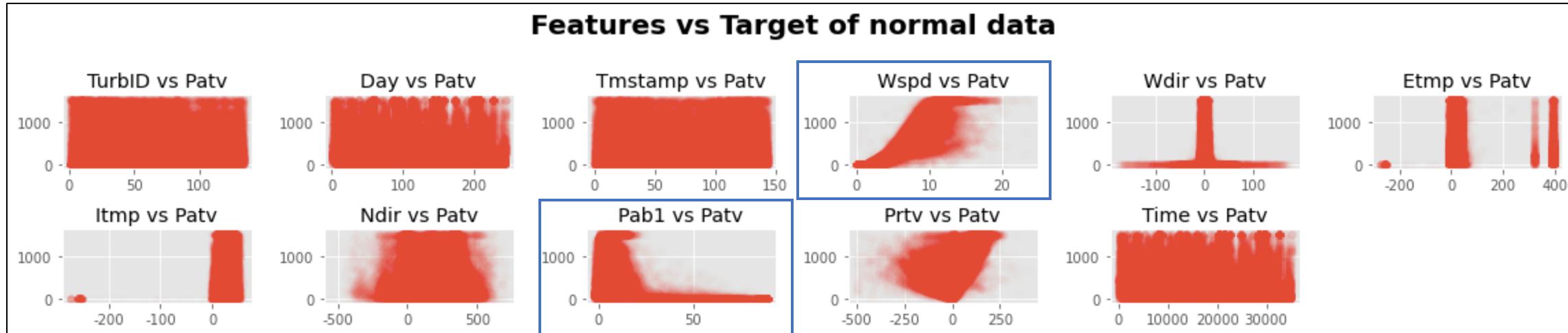
Pab가 90도가 넘어 Patv= 0인 경우가 대부분



# Wind Power Forecasting

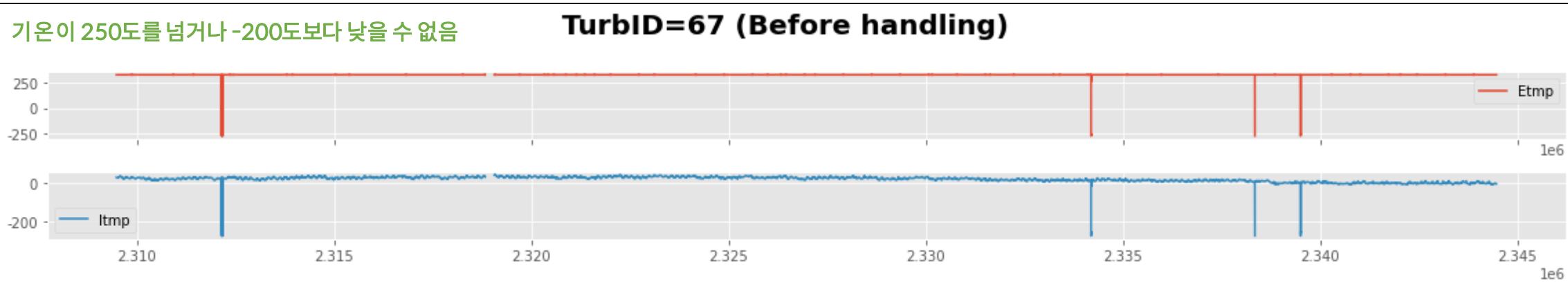
## ① Mark abnormal Patv (continued)

