데이터 시각화 ¶

In [1]:
```
!pip install folium requests
```

```
Requirement already satisfied: folium in /usr/local/lib/python3.10/dist-pa
ckages (0.17.0)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-
packages (2.32.3)
Requirement already satisfied: branca>=0.6.0 in /usr/local/lib/python3.10/
dist-packages (from folium) (0.7.2)
Requirement already satisfied: jinja2>=2.9 in /usr/local/lib/python3.10/di
st-packages (from folium) (3.1.4)
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-pac
kages (from folium) (1.26.4)
Requirement already satisfied: xyzservices in /usr/local/lib/python3.10/di
st-packages (from folium) (2024.6.0)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/
python3.10/dist-packages (from requests) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/d
ist-packages (from requests) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python
3.10/dist-packages (from requests) (2.0.7)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python
3.10/dist-packages (from requests) (2024.7.4)
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.1
0/dist-packages (from jinja2>=2.9->folium) (2.1.5)
```

In [2]:

```
import folium
import random
from IPython.display import display

# Folium 라이브러리 설치 (이미 설치되어 있지 않은 경우)
!pip install folium

# 날씨 데이터를 표시할 도시 목록
cities = [
    {"name": "서울", "lat": 37.5665, "lon": 126.9780},
    {"name": "부산", "lat": 35.1796, "lon": 129.0756},
    {"name": "대구", "lat": 35.8714, "lon": 128.6014},
    {"name": "인천", "lat": 37.4563, "lon": 126.7052},
    {"name": "광주", "lat": 35.1595, "lon": 126.8526},
    {"name": "대전", "lat": 36.3504, "lon": 127.3845},
    {"name": "울산", "lat": 35.5384, "lon": 129.3114},
    {"name": "세종", "lat": 36.4800, "lon": 127.2890},
    {"name": "제주", "lat": 33.4996, "lon": 126.5312},
]

# 날씨 상태 목록
weather_conditions = ["맑음", "흐림", "비", "눈", "안개"]

# 지도 생성
m = folium.Map(location=[36.5, 127.5], zoom_start=7)

# 각 도시의 가상 날씨 데이터 생성 및 마커 추가
for city in cities:
    # 가상의 온도 (10°C ~ 30°C)
    temp = round(random.uniform(10, 30), 1)
    # 무작위 날씨 상태 선택
    weather = random.choice(weather_conditions)

    # 날씨에 따른 아이콘 선택
    if weather == "맑음":
        icon = "sun"
    elif weather == "흐림":
        icon = "cloud"
    elif weather == "비":
        icon = "rain"
    elif weather == "눈":
        icon = "snow"
    else:
        icon = "fog"

    # 마커 및 팝업 생성
    folium.Marker(
        location=[city['lat'], city['lon']],
        popup=f"{city['name']}: {temp}° C, {weather}",
        tooltip=city['name'],
        icon=folium.Icon(color="red", icon=icon)
    ).add_to(m)

# 지도 표시
display(m)
```

```
Requirement already satisfied: folium in /usr/local/lib/python3.10/dist-pa
ckages (0.17.0)
Requirement already satisfied: branca>=0.6.0 in /usr/local/lib/python3.10/
dist-packages (from folium) (0.7.2)
Requirement already satisfied: jinja2>=2.9 in /usr/local/lib/python3.10/di
st-packages (from folium) (3.1.4)
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-pac
kages (from folium) (1.26.4)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-
packages (from folium) (2.32.3)
Requirement already satisfied: xyzservices in /usr/local/lib/python3.10/di
st-packages (from folium) (2024.6.0)
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.1
0/dist-packages (from jinja2>=2.9->folium) (2.1.5)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/
python3.10/dist-packages (from requests->folium) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/d
ist-packages (from requests->folium) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python
3.10/dist-packages (from requests->folium) (2.0.7)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python
3.10/dist-packages (from requests->folium) (2024.7.4)
```

