고용구조와 재무성과 데이터 분석

인하공업전문대학 컴퓨터정보과 3학년 2학기 빅데이터 프로젝트 최종 보고서

202344083 남동관

목차

1-1 프로젝트 주제

1-1 프로젝트 주제

1. 고용구조와 재무성과의 상관관계 히트맵
2. 고용구조증감율과 매출증감율의 관계 분석 및 회귀 모델링
3. 고용구조와 영업이익의 관계 분석 및 회귀 모델링
4. 고용구조와 매출액의 관계 분석 및 회귀 모델링

1-2 프로젝트 목표

1-2 프로젝트 목표

- 기업의 고용구조가 재무성과와 어떤 방식으로 연결되는지를 분석하고, 고용구조의 변화가 실제로 재무지표에 유의미한 영향을 미치는지 검증함으로써 인사 정책이 재무적 성과에 제공할 수 있는 인사이트를 탐색한다.

- 또한 단순한 통계적 상관관계 파악을 넘어, 한 학기 동안 학습한 데이터 수집·전처리·시각화·머신러닝 기법을 실제 산업 데이터를 기반으로 종합적으로 적용해 보는 경험을 목표로 한다.

1-3 사용할 데이터 및 프로젝트 범위

1-3 사용할 데이터 및 프로젝트 범위

- DART Open API (https://opendart.fss.or.kr/)
- 1. dart_fss 라이브러리를 이용하여 코스피 기업들의 직원현황 및 재무정보 수집
- 2. pandas 라이브러리를 이용하여 데이터 정제 및 csv 파일로 저장
- 3. sklearn 라이브러리를 이용하여 머신러닝 기반 회귀 분석
- 4. matplotlib 라이브러리를 이용하여 데이터 시각화

2-1 데이터 수집 및 전처리 실행 순서

2-1 데이터 수집 및 전처리 실행 순서

```
fetch_dart_corp_code_xml()                  # 1. DART 기업 코드 XML 다운로드
kospi_list = process_krx_kospi_list()       # 2. KRX 리스트 필터링
process_match_corp_codes(kospi_list)        # 3. KRX-DART 코드 매칭

fetch_employment_raw_data()                 # 4. 직원 현황 Raw 데이터 수집
process_combine_emp_by_corp()               # 5. 직원 데이터 사업부문 통합
process_filter_valid_emp_data()             # 6. 데이터가 온전한 기업 필터링
process_calculate_emp_metrics()             # 7. 직원 데이터 최종 파생 변수 및 증감율 계산

fetch_financial_raw_data()                  # 8. 재무 데이터 Raw 수집
process_adjust_financial_half()             # 9. 사업보고서 하반기 실적 보정
process_calculate_financial_rates()         # 10. 재무 데이터 증감율 계산

finalize_merge_data()                       # 11. 고용-재무 데이터 통합
finalize_clean_and_save()                   # 12. 최종 데이터 정제 및 저장
```

2-2 기업 코드 및 리스트 수집

2-2 DART 기업 코드 XML 다운로드

```python
# ---------------------------------------------------------------------
# 1. 기업 코드 및 리스트 수집
# ---------------------------------------------------------------------

def fetch_dart_corp_code_xml():
    """
    DART API를 통해 전체 상장 기업의 기업 코드를 다운로드하여 'corpCode.xml'로 저장함.
    데이터 수집의 선행 조건.
    """
    try:
        url = f"https://opendart.fss.or.kr/api/corpCode.xml"
        params = {
            'crtfc_key': api_key
        }
        resp = requests.get(url, params=params)
        resp.raise_for_status() # HTTP 오류 발생 시 예외 발생

        # 응답 내용을 XML 파일로 저장
        with open("corpCode.xml", "wb") as f:
            f.write(resp.content)
        print("XML 파일 다운로드 완료!")

    except Exception as e:
        print("실패:", e)
```

2-2 KRX 리스트 필터링

```python
def process_krx_kospi_list():
    """
    KRX 상장법인 목록 CSV를 불러와 KOSPI 시장, 2014년 이전 상장, 12월 결산 법인만을 필터링함.
    분석 대상 기업 리스트를 정의함.
    """
    file_path = 'data/상장법인목록.csv'

    # KRX CSV 파일 인코딩 ('cp949')으로 읽기
    df = pd.read_csv(file_path, encoding='cp949')

    # 상장일을 datetime 객체로 변환
    df['상장일'] = pd.to_datetime(df['상장일'], errors='coerce')

    # 분석 대상 기업 조건 필터링: KOSPI ('유가'), 오래된 상장사, 12월 결산
    kospi_list = df[
        (df['시장구분'] == '유가') &
        (df['상장일'] <= '2014-01-01') &
        (df['결산월'] == '12월')
    ]

    print(f"조건에 맞는 기업 수: {len(kospi_list)}개")

    return kospi_list
```

2-2 KRX-DART 기업코드 매칭

```python
def process_match_corp_codes(kospi_list):
    """
    KRX 리스트의 회사명과 DART 기업 코드 리스트를 매칭하여 최종 분석 대상 기업 코드를 확보함.
    결과를 'kospi_corps.csv'로 저장함.
    """
    # DART 기업 코드 XML 파일 불러오기
    tree = ET.parse('data/dart_corp_list.xml')
    root = tree.getroot()

    # XML 리스트를 DataFrame으로 변환
    corp_list = []
    for lst in root.findall('list'):
        corp_list.append({
            'corp_code': lst.find('corp_code').text,
            'corp_name': lst.find('corp_name').text,
            'stock_code': lst.find('stock_code').text,
            'modify_date': lst.find('modify_date').text
        })
    dart_df = pd.DataFrame(corp_list)

    # KRX 리스트와 DART 코드를 '회사명' 기준으로 Left Join
    kospi_df = kospi_list.merge(dart_df[['corp_name', 'corp_code']],
                                left_on='회사명', right_on='corp_name', how='left')
    kospi_df = kospi_df.rename(columns={'회사명':'corp_name'})

    # DART 코드를 8자리 문자열로 포맷팅 (엑셀 인식 오류 방지)
    kospi_df['corp_code'] = kospi_df['corp_code'].astype(str).str.zfill(8)
    kospi_df['corp_code'] = kospi_df['corp_code'].apply(lambda x: f'"{x}"')

    kospi_df.to_csv('./data/kospi_corps.csv', encoding="utf-8-sig", index=False)
    print('kospi_corps.csv 저장 완료')
```

2-3 고용 현황 데이터 수집

2-3 직원 현황 Raw 데이터 수집

```python
def fetch_employment_raw_data():
    """
    선정된 KOSPI 기업들을 대상으로 2016년부터 2024년까지의 직원 현황(empSttus) 정보를 DART API에서 수집함.
    반기보고서와 사업보고서를 모두 수집하여 시간 경과에 따른 변화를 관찰할 수 있도록 함.
    """
    kospi_df = pd.read_csv('./data/kospi_corps.csv', encoding='utf-8-sig')
    result = []

    # DART 직원현황 API에서 필요한 컬럼 목록
    needed_cols = [
        'corp_code', 'corp_name', 'sexdstn', 'fo_bbm', 'rgllbr_co',
        'cnttk_co', 'sm', 'fyer_salary_totamt',
    ]

    for row in kospi_df.itertuples():
        corp_name = row.corp_name
        # DART 코드는 따옴표 제거 후 사용
        corp_code = row.corp_code.replace('"','')
```

2-3 직원 현황 Raw 데이터 수집

110	302926	현대로템	남	공통부문	272	14	286	9,589,000,000	2016 반기보고서
111	302926	현대로템	여	공통부문	10	23	33	465,000,000	2016 반기보고서
112	302926	현대로템	남	철도부문	1,715	71	1,786	56,932,000,000	2016 반기보고서
113	302926	현대로템	여	철도부문	27	19	46	902,000,000	2016 반기보고서
114	302926	현대로템	남	중기부문	897	21	918	29,549,000,000	2016 반기보고서
115	302926	현대로템	여	중기부문	12	3	15	375,000,000	2016 반기보고서
116	302926	현대로템	남	플랜트부문	417	54	471	15,952,000,000	2016 반기보고서
117	302926	현대로템	여	플랜트부문	4	6	10	163,000,000	2016 반기보고서
118	302926	현대로템	남	공통부문	273	16	289	24,453,000,000	2016 사업보고서
119	302926	현대로템	여	공통부문	11	19	30	1,161,000,000	2016 사업보고서
120	302926	현대로템	남	철도부문	1,724	83	1,807	153,617,000,000	2016 사업보고서
121	302926	현대로템	여	철도부문	26	13	39	2,219,000,000	2016 사업보고서
122	302926	현대로템	남	중기부문	899	15	914	80,839,000,000	2016 사업보고서
123	302926	현대로템	여	중기부문	12	2	14	1,042,000,000	2016 사업보고서
124	302926	현대로템	남	플랜트부문	417	56	473	37,965,000,000	2016 사업보고서
125	302926	현대로템	여	플랜트부문	4	5	9	377,000,000	2016 사업보고서
126	302926	현대로템	남	공통부문	295	20	315	10,855,000,000	2017 반기보고서
127	302926	현대로템	여	공통부문	14	19	33	594,000,000	2017 반기보고서
128	302926	현대로템	남	철도부문	1,642	80	1,722	57,294,000,000	2017 반기보고서
129	302926	현대로템	여	철도부문	22	10	32	705,000,000	2017 반기보고서
130	302926	현대로템	남	방산부문	959	9	968	33,767,000,000	2017 반기보고서
131	302926	현대로템	여	방산부문	12	1	13	388,000,000	2017 반기보고서
132	302926	현대로템	남	플랜트부문	412	52	464	15,996,000,000	2017 반기보고서
133	302926	현대로템	여	플랜트부문	5	3	8	181,000,000	2017 반기보고서
134	302926	현대로템	남	공통부문	300	12	312	26,137,000,000	2017 사업보고서
135	302926	현대로템	여	공통부문	14	16	30	1,351,000,000	2017 사업보고서
136	302926	현대로템	남	철도부문	1,639	66	1,705	144,154,000,000	2017 사업보고서
137	302926	현대로템	여	철도부문	23	7	30	1,759,000,000	2017 사업보고서
138	302926	현대로템	남	방산부문	954	12	966	83,623,000,000	2017 사업보고서
139	302926	현대로템	여	방산부문	11	-	11	863,000,000	2017 사업보고서
140	302926	현대로템	남	플랜트부문	432	62	494	41,403,000,000	2017 사업보고서
141	302926	현대로템	여	플랜트부문	4	4	8	372,000,000	2017 사업보고서
142	302926	현대로템	남	공통부문	389	11	400	14,167,000,000	2018 반기보고서
143	302926	현대로템	여	공통부문	14	15	29	522,000,000	2018 반기보고서
144	302926	현대로템	남	철도부문	1,598	55	1,653	58,793,000,000	2018 반기보고서
145	302926	현대로템	여	철도부문	26	5	31	693,000,000	2018 반기보고서
146	302926	현대로템	남	방산부문	885	10	895	32,812,000,000	2018 반기보고서
147	302926	현대로템	여	방산부문	11	-	11	356,000,000	2018 반기보고서
148	302926	현대로템	남	플랜트부문	415	64	479	16,947,000,000	2018 반기보고서
149	302926	현대로템	여	플랜트부문	3	3	6	122,000,000	2018 반기보고서
150	302926	현대로템	남	공통부문	412	11	423	36,239,000,000	2018 사업보고서
151	302926	현대로템	여	공통부문	14	14	28	1,282,000,000	2018 사업보고서
152	302926	현대로템	남	철도부문	1,684	62	1,746	149,875,000,000	2018 사업보고서
153	302926	현대로템	여	철도부문	26	5	31	1,775,000,000	2018 사업보고서
154	302926	현대로템	남	방산부문	894	9	903	81,255,000,000	2018 사업보고서
155	302926	현대로템	여	방산부문	12	-	12	879,000,000	2018 사업보고서

```python
    for year in range(2016, 2025):
        # '11012': 반기보고서, '11011': 사업보고서
        for reprt_code, reprt_name in [('11012', '반기보고서'), ('11011', '사업보고서')]:

            try:
                url = "https://opendart.fss.or.kr/api/empSttus.json"
                params = {
                    'crtfc_key': api_key,
                    'corp_code': corp_code,
                    'bsns_year': year,
                    'reprt_code': reprt_code
                }

                resp = requests.get(url, params=params)
                js = resp.json()

                if js.get('status') == '000':
                    emp_list = js['list']
                    emp_df = pd.DataFrame(emp_list)

                    # 필요한 컬럼만 추출 및 보고서 정보 추가
                    emp_df = emp_df[[col for col in needed_cols if col in emp_df.columns]]
                    emp_df['연도'] = year
                    emp_df['보고서유형'] = reprt_name

                    # 컬럼명 한글화 (분석 용이성 확보)
                    emp_df = emp_df.rename(columns={
                        'corp_code' : '종목코드', 'corp_name' : '기업명',
                        'sexdstn' : '성별', 'fo_bbm' : '사업부문',
                        'rgllbr_co' : '정규직', 'cnttk_co' : '계약직',
                        'sm' : '직원합', 'fyer_salary_totamt' : '총급여'
                    })

                    result.append(emp_df)
                else:
                    print(f"데이터 없음: {js.get('message')}")

                print(f"{corp_name} {year}년 {reprt_name} 수집 완료")
                time.sleep(0.3) # API 부하 방지 및 제한 회피를 위한 대기

            except Exception as e:
                print(f"{year}년 {reprt_name} 실패: {e}")

# 결과 합치기 및 CSV 저장
os.makedirs('./data', exist_ok=True)
if result:
    final_df = pd.concat(result, ignore_index=True)
else:
    final_df = pd.DataFrame()
final_df.to_csv('./data/emp_raw.csv', encoding="utf-8-sig", index=False)
print('emp_raw.csv 저장 완료')
```

2-3 직원 데이터 사업부문 통합

```python
def process_combine_emp_by_corp():
    """
    수집된 직원 현황 Raw 데이터를 기업, 연도, 보고서유형, 성별 기준으로 통합하고 숫자형으로 변환함.
    사업부문별로 분리된 데이터를 통합하여 기업 전체의 합산 데이터를 만듦.
    """
    emp_raw_df = pd.read_csv('./data/emp_raw.csv')

    # 숫자 타입 변환 및 콤마 제거 처리 (데이터 정제)
    for col in ['정규직', '계약직', '직원합', '총급여']:
        emp_raw_df[col] = emp_raw_df[col].astype(str).str.replace(',', '')
        emp_raw_df[col] = pd.to_numeric(emp_raw_df[col], errors='coerce').fillna(0)

    # 주요 식별자 기준으로 그룹화 후 합산 (사업부문 통합)
    emp_df = emp_raw_df.groupby(
        ['기업명', '연도', '보고서유형', '성별'],
        as_index=False
    ).agg({
        '정규직': 'sum',
        '계약직': 'sum',
        '총급여': 'sum'
    })

    emp_df.to_csv('./data/emp_combine.csv', encoding="utf-8-sig", index=False)
    print('emp_combine.csv 저장 완료')
```

22013	현대로템	2016	반기보고서	남	3301	160	1.12022E+11
22014	현대로템	2016	반기보고서	여	53	51	1905000000
22015	현대로템	2016	사업보고서	남	3313	170	2.96874E+11
22016	현대로템	2016	사업보고서	여	53	39	4799000000
22017	현대로템	2017	반기보고서	남	3308	161	1.17912E+11
22018	현대로템	2017	반기보고서	여	53	33	1868000000
22019	현대로템	2017	사업보고서	남	3325	152	2.95317E+11
22020	현대로템	2017	사업보고서	여	52	27	4345000000
22021	현대로템	2018	반기보고서	남	3287	140	1.22719E+11
22022	현대로템	2018	반기보고서	여	54	23	1693000000
22023	현대로템	2018	사업보고서	남	3402	137	3.0718E+11
22024	현대로템	2018	사업보고서	여	55	22	4247000000
22025	현대로템	2019	반기보고서	남	3275	141	1.30945E+11
22026	현대로템	2019	반기보고서	여	55	25	1997000000
22027	현대로템	2019	사업보고서	남	3331	144	2.9856E+11
22028	현대로템	2019	사업보고서	여	56	30	4842000000
22029	현대로템	2020	반기보고서	남	3073	198	1.2113E+11
22030	현대로템	2020	반기보고서	여	56	28	2115000000
22031	현대로템	2020	사업보고서	남	3107	217	2.88876E+11
22032	현대로템	2020	사업보고서	여	58	35	5146000000
22033	현대로템	2021	반기보고서	남	3059	247	1.26605E+11
22034	현대로템	2021	반기보고서	여	60	31	2270000000
22035	현대로템	2021	사업보고서	남	3109	243	3.06692E+11
22036	현대로템	2021	사업보고서	여	63	29	5643000000
22037	현대로템	2022	반기보고서	남	3043	269	1.26252E+11
22038	현대로템	2022	반기보고서	여	74	30	2588000000
22039	현대로템	2022	사업보고서	남	3197	273	3.32981E+11
22040	현대로템	2022	사업보고서	여	84	37	6978000000
22041	현대로템	2023	반기보고서	남	3350	260	1.48656E+11
22042	현대로템	2023	반기보고서	여	118	38	4181000000
22043	현대로템	2023	사업보고서	남	3477	285	3.93222E+11
22044	현대로템	2023	사업보고서	여	134	42	12070000000
22045	현대로템	2024	반기보고서	남	3587	318	1.60342E+11
22046	현대로템	2024	반기보고서	여	149	34	5205000000
22047	현대로템	2024	사업보고서	남	3640	333	4.83222E+11
22048	현대로템	2024	사업보고서	여	164	45	17440000000

2-3 데이터가 온전한 기업 필터링

```python
def process_filter_valid_emp_data():
    """
    통합된 직원 데이터를 대상으로 2016년부터 2024년까지의 모든 연도별/보고서유형 데이터가 온전히 존재하는
    (기업명, 성별) 쌍만을 필터링하여 분석의 일관성을 확보함.
    """
    emp_df = pd.read_csv('./data/emp_combine.csv')

    # 기준 연도 및 보고서 유형 정의
    years = list(range(2016, 2025))
    report_types = ['반기보고서', '사업보고서']

    # 각 그룹별 (연도, 보고서유형) 조합이 모두 존재하는지 확인
    grouped = emp_df.groupby(['기업명', '성별'])

    valid_idx = []
    all_pairs = set((y, r) for y in years for r in report_types)

    for (corp, gender), group in grouped:
        pairs = set([tuple(x) for x in group[['연도', '보고서유형']].values])
        # 전체 연도·보고서유형 조합이 현재 그룹의 부분집합인지 확인
        if all_pairs.issubset(pairs):
            valid_idx.append((corp, gender))

    # 유효한 인덱스만 필터링
    valid_emp_df = emp_df.set_index(['기업명', '성별']).loc[valid_idx].reset_index()

    valid_emp_df.to_csv('./data/emp_valid.csv', encoding="utf-8-sig", index=False)
    print('emp_valid.csv 저장 완료')
```

2-3 직원 데이터 최종 파생 변수 및 증감율 계산

```python
def process_calculate_emp_metrics():
    """
    유효한 직원 데이터를 기반으로 최종 분석에 사용될 파생 변수 및 테이블을 생성함.
    핵심: 연간 총 급여를 '사업보고서 총 급여'에서 '반기보고서 총 급여'를 차감하여 계산함 (중복 계산 방지).
    성별 및 정규/계약직 구분을 컬럼으로 피벗(Pivot)하여 최종 테이블을 완성함.
    """
    df = pd.read_csv('./data/emp_valid.csv')

    # 1. 숫자형 컬럼 변환 (재확인)
    numeric_cols = ['정규직', '계약직', '총급여']
    for col in numeric_cols:
        df[col] = pd.to_numeric(df[col], errors='coerce').fillna(0)

    # 2. 반기/사업 데이터 분리
    semi = df[df['보고서유형'] == '반기보고서']
    annual = df[df['보고서유형'] == '사업보고서']

    # 3. 반기-사업 Merge → 순수 연간 급여 계산
    # 사업보고서 급여(누적) - 반기보고서 급여(누적) = 하반기 급여 (순수 연간 급여)
    merged = pd.merge(
        annual, semi, on=['기업명', '연도', '성별'],
        suffixes=('_annual', '_semi'), how='left'
    )

    merged['순수연간급여'] = merged['총급여_annual'] - merged.get('총급여_semi', 0)
    # 급여가 음수이거나 0일 경우, 사업보고서의 총 급여를 연간 급여로 간주 (예외 처리)
    merged.loc[merged['순수연간급여'] <= 0, '순수연간급여'] = merged['총급여_annual']

    # 4. 사업보고서 데이터 Pivot (성별/고용 형태를 컬럼으로 변환)
    pivot_annual = merged.pivot_table(
        index=['기업명', '연도'],
        columns='성별',
        values=['정규직_annual', '계약직_annual', '순수연간급여'],
        aggfunc='sum', fill_value=0
    )
    pivot_annual.columns = ['_'.join(col).strip() for col in pivot_annual.columns.values]
    pivot_annual = pivot_annual.reset_index()

    # 5. 반기보고서 데이터 Pivot
    pivot_semi = semi.pivot_table(
        index=['기업명', '연도'],
        columns='성별',
        values=['정규직', '계약직', '총급여'],
        aggfunc='sum', fill_value=0
    )
    pivot_semi.columns = ['_'.join(col).strip() for col in pivot_semi.columns.values]
    pivot_semi = pivot_semi.reset_index()
```

```python
# 6. 사업보고서 테이블 생성 (연간 최종 급여 및 직원 수 계산)
annual_rows = []
for row in pivot_annual.itertuples():
    #... (남성/여성, 정규/계약직 인원 및 급여 추출)
    m_reg = getattr(row, '정규직_annual_남', 0)
    m_con = getattr(row, '계약직_annual_남', 0)
    m_pay = getattr(row, '순수연간급여_남', 0)
    f_reg = getattr(row, '정규직_annual_여', 0)
    f_con = getattr(row, '계약직_annual_여', 0)
    f_pay = getattr(row, '순수연간급여_여', 0)

    tot_reg, tot_con = m_reg + f_reg, m_con + f_con
    tot_pay = m_pay + f_pay
    tot_emp = tot_reg + tot_con
    avg_pay = tot_pay / tot_emp if tot_emp > 0 else None

    annual_rows.append({
        '기업명': row.기업명, '연도': row.연도, '보고서유형': '사업보고서',
        '평균급여': avg_pay, '남성직원수': m_reg + m_con, '여성직원수': f_reg + f_con,
        '정규직수': tot_reg, '계약직수': tot_con,
        '남성정규직수': m_reg, '남성계약직수': m_con,
        '여성정규직수': f_reg, '여성계약직수': f_con,
    })
annual_df = pd.DataFrame(annual_rows)

# 7. 반기보고서 데이터 생성
semi_rows = []
for row in pivot_semi.itertuples():
    #... (남성/여성, 정규/계약직 인원 및 급여 추출)
    m_reg = getattr(row, '정규직_남', 0)
    m_con = getattr(row, '계약직_남', 0)
    m_pay = getattr(row, '총급여_남', 0)
    f_reg = getattr(row, '정규직_여', 0)
    f_con = getattr(row, '계약직_여', 0)
    f_pay = getattr(row, '총급여_여', 0)

    tot_reg, tot_con = m_reg + f_reg, m_con + f_con
    tot_pay = m_pay + f_pay
    tot_emp = tot_reg + tot_con
    avg_pay = tot_pay / tot_emp if tot_emp > 0 else None

    semi_rows.append({
        '기업명': row.기업명, '연도': row.연도, '보고서유형': '반기보고서',
        '평균급여': avg_pay, '남성직원수': m_reg + m_con, '여성직원수': f_reg + f_con,
        '정규직수': tot_reg, '계약직수': tot_con,
        '남성정규직수': m_reg, '남성계약직수': m_con,
        '여성정규직수': f_reg, '여성계약직수': f_con,
    })
semi_df = pd.DataFrame(semi_rows)
```

2-3 직원 데이터 최종 파생 변수 및 증감율 계산

```python
# 8. 반기 + 사업 데이터 통합
total = pd.concat([semi_df, annual_df])
total = total.sort_values(['기업명', '연도', '보고서유형']).reset_index(drop=True)

# 9. 직원 현황 변수들의 증감율 계산
rate_cols = [
    '평균급여', '남성직원수', '여성직원수',
    '정규직수', '계약직수',
    '남성정규직수', '남성계약직수',
    '여성정규직수', '여성계약직수'
]

for col in rate_cols:
    total[col + '증감율'] = np.nan # 초기화

for corp in total['기업명'].unique():
    corp_df = total[total['기업명'] == corp]

    for year in corp_df['연도'].unique():
        ydf = corp_df[corp_df['연도'] == year]

        # (1) 반기보고서 증감율 = 전년도 사업보고서 대비 증감율
        h1 = ydf[ydf['보고서유형'] == '반기보고서']
        prev = total[
            (total['기업명'] == corp) &
            (total['연도'] == year - 1) &
            (total['보고서유형'] == '사업보고서')
        ]

        if not h1.empty and not prev.empty:
            h1_idx = h1.index[0]
            for col in rate_cols:
                before = prev[col].values[0]
                after = h1[col].values[0]
                if pd.notna(before) and before != 0:
                    total.loc[h1_idx, col + '증감율'] = (after - before) / before

        # (2) 사업보고서 증감율 = 같은 연도 반기보고서 대비 증감율
        ann = ydf[ydf['보고서유형'] == '사업보고서']
        if not ann.empty and not h1.empty:
            ann_idx = ann.index[0]
            for col in rate_cols:
                before = h1[col].values[0]
                after = ann[col].values[0]
                if pd.notna(before) and before != 0:
                    total.loc[ann_idx, col + '증감율'] = (after - before) / before
```

```python
# 10. 최종 저장
total.to_csv('./data/emp_final.csv', index=False, encoding='utf-8-sig')
print("emp_final.csv 저장 완료!")
```

기업명	연도	보고서유형	평균급여	남성직원수	여성직원수	정규직수	계약직수	남성정규직수	남성계약직수	여성정규직수	여성계약직수
현대로템	2016	사업보고서	52516363.64	3483	92	3366	209	3313	170	53	39
현대로템	2017	반기보고서	33693389.59	3469	86	3361	194	3308	161	53	33
현대로템	2017	사업보고서	50585489.31	3477	79	3377	179	3325	152	52	27
현대로템	2018	반기보고서	35505707.76	3427	77	3341	163	3287	140	54	23
현대로템	2018	사업보고서	51718750	3539	77	3457	159	3402	137	55	22
현대로템	2019	반기보고서	38026887.87	3416	80	3330	166	3275	141	55	25
현대로템	2019	사업보고서	47868576.24	3475	86	3387	174	3331	144	56	30
현대로템	2020	반기보고서	36734724.29	3271	84	3129	226	3073	198	56	28
현대로템	2020	사업보고서	49978636.23	3324	93	3165	252	3107	217	58	35
현대로템	2021	반기보고서	37937886.37	3306	91	3119	278	3059	247	60	31
현대로템	2021	사업보고서	53269454.12	3352	92	3172	272	3109	243	63	29
현대로템	2022	반기보고서	37716627.63	3312	104	3117	299	3043	269	74	30
현대로템	2022	사업보고서	58791144.53	3470	121	3281	310	3197	273	84	37
현대로템	2023	반기보고서	40583377.59	3610	156	3468	298	3350	260	118	38
현대로템	2023	사업보고서	64107414.93	3762	176	3611	327	3477	285	134	42
현대로템	2024	반기보고서	40495841.49	3905	183	3736	352	3587	318	149	34
현대로템	2024	사업보고서	80132711.62	3973	209	3804	378	3640	333	164	45

평균급여증감율	남성직원수증감율	여성직원수증감율	정규직수증감율	계약직수증감율	남성정규직수증감율	남성계약직수증감율	여성정규직수증감율	여성계약직수증감율
64.33403527	0.635654435	-11.53846154	0.357781753	-0.947867299	0.363526204	6.25	0	-23.52941176
-35.84211233	-0.40195234	-6.52173913	-0.148544266	-7.177033493	-0.150920616	-5.294117647	0	-15.38461538
50.13475915	0.23061401	-8.139534884	0.476048795	-7.731958763	0.513905683	-5.590062112	-1.886792453	-18.18181818
-29.81048865	-1.438021283	-2.53164557	-1.066034942	-8.938547486	-1.142857143	-7.894736842	3.846153846	-14.81481481
45.66319969	3.268164575	0	3.472014367	-2.45398773	3.49863097	-2.142857143	1.851851852	-4.347826087
-26.47369112	-3.475558067	3.896103896	-3.673705525	4.402515723	-3.733098178	2.919708029	0	13.63636364
25.88086725	1.727166276	7.5	1.711711712	4.819277108	1.709923664	2.127659574	1.818181818	20
-23.25920849	-5.870503597	-2.325581395	-7.617360496	29.88505747	-7.745421795	37.5	0	-6.666666667
36.05284154	1.620299603	10.71428571	1.150527325	11.50442478	1.106410674	9.595959596	3.571428571	25
-24.09179355	-0.541516245	-2.150537634	-1.453396524	10.31746032	-1.544898616	13.82488479	3.448275862	-11.42857143
40.41228761	1.391409558	1.098901099	1.699262584	-2.158273381	1.634521085	-1.619433198	5	-6.451612903
-29.19651936	-1.193317422	13.04347826	-1.733921816	9.926470588	-2.12286909	10.69958848	17.46031746	3.448275862
55.87593116	4.770531401	16.34615385	5.261469362	3.678929766	5.060795268	1.486988848	13.51351351	23.33333333
-30.97025425	4.034582133	28.92561983	5.699481865	-3.870967742	4.785736628	-4.761904762	40.47619048	2.702702703
57.96471053	4.210526316	12.82051282	4.123414072	9.731543624	3.791044776	9.615384615	13.55932203	10.52631579
-36.83126744	3.801169591	3.977272727	3.461644974	7.645259939	3.163646822	11.57894737	11.19402985	-19.04761905
97.87886528	1.741357234	14.20765027	1.82012848	7.386363636	1.477557848	4.716981132	10.06711409	32.35294118

