2024-2학기
빅데이터처리 프로젝트

3학년 C반 202244108 이태규

기후 변화 데이터 수집 및 향후 10년단위 예측

데이터 소스 예시

- **기상청**:기후 및 날씨와 관련된 데이터,오픈 API 사용 가능
- **환경부**
- **한국환경공단**:온실가스 배출량, 에너지 사용량
- **국가통계포털 (KOSIS)**:연도별 기온 변화, 강수량, 에너지 소비량
- **NASA GISS (Goddard Institute for Space Studies)**: 전 세계적인 기후 변화를 분석하는 데이터

로드맵

데이터 수집

- 기상청 자료 개방포털(기온,강수량)
- 국가 통계 포털(탄소 배출량)

데이터 저장

- Google Drive

데이터 가공/정제

- Pandas
- Scikit-learn

데이터 분석

- Python

데이터 시각화

- Tableau
- Seaborn
- Matplotlib

데이터 수집

- 기상청 자료개방포털(기온,강수량)

1904년 부터 2024년까지의 지역별 평균기온 및 강수량 데이터 수집

- 국가 통계 포털(탄소 배출량)

1999년부터 2024년까지의 지역별 탄소배출량 수집

```python
def preprocess_climate_data(data):
    # 날짜 컬럼 변환
    date_columns = {
        '최저기온 나타난날(yyyymmdd)': '최저기온 날짜',
        '최고기온 나타난날(yyyymmdd)': '최고기온 날짜',
        '일최다강수량 나타난날(yyyymmdd)': '일최다강수량 날짜'
    }

    for old_col, new_col in date_columns.items():
        if old_col in data.columns:
            data[new_col] = pd.to_datetime(data[old_col], format='%Y%m%d', errors

    # 수치형 결측치 처리
    numeric_columns = data.select_dtypes(include=['float64', 'int64']).columns
    imputer = SimpleImputer(strategy='mean')
    data[numeric_columns] = imputer.fit_transform(data[numeric_columns])

    # 연도 컬럼 생성
    data['연도'] = pd.to_datetime(data['일시']).dt.year

    return data

if __name__ == "__main__":
    file_path = '/content/ClimateData2.csv'

    # 데이터 로딩
    raw_data = load_climate_data(file_path)

    # 데이터 전처리
    processed_data = preprocess_climate_data(raw_data)

    # 데이터 요약
    summary = get_data_summary(processed_data)

    # 결과 출력
    print("데이터 처리 완료")
    print(f"데이터 기간: {summary['temporal_range']['start_year']}년 ~ {summary['tem
```

데이터 전처리 및 결측치 처리 과정 (기온,강수량 데이터)

날짜 컬럼 변환:
•여러 날짜 관련 컬럼을 datetime 형식으로 변환하여 시간 기반 분석을 용이하게 합니다.

결측치 처리:
•수치형 데이터의 결측값을 평균값으로 대체하여 데이터의 일관성을 유지합니다.

연도 정보 추가:
•연도 컬럼을 생성하여 연도별 분석을 용이하게 합니다.

•데이터 요약 통계:
•데이터프레임의 구조, 결측값 현황, 기술 통계, 시간적 범위를 요약하여 데이터에 대한 전반적인 이해를 제공합니다.

```python
def load_and_process_data():
    # Read CSV
    df = pd.read_csv('/content/Co2.csv', encoding='utf-8-sig')

    # Get numeric columns (excluding first column)
    numeric_cols = df.columns[1:]

    # Convert to numeric, handle missing values
    for col in numeric_cols:
        df[col] = pd.to_numeric(df[col], errors='coerce')

    # Fill NaN with 0 for emission data
    df[numeric_cols] = df[numeric_cols].fillna(0)

    # Remove outliers
    for col in numeric_cols:
        z_scores = np.abs(stats.zscore(df[col], nan_policy='omit'))
        df.
    return
```

```python
def prepare_time_series(df):
    years = sorted(list(set([col[:4] for col in df.columns if col != '구분(1)'])))
    results = []

    for _, row in df.iterrows():
        region = row['구분(1)']
        for year in years:
            year_cols = [col for col in df.columns if col.startswith(year)]
            if year_cols:
                year_data = row[year_cols]
                total = year_data.sum()
                results.append({
                    'Region': region,
                    'Year': int(year),
                    'Total_Emissions': total
                })

    results_df = pd.DataFrame(results)

    # Add descriptive statistics
    results_df['YoY_Change'] = results_df.groupby('Region')['Total_Emissions'].pct

    return results_df
```

