국가 별 1인 생산기반 CO2 배출량 분석

컴퓨터공학과 2023208018 이효민

Table of contents

1 개요 및 필요성

2 관련 연구 및 기대 효과

3 데이터 분석

4 결론

1

개요 및 필요성

CO2(탄소) 배출이란?

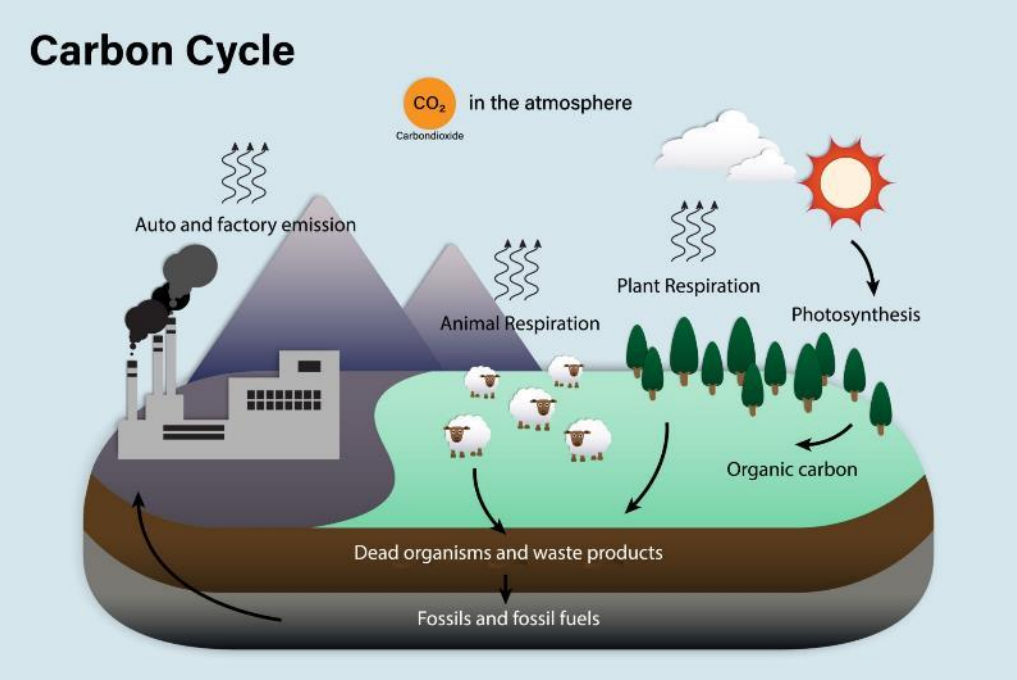
화석 연료 사용과 같은 다양한 이유로 인해
이산화탄소 기체가 대기 중으로 배출되는 현상

Part 1 >> **필요성**

탄소는 순환한다

하지만 인간의 탄소 배출량은

지구가 감당할 수 있는 양을 넘어섰다



취재K

탄소 배출 안 멈추면?…30년 뒤 여름은 석 달 내내 '찜통'

입력 2021.01.18 (14:05)

"지구 '1.5도 상승' 지키려면…2030년 탄소배출 43% 감축해야"

기후변화: 관측이래 가장 더웠다…세계 곳곳에 이상고온 현상

맷 맥그래스
Environment correspondent

2023년 7월 5일

Part 1 >> 필요성

탄소배출의 심각성과
영향력을 깨달을 수 있다

개인의 작은 노력이라도
함께 실천할 수 있다

어느 나라의 배출량이 높은 지
이유가 무엇인지 알 수 있다

2

관련 연구 및 기대 효과

Part 2 >> 관련 연구

IPCC 평가보고서

IPCC는 1988년 세계기상기구와 유엔환경계획이 기후변화의 과학적 규명을 위해 공동으로 설립한 국제협의체이다

- **전 지구 온실가스 배출 현황**
  - **(지구온난화)** 인간 활동으로 인한 온실가스 배출로 전 지구 지표 온도는 1850~1900년 대비 현재(2011~2020년) 1.1℃ 상승하였으며, 과거와 현재 모두 전 지구 온실가스 배출량의 지역·국가·개인에 따른 기여도는 균등하지 않음
  - **(누적 배출량)** 1850~2019년까지의 총 누적 탄소 배출량은 2400±240 $GtCO_2$임
  - **(연간 배출량)** 2019년 기준 전체 온실가스의 연간 배출량은 59±6.6 $GtCO_2$-eq로, 이는 2010년 대비 12% 증가한 수치임
  - **(온난화 기여도)** 현재의 전 지구 평균온도 상승량(약 1.1℃) 중 이산화탄소는 0.8℃, 메탄은 0.5℃, 질소산화물은 0.1℃, 불화가스는 0.1℃ 정도 기여한 것으로 분석되며, 기타 인간 유발 물질(에어로졸 등) 및 자연적 인자로 인한 냉각 효과가 일부 있는 것으로 분석됨

- **지구 온난화를 제한하기 위한 우리의 행동 방향성 제시**
  - **(넷제로(net zero) 필요성)** 인간이 유발한 온난화를 제한하려면 CO_2를 포함한 전체 온실가스의 배출량이 넷제로*가 되어야 함
    * 전체 온실가스의 배출 및 제거량을 이산화탄소 환산량으로 계산했을 때, 배출량과 제거량의 상쇄되어 순 배출량이 0이 되는 것
  - **(CO_2 감축 전략)** ①공급 부문 관리*, ②수요 관리** 및 효율 향상, ③이산화탄소 제거 접근법***을 활용하여 넷제로 배출을 달성할 수 있음
    * 에너지, 산업, 도시, 건물, 수송, 농업·임업·기타토지이용(AFOLU) 부문의 온실가스 감축 증대(예: 탄소배출저감기술을 활용하지 않은(unabated) 화석연료를 재생에너지 보급 또는 탄소 포집 및 저장(CCS) 기술 활용 등을 통해 저탄소·무탄소 전원으로 전환)
    ** 사회문화 및 행태적 변화, 인프라 활용, 최종소비자 기술 채택에 대한 관리 및 소비자 변화 유도를 통해 배출량 저감
    *** 이산화탄소 제거(CDR, carbon dioxide removal): 대기 중에서 온실가스를 직접적으로 제거(포집)하여 토지·지중·해양 저장소 또는 상품에 저장하는 감축 활동