2-4 재무 성과 데이터 수집 및 전처리

2-4 재무 데이터 Raw 수집

```python
def fetch_financial_raw_data():
    """
    선정된 KOSPI 기업들을 대상으로 2016년부터 2024년까지의 재무제표(fnlttSinglAcnt) 정보를 수집함.
    주요 항목인 '매출액'과 '영업이익'을 통합재무제표(CFS) 또는 개별재무제표(OFS)에서 추출함.
    """
    kospi_df = pd.read_csv('./data/kospi_corps.csv', encoding='utf-8-sig')
    kospi_count = len(kospi_df)
    result = []

    for row in kospi_df.itertuples():
        corp_name = row.corp_name
        corp_code = row.corp_code.replace("'",'')

        for year in range(2016, 2025):
            for reprt_code, reprt_name in [('11012', '반기보고서'), ('11011', '사업보고서')]:

                try:
                    url = "https://opendart.fss.or.kr/api/fnlttSinglAcnt.json"
                    params = {
                        'crtfc_key': api_key, 'corp_code': corp_code,
                        'bsns_year': year, 'reprt_code': reprt_code
                    }

                    resp = requests.get(url, params=params)
                    js = resp.json()

                    if js.get('status') == '000':
                        df = pd.DataFrame(js['list'])

                        cfs_df = df[df['fs_div'] == 'CFS'] # 연결재무제표 (우선 사용)
                        ofs_df = df[df['fs_div'] == 'OFS'] # 개별/별도재무제표

                        if not cfs_df.empty:
                            df = cfs_df
                        elif not ofs_df.empty:
                            df = ofs_df
                        else:
                            continue

                        # 매출액, 영업이익 추출 (반기보고서는 thstrm_add_amount, 사업보고서는 thstrm_amount 사용)
                        if reprt_name == "반기보고서":
                            sales = df[df["account_nm"] == "매출액"]["thstrm_add_amount"]
                            op_profit = df[df["account_nm"].isin(["영업이익", "영업손익"])]["thstrm_add_amount"]
                        else:
                            sales = df[df["account_nm"] == "매출액"]["thstrm_amount"]
                            op_profit = df[df["account_nm"].isin(["영업이익", "영업손익"])]["thstrm_amount"]

                        result.append({
                            "기업명": corp_name, "연도": year, "보고서유형": reprt_name,
                            "매출액": sales.iloc[0] if not sales.empty else None,
                            "영업이익": op_profit.iloc[0] if not op_profit.empty else None
                        })
                    else:
                        print(f"데이터 없음: {js.get('message')}")

                    print(f"{corp_name} {year}년 {reprt_name} 수집 완료")
                    time.sleep(0.3)

                except Exception as e:
                    print(f"{year}년 {reprt_name} 실패: {e}")

# 결과 합치기 및 CSV 저장
os.makedirs('./data', exist_ok=True)
if result:
    final_df = pd.DataFrame(result)
else:
    final_df = pd.DataFrame()
final_df.to_csv('./data/financial_raw.csv', encoding="utf-8-sig", index=False)
print('financial_raw.csv 저장 완료')
```

2-4 재무 데이터 Raw 수집

기업명	연도	보고서유형	매출액	영업이익
종근당	2016	반기보고서	407,634,598,539	18,830,521,450
종근당	2016	사업보고서	831,985,529,732	61,249,544,720
종근당	2017	반기보고서	420,707,254,809	33,094,684,925
종근당	2017	사업보고서	884,362,807,619	77,764,992,125
신송홀딩스	2016	반기보고서	107,505,681,062	-2,533,080,142
신송홀딩스	2016	사업보고서	206,895,556,130	-8,056,999,711
신송홀딩스	2017	반기보고서	104,968,709,468	-2,495,670,166
신송홀딩스	2017	사업보고서	291,761,499,391	-690,876,702
아세아시멘트	2016	반기보고서	215,668,435,966	20,020,402,369
아세아시멘트	2016	사업보고서	455,724,888,278	56,975,798,773
아세아시멘트	2017	반기보고서	226,728,646,265	23,042,877,165
아세아시멘트	2017	사업보고서	461,167,085,734	53,220,127,166
현대로템	2016	반기보고서	1,446,612,049,000	69,786,243,000
현대로템	2016	사업보고서	2,984,783,033,000	106,233,799,000
현대로템	2017	반기보고서	1,320,411,625,000	50,143,925,000
현대로템	2017	사업보고서	2,725,658,171,000	45,425,175,000
한진칼	2016	반기보고서	458,004,035,239	44,683,311,902
한진칼	2016	사업보고서	991,024,369,234	98,977,829,886
한진칼	2017	반기보고서	551,115,802,084	62,674,523,851
한진칼	2017	사업보고서	1,149,657,255,468	115,286,289,305
NHN	2016	반기보고서	414,240,056,087	19,534,679,912
NHN	2016	사업보고서	856,420,458,207	26,368,038,176
NHN	2017	반기보고서	451,627,267,639	19,141,343,208
NHN	2017	사업보고서	909,115,708,183	34,732,711,952
DSR	2016	반기보고서	100,476,424,690	6,909,243,058
DSR	2016	사업보고서	201,060,062,780	13,355,716,132
DSR	2017	반기보고서	111,556,744,212	8,417,202,280
DSR	2017	사업보고서	226,371,576,785	13,891,641,411

2-4 사업보고서 하반기 실적 보정

```python
def process_adjust_financial_half():
    """
    Raw 재무 데이터를 기반으로 순수 연간(하반기) 실적을 계산함.
    사업보고서 실적 = (사업보고서 누적) - (반기보고서 누적)
    """
    df = pd.read_csv('./data/financial_raw.csv', encoding='utf-8-sig')

    # 1. 숫자형 변환 및 콤마 제거
    for col in ['매출액', '영업이익']:
        df[col] = df[col].astype(str).str.replace(',', '', regex=False)
        df[col] = pd.to_numeric(df[col], errors='coerce')

    # 2. 반기/사업 분리 및 인덱스 설정
    half = df[df['보고서유형'] == '반기보고서'].set_index(['기업명', '연도'])[['매출액', '영업이익']]
    annual = df[df['보고서유형'] == '사업보고서'].set_index(['기업명', '연도'])[['매출액', '영업이익']]

    # 3. 사업보고서 실적 보정 (순수 하반기 실적 계산)
    annual_adj = annual - half
    annual_adj = annual_adj.reset_index()
    annual_adj['보고서유형'] = '사업보고서'

    # 4. 반기보고서는 원본 그대로 유지
    half_raw = half.reset_index()
    half_raw['보고서유형'] = '반기보고서'

    # 5. 합치기 및 저장
    final = pd.concat([half_raw, annual_adj], ignore_index=True)
    final = final.sort_values(['기업명', '연도', '보고서유형'])
    final.to_csv('./data/financial_adjust.csv', encoding='utf-8-sig', index=False)
    print('financial_adjust.csv 저장 완료')
```

1	기업명	연도	매출액	영업이익	보고서유형
8071	종근당	2016	4.07635E+11	18830521450	반기보고서
8072	종근당	2016	4.24351E+11	42419023270	사업보고서
8073	종근당	2017	4.20707E+11	33094684925	반기보고서
8074	종근당	2017	4.63656E+11	44670307200	사업보고서
8075	종근당	2018	4.55878E+11	36880287068	반기보고서
8076	종근당	2018	5.0034E+11	38859607180	사업보고서
8077	종근당	2019	5.0057E+11	34142583337	반기보고서
8078	종근당	2019	5.78768E+11	40433029423	사업보고서
8079	종근당	2020	6.07388E+11	62228594353	반기보고서
8080	종근당	2020	6.95617E+11	61707284497	사업보고서
8081	종근당	2021	6.39449E+11	53518002757	반기보고서
8082	종근당	2021	7.04111E+11	41240691257	사업보고서
8083	종근당	2022	7.07398E+11	51991700563	반기보고서
8084	종근당	2022	7.80947E+11	57914360968	사업보고서
8085	종근당	2023	7.61187E+11	76491803560	반기보고서
8086	종근당	2023	9.08218E+11	1.70107E+11	사업보고서
8087	종근당	2024	7.58305E+11	66672080808	반기보고서
8088	종근당	2024	8.28126E+11	32789465307	사업보고서
8089	종근당바이오	2016	56477308903	7928130754	반기보고서
8090	종근당바이오	2016	56536093239	3687792805	사업보고서
8091	종근당바이오	2017	60630490387	4231603977	반기보고서
8092	종근당바이오	2017	57556581777	4801098666	사업보고서
8093	종근당바이오	2018	61650387331	2448540572	반기보고서
8094	종근당바이오	2018	62859631673	4560443650	사업보고서
8095	종근당바이오	2019	64274160073	6527582610	반기보고서
8096	종근당바이오	2019	72910028479	8884991069	사업보고서
8097	종근당바이오	2020	65132306931	7325240333	반기보고서
8098	종근당바이오	2020	59467652483	281781169	사업보고서
8099	종근당바이오	2021	71015549992	-3608002508	반기보고서
8100	종근당바이오	2021	71222868412	-7824267788	사업보고서
8101	종근당바이오	2022	83536560691	-4265289797	반기보고서
8102	종근당바이오	2022	72505766350	-10545053740	사업보고서
8103	종근당바이오	2023	82676913140	-9472319144	반기보고서
8104	종근당바이오	2023	77675865214	-10680201237	사업보고서
8105	종근당바이오	2024	96450072994	8167725845	반기보고서
8106	종근당바이오	2024	75305935728	2785818640	사업보고서

2-4 재무 데이터 증감율 계산

```python
def process_calculate_financial_rates():
    """
    보정된 재무 데이터를 사용하여 '매출액증감율' 및 '영업이익증감율'을 계산함.
    - 반기보고서 증감율: 전년도 사업보고서 대비
    - 사업보고서 증감율: 같은 연도 반기보고서 대비
    """
    df = pd.read_csv('./data/financial_adjust.csv', encoding='utf-8-sig')

    for col in ['매출액', '영업이익']:
        df[col] = pd.to_numeric(df[col], errors='coerce').fillna(0)

    df = df.sort_values(['기업명', '연도', '보고서유형']).reset_index(drop=True)
    results = []

    for corp, g in df.groupby('기업명'):
        g = g.sort_values(['연도', '보고서유형'])

        for idx, row in g.iterrows():
            year = row['연도']
            report = row['보고서유형']
            sales = row['매출액']
            op = row['영업이익']

            sales_rate, op_rate = None, None

            if report == "반기보고서": # 기준: 전년도 사업보고서
                prev = g[(g['연도'] == year - 1) & (g['보고서유형'] == '사업보고서')]
            else: # 기준: 같은 해 반기보고서
                prev = g[(g['연도'] == year) & (g['보고서유형'] == '반기보고서')]

            if not prev.empty:
                prev_sales = prev.iloc[0]['매출액']
                prev_op = prev.iloc[0]['영업이익']

                # 증감율 계산 ((현재 - 이전) / 이전)
                if prev_sales != 0:
                    sales_rate = (sales - prev_sales) / prev_sales
                if prev_op != 0:
                    op_rate = (op - prev_op) / prev_op

            results.append({
                '기업명': corp, '연도': year, '보고서유형': report,
                '매출액': sales, '영업이익': op,
                '매출액증감율': sales_rate, '영업이익증감율': op_rate
            })

    final = pd.DataFrame(results)
    final.to_csv('./data/financial_final.csv', encoding='utf-8-sig', index=False)
    print("financial_final.csv 저장 완료!")
```

기업명	연도	보고서유형	매출액	영업이익	매출액증감율	영업이익증감율
종근당	2016	사업보고서	4.24351E+11	42419023270	4.100813011	125.2673851
종근당	2017	반기보고서	4.20707E+11	33094684925	-0.858646963	-21.98150176
종근당	2017	사업보고서	4.63656E+11	44670307200	10.20859458	34.97728503
종근당	2018	반기보고서	4.55878E+11	36880287068	-1.67734051	-17.43892223
종근당	2018	사업보고서	5.0034E+11	38859607180	9.753026601	5.366878268
종근당	2019	반기보고서	5.0057E+11	34142583337	0.045833135	-12.13862976
종근당	2019	사업보고서	5.78768E+11	40433029423	15.62183538	18.42404842
종근당	2020	반기보고서	6.07388E+11	62228594353	4.945049871	53.90534729
종근당	2020	사업보고서	6.95617E+11	61707284497	14.52595633	-0.83773362
종근당	2021	반기보고서	6.39449E+11	53518002757	-8.0746129	-13.27117504
종근당	2021	사업보고서	7.04111E+11	41240691257	10.11209598	-22.9405263
종근당	2022	반기보고서	7.07398E+11	51991700563	0.466842738	26.06893575
종근당	2022	사업보고서	7.80947E+11	57914360968	10.39719479	11.39154969
종근당	2023	반기보고서	7.61187E+11	76491803560	-2.530289061	32.07743689
종근당	2023	사업보고서	9.08218E+11	1.70107E+11	19.31597114	122.3865402
종근당	2024	반기보고서	7.58305E+11	66672080808	-16.50627731	-60.80590778
종근당	2024	사업보고서	8.28126E+11	32789465307	9.207582765	-50.81979607
종근당바이오	2016	반기보고서	56477308903	7928130754		
종근당바이오	2016	사업보고서	56536093239	3687792805	0.104084874	-53.48471261
종근당바이오	2017	반기보고서	60630490387	4231603977	7.242094233	14.74625069
종근당바이오	2017	사업보고서	57556581777	4801098666	-5.069905571	13.45812822
종근당바이오	2018	반기보고서	61650387331	2448540572	7.112662753	-49.00041132
종근당바이오	2018	사업보고서	62859631673	4560443650	1.961454574	86.25150435
종근당바이오	2019	반기보고서	64274160073	6527582610	2.250296991	43.13481562
종근당바이오	2019	사업보고서	72910028479	8884991069	13.43598796	36.11457104
종근당바이오	2020	반기보고서	65132306931	7325240333	-10.6675607	-17.55489368
종근당바이오	2020	사업보고서	59467652483	281781169	-8.697150024	-96.15328431
종근당바이오	2021	반기보고서	71015549992	-3608002508	19.41878824	-1380.42712
종근당바이오	2021	사업보고서	71222868412	-7824267788	0.291933837	116.8587126
종근당바이오	2022	반기보고서	83536560691	-4265289797	17.28895866	-45.48640317
종근당바이오	2022	사업보고서	72505766350	-10545053740	-13.20475041	147.2294789
종근당바이오	2023	반기보고서	82676913140	-9472319144	14.02805225	-10.17286988
종근당바이오	2023	사업보고서	77675865214	-10680201237	-6.048904992	12.75170393
종근당바이오	2024	반기보고서	96450072994	8167725845	24.16993712	-176.4753928
종근당바이오	2024	사업보고서	75305935728	2785818640	-21.92236523	-65.89235862

