깃허브 주소: https://github.com/krustingkevin/ai-crypto-2023

유형준, 고범

```
import pandas as pd

# 진행 상황 출력 함수
def progress_callback(progress):
    if progress < 4:
        print(f"현재 계산: {progress}/4 단계 완료")
    elif progress == 4:
        print("현재 계산: 4/4 단계 완료\n파일을 저장중입니다. 실행이 완료될 때 까지
기다려주십시오")
    elif progress == 5:
        print("스크립트 실행이 완료되었습니다.")

# CSV 파일 읽기
df = pd.read_csv('2023-05-10-upbit-BTC-book.csv')

# 이전 10개 price의 평균값 계산
df['price_avg_for_10'] = df['price'].rolling(10, min_periods=1).mean()
progress_callback(1)

# book-delta 계산
df['book-delta'] = df.groupby('type')['quantity'].diff()
progress_callback(2)

# book-imbalance 계산
df['book-imbalance'] = df.groupby('type')['quantity'].transform('sum')
progress_callback(3)

# mid_price 계산
df['mid_price'] = (df['price'].shift(1) + df['price']) / 2
progress_callback(4)

# 필요한 열 선택
selected_columns = ['timestamp', 'mid_price', 'book-imbalance', 'book-delta',
'price_avg_for_10']
new_df = df[selected_columns]

# CSV 파일로 저장
new_df.to_csv('2023-05-10-upbit-btc-feature.csv', index=False)
progress_callback(5)
```

기존 코드

```python
import pandas as pd
import os

# 진행 상황 출력 함수
def progress_callback(progress):
    if progress < 4:
        print(f"현재 계산: {progress}/4 단계 완료")
    elif progress == 4:
        print("현재 계산: 4/4 단계 완료\n파일을 저장중입니다. 실행이 완료될 때 까지 기다려주십시오")
    elif progress == 5:
        print("스크립트 실행이 완료되었습니다.")

# CSV 파일 읽기
df = pd.read_csv('2023-05-10-upbit-BTC-book.csv')

# 이전 10개 price의 평균값 계산
df['price_avg_for_10'] = df['price'].rolling(10, min_periods=1).mean()
progress_callback(1)

# book-delta 계산
df['book-delta'] = df.groupby('type')['quantity'].diff()
progress_callback(2)

# book-imbalance 계산
df['book-imbalance'] = df.groupby('type')['quantity'].transform('sum')
progress_callback(3)

# mid_price 계산
df['mid_price'] = (df['price'].shift(1) + df['price']) / 2
progress_callback(4)

# 필요한 열 선택
selected_columns = ['timestamp', 'mid_price', 'book-imbalance', 'book-delta', 'price_avg_for_10']
new_df = df[selected_columns]

# 결과를 25MB 단위로 분할하여 저장
output_folder = 'output'
os.makedirs(output_folder, exist_ok=True)
max_file_size = 24 * 1024 * 1024  # 25MB
file_number = 1
file_path = f'{output_folder}/2023-05-10-upbit-btc-feature_{file_number}.csv'
chunk_size = 10000  # 8만 개의 행씩 저장

for start in range(0, len(new_df), chunk_size):
    end = start + chunk_size
    chunk = new_df[start:end]
```

```
    if not os.path.exists(file_path):
        with open(file_path, 'w') as f:
            chunk.to_csv(f, index=False)
    elif os.path.getsize(file_path) > max_file_size:
        file_number += 1
        file_path = f'{output_folder}/2023-05-10-upbit-btc-
feature_{file_number}.csv'
        with open(file_path, 'w') as f:
            chunk.to_csv(f, index=False, header=True)
    else:
        with open(file_path, 'a') as f:
            chunk.to_csv(f, index=False, header=False)

progress_callback(5)
```

깃허브의 파일 25MB 제한으로 인한 코드