- **지구온난화 진행 현황**
  - **(1.5℃ 도달)** 지속되어온 온실가스 배출로 인해 온난화가 심화되어 거의 모든 미래 배출 시나리오에서 가까운 미래(2021~2040년)에 1.5℃에 도달할 것으로 분석됨
  - **(현상)** 지구온난화를 제한하더라도 이미 발생한 비가역적 변화(해수면 상승, 남극 빙상 붕괴, 생물다양성 손실 등)는 돌이킬 수 없으며, 지구온난화가 진행될 수록 이러한 급격하고 비가역적인 변화가 일어날 가능성이 높아지고, 손실과 피해는 증가하며, 적응 한계에 도달하게 됨

- **탄소배출허용총량(carbon budget)**
  - **(탄소배출허용총량)** 지구온난화를 1.5℃로 제한하기 위한 목표 상의 2020년 초 이후의 잔여 탄소배출허용량(remaining carbon budget)은 500 $GtCO_2$(50% 확률)이고, 2℃ 미만으로 제한하기 위한 목표 상의 잔여 탄소배출허용량은 1,150 $GtCO_2$(67% 확률)로 분석됨
  - **(화석연료 인프라)** 현재 이미 계획되어 있는 화석연료 인프라에서 발생할 것으로 추산되는 CO_2 잠재 배출량만으로도 1.5℃ 목표 달성을 위한 잔여 탄소배출허용량을 초과함

Part 2 >> 관련 연구

탄소배출권

탄소배출권거래(Emissions Trading)는 온실가스 배출 권리인 '탄소배출권'을 시장을 통해 사고파는 행위를 의미한다
'탄소배출권'은 할당량 및 크레딧을 포괄하는 개념으로 할당량은 국가 또는 지역 내에서 정한 온실가스 배출총량만큼
발전 설비나 생산 설비 등 주요 온실가스 배출원에 지급된 온실가스 배출 권리를 의미하며,
크레딧은 외부 온실가스 저감 프로젝트에 대하여 기준 전망치 대비 온실가스 배출량을 줄였다는 증서로서
해당 프로젝트에 지급되는 배출권을 의미한다.

3

데이터 분석

Part 3 ⟩⟩ 데이터 분석

Part 3 》 데이터 분석

사용한 데이터

Countrywise Production-Based CO2 Emissions

CO2 Atlas: Navigating Global Emissions, Country by Country

Dataset Glossary (Column-wise)

- **ISO3** - ISO 3166-1 alpha-3 code representing the country
- **Country** - Name of the country
- **Continent** - Continent where the country is located
- **Hemisphere** - Hemisphere (Northern or Southern) to which the country belongs
- **Metric tons of CO2e per capita (1990)** - CO2 emissions in metric tons per capita for 1990
- **Metric tons of CO2e per capita (1995)** - CO2 emissions in metric tons per capita for 1995
- **Metric tons of CO2e per capita (2000)** - CO2 emissions in metric tons per capita for 2000
- **Metric tons of CO2e per capita (2005)** - CO2 emissions in metric tons per capita for 2005
- **Metric tons of CO2e per capita (2010)** - CO2 emissions in metric tons per capita for 2010
- **Metric tons of CO2e per capita (2013)** - CO2 emissions in metric tons per capita for 2013
- **Metric tons of CO2e per capita (2018)** - CO2 emissions in metric tons per capita for 2018

Part 3 》 데이터 분석 – 데이터 불러오기

```
# This Python 3 environment comes with many helpful analytics libraries installed
# It is defined by the kaggle/python Docker image: https://github.com/kaggle/docker-python
# For example, here's several helpful packages to load

import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

# Input data files are available in the read-only "../input/" directory
# For example, running this (by clicking run or pressing Shift+Enter) will list all files under the input directory

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))

# You can write up to 20GB to the current directory (/kaggle/working/) that gets preserved as output when you create a version using "Save & Run All"
# You can also write temporary files to /kaggle/temp/, but they won't be saved outside of the current session
```

```
/kaggle/input/production-based-co2-emissions/production_based_co2_emissions.csv
```

사용한 패키지

```
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import explained_variance_score
from sklearn.metrics import confusion_matrix
from sklearn.model_selection import train_test_split  # 함수
```

Part 3 》 데이터 분석 – 데이터 불러오기

Csv 파일을 불러오고 가장 위의 10개의 데이터를 불러오는 코드

```
data = pd.read_csv('/kaggle/input/production-based-co2-emissions/production_based_co2_emissions.csv')
data.head(10)
```

3] :

	ISO3	Country	Continent	Hemisphere	Metric tons of CO2e per capita (1990)	Metric tons of CO2e per capita (1995)	Metric tons of CO2e per capita (2000)	Metric tons of CO2e per capita (2005)	Metric tons of CO2e per capita (2010)	Metric tons of CO2e per capita (2013)	Metric tons of CO2e per capita (2018)
0	AFG	Afghanistan	Asia	Northern Hemisphere	1.24	0.84	0.82	0.76	1.06	1.40	2.66
1	ALB	Albania	Europe	Northern Hemisphere	3.56	2.15	2.36	2.68	2.83	2.85	3.52
2	DZA	Algeria	Africa	Northern Hemisphere	3.50	3.42	3.75	4.04	4.58	4.88	5.18
3	AGO	Angola	Africa	Southern Hemisphere	4.29	4.43	4.20	5.06	6.32	5.99	2.59
4	ATG	Antigua and Barbuda	America	Northern Hemisphere	6.31	6.53	7.08	8.77	10.69	11.22	12.59
5	ARG	Argentina	America	Southern Hemisphere	7.12	7.25	7.49	7.73	7.77	7.86	8.22
6	ARM	Armenia	Asia	Northern Hemisphere	NaN	1.94	1.82	2.30	2.46	2.97	3.17
7	AUS	Australia	Oceania	Southern Hemisphere	28.06	27.28	30.84	27.60	25.39	25.06	24.63
8	AUT	Austria	Europe	Northern Hemisphere	9.71	9.55	9.59	10.79	9.85	9.15	8.48
9	AZE	Azerbaijan	Asia	Northern Hemisphere	NaN	7.50	6.79	6.61	6.36	7.43	8.07

