5주차: Pandas + Matplotlib

데이터 분석 & 시각화

Python 백엔드 심화반

오늘의 학습 목표

- 🐼 **Pandas**로 데이터를 자유자재로 다루기
- 📊 **Matplotlib**으로 인사이트 있는 시각화 만들기
- 🔗 4주차 크롤링 데이터를 분석 가능한 형태로 가공
- 📈 실무에서 바로 쓸 수 있는 차트 제작

목차

Part 1: Pandas 데이터 처리

1-1. 데이터 불러오기

```
import pandas as pd

# CSV 파일 읽기
df = pd.read_csv('sales_data.csv', encoding='utf-8')

# JSON 파일 읽기
df_json = pd.read_json('products.json')

# Excel 파일 읽기
df_excel = pd.read_excel('report.xlsx', sheet_name='Sheet1')

# 데이터 확인
print(df.head())  # 상위 5개 행
print(df.shape)   # (행, 열) 개수
```

1-2. 데이터 저장하기

```
# CSV로 저장 (인덱스 제외)
df.to_csv('output.csv', index=False, encoding='utf-8-sig')

# Excel로 저장
df.to_excel('output.xlsx', index=False)

# JSON으로 저장
df.to_json('output.json', orient='records', force_ascii=False)
```

💡 **Tip**: `encoding='utf-8-sig'`를 사용하면 Excel에서 한글이 깨지지 않습니다!

1-3. 데이터 탐색하기

```
# 데이터 기본 정보
df.info()          # 컬럼 타입, null 값 확인
df.describe()      # 수치형 데이터 통계
df.value_counts() # 값별 개수 세기

# 결측치 확인
df.isnull().sum()  # 컬럼별 결측치 개수

# 중복 확인
df.duplicated().sum()  # 중복된 행 개수
```

1-4. 데이터 정제 - 결측치 처리

```
# 결측치 확인
print(df.isnull().sum())

# 결측치 제거
df_clean = df.dropna()  # 결측치가 있는 행 전체 제거
df_clean = df.dropna(subset=['price'])  # 특정 컬럼만 확인

# 결측치 채우기
df['price'].fillna(0, inplace=True)  # 0으로 채우기
df['category'].fillna('기타', inplace=True)  # 문자열로 채우기
df['rating'].fillna(df['rating'].mean(), inplace=True)  # 평균값으로
```

1-5. 데이터 정제 - 타입 변환

```
# 문자열 → 숫자 변환
df['price'] = df['price'].str.replace(',', '').astype(int)
df['price'] = pd.to_numeric(df['price'], errors='coerce')

# 날짜 변환
df['date'] = pd.to_datetime(df['date'])

# 카테고리 타입 변환 (메모리 효율)
df['category'] = df['category'].astype('category')
```

1-6. 조건부 필터링

```
# 단일 조건
expensive = df[df['price'] > 50000]

# 복수 조건 (AND)
filtered = df[(df['price'] > 30000) & (df['rating'] >= 4.0)]

# 복수 조건 (OR)
popular = df[(df['views'] > 1000) | (df['rating'] >= 4.5)]

# isin() 활용
brands = ['삼성', 'LG', '애플']
brand_products = df[df['brand'].isin(brands)]

# 문자열 포함 검색
smartphone = df[df['name'].str.contains('스마트폰')]
```

1-7. 데이터 변환 - GroupBy

```
# 브랜드별 평균 가격
brand_avg = df.groupby('brand')['price'].mean()

# 여러 집계 함수 적용
brand_stats = df.groupby('brand').agg({
    'price': ['mean', 'min', 'max', 'count'],
    'rating': 'mean'
})

# 여러 컬럼으로 그룹화
category_brand = df.groupby(['category', 'brand'])['price'].mean()
```

1-8. 데이터 변환 - Pivot Table

```
# 피벗 테이블 생성
pivot = pd.pivot_table(
    data=df,
    index='category',       # 행
    columns='brand',        # 열
    values='price',         # 값
    aggfunc='mean',         # 집계 함수
    fill_value=0            # 빈 값 채우기
)

print(pivot)
```