2-5. 최종 데이터 통합 및 정제

2-5 고용-재무 데이터 통합

```python
def finalize_merge_data():
    """
    직원 현황 최종 데이터(emp_final.csv)와 재무 성과 최종 데이터(financial_final.csv)를
    '기업명', '연도', '보고서유형'을 기준으로 Left Join하여 통합 분석 데이터셋을 생성함.
    """
    emp = pd.read_csv('./data/emp_final.csv')
    fin = pd.read_csv('./data/financial_final.csv')

    # 재무 변수 숫자 변환 재확인
    for col in ['매출액', '영업이익']:
        if col in fin.columns:
            fin[col] = fin[col].astype(str).str.replace(',', '', regex=False)
            fin[col] = pd.to_numeric(fin[col], errors='coerce').fillna(0)

    # 병합 수행
    merged = pd.merge(
        emp,
        fin[['기업명', '연도', '보고서유형', '매출액', '매출액증감율', '영업이익', '영업이익증감율']],
        on=['기업명', '연도', '보고서유형'],
        how='left'
    )

    # 결측치 0으로 처리 (재무 지표)
    merged['매출액'] = merged['매출액'].fillna(0)
    merged['영업이익'] = merged['영업이익'].fillna(0)

    merged.to_csv('./data/emp_financial_merged.csv', encoding='utf-8-sig', index=False)
    print("emp_financial_merged.csv 저장 완료!")
```

2-5 고용-재무 데이터 통합

기업명	연도	보고서유형	평균급여	남성직원	여성직원	정규직수	계약직수	남성정규	남성계약	여성정규	여성계약	평균급여	남성직원	여성직원	정규직수	계약직수	남성정규	남성계약	여성정규	여성계약	매출액	매출액증	영업이익	영업이익
종근당	2016	반기보고서	29860314	1352	495	1766	81	1317	35	449	46										4E+11		2E+10	
종근당	2016	사업보고서	30250664	1373	510	1791	92	1329	44	462	48	1.3073	1.5533	3.0303	1.4156	13.58	0.9112	25.714	2.8953	4.3478	4E+11	4.1008	4E+10	125.27
종근당	2017	반기보고서	29856921	1398	538	1830	106	1352	46	478	60	-1.3016	1.8208	5.4902	2.1776	15.217	1.7306	4.5455	3.4632	25	4E+11	-0.8586	3E+10	-21.982
종근당	2017	사업보고서	28856929	1467	539	1995	11	1466	1	529	10	-3.3493	4.9356	0.1859	9.0164	-89.623	8.432	-97.826	10.669	-83.333	5E+11	10.209	4E+10	34.977
종근당	2018	반기보고서	29235863	1499	570	2055	14	1498	1	557	13	1.3131	2.1813	5.7514	3.0075	27.273	2.1828	0	5.293	30	5E+11	-1.6773	4E+10	-17.439
종근당	2018	사업보고서	31028571	1492	608	2085	15	1490	2	595	13	6.1319	-0.467	6.6667	1.4599	7.1429	-0.534	100	6.8223	0	5E+11	9.753	4E+10	5.3669
종근당	2019	반기보고서	32526128	1584	655	2215	24	1584	0	631	24	4.8264	6.1662	7.7303	6.235	60	6.3087	-100	6.0504	84.615	5E+11	0.0458	3E+10	-12.139
종근당	2019	사업보고서	37527864	1581	662	2222	21	1574	7	648	14	15.378	-0.1894	1.0687	0.316	-12.5	-0.6313		2.6941	-41.667	6E+11	15.622	4E+10	18.424
종근당	2020	반기보고서	33597736	1619	678	2259	38	1605	14	654	24	-10.473	2.4035	2.4169	1.6652	80.952	1.9695	100	0.9259	71.429	6E+11	4.945	6E+10	53.905
종근당	2020	사업보고서	35263436	1593	677	2263	7	1590	3	673	4	4.9578	-1.6059	-0.1475	0.1771	-81.579	-0.9346	-78.571	2.9052	-83.333	7E+11	14.526	6E+10	-0.8377
종근당	2021	반기보고서	35397928	1619	698	2287	30	1611	8	676	22	0.3814	1.6321	3.1019	1.0605	328.57	1.3208	166.67	0.4458	450	6E+11	-8.0746	5E+10	-13.271
종근당	2021	사업보고서	35654351	1688	748	2380	56	1660	28	720	28	0.7244	4.2619	7.1633	4.0665	86.667	3.0416	250	6.5089	27.273	7E+11	10.112	4E+10	-22.941
종근당	2022	반기보고서	36990087	1659	762	2378	43	1644	15	734	28	3.7463	-1.718	1.8717	-0.084	-23.214	-0.9639	-46.429	1.9444	0	7E+11	0.4668	5E+10	26.069
종근당	2022	사업보고서	37964942	1639	757	2361	35	1625	14	736	21	2.6354	-1.2055	-0.6562	-0.7149	-18.605	-1.1557	-6.6667	0.2725	-25	8E+11	10.397	6E+10	11.392
종근당	2023	반기보고서	38521720	1595	730	2291	34	1582	13	709	21	1.4666	-2.6846	-3.5667	-2.9648	-2.8571	-2.6462	-7.1429	-3.6685	0	8E+11	-2.5303	8E+10	32.077
종근당	2023	사업보고서	39444062	1603	721	2282	42	1585	18	697	24	2.3943	0.5016	-1.2329	-0.3928	23.529	0.1896	38.462	-1.6925	14.286	9E+11	19.316	2E+11	122.39
종근당	2024	반기보고서	41496381	1619	730	2286	63	1590	29	696	34	5.2031	0.9981	1.2483	0.1753	50	0.3155	61.111	-0.1435	41.667	8E+11	-16.506	7E+10	-60.806
종근당	2024	사업보고서	40041952	1606	730	2263	73	1572	34	691	39	-3.505	-0.803	0	-1.0061	15.873	-1.1321	17.241	-0.7184	14.706	8E+11	9.2076	3E+10	-50.82
종근당바이오	2016	반기보고서	31173796	252	32	276	8	246	6	30	2										6E+10		8E+09	
종근당바이오	2016	사업보고서	31358807	257	33	282	8	250	7	32	1	0.5935	1.9841	3.125	2.1739	0	1.626	16.667	6.6667	-50	6E+10	0.1041	4E+09	-53.485
종근당바이오	2017	반기보고서	30448020	267	36	291	12	258	9	33	3	-2.9044	3.8911	9.0909	3.1915	50	3.2	28.571	3.125	200	6E+10	7.2421	4E+09	14.746
종근당바이오	2017	사업보고서	33292472	272	35	299	8	266	6	33	2	9.342	1.8727	-2.7778	2.7491	-33.333	3.1008	-33.333	0	-33.333	6E+10	-5.0699	5E+09	13.458
종근당바이오	2018	반기보고서	30409669	293	39	332	0	293	0	39	0	-8.659	7.7206	11.429	11.037	-100	10.15	-100	18.182	-100	6E+10	7.1127	2E+09	-49
종근당바이오	2018	사업보고서	32045103	304	45	349	0	304	0	45	0	5.378	3.7543	15.385	5.1205		3.7543		15.385		6E+10	1.9615	5E+09	86.252
종근당바이오	2019	반기보고서	30043039	337	48	385	0	337	0	48	0	-6.2476	10.855	6.6667	10.315		10.855		6.6667		6E+10	2.2503	7E+09	43.135
종근당바이오	2019	사업보고서	29380644	367	54	421	0	367	0	54	0	-2.2048	8.9021	12.5	9.3506		8.9021		12.5		7E+10	13.436	9E+09	36.115
종근당바이오	2020	반기보고서	28933573	414	68	482	0	414	0	68	0	-1.5217	12.807	25.926	14.489		12.807		25.926		7E+10	-10.668	7E+09	-17.555
종근당바이오	2020	사업보고서	29558448	433	72	505	0	433	0	72	0	2.1597	4.5894	5.8824	4.7718		4.5894		5.8824		6E+10	-8.6972	3E+08	-96.153
종근당바이오	2021	반기보고서	29956055	451	80	531	0	451	0	80	0	1.3452	4.157	11.111	5.1485		4.157		11.111		7E+10	19.419	-4E+09	-1380.4
종근당바이오	2021	사업보고서	31366492	450	81	531	0	450	0	81	0	4.7084	-0.2217	1.25	0		-0.2217		1.25		7E+10	0.2919	-8E+09	116.86
종근당바이오	2022	반기보고서	30617791	468	92	560	0	468	0	92	0	-2.3869	4	13.58	5.4614		4		13.58		8E+10	17.289	-4E+09	-45.486
종근당바이오	2022	사업보고서	31831835	458	100	558	0	458	0	100	0	3.9652	-2.1368	8.6957	-0.3571		-2.1368		8.6957		7E+10	-13.205	-1E+10	147.23
종근당바이오	2023	반기보고서	31481278	470	106	576	0	470	0	106	0	-1.1013	2.6201	6	3.2258		2.6201		6		8E+10	14.028	-9E+09	-10.173
종근당바이오	2023	사업보고서	35954308	410	90	500	0	410	0	90	0	14.209	-12.766	-15.094	-13.194		-12.766		-15.094		8E+10	-6.0489	-1E+10	12.752
종근당바이오	2024	반기보고서	34240714	400	93	493	0	400	0	93	0	-4.766	-2.439	3.3333	-1.4		-2.439		3.3333		1E+11	24.17	8E+09	-176.48
종근당바이오	2024	사업보고서	33260743	411	98	507	2	409	2	98	0	-2.862	2.75	5.3763	2.8398		2.25		5.3763		8E+10	-21.922	3E+09	-65.892

2-5 최종 데이터 정제 및 저장

```python
def finalize_clean_and_save():
    """
    통합 데이터셋에서 분석의 신뢰도를 저해하는 이상치 기업(매출액/영업이익이 0인 기록이 존재하는 기업)을 제거함.
    최종 분석 데이터셋인 'final_emp_financial.csv'를 생성함.
    """
    df = pd.read_csv('./data/emp_financial_merged.csv', encoding='utf-8-sig')

    # 숫자형 변환 (재확인)
    for col in ['매출액', '영업이익']:
        df[col] = pd.to_numeric(df[col], errors='coerce').fillna(0)

    # 2016 반기보고서 제거 (전년도 데이터가 없어 증감율 계산 불가)
    df = df[~((df['연도'] == 2016) & (df['보고서유형'] == '반기보고서'))]

    # 매출액 및 영업이익이 0인 기록이 있는 기업 필터링
    zero_problem = (
        df[(df['연도'] >= 2017) & (df['연도'] <= 2024)]
        .groupby('기업명')
        .apply(lambda x: ((x['매출액'] == 0) & (x['영업이익'] == 0)).any())
    )

    bad_corps = zero_problem[zero_problem].index.tolist()
    print("매출·영업이익 모두 0인 해가 존재하는 기업 수:", len(bad_corps))

    # 문제 기업 전체 데이터 제거
    df_clean = df[~df['기업명'].isin(bad_corps)]

    # 최종 저장 (모든 NaN을 0으로 처리)
    df_clean = df_clean.fillna(0)
    df_clean.to_csv('./data/final_emp_financial.csv', encoding='utf-8-sig', index=False)
    print("final_emp_financial.csv 저장 완료!")
```