Part 3 》 데이터 분석 – 데이터 불러오기

```
print(data.columns)
```

```
Index(['ISO3', 'Country', 'Continent', 'Hemisphere',
       'Metric tons of CO2e per capita (1990)',
       'Metric tons of CO2e per capita (1995)',
       'Metric tons of CO2e per capita (2000)',
       'Metric tons of CO2e per capita (2005)',
       'Metric tons of CO2e per capita (2010)',
       'Metric tons of CO2e per capita (2013)',
       'Metric tons of CO2e per capita (2018)'],
      dtype='object')
```

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 181 entries, 0 to 180
Data columns (total 11 columns):
 #   Column                                  Non-Null Count  Dtype
---  ------                                  --------------  -----
 0   ISO3                                    181 non-null    object
 1   Country                                 181 non-null    object
 2   Continent                               181 non-null    object
 3   Hemisphere                              181 non-null    object
 4   Metric tons of CO2e per capita (1990)   154 non-null    float64
 5   Metric tons of CO2e per capita (1995)   177 non-null    float64
 6   Metric tons of CO2e per capita (2000)   179 non-null    float64
 7   Metric tons of CO2e per capita (2005)   179 non-null    float64
 8   Metric tons of CO2e per capita (2010)   181 non-null    float64
 9   Metric tons of CO2e per capita (2013)   180 non-null    float64
 10  Metric tons of CO2e per capita (2018)   180 non-null    float64
dtypes: float64(7), object(4)
memory usage: 15.7+ KB
```

```
for column in data:
    print(column,':', data[column].nunique())
```

```
ISO3 : 181
Country : 181
Continent : 5
Hemisphere : 2
Metric tons of CO2e per capita (1990) : 146
Metric tons of CO2e per capita (1995) : 161
Metric tons of CO2e per capita (2000) : 167
Metric tons of CO2e per capita (2005) : 168
Metric tons of CO2e per capita (2010) : 171
Metric tons of CO2e per capita (2013) : 165
Metric tons of CO2e per capita (2018) : 170
```

+ Code + Markdown

```
data.describe()
```

	Metric tons of CO2e per capita (1990)	Metric tons of CO2e per capita (1995)	Metric tons of CO2e per capita (2000)	Metric tons of CO2e per capita (2005)	Metric tons of CO2e per capita (2010)	Metric tons of CO2e per capita (2013)	Metric tons of CO2e per capita (2018)
count	154.000000	177.000000	179.000000	179.000000	181.000000	180.000000	180.000000
mean	6.556169	6.868023	7.049832	7.408603	7.226464	7.136833	6.794833
std	7.932022	8.815395	8.637687	9.340650	8.239457	7.900280	6.556862
min	0.420000	0.420000	0.350000	0.260000	0.290000	0.320000	0.490000
25%	1.467500	1.690000	1.760000	1.905000	1.960000	1.917500	2.340000
50%	3.450000	4.040000	4.380000	4.570000	4.730000	4.875000	4.870000
75%	8.615000	8.690000	8.845000	9.485000	9.300000	9.067500	8.577500
max	48.330000	73.130000	65.640000	76.410000	62.470000	54.410000	38.750000

Part 3 》 데이터 분석 – 데이터 전처리

1. 데이터 전처리를 위해 결측치 확인

```
missing_values = data.isnull().sum()
missing_values
```

```
ISO3                                    0
Country                                 0
Continent                               0
Hemisphere                              0
Metric tons of CO2e per capita (1990)   27
Metric tons of CO2e per capita (1995)    4
Metric tons of CO2e per capita (2000)    2
Metric tons of CO2e per capita (2005)    2
Metric tons of CO2e per capita (2010)    0
Metric tons of CO2e per capita (2013)    1
Metric tons of CO2e per capita (2018)    1
dtype: int64
```

2. 결측치의 정도를 확인(하나의 열 당 결측치가 10% 내외이기 때문에 채워서 사용을 결정)

```
# Checking the percentage of missing values for each column
missing_percentage = (data.isnull().sum() / len(data)) * 100
missing_percentage

# This will display the percentage of missing values for each column, helping us decide on the best strategy for handling them.
```

```
ISO3                                    0.000000
Country                                 0.000000
Continent                               0.000000
Hemisphere                              0.000000
Metric tons of CO2e per capita (1990)   14.917127
Metric tons of CO2e per capita (1995)    2.209945
Metric tons of CO2e per capita (2000)    1.104972
Metric tons of CO2e per capita (2005)    1.104972
Metric tons of CO2e per capita (2010)    0.000000
Metric tons of CO2e per capita (2013)    0.552486
Metric tons of CO2e per capita (2018)    0.552486
dtype: float64
```

```
# Filling missing values with the mean of the respective column
for column in data.columns:
    if data[column].dtype == 'float64':
        data[column].fillna(data[column].mean(), inplace=True)
```

3. 결측치를 평균값으로 추가

Part 3 》 데이터 분석 – 데이터 전처리

IQR 방법으로 특이치를 알아내기 위해 숫자 정보만 추출

```
# Select only the numeric columns for outlier detection
numeric_data = data.select_dtypes(include=[np.number])

# Using the IQR method to detect and handle outliers for numeric columns
Q1 = numeric_data.quantile(0.25)
Q3 = numeric_data.quantile(0.75)
IQR = Q3 - Q1

# Filtering out outliers
filtered_data = data[~((numeric_data < (Q1 - 1.5 * IQR)) | (numeric_data > (Q3 + 1.5 * IQR))).any(axis=1)]