Pab로 인한 Patv=0 분리 후, corr(Wspd, Patv) 상승: 0.81 → 0.88

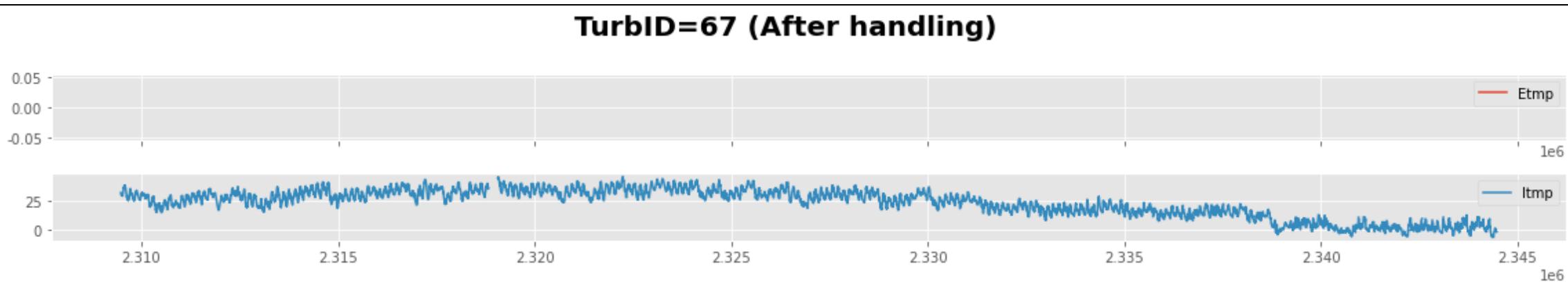


## ② Outlier handling

- 다음과 같이 연속적으로 outlier가 이어지는 경우는 직접 null값으로 처리



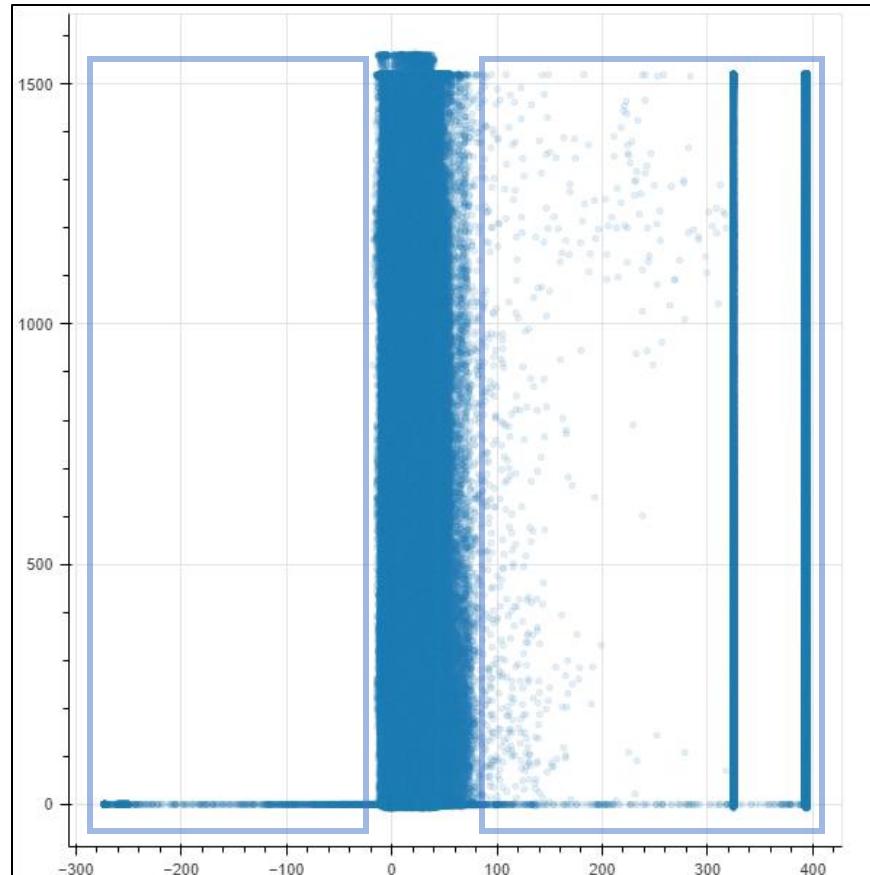
**TurbID=67 (After handling)**



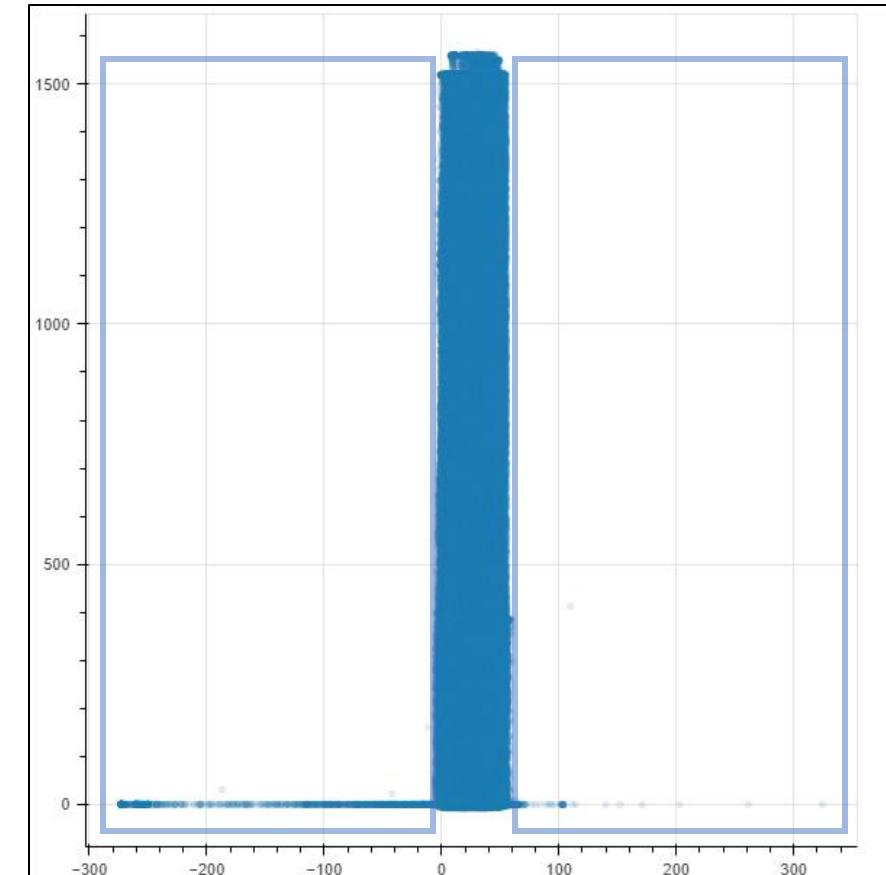
## ② Outlier handling

2. Ndir, Wdir, Pab: 주어진 정상 범위(not abnormal)로 clipping

Etmp: -20도~80도 안에 들도록 clipping

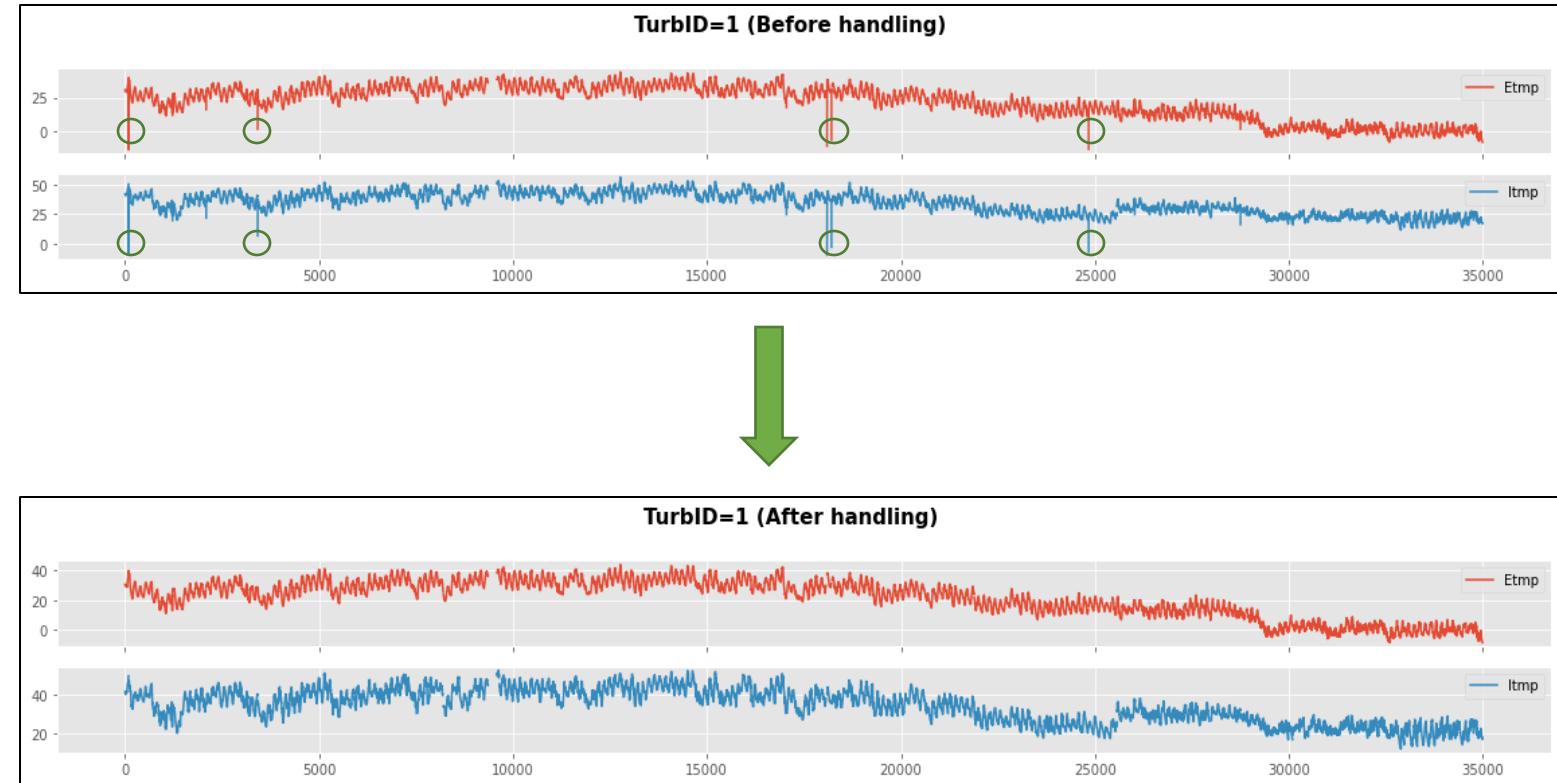
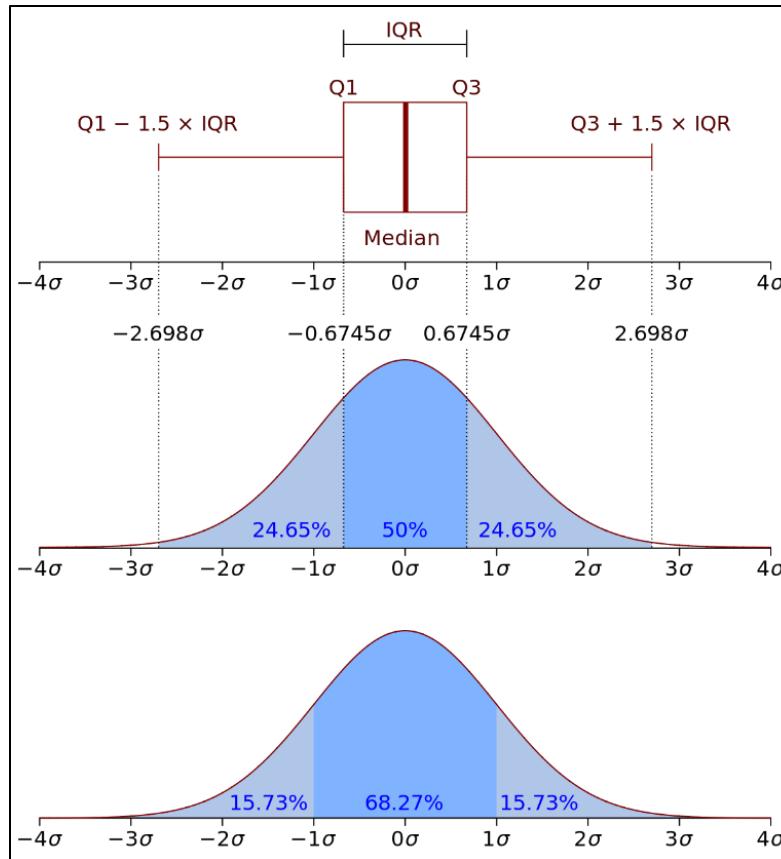


Itmp: -10도~65도 안에 들도록 clipping



## ② Outlier handling

3. 각 sample에 대하여 해당값 혹은 1차 차분값이 주변 2일 동안의 값들에 대하여 이상치라고 판단되는 경우 null로 채움  
(Ettmp와 Itmp는 종모양의 분포를 가지고 있기 때문에  $[Q1 - 1.5 \times IQR, Q3 + 1.5 \times IQR]$  이외의 값을 이상치로 설정하는 것이 적절)



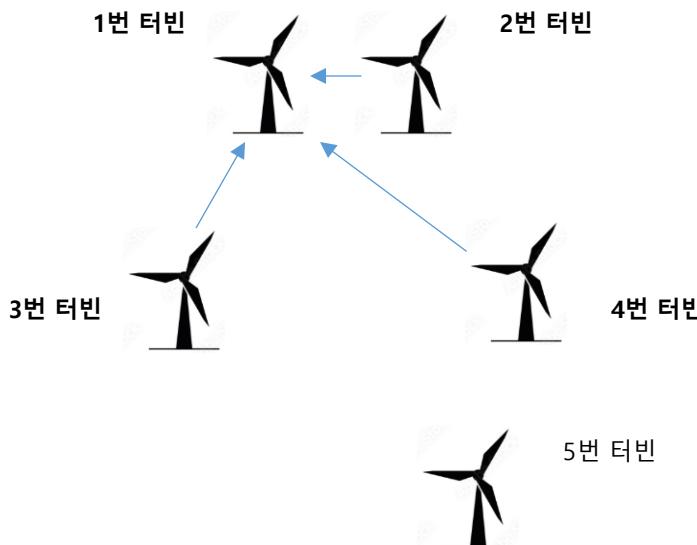
## ③ Imputing

### 1. Greedy imputing

이상치를 처리하고 난 후, 터빈의 데이터가 상당수 소실되는 경우 다음 알고리즘을 통해 값을 채움

#### Algorithm Greedy imputing

1. Imputing하고자 하는 터빈(대상 터빈)과 가장 가까운 k개의 터빈을 선택
2. 선택된 터빈들을 대상 터빈의 값과 비교하여 유사한 순으로 정렬
3. 대상 터빈의 값이 존재하지 않은 timestep의 값을 선택된 터빈들에서 순서대로 채워넣음



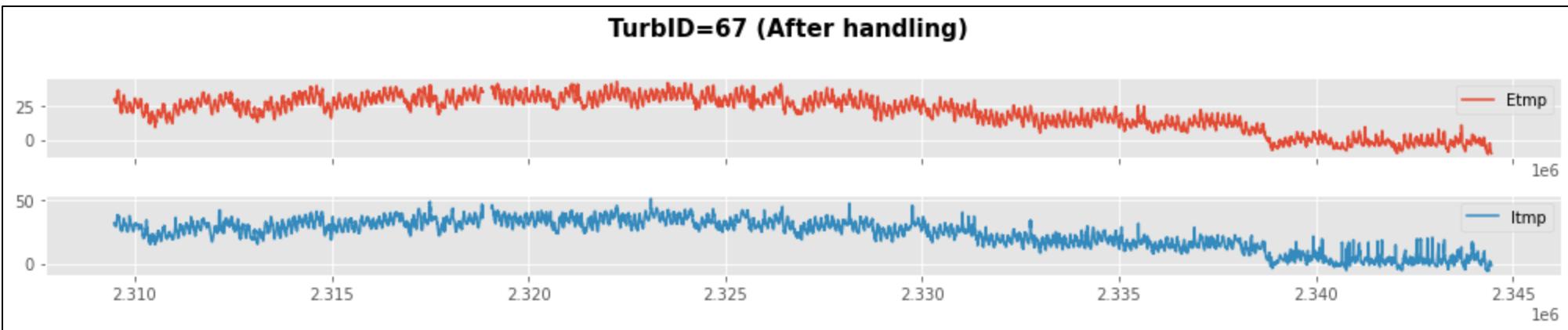
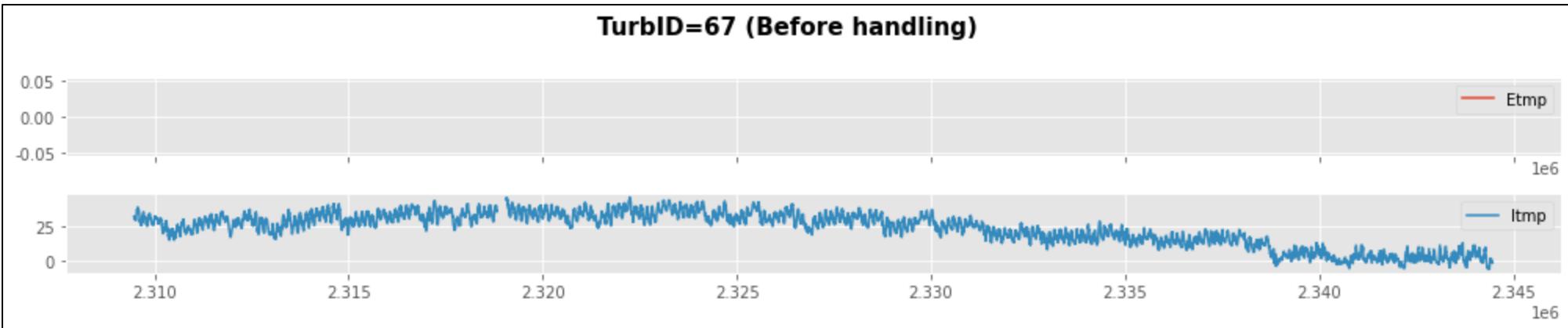
Ex) 1번 터빈에 대한 Greedy imputing 수행과정