Make this Notebook Trusted to load map: File -> Trust Notebook

이미지 처리 및 컴퓨터 비전 도구

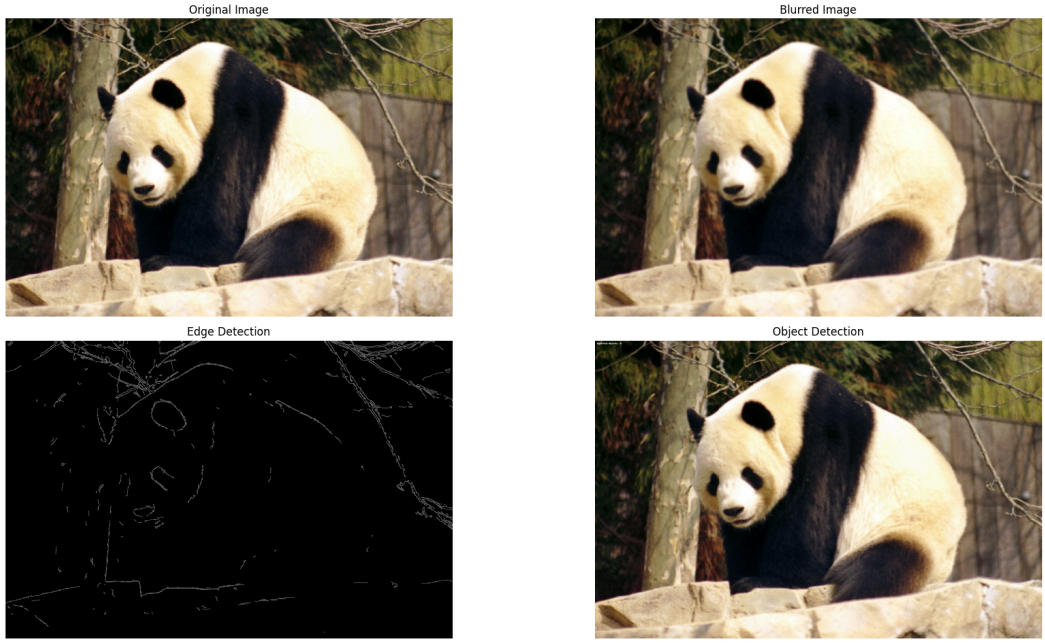
In [3]:
```
!pip install opencv-python-headless pillow numpy matplotlib
```

```
Requirement already satisfied: opencv-python-headless in /usr/local/lib/py
thon3.10/dist-packages (4.10.0.84)
Requirement already satisfied: pillow in /usr/local/lib/python3.10/dist-pa
ckages (9.4.0)
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-pac
kages (1.26.4)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dis
t-packages (3.7.1)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.
10/dist-packages (from matplotlib) (1.2.1)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/d
ist-packages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python
3.10/dist-packages (from matplotlib) (4.53.1)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python
3.10/dist-packages (from matplotlib) (1.4.5)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.1
0/dist-packages (from matplotlib) (24.1)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.
10/dist-packages (from matplotlib) (3.1.2)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/pyth
on3.10/dist-packages (from matplotlib) (2.8.2)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-
packages (from python-dateutil>=2.7->matplotlib) (1.16.0)
```