# The Interquartile Range (IQR) method is used to detect and filter out outliers. This method is robust to ex
```

```
# Checking for any duplicate rows
duplicates = data.duplicated().sum()
duplicates
```

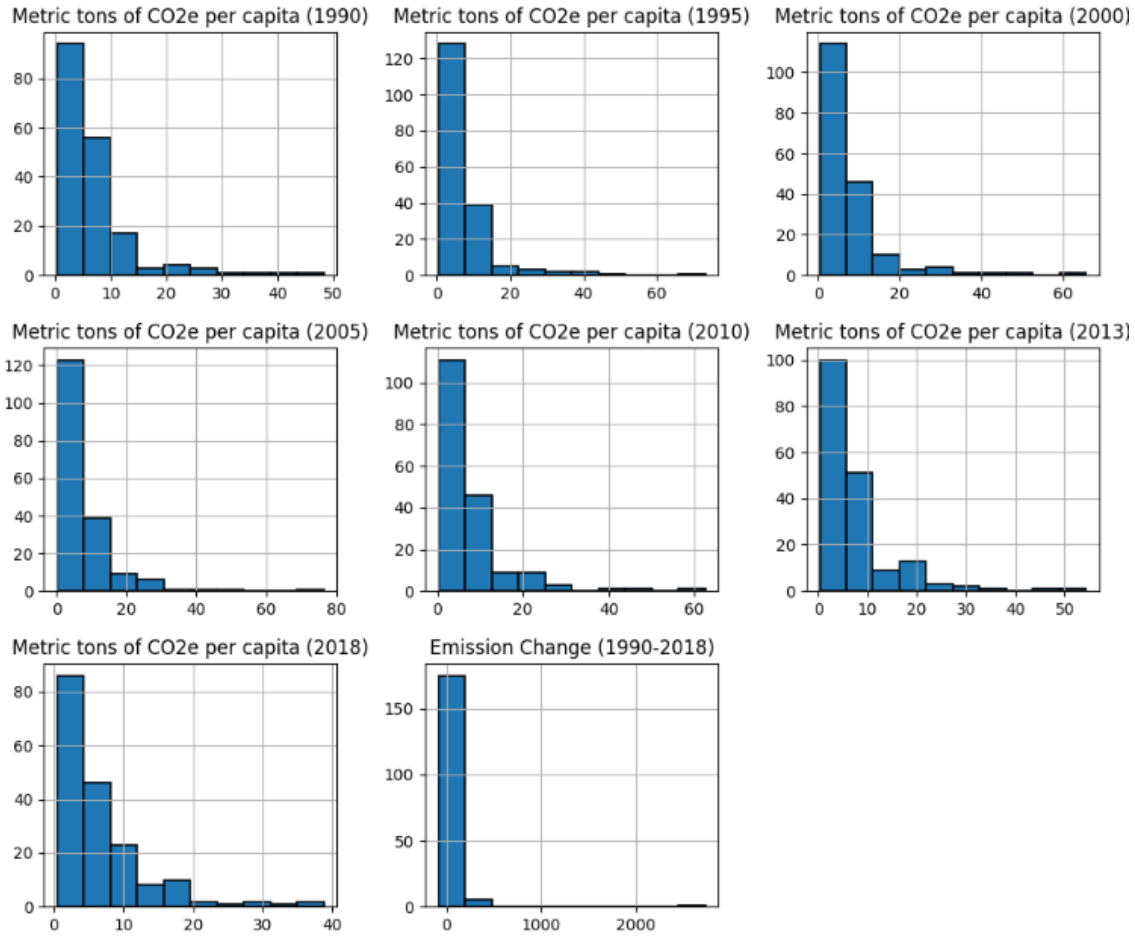
```
# If there are duplicates, we can drop them
if duplicates > 0:
    data.drop_duplicates(inplace=True)

# If any duplicates are found, this code will remove them to ensure data consistency.
```

중복된 행을 확인하고 있다면 drop하는 코드(현재 데이터에서는 중복된 행이 없다)

Part 3 》 # 데이터 분석 –시각화

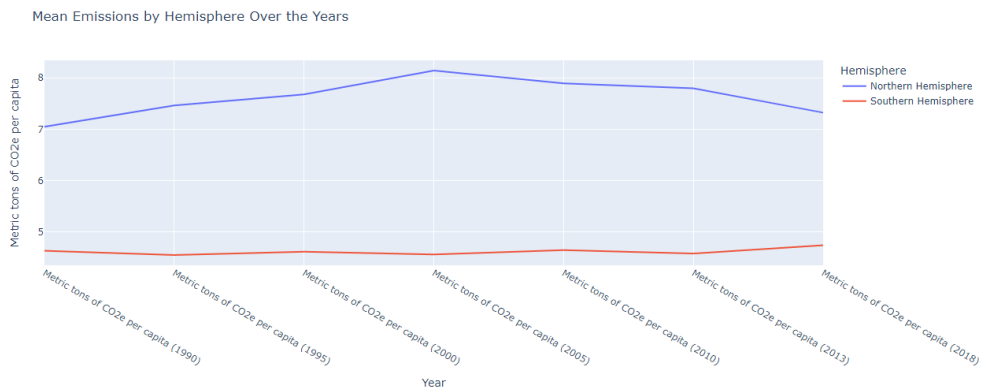
Hist로 시각화

```
data.hist(edgecolor='black', linewidth=1.2)
fig = plt.gcf()
fig.set_size_inches(12,10)
plt.show()
```



북반구와 남반구의 차이를 시각화

```
import plotly.express as px

# Select only numeric columns but keep 'Hemisphere'
numeric_data = data.select_dtypes(include=[np.number])
numeric_data['Hemisphere'] = data['Hemisphere']

# Grouping data by hemisphere and calculating the mean emissions for each year
hemisphere_grouped = numeric_data.groupby('Hemisphere').mean()

# Drop the 'Emission Change (1990-2018)' column
hemisphere_grouped = hemisphere_grouped.drop(['Emission Change (1990-2018)'], axis=1)

# Transpose the data for plotting
hemisphere_grouped_T = hemisphere_grouped.transpose().reset_index()
hemisphere_grouped_T = hemisphere_grouped_T.rename(columns={"index": "Year"})

# Melt the data for plotly express
melted_data = hemisphere_grouped_T.melt(id_vars=["Year"],
                                        value_vars=hemisphere_grouped_T.columns[1:],
                                        var_name="Hemisphere",
                                        value_name="Emissions")