1-9. 새로운 컬럼 생성

```
# 직접 계산
df['price_per_rating'] = df['price'] / df['rating']

# apply 함수 활용
def price_level(price):
    if price < 30000:
        return '저가'
    elif price < 100000:
        return '중가'
    else:
        return '고가'

df['price_level'] = df['price'].apply(price_level)

# map 함수 활용
brand_country = {'삼성': '한국', 'LG': '한국', '애플': '미국'}
df['country'] = df['brand'].map(brand_country)
```

실습 1: 온라인 쇼핑 데이터 분석

```
# 데이터 로드
df = pd.read_csv('shopping_data.csv')

# 1. 가격 컬럼 정제
df['price'] = df['price'].str.replace(',', '').astype(int)

# 2. 브랜드별 통계
brand_analysis = df.groupby('brand').agg({
    'price': ['mean', 'count'],
    'rating': 'mean',
    'reviews': 'sum'
}).round(2)

# 3. 가격대별 분류
df['price_range'] = pd.cut(df['price'],
                           bins=[0, 30000, 100000, float('inf')],
                           labels=['저가', '중가', '고가'])
```

Part 2: Matplotlib 시각화

2-1. Matplotlib 기초 설정

```
import matplotlib.pyplot as plt
import matplotlib.font_manager as fm

# 한글 폰트 설정 (Windows)
plt.rcParams['font.family'] = 'Malgun Gothic'

# 한글 폰트 설정 (Mac)
# plt.rcParams['font.family'] = 'AppleGothic'

# 마이너스 부호 표시 설정
plt.rcParams['axes.unicode_minus'] = False

# 그래프 크기 기본값 설정
plt.rcParams['figure.figsize'] = (10, 6)
```

2-2. Pandas 직접 시각화

```
# 선 그래프
df.groupby('date')['sales'].sum().plot(kind='line')
plt.title('일별 매출 추이')
plt.show()

# 막대 그래프
df.groupby('brand')['price'].mean().plot(kind='bar')
plt.title('브랜드별 평균 가격')
plt.ylabel('가격 (원)')
plt.show()

# 산점도
df.plot(kind='scatter', x='price', y='rating', alpha=0.5)
plt.title('가격 vs 평점')
plt.show()
```

2-3. 기본 차트 패턴

```
# Figure와 Axes 객체 생성
fig, ax = plt.subplots()

# 선 그래프 with 스타일
ax.plot(dates, values, 'b-', linewidth=2, label='매출')
ax.set_xlabel('날짜')
ax.set_ylabel('매출액 (원)')
ax.set_title('월별 매출 추이')
ax.legend()
ax.grid(True, alpha=0.3)

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

2-4. 막대 그래프 커스터마이징

```
# 데이터 준비
brands = df.groupby('brand')['price'].mean().sort_values(ascending=False)

# 막대 그래프 with 색상
colors = ['#FF6B6B', '#4ECDC4', '#45B7D1', '#FFA07A', '#98D8C8']
bars = plt.bar(brands.index, brands.values, color=colors[:len(brands)])

# 값 표시
for bar in bars:
    height = bar.get_height()
    plt.text(bar.get_x() + bar.get_width()/2., height,
             f'{int(height):,}원',
             ha='center', va='bottom')

plt.title('브랜드별 평균 가격', fontsize=16, fontweight='bold')
plt.xticks(rotation=45)
plt.tight_layout()
```

2-5. 히스토그램과 분포

```
# 히스토그램
plt.figure(figsize=(12, 5))

# 서브플롯 1: 히스토그램
plt.subplot(1, 2, 1)
plt.hist(df['price'], bins=30, color='skyblue', edgecolor='black')
plt.xlabel('가격')
plt.ylabel('빈도')
plt.title('가격 분포')

# 서브플롯 2: 박스플롯
plt.subplot(1, 2, 2)
df.boxplot(column='price', by='category')
plt.title('카테고리별 가격 분포')
plt.suptitle('')  # 기본 제목 제거

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

2-6. 파이 차트

```
# 카테고리별 상품 비율
category_counts = df['category'].value_counts()

# 파이 차트
plt.figure(figsize=(8, 8))
colors = plt.cm.Set3(range(len(category_counts)))
explode = [0.05 if x < 0.1 else 0 for x in category_counts/category_counts.sum()]

plt.pie(category_counts, labels=category_counts.index,
        autopct='%1.1f%%', colors=colors, explode=explode,
        shadow=True, startangle=90)

plt.title('카테고리별 상품 비율', fontsize=16)
plt.show()
```