$k=3$

터빈 ID	거리	유사도 (MAE)	Time=1	Time=2	Time=3	Time=4	Time=5
1번 (기준)			10	20			
3번	2	0	10			40	
2번	1	1	11	21	31		
4번	3	2		22		44	55
5번	X	3	13	23	34	45	56

## ③ Imputing

### 1. Greedy imputing(continued)



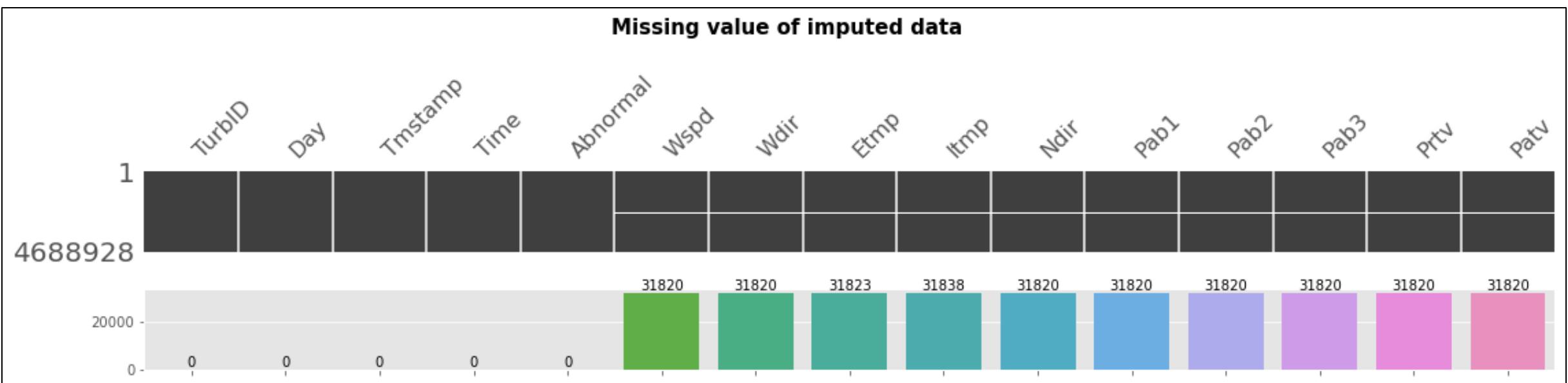
## ③ Imputing

## 2. Linear interpolation

각 turbine에 대하여 threshold(e.g. 12시간) 이상 연속적이지 않은 결측치를 linear interpolation으로 채움

(긴 sequence를 얹지로 interpolation하면 시계열 특성이 망가지게 될 염려가 있음)

최종 imputing 결과

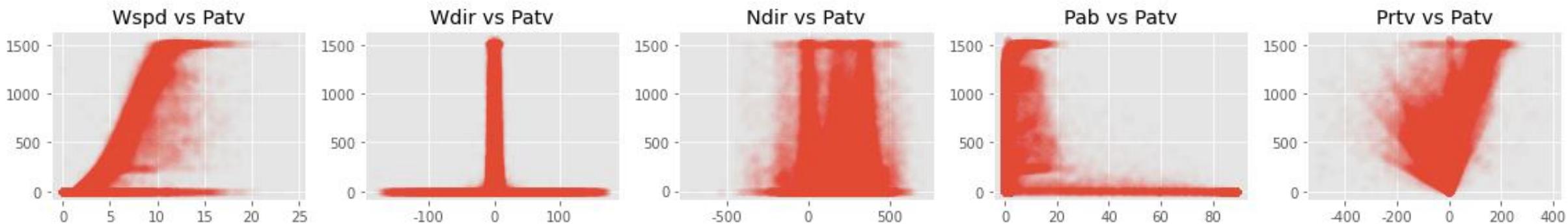


## ④ Feature engineering

### 1. Dummy variables ([insight from EDA](#))

```
data['Wspd_extreme'] = data['Wspd'] < 1
data['Wdir_extreme'] = data['Wdir'].abs() > 10
data['Ndir_extreme'] = data['Ndir'] < -90
data['Pab_extreme1'] = data['Pab'] < 0.03
data['Pab_extreme2'] = data['Pab'] > 20
data['Prtv_pos']      = data['Prtv'] > 0
```

**Features vs Target**



## ④ Feature engineering

## 2. Multiplicative variables(interactive effect)

```
data['Wspd_active'] = data['Wspd'] - 1
data['Wdir_active'] = data['Wdir'].abs() - 10
data['Ndir_cos_abs'] = np.abs(np.cos(Ndir_rad)) # 0도, 180도, 360도
data['Prtv_abs']     = data['Prtv'].abs()
```

```
data['Wspd_comb'] = data['Wspd_extreme'] * data['Wspd_active']
data['Wdir_comb'] = data['Wdir_extreme'] * data['Wdir_active']
data['Ndir_comb'] = data['Ndir_extreme'] * data['Ndir_cos_abs']
data['Prtv_comb'] = data['Prtv_pos'] * data['Prtv_abs']
```

## ④ Feature engineering

## 3. Feature extraction(domain knowledge)

```
ALPHA          = 40
data['Pab']     = (data['Pab1'] + data['Pab2'] + data['Pab3'])/3
Pab_rad        = np.radians(data['Pab']+ALPHA)
data['TSR']      = 1 / np.tan(Pab_rad)
data['RPM']       = data['Wspd_active'] * data['TSR']
data['Wspd_cube'] = data['Wspd_active']**3
data['Patv_pos']   = np.maximum(data['Patv'], min_val)
data['Patan_abs'] = np.arctan(data['Prtv_abs'] / data['Patv_pos'])

Wdir_rad        = np.radians(data['Wdir'])
data['Wdir_cos'] = np.cos(Wdir_rad)
data['Wdir_sin'] = np.sin(Wdir_rad)
data['Wspd_cos'] = data['Wspd_active'] * np.cos(Wdir_rad)
data['Wspd_sin'] = data['Wspd_active'] * np.sin(Wdir_rad)
data['Ndir_sin_abs'] = np.abs(np.sin(Ndir_rad))
```

## ④ Feature engineering

## 4. Time encoding

```
DAY          = 6*24  # 10minute * 6 * 24hour
Time_in_day = data['Time'] * (2*np.pi) / DAY
data['Day_cos'] = np.cos(Time_in_day)
data['Day_sin'] = np.sin(Time_in_day)

YEAR         = 365*DAY
Time_in_year = data['Time'] * (2*np.pi) / YEAR
data['Year_cos'] = np.cos(Time_in_year)
data['Year_sin'] = np.sin(Time_in_year)
```

## 4) Training

### ① Data split

Training set : Day  $\in \{1\text{일}, \dots, 217\text{일}\}$

Validation set : Day  $\in \{218\text{일}, \dots, 241\text{일}\}$

- Input sequence의 길이: 2일
- Null값은 포함하는 샘플은 제외하여 sliding window로 생성

### ② Model

GRU (# units=32, 1 hidden layer)

→ Seq2Seq 모델로 Transformer를 먼저 사용해보았으나 오버피팅이 너무 심해 가장 간단한 모델을 사용

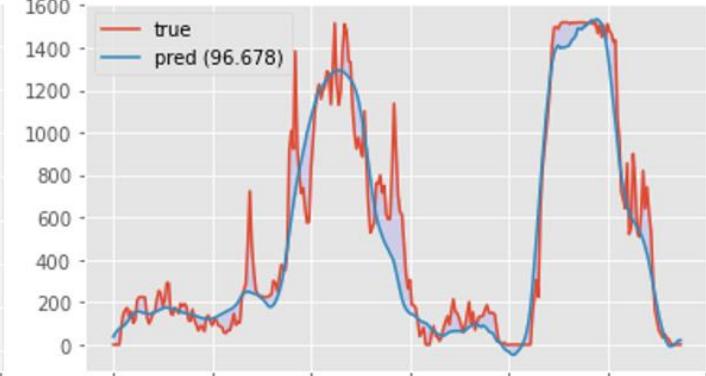
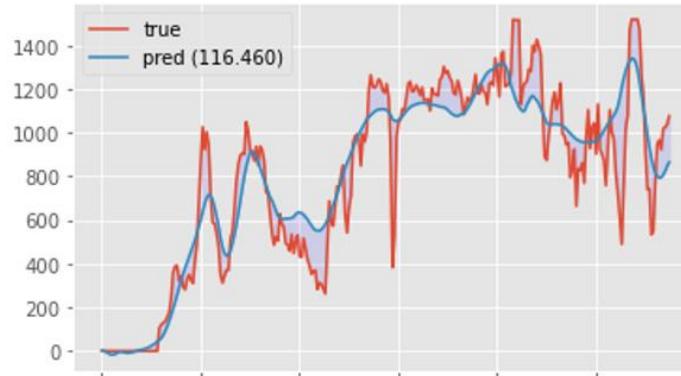
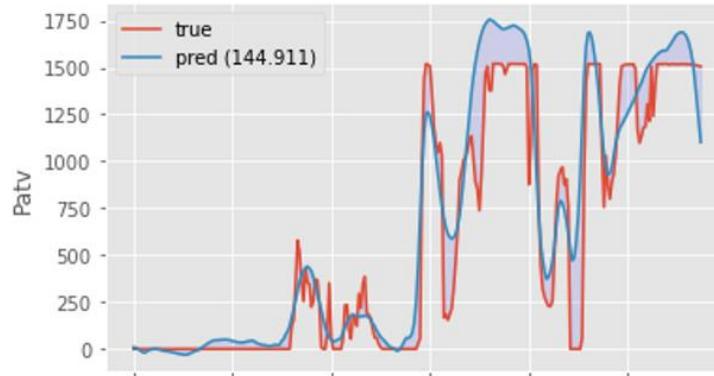
Outputs: [Wspd, Patv]