# Plotting using plotly express
fig = px.line(melted_data, x="Year", y="Emissions", color="Hemisphere",
              title="Mean Emissions by Hemisphere Over the Years",
              labels={"Emissions": "Metric tons of CO2e per capita"})

fig.show()
```



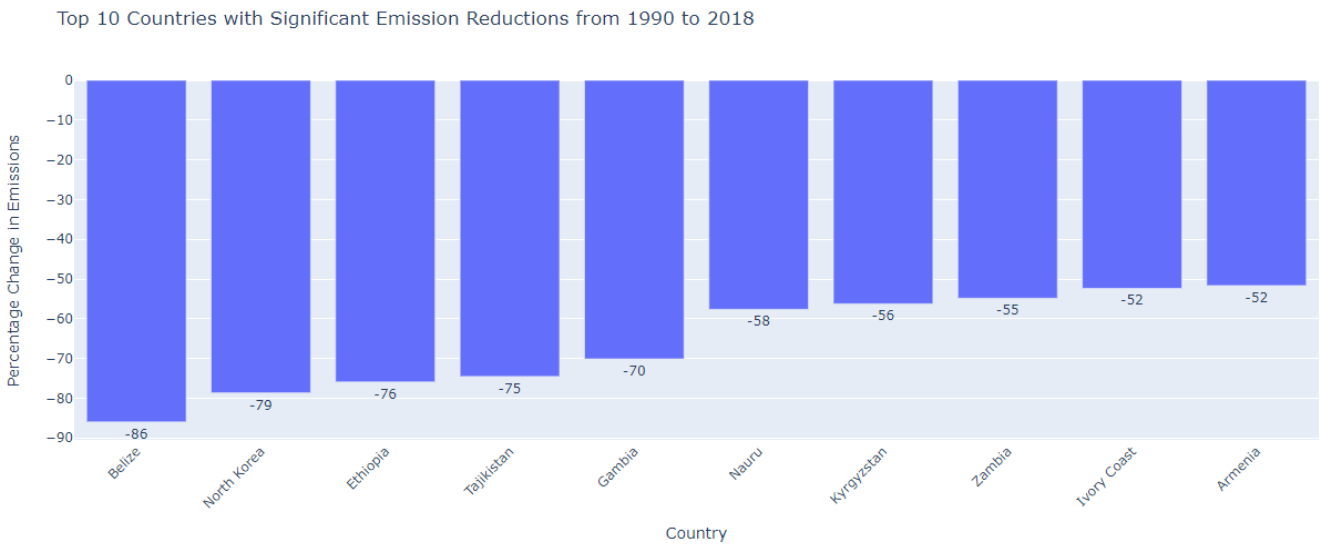
Part 3 》 데이터 분석 –시각화

반구와 배출량 사이의 상관관계가 있는지 보았지만 거의 없다는 결과

```
# Converting Hemisphere to numeric values for correlation analysis
data['Hemisphere_numeric'] = data['Hemisphere'].apply(lambda x: 1 if x == 'Northern Hemisphere' else 0)

# Calculating correlation
correlation = data['Hemisphere_numeric'].corr(data['Metric tons of CO2e per capita (2018)'])

print(f"Correlation between Hemisphere and Emission levels in 2018: {correlation}")
```

```
Correlation between Hemisphere and Emission levels in 2018: 0.15938254637861624
```

```
# Sorting countries based on the 'Emission Change (1990-2018)' column
top_reduced_countries = data.sort_values(by='Emission Change (1990-2018)').head(10)

# Plotting using plotly express
fig = px.bar(top_reduced_countries,
             x='Country',
             y='Emission Change (1990-2018)',
             title='Top 10 Countries with Significant Emission Reductions from 1990 to 2018',
             labels={'Emission Change (1990-2018)': 'Percentage Change in Emissions', 'Country': 'Country'},
             text='Emission Change (1990-2018)')

fig.update_traces(texttemplate='%{text:.2s}', textposition='outside')
fig.update_layout(xaxis_tickangle=-45)

fig.show()
```



1990–2018 사이에 탄소 배출량이 줄어든 국가 TOP 10

Part 3 》 데이터 분석 –모델 학습

```
def split_4_parts(data, input_cols, target):
    # 학습용(문제, 정답), 테스트용(문제, 정답)으로 데이터 나누기
    train, test = train_test_split(data, train_size = 0.8)
    train_country, test_country = train['Country'], test['Country']

    # 학습용 문제와 정답
    a = train[input_cols]
    b = train[target]

    # 시험 문제와 정답
    c = test[input_cols]
    d = test[target]

    return a, b, c, d, train_country, test_country
```

1. 모델에게 데이터를 학습시키기 위해 학습용 데이터와 테스트용 데이터를 분리
   (분리 비율은 8 : 2)
2. 학습 결과 과정에서 해당 나라를 확인하고 싶지만 학습을 시키려면
   정수/실수형으로 바꿔야 하기 때문에 따로 변수에 넣었습니다.
3. 학습용 데이터는 1990-2013년까지의 탄소 배출량,
   정답용 데이터는 2018년도의 탄소 배출량
4. 나눠진 비율 확인

```
a,b,c,d, train_country, test_country = split_4_parts(data, ['Metric tons of CO2e per capita (1990)',
            'Metric tons of CO2e per capita (1995)', 'Metric tons of CO2e per capita (2000)',
            'Metric tons of CO2e per capita (2005)', 'Metric tons of CO2e per capita (2010)',
            'Metric tons of CO2e per capita (2013)'], ['Metric tons of CO2e per capita (2018)'])
```