2-5 고용-재무 데이터 통합

기업명	연도	보고서유형	평균급여	남성직원수	여성직원수	정규직수	계약직수	남성정규직수	남성계약직수	여성정규직수	여성계약직수	평균급여증감	남성직원수증	여성직원수증	정규직수증감	계약직수증감	남성정규직수	남성계약직수	여성정규직수	여성계약직수	매출액	매출액증감	영업이익	영업이익증
AK홀딩스	2016	사업보고서	43785714	12	2	14	0	12	0	2	0	-2.85261	0	0	0	0	0	0	0	0	1.54E+12	12.00374	1.17E+11	21.8661
AK홀딩스	2017	반기보고서	42250000	14	2	15	1	13	1	2	0	-3.50734	16.66667	0	7.142857	0	8.333333	0	0	0	1.62E+12	4.703382	1.21E+11	2.983435
AK홀딩스	2017	사업보고서	43733333	13	2	14	1	12	1	2	0	3.510848	-7.14286	0	-6.66667	0	-7.69231	0	0	0	1.78E+12	9.898451	1.45E+11	19.78257
AK홀딩스	2018	반기보고서	53833333	13	5	18	0	13	0	5	0	23.09451	0	150	28.57143	-100	8.333333	-100	150	0	1.81E+12	1.913237	1.54E+11	6.102802
AK홀딩스	2018	사업보고서	56277778	15	3	17	1	14	1	3	0	4.540764	15.38462	-40	-5.55556	0	7.692308	0	-40	0	1.9E+12	5.014889	1.16E+11	-24.4152
AK홀딩스	2019	반기보고서	43571429	12	2	13	1	11	1	2	0	-22.5779	-20	-33.3333	-23.5294	0	-21.4286	0	-33.3333	0	1.89E+12	-0.70549	1.07E+11	-7.44312
AK홀딩스	2019	사업보고서	34600000	16	4	19	1	15	1	4	0	-20.5902	33.33333	100	46.15385	0	36.36364	0	100	0	1.87E+12	-0.82635	2.39E+10	-77.7861
AK홀딩스	2020	반기보고서	42523810	15	6	20	1	14	1	6	0	22.90118	-6.25	50	5.263158	0	-6.66667	0	50	0	1.33E+12	-28.8717	-1E+11	-535.468
AK홀딩스	2020	사업보고서	47666667	16	5	20	1	15	1	5	0	12.09406	6.666667	-16.6667	0	0	7.142857	0	-16.6667	0	1.29E+12	-3.23142	-1.2E+11	13.125
AK홀딩스	2021	반기보고서	55947368	15	4	18	1	14	1	4	0	17.3721	-6.25	-20	-10	0	-6.66667	0	-20	0	1.52E+12	18.14336	-5.7E+10	-51.6539
AK홀딩스	2021	사업보고서	50842105	16	3	18	1	15	1	3	0	-9.12512	6.666667	-25	0	0	7.142857	0	-25	0	1.65E+12	8.093512	-1.1E+11	91.47071
AK홀딩스	2022	반기보고서	39666667	17	4	20	1	16	1	4	0	-21.9807	6.25	33.33333	11.11111	0	6.666667	0	33.33333	0	1.8E+12	9.376397	-3.3E+10	-70.0825
AK홀딩스	2022	사업보고서	40739130	19	4	22	1	18	1	4	0	2.70369	11.76471	0	10	0	12.5	0	0	0	1.99E+12	10.47314	1.07E+10	-132.995
AK홀딩스	2023	반기보고서	48681818	19	3	21	1	18	1	3	0	19.49646	0	-25	-4.54545	0	0	0	-25	0	2.18E+12	9.81911	1.6E+11	1391.517
AK홀딩스	2023	사업보고서	50217391	20	3	22	1	19	1	3	0	3.154305	5.263158	0	4.761905	0	5.555556	0	0	0	2.3E+12	5.164289	1.19E+11	-25.8473
AK홀딩스	2024	반기보고서	56347826	20	3	23	0	20	0	3	0	12.20779	0	0	4.545455	-100	5.263158	-100	0	0	2.33E+12	1.288198	1.19E+11	0.464568
AK홀딩스	2024	사업보고서	37047619	18	3	21	0	18	0	3	0	-34.2519	-10	0	-8.69565	0	-10	0	0	0	2.16E+12	-7.02122	1.09E+10	-90.8548
BYC	2016	사업보고서	14989131	382	372	665	89	313	69	352	20	5.859266	-2.05128	-6.76692	-5.27066	2.298851	-3.09598	2.985075	-7.12401	0	1.01E+11	-8.60281	6.52E+09	-32.2931
BYC	2017	반기보고서	14957249	376	379	670	85	308	68	362	17	-0.2127	-1.57068	1.88172	0.75188	-4.49438	-1.59744	-1.44928	2.840909	-15	9.12E+10	-9.85553	9.95E+09	52.57794
BYC	2017	사업보고서	17297975	369	379	667	81	305	64	362	17	15.64944	-1.8617	0	-0.44776	-4.70588	-0.97403	-5.88235	0	0	1.05E+11	14.71742	7.26E+09	-27.0729
BYC	2018	반기보고서	16469661	319	353	616	56	277	42	339	14	-4.7885	-13.5501	-6.86016	-7.64618	-30.8642	-9.18033	-34.375	-6.35359	-17.6471	9.88E+10	-5.56207	9E+09	23.98469
BYC	2018	사업보고서	17395453	305	357	608	54	265	40	343	14	5.621199	-4.38871	1.133144	-1.2987	-3.57143	-4.33213	-4.7619	1.179941	0	9.91E+10	0.267589	1.23E+10	37.25144
BYC	2019	반기보고서	16351578	298	340	591	47	264	34	327	13	-6.00085	-2.29508	-4.7619	-2.79605	-12.963	-0.37736	-15	-4.66472	-7.14286	7.96E+10	-19.6128	1.13E+10	-8.43401
BYC	2019	사업보고서	17219737	312	350	610	52	276	36	334	16	5.309327	4.697987	2.941176	3.21489	10.6383	4.545455	5.882353	2.140673	23.07692	9.1E+10	14.21374	1.21E+10	6.882249
BYC	2020	반기보고서	16716250	309	336	604	41	281	28	323	13	-2.9239	-0.96154	-4	-0.98361	-21.1538	1.811594	-22.2222	-3.29341	-18.75	8.04E+10	-11.5796	1.07E+10	-11.1386
BYC	2020	사업보고서	18296383	273	317	557	33	250	23	307	10	9.452679	-11.6505	-5.65476	-7.78146	-19.5122	-11.032	-17.8571	-4.95356	-23.0769	8.14E+10	1.2677	1.22E+10	13.60768
BYC	2021	반기보고서	17502412	274	297	544	27	252	22	292	5	-4.3395	0.3663	-6.30915	-2.33393	-18.1818	0.8	-4.34783	-4.88599	-50	7.48E+10	-8.13711	9.3E+09	-23.7844
BYC	2021	사업보고서	18543657	270	296	537	29	246	24	291	5	5.949156	-1.45985	-0.3367	-1.28676	7.407407	-2.38095	9.090909	-0.34247	0	8.96E+10	19.80496	1.74E+10	87.24333
BYC	2022	반기보고서	18090288	267	302	537	32	241	26	296	6	-2.44487	-1.11111	2.027027	0	10.34483	-2.03252	8.333333	1.718213	20	7.91E+10	-11.7246	1.02E+10	-41.5098
BYC	2022	사업보고서	19266274	264	306	540	30	239	25	301	5	6.500645	-1.1236	1.324503	0.558659	-6.25	-0.82988	-3.84615	1.689189	-16.6667	9.06E+10	14.4527	1.51E+10	48.67795
BYC	2023	반기보고서	19117241	257	304	530	31	230	27	300	4	-0.77354	-2.65152	-0.65359	-1.85185	3.333333	-3.76569	8	-0.33223	-20	7.81E+10	-13.7757	8.87E+09	-41.3952
BYC	2023	사업보고서	20112614	253	304	531	26	230	23	301	3	5.206679	-1.55642	0	0.188679	-16.129	0	-14.8148	0.333333	-25	9.03E+10	15.67025	1.86E+10	109.879
BYC	2024	반기보고서	19475918	244	305	521	28	219	25	302	3	-3.16565	-3.55731	0.328947	-1.88324	7.692308	-4.78261	8.695652	0.332226	0	7.99E+10	-11.5605	7.34E+09	-60.6057
BYC	2024	사업보고서	18562754	238	318	525	31	210	28	315	3	-4.68868	-2.45902	4.262295	0.767754	10.71429	-4.10959	12	4.304636	0	8.53E+10	6.826532	1.65E+10	124.8329
CJ	2016	사업보고서	34558824	21	13	34	0	21	0	13	0	-4.54915	0	0	0	0	0	0	0	0	1.24E+13	7.102723	5.72E+11	-16.0459
CJ	2017	반기보고서	37483871	18	13	31	0	18	0	13	0	8.463967	-14.2857	0	-8.82353	0	-14.2857	0	0	0	1.29E+13	3.768258	6.29E+11	10.06357
CJ	2017	사업보고서	44071429	15	13	28	0	15	0	13	0	17.57438	-16.6667	0	-9.67742	0	-16.6667	0	0	0	1.4E+13	9.251617	6.97E+11	10.68413
CJ	2018	반기보고서	38176471	20	14	34	0	20	0	14	0	-13.3759	33.33333	7.692308	21.42857	0	33.33333	0	7.692308	0	1.4E+13	-0.0515	6.43E+11	-7.71887
CJ	2018	사업보고서	3.63E+08	38	20	57	1	37	1	20	0	849.9442	90	42.85714	67.64706	0	85	0	42.85714	0	1.55E+13	10.33032	6.9E+11	7.283061
CJ	2019	반기보고서	1.25E+08	45	19	62	2	43	2	19	0	-65.5665	18.42105	-5	8.77193	100	16.21622	100	-5	0	1.62E+13	4.919759	7.08E+11	2.731471
CJ	2019	사업보고서	2.49E+08	45	20	63	2	43	2	20	0	99.42404	0	5.263158	1.612903	0	0	0	5.263158	0	1.75E+13	7.892441	8.01E+11	13.01183
CJ	2020	반기보고서	1.41E+08	41	15	50	6	36	5	14	1	-43.4665	-8.88889	-25	-20.6349	200	-16.2791	150	-30	0	1.56E+13	-10.8211	6.23E+11	-22.1883
CJ	2020	사업보고서	3.45E+08	38	15	47	6	33	5	14	1	145.2811	-7.31707	0	-6	0	-8.33333	0	0	0	1.64E+13	4.676285	7.67E+11	23.16315
CJ	2021	반기보고서	1.27E+08	44	16	54	6	39	5	15	1	-63.2323	15.78947	6.666667	14.89362	0	18.18182	0	7.142857	0	1.63E+13	-0.10506	1.01E+12	32.2731
CJ	2021	사업보고서	4.01E+08	43	16	53	6	38	5	15	1	216.1914	-2.27273	0	-1.85185	0	-2.5641	0	0	0	1.81E+13	10.93852	8.67E+11	-14.5945
CJ	2022	반기보고서	1.45E+08	46	17	55	8	39	7	16	1	-63.933	6.976744	6.25	3.773585	33.33333	2.631579	40	6.666667	0	1.94E+13	7.120237	1.19E+12	36.8032
CJ	2022	사업보고서	4.31E+08	51	16	59	8	44	7	15	1	197.5722	10.86957	-5.88235	7.272727	0	12.82051	0	-6.25	0	2.15E+13	10.65484	9.68E+11	-18.3425

2-6 데이터 수집 결과

2-6 데이터 수집 결과

- 2016년 이전의 재무 데이터는 양식과 산정방식이 지금과 다르기에 사용할 수 없었음.
- 일부 증권사, 금융기업은 직원 급여를 제공하지 않아 사용할 수 없었음.
- 상장폐지되거나 합병된 기업의 경우 제외함.

- 따라서 **총 594개 코스피 상장 기업**의 2016~2024 반기보고서, 사업보고서 데이터를 수집하였고 증감율 계산을 위해 2016년 반기보고서를 제외한 **2017년 사업보고서 ~ 2024년 반기보고서** 까지의 데이터를 사용하기로 최종 결정함.

3-1 고용구조와 재무성과의 상관관계 히트맵

3-1 고용구조와 재무성과의 상관관계 히트맵

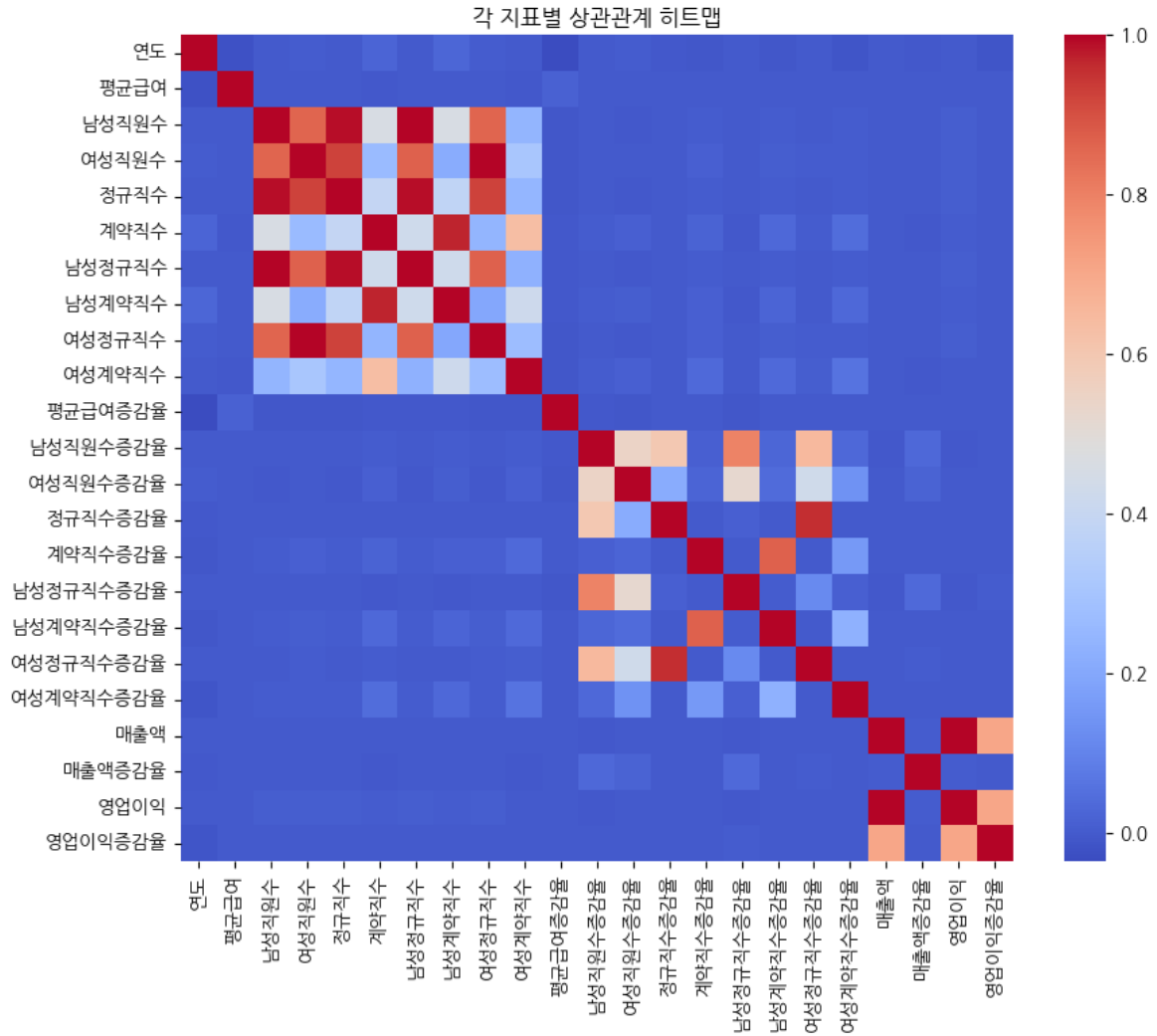
- **1. 데이터 로드**
- **2. 보고서유형(반기/사업)의 모델 활용을 위한 수치 변환**

  반기보고서 -> 0 / 사업보고서 -> 1
- **3. 숫자형 변수들만 추출하고 corr() 메소드를 이용하여 상관관계 히트맵 출력**

```
drive.mount('/content/gdrive/')

file_path = '/content/gdrive/MyDrive/빅데이터/과제/final_emp_financial.csv'

df_original = pd.read_csv(file_path)

print(df_original.info())

numeric_cols = df_original.select_dtypes(include=['int64', 'float64']).columns
df_original[numeric_cols].describe()

df_original['보고서유형코드'] = df_original['보고서유형'].map({'반기보고서': 0, '사업보고서': 1})
df_original = df_original.sort_values(["기업명", "연도", "보고서유형코드"]).reset_index(drop=True)
```

```
plt.figure(figsize=(14,10))
sns.heatmap(df_original[numeric_cols].corr(), annot=False, cmap="coolwarm")
plt.title("각 지표별 상관관계 히트맵")
plt.show()
```

3-1 고용구조와 재무성과의 상관관계 히트맵



3-1 고용구조와 재무성과의 상관관계 히트맵

- 직원 수 관련 지표들은 서로 강한 양의 상관관계를 가짐.
  이는 기업 규모가 직원 전반에 균등하게 영향을 미친다는 것을 의미함.
- 직원 수 증감률 지표들 역시 강한 그룹을 형성함.
- 특정 인력군을 확충하는 기업은 다른 인력군도 함께 증가시키는 경향이 있음.
- 반면 고용구조 지표와 재무성과의 상관관계는 낮게 나타남.
  이는 인력 구성 자체가 매출이나 이익과 직접적인 선형 관계를 가지지 않음을 시사함.