Loss : MSE

## 5) Result

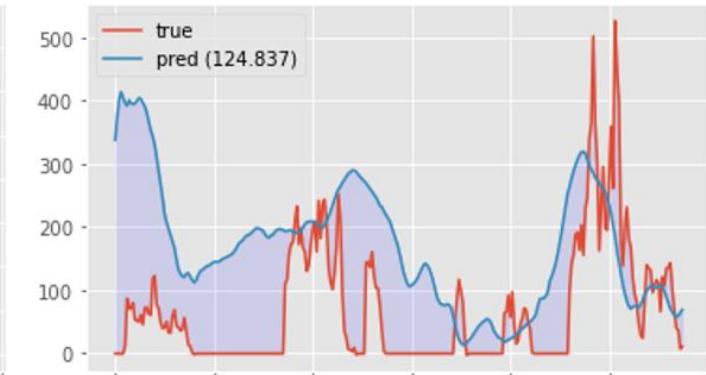
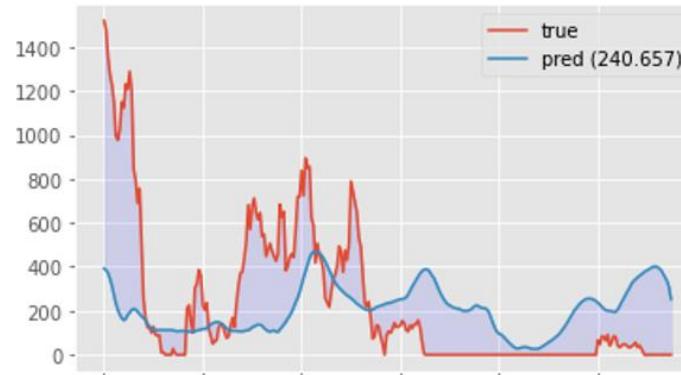
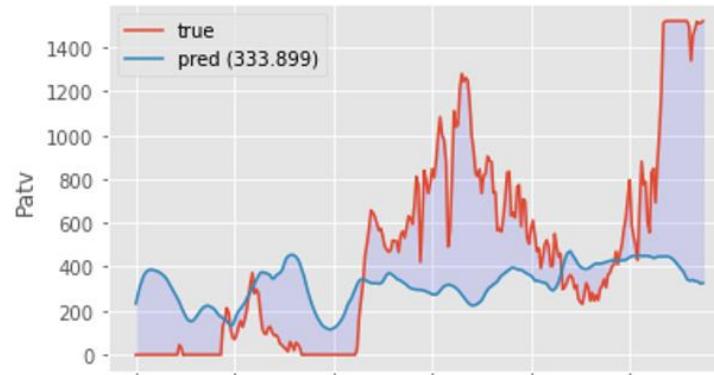
### ① Training set

평가점수 = 129.80



### ② Validation set

평가점수 = 250.07



## 5) Result

### ③ Test set

평가점수 = 168.6642

#	팀	팀 멤버	점수
1	거북이알	한상 Lo	168.6642
2	찌개사랑	종비 ai ko	178.5241
3	시계열_사나이들	He 다오 DACON pr	179.41804
4	에이셉토치	팔협 킹보	192.2706
5	아기돼지삼형제	짱짱 간지	232.10154

Test set에서 2위 팀의 점수와 비교하여 5.6% 차이를 두며 1등 수상



### 3. 결론

- 데이터의 실측값과 이상치를 처리하는 전처리 작업을 통해 좋은 성적을 낼 수 있었음
- 모델의 복잡도를 충분히 줄인 경우에도 overfitting이 발생한 것으로 보아,

1) 너무 많은 feature들을 사용하여 데이터를 구성

→ 더욱 유의미한 feature들을 선택

2) Training set의 데이터와 validation set의 분포, test set의 분포의 차이가 심함

→ 학습 구간을 validation set 근처로 축소시키고 validation set 구간의 길이를 줄이는 것도 고려해볼만함

3) 모델이 데이터의 주요한 패턴을 제대로 학습하지 못함

→ 유의미한 feature들을 선택한 후, 개선된 시계열 예측 모델을 사용

4) 다양한 크기의 sequence length를 고려

→ 적절한 크기의 time dependency를 선택함으로써 오버피팅을 방지

5) 각 터빈의 위치 정보를 직접적으로 사용

→ 위치 정보를 기반으로 GNN 모델을 사용

다양한 수행을 통해 개선된 결과를 얻을 수 있을 것으로 생각됨

---

# CONTENTS

Court Decision Prediction

---

# Court Decision Prediction

## 1. 프로젝트 소개

법률 사건에 대한 판결을 예측하는 프로젝트

- GitHub      <https://github.com/alchemine/court-decision-prediction>
- Dacon      <https://dacon.io/competitions/official/236112>

## 2. 상세 설명

- 미국 대법원 판결에 대한 원고와 피고, 판결의 근거가 되는 사실관계로부터 판결 결과(원고의 승소 여부)를 예측하는 binary classification
- 각 class의 개수가 약 1:2 정도로 **imbalance**가 존재하여 accuracy 뿐만 아니라 **precision, recall**을 제대로 관리할 수 있는 능력이 필요
- 총 3가지 방법론을 사용
  - Grouping한 원고와 피고를 입력으로 하여 판결 결과를 예측
  - 원고와 피고, 사실관계를 전처리 및 TF-IDF embedding 하여 판결 결과를 예측
  - 사실관계를 포함한 prompt를 입력으로 하는 LLM을 이용하여 판결 결과를 예측

## 2. 상세 설명 (continued)

### 1) Data Overview

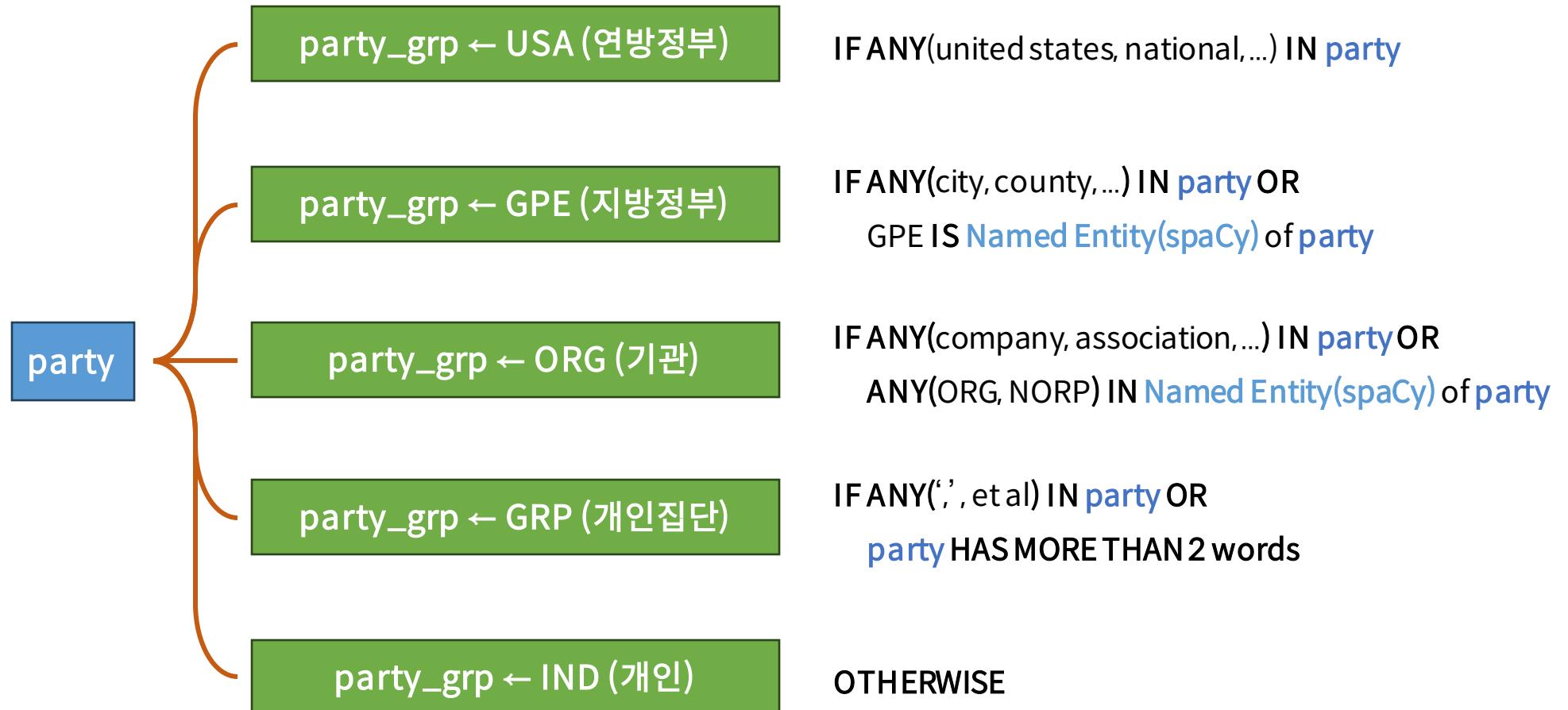
Features

	ID	first_party	second_party	facts	first_party_winner
0	TRAIN_0000	Phil A. St. Amant	Herman A. Thompson	<p>On June 27, 1962, Phil St. Amant, a candidate for public office, made a television speech in Baton Rouge, Louisiana. During this speech, St. Amant accused his political opponent of being a Communist and of being involved in criminal activities with the head of the local Teamsters Union. Finally, St. Amant implicated Herman Thompson, an East Baton Rouge deputy sheriff, in a scheme to move money between the Teamsters Union and St. Amant's political opponent. #nThompson successfully sued St. Amant for defamation. Louisiana's First Circuit Court of Appeals reversed, holding that Thompson did not show St. Amant acted with "malice." Thompson then appealed to the Supreme Court of Louisiana. That court held that, although public figures forfeit some of their First Amendment protection from defamation, St. Amant accused Thompson of a crime with utter disregard of whether the remarks were true. Finally, that court held that the First Amendment protects uninhibited, robust debate, rathe...</p>	1