```
print(a.shape, b.shape, c.shape, d.shape)
```

```
(144, 6) (144, 1) (37, 6) (37, 1)
```

Part 3 》 데이터 분석 –모델 학습

```
from sklearn import linear_model

gildong = LinearRegression()
gildong.fit(a, b)
```

선형 회귀 모형을 이용하여 학습 후 예측 시작

```
▾ LinearRegression
LinearRegression()
```

```
score = gildong.score(c, d)
print(format(score,'.3f'))
```

```
0.866
```

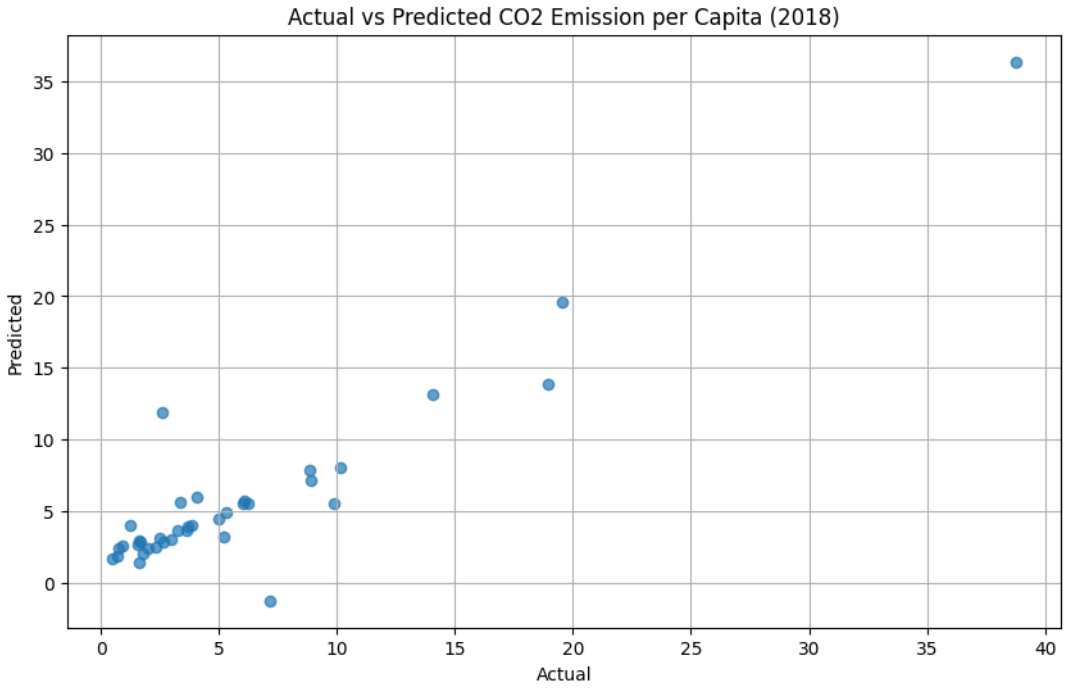
```
predicted = gildong.predict(c)
values_list = test_country.values
horizontal_stacked_array = np.vstack([values_list,predicted[:,0]])
print(horizontal_stacked_array)
```

분리된 나라와 모델이 예측한 탄소 배출량

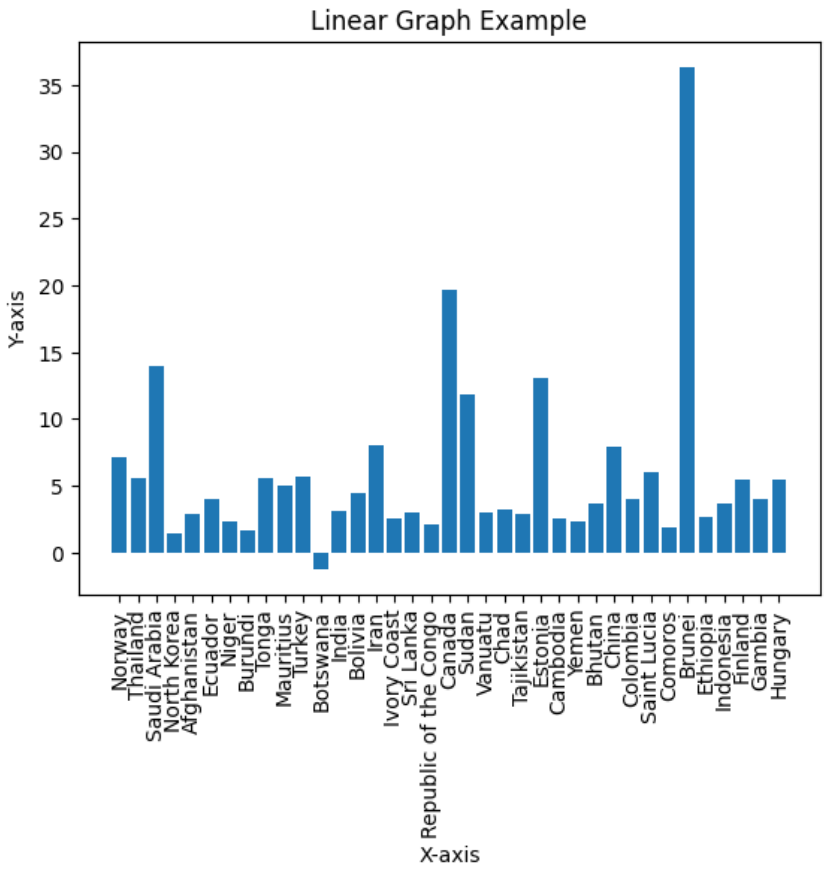
```
[['Norway' 'Thailand' 'Saudi Arabia' 'North Korea' 'Afghanistan'
  'Ecuador' 'Niger' 'Burundi' 'Tonga' 'Mauritius' 'Turkey' 'Botswana'
  'India' 'Bolivia' 'Iran' 'Ivory Coast' 'Sri Lanka'
  'Republic of the Congo' 'Canada' 'Sudan' 'Vanuatu' 'Chad' 'Tajikistan'
  'Estonia' 'Cambodia' 'Yemen' 'Bhutan' 'China' 'Colombia' 'Saint Lucia'
  'Comoros' 'Brunei' 'Ethiopia' 'Indonesia' 'Finland' 'Gambia' 'Hungary']
 [7.120152041528305 5.53255146735037 13.915303455727692
  1.4639088516994572 2.8606633223974334 4.0147592239638525
  2.3811210364303284 1.6628654327997148 5.5993156965697946
  4.964284042726323 5.7036464327629055 -1.2233821166905763
  3.15491327973913 4.478793635117365 8.062352623442058 2.5828709895412434
  3.00124327064501 2.065220566314884 19.610899867155986
  11.868151979901386 3.0243897480826236 3.204515631492897
  2.8624145380986574 13.113613972360358 2.505844825436321
  2.3766632430014467 3.6759297808210993 7.897349352809673
  3.9582580176725464 6.023469620641162 1.8602488092428693
  36.351796623787486 2.6770050763471973 3.642493720634153
  5.508794048517652 3.9879741402656914 5.519116814735696]]
```

Part 3 》 데이터 분석 –모델 학습