In [4]:

```
import cv2
import numpy as np
from PIL import Image, ImageDraw, ImageFont
import matplotlib.pyplot as plt
import urllib.request

# 이미지 다운로드 함수
def download_image(url, filename):
    urllib.request.urlretrieve(url, filename)

# 이미지 URL (공개 도메인 이미지)
image_url = "https://upload.wikimedia.org/wikipedia/commons/3/3c/Giant_Panda
image_filename = "panda.jpg"

# 이미지 다운로드
download_image(image_url, image_filename)

# 이미지 로드
image = cv2.imread(image_filename)
image_rgb = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)

# 1. 이미지 필터 적용 (예: 가우시안 블러)
blurred = cv2.GaussianBlur(image_rgb, (15, 15), 0)

# 2. 엣지 감지
edges = cv2.Canny(image_rgb, 100, 200)

# 3. 객체 감지 (여기서는 얼굴 감지를 예로 들겠습니다)
face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_fr
gray = cv2.cvtColor(image_rgb, cv2.COLOR_RGB2GRAY)
faces = face_cascade.detectMultiScale(gray, 1.3, 5)

# 결과 이미지 생성
result = image_rgb.copy()
for (x, y, w, h) in faces:
    cv2.rectangle(result, (x, y), (x+w, y+h), (255, 0, 0), 2)

# Pillow를 사용하여 결과 표시
pil_image = Image.fromarray(result)
draw = ImageDraw.Draw(pil_image)
font = ImageFont.load_default()
draw.text((10, 10), "Detected Objects: " + str(len(faces)), font=font, fill=

# 결과 표시
plt.figure(figsize=(20, 10))

plt.subplot(2, 2, 1)
plt.title("Original Image")
plt.imshow(image_rgb)
plt.axis('off')

plt.subplot(2, 2, 2)
plt.title("Blurred Image")
plt.imshow(blurred)
plt.axis('off')

plt.subplot(2, 2, 3)
plt.title("Edge Detection")
plt.imshow(edges, cmap='gray')
plt.axis('off')
```

```
plt.subplot(2, 2, 4)
plt.title("Object Detection")
plt.imshow(pil_image)
plt.axis('off')

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



- 판다 이미지를 다운로드합니다.
- 원본 이미지, 블러 처리된 이미지, 엣지 감지 결과를 생성합니다.
- 얼굴 감지를 시도합니다 (판다 이미지에서는 사람 얼굴이 감지되지 않을 것입니다).
- 결과를 4개의 서브플롯으로 표시합니다.

In [5]:

```
import cv2
import numpy as np
from PIL import Image, ImageDraw, ImageFont
import matplotlib.pyplot as plt
import urllib.request

# 이미지 다운로드 함수
def download_image(url, filename):
    urllib.request.urlretrieve(url, filename)

# 이미지 URL (공개 도메인 이미지)
image_url = "https://upload.wikimedia.org/wikipedia/commons/3/37/Dagestani_m
image_filename = "people.jpg"

# 이미지 다운로드
download_image(image_url, image_filename)

# 이미지 로드
image = cv2.imread(image_filename)
image_rgb = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)

# 1. 이미지 필터 적용 (예: 가우시안 블러)
blurred = cv2.GaussianBlur(image_rgb, (15, 15), 0)

# 2. 엣지 감지
edges = cv2.Canny(image_rgb, 100, 200)

# 3. 얼굴 감지
face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_fr
gray = cv2.cvtColor(image_rgb, cv2.COLOR_RGB2GRAY)
faces = face_cascade.detectMultiScale(gray, 1.3, 5)

# 결과 이미지 생성
result = image_rgb.copy()
for (x, y, w, h) in faces:
    cv2.rectangle(result, (x, y), (x+w, y+h), (255, 0, 0), 2)

# Pillow를 사용하여 결과 표시
pil_image = Image.fromarray(result)
draw = ImageDraw.Draw(pil_image)
font = ImageFont.load_default()
draw.text((10, 10), f"Detected Faces: {len(faces)}", font=font, fill=(255, 2

# 결과 표시
plt.figure(figsize=(20, 15))

plt.subplot(2, 2, 1)
plt.title("Original Image")
plt.imshow(image_rgb)
plt.axis('off')

plt.subplot(2, 2, 2)
plt.title("Blurred Image")
plt.imshow(blurred)
plt.axis('off')

plt.subplot(2, 2, 3)
plt.title("Edge Detection")
plt.imshow(edges, cmap='gray')
plt.axis('off')
```

```
plt.subplot(2, 2, 4)
plt.title("Face Detection")
plt.imshow(pil_image)
plt.axis('off')

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

코드 작업 수행

- 두 명의 사람이 있는 이미지를 다운로드합니다.
- 원본 이미지, 블러 처리된 이미지, 엣지 감지 결과를 생성합니다.
- 얼굴 감지를 수행하고 감지된 얼굴 주위에 사각형을 그립니다.
- 결과를 4개의 서브플롯으로 표시합니다.