## 2. 상세 설명 (continued)

### 2) Grouping한 원고와 피고를 입력으로 하여 판결 결과를 예측

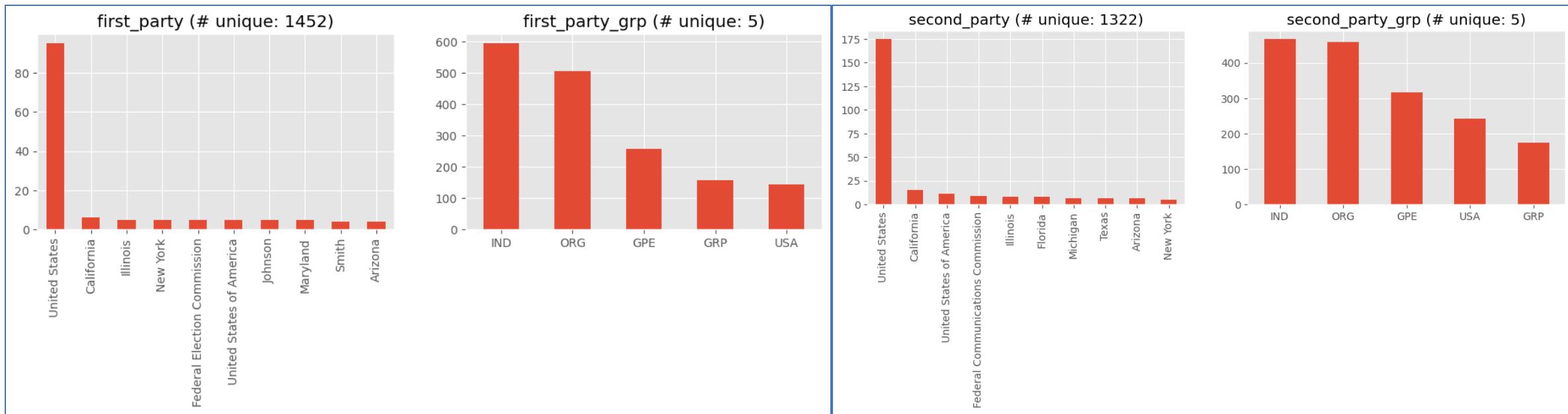
- ① Grouping: first\_party, second\_party → first\_party\_grp, second\_party\_grp



## 2. 상세 설명 (continued)

### 2) Grouping한 원고와 피고를 입력으로 하여 판결 결과를 예측 (continued)

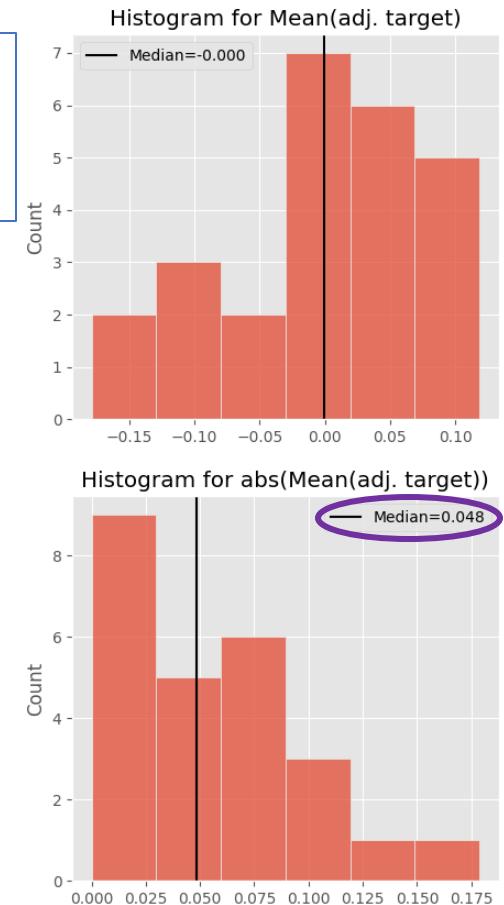
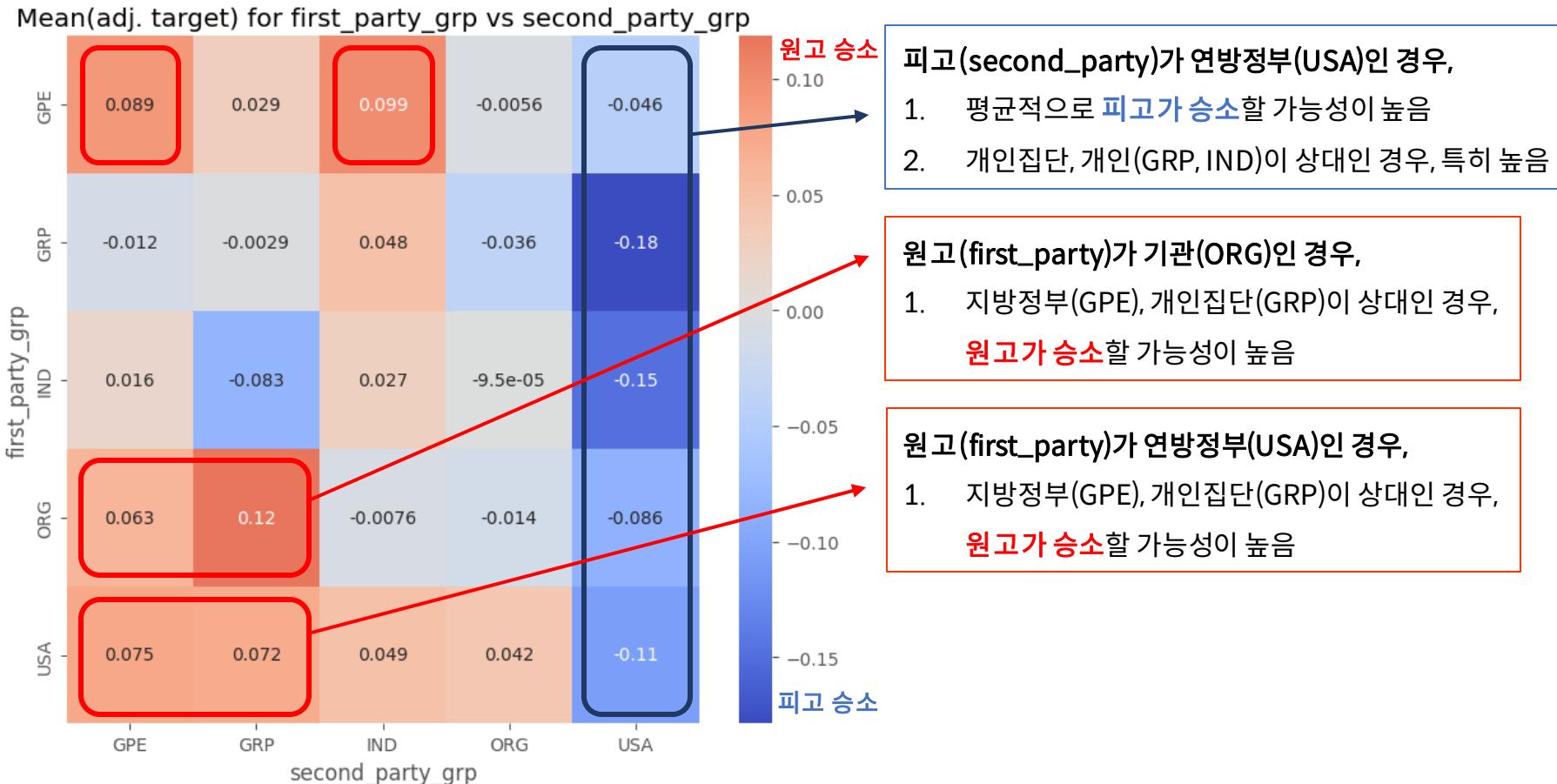
#### ① Grouping: first\_party, second\_party → first\_party\_grp, second\_party\_grp (continued)



## 2. 상세 설명 (continued)

### 2) Grouping한 원고와 피고를 입력으로 하여 판결 결과를 예측 (continued)

#### ① Grouping: first\_party, second\_party → first\_party\_grp, second\_party\_grp (continued)

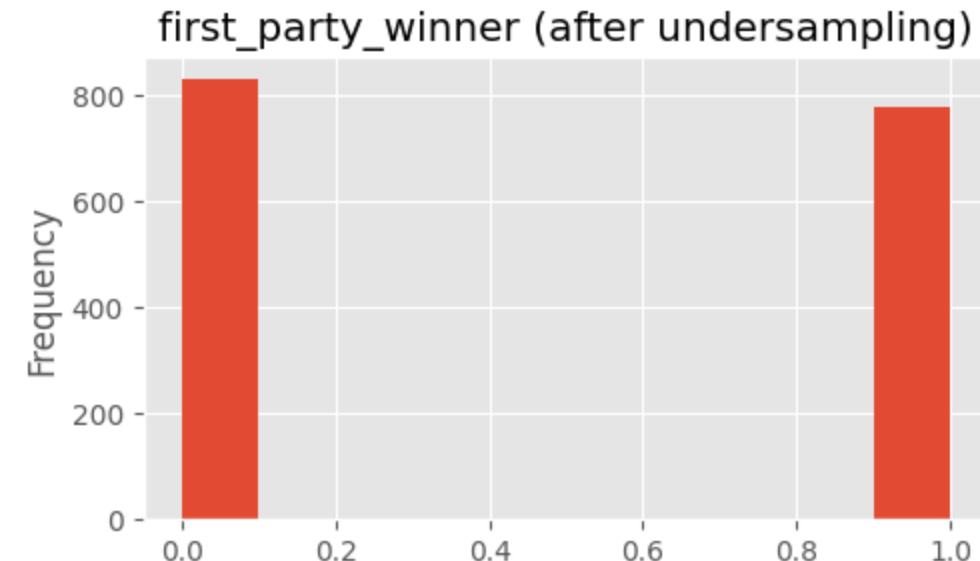
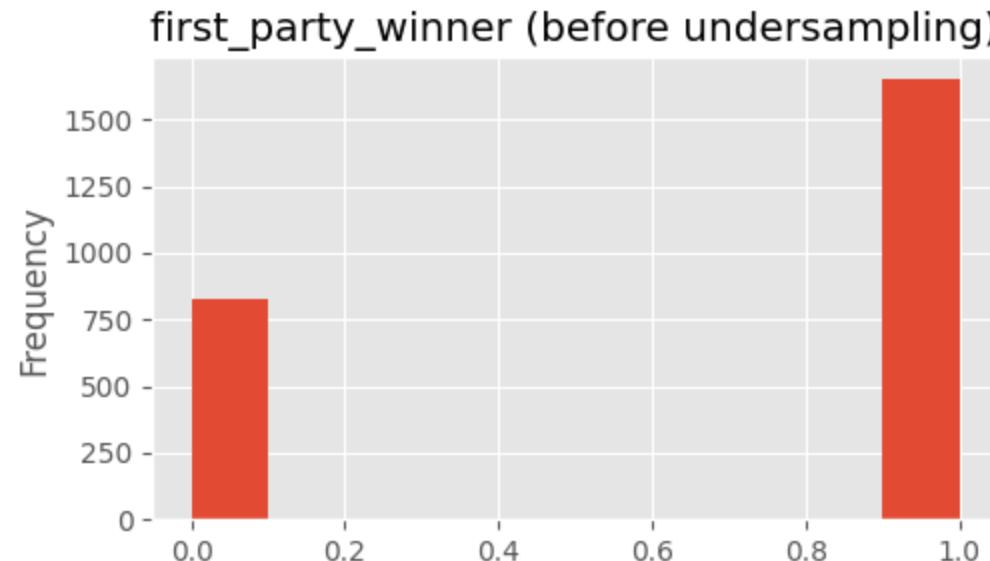