```
plt.figure(figsize=(10, 6))
plt.scatter(d, predicted, alpha=0.7)
plt.title('Actual vs Predicted CO2 Emission per Capita (2018)')
plt.xlabel('Actual')
plt.ylabel('Predicted')
plt.grid(True)
plt.show()
```



정확도에 대해 그래프로 나타낸 모습




```
x = values_list
y = predicted[:,0]

# 선형 그래프 그리기
plt.bar(x, y)

# 그래프에 제목과 라벨 추가
plt.title('Linear Graph Example')
plt.xlabel('X-axis')
plt.ylabel('Y-axis')

plt.xticks(rotation=90)

# 그래프 보여주기
plt.show()
```

모델이 예측한 결과를 나라별 그래프로 나타낸 모습

Part 3 》 # 데이터 분석 –모델 학습

K-NN 알고리즘

```
from sklearn.neighbors import KNeighborsRegressor

babo = KNeighborsRegressor(n_neighbors=10)
babo.fit(a, b)
score = babo.score(c, d)
print(format(score,'.3f'))
```

```
0.807
```

랜덤포레스트 알고리즘

```
cheolsu = RandomForestRegressor(n_estimators=28,random_state=0)
cheolsu.fit(a, b)
score = cheolsu.score(c, d)
print(format(score,'.3f'))

predicted = youngja.predict(c)
print(predicted, '\n', predicted.shape)
```

```
0.926
[ 8.48  2.59 16.48  4.54  1.51  3.55  1.55  1.91  3.2   4.31  5.85  5.41
  2.77  4.31  8.02  1.55  2.17  1.87 15.43  6.46  3.52  2.34  1.51 15.
  1.87  1.53  2.18  5.61  3.68  5.6   0.5  35.89  1.53  3.52  9.72  3.6
  5.41]
```

```
youngja = DecisionTreeRegressor(random_state = 0)
youngja.fit(a,b)
score = youngja.score(c, d)
print(format(score,'.3f'))

predicted = youngja.predict(c)
print(predicted, '\n', predicted.shape)
```

```
0.941
[ 8.48  2.59 16.48  4.54  1.51  3.55  1.55  1.91  3.2   4.31  5.85  5.41
  2.77  4.31  8.02  1.55  2.17  1.87 15.43  6.46  3.52  2.34  1.51 15.
  1.87  1.53  2.18  5.61  3.68  5.6   0.5  35.89  1.53  3.52  9.72  3.6
  5.41]
```

결정트리 알고리즘

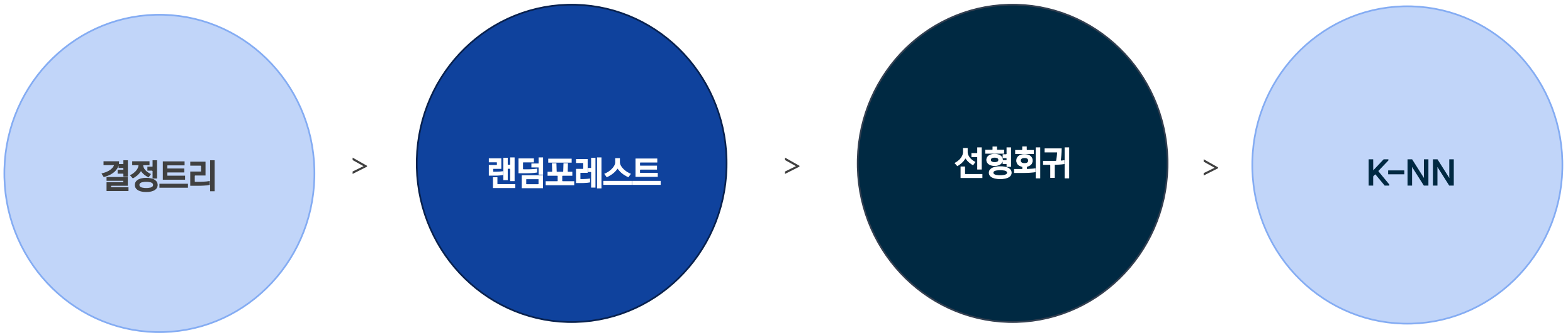
```
score = gildong.score(c, d)
print(format(score,'.3f'))
```