3-2 고용구조증감율과 매출증감율의 관계 분석 및 회귀 모델링

3-2 고용구조증감율과 매출증감율의 관계 분석 및 회귀 모델링

1. 주제 선정 이유

고용구조 변화가 매출액 증감율에 미치는 영향을 분석하고자 함.

2. 데이터 및 변수 설명

독립 변수 : 평균급여증감율, 남성직원수증감율, 여성직원수증감율, 정규직수증감율, 계약직수증감율
종속 변수 : 매출액 증감율
전처리 : 결측치 행 제거, Train/Test = 80:20

3. 분석 방법

해당 주제에서는 Random Forest Regressor를 사용함.
모델은 300개의 트리를 사용해 학습하였으며, 랜덤 시드를 고정하여 재현성을 확보함.
모델 학습 후에는MSE, RMSE, R^2 점수를 산출하여 예측 정확도를 평가하였고,
Feature Importance를 통해 어떤 변수의 영향력이 큰지 분석함.
또한 실제값과 예측값의 관계를 시각화하여 모델의 예측 패턴을 확인함.

3-2 고용구조증감율과 매출증감율의 관계 분석 및 회귀 모델링

4. 모델 성능 결과 정리

MSE = 402,077.61 / RMSE = 634.09 / **R^2 = -0.0037**

R^2이 음수로 나타났다는 점은 고용구조 증감율만으로 매출액 증감율을 효과적으로 설명하기 어렵다는 것을 의미함.

즉, 매출액 증감의 주요 요인은 고용구조보다 외부 요인에 의해 훨씬 더 크게 좌우될 가능성이 높음.

5. 데이터 및 변수 설명

변수 중요도 분석 결과, 매출액 증감율에 영향을 미치는 변수 순위는 다음과 같음.

정규직수증감율 > 평균급여증감율 > 남성직원수증감율 > 계약직수증감율 > 여성직원수증감율

특히 정규직 증가율이 가장 큰 영향 요인으로 나타났는데, 이는 기업의 인력 충원 전략이 매출 성장에 기여할 수 있음을 시사함.

3-2 고용구조증감율과 매출증감율의 관계 분석 및 회귀 모델링

6. 실제값 vs 예측값 시각화 해석 (로그스케일 산점도)

실제값과 예측값을 비교한 산점도는 대부분의 점이 중앙에 밀집되어 나타나는 형태를 보였음.

이는 모델이 고용구조 변화율로부터 매출액 변화율을 충분히 설명하지 못하고 있음을 의미함.

7. 결론

분석 결과, 고용구조 변화율만으로는 매출액 증감율을 설명하기 어렵다는 한계가 나타남.

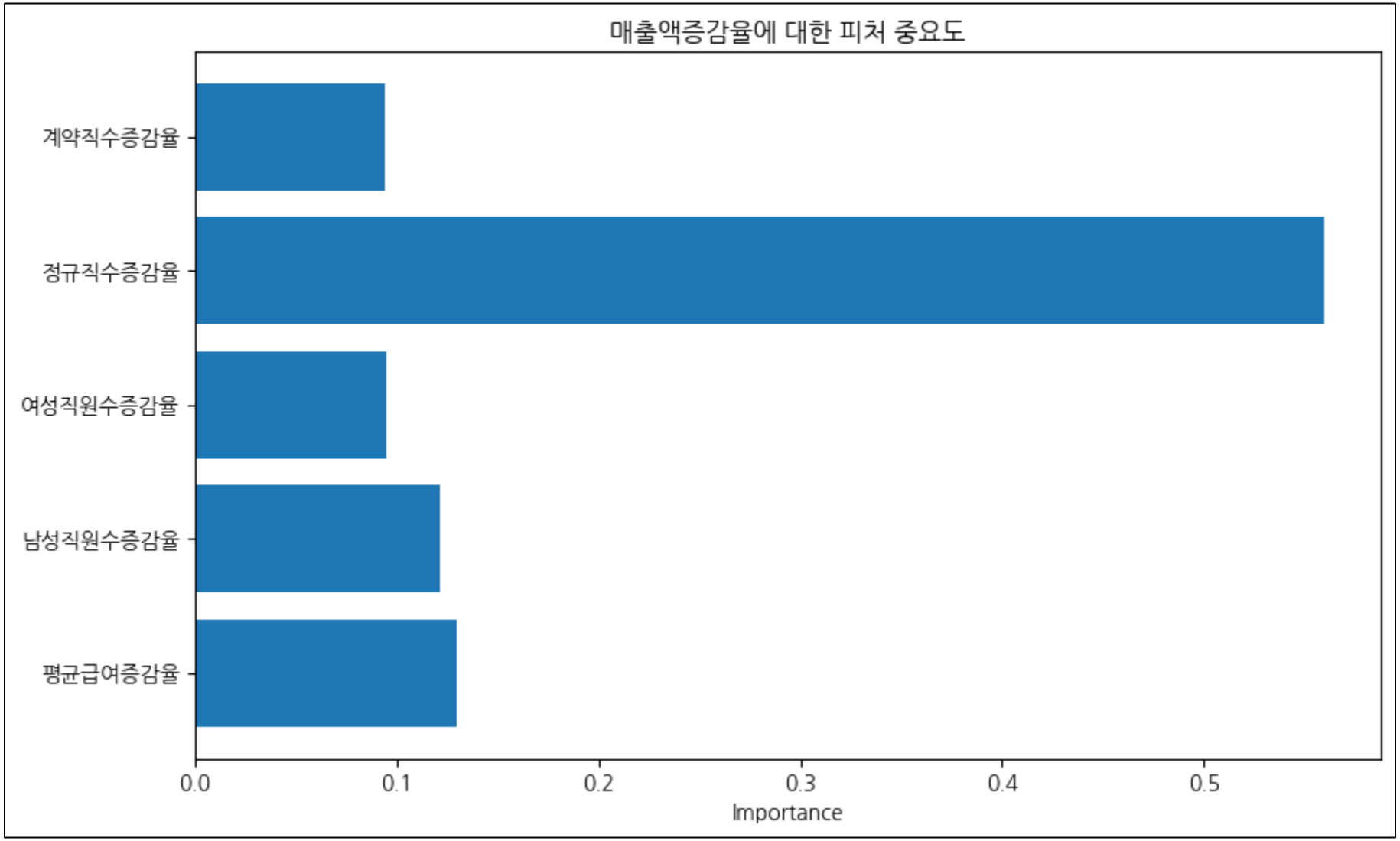
다만 정규직수 증감율 등 일부 변수는 일정한 영향력을 보였으므로, 고용 전략이 기업 실적에 영향을 줄 가능성은 존재함.

3-2 고용구조증감율과 매출증감율의 관계 분석 및 회귀 모델링

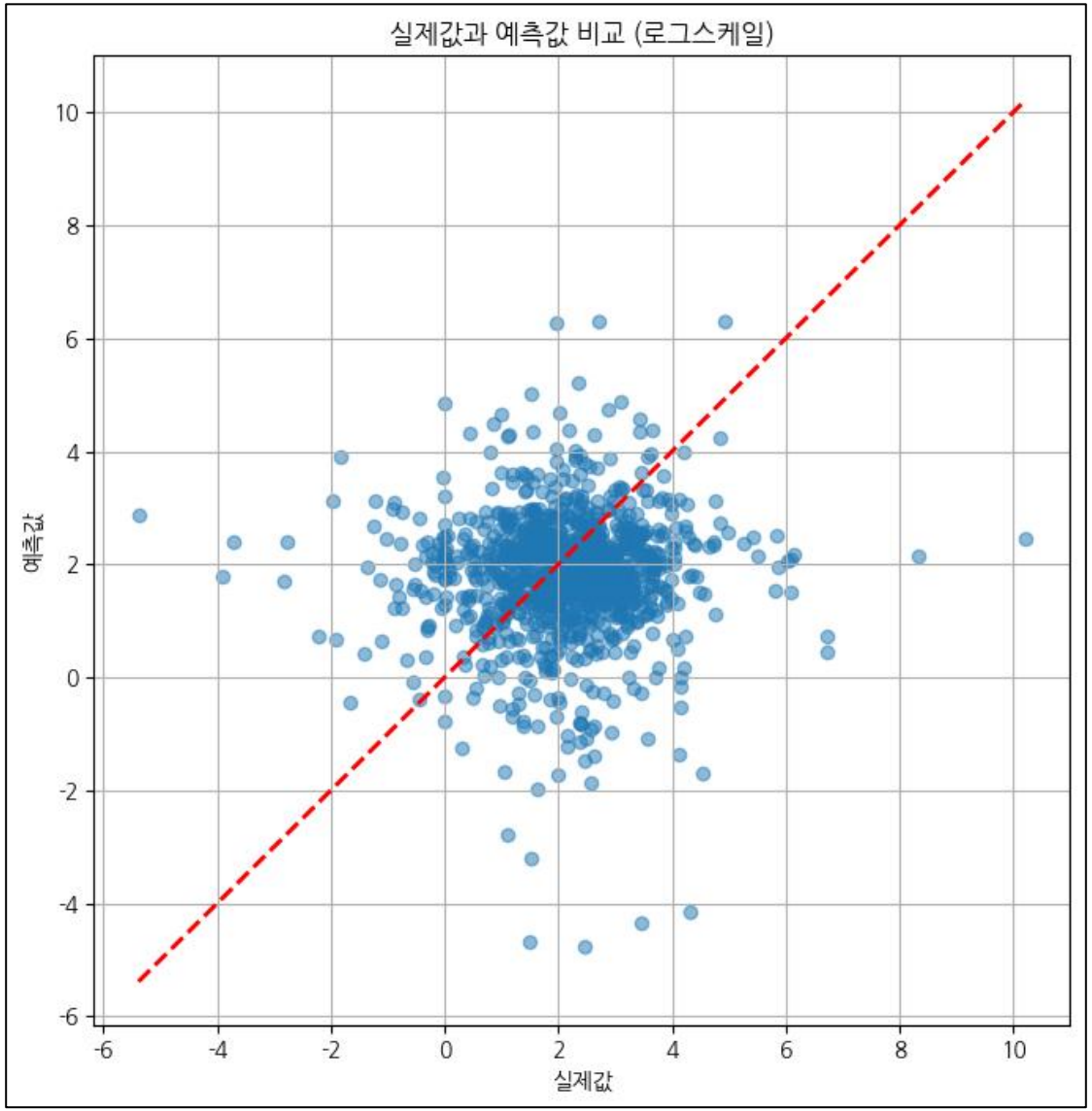
```
# 고용구조와 매출액증감율의 관계 분석 및 회귀 모델링

features = [
    '평균급여증감율',
    '남성직원수증감율', '여성직원수증감율',
    '정규직수증감율', '계약직수증감율',
]

target = '매출액증감율'

df = df_original

# 결측치 제거
data = df[features + [target]].dropna()

# 학습 데이터셋 분할
from sklearn.model_selection import train_test_split

X = data[features]
y = data[target]

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, shuffle=True, random_state=42
)

# 랜덤 포레스트 회귀
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error, r2_score

model = RandomForestRegressor(n_estimators=300, random_state=42)
model.fit(X_train, y_train)

pred = model.predict(X_test)
```

```
# 모델 성능 평가
mse = mean_squared_error(y_test, pred)
r2 = r2_score(y_test, pred)

print("MSE:", mse)
print("RMSE:", mse**0.5)
print("R² :", r2)

# importance 시각화
import numpy as np

importance = model.feature_importances_

plt.figure(figsize=(10,6))
plt.barh(features, importance)
plt.xlabel("Importance")
plt.title("매출액증감율에 대한 피처 중요도")
plt.show()

# 실제값 vs 예측값 시각화 (로그스케일)
plt.figure(figsize=(8, 8))
plt.scatter(np.log1p(y_test), np.log1p(pred), alpha=0.5)
plt.plot([np.log1p(y_test).min(), np.log1p(y_test).max()],
         [np.log1p(y_test).min(), np.log1p(y_test).max()],
         'r--', linewidth=2)

plt.xlabel("실제값")
plt.ylabel("예측값")
plt.title("실제값과 예측값 비교 (로그스케일)")
plt.grid(True)
plt.show()
```

3-2 고용구조증감율과 매출증감율의 관계 분석 및 회귀 모델링



3-2 고용구조증감율과 매출증감율의 관계 분석 및 회귀 모델링



3-3 고용구조와 영업이익의 관계 분석 및 회귀 모델링

3-3 고용구조와 영업이익의 관계 분석 및 회귀 모델링

1. 주제 선정 이유

고용구조가 영업이익에 미치는 영향을 분석하고자 함.

2. 데이터 및 변수 설명

독립 변수 : 평균급여, 남성직원수, 여성직원수, 정규직수, 계약직수, 남성정규직수, 여성정규직수, 남성계약직수, 여성계약직수

종속 변수 : 영업이익

전처리 : 영업이익·평균급여에서 숫자만 추출, 0 이하 값 제거 후 로그변환, 결측치 행 제거, Train/Test = 80:20, 종속변수는 log(1+y) 변환

3. 분석 방법

XGBoost 회귀 모델을 사용하여 영업이익을 예측함.

모델은 최적화를 통해 트리 400개, max_depth=6, learning_rate=0.05 등 파라미터를 사용함.

Subsample, colsample을 통해 과적합을 방지함.

학습 후 MSE, RMSE, R^2를 계산하여 모델의 예측력을 평가함.

Feature Importance 시각화를 통해 어떤 변수가 영업이익에 더 큰 영향을 미쳤는지 파악함.

또한 실제값과 예측값을 비교하여 예측 패턴을 시각적으로 확인함.

3-3 고용구조와 영업이익의 관계 분석 및 회귀 모델링

4. 모델 성능 결과 정리

MSE : 3.221e+30 / RMSE : 1794733011232615 / R^2 : -0.0005165

R^2가 0 이하이며, 오차가 매우 크게 나타나는 것을 볼 때 현재 사용한 고용구조 절대 지표만으로는 영업이익을 설명할 수 없으며, XGBoost와 같은 고성능 모델에서도 의미 있는 예측력을 확보하지 못했음을 의미함.

이는 영업이익이 고용구조 외부의 요인에 크게 좌우된다는 점을 시사함.