## 2. 상세 설명 (continued)

### 2) Grouping한 원고와 피고를 입력으로 하여 판결 결과를 예측 (continued)

#### ② Undersampling: Neighbourhood Cleaning Rule

kNN-neighbourhood 기반 노이즈 샘플들을 제거

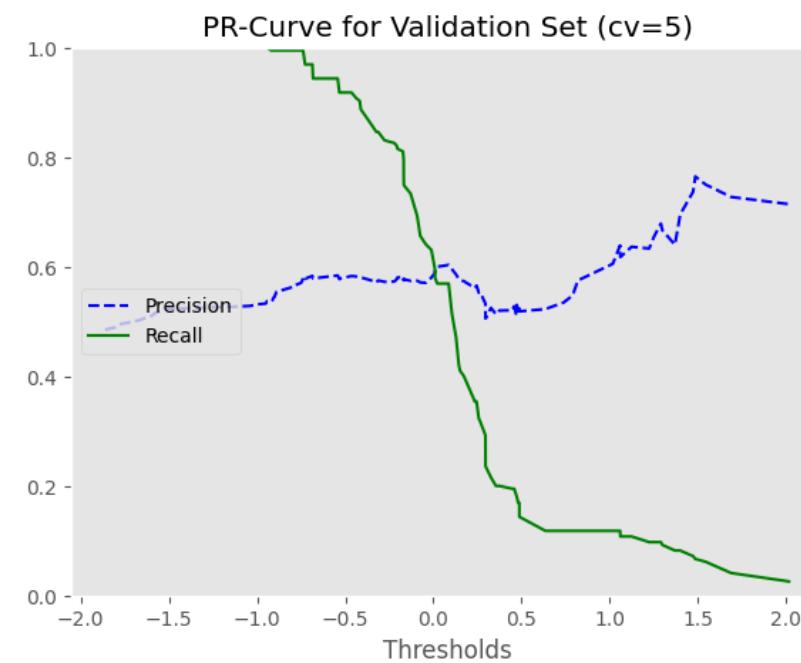
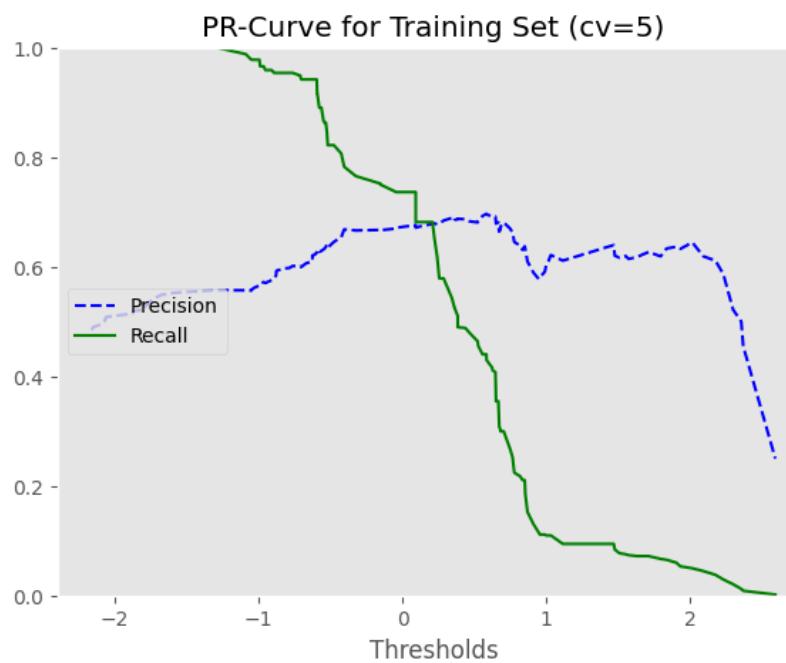


## 2. 상세 설명 (continued)

### 2) Grouping한 원고와 피고를 입력으로 하여 판결 결과를 예측 (continued)

#### ③ PCA + LogisticRegression 학습 결과

복잡한 모델(CatBoost, LGBM 등) 사용 시, recall=1



#### Validation score

	precision	recall	f1-score	support
0	0.64	0.55	0.59	207
1	0.59	0.68	0.63	195
accuracy			0.61	402
macro avg	0.62	0.61	0.61	402
weighted avg	0.62	0.61	0.61	402

## 2. 상세 설명 (continued)

### 3) 원고와 피고, 사실관계를 전처리 및 embedding하여 판결 결과를 예측

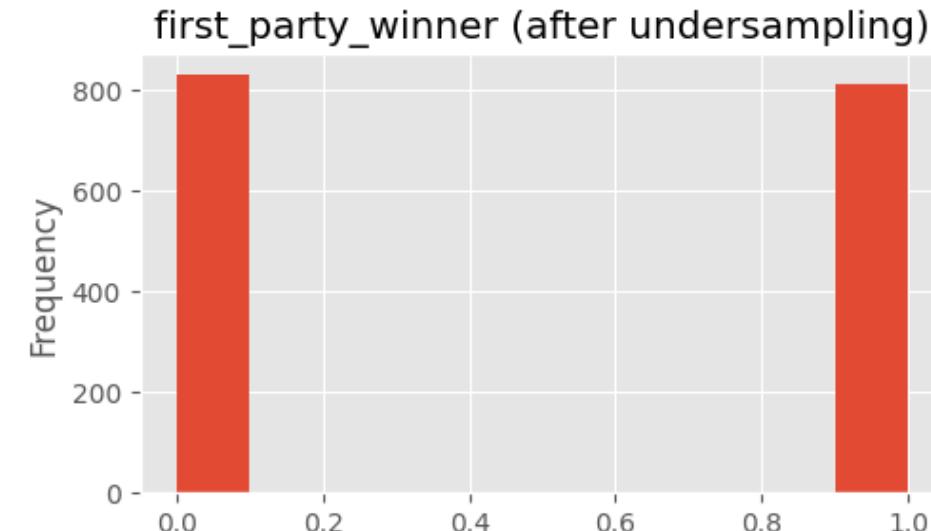
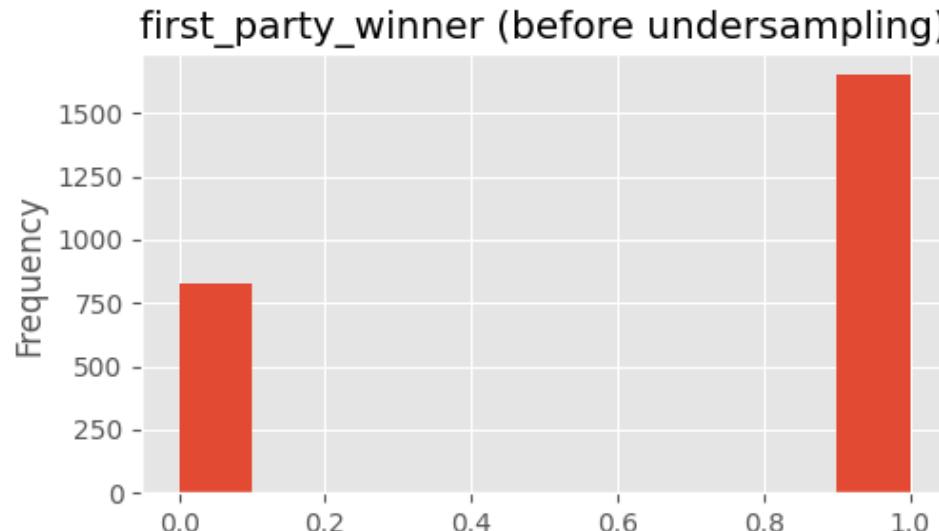
#### ① Preprocessing Text

- **Cleaning**: Corpus에서 한 글자 단어, 공백, 특수문자 등 노이즈 데이터 제거
- **Tokenization**: nltk.tokenize.TreebankWordTokenizer 사용
- **Stopword**: nltk.corpus.stopwords 사용
- **Lemmatization**: nltk.stem.WordNetLemmatizer 사용
- **Vectorization**: CountVectorizer(원고/피고), TfidfVectorizer(사실관계)
- **Concatenation**

## 2. 상세 설명 (continued)

3) 원고와 피고, 사실관계를 전처리 및 embedding하여 판결 결과를 예측

### ② Undersampling: Neighbourhood Cleaning Rule

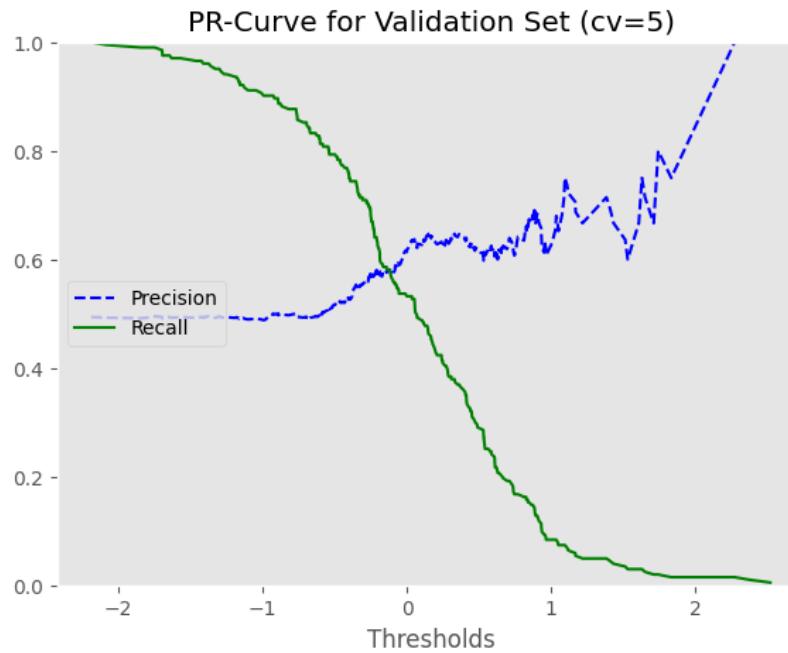
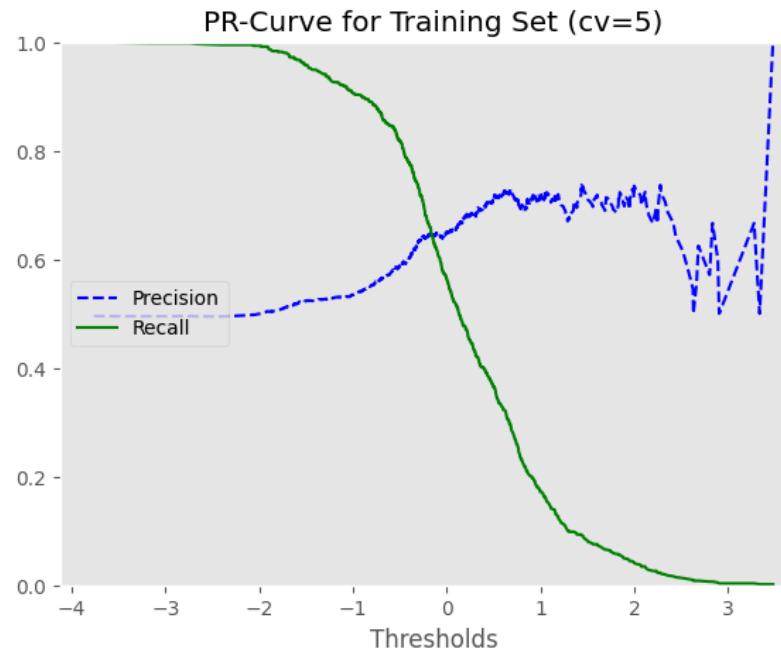


## 2. 상세 설명 (continued)

### 3) 원고와 피고, 사실관계를 전처리 및 embedding하여 판결 결과를 예측

#### ③ PCA + LogisticRegression 학습 결과

복잡한 모델(CatBoost, LGBM 등) 사용 시, 오버피팅 발생



#### Validation score

	precision	recall	f1-score	support
0	0.67	0.72	0.69	208
1	0.68	0.63	0.66	203
accuracy			0.67	411
macro avg	0.67	0.67	0.67	411
weighted avg	0.67	0.67	0.67	411

## 2. 상세 설명 (continued)

### 4) 사실관계를 포함한 prompt를 입력으로 하는 LLM을 이용하여 판결 결과를 예측

#### ① Prompt 구성

USER:

- first\_party: **Phil A. St. Amant (IND)**
- second\_party: **Herman A. Thompson (IND)**
- facts:

Baton Rouge, Louisiana. During this speech, St. Amant accused his political opponent of being a Communist and of being involved in criminal activities with the head of the local Teamsters Union. ... Finally, that court held that the First Amendment protects uninhibited, robust debate, rather than an open season to shoot down the good name of anyone who happens to be a public servant.

- Question: Do the first\_party win the case? Answer with only 1 or 0. DO NOT ANSWER ANY OTHER WORDS.

ASSISTANT:

- Answer:

Token의 개수를 일정하게 하기 위해, facts에서 다음 조건에 맞는 문장들만을 선택하고 최대 1000자로 한정

1. 피고/원고가 포함된 문장
2. 연결어(finally, however, hence, therefore 등)로 시작하는 문장
3. 핵심단어(court, judge, adjudicate, reverse 등)가 포함된 문장 (lemma 비교)

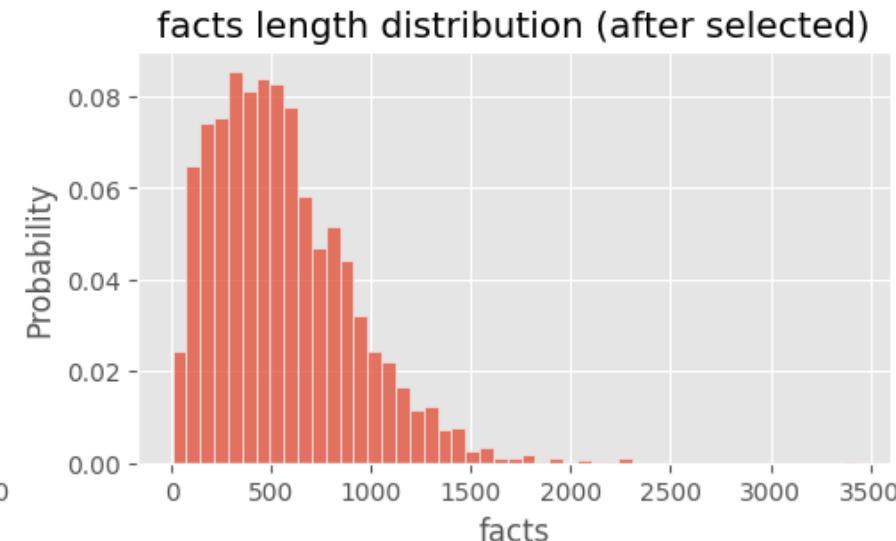
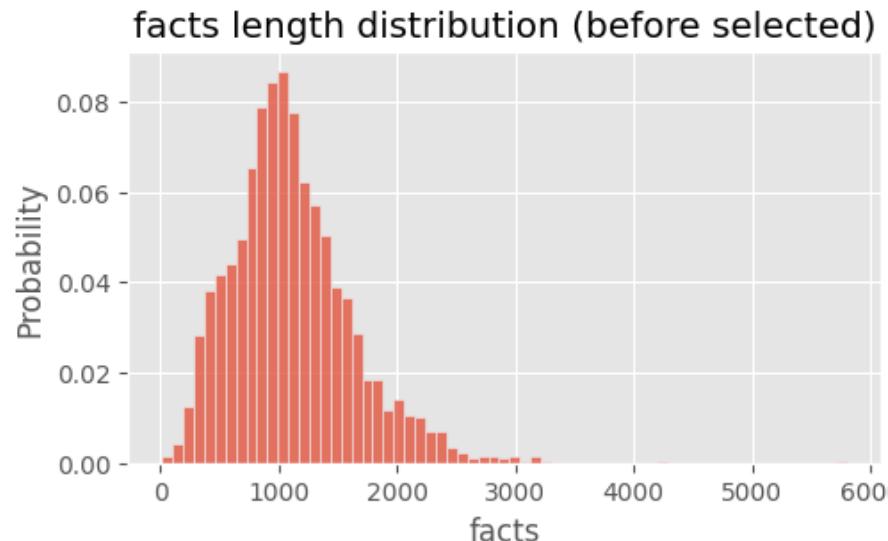
## 2. 상세 설명 (continued)

### 4) 사실관계를 포함한 prompt를 입력으로 하는 LLM을 이용하여 판결 결과를 예측

#### ① Prompt 구성 (continued)

Token의 개수를 일정하게 하기 위해, facts에서 다음 조건에 맞는 문장들만을 선택하고 뒤에서부터 최대 1000자로 한정 (PAD: EOS token)

1. 피고/원고가 포함된 문장
2. 연결어(finally, however, hence, therefore 등)로 시작하는 문장
3. 핵심단어(court, judge, adjudicate, reverse 등)가 포함된 문장 (lemma 비교)



## 2. 상세 설명 (continued)

### 4) 사실관계를 포함한 prompt를 입력으로 하는 LLM을 이용하여 판결 결과를 예측

#### ② LLM Modeling

[vicuna-13b](#) 모델 사용

Vicuna Model Card

**Model Details**

Vicuna is a chat assistant trained by fine-tuning LLaMA on user-shared conversations collected from ShareGPT.

- Developed by: [LMSYS](#)
- Model type: An auto-regressive language model based on the transformer architecture.
- License: Non-commercial license
- Finetuned from model: [LLaMA](#).

Downloads last month: 27,088

Hosted inference API: Inference API has been turned off for this model.

Spaces using lmsys/vicuna-13b-v1.3: 2

- alexkueck/ChatBotLI2Klein
- alexshengzhili/calahealthgpt

```
LlamaForCausalLM(
    (model): LlamaModel(
        (embed_tokens): Embedding(32000, 5120, padding_idx=0)
        (layers): ModuleList(
            (0-39): 40 x LlamaDecoderLayer(
                (self_attn): LlamaAttention(
                    (q_proj): Linear(in_features=5120, out_features=5120, bias=False)
                    (k_proj): Linear(in_features=5120, out_features=5120, bias=False)
                    (v_proj): Linear(in_features=5120, out_features=5120, bias=False)
                    (o_proj): Linear(in_features=5120, out_features=5120, bias=False)
                    (rotary_emb): LlamaRotaryEmbedding()
                )
                (mlp): LlamaMLP(
                    (gate_proj): Linear(in_features=5120, out_features=13824, bias=False)
                    (down_proj): Linear(in_features=13824, out_features=5120, bias=False)
                    (up_proj): Linear(in_features=5120, out_features=13824, bias=False)
                    (act_fn): SiLUActivation()
                )
                (input_layernorm): LlamaRMSNorm()
                (post_attention_layernorm): LlamaRMSNorm()
            )
        )
        (norm): LlamaRMSNorm()
    )
    (lm_head): Linear(in_features=5120, out_features=2, bias=True)
)
```

Binary classification 설정에 따라 수정

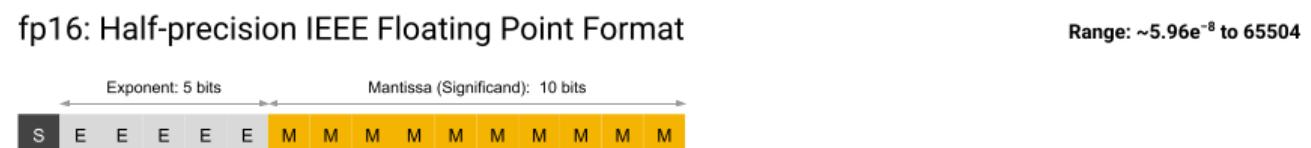
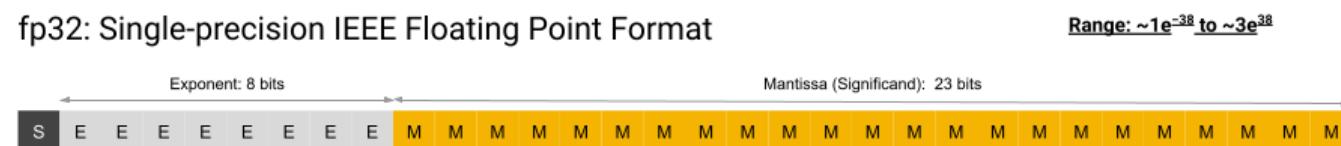
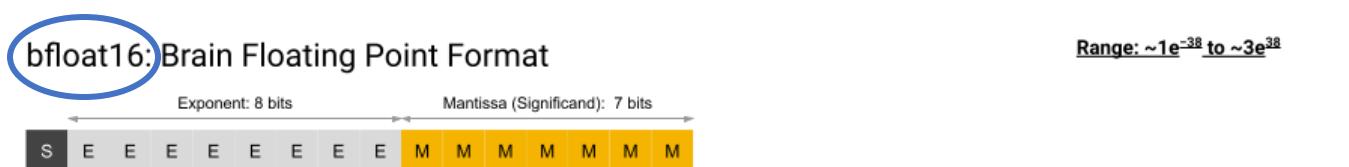
## 2. 상세 설명 (continued)

### 4) 사실관계를 포함한 prompt를 입력으로 하는 LLM을 이용하여 판결 결과를 예측

#### ② LLM Modeling (continued)

bfloating16 자료형을 이용하여 모델의 메모리 감소

### Floating Point Formats



## 2. 상세 설명 (continued)

### 4) 사실관계를 포함한 prompt를 입력으로 하는 LLM을 이용하여 판결 결과를 예측

#### ② LLM Modeling (continued)

LoRA 학습을 통해 효율적으로 새로운 데이터에 대한 fine-tuning 수행

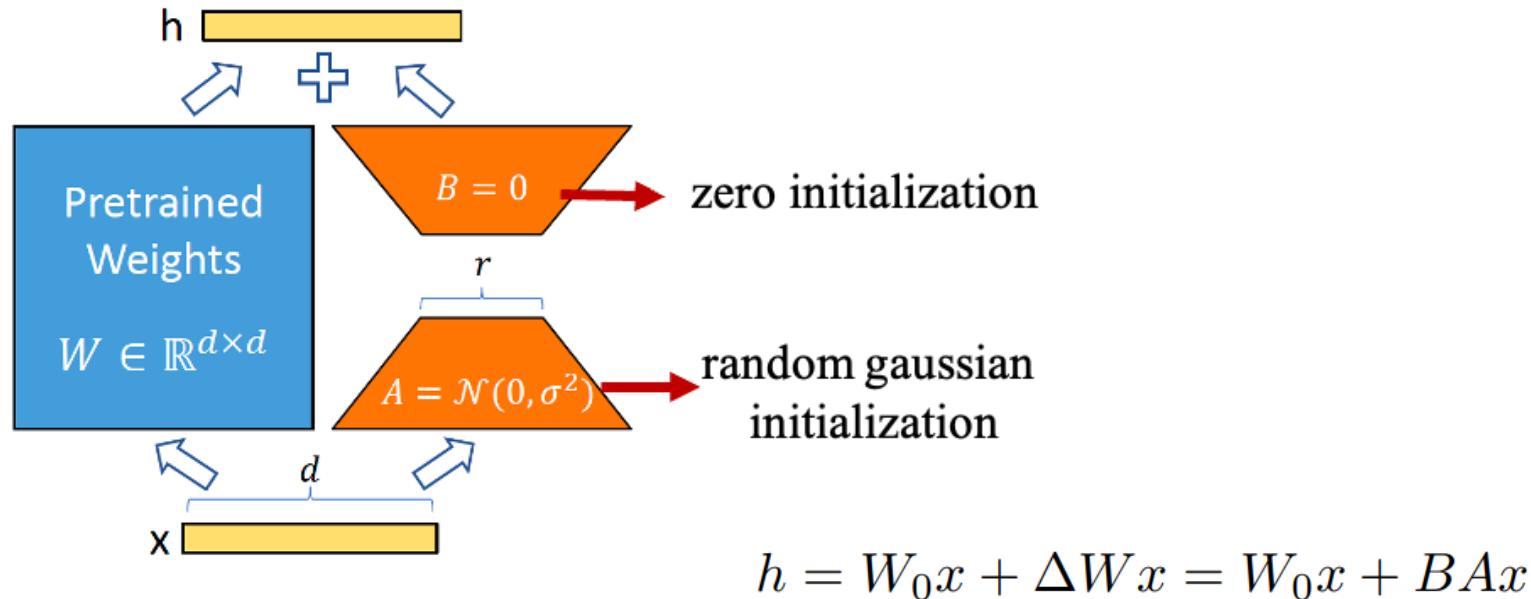


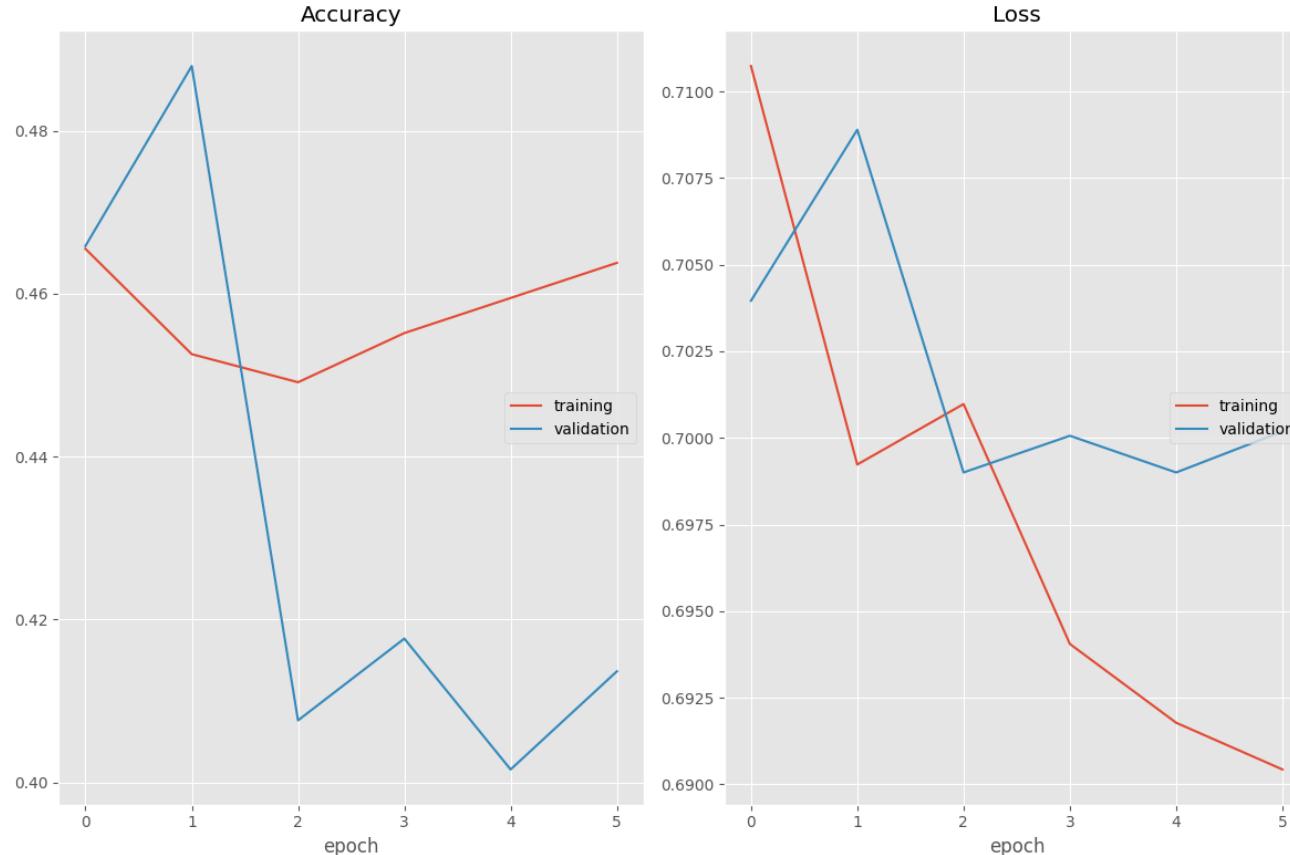
Figure 1: Our reparametrization. We only train  $A$  and  $B$ .

## 2. 상세 설명 (continued)

### 4) 사실관계를 포함한 prompt를 입력으로 하는 LLM을 이용하여 판결 결과를 예측

#### ③ 학습 결과

Training loss는 떨어지는 모습을 보여주었으나, validation loss는 제대로 학습이 되지 않았음



### 3. 결론

- 총 3가지 방법론을 사용하여 판결 결과를 예측하고자 하였음
  - Grouping한 원고와 피고를 입력으로 하여 판결 결과를 예측
  - 원고와 피고, 사실관계를 전처리 및 TF-IDF embedding 하여 판결 결과를 예측**
  - 사실관계를 포함한 prompt를 입력으로 하는 LLM을 이용하여 판결 결과를 예측
- 평균적으로 **2번 방법론**이 1번 방법론에 비하여 더 높은 precision, recall을 보여주었을 뿐만 아니라, 변동이 덜한 그래프의 모습을 보여주어 학습의 안정성을 유추해볼 수 있음
- 한편, LLM을 사용한 3번 방법론은 GPU 자원을 충분하게 사용하지 못하는 상황에서 학습이 이루어져 추가적인 학습과 hyperparameter tuning이 필요하다는 한계가 있지만, training loss와 달리 적절히 감소하지 않는 validation loss, 오버피팅이 발생한 것을 확인할 수 있었음
- 상대적으로 복잡도가 낮은 LLM을 사용한 경우도 학습의 형태가 비슷한 것으로 보아, 데이터가 복잡하고 노이즈가 많아 오버피팅이 발생한 것으로 추정됨
- 1번 방법론과 2번 방법론, 더욱 정제된 데이터셋을 사용한 3번 방법론은 각각 독립적인 모델들로, 양상을 학습 시, 높은 수준의 성능 향상을 기대해볼 수 있을 것으로 기대됨

THANKYOU

감사합니다