성능 개선

남자의 얼굴이 인식되지 않은 주요 원인

- 얼굴 각도: 남자의 얼굴이 카메라를 정면으로 바라보고 있지 않을 수 있습니다. 옆모습이나 각도가 있는 얼굴은 인식하기 어려울 수 있습니다.
- 조명 조건: 그림자나 불균형한 조명으로 인해 얼굴의 특징이 잘 보이지 않을 수 있습니다.
- 얼굴 가림: 수염, 모자, 안경 등이 얼굴의 일부를 가리고 있을 수 있습니다.

- 해상도 및 화질: 이미지의 해상도가 낮거나 화질이 좋지 않으면 얼굴 감지가 어려울 수 있습니다.
- 알고리즘의 한계: 사용된 Haar Cascade 분류기는 간단하고 빠르지만, 복잡한 상황에서는 정확도가 떨어질 수 있습니다.

In [6]:

```
import cv2
import numpy as np
from PIL import Image, ImageDraw, ImageFont
import matplotlib.pyplot as plt
import urllib.request

def download_image(url, filename):
    urllib.request.urlretrieve(url, filename)

image_url = "https://upload.wikimedia.org/wikipedia/commons/3/37/Dagestani_m
image_filename = "people.jpg"

download_image(image_url, image_filename)

image = cv2.imread(image_filename)
image_rgb = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)

# 이미지 전처리: 대비 향상
clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(8,8))
gray = cv2.cvtColor(image_rgb, cv2.COLOR_RGB2GRAY)
gray = clahe.apply(gray)

blurred = cv2.GaussianBlur(image_rgb, (15, 15), 0)
edges = cv2.Canny(image_rgb, 100, 200)

face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_fr
faces = face_cascade.detectMultiScale(gray, scaleFactor=1.1, minNeighbors=5

result = image_rgb.copy()
for (x, y, w, h) in faces:
    cv2.rectangle(result, (x, y), (x+w, y+h), (255, 0, 0), 2)

pil_image = Image.fromarray(result)
draw = ImageDraw.Draw(pil_image)
font = ImageFont.load_default()
draw.text((10, 10), f"Detected Faces: {len(faces)}", font=font, fill=(255, 2

plt.figure(figsize=(20, 15))

plt.subplot(2, 2, 1)
plt.title("Original Image")
plt.imshow(image_rgb)
plt.axis('off')

plt.subplot(2, 2, 2)
plt.title("Blurred Image")
plt.imshow(blurred)
plt.axis('off')

plt.subplot(2, 2, 3)
plt.title("Edge Detection")
plt.imshow(edges, cmap='gray')
plt.axis('off')

plt.subplot(2, 2, 4)
plt.title("Face Detection")
plt.imshow(pil_image)
plt.axis('off')

plt.tight_layout()
```