```
0.866
```

선형회귀 알고리즘

Part 3 》 # 데이터 분석 –모델 학습

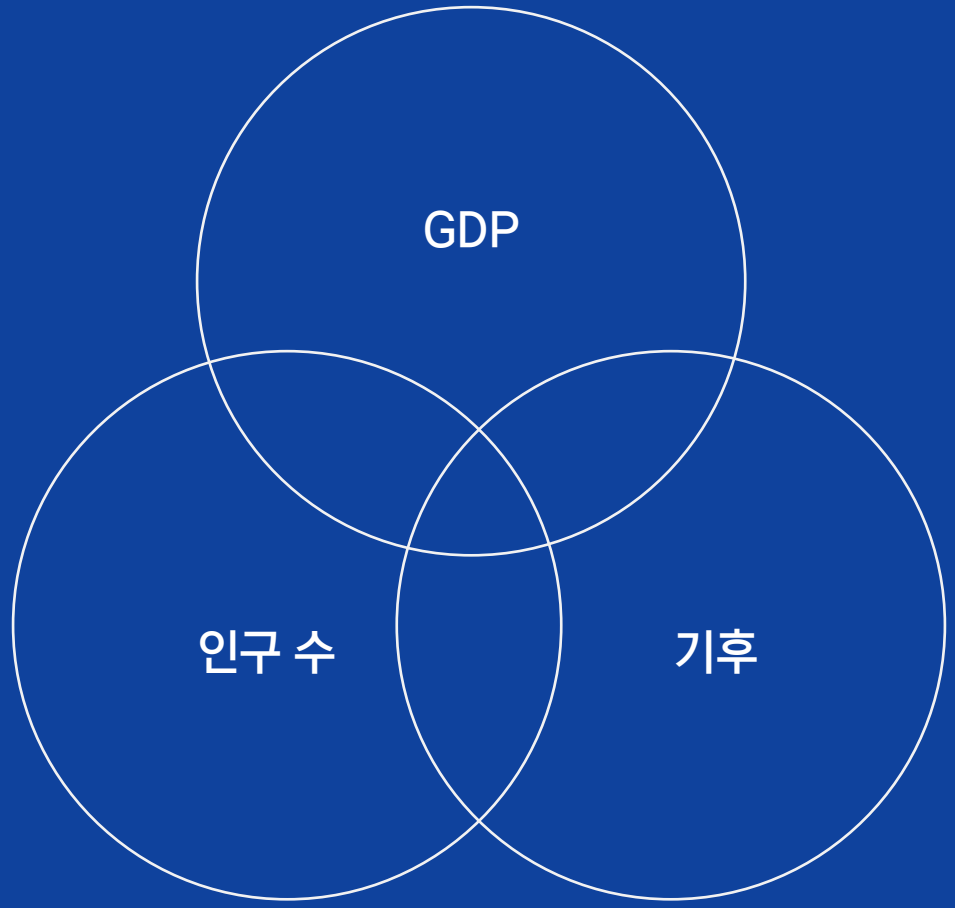
모델 정확도



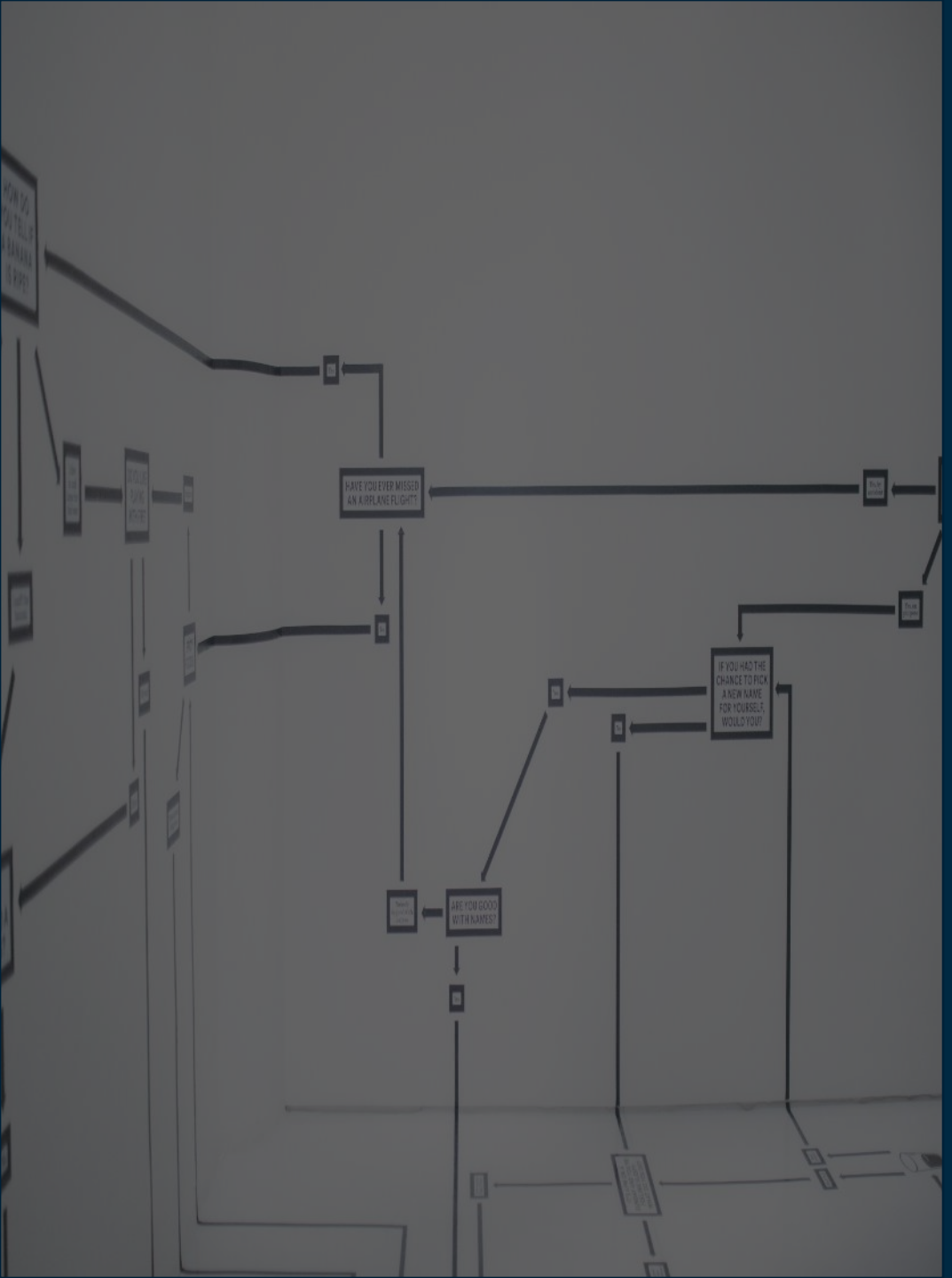
4

결론

Part 4 》 # 결론



나라 별, 대륙 별, 반구 별 탄소 배출량을 예측 가능하지만
1인 당 탄소배출량 데이터 하나만으로는 원인/결과 등과 정확한 예측, 분석이 불가능하다



데이터를 분석하고 여러 모델을 학습하면서 주제에 대한 생각을
다시 해보고 인공지능에 대해 더 많이 알 수 있는 유익한 시간이었습니다.

다만, 데이터셋을 결정할 때 조금 더 신중히 결정했더라면 더 좋은 결과를
얻을 수 있었을 것 같아 아쉬운 마음이 들었습니다.

이번 경험을 양분 삼아 다음에는 관련 데이터셋을 더 추가하고 공부하여
좋은 결과를 얻고 싶습니다.

참고 문헌

IPCC 제6차 평가보고서 종합보고서 기반, 기후기술 대응 시사점 : 탄소중립 10대 핵심기술을 중심으로 – 오채운, 송예원, 김태호

탄소배출권과 탄소 시장 – 안승광

감사합니다