5. 데이터 및 변수 설명

변수 중요도 분석 결과, 매출액 증감율에 영향을 미치는 변수 순위는 다음과 같음.

[**정규직수 > 남성정규직수 > 남성직원수 > 평균급여 > 여성정규직수 > 여성계약직수 > 남성계약직수 > 여성직원수 > 계약직수]**

정규직 수 관련 변수가 높은 중요도를 보였는데, 이는 고용의 안정성 또는 기업 규모가 영업이익과 연관이 있을 수 있음을 시사함.

하지만 전체 모델 성능이 낮기 때문에 단일 변수의 영향력도 제한적으로 해석해야 함.

3-3 고용구조와 영업이익의 관계 분석 및 회귀 모델링

6. 실제값 vs 예측값 시각화 해석 (로그스케일 산점도)

실제값과 예측값의 비교 그래프에서 점들이 대각선에서 크게 벗어나 있으며,

전체적으로 넓게 퍼져 있어 모델이 실제 영업이익 규모를 제대로 설명하지 못하고 있음.

특히 고이익 구간에서 예측 오차가 매우 크게 나타나 XGBoost 모델이 패턴을 포착하지 못함을 알 수 있음.

7. 결론

이번 분석에서는 고용구조를 기반으로 XGBoost 모델을 학습하였으나, 영업이익을 예측하기에는 한계가 매우 큰 것으로 나타남.

이는 영업이익이 고용지표보다 훨씬 복잡하고 다양한 외부 요인에 의해 결정되기 때문인 것으로 판단되며,

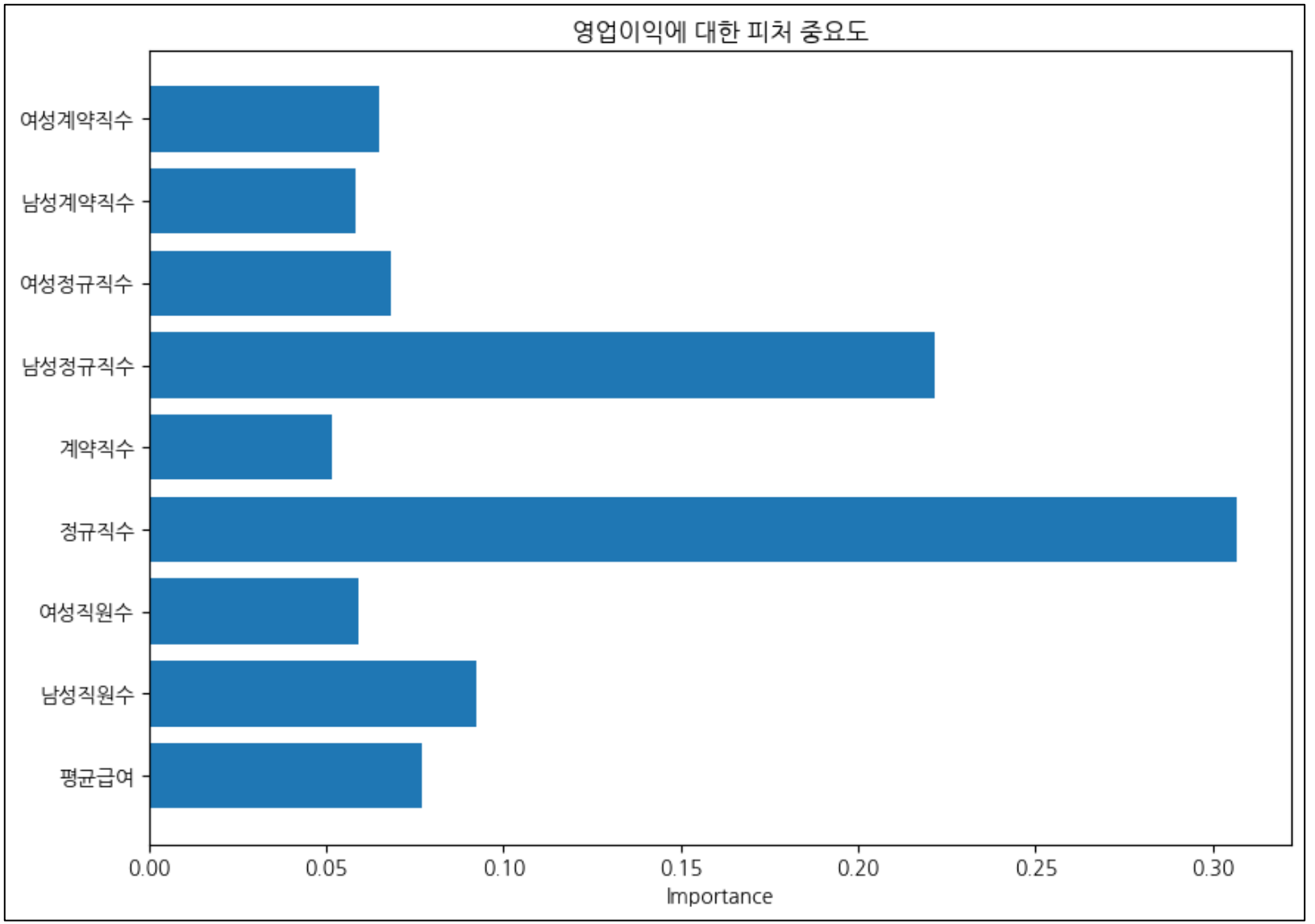
향후 추가적인 재무 및 사업 지표가 포함되어야 영업이익과의 실질적인 관계를 보다 정확하게 분석할 수 있을 것으로 판단됨.

3-3 고용구조와 영업이익의 관계 분석 및 회귀 모델링

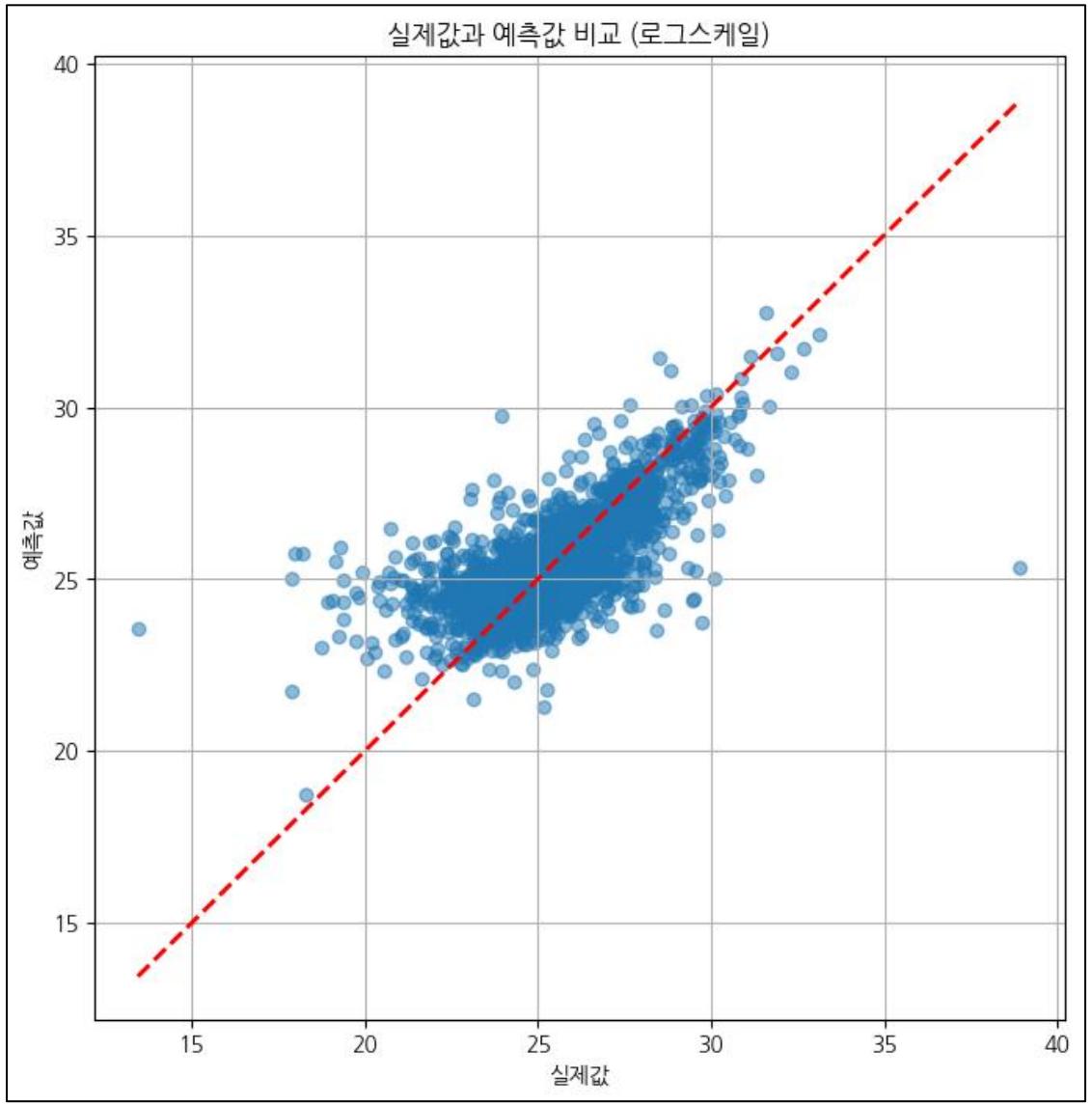
```
# 고용구조와 영업이익의 관계 분석 xgboost

df = df_original

# 1. 매출액 / 평균급여 숫자 정제
for col in ['영업이익', '평균급여']:
    df[col] = (
        df[col]
        .astype(str)
        .str.replace(r'[^0-9]', '', regex=True)   # 숫자만 남기기
    )
    df[col] = pd.to_numeric(df[col], errors='coerce')
# 로그 변환 가능하도록 양수만 남기기
df = df[(df['영업이익'] > 0) & (df['평균급여'] > 0)]

# 2. 사용할 변수
features = [
    '평균급여',
    '남성직원수', '여성직원수',
    '정규직수', '계약직수',
    '남성정규직수', '여성정규직수',
    '남성계약직수', '여성계약직수',
]

target = '영업이익'

# 3. 결측치 제거
data = df[features + [target]].dropna()

X = data[features]
y = data[target]

# 4. 로그 변환
y_log = np.log1p(y)  # log(1+y)


# 5. 학습 데이터 분리
X_train, X_test, y_train_log, y_test_log = train_test_split(
    X, y_log, test_size=0.2, random_state=42
)
```

```
# 6. XGBoost 회귀 모델 학습
model = XGBRegressor(
    n_estimators=400,
    learning_rate=0.05,
    max_depth=6,
    subsample=0.8,
    colsample_bytree=0.8,
    objective='reg:squarederror',
    random_state=42
)
model.fit(X_train, y_train_log)

# 7. 예측 및 평가
pred_log = model.predict(X_test)
y_pred = np.expm1(pred_log)     # 예측 복원
y_test = np.expm1(y_test_log)   # 실제값 복원

mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)
r2 = r2_score(y_test, y_pred)

print("MSE :", mse)
print("RMSE:", rmse)
print("R²  :", r2)

# 8. Feature Importance 시각화
importance = model.feature_importances_

plt.figure(figsize=(10,7))
plt.barh(features, importance)
plt.title("영업이익에 대한 피처 중요도")
plt.xlabel("Importance")
plt.show()

# 9. 실제값 vs 예측값 시각화 (로그스케일)
plt.figure(figsize=(8, 8))
plt.scatter(np.log1p(y_test), np.log1p(y_pred), alpha=0.5)
plt.plot([np.log1p(y_test).min(), np.log1p(y_test).max()],
         [np.log1p(y_test).min(), np.log1p(y_test).max()],
         'r--', linewidth=2)

plt.xlabel("실제값")
plt.ylabel("예측값")
plt.title("실제값과 예측값 비교 (로그스케일)")
plt.grid(True)
plt.show()
```

3-3 고용구조와 영업이익의 관계 분석 및 회귀 모델링



3-3 고용구조와 영업이익의 관계 분석 및 회귀 모델링



3-4 고용구조와 매출액의 관계 분석 및 회귀 모델링

3-4 고용구조와 매출액의 관계 분석 및 회귀 모델링

1. 주제 선정 이유

고용구조가 매출액에 미치는 영향을 분석하고자 함.

2. 데이터 및 변수 설명

독립 변수 : 평균급여, 남성직원수, 여성직원수, 정규직수, 계약직수,남성정규직수, 여성정규직수, 남성계약직수, 여성계약직수

종속 변수 : 매출액

전처리 : 매출액·평균급여에서 숫자만 추출, 0 이하 값 제거 후 로그변환, 결측치 행 제거, Train/Test = 80:20, 종속변수는 log(1+y) 변환

3. 분석 방법

XGBoost 회귀 모델을 사용하여 영업이익을 예측함.

모델은 최적화를 통해 트리 400개, max_depth=6, learning_rate=0.05 등 파라미터를 사용함.

Subsample, colsample을 통해 과적합을 방지함.

학습 후 MSE, RMSE, R^2를 계산하여 모델의 예측력을 평가함.

Feature Importance 시각화를 통해 어떤 변수가 영업이익에 더 큰 영향을 미쳤는지 파악함.

또한 실제값과 예측값을 비교하여 예측 패턴을 시각적으로 확인함.

3-4 고용구조와 매출액의 관계 분석 및 회귀 모델링

4. 모델 성능 결과 정리

MSE : 1.00139e+27 / RMSE : 31644878682239 / R^2 : 0.86851977843757 16

R^2 값이 약 0.87로 매우 높게 나타났으며, 이는 고용구조 지표만으로도 매출액의 상당 부분을 설명할 수 있음을 의미함.

매출액은 이익과 달리 규모 중심의 지표이므로,직원 규모·평균급여와 같은 기업 규모 관련 변수들이

매출액과 구조적으로 연결되어 있는 영향이 반영된 것으로 보임.

5. 데이터 및 변수 설명

변수 중요도 분석 결과, 매출액 증감율에 영향을 미치는 변수 순위는 다음과 같음.

[남성정규직수 > 정규직수 > 남성직원수 > 평균급여 > 여성계약직수 > 여성정규직수 > 남성계약직수 > 여성직원수 > 계약직수]

정규직 관련 변수, 특히 남성 정규직 규모가 가장 중요한 요인으로 나타났음을 의미함.

이는 기업 규모가 확대되며 정규직 인력이 증가하는 구조적 특성과 정규직 인력 규모가 매출 규모와 직접적으로 연결된다는 점을 보여줌.

반면 계약직 규모는 상대적으로 영향력이 낮게 나타났으며, 이는 계약직 변동보다는 정규직 기반의 조직 구조가 매출과 더 밀접한 관계를 가진다는 것을 나타냄.