2-7. 상관관계 히트맵

```
import seaborn as sns

# 상관관계 계산
corr = df[['price', 'rating', 'reviews', 'sales']].corr()

# 히트맵
plt.figure(figsize=(8, 6))
sns.heatmap(corr, annot=True, fmt='.2f', cmap='coolwarm',
            square=True, linewidths=1)
plt.title('변수 간 상관관계', fontsize=16)
plt.show()
```

2-8. 대시보드 스타일 시각화

```
# 4개의 서브플롯으로 대시보드 구성
fig, ((ax1, ax2), (ax3, ax4)) = plt.subplots(2, 2, figsize=(15, 10))

# 1. 일별 매출 추이
daily_sales = df.groupby('date')['sales'].sum()
ax1.plot(daily_sales.index, daily_sales.values, 'b-', linewidth=2)
ax1.set_title('일별 매출 추이')
ax1.grid(True, alpha=0.3)

# 2. 브랜드별 평균 가격
brand_avg = df.groupby('brand')['price'].mean().sort_values()
ax2.barh(brand_avg.index, brand_avg.values)
ax2.set_title('브랜드별 평균 가격')

# 3. 가격 분포
ax3.hist(df['price'], bins=30, color='green', alpha=0.7)
ax3.set_title('가격 분포')

# 4. 평점별 상품 수
rating_counts = df['rating'].value_counts().sort_index()
ax4.bar(rating_counts.index, rating_counts.values, color='orange')
ax4.set_title('평점별 상품 수')

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

실전 프로젝트: 쇼핑몰 데이터 분석

```
# 1. 데이터 로드 및 전처리
df = pd.read_csv('shopping_data.csv')
df['price'] = df['price'].str.replace(',', '').astype(int)
df['date'] = pd.to_datetime(df['date'])

# 2. 분석
top_brands = df.groupby('brand').agg({
    'price': 'mean',
    'rating': 'mean',
    'sales': 'sum'
}).sort_values('sales', ascending=False).head(10)

# 3. 시각화
fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(15, 6))

# 매출 상위 10개 브랜드
top_brands['sales'].plot(kind='bar', ax=ax1, color='skyblue')
ax1.set_title('브랜드별 총 매출 TOP 10')
ax1.set_ylabel('매출액')

# 가격 vs 평점 산점도
scatter = ax2.scatter(df['price'], df['rating'],
                      c=df['sales'], s=50, alpha=0.6, cmap='viridis')
ax2.set_xlabel('가격')
ax2.set_ylabel('평점')
ax2.set_title('가격 vs 평점 (색상: 매출)')
plt.colorbar(scatter, ax=ax2)

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

유용한 팁 모음

🎨 시각화 개선하기

- `plt.style.use('seaborn')` - 깔끔한 스타일 적용
- `alpha=0.7` - 투명도 조절
- `plt.tight_layout()` - 레이아웃 자동 조정

📊 Pandas 성능 향상

- `chunksize` 파라미터로 대용량 파일 처리
- `pd.to_numeric(errors='coerce')` - 안전한 타입 변환
- `.copy()` 사용으로 SettingWithCopyWarning 방지

미션

🎯 미션: 나만의 데이터 대시보드 만들기

1. 제공된 `ecommerce_data.csv` 파일 분석
2. 다음 내용을 포함한 시각화 대시보드 제작:
    - 월별 매출 추이
    - 카테고리별 판매량
    - 시간대별 주문 패턴
    - 고객 등급별 구매 금액 분포
3. 인사이트 3가지 도출하여 마크다운으로 정리

Q&A

궁금한 점이 있으신가요?

📧 **질문은 슬랙 채널로!**

다음 주: Flask/FastAPI로 데이터 API 만들기

참고 자료

📚 추천 학습 자료

- Pandas 공식 문서
- Matplotlib 갤러리
- Python Graph Gallery

수고하셨습니다! 👏

오늘 배운 내용 복습하기

✅ Pandas로 데이터 자유자재로 다루기
✅ Matplotlib으로 인사이트 시각화하기
✅ 실무 활용 가능한 분석 스킬 습득