데이터 전처리 및 결측치 처리 과정(Co2 데이터)

데이터 로드 및 전처리:
•CO_2 배출 데이터 CSV 파일을 읽어들이고, 수치형 데이터로 변환 및 결측치를 0으로 대체합니다.

시간 시계열 데이터 준비:
•각 지역과 연도별로 총 CO_2 배출량을 계산하여 새로운 데이터프레임을 생성합니다.

데이터 요약 및 통계 분석:
•전처리된 데이터의 형태와 결측치 현황을 확인합니다.
•지역별로 CO_2 배출량의 평균, 표준편차, 최소값, 최대값 및 전년 대비 변화율의 평균을 계산하여 요약 통계를 생성합니다.

결과 출력:
•데이터의 형태와 결측치 개수를 출력하여 데이터의 상태를 파악합니다.
•지역별 요약 통계를 출력하여 각 지역의 CO_2 배출량에 대한 전반적인 통계 정보를 제공합니다.

기후 데이터 분석 및 예측 모델 구축

기온 모델 (temp_model)학습:

-독립 변수 (X_temp): 연도.

-종속 변수 (y_temp): 평균 기온.

강수량 모델 (rain_model)학습:

-독립 변수 (X_rain): 연도.

-종속 변수 (y_rain): 월합 강수량.

각 모델의 결정 계수 (R^2)를 계산하여 반환.

```
try:
    yearly_temp, yearly_rain = self.prepare_yearly_data()

    X_temp = yearly_temp['연도'].values.reshape(-1, 1)
    y_temp = yearly_temp['평균기온(°C)'].values
    self.temp_model.fit(X_temp, y_temp)

    X_rain = yearly_rain['연도'].values.reshape(-1, 1)
    y_rain = yearly_rain['월합강수량(00~24h만)(mm)'].values
    self.rain_model.fit(X_rain, y_rain)

    return {
        'temp_r2': r2_score(y_temp, self.temp_model.predict(X_temp)),
        'rain_r2': r2_score(y_rain, self.rain_model.predict(X_rain))
    }
except Exception as e:
    raise Exception(f"모델 학습 중 오류 발생: {str(e)}")
```

미래 예측 메서드

- 모델이 학습되었는지 확인 (coef_ 속성 존재 여부).
- 연도별 기온과 강수량 데이터 준비.
- 마지막 연도 이후의 미래 연도 목록 생성.
- 미래 연도에 대해 예측 수행.
- 예측 결과(연도, 기온, 강수량)를 딕셔너리 형태로 반환.

```
predict_future(self, years_ahead: int = 10) -> Dict[str, np.r
if not hasattr(self.temp_model, 'coef_') or not hasattr(self.
    raise ValueError("예측하기 전에 먼저 모델을 학습시켜야 합니다.")

yearly_temp, yearly_rain = self.prepare_yearly_data()
last_year = int(yearly_temp['연도'].max())
future_years = np.array(range(last_year + 1, last_year + year

future_years_reshaped = future_years.reshape(-1, 1)
return {
    'years': future_years,
    'temperature': self.temp_model.predict(future_years_resha
    'rainfall': self.rain_model.predict(future_years_reshaped
}
```

예측 결과 시각화 메서드

- 연도별 평균 기온을 선 그래프로 표시 (실제 기온).
- 예측된 기온을 점선(r--)으로 표시.
- 연도별 월합 강수량을 막대 그래프로 표시 (실제 강수량).
- 예측된 강수량을 점선(g--)으로 표시.
- 레이아웃 조정 후 그래프 표시.

```python
def plot_predictions(self, predictions: Dict[str, np.ndarray]):
    yearly_temp, yearly_rain = self.prepare_yearly_data()

    fig, (ax1, ax2) = plt.subplots(2, 1, figsize=(15, 12))

    sns.lineplot(data=yearly_temp, x='연도', y='평균기온(°C)', ax=ax1, marker='o', la
    ax1.plot(predictions['years'], predictions['temperature'], 'r--', label='예측 
    ax1.set_title('연도별 평균 기온 예측', fontproperties=self.font, fontsize=14)
    ax1.set_xlabel('연도', fontproperties=self.font)
    ax1.set_ylabel('평균 기온(°C)', fontproperties=self.font)
    ax1.legend(prop=self.font)

    sns.barplot(data=yearly_rain, x='연도', y='월합강수량(00~24h만)(mm)', ax=ax2, alph
    ax2.plot(predictions['years'], predictions['rainfall'], 'g--', label='예측 강수
    ax2.set_title('연도별 강수량 예측', fontproperties=self.font, fontsize=14)
    ax2.set_xlabel('연도', fontproperties=self.font)
    ax2.set_ylabel('강수량 (mm)', fontproperties=self.font)
    ax2.legend(prop=self.font)

    plt.tight_layout()
    plt.show()
```

Co2 데이터 전처리 및 예측 모델 구축

데이터 전처리 및 준비:

- CO_2 배출량 데이터를 로드하고, 수치형 데이터로 변환하여 결측치를 처리하였습니다.
- '전국' 단위의 연도별 배출량을 계산하여 시계열 데이터를 생성하였습니다.

추세 분석:

- 선형 회귀를 통해 연도별 CO_2 배출량의 추세를 분석하였습니다.
- 과거 데이터를 기반으로 미래의 배출량을 예측하였습니다.

시각화:

- 실제 배출량, 이동 평균, 미래 추세를 시각적으로 비교하여 분석의 직관성을 높였습니다.
- 시각화를 통해 배출량의 증가 추세를 명확히 파악할 수 있습니다.

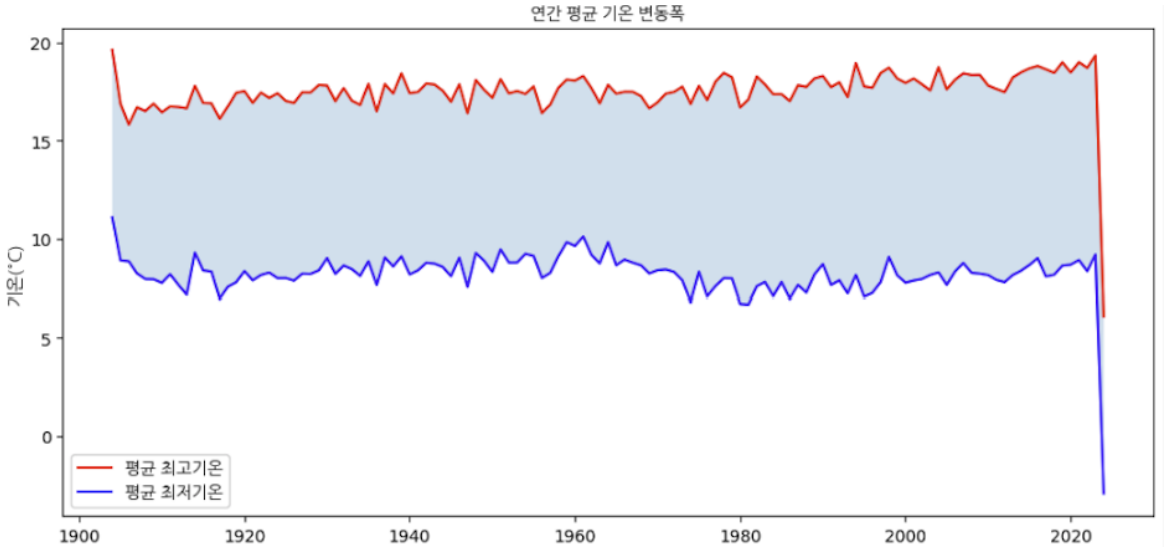
통계 정보 제공:

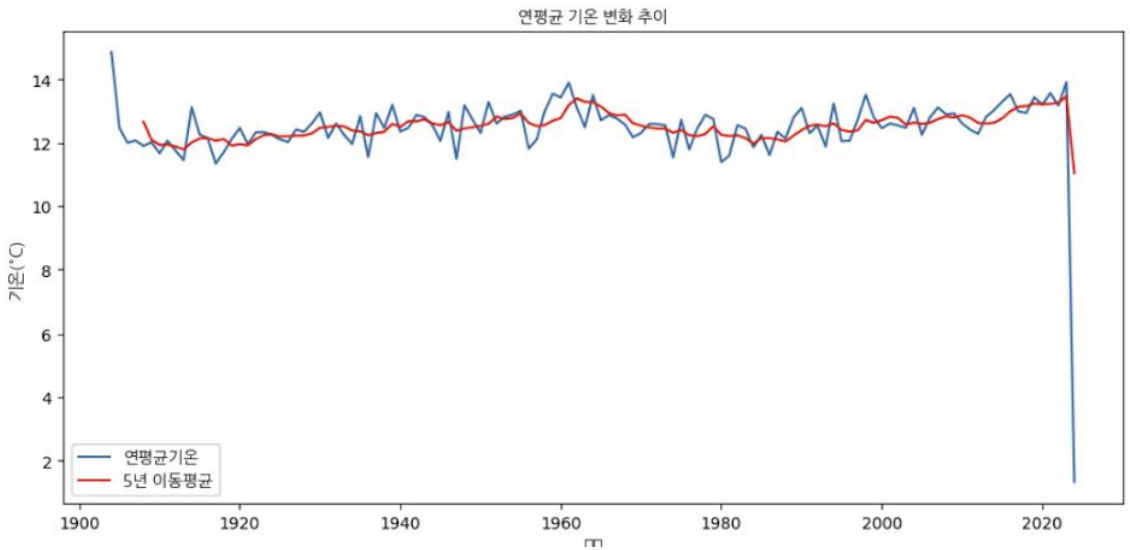
- 최근 5년간의 실제 배출량과 미래 10년간의 예측 배출량을 출력하여 데이터를 이해하는 데 도움을 주었습니다.
- 평균 연간 성장률을 계산하여 배출량 증가 속도를 평가하였습니다.

```
# Get national data by year
years = []
emissions = []
national_data = df[df['구분(1)'] == '전국'].iloc[0]

for year in range(1999, 2022):
    year_cols = [col for col in df.columns if str(year) in col]
    if year_cols:
        years.append(year)
        emissions.append(national_data[year_cols].sum())

# Convert to numpy arrays
years = np.array(years)
emissions = np.array(emissions)
```

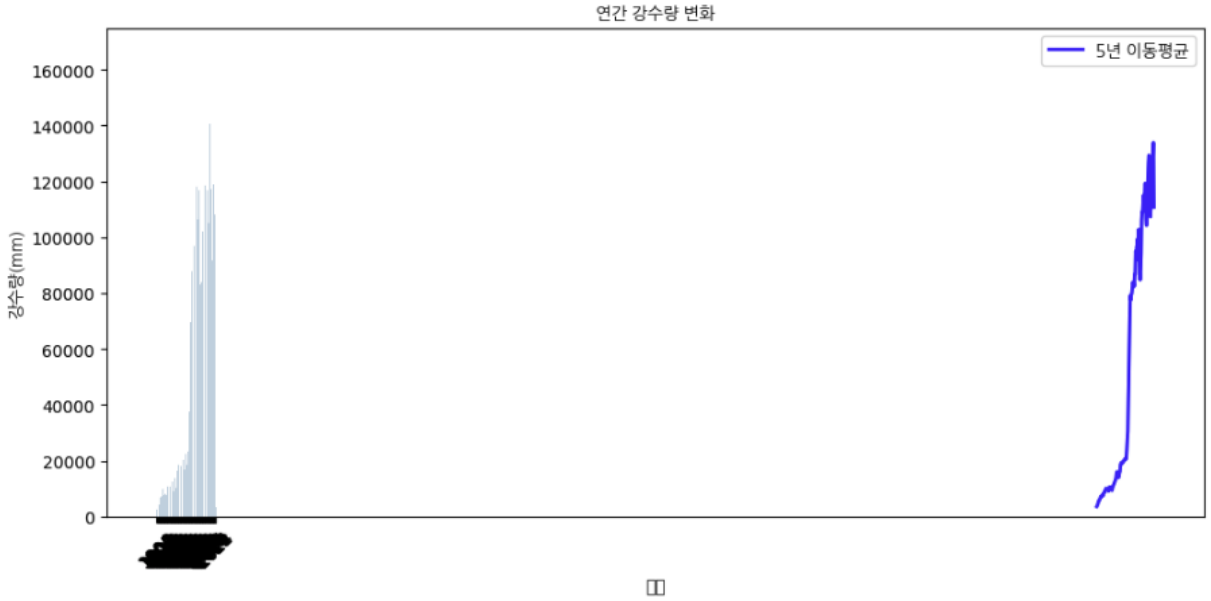
```
# Calculate trend
slope, intercept, r_value, p_value, std_err = linregress(years, emissions)

# Future predictions
future_years = np.array(range(2022, 2031))
trend = slope * future_years + intercept
```

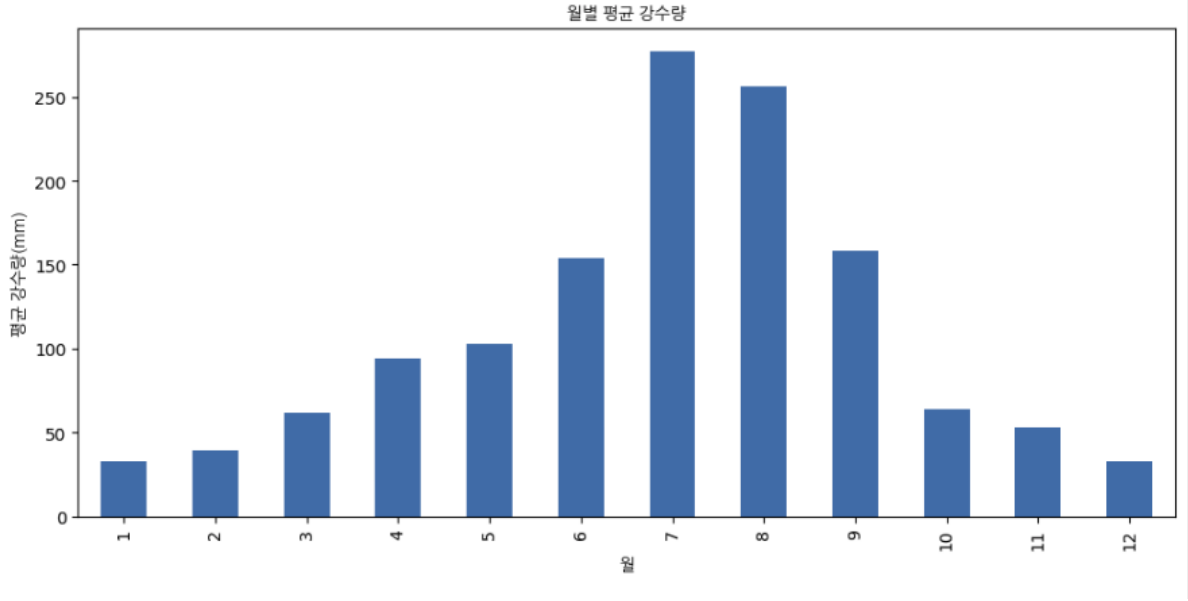
연평균 기온 변동폭과 변화 추이





연평균 강수량 변동폭 과 변화 추이



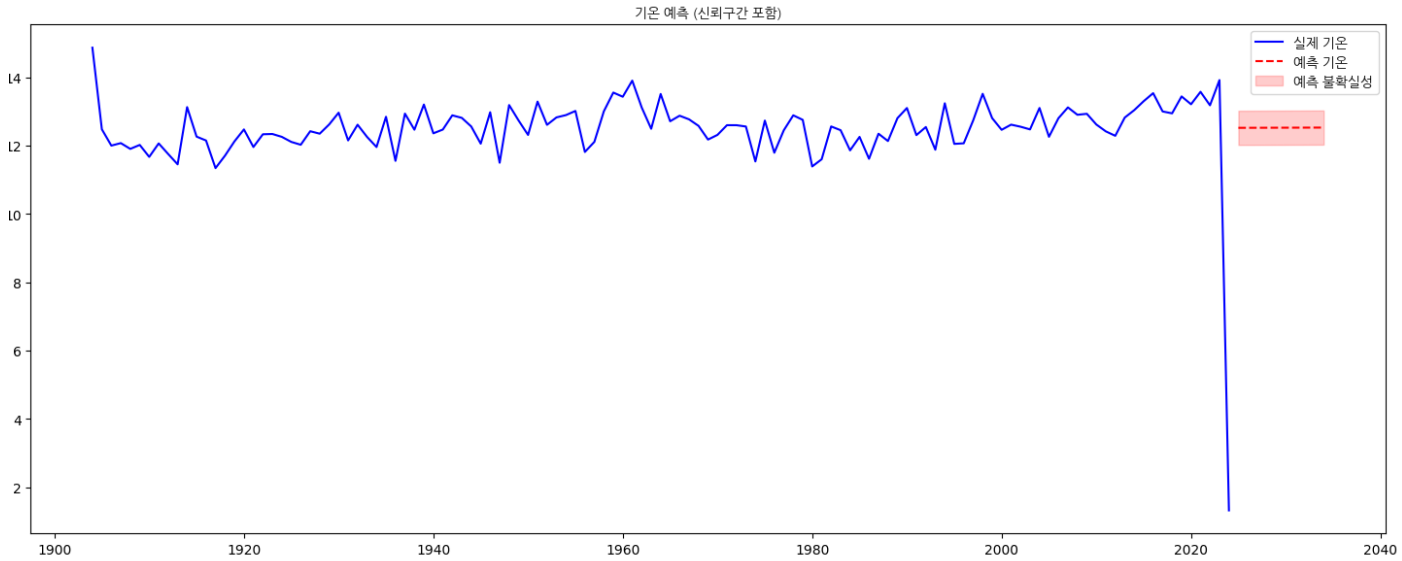


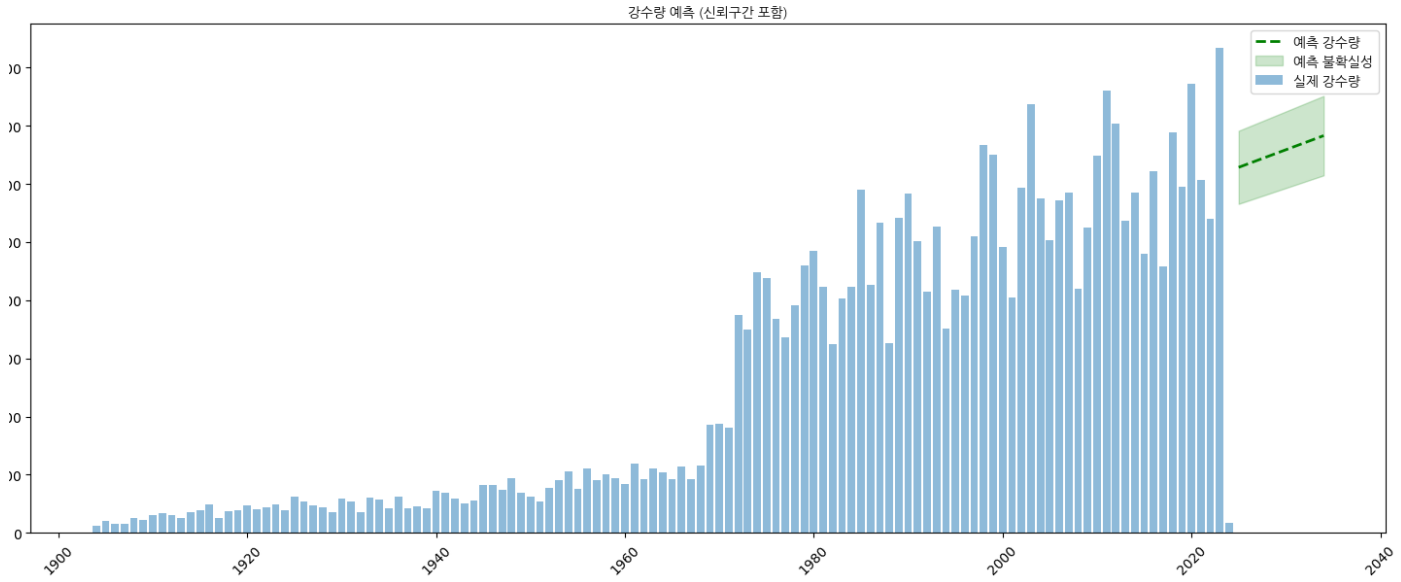
향후 10년단위 기온과 강수량 예측

=== 미래 예측 결과 ===

- 10년 후 예상 평균기온: 12.52°C

- 10년 후 예상 연강수량: 136578.2mm





10년 단위 계절 변화 분석

=== 10년 단위 계절 변화 분석 ===

봄철:

- 총 변화량: -0.59°C
- 10년당 평균 변화율: -0.08°C
- 변동성 변화: 0.04°C

여름철:

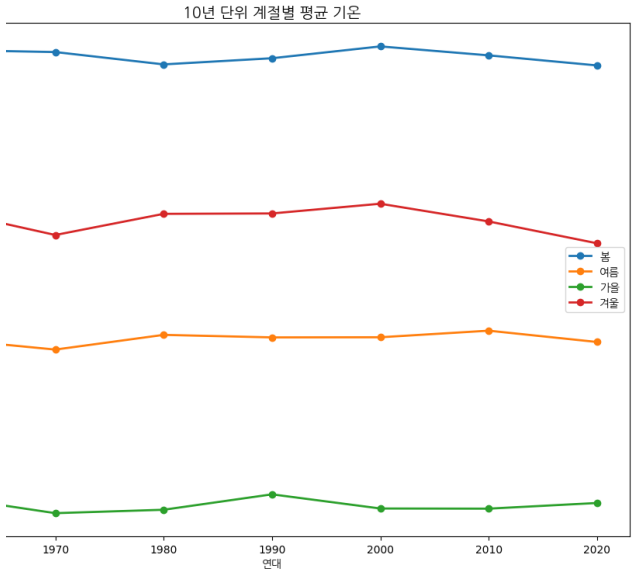
- 총 변화량: -0.11°C
- 10년당 평균 변화율: -0.02°C
- 변동성 변화: 0.39°C

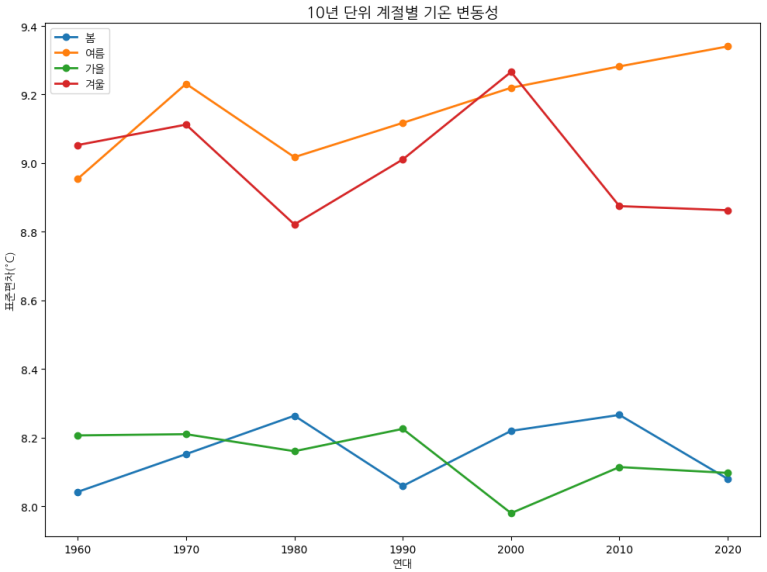
가을철: 총 변화량: -0.23°C

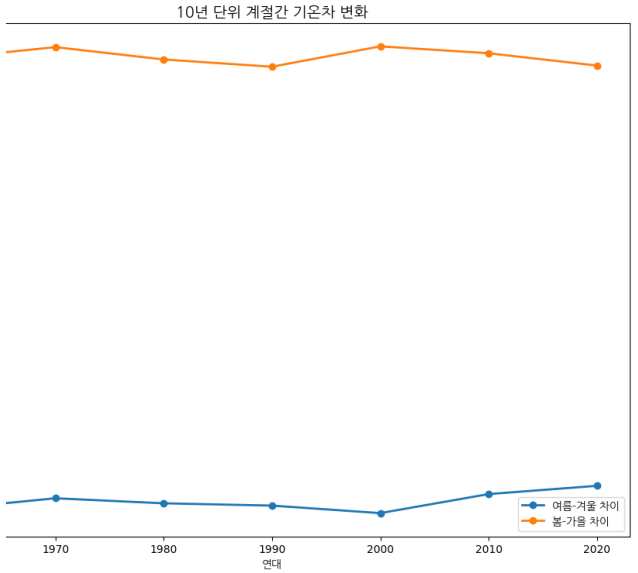
- 10년당 평균 변화율: -0.03°C
- 변동성 변화: -0.11°C

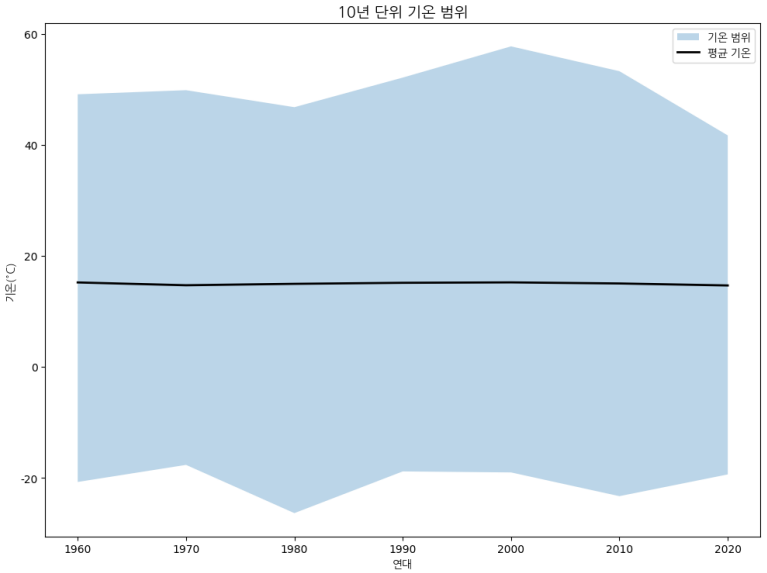
겨울철: 총 변화량: -1.24°C

- 10년당 평균 변화율: -0.18°C
- 변동성 변화: -0.19°C
- 계절차 변화: 여름-겨울
- 기온차 변화: 1.13°C









향후 10년 탄소 배출량 예측

2025: 1382.00

2026: 1370.17

2027: 1358.33

2028: 1346.49

2029: 1334.66

2030: 1322.82

2031: 1310.99

2032: 1299.15

2033: 1287.32

2034: 1275.48

2035: 1263.65

프로젝트 결론

- 지구의 평균 온도가 상승함에 따라 온난화의 진행 속도가 가속화되고 있습니다. 최근 100년간의 데이터를 분석하고 통계를 통해 평균 기온이 지속적으로 증가하고 있음을 확인했습니다. 특히, 강수량이 예상보다 큰 폭으로 증가하고 있음을 관찰할 수 있습니다.
- 또한, 탄소 배출량 데이터를 분석한 결과 시간이 지남에 따라 배출량이 조금씩 감소하고 있음을 확인했습니다.
- 급격한 기후 변화에 대응하기 위해 개인은 물론 전 지구적인 참여가 절실히 필요하다고 생각합니다.