3-4 고용구조와 매출액의 관계 분석 및 회귀 모델링

6. 실제값 vs 예측값 시각화 해석 (로그스케일 산점도)

실제값과 예측값을 비교한 산점도에서는 대부분의 점이 대각선 근처에 잘 분포하고 있으며,

이는 모델이 매출액의 로그 패턴을 비교적 안정적으로 예측하고 있음을 의미함.

일부 고매출 기업의 구간에서 약간의 편차가 존재하지만, 전반적으로 예측 정확도가 높게 나타남.

7. 결론

분석 결과, XGBoost 모델은 고용구조 지표만으로도 매출액을 매우 높은 수준으로 설명하는 것으로 나타났음.

특히 정규직 관련 변수들은 매출액과 강한 연관성을 보이며, 상대적으로 계약직은 매출액과 연관성이 거의 없는 것으로 나타남.

이는 기업의 인력 구성과 규모가 매출 규모와 직접적으로 연결되어 있음을 시사함.

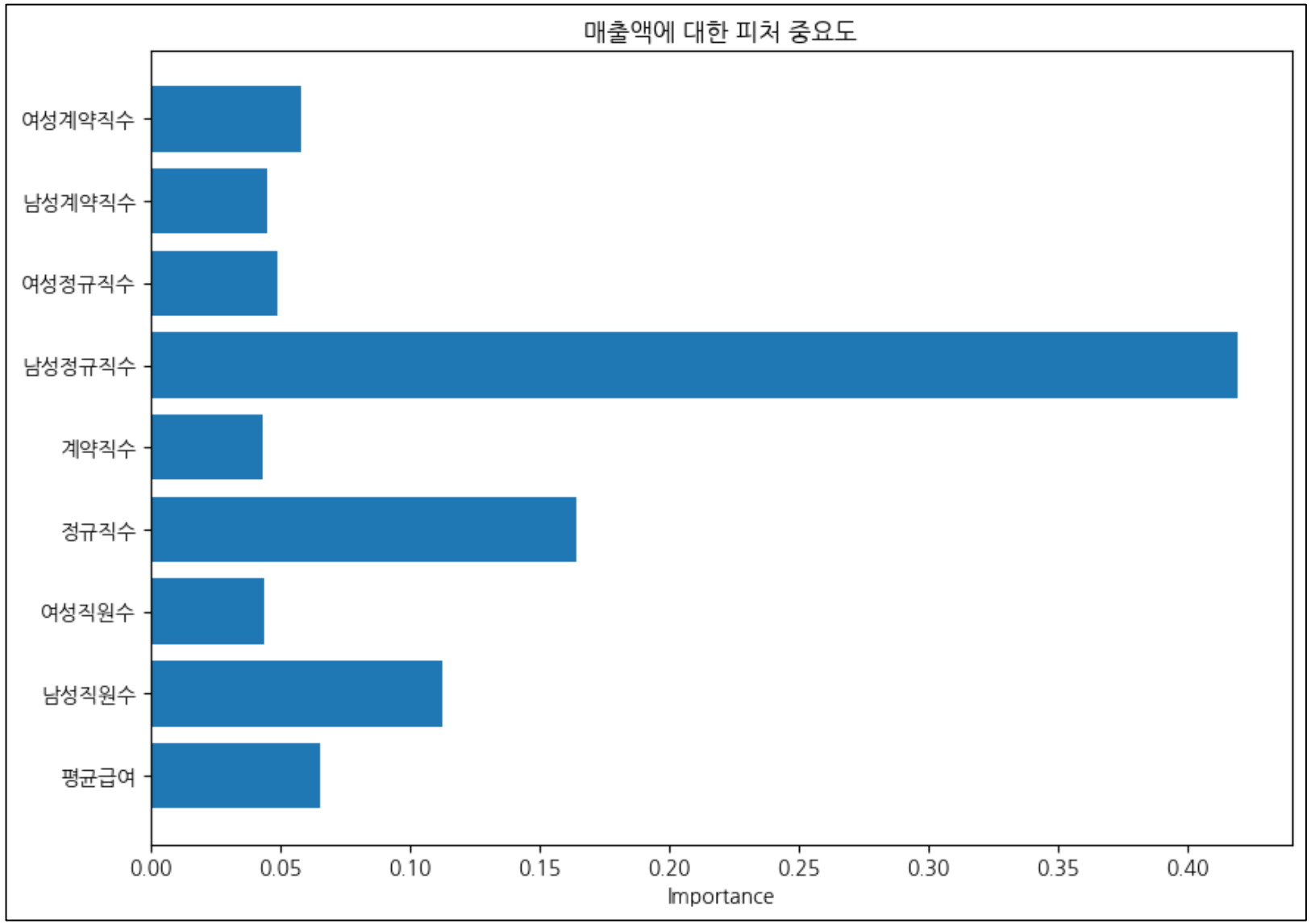
3-4 고용구조와 매출액의 관계 분석 및 회귀 모델링

```python
# 고용구조와 매출액의 관계 분석 xgboost

df = df_original

# 1. 매출액 / 평균급여 숫자 정제
for col in ['매출액', '평균급여']:
    df[col] = (
        df[col]
        .astype(str)
        .str.replace(r'[^0-9]', '', regex=True)   # 숫자만 남기기
    )
    df[col] = pd.to_numeric(df[col], errors='coerce')

# 로그 변환 가능하도록 양수만 남기기
df = df[(df['매출액'] > 0) & (df['평균급여'] > 0)]

# 2. 사용할 변수
features = [
    '평균급여',
    '남성직원수', '여성직원수',
    '정규직수', '계약직수',
    '남성정규직수', '여성정규직수',
    '남성계약직수', '여성계약직수',
]

target = '매출액'
```

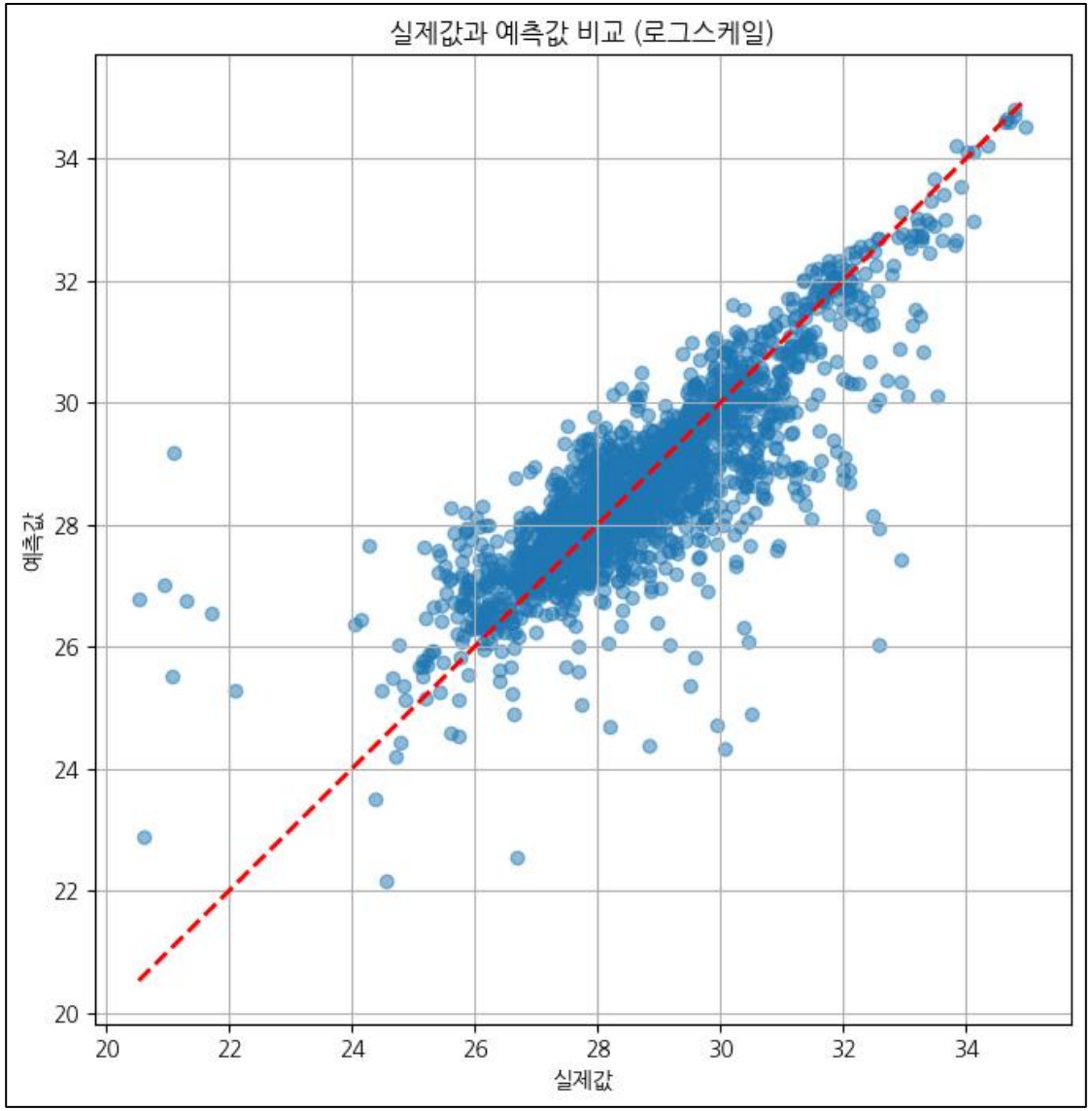
```python
# 3. 결측치 제거
data = df[features + [target]].dropna()

X = data[features]
y = data[target]

# 4. 로그 변환
y_log = np.log1p(y)  # log(1+y)

# 5. 학습 데이터 분리
X_train, X_test, y_train_log, y_test_log = train_test_split(
    X, y_log, test_size=0.2, random_state=42
)

# 6. XGBoost 회귀 모델 학습
model = XGBRegressor(
    n_estimators=400,
    learning_rate=0.05,
    max_depth=6,
    subsample=0.8,
    colsample_bytree=0.8,
    objective='reg:squarederror',
    random_state=42
)

model.fit(X_train, y_train_log)
```

```python
# 7. 예측 및 평가
pred_log = model.predict(X_test)
y_pred = np.expm1(pred_log)     # 예측 복원
y_test = np.expm1(y_test_log)   # 실제값 복원

mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)
r2 = r2_score(y_test, y_pred)

print("MSE :", mse)
print("RMSE:", rmse)
print("R²  :", r2)

# 8. Feature Importance 시각화
importance = model.feature_importances_

plt.figure(figsize=(10,7))
plt.barh(features, importance)
plt.title("매출액에 대한 피처 중요도")
plt.xlabel("Importance")
plt.show()

# 9. 실제값 vs 예측값 시각화 (로그스케일)
plt.figure(figsize=(8, 8))
plt.scatter(np.log1p(y_test), np.log1p(y_pred), alpha=0.5)
plt.plot([np.log1p(y_test).min(), np.log1p(y_test).max()],
         [np.log1p(y_test).min(), np.log1p(y_test).max()],
         'r--', linewidth=2)

plt.xlabel("실제값")
plt.ylabel("예측값")
plt.title("실제값과 예측값 비교 (로그스케일)")
plt.grid(True)
plt.show()
```

3-4 고용구조와 매출액의 관계 분석 및 회귀 모델링



3-4 고용구조와 매출액의 관계 분석 및 회귀 모델링



4. 프로젝트 후기

4. 프로젝트 후기

이번 프로젝트를 통해 실제 기업 데이터를 활용해 고용구조와 재무성과의 연관성을 분석하는 경험을 얻을 수 있었으며 데이터 수집부터 전처리, 시각화, 회귀 분석까지 전 과정을 직접 수행하면서 다음과 같은 배움을 얻었다.

1. **실제 산업 데이터의 복잡성 경험**

   재무제표의 기준이나 산정 방식이 연도별 제도 변화에 따라 달라져 동일한 지표를 일관성 있게 다루기 어려웠고,
   그 결과 목표했던 규모의 데이터를 온전히 확보하기 힘들었다. 이 과정을 통해 좋은 분석은 결국 탄탄한 데이터 구조 설계에서 출발한다는 사실을 깨달았다. 앞으로 관련 개발 또는 데이터 엔지니어링 업무를 맡게 된다면, 변화에 강하고 장기적으로 유지될 수 있는 데이터 스키마 설계가 얼마나 중요한지 알고 신중하게 설계해야겠다는 생각을 하게 되었다.

2. **전처리의 중요성**

   결측치 처리뿐 아니라, CSV 파일 변환 과정에서 쉼표나 공백 때문에 파싱 에러가 발생하는 등 예상치 못한 문제가 반복되었고, 그때마다 정규표현식, Pandas 옵션, 인코딩 설정 등을 점검하며 해결해야 했다. 단순히 데이터를 불러와서 사용하는 것이 아니라, 데이터가 컴퓨터에서 어떻게 저장되고, 어떤 방식으로 파싱되고, 어떤 오류가 발생하는지까지 고려해야 한다는 점을 몸소 느꼈다.

4. 프로젝트 후기

3. **모델링은 단순한 알고리즘 선택이 아님**

   여러 차례의 모델 학습과 시각화 과정에서 모델링이 단순히 알고리즘 선택으로 끝나는 것이 아니라, 데이터의 특성과 분포, 스케일 차이를 이해하는 과정이라는 것도 배웠다. 스케일을 맞추지 않은 상태에서 회귀 모델을 적용하면 극단적인 값이 모델을 왜곡하는 문제를 경험했고, 이를 해결하기 위해 로그 스케일 적용, 이상치 제거, 데이터 표준화 등을 반복하면서 분석의 완성도가 달라지는 것을 확인했다.

전반적으로 이번 프로젝트는 단순한 과제 수행을 넘어 데이터 엔지니어링–데이터 분석–머신러닝 모델링까지 이어지는 전체 파이프라인을 직접 경험해 볼 수 있는 과정이었다. 졸업을 앞두고 실제 업무에서 마주하게 될 문제 해결 방식과 기술적 사고를 연습할 수 있었던 점에서 매우 값진 시간이었다.다만, 프로젝트 후반에서야 이 정도 규모의 데이터만으로는 충분히 유의미한 예측 모델을 만들기 어렵다는 사실을 깨닫게 되어 아쉬움도 남았다. 더 이른 단계에서 데이터의 한계와 방향성을 점검했더라면 분석의 깊이를 조금 더 확보할 수 있었을 것이라 생각한다.그럼에도 불구하고, 평소 주식과 기업 분석에 관심이 많았던 만큼 실제 기업의 고용 구조와 재무 지표를 직접 다루고 시각화하고 모델링해보는 과정은 매우 흥미로웠다. 단순한 관심사를 넘어, 데이터를 기반으로 기업 구조를 읽어내는 경험을 해볼 수 있었던 의미 있는 프로젝트였다.