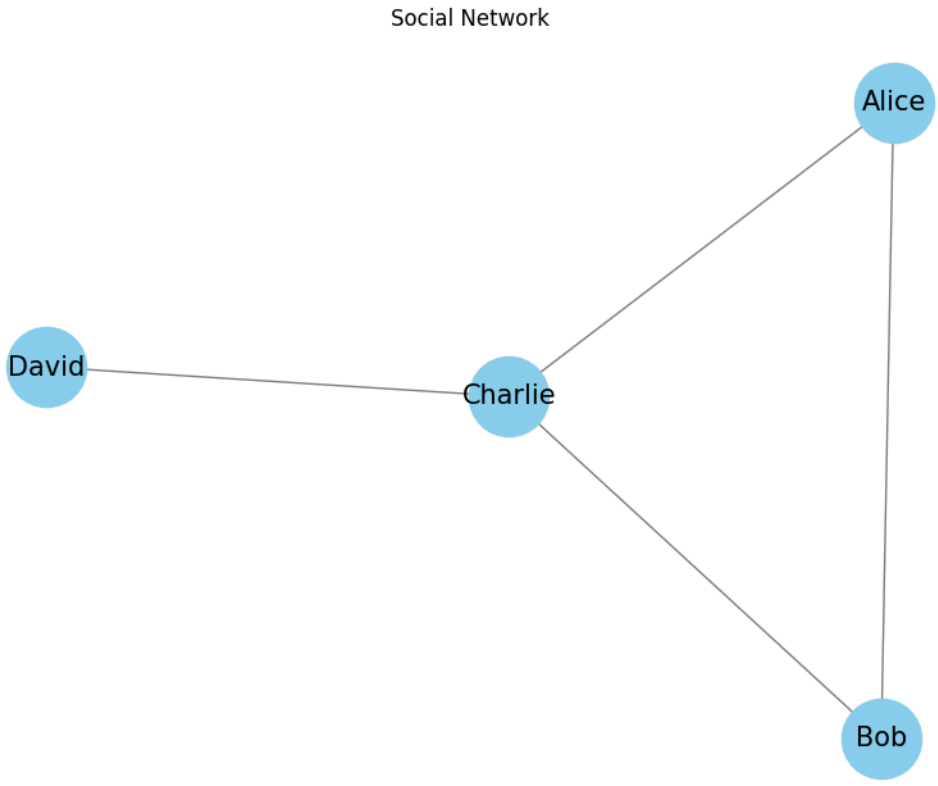
```
plt.show()
```

- 이 코드에서는 CLAHE(Contrast Limited Adaptive Histogram Equalization)를 사용하여 이미지의 대비를 향상시켰고, detectMultiScale 함수의 파라미터를 조정.

네트워크 그래스 생성

In [7]:

```
import networkx as nx
import matplotlib.pyplot as plt

# 빈 그래프 생성
G = nx.Graph()

# 노드 추가
G.add_node("Alice")
G.add_node("Bob")
G.add_node("Charlie")
G.add_node("David")

# 엣지(연결) 추가
G.add_edges_from([("Alice", "Bob"), ("Alice", "Charlie"), ("Bob", "Charlie")

# 그래프 시각화
plt.figure(figsize=(8, 6))
nx.draw(G, with_labels=True, node_color='skyblue', node_size=2000, font_siz
plt.title("Social Network")
plt.show()
```



워드 클라우드 예제

In [16]:
```
!pip install wordcloud matplotlib
```

```
Requirement already satisfied: wordcloud in /usr/local/lib/python3.10/dist
-packages (1.9.3)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dis
t-packages (3.7.1)
Requirement already satisfied: numpy>=1.6.1 in /usr/local/lib/python3.10/d
ist-packages (from wordcloud) (1.26.4)
Requirement already satisfied: pillow in /usr/local/lib/python3.10/dist-pa
ckages (from wordcloud) (9.4.0)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.
10/dist-packages (from matplotlib) (1.2.1)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/d
ist-packages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python
3.10/dist-packages (from matplotlib) (4.53.1)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python
3.10/dist-packages (from matplotlib) (1.4.5)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.1
0/dist-packages (from matplotlib) (24.1)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.
10/dist-packages (from matplotlib) (3.1.2)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/pyth
on3.10/dist-packages (from matplotlib) (2.8.2)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-
packages (from python-dateutil>=2.7->matplotlib) (1.16.0)
```

In [17]:

```
from wordcloud import WordCloud
import matplotlib.pyplot as plt

# 텍스트 데이터
text = """
Python is a great programming language for data analysis and visualization.
It is widely used in data science, machine learning, and artificial intellig
With Python, you can create complex models, process large datasets, and gene
insights through various libraries like pandas, numpy, and matplotlib.
Python is also known for its simplicity and readability, making it a popular
among beginners and experts alike.
"""

# 워드 클라우드 생성
wordcloud = WordCloud(width=800, height=400, background_color='white').gene

# 워드 클라우드 시각화
plt.figure(figsize=(10, 6))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.show()
```

In [ ]: