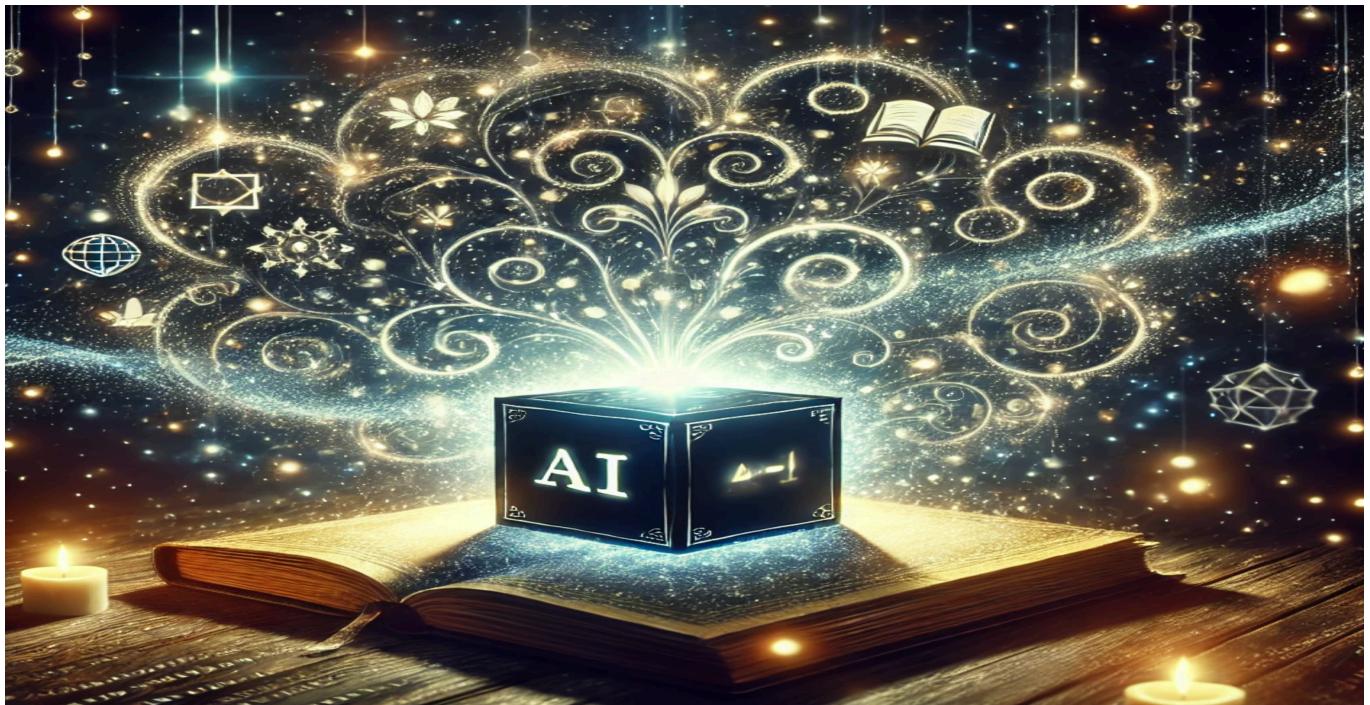


# 🔥 Warm Up 🔥

---

🎩 GPT는 마술이다..?

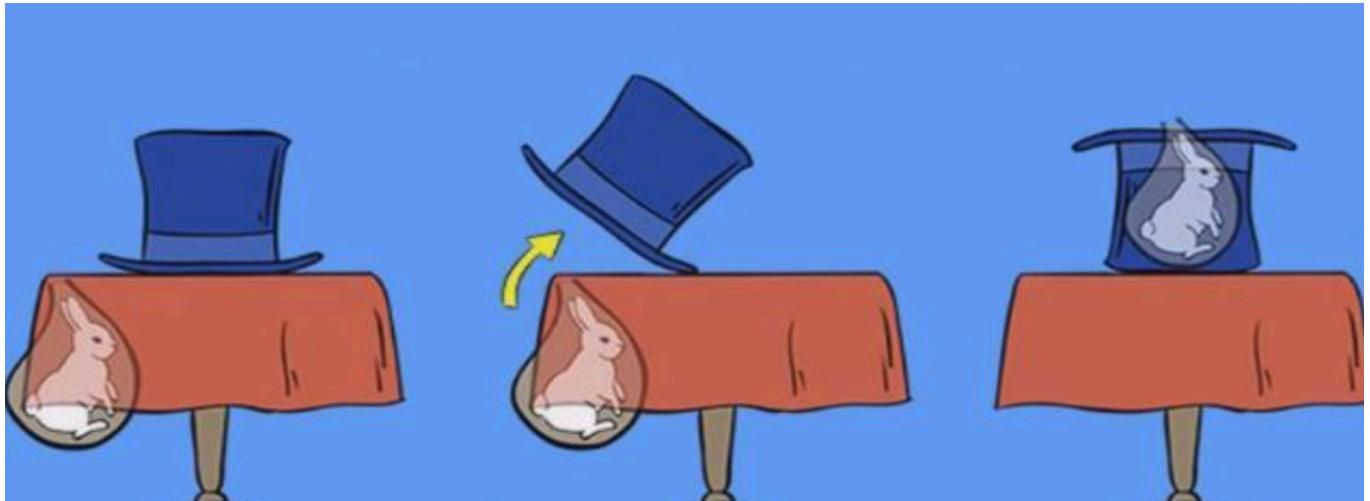


GPT는 마치 마술처럼 우리의 상상력을 현실로 바꿔줍니다. 우리가 단순히 질문을 입력하는 순간, 마치 마술처럼 우리의 생각을 파악하고 아이디어를 구체화하여 해결해줍니다. 이러한 마술은 특정 도메인에 국한되지 않고, 예술, 과학, 비즈니스 등 다양한 분야에서 놀라운 성과를 창출할 수 있습니다.

## ▣ 참고자료

- ▣ 알트먼이 GPT-4o를 "마법 같다"고 말한 이유는

✍️ 마술은 장비와 기술로 이루어진다!



신기한 마술의 이면에는 정교한 도구와 숙련된 마술사의 손기술이 숨어있습니다. 예를 들어, 마술사가 모자에서 토끼를 꺼내는 마술은 다음과 같은 두 가지 요소에 의해 이루어집니다.

1. 특수 제작한 모자를 이용하는 것.

2. 마술사의 재빠른 손동작으로 순식간에 토끼가 든 자루를 모자 안에 넣는 것.

## 💡 GPT로 마술을 부리는 방법

이는 우리가 마술처럼 느끼는 GPT에서도 유사한 장치가 숨겨져 있습니다. 아래 마술과 GPT의 비교 예시를 통해 구체화해보도록 하겠습니다.

1. 특수 제작한 모자 <-> 대규모 언어 모델(Large Language Model, LLM)
2. 재빠른 손동작 <-> 정확한 요청

## ✓ 목표

- 마술과도 같은 GPT의 트릭을 알아내고 다양하게 사용해보기
- 데이터 분석을 위한 python 지식 배우기
- 고급 데이터 분석 경험 해보기
- 데이터 분석의 전반적인 지식 배우기

## 📌 이렇게 만들었어요!



chatGPT가 마법과도 같은 블랙박스임을 나타내는 그림을 그려줄래

여기 ChatGPT를 신비로운 블랙박스로 표현한 그림이 있습니다. 이 그림은 AI가 마치 마법처럼 결과를 생성하는 동시에 그 내부가 보이지 않는다는 아이디어를 담고 있습니다.

□ 🔍 ⏪ ⏵ ⏹ ⏷ ⏸ ⏹ ⏵

✉️ ChatGPT에게 메시지 쓰기

🔊

# 🎩 ChatGPT랑 친해지기

## 1. 언어모델 (Language Model, LM)

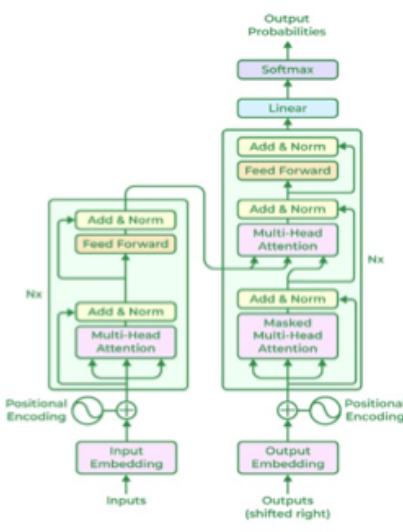
### 1.1 언어모델이란?

#### 👀 정의

주어진 단어들을 보고 다음으로 올 확률이 높은 단어를 예측하는 도구

#### 📝 언어모델의 발전

언어 모델은 빈도수 기반의 확률 추론을 하는 통계적 언어모델부터 복잡한 연산을 수행하는 신경망 언어모델까지 단점을 극복하며 발전을 이루어왔습니다. 최근 가장 많이 활용되는 **Transformer** 기반의 언어모델들 역시 신경망 언어모델이며, 어텐션 메커니즘을 통해 문맥을 효과적으로 이해하고 처리 속도를 대폭 향상시키는 혁신적인 구조를 갖추고 있습니다.



Transformer

#### 🌈 예시: 언어모델

오늘 점심은 [다음 나올 단어]  
구내식당-90%, 외식-9.9%, 휴지-0.0001%, ...

- 상황: 언어 모델에게 오늘 점심은이라는 단어를 입력하였을 때, 다음 단어에 대한 예측
- : 오늘 점심은 구내식당

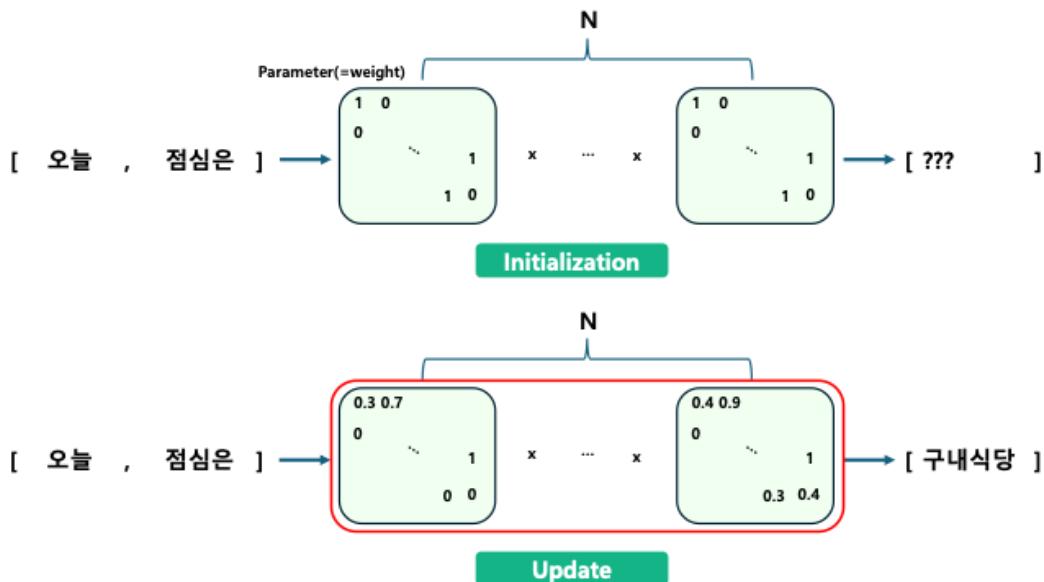
#### 📍 다음 단어를 어떻게 예측할 수 있을까?

- 다음 나올 단어에 대한 확률을 계산하기 위해 많은 문장들을 학습시킵니다.
- 학습을 통해 기존 문장들에서 오늘 과 점심은 이라는 단어 뒤에는 구내식당이라는 단어가 나올 확률이 높음을 알 수 있습니다.
- 즉, 언어 모델은 학습 데이터에 의존적이며, 이에 따라 다른 문장을 도출할 수 있습니다.

## 💡 언어모델의 학습은 어떻게 시킬까?

앞서 최신 언어 모델은 Transformer 구조로 되어 있다고 언급했습니다. 이는 변신하는 로봇과 같은 막연한 개념이 아니라, 실제로는 단순한 행렬 연산을 통해 문장 간의 관계를 계산하고 parameter를 update하는 방식입니다. 즉, 데이터를 사용해 행렬연산을 반복적으로 수행하고, 최적의 parameter를 찾는 과정을 학습이라고 부릅니다.

- Transformer <-> 뇌
- Parameter Update <-> 학습
- 예시



## 📌 더 알아보기

- NLP 분야에서 학습 데이터는 corpus라 불립니다.
- 보유한 학습 데이터는 도메인에 의존적이기에 언어 모델은 도메인에 따라 다르게 구축할 수 있습니다.

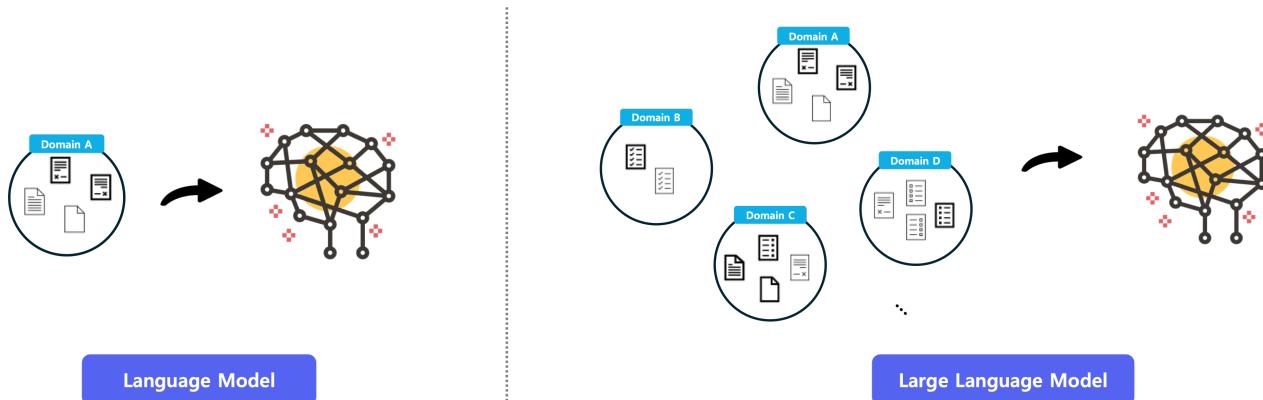
## 1.2 대규모 언어모델이란?

### 👀 Definition

기존 언어 모델보다 훨씬 많은 데이터를 학습시켜 다양한 언어 예측 작업을 수행할 수 있는 도구

## 💡 vs 언어모델

대규모 언어모델은 기존 언어모델보다 훨씬 더 많은 데이터를 학습했기 때문에, 풍부한 지식을 기반으로 다양한 일반적인 상황에서 더 정확하고 포괄적인 답변을 생성할 수 있습니다. 즉, 이전 언어모델은 작은 데이터로 학습되어 특정 도메인에 한정된 답변만을 제공하는 경향이 있었던 반면, 대규모 언어모델은 광범위한 데이터를 학습하여 다양한 주제와 맥락에서 유연하고 다각적인 답변을 제공할 수 있습니다.



### 💡 더 많은 데이터를 어떻게 학습시킬까?

앞서 언어모델에서 학습은 Transformer의 Parameter를 update하는 것이라고 하였습니다. 이는 현실의 학습의 개념으로 보자면 Transformer라는 뇌에 문서들을 저장시키는 것과 동일합니다. 따라서 훨씬 많은 데이터를 언어모델에 학습시키려면, 더 큰 뇌와 데이터를 이해하기 위한 연산 능력이 필요합니다. 이때 큰 뇌는 컴퓨터의 저장 공간으로 메모리 크기를 나타내고, 연산 능력은 컴퓨터의 프로세싱 파워를 의미합니다. 즉, 고성능의 컴퓨터가 필수적이며, 이러한 컴퓨터는 방대한 데이터를 효율적으로 처리하고 학습하는 데 중요한 역할을 합니다.

### 💡 참조

- 💼 Naver Cloud Platform-GPU Price

## 2. 할루시네이션

### 2.1 할루시네이션이란?

#### 👀 정의

모델이 사실이 아닌 정보를 생성하거나, 실제로 존재하지 않는 데이터를 만들어내는 현상

### 💡 왜 할루시네이션이 나타날까?

- 모델의 학습 방식:** 모델의 학습은 대규모 텍스트 데이터를 기반으로 이루어집니다. 이 과정에서 모델은 다양한 정보를 이 함께 존재하는 텍스트에서 단어와 문장 간의 패턴을 학습하게 되며, 이로인해 특정 입력에 대해 가능성성이 높은 출력을 생성합니다. 즉, 모델이 질문을 이해하거나 기억하는 것이 아니라 단지 패턴을 따르는 것이기 때문에, 잘못된 또는 존재하지 않는 정보를 생성할 수 있습니다.
- 모델의 목표:** 언어 모델은 정확한 정보를 생성하는 것을 목표로 하기보다는, 주어진 텍스트 입력에 가장 그럴듯한 출력을 생성하는 것을 목표로 합니다. 이로인해, 모델은 사실과 일치하지 않지만 언어적으로는 타당해 보이는 출력을 생성할 수 있습니다.
- 데이터의 시점:** 언어 모델은 학습된 시점까지의 데이터를 바탕으로 정보를 생성합니다. 이후 발생한 사실이나 최신 정보를 반영하지 못하기 때문에, 최신 정보와 관련된 질문에 대해 잘못된 정보를 제공할 수 있습니다.

### ☰ 실습: 할루시네이션 경험하기

🤖 : 첫번째 실습입니다. ChatGPT에 접속하여 아래와 같이 상장 기업에 대한 주가를 물어보는 질문을 해보겠습니다.

삼성전자 주가 알려줘

### 💡 어떻게 할루시네이션을 방지할 수 있을까?

- **최신 데이터 사용:** 모델을 **최신 데이터**로 주기적으로 **학습**하여 최신 정보를 반영하도록 합니다.
- **프롬프트 엔지니어링:** 명확하고 구체적인 질문을 통해 잘못된 정보를 생성할 가능성을 줄입니다.

## 3. 프롬프트

### 3.1 프롬프트란?

#### 👀 정의

AI 모델에게 작업을 지시하거나 질문을 던질 때 사용함

#### 📝 프롬프트 엔지니어링

프롬프트 엔지니어링은 AI 모델이 원하는 결과를 생성하도록 입력을 설계하는 과정입니다. 효과적인 프롬프트는 명확하고 구체적이며, 원하는 결과를 도출하기 위한 정보와 제약 조건을 포함합니다. 프롬프트의 구조와 언어적 표현을 잘 설계하면 모델의 성능을 극대화할 수 있습니다

#### 🎹 실습: 할루시네이션 개선하기

🤖 : 앞서 주가 정보에 대해 잘못된 답을 주었던 것을 개선해보도록 하겠습니다.

삼성전자 주가알려줘, 단 최신 정보를 모른다면 대답하지마

### 3.2 Few-Shot Learning

#### 👀 정의

모델에 질문시 예시를 더하여 향상된 답변을 얻는 방법으로 다양한 작업에 적용될 수 있음

#### 📝 In-context learning

In-context learning은 GPT와 같은 대규모 언어 모델에서 사용되는 학습 기법 중 하나입니다. 이 기법의 핵심 아이디어는 모델이 주어진 문맥(context) 내에서 학습하고 정보를 추론하는 능력을 갖추게 하는 것입니다. 대규모 언어 모델의 경우 규모가 매우 크기 때문에 학습을 위해서는 많은 자원이 필요합니다. 따라서 주어진 입력 문맥을 활용하여 학습과 유사한 답변을 효과적으로 얻을 수 있습니다.

#### 🌈 예시: ConversationSummaryMemory

좋은 운동하셨네요! 하체 운동은 근력과 체력 모두에 큰 도움이 되죠. 오늘 하체 운동으로 어떤 동작들을 하셨나요? 많이 힘드셨을 것 같은데, 어떻게 느껴지셨어요?

내일은 무슨운동하면 좋을까?

오늘 하체 운동을 하셨으니 내일은 상체나 코어 운동을 하시는 게 좋을 것 같아요. 이렇게 하면 몸의 다른 부위를 휴식시켜 줄 수 있고, 전신의 균형을 맞출 수 있습니다. 아래 몇 가지 추천 드릴게요:

1. 상체 운동:
  - 가슴 운동: 벤치프레스, 푸쉬업
  - 등 운동: 플립, 바벨 로우
  - 어깨 운동: 덤벨 솔더 프레스, 사이드 레터럴 레이즈
  - 팔 운동: 덤벨 컬, 트라이셉스 익스텐션

2. 코어 운동:
  - 폴랭크, 러시안 트위스트, 레그 레이즈, 크런치
3. 전신 운동: 버피, 캐틀벨 스윙 등 전신을 사용하는 운동도 추가하면 좋습니다.

몸 상태에 맞춰 강도를 조절하고, 충분한 스트레칭과 휴식을 통해 회복을 도와주세요. 즐거운 운동 되세요!

GPT는 우리의 정보를 질문시마다 학습하지 않습니다. 그럼에도 나와의 대화를 기억한다는 듯 앞서 질문들에 대한 정보를 참고하여 대답을 합니다. 이는 아래와 같이 프롬프트로 우리의 대화 내역을 요약하는 질문을 수행하고, 이를 새로운 질문에 포함시켜 전달하기 때문입니다. 이때 요약을 위해 한가지의 예시를 포함하고 있는데, 이를 통해 매번 다른 질문이 포함되어 있어도 일관되고 정확한 요약을 얻을 수 있습니다. 이렇게 프롬프트에 약간의 예시를 포함해 향상된 답변을 얻는 방식이 *few-shot learning*입니다.

```
# Langchain의 Summary Prompt
_DEFAULT_SUMMARIZER_TEMPLATE = """Progressively summarize the lines of
conversation provided, adding onto the previous summary returning a new
summary.
```

#### EXAMPLE

Current summary:

The human asks what the AI thinks of artificial intelligence. The AI thinks artificial intelligence is a force for good.

New lines of conversation:

Human: Why do you think artificial intelligence is a force for good?

AI: Because artificial intelligence will help humans reach their full potential.

New summary:

The human asks what the AI thinks of artificial intelligence. The AI thinks artificial intelligence is a force for good because it will help humans reach their full potential.

END OF EXAMPLE

Current summary:

{summary}

New lines of conversation:

{new\_lines}

New summary: "한글번역 필요함"

[한글번역 필요함]

### III 실습: Few-shot Learning

👤 : 미생의 명장면인 장그래 레전드 PT를 아래 회의록 양식에 맞추어 ChatGPT를 통해 작성해 볼까요?

- 🔗 미생-장그래의 레전드 PT
- ▣ 장그래 레전드 PT 텍스트
- 회의록 예시

#### 1. 회의 정보

- 제목: 기능 명세 작성 (채팅/메시지)
- 일시: 2024.08.06
- 장소: 온라인

#### 2. 참석자

- 바비 챈튼
- 데니스 로

#### 3. 목적

채팅/메시지에 대한 기능 명세 작성

#### 4. 채팅/메시지 기능 명세

[frontend]

- 채팅 메인 화면 개발
- DM 메인 화면 개발

[backend]

- 채팅방 모델링
- 채팅 메세지 모델링

[프로세스 정의서 작성]

- 채팅 프로세스 정의

#### 5. 차기 회의 일정

- 일시: 2024.08.07
- 장소: 온라인
- 안건: 서베이

# 작성자: 조지 베스트

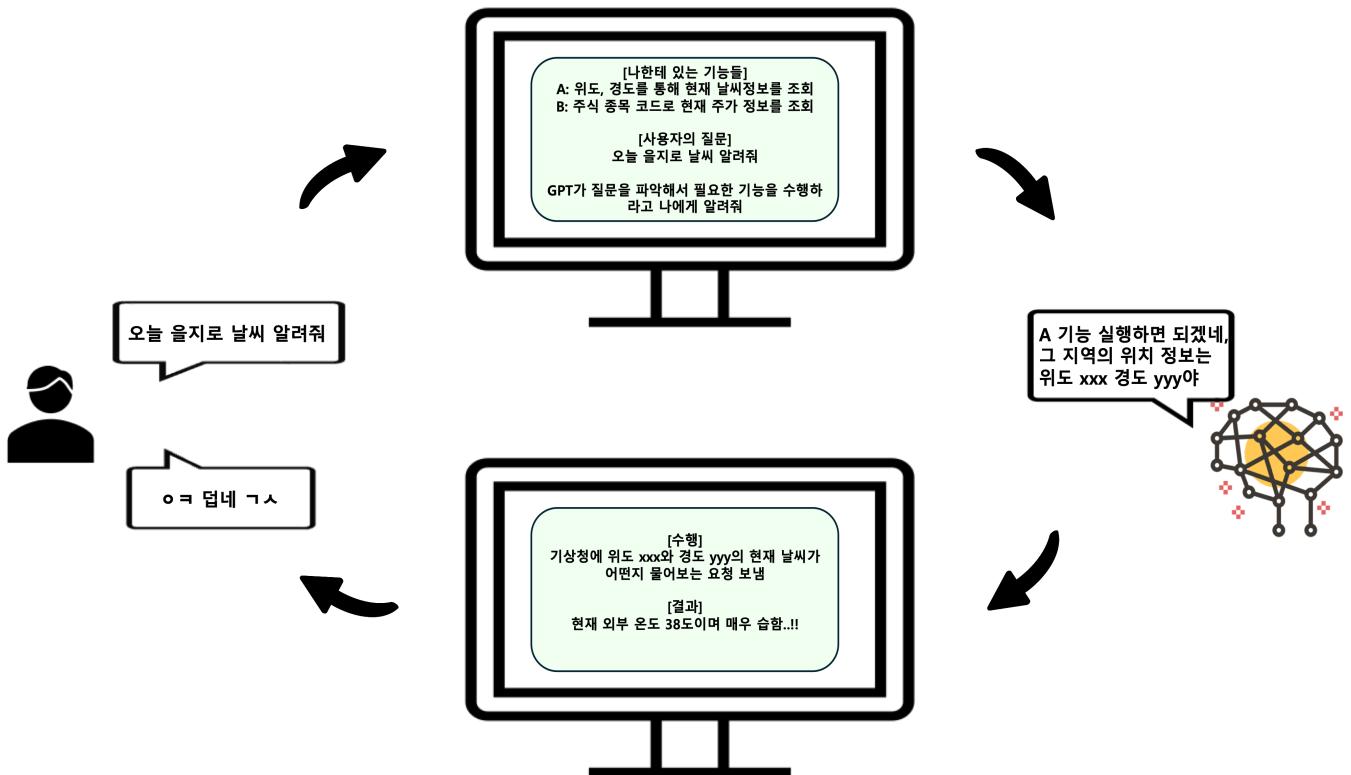
## ♥ [Bonus] Function Calling

### 정의

대형 언어 모델이 외부 시스템, 도구, 또는 API와 상호작용하기 위해 특정 기능을 호출하는 메커니즘

- GPT가 알지 못하는 실시간 정보들을 기능으로 구현해놓고, GPT에게 필요하다면 이를 활용하라고 전달

### 🌈 예시: Function calling1



### 🌈 예시: Function calling2

🤖 : 앞서 주가 정보의 끝판왕을 만나봅니다.

### 참조

- 📖 Langchain 공식문서-Function calling

# ⭐ 고급 데이터 분석(Advanced Data Analysis, ADA)

ChatGPT의 고급 데이터 분석(Advanced Data Analysis, ADA) 기능은 사용자가 보다 복잡한 데이터 분석 작업을 수행할 수 있도록 지원하는 기능입니다. 이 기능은 Python과 같은 프로그래밍 언어를 활용하여 다양한 데이터를 분석하고 시각화하는 데 유용합니다.

## 1. 데이터 전처리 & 한글 패치

데이터 분석에서 가장 많은 시간을 차지하는 부분은 데이터를 정리하는 데이터 전처리 작업입니다. 불필요한 부분을 제거하고, 데이터의 각 열마다 동일한 포맷의 데이터가 들어가도록 형식을 정리하는 등의 과정이 전처리 과정에 포함됩니다. 지금까지는 데이터를 전처리하기 까지의 많은 과정을 데이터 분석가가 수동으로 작업을 해왔습니다. 그러나 이제 ChatGPT와 함께라면 내가 전처리할 데이터를 업로드하고 요청하는 것만으로 빠게 수행할 수 있습니다.

### 📌 Tip1. 파일명을 잘 입력하자

- 파일명 자체도 GPT가 해석하기 위한 중요한 정보
- 한국어도 가능하나 영어로 할 시에 효과가 더 높음

### 📌 Tip2. Divide & Conquer

- 많은 내용의 질문을 하지 않고 단계적으로 절차를 분할하여 명확하게 전달

### 1.1 데이터 전처리

#### 🌈 실습: 네이버 검색량 전처리

👤 : 아래 데이터를 다운로드 받아 GPT에 업로드하고, 전처리를 위한 프롬프트를 전달해 봅니다.

#### 🔗 전처리 대상 데이터

- ▣ 네이버 트렌드의 검색량 키워드
- ChatGPT, 제미나이, 클로드의 검색량을 나타냄

### 📌 Tip3. 첫인상 심어주기

- 첫 대화에 이 데이터가 어떤 데이터인지 간략하게 설명
- 또한 어떠한 역할을 수행하는 챗봇이 될지 역할 부여

#### 💬 Chat Prompt

너는 전처리를 위한 챗봇이다.

전달하는 데이터는 네이버 키워드 검색량 데이터로, ChatGPT, 제미나이, 클로드의 일간 검색량을 포함하고 있다.

이 데이터를 살펴보고, 데이터를 전처리하기 위한 과정을 설명 및 수행해라.  
또한, 모든 과정은 한국어로 설명해라.

### 📌 Tip4. 작업 중 저장은 필수

- 데이터 전처리/ 가공 시, 다운로드 링크를 제공하라고 해서 결과물을 확인

### 💬 Chat Prompt

전처리된 결과물을 확인할 수 있게 다운로드 링크 제공해줘

## 1.2 한글 패치

### 🌈 실습: GPT환경에 패키지 설치

💻 : 전처리된 데이터에 대해 시각화를 요청해 봅니다.

### 💬 Chat Prompt

일간 키워드 검색량을 비교할 수 있게 라인차트를 그려줘

### 📍 한국어가 깨지는 이유

- 코드가 실행되는 ChatGPT의 환경에 한국어가 설치되어 있지 않기 때문
- 컴퓨터의 가상환경에 한국어를 사용할 수 있는 라이브러리 설치
- 다만, 하나의 창에서만 유효하고, 3시간 정도 유지됨

### 🔗 한국어 패치 다운로드

- 💾 Korea matplotlib
- 💾 나눔고딕 폰트

### 💬 Chat Prompt

가상환경에 한국어를 사용할 수 있도록 whl 파일과 ttf 파일을 제공하였으니, 아래 과정을 수행하며 한국어 환경 설정 수행

- Code interpreter로 제공한 파일들을 다운로드 및 수행
- 가상환경에 설치 및 한국어를 사용할 수 있도록 설정
- 설치 과정 및 코드는 생략하고 완료후 테스트 그래프 제시
- 모든 설명은 한국어로 진행

## 2. 데이터 시각화

복잡한 차트를 시각화하기 위한 코드를 챗GPT가 알아서 생성하고, 결과물로 차트를 만들어 주기 때문에 자유도 있게 차트를 생성할 수 있습니다. 이를 통해 차트를 만든 다음 한글, PPT, 워드 등 보고서 양식에 활용할 수 있습니다.

### 2.1 데이터 시각화

### 🌈 실습: 대한민국 출산율 사각화

🤖 : 아래 데이터를 다운로드 받아 GPT에 업로드하고, 시각화를 위한 프롬프트를 전달해 봅니다. 작업을 수행하기 전에 한국어 설정을 잊지 마세요!

### 🔗 시각화 대상 데이터

- ▣ 대한민국 출산율 데이터
- 연도 별 출생아수, 합계출산율, 출생성비를 포함하는 데이터

### 💬 Chat Prompt

너는 전달받은 데이터를 시각화하는 챗봇이다. 아래 규칙에 맞추어서 시각화를 수행하라.

- 차트제목: 연간 대한민국 합계 출산율 추이
- X축: 연도
- Y축: 합계출산율(명)
- 데이터 레이블은 차트 위에 볼드체로 표기
- 범례는 우측 상단에 표시
- 출생아수(명) 컬럼은 막대 그래프로 표현
- 합계출산율(명) 컬럼은 꺽은선 그래프로 표현
- 출생아수(명)와 합계출산율(명)의 스케일이 다를 경우 합계출산율을 기준으로 출생아수(명)의 그래프를 스케일링

시각화 수행

## 2.2 추가 요청

🤖 : 이후 입맞에 맞게 더욱 그래프를 세부 조정해볼 수 있습니다.

### 💬 Chat Prompt

가장 출생아수(명)이 많은 연도는 막대그래프를 초록색으로 그려줄래?

## 3. 고급 데이터 분석



- 데이터 업로드 및 설명: 데이터를 업로드하여 기본적인 데이터 구조를 이해시킴
- 데이터 전처리: 데이터를 분석 가능한 형태로 정돈
- 탐색적 데이터 분석: 데이터 전반에 대해 심도 있게 탐색
- 분석 과제 설정: 무엇을 분석할지 과제를 설정
- 인사이트 도출: 과제를 분석하면서 인사이트를 도출

### 📌 Tip5. 심도있는 인사이트 도출

- 탐색적 데이터 분석, 분석 과제 설정 등 여러 내용을 챗GPT의 발화 내용을 기반으로 도출함

- 전통적인 데이터 분석 방법론을 따라가면서 챗GPT가 데이터를 살펴보고 분석할 수 있게 여러 시도를 함

## 🌈 실습: 마이크로소프트 주가 데이터 분석

👨‍💻 : 마이크로소프트의 주식 가격 데이터를 분석해봅니다.

### 🔗 데이터 분석 대상 데이터

- 💻 마이크로소프트 주가
- 2022년 10월부터 24년 5월까지의 데이터

### 3.1 데이터 업로드 및 설명

#### 💬 Chat Prompt

제시한 데이터는 마이크로소프트의 주가 데이터이다. 이 데이터를 살펴보고 설명해라. 향후 모든 과정은 한국어로 수행하라.

### 3.2 데이터 전처리

#### 💬 Chat Prompt

제시한 데이터를 살펴보고 아래 과정을 수행해라.

- 데이터를 전처리하기 위한 과정을 설명 및 수행
- 전처리 수행 후 다운로드 링크를 제시

### 3.3 탐색적 데이터 분석(EDA)

#### 💬 Chat Prompt

제시한 데이터를 깊이 있게 이해할 수 있도록 일변량 EDA를 실시해라.

#### 💬 Chat Prompt

제시한 데이터를 깊이 있게 이해할 수 있도록 다변량 EDA를 실시해라.

### 📌 Tip6. 매직 키워드

- EDA는 탐색적 데이터 분석을 수행할 수 있는 매직 키워드
- 하지만 포괄적으로 질문하는 것 보다 체계적으로 질문하는 것이 좋음

### 📝 일변량 vs 다변량

- 일변량: 하나의 변수에 대해서만 탐색하여 설명
- 다변량: 두 개 이상의 변수의 관계를 비교 탐색하여 설명

## 3.4 분석 과제 설정

### 💬 Chat Prompt

제시한 데이터를 기반으로 분석할만한 가설을 제안해라.

### 📌 Tip6. 분석 과제 도출

- 분명한 계획이 있다면 분석 과제를 명확히 설정
- 그렇지 않은 경우 분석 방향 제시, 시각화 아이디어, 가설 제안 등으로 챗GPT에게 발화 요청
- 마음에 들지 않은 경우 다시 생성하기 요청
- 제시한 가설 외에도 명시적으로 생성하도록 요청 할 수 있음

### 💬 Chat Prompt

제안한 가설 외에 3가지 더 제안해라.

## 3.5 인사이트 도출

### 💬 Chat Prompt

OpenAI에 Microsoft가 많은 투자를 하는 것으로 알고 있다. OpenAI와 관련된 이벤트가 Microsoft의 주가에 어떤 영향을 주었는지 분석을 수행할 것이다.

2022년 10월 이후 주어진 데이터의 마지막 날짜까지의 OpenAI의 주요 이벤트를 리스트업하여 제시하라. 인터넷 기반으로 검색하되, 공신력 있는 기사와 OpenAI 혹은 Microsoft가 발표한 내용만을 대상으로 하여라.

### 📌 Tip7. 외부 자료 조사

- Function Calling 외에도 GPT-4o는 외부 인터넷 조사를 통해 데이터를 활용할 수 있음

### 💬 Chat Prompt

리스트업한 이벤트의 발생 날짜와 해당 날짜 근처의 Microsoft 주가 데이터를 보유한 데이터에서 찾고 비교해서 OpenAI의 이벤트가 Microsoft의 주가에 영향을 줬는지 분석하라.

### 💬 Chat Prompt

앞선 분석 내용을 한 눈에 볼 수 있도록 라인차트로 시각화해라. 주요 이벤트 시점에서는 표시를 통해 이벤트 전후로 어떻게 주가가 변동되고 있는지를 나타내라.

### 💬 Chat Prompt

지금까지의 분석 결과를 바탕으로 너의 의견을 결론으로 제시하라.

## 4. Lessons Learn

ChatGPT의 고급 데이터 분석을 사용한다면 코딩 없이 프롬프트만으로 전처리, 시각화를 거쳐 데이터 분석 모델링까지 수행할 수 있습니다. 그런데 아무리 좋은 마술 도구가 있더라도 마술사의 손놀림이 능숙하지 않다면 관객들이 현혹되지 않는 것처럼, 분석가의 데이터 분석 지식이나 활용 능력이 부족하다면 고급 데이터 분석의 효과도 크게 떨어질 수 있습니다. 결국, 분석가는 도구를 효과적으로 활용할 수 있는 기본적인 데이터 분석 지식과 경험을 갖추어야만, 그 도구의 진정한 가치를 발휘할 수 있습니다.

### 목표

- 데이터 분석 전반의 지식 배우기
- ChatGPT를 통해 지식을 코드로 변환하기

### [Bonus] 사전 설문지 데이터 분석

강의를 구성하기 위해 사전 설문지를 통해 분석 해보았습니다. 함께 확인해볼까요?

# 🔥 Warm Up 🔥

---



- **Pandas**는 Python에서 데이터를 조작하고 분석하는 강력한 도구
- 대용량 데이터 처리를 쉽게 할 수 있도록 설계된 데이터 분석 라이브러리
- **Pandas**는 테이블 형식의 데이터를 다루는 데 최적화되어 있으며, 엑셀과 유사한 기능 제공

## 📍 엑셀과 Pandas의 비교

기능	엑셀 (Excel)	Pandas
데이터 크기	<ul style="list-style-type: none"> <li>- 수십만 행까지 처리 가능</li> <li>- 대용량 데이터 처리에 한계가 있음</li> </ul>	<ul style="list-style-type: none"> <li>- 수백만 행 이상의 대규모 데이터도 빠르게 처리 가능</li> </ul>
데이터 분석	<ul style="list-style-type: none"> <li>- 수식과 매크로 사용</li> <li>- 직관적이지만 복잡한 분석은 어려움</li> </ul>	<ul style="list-style-type: none"> <li>- 복잡한 데이터 분석과 변환을 코드 몇 줄로 수행 가능</li> </ul>
시각화	<ul style="list-style-type: none"> <li>- 데이터 시각화 기능이 강력</li> <li>- 그래프 작성이 쉬움</li> </ul>	<ul style="list-style-type: none"> <li>- matplotlib, seaborn 등의 라이브러리와 통합 가능</li> </ul>
사용 용이성	<ul style="list-style-type: none"> <li>- GUI 기반으로 사용이 직관적</li> <li>- 초보자도 쉽게 접근 가능</li> </ul>	<ul style="list-style-type: none"> <li>- 코드 기반으로 초보자에게는 다소 어려울 수 있음</li> </ul>
기능 확장성	<ul style="list-style-type: none"> <li>- 수식과 기능 제한</li> <li>- 복잡한 통계나 머신러닝 기능 부족</li> </ul>	<ul style="list-style-type: none"> <li>- Python 생태계와의 통합으로 기능 확장 가능</li> <li>- 복잡한 통계 및 머신러닝 작업 가능</li> </ul>

## 📍 Numpy와의 비교

- 매우 큰 데이터에 과학 계산을 하기 위한 파이썬의 패키지에는 Numpy가 있음
- Numpy 데이터에 라벨을 붙여 행과 열의 라벨로 데이터를 분석하고 처리할 수 있도록 한 것이 Pandas
- Numpy는 계산을 위해 사용된다면 Pandas는 전처리, 시각화 등을 위해 사용함

# Pandas

## 1 Pandas 데이터 구조

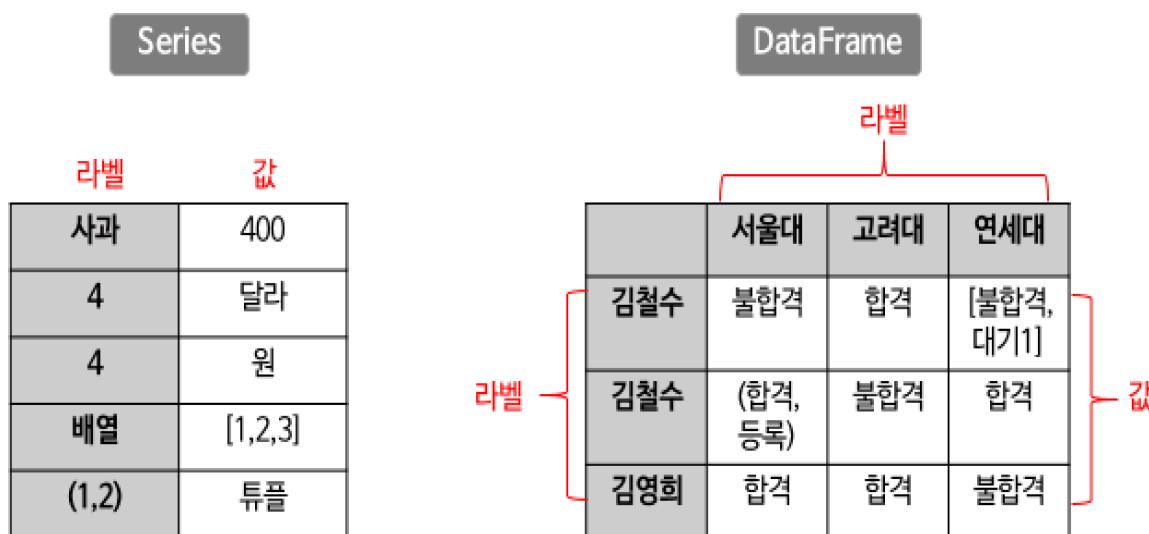
- Pandas에서 제공하는 데이터 구조는 Series와 DataFrame이 있음

### 1.1 Series

- 각 값에 라벨(인덱스)이 있고 하나의 데이터 형식으로 이루어진 1차원 배열
- 라벨로 데이터에 접근 할 수 있음

### 1.2 DataFrame

- 행과 열에 라벨이 있는 2차원 배열 ↪ Excel
- 각 열은 서로 다른 데이터 타입을 지닐 수 있음



## 2 DataFrame 기본

Pandas의 DataFrame을 사용하여 분석하기 위한 가장 첫 단계는 데이터를 호출하고 데이터의 내용과 요약, 통계정보를 확인하는 것입니다. 데이터나 분석 도구에 따라 컬럼명이나 컬럼의 타입을 변경해야 할 때도 있습니다.

### 2.1 Pandas 사용 준비

👨‍💻 : olist 구매 기록 데이터를 chat gpt에 업로드 해주세요!

#### 🔗 실습 데이터

- olist 구매기록 데이터

#### 💬 Chat Prompt

제시한 데이터는 이커머스회사 olist의 구매기록 데이터이다. 이 데이터의 컬럼을 살펴보고 설명해라.

모든 과정에서 설명시 한국어를 사용하라.

### 🤖 GPT 출력:Message

이 데이터는 이커머스 회사인 Olist의 구매 기록을 포함하고 있으며, 총 9개의 컬럼으로 구성되어 있습니다. 각 컬럼의 설명은 다음과 같습니다:

1. **order\_id**: 주문 ID로, 각 주문을 고유하게 식별할 수 있는 문자열입니다.
2. **customer\_id**: 고객 ID로, 주문을 한 고객을 고유하게 식별할 수 있는 문자열입니다.
3. **order\_status**: 주문 상태를 나타내는 컬럼으로, 'delivered'(배송 완료)와 같은 상태 정보가 포함됩니다.
4. **order\_purchase\_timestamp**: 주문이 이루어진 시점을 나타내는 타임스탬프로, 주문이 생성된 날짜와 시간이 기록되어 있습니다.
5. **product\_category**: 구매된 상품의 카테고리를 나타내는 문자열입니다.
6. **payment\_method**: 결제 수단을 나타내는 컬럼으로, 'credit\_card'(신용카드), 'boleto'(브라질의 지불 수단) 등이 포함됩니다.
7. **installments**: 할부 횟수를 나타내는 컬럼으로, 고객이 구매할 때 선택한 할부 개수입니다. 일부 값은 결측치(NaN)로 표시됩니다.
8. **item\_price\_usd**: 상품의 가격을 미국 달러(USD)로 나타내는 컬럼입니다.
9. **shipping\_cost\_usd**: 배송비를 미국 달러(USD)로 나타내는 컬럼입니다.

이 데이터는 997개의 레코드(행)로 구성되어 있으며, 각 컬럼의 데이터 타입은 문자열(object)과 부동 소수점(float64)으로 구성되어 있습니다. 결측치는 `installments`, `item_price_usd`, `shipping_cost_usd` 컬럼에서 발견됩니다.

### 🤖 GPT 출력:Code

```
import pandas as pd

# Load the dataset
file_path = 'your_file.csv'
data = pd.read_csv(file_path)

# Display the first few rows and columns names of the dataset to
# understand its structure
data.head(), data.columns
```

	order_id	customer_id
0	e481f51cbdc54678b7cc49136f2d6af7	9ef432eb6251297304e76186b10a928d
1	128e10d95713541c87cd1a2e48201934	a20e8105f23924cd00833fd87daa0831
2	0e7e841ddf8f8f2de2bad69267ecfbef	26c7ac168e1433912a51b924fb34d34
3	bfc39df4f36c3693ff3b63fcbea9e90a	53904ddbea91e1e92b2b3f1d09a7af86
4	53cdb2fc8bc7dce0b6741e2150273451	b0830fb4747a6c6d20dea0b8c802d7ef

	order_status	order_purchase_timestamp	product_category	payment_method
0	delivered	2017.10.2 10:56	housewares	credit_card
1	delivered	2017.8.15 18:29	housewares	credit_card

```

2     delivered          2017.8.2 18:24    housewares credit_card
3     delivered          2017.10.23 23:26   housewares boleto
4     delivered          2018.7.24 20:41   perfumery boleto

installments  item_price_usd  shipping_cost_usd
0            1.0           5.6981        1.6568
1            3.0           5.6981        1.4782
2            NaN           5.6981        1.4782
3            1.0           5.6981        2.6790
4            1.0          22.5530        4.3244 ,
Index(['order_id', 'customer_id', 'order_status',
       'order_purchase_timestamp',
       'product_category', 'payment_method', 'installments',
       'item_price_usd',
       'shipping_cost_usd'],
      dtype='object')

```

## 2.2 DataFrame 출력하기

- `head()`, `tail()` 함수를 사용해 데이터를 부분적으로 확인
- 함수의 인자로 아무 숫자도 지정하지 않으면 컬럼명을 포함하여 6줄을 출력

### MISSION

Dataframe의 `head()`와 `tail()` 사용해보기

```
print(df.head())
```

```

order_id          customer_id \
0 e481f51cbdc54678b7cc49136f2d6af7 9ef432eb6251297304e76186b10a928d
1 128e10d95713541c87cd1a2e48201934 a20e8105f23924cd00833fd87daa0831
2 0e7e841ddf8f8f2de2bad69267ecfbcf 26c7ac168e1433912a51b924fb34d34
3 bfc39df4f36c3693ff3b63fcbea9e90a 53904ddbea91e1e92b2b3f1d09a7af86
4 53cdb2fc8bc7dce0b6741e2150273451 b0830fb4747a6c6d20dea0b8c802d7ef

order_status order_purchase_timestamp product_category payment_method \
0     delivered          2017.10.2 10:56    housewares credit_card
1     delivered          2017.8.15 18:29    housewares credit_card
2     delivered          2017.8.2 18:24    housewares credit_card
3     delivered          2017.10.23 23:26   housewares      boleto
4     delivered          2018.7.24 20:41   perfumery      boleto

installments  item_price_usd  shipping_cost_usd
0            1.0           5.6981        1.6568
1            3.0           5.6981        1.4782
2            NaN           5.6981        1.4782
3            1.0           5.6981        2.6790
4            1.0          22.5530        4.3244

order_id          customer_id \
992  65b923f47b964f5e6be363f33a1cd0ed c0dfb57c7c700f11633aba234dcce6b7

```

993	c762b7683314cc1cf880475f78aa3821	79c4c03824db69d2f61c1e8d8ec24443
994	088a21dbb29d74a07738fb93a96b507a	fd04c6764e22cc8daca7064ed73fdf6c
995	088a21dbb29d74a07738fb93a96b507a	fd04c6764e22cc8daca7064ed73fdf6c
996	088a21dbb29d74a07738fb93a96b507a	fd04c6764e22cc8daca7064ed73fdf6c
...		
993	2.0	17.0981
994	1.0	NaN
995	1.0	17.0981
996	1.0	17.0981

## 2.3 데이터의 타입 확인하기

- `object`: 문자열
- `int, float, decimal` ..: 숫자형
- `bool`: 불리안

### 💬 Chat Prompt

제시한 데이터에서 컬럼들의 타입을 알려줘

### 🤖 GPT 출력: Message

제시한 데이터의 각 컬럼에 대한 데이터 타입은 다음과 같습니다:

- `order_id: object` (문자열)
- `customer_id: object` (문자열)
- `order_status: object` (문자열)
- `order_purchase_timestamp: object` (문자열)
- `product_category: object` (문자열)
- `payment_method: object` (문자열)
- `installments: float64` (실수형)
- `item_price_usd: float64` (실수형)
- `shipping_cost_usd: float64` (실수형)

대부분의 컬럼이 문자열로 되어 있으며, `installments`, `item_price_usd`, `shipping_cost_usd` 컬럼은 실수형(`float64`) 데이터 타입을 가지고 있습니다.

### 🤖 GPT 출력: Code

```
# Checking the data types of each column in the dataset
data_types = data.dtypes
data_types
```

## 2.4 데이터 타입 변경하기

👨‍💻 : 구매 빈도는 정수여야 할 것 같습니다. 데이터의 타입을 변경하도록 해볼까요?

 Chat Prompt

installments의 컬럼을 정수형으로 변경해줘

 GPT 출력: Message

installments 컬럼을 정수형(int64)으로 성공적으로 변환했습니다. 이제 이 컬럼은 더 이상 실수형이 아닌 정수형 데이터를 포함하고 있습니다.

 GPT 출력: Code

```
# Converting the 'installments' column to integer type
data['installments'] = data['installments'].fillna(0).astype(int)

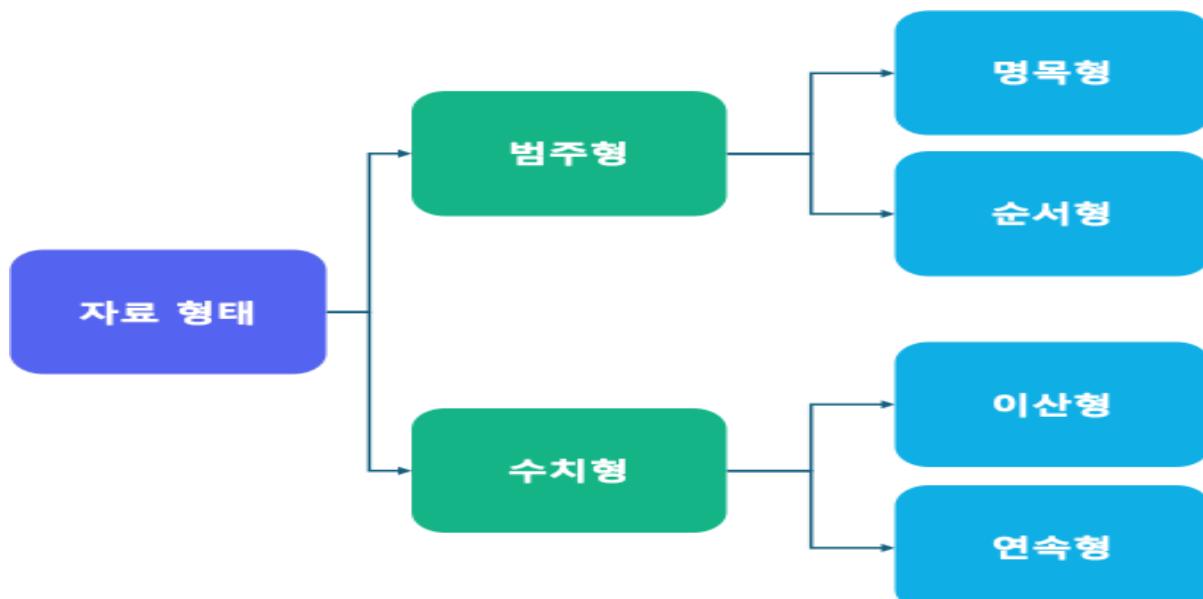
# Verifying the change
data['installments'].dtype
```

 더 알아보기

- 데이터 타입을 변경하는 것은 전처리의 영역에 속함

## 2.5 데이터의 자료형태 확인하기

- 데이터의 자료형태는 시각화, 전처리 그리고 분석 시 모두 중요하게 적용되는 개념
- 각 형태에 따라 다른 기법이 사용됨



분류	상세 분류	설명	예시
범주형		몇 개의 범주로 나누어진 자료를 의미	

분류	상세 분류	설명	예시
명목형	순서 관계가 없는 자료		성별(남성, 여성), 혈액형(A형, B형, O형, AB형 등)
순서형	순서 관계가 존재하는 자료		교육 수준(고졸, 대졸, 석사 등), 등급(1등급, 2등급 등)
수치형	이산형과 연속형으로 이루어진 자료를 의미		
이산형	구분된 값을 가짐		출산 횟수, 방문 횟수
연속형	연속적인 값을 가짐		신장, 체중, 온도

### 💬 Chat Prompt

제시한 데이터에서 컬럼들의 타입을 수치형, 범주형으로 구분하여 알려주고 리스트로 저장해줘

### 🤖 GPT 출력:Message

- 범주형(Categorical) 컬럼:
  - order\_id
  - customer\_id
  - order\_status
  - order\_purchase\_timestamp
  - product\_category
  - payment\_method
- 수치형(Numerical) 컬럼:
  - installments
  - item\_price\_usd
  - shipping\_cost\_usd

### 🤖 GPT 출력: Code

```
# 수치형(Numerical) 컬럼 리스트
numerical_columns = ['installments', 'item_price_usd',
'shipping_cost_usd']

# 범주형(Categorical) 컬럼 리스트
categorical_columns = ['order_id', 'customer_id', 'order_status',
'order_purchase_timestamp', 'product_category', 'payment_method']

numerical_columns, categorical_columns
```

```
(['order_id',
  'customer_id',
```

```
'order_status',
'order_purchase_timestamp',
'product_category',
'payment_method'],
['installments', 'item_price_usd', 'shipping_cost_usd'])
```

## 2.6 DataFrame 요약 및 통계 확인하기

- 데이터의 내용을 확인했다면 그 이후에 데이터프레임의 요약, 통계정보를 확인함
- `info()` 함수를 통해 총 데이터의 건수, 각 컬럼의 데이터 형식 그리고 결측치의 건수를 확인할 수 있음
- 수치형 컬럼이 존재하는 경우 `describe()` 함수를 사용하여 수치형 컬럼들의 n-percentile 분포도, 평균값, 최대값, 최소값을 확인할 수 있음
- 이러한 과정은 EDA에 속함

### 💬 Chat Prompt

제시한 데이터의 요약 및 통계정보를 확인해줘

### 🤖 GPT 출력: Message

제시된 데이터의 요약 및 통계 정보는 다음과 같습니다:

수치형 컬럼의 통계 정보:

- `installments`:
  - 총 개수: 997
  - 평균: 3.62
  - 표준 편차: 31.68
  - 최소값: 0
  - 1사분위값: 1
  - 중앙값: 1
  - 3사분위값: 4
  - 최대값: 999
- `item_price_usd`:
  - 총 개수: 898
  - 평균: 29.10
  - 표준 편차: 333.61
  - 최소값: 1.805
  - 1사분위값: 13.11
  - 중앙값: 17.10
  - 3사분위값: 20.90
  - 최대값: 9999
- `shipping_cost_usd`:
  - 총 개수: 898
  - 평균: 115.49
  - 표준 편차: 3336.87
  - 최소값: -9.595
  - 1사분위값: 2.46
  - 중앙값: 3.06
  - 3사분위값: 5.17

- 최대값: 99999

범주형 컬럼의 요약 정보:

- order\_id:
  - 총 개수: 997
  - 고유값: 894
  - 최빈값: 52eb8ce94864a3bb1be410a545f9ed4c (6회 발생)
- customer\_id:
  - 총 개수: 997
  - 고유값: 894
  - 최빈값: 4aa0005ef4b8cb5ce874a4a86b36ed05 (6회 발생)
- order\_status:
  - 총 개수: 997
  - 고유값: 4
  - 최빈값: delivered (982회 발생)
- order\_purchase\_timestamp:
  - 총 개수: 997
  - 고유값: 893
  - 최빈값: 2017.11.24 23:37 (6회 발생)
- product\_category:
  - 총 개수: 997
  - 고유값: 19
  - 최빈값: health\_beauty (278회 발생)
- payment\_method:
  - 총 개수: 997
  - 고유값: 4
  - 최빈값: credit\_card (748회 발생)

이 정보들은 데이터의 전반적인 분포와 특성을 이해하는 데 도움이 됩니다.

### GPT 출력: Code

```
# Generating summary statistics for numerical columns
numerical_summary = data.describe()

# Generating summary statistics for categorical columns
categorical_summary = data.describe(include=['object'])

# 수치형 데이터의 요약 통계량
numerical_summary, categorical_summary
```

	installments	item_price_usd	shipping_cost_usd
count	997.000000	898.000000	898.000000
mean	3.623872	29.099161	115.493469
std	31.681427	333.614822	3336.873052
min	0.000000	1.805000	-9.595000
25%	1.000000	13.110000	2.464300
50%	1.000000	17.098100	3.064700
75%	4.000000	20.898100	5.169425
max	999.000000	9999.000000	99999.000000,

```

order_id
customer_id \
  count           997
997
  unique          894
894
  top      52eb8ce94864a3bb1be410a545f9ed4c
4aa0005ef4b8cb5ce874a4a86b36ed05
  freq            6
6

  order_status order_purchase_timestamp product_category
payment_method
  count           997                 997           997
997
  unique          4                  893           19
4
  top      delivered      2017.11.24 23:37  health_beauty
credit_card
  freq            982                 6             278
748  )

```

### 💬 Chat Prompt

각 범주형 데이터의 고유값에 대해 설명해줘, 단, ID값과 날짜는 제외하고 알려줘

범주형 데이터의 고유값에 대한 설명은 다음과 같습니다(ID값과 날짜는 제외) :

`order_status` (주문 상태) :

- `delivered`: 982회 발생 (배송 완료)
- `shipped`: 12회 발생 (배송 중)
- `invoiced`: 2회 발생 (청구서 발행됨)
- `canceled`: 1회 발생 (주문 취소됨)

`product_category` (상품 카테고리) :

- `health_beauty`: 278회 발생 (건강 및 미용)
- `garden_tools`: 219회 발생 (정원 도구)
- `perfumery`: 125회 발생 (향수)
- `auto`: 91회 발생 (자동차 관련)
- `bed_bath_table`: 52회 발생 (침실, 욕실, 테이블 관련)
- `stationery`: 41회 발생 (문구류)
- `baby`: 40회 발생 (유아용품)
- `toys`: 35회 발생 (장난감)
- `watches_gifts`: 27회 발생 (시계 및 선물)
- `computers_accessories`: 23회 발생 (컴퓨터 및 액세서리)
- `telephony`: 19회 발생 (전화기 관련)
- `electronics`: 13회 발생 (전자기기)
- `office_furniture`: 9회 발생 (사무용 가구)

- cool\_stuff: 6회 발생 (멋진 물건)
- pet\_shop: 5회 발생 (애완동물 용품)
- furniture\_decor: 5회 발생 (가구 및 장식)
- housewares: 4회 발생 (가정용품)
- construction\_tools\_construction: 4회 발생 (건설 도구)
- air\_conditioning: 1회 발생 (에어컨)

payment\_method (결제 수단) :

- credit\_card: 748회 발생 (신용카드)
- boleto: 202회 발생 (브라질의 지불 수단)
- voucher: 35회 발생 (바우처)
- debit\_card: 12회 발생 (직불카드)

이 고유값들은 각 범주형 컬럼에서 나타나는 값들의 빈도와 분포를 보여줍니다.

### 📌 더 알아보기

- 데이터 요약, 통계정보를 통해 이상치 및 결측치에 대한 처리를 어떻게 해야할지 대략적인 감을 잡을 수 있음

## 3. 조건에 맞는 데이터 탐색

### 3.1 단일 조건으로 데이터 탐색

#### 💬 Chat Prompt

제시한 데이터에서 product\_category가 장난감인 것들만 탐색해줘

#### 🤖 GPT 출력: Code

```
# Filtering the dataset for rows where the product_category is 'toys'
toys_data = data[data['product_category'] == 'toys']

# Displaying the filtered data
print(toys_data.head())
```

	order_id	customer_id \
424	432aaf21d85167c2c86ec9448c4e42cc	3df704f53d3f1d4818840b34ec672a9f
425	4e2160ff7a61b2bf4fb92e527ea924d3	ea36ec29b56edeb534644f21b9561efc
426	588e9a3844482484a31e604c2767b82f	c6d04a9495cc4d8e22a4276131aac50e
427	0d2d1445891e7aba669fa2069d695601	739ee2d2d5960609ad780937fd9fa282
428	ce0e053019fe2441df8363b9bf4208a7	eb0e5e03030a71a038d3e71bdbe82d86

	order_status	order_purchase_timestamp	product_category	payment_method
\				
424	delivered	2018.3.1 14:14	toys	credit_card
425	delivered	2018.2.9 13:48	toys	credit_card

426	delivered	2018.3.24 21:50	toys	sigirace.github.io boleto
427	delivered	2017.10.9 19:54	toys	boleto
428	delivered	2017.12.4 20:15	toys	credit_card
installments item_price_usd shipping_cost_usd				
424	1	7.2675	3.0609	
425	1	7.2675	3.0609	
426	1	8.0731	9.9712	
427	1	7.2675	2.5650	
428	2	7.2675	5.1091	

### 3.2 다중 조건으로 데이터 탐색

#### 💬 Chat Prompt

제시한 데이터에서 product\_category가 장난감이고 installments이 평균 이상인 것들만 탐색해줘

#### 🤖 GPT 출력: Code

```
# Calculating the mean of 'installments'
installments_mean = data['installments'].mean()

# Filtering the dataset for rows where the product_category is 'toys' and
# installments are above the mean
toys_above_mean_installments = data[(data['product_category'] == 'toys') &
(data['installments'] > installments_mean)]

print(toys_above_mean_installments)
```

		order_id	customer_id	\
445	fa5927f07ada4f44128f6d9fab3b70f6	8af801d95c9bc9bd7cf7845826e4b72		
452	d9971834a50b8fc323aa7f527978169d	4899884819536b44b5b533849b3683ce		
453	d9971834a50b8fc323aa7f527978169d	4899884819536b44b5b533849b3683ce		
455	7c74b6a47ffad9deb24bd2a1f0b0acc5	95e4216612b16855852e2b9efa6174d2		
order_status order_purchase_timestamp product_category payment_method				
\				
445	delivered	2018.2.2 10:18	toys	credit_card
452	delivered	2017.9.27 0:07	toys	credit_card
453	delivered	2017.9.27 0:07	toys	credit_card
455	delivered	2017.12.4 17:22	toys	credit_card
installments item_price_usd shipping_cost_usd				
445	5	7.2675	3.4048	
452	10	7.2675	3.0609	
453	10	7.2675	3.0609	
455	5	7.2675	3.0609	

## 4. 데이터 요약

데이터를 요약하여 원하는 정보를 얻는 것은 데이터가 지닌 값의 특징을 알게 해줍니다. 이와 비슷한 것을 2장에서 수행해보았는데, 이번 장에서는 세부적으로 데이터를 요약하여 집계하는 방법을 알아보겠습니다.

### 4.1 그룹화 집계

#### 👀 Definition

그룹화는 하나 이상의 데이터를 조건에 따라 여러개 그룹으로 묶는 것

- groupby() 함수를 사용함

#### 📌 **groupby**

`DataFrame.groupby(by=None, axis=0, sort=False, dropna=True).FUN()`

- by: 그룹을 결정하는 데 사용
- axis: 행(0)과 열(1) 지정
- sort: 집계된 내용을 정렬할지 여부
- dropna: True이면 결측값이 행/열과 함께 삭제, False이면 결측값도 그룹의 키로 처리
- FUN: 집계함수

#### 🔥 집계함수

- count(): 값의 개수
- sum(): 값들의 합
- std(): 표준편차
- var(): 분산
- min(): 최솟값
- max(): 최대값
- mean(): 평균값
- median(): 중앙값

💻 : product\_category 별로 다양한 집계를 수행해 봅니다.

#### 💬 Chat Prompt

product\_category 값을 기준으로 집계 함수 count를 수행해줘

#### 🤖 GPT 출력: Message

product\_category 값을 기준으로 각 카테고리의 주문 수를 집계한 결과는 다음과 같습니다:

- health\_beauty: 278
- garden\_tools: 219
- perfumery: 125
- auto: 91
- bed\_bath\_table: 52

- stationery: 41
- baby: 40
- toys: 35
- watches\_gifts: 27
- computers\_accessories: 23
- telephony: 19
- electronics: 13
- office\_furniture: 9
- cool\_stuff: 6
- pet\_shop: 5
- furniture\_decor: 5
- housewares: 4
- construction\_tools\_construction: 4
- air\_conditioning: 1

이 결과는 각 카테고리에 속한 주문의 빈도를 보여줍니다.

#### GPT 출력: Code

```
# product_category를 기준으로 각 카테고리의 주문 수를 집계
category_count = data['product_category'].value_counts()

category_count
```

health_beauty	278
garden_tools	219
perfumery	125
auto	91
bed_bath_table	52
stationery	41
baby	40
toys	35
watches_gifts	27
computers_accessories	23
telephony	19
electronics	13
office_furniture	9
cool_stuff	6
pet_shop	5
furniture_decor	5
housewares	4
construction_tools_construction	4
air_conditioning	1

Name: product\_category, dtype: int64

#### Chat Prompt

product\_category 값을 기준으로 installments의 평균을 구해줘 (집계함수 mean 적용)

### 🤖 GPT 출력: Message

product\_category 값을 기준으로 installments의 평균을 계산한 결과는 다음과 같습니다:

- air\_conditioning: 0.00
- auto: 2.09
- baby: 3.23
- bed\_bath\_table: 1.75
- computers\_accessories: 2.13
- construction\_tools\_construction: 10.00
- cool\_stuff: 3.67
- electronics: 2.15
- furniture\_decor: 0.60
- garden\_tools: 2.71
- health\_beauty: 6.60
- housewares: 1.25
- office\_furniture: 2.11
- perfumery: 2.78
- pet\_shop: 1.40
- stationery: 1.10
- telephony: 1.74
- toys: 1.89
- watches\_gifts: 4.11

이 값들은 각 상품 카테고리별 할부 횟수의 평균을 나타냅니다.

### 🤖 GPT 출력: Code

```
# product_category를 기준으로 installments의 평균을 계산
installments_mean_by_category = data.groupby('product_category')
['installments'].mean()

installments_mean_by_category
```

product_category	
air_conditioning	0.000000
auto	2.087912
baby	3.225000
bed_bath_table	1.750000
computers_accessories	2.130435
construction_tools_construction	10.000000
cool_stuff	3.666667
electronics	2.153846
furniture_decor	0.600000
garden_tools	2.712329

health_beauty	6.597122
housewares	1.250000
office_furniture	2.111111
perfumery	2.776000
pet_shop	1.400000
stationery	1.097561
telephony	1.736842
toys	1.885714
watches_gifts	4.111111

## 💬 Chat Prompt

집계 함수로 구한 product\_category 값을 기준으로 installments의 평균에 대해 결과를 해석해 서 설명해줘

## 🤖 GPT 출력: Message

각 product\_category(상품 카테고리)별로 installments(할부 횟수)의 평균을 분석한 결과는 다음과 같습니다:

### 1. 높은 할부 횟수:

- construction\_tools\_construction (10.00): 이 카테고리에서는 평균 10회의 할부가 적용되었습니다. 이는 고가의 상품이 포함될 가능성이 높으며, 구매자들이 비용을 분할하여 지불하는 경향이 강하다는 것을 나타냅니다.

- health\_beauty (6.60): 건강 및 미용 제품도 비교적 높은 할부 횟수를 보입니다. 이는 고객들이 이 카테고리의 상품을 구매할 때 할부를 자주 선택한다는 것을 의미합니다.

### 2. 중간 할부 횟수:

- watches\_gifts (4.11), cool\_stuff (3.67), baby (3.23): 이 카테고리들은 중간 수준의 할부 횟수를 가지고 있으며, 이는 구매자들이 이러한 제품을 구매할 때 어느 정도의 분할 지불을 선호한다는 것을 보여줍니다.

- perfumery (2.78), auto (2.09): 이 카테고리들도 중간 정도의 할부 횟수를 보입니다. 특히 향수(perfumery)와 자동차 관련 상품(auto)도 일정 수준의 할부를 통해 구매된다는 것을 알 수 있습니다.

### 3. 낮은 할부 횟수:

- stationery (1.10), housewares (1.25), telephony (1.74): 문구류, 가정용품, 전화기 관련 제품들은 평균적으로 낮은 할부 횟수를 보입니다. 이는 이들 제품이 비교적 저렴하거나, 구매자들이 할부 대신 일시불로 지불하는 경우가 많다는 것을 시사합니다.

- air\_conditioning (0.00): 에어컨 관련 카테고리의 경우, 할부가 전혀 선택되지 않았습니다. 이는 이 데이터셋에서 해당 제품이 매우 적거나, 일시불로 지불된다는 것을 의미할 수 있습니다.

### 결론:

이 결과는 카테고리별로 할부 지불 방식이 다르며, 고가의 상품일수록 할부 횟수가 높아지는 경향이 있음을 보여줍니다. 반면, 저렴하거나 상대적으로 일상적인 제품들은 대부분 낮은 할부 횟수로 구매되거나 일시불로 처리됩니다. 이는 고객의 구매 패턴과 선호도를 파악하는 데 유용한 정보입니다.

## 5. 데이터 정렬하기

## 5.1 컬럼의 값에 대해 데이터 정렬하기

### 💬 Chat Prompt

order\_purchase\_timestamp을 기준으로 오름차순 정렬을 수행해줘

### 🤖 GPT 출력: Message

order\_purchase\_timestamp를 기준으로 오름차순 정렬을 수행한 결과, 가장 오래된 주문부터 순차적으로 정렬되었습니다. 상위 5개의 데이터는 다음과 같습니다:

1. 2017.1.23 18:29 – furniture\_decor 카테고리, boleto 결제, 할부 없음.
2. 2017.1.30 10:47 – auto 카테고리, credit\_card 결제, 할부 없음.
3. 2017.1.31 17:40 – auto 카테고리, credit\_card 결제, 1회 할부.
4. 2017.10.1 20:04 – health\_beauty 카테고리, credit\_card 결제, 1회 할부.
5. 2017.10.1 09:15 – auto 카테고리, boleto 결제, 1회 할부.

이 데이터를 통해 시간 순서대로 주문 내역을 확인할 수 있습니다.

### 🤖 GPT 출력: Code

```
# order_purchase_timestamp을 기준으로 오름차순 정렬 수행
data_sorted_by_timestamp = data.sort_values(by='order_purchase_timestamp',
                                             ascending=True)

# 정렬된 데이터 확인 (상위 5개)
data_sorted_by_timestamp.head()
```

	order_id	customer_id	\	
248	76c6e866289321a7c93b82b54852dc33	f54a9f0e6b351c431402b8461ea51999		
189	9559e0b00c67d159a266152428f85615	c3498e41d1549d56c4cbde7cbfdabeb3		
244	35aa6b03b34d8d0a5c1eaae2bb5fcf76	18f5e4ab94b6b5fdcf32dade57defe50		
511	80dabd6ee5b0e0d7d5672d511ca8cc7e	fbea4ed949b70f597425d14164f82851		
247	3389e2333066b728a245423a73f3f82c	45c155ac404d868fd8f1d07e8cd916c3		
	order_status	order_purchase_timestamp	product_category	payment_method
\				
248	delivered	2017.1.23 18:29	furniture_decor	boleto
189	delivered	2017.1.30 10:47	auto	credit_card
244	delivered	2017.1.31 17:40	auto	credit_card
511	delivered	2017.10.1 20:04	health_beauty	credit_card
247	delivered	2017.10.1 9:15	auto	boleto
	installments	item_price_usd	shipping_cost_usd	
248	0	3.7810	3.0495	
189	0	28.1010	3.4200	
244	1	9.4981	2.7588	

511	1	17.0981	3.1141
247	1	NaN	2.8690

## 📌 더 알아보기

- 정렬을 수행한 후에는 인덱스가 섞인 상태
- 인덱스가 섞이면 추후 분석과정에서 오류가 발생할 수 있으므로 인덱스 또한 정렬이 필요
- 이때, 값은 건들이지 않고 인덱스만을 정렬함

## 💬 Chat Prompt

인덱스를 정렬해주고, 과거의 인덱스는 버려줘

## 🤖 GPT 출력: Message

이제 데이터는 `order_purchase_timestamp`를 기준으로 오름차순 정렬되었으며, 인덱스도 새로운 순서에 맞게 재설정되었습니다. 과거의 인덱스는 제거되었으며, 현재 데이터는 정렬된 순서에 맞는 새로운 인덱스를 가지고 있습니다.

## 🤖 GPT 출력: Code

```
# 인덱스를 정렬된 데이터의 순서에 맞게 재설정하고, 이전 인덱스는 제거
data_sorted_by_timestamp.reset_index(drop=True, inplace=True)

# 재설정된 인덱스를 가진 데이터 확인 (상위 5개)
data_sorted_by_timestamp.head()
```

```
# 인덱스를 정렬된 데이터의 순서에 맞게 재설정하고, 이전 인덱스는 제거
data_sorted_by_timestamp.reset_index(drop=True, inplace=True)

# 재설정된 인덱스를 가진 데이터 확인 (상위 5개)
print(data_sorted_by_timestamp.head())
```

	order_id	customer_id \
0	f54a9f0e6b351c431402b8461ea51999	76c6e866289321a7c93b82b54852dc33
1	c3498e41d1549d56c4cbde7cbfdabeb3	9559e0b00c67d159a266152428f85615
2	18f5e4ab94b6b5fdcf32dade57defe50	35aa6b03b34d8d0a5c1eaae2bb5fcf76
3	fbea4ed949b70f597425d14164f82851	80dabd6ee5b0e0d7d5672d511ca8cc7e
4	45c155ac404d868fd8f1d07e8cd916c3	3389e2333066b728a245423a73f3f82c

	order_status	order_purchase_timestamp	product_category	payment_method \
0	delivered	2017.1.23 18:29	furniture_decor	boleto
1	delivered	2017.1.30 10:47	auto	credit_card
2	delivered	2017.1.31 17:40	auto	credit_card

```
3     delivered      2017.10.1 20:04    health_beauty
4     delivered      2017.10.1 9:15       auto

```

	installments	item_price_usd	shipping_cost_usd
0	0	3.7810	3.0495
1	0	28.1010	3.4200
2	1	9.4981	2.7588
3	1	17.0981	3.1141
4	1	NaN	2.8690

## 6. DataFrame에 함수 적용하기

Apply와 Map 함수를 사용하여 데이터의 행 또는 열에 자유롭게 함수를 적용하고 값을 변경할 수 있습니다.

- apply/ map 둘 다 동일한 기능 수행
- 데이터프레임에 행 또는 열 방향으로 지정한 함수 실행

### 🔥 꿀팁

- lambda는 apply를 시작할 때 "lambda 뒤에 오는 변수를 가지고 수행할거야!"라는 뜻
- 이 변수는 하나의 컬럼일 수도 있고 여러 컬럼일 수도 있으며 다양한 처리를 수행할 수 있음
- 대상이 여러 컬럼일 경우 axis=1을 꼭 명시해야함

### 6.1 하나의 컬럼에 대한 수행

#### 💬 Chat Prompt

데이터프레임에 `installment_divide`라는 컬럼을 추가해줘  
`installment_divide`라는 `installment` 컬럼을 12로 나눈 값이야, `apply`를 적용해서 수행해줘

#### 🤖 GPT 출력: Message

데이터프레임에 `installment_divide`라는 컬럼이 추가되었으며, `installments` 컬럼의 값을 12로 나눈 결과가 해당 컬럼에 저장되었습니다.  
예를 들어, `installments` 값이 1인 경우 `installment_divide` 값은 `0.0833`입니다.

#### 🤖 GPT 출력: Code

```
# installment_divide라는 컬럼을 추가하여 installments 컬럼의 값을 12로 나눈 값을 할당
data_sorted_by_timestamp['installment_divide'] =
data_sorted_by_timestamp['installments'].apply(lambda x: x / 12)

# 추가된 컬럼 확인 (상위 5개)
print(data_sorted_by_timestamp[['installments', 'installment_divide']].head())
```

	installments	installment_divide
0	0	0.000000
1	0	0.000000
2	1	0.083333
3	1	0.083333
4	1	0.083333

## 6.2 여러 컬럼에 대한 수행

### 💬 Chat Prompt

데이터 프레임에 monthly\_price 컬럼을 만들어줘  
 installment\_divide 컬럼과 item\_price\_usd를 곱한 값이야, apply를 적용해서 수행해줘

### 🤖 GPT 출력: Message

데이터프레임에 monthly\_price라는 컬럼이 추가되었습니다. 이 컬럼은 installment\_divide와 item\_price\_usd를 곱한 값으로 계산되었습니다. 예를 들어, installments 값이 1이고 item\_price\_usd가 9.4981인 경우 monthly\_price 값은 0.791508입니다.

이제 각 상품의 할부 금액에 해당하는 월별 가격을 확인할 수 있습니다.

### 🤖 GPT 출력: Code

```
# monthly_price 컬럼을 추가하여 installment_divide와 item_price_usd를 곱한 값을 할당
data_sorted_by_timestamp['monthly_price'] =
data_sorted_by_timestamp.apply(
    lambda row: row['installment_divide'] * row['item_price_usd'], axis=1
)

# 추가된 컬럼 확인 (상위 5개)
print(data_sorted_by_timestamp[['installment_divide', 'item_price_usd',
  'monthly_price']].head())
```

```
# monthly_price 컬럼을 추가하여 installment_divide와 item_price_usd를 곱한 값을 할당
data_sorted_by_timestamp['monthly_price'] =
data_sorted_by_timestamp.apply(
    lambda row: row['installment_divide'] * row['item_price_usd'], axis=1
)

# 추가된 컬럼 확인 (상위 5개)
```

```
print(data_sorted_by_timestamp[['installment_divide', 'item_price_usd',  
'monthly_price']].head())
```

## 7. 데이터 저장

### 💬 Chat Prompt

현재까지 만들어진 데이터를 다운로드 받을 수 있는 링크를 만들어줘

### 🤖 GPT 출력: Message

데이터를 다운로드할 수 있는 링크를 생성했습니다. 아래 링크를 클릭하여 데이터를 다운로드할 수 있습니다:

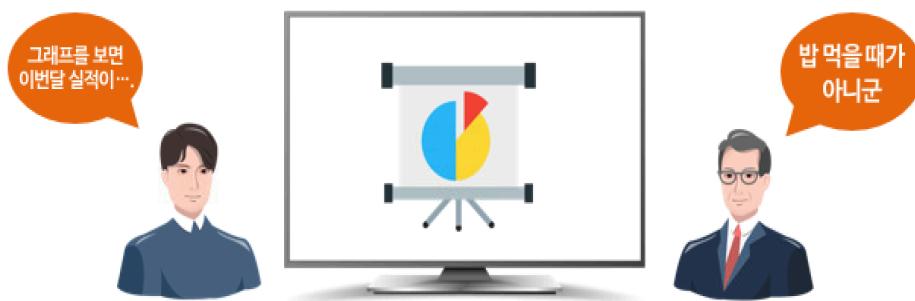
다운로드 링크

# 🔥 Warm Up 🔥

## 데이터 위주의 보고



## 데이터 시각화를 통한 보고



### 💡 시각화를 통한 보고와 EDA(탐색적 데이터 분석)를 통한 전처리 단계 비교

특징	시각화를 통한 보고	EDA(탐색적 데이터 분석)를 통한 전처리
목적	데이터의 주요 정보를 시각적으로 전달	데이터의 구조와 특성을 이해하고, 분석을 위한 데이터 준비
주요 활동	그래프와 차트 작성, 데이터 패턴과 추세 시각화	데이터 분포 확인, 이상치 탐지, 변수 간 관계 파악
사용 도구	다양한 차트 및 그래프 도구 (예: 막대 그래프, 파이 차트)	통계적 방법 및 다양한 시각화 기법 (예: 히스토그램, 상자 그림)
결과물	인사이트와 결론 도출, 의사결정에 필요한 시각적 자료	데이터 정제 및 변환, 분석에 필요한 데이터셋 구축
주요 대상	경영진, 비즈니스 이해 관계자, 의사결정권자	데이터 분석가, 데이터 과학자
데이터 분석 범위	요약된 데이터나 주요 메트릭 중심	전체 데이터셋을 대상으로 한 상세 분석
시각화 목적	데이터를 이해하기 쉽고 명확하게 전달	데이터의 특성과 문제점을 파악하고, 분석 방향 설정
결과 활용	보고서, 프레젠테이션, 비즈니스 전략 수립	데이터 모델링 의사결정, 추가 분석 작업, 데이터 품질 향상
초점	주요 결과와 KPI(Key Performance Indicator)에 집중	데이터 탐색, 이상치 식별, 데이터 분포 및 상관 관계 분석

특징	시각화를 통한 보고	EDA(탐색적 데이터 분석)를 통한 전처리
예시	매출 현황 보고, 시장 점유율 그래프, 성과 리뷰	결측값 처리, 데이터 변환, 특성 공학(feature engineering)
시간 투자	상대적으로 짧고 주기적인 분석	초기 분석 단계에서 상당한 시간 투자, 데이터 이해를 위한 노력

## ● Definition

EDA란 데이터분석의 첫 번째 단계로, 데이터의 특징과 데이터에 내재된 관계를 알아내기 위해 통계적 분석과 그래프를 활용해 탐구하는 것

- 분석하고자 하는 데이터의 유형과 범위, 수준을 확인
- 데이터의 분포를 통해 의미를 추출
- 분석 목적을 달성하기 위한 적절한 알고리즘을 선택하기 위한 자료로 사용됨

 EDA와 시각화

💡 전처리와 EDA의 관계

이전 고급 데이터 분석 파트에서 데이터 분석 프로세스를 아래와 같이 EDA가 전처리 이후라고 소개드린적이 있습니다. 그런데 사실 데이터 전처리와 EDA는 서로 병행하며 이루어지는 경우가 많습니다. 이상치, 데이터의 불균형 등 시각화를 통해 명확히 파악할 수 있는 전처리 방식들이 있기 때문입니다. 따라서 이전 데이터 셋은 잠시 묵혀두고 전처리를 수행하기 위해 새로운 데이터셋으로 EDA를 맛보는 과정을 살펴보도록 하겠습니다.

🔗 한국어 폰트 다운로드

-  Korea matplotlib
-  나눔고딕 폰트

💬 Chat Prompt

가상환경에 한국어를 사용할 수 있도록 whl 파일과 ttf 파일을 제공하였으니, 아래 과정을 수행하여 한국어 환경 설정 수행

- Code interpreter로 제공한 파일들을 다운로드 및 수행
- 가상환경에 설치 및 한국어를 사용할 수 있도록 설정
- 설치 완료후 테스트 그래프 제시
- 모든 설명은 한국어로 진행

🤖 GPT 출력: Code

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

# 폰트 파일 경로
font_path =
'/Users/sigi/sigirace.github.io/_posts/kang_lecture/gpt_ada/data/NanumBarunGothic.ttf'

# 폰트 등록
fm.fontManager.addfont(font_path)

# 한글 폰트 설정
plt.rc('font', family='NanumBarunGothic')
```

 : 아래의 데이터를 다운로드 받아 GPT에 넣어주세요!

🔗 실습 데이터

-  인사 정보 데이터

💬 Chat Prompt

해당 데이터는 인사 정보가 담긴 데이터이다. 이 데이터의 컬럼을 살펴보고 설명해라. 모든 과정에서 설명시 한국어를 사용하라.

### 🤖 GPT 출력: Message

주어진 데이터는 인사 정보가 담긴 설문 조사 데이터로, 다음과 같은 컬럼들로 구성되어 있습니다:

1. ID: 각 직원의 고유 식별자.
2. 이름: 직원의 이름.
3. 입사경로: 직원이 회사에 입사하게 된 경로.
4. 성별: 직원의 성별.
5. 나이: 직원의 현재 나이.
6. 근속연수: 회사에서 근무한 연수.
7. 대학: 직원이 졸업한 대학.
8. 대학 구분: 졸업한 대학의 구분 (예: 일반대학, 전문대학 등).
9. 학점: 대학에서 받은 학점.
10. 학력: 직원의 학력 수준.
11. 입사년도: 회사에 입사한 연도.
12. 직무: 회사에서 담당하는 직무.
13. 소속 팀: 현재 속해있는 팀.
14. 퇴직 여부: 직원의 퇴직 여부를 나타내는 컬럼.
15. 관리자 여부: 관리자인지 여부를 나타내는 컬럼.
16. 초과 근로 시간: 초과 근무 시간의 총량.
17. 성과 평가 점수: 직원의 성과 평가 점수.
18. 미사용 휴가: 사용하지 않은 휴가 일수.
19. 연봉: 직원의 연봉.
20. 근무지: 직원의 근무지 위치.
21. 인지능력 평가: 직원의 인지능력에 대한 평가 점수.
22. 인내심 평가: 직원의 인내심에 대한 평가 점수.
23. 실행력 평가: 직원의 실행력에 대한 평가 점수.
24. 매니저의 비전: 매니저의 비전 평가 점수.
25. 매니저의 커뮤니케이션 능력: 매니저의 커뮤니케이션 능력 평가 점수.
26. 매니저의 실행력: 매니저의 실행력 평가 점수.
27. 직업 만족도: 직원의 직업 만족도 점수.
28. 달성 매출: 직원이 달성한 매출.
29. 불량 발생 횟수: 직원이 일으킨 불량 발생 횟수.

이 컬럼들은 직원의 기본 정보부터 성과, 만족도, 매니저 평가까지 다양한 측면에서의 정보를 담고 있습니다.

### 🤖 GPT 출력: Code

```
import pandas as pd

# 엑셀 파일 불러오기
file_path = '/mnt/data/employee satisfaction survey.xlsx'
data = pd.read_excel(file_path)
```

```
# 데이터의 컬럼 확인
data.columns
```

```
Index(['ID', '이름', '입사경로', '성별', '나이', '근속연수', '대학', '대학 구분', '학점', '학력',
       '입사년도', '직무', '소속 팀', '퇴직 여부', '관리자 여부', '초과 근로 시간', '성과 평가 점수',
       '미사용 휴가', '연봉', '근무지', '인지능력 평가', '인내심 평가', '실행력 평가', '매니저의 비전',
       '매니저의 커뮤니케이션 능력', '매니저의 실행력', '직업 만족도', '달성 매출', '불량 발생 횟수'],
      dtype='object')
```

## 1. 막대그래프와 히스토그램

### 1.1 막대그래프

막대그래프는 범주형 데이터를 요약하고 시각적으로 비교하는 데 효과적인 그래프입니다. 막대그래프를 사용하기 위해서는 도수분포표 혹은 집계함수를 통해 범주형 데이터에 대한 데이터 셋을 만들어야 합니다.

#### 💬 Chat Prompt

퇴직 여부를 기준으로 count 집계함수를 수행해 주고 막대그래프로 시각화 수행

#### 🤖 GPT 출력: Message

퇴직 여부에 따른 인원 수를 집계하여 막대그래프로 시각화하였습니다. 그래프에서 퇴직한 인원과 퇴직하지 않은 인원의 수를 한눈에 비교할 수 있습니다.

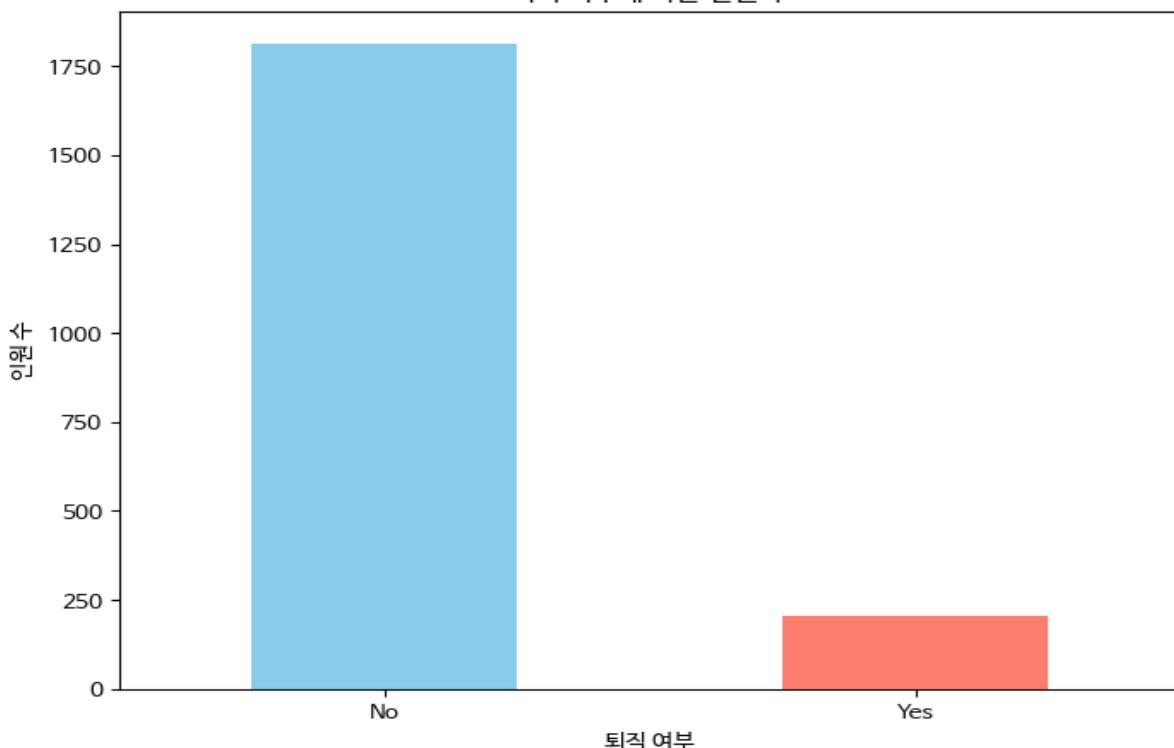
#### 🤖 GPT 출력: Code

```
import matplotlib.pyplot as plt

# 퇴직 여부를 기준으로 count 집계
retirement_counts = data['퇴직 여부'].value_counts()

# 막대그래프로 시각화
plt.figure(figsize=(8, 6))
retirement_counts.plot(kind='bar', color=['skyblue', 'salmon'])
plt.title('퇴직 여부에 따른 인원 수')
plt.xlabel('퇴직 여부')
plt.ylabel('인원 수')
plt.xticks(rotation=0)
plt.show()
```

퇴직 여부에 따른 인원 수



### 📌 더 알아보기: 막대 그래프 활용

- 막대 그래프는 각 범주의 값의 개수 차이를 비교하고 개수 차이가 극단적인지 확인할 때 주로 쓰임
- 분류 문제에서 타겟의 분포 차이가 많이 나는가를 검증

## 1.2 히스토그램

히스토그램은 연속형 자료에 대한 도수분포표를 시각화하여 나타낸 것으로 서로 겹치지 않는 특정 구간에 따른 데이터의 빈도 수를 표현합니다. 연속형 데이터이기 때문에 각 구간은 서로 연속되고, 막대는 서로 인접하여있습니다.

### 💬 Chat Prompt

연봉 컬럼에 대해서 히스토그램을 통해 시각화 수행

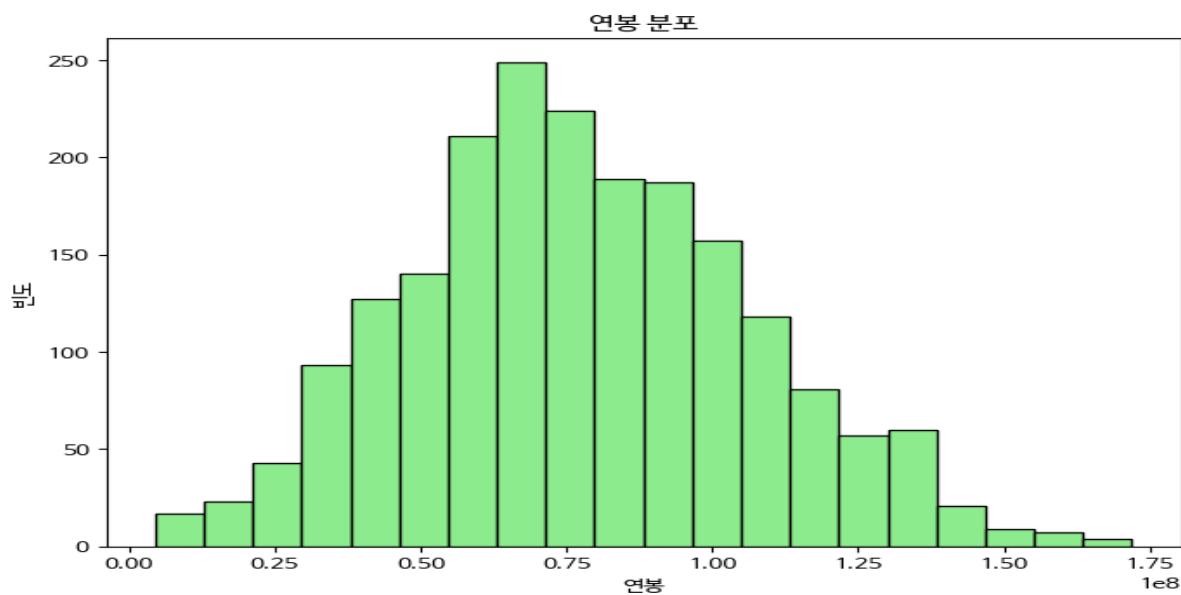
### 🤖 GPT 출력: Message

연봉 컬럼에 대한 히스토그램을 통해 연봉 분포를 시각화했습니다. 이 그래프를 통해 직원들의 연봉이 어떻게 분포되어 있는지 확인할 수 있습니다.

### 🤖 GPT 출력: Code

```
# 연봉 컬럼에 대한 히스토그램 시각화
plt.figure(figsize=(8, 6))
plt.hist(data['연봉'].dropna(), bins=20, color='lightgreen',
edgecolor='black')
plt.title('연봉 분포')
```

```
plt.xlabel('연봉')
plt.ylabel('빈도')
plt.show()
```



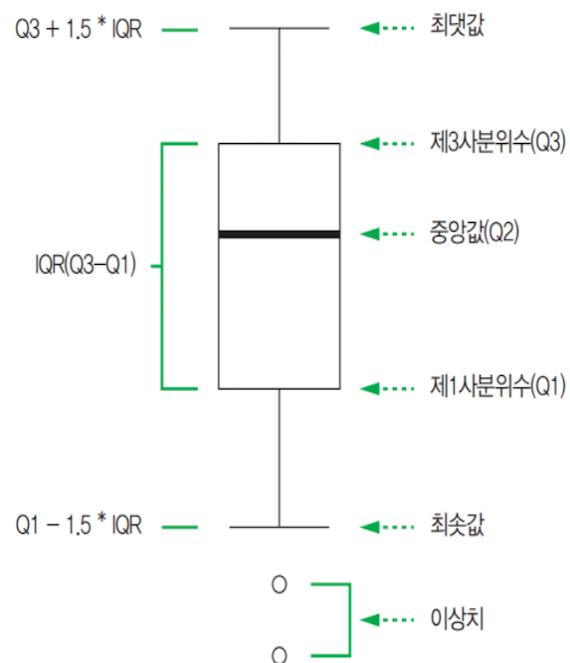
### 📌 더 알아보기: 히스토그램 활용

- 히스토그램은 데이터의 분포를 알아보는데 사용
- 연속형 데이터의 분포를 통해 특정 값에 치우쳐져 있는지, 극단적인 범위의 값이 있는지 확인

## 2. 상자-수염 그림(Box plot)

Box Plot은 사분위수를 이용하여 수치형 변수의 값의 분포를 확인하는 그래프입니다. 상자에 수염같은 선이 붙어있다고 하여 상자 수염이라고도 불립니다. 상자의 크기, 중앙값 선의 위치, 수염의 길이를 통해 값의 분포와 대칭 정도, 이상치까지 한 컬럼의 값에 대한 다양한 정보를 한눈에 확인할 수 있습니다.

### 💡 상자-수염 그래프의 해석



## 1. 상자

- 데이터 값의 50%가 해당함
- 상자의 아랫면과 윗면은 각 25%(1사분위수), 75%(3사분위수) 위치에 존재하는 값
- 상자 중앙의 두꺼운 선은 중앙값을 의미함
- 중앙값의 위치에 따라 값이 어디에 치우쳐져 있는지 확인할 수 있음

## 2. IQR

- 중앙 50%데이터가 펴진 정도
- 3사분위수와 1사분위수의 차이로 구할 수 있음
- 통계학적으로 이상치를 판단하는 수식에 사용됨

## 3. 수염

- $1.5 * \text{IQR}$  범위인  $Q1 - 1.5 * \text{IQR}$  부터  $Q3 + 1.5 * \text{IQR}$  까지의 범위를 수염으로 지정 (1.5가 아닌 다른 값으로도 사용 가능)
- 수염을 벗어나는 값은 이상치로 판단

💡 : Box plot은 수치형 데이터에 대해서 적용하기에 자료 형태를 구분한 뒤 수행하겠습니다.

### 💬 Chat Prompt

제시한 데이터에서 컬럼을 수치형, 범주형으로 구분하여 알려주고 리스트에 저장해줘

### 🤖 GPT 출력: Message

제시된 데이터에서 컬럼을 수치형과 범주형으로 구분한 결과는 다음과 같습니다:

수치형 컬럼:

- 나이
- 근속연수
- 학점
- 입사년도
- 초과 근로 시간
- 성과 평가 점수
- 미사용 휴가
- 연봉
- 인지능력 평가
- 인내심 평가
- 실행력 평가
- 매니저의 비전
- 매니저의 커뮤니케이션 능력
- 매니저의 실행력
- 직업 만족도
- 달성 매출
- 불량 발생 횟수

범주형 컬럼:

- 이름
- 입사경로
- 성별
- 대학
- 대학 구분
- 학력
- 직무
- 소속 팀
- 퇴직 여부
- 관리자 여부
- 근무지

이 컬럼들은 각각 수치형 변수와 범주형 변수로 리스트에 저장되어 있습니다.

### GPT 출력: Code

```
# 수치형 컬럼과 범주형 컬럼으로 구분
numeric_columns = data.select_dtypes(include=['float64',
'int64']).columns.drop('ID').tolist()
categorical_columns = data.select_dtypes(exclude=['float64',
'int64']).columns.tolist()

# 결과 출력
numeric_columns, categorical_columns
```

### 꿀팁

- 데이터의 스케일이 다르다면 하나의 화면에 모든 컬럼에 대한 시각화보다는 개별적인 시각화가 파악이 쉬움
- 개별적으로 그리되 하나의 화면에 그릴 수 있는 그리드를 활용

### Chat Prompt

수치형 컬럼들에 대해서 개별적으로 box plot 시각화 수행

단, box plot은 개별적으로 수행하 되, 하나의 그리드에 표현

### GPT 출력: Message

모든 수치형 컬럼에 대해 개별적으로 Box Plot을 생성하고, 이를 하나의 그리드에 표시했습니다. 이렇게 하면 각 변수의 분포와 이상치를 비교하여 볼 수 있습니다.

### GPT 출력: Code

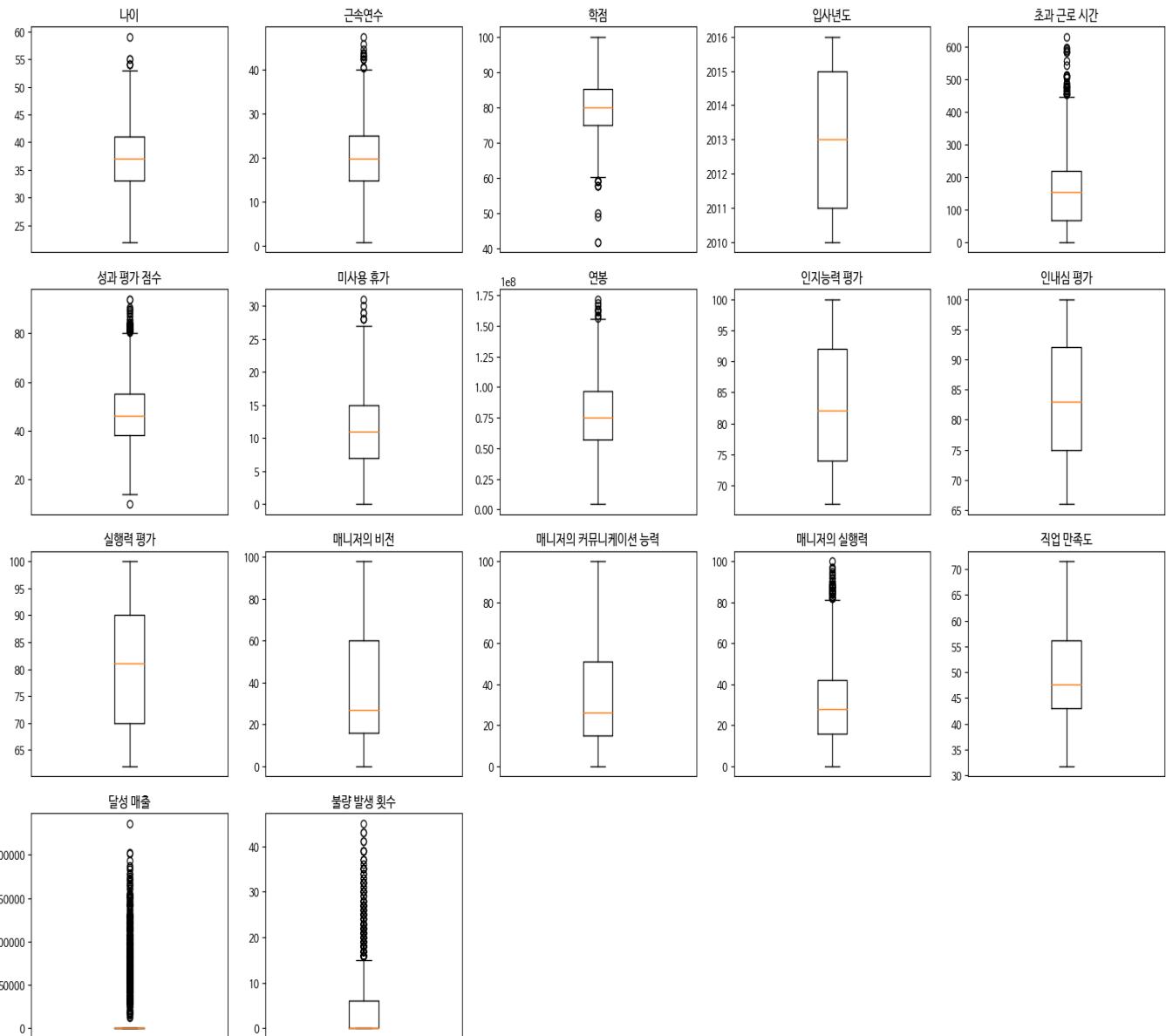
```
import matplotlib.pyplot as plt

# 수치형 컬럼들에 대해 Box plot을 하나의 그리드에 표현
```

```
plt.figure(figsize=(18, 12))

for i, column in enumerate(numeric_columns, 1):
    plt.subplot(4, 5, i) # 4x5 그리드 설정
    plt.boxplot(data[column].dropna())
    plt.title(f'{column}')
    plt.xticks([])

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



### 💬 Chat Prompt

IQR 관점에서 이상치가 있는 컬럼들을 알려줘

### 🤖 GPT 출력: Message

IQR 관점에서 이상치가 있는 것으로 확인된 컬럼들은 다음과 같습니다:

- 나이
- 근속연수
- 학점
- 초과 근로 시간
- 성과 평가 점수
- 미사용 휴가
- 연봉
- 매니저의 실행력
- 달성 매출
- 불량 발생 횟수

이 컬럼들은 IQR을 기준으로 이상치가 존재하는 컬럼들입니다.

#### 🤖 GPT 출력: Code

```
# IQR을 계산하여 이상치가 있는 컬럼을 찾는 함수
def detect_outliers_iqr(df, columns):
    outliers = []
    for column in columns:
        Q1 = df[column].quantile(0.25)
        Q3 = df[column].quantile(0.75)
        IQR = Q3 - Q1
        lower_bound = Q1 - 1.5 * IQR
        upper_bound = Q3 + 1.5 * IQR
        if df[(df[column] < lower_bound) | (df[column] >
upper_bound)].shape[0] > 0:
            outliers.append(column)
    return outliers

# 이상치가 있는 컬럼 확인
outlier_columns = detect_outliers_iqr(data, numeric_columns)
outlier_columns
```

```
[ '나이',
  '근속연수',
  '학점',
  '초과 근로 시간',
  '성과 평가 점수',
  '미사용 휴가',
  '연봉',
  '매니저의 실행력',
  '달성 매출',
  '불량 발생 횟수' ]
```

#### 📌 더 알아보기: 이상치의 기준 선정

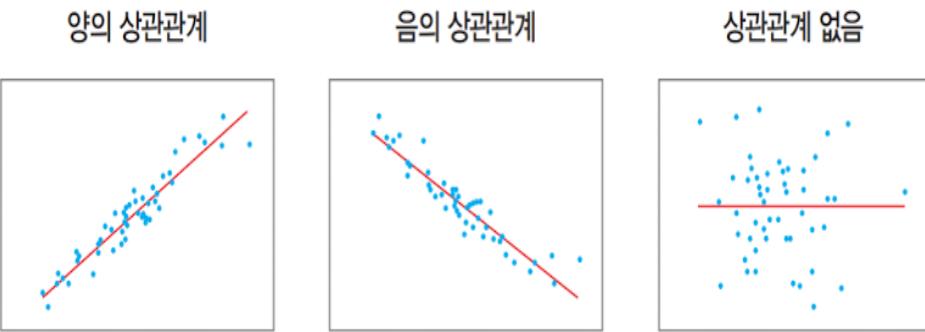
- 이상치는 업무 도메인과 밀접하게 연관되어 있음
- 확연히 범위를 벗어나는 이상치가 아닌 경우 도메인 전문가와 처리 방안에 대한 협의가 필요함
- 특히 연속적인 이상치가 나타나는 경우는 이상치가 아닐 확률이 높음
- 도메인 전문가의 도움을 받지 못하는 상황에서는 보수적으로 이상치를 제거 (ex, 상위 5%, 가장 큰 값, ...)
- IQR 만으로 제거한다면 너무 많은 데이터를 제거할 수 있음

### 3. 산점도

산점도는 두 개의 수치형 변수 각각의 분포와 함께 두 변수의 관계를 확인하는 가장 기본적인 그래프입니다. 두 개의 축을 가진 2차원 도표 안에 점들이 흩어져 있는 형태입니다.

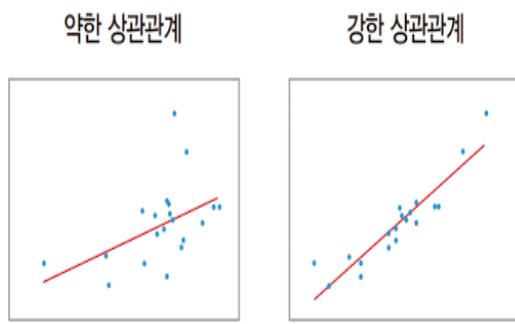
#### 💡 산점도 해석

##### 1. 관계의 유형



- 산점도는 두 변수의 관계 유형과 강도를 판단
- 관계의 유형은 점들이 흩어져 있는 모양을 보고 판단

##### 2. 관계의 강도



- 데이터가 적합선에 얼마나 가깝게 모여 있는지를 평가하여 두 변수 간 관계의 강도를 추정
- 산점도로는 관계의 해석을 수행하고, 상관관계 분석을 수행하여 강도를 수치화할 수 있음

💻 : 산점도는 모든 수치형 컬럼을 비교하기에 수행 시간이 길 수 있습니다.

#### 💬 Chat Prompt

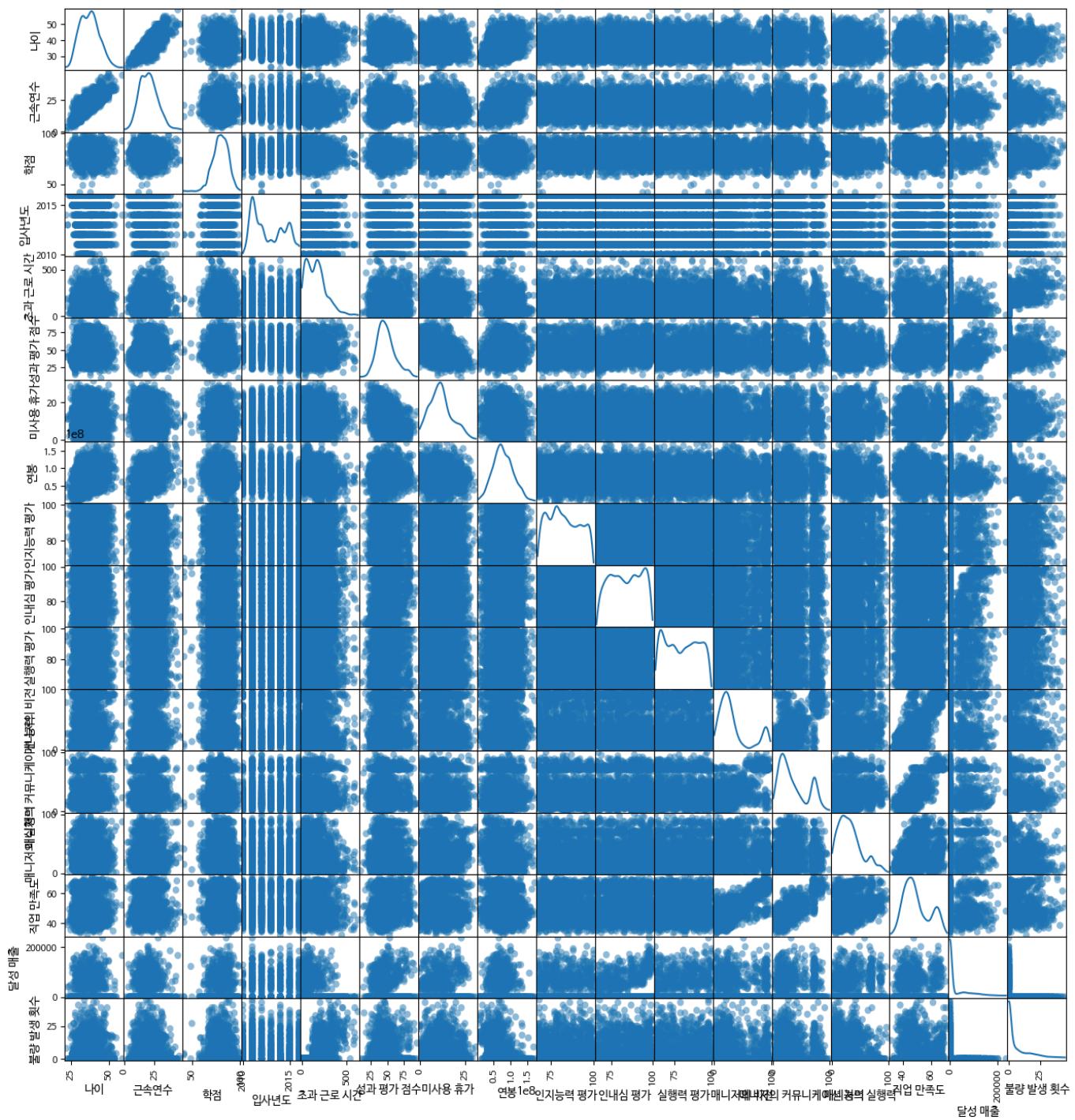
수치형 데이터들에 대한 산점도 시각화를 수행

 GPT 출력: Message

수치형 데이터들에 대한 산점도를 시각화했습니다. 이 산점도를 통해 각 수치형 변수들 간의 관계를 시각적으로 확인할 수 있습니다. 대각선에는 각 변수의 밀도 함수가 표시되어 있으며, 각 변수 쌍 사이의 관계를 점으로 표현한 그래프들이 나와 있습니다.

 GPT 출력: Code

```
# 수치형 데이터들에 대한 산점도 시각화
pd.plotting.scatter_matrix(data[numerical_columns], figsize=(16, 16),
diagonal='kde', marker='o')
plt.suptitle("수치형 데이터들에 대한 산점도 시각화", fontsize=16)
plt.show()
```



### 💬 Chat Prompt

산점도 수행결과를 보고 주목할만한 관계의 유형과 관계의 강도를 설명해줘

### 🤖 GPT 출력: Message

산점도 행렬을 통해 수치형 변수들 간의 관계를 시각적으로 확인한 결과, 몇 가지 주목할 만한 관계와 그 강도에 대해 설명하겠습니다:

1. 연봉과 성과 평가 점수:
  - 관계 유형: 양의 선형 관계.
  - 관계 강도: 강한 관계.
  - 설명: 연봉이 높을수록 성과 평가 점수도 높아지는 경향이 있습니다. 이는 성과가 우수한 직원들이 더 높은 연봉을 받는다는 것을 시사합니다.
2. 연봉과 근속연수:
  - 관계 유형: 양의 선형 관계.
  - 관계 강도: 중간에서 강한 관계.
  - 설명: 근속연수가 길어질수록 연봉이 높아지는 경향이 있습니다. 이는 회사에서 오랫동안 근무한 직원들이 더 높은 연봉을 받을 가능성이 크다는 것을 나타냅니다.
3. 달성 매출과 연봉:
  - 관계 유형: 양의 선형 관계.
  - 관계 강도: 강한 관계.
  - 설명: 달성 매출이 높은 직원일수록 연봉이 높다는 것을 알 수 있습니다. 이는 성과가 뛰어난 직원이 높은 보상을 받는다는 것을 반영합니다.
4. 초과 근로 시간과 연봉:
  - 관계 유형: 약한 양의 관계.
  - 관계 강도: 약한 관계.
  - 설명: 초과 근로 시간이 많을수록 연봉이 약간 높아지는 경향이 있지만, 그 관계는 상대적으로 약합니다.
5. 불량 발생 횟수와 달성 매출:
  - 관계 유형: 약한 음의 관계.
  - 관계 강도: 약한 관계.
  - 설명: 불량 발생 횟수가 적을수록 달성 매출이 높은 경향이 있지만, 그 관계는 약합니다. 이는 품질 관리와 성과가 어느 정도 연관이 있음을 시사합니다.

이러한 관계들을 통해 중요한 인사이트를 얻을 수 있으며, 이는 직원들의 성과 평가나 보상 체계에 반영될 수 있는 중요한 요소들이라고 할 수 있습니다.

## 💬 Chat Prompt

제시한 컬럼들에 대해 각각 산점도 시각화 수행  
단, 산점도는 개별적으로 수행하되, 하나의 그리드에 표현

## 🤖 GPT 출력: Message

제시된 주요 컬럼들에 대해 연봉을 기준으로 산점도를 개별적으로 수행한 결과를 하나의 그리드에 표현했습니다. 각 산점도를 통해 연봉과 다른 변수들 간의 관계를 시각적으로 확인할 수 있습니다. 이를 통해 변수들 간의 상관 관계를 보다 명확하게 파악할 수 있습니다.

## 🤖 GPT 출력: Code

```
# 선택된 주요 컬럼들에 대해 개별 산점도 시각화
selected_columns = ['연봉', '성과 평가 점수', '근속연수', '달성 매출', '초과 근로 시간', '불량 발생 횟수']

# 산점도를 개별적으로 수행하되, 하나의 그리드에 표현
```

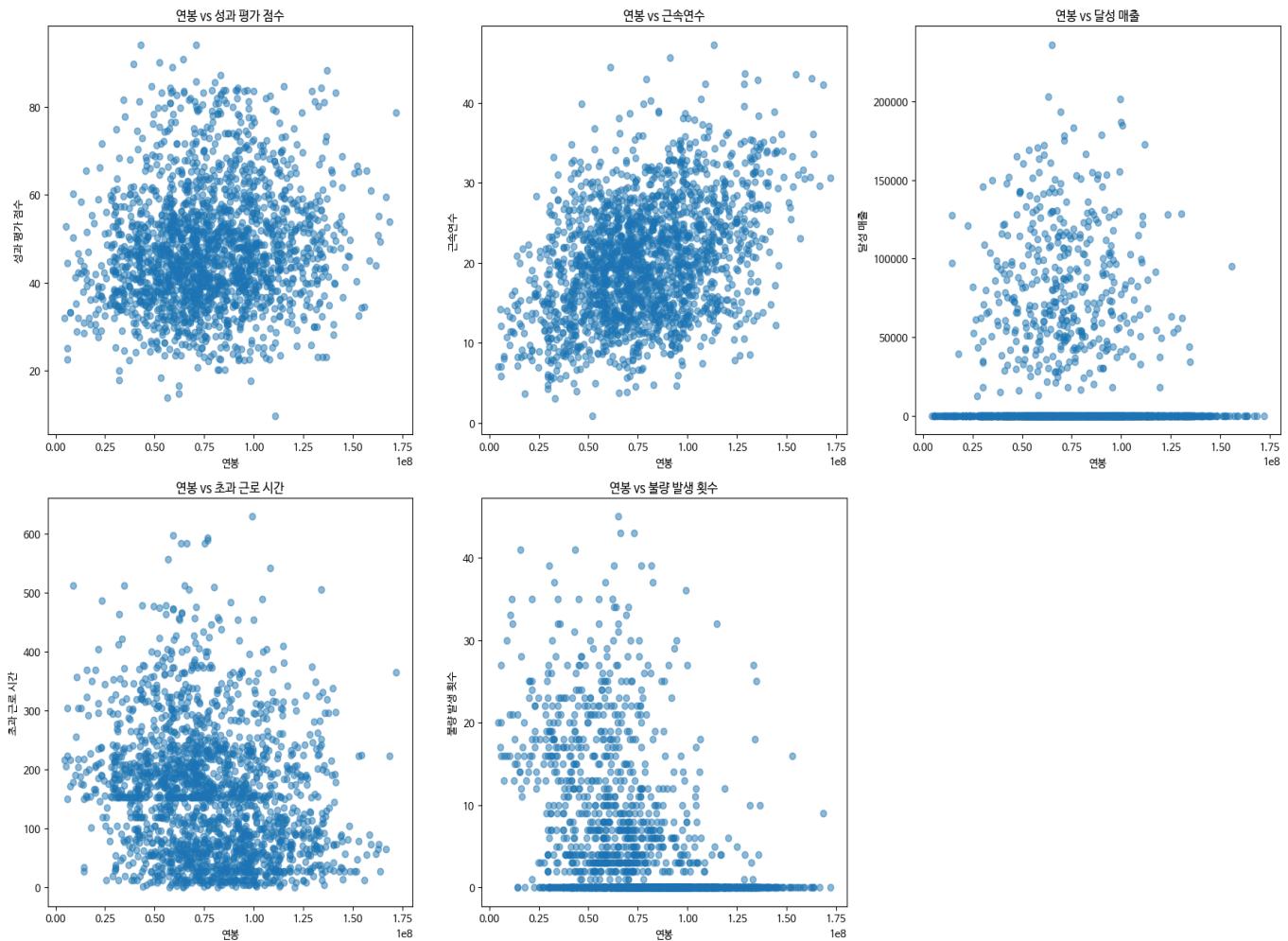
```

plt.figure(figsize=(18, 12))

for i, column in enumerate(selected_columns[1:], 1): # 연봉을 기준으로 다른 컬럼
    plt.subplot(2, 3, i)
    plt.scatter(data['연봉'], data[column], alpha=0.5)
    plt.title(f'연봉 vs {column}')
    plt.xlabel('연봉')
    plt.ylabel(column)

plt.tight_layout()
plt.show()

```



## 4. 선 그래프

### 4.1 수평선, 수직선 그래프

- 수평선과 수직선은 그래프의 한계점, 평균값 등을 표시하기 위해 사용합니다.

#### 수평선-hlines

```
plt.hlines(y, xmin, xmax, colors=None, linestyles='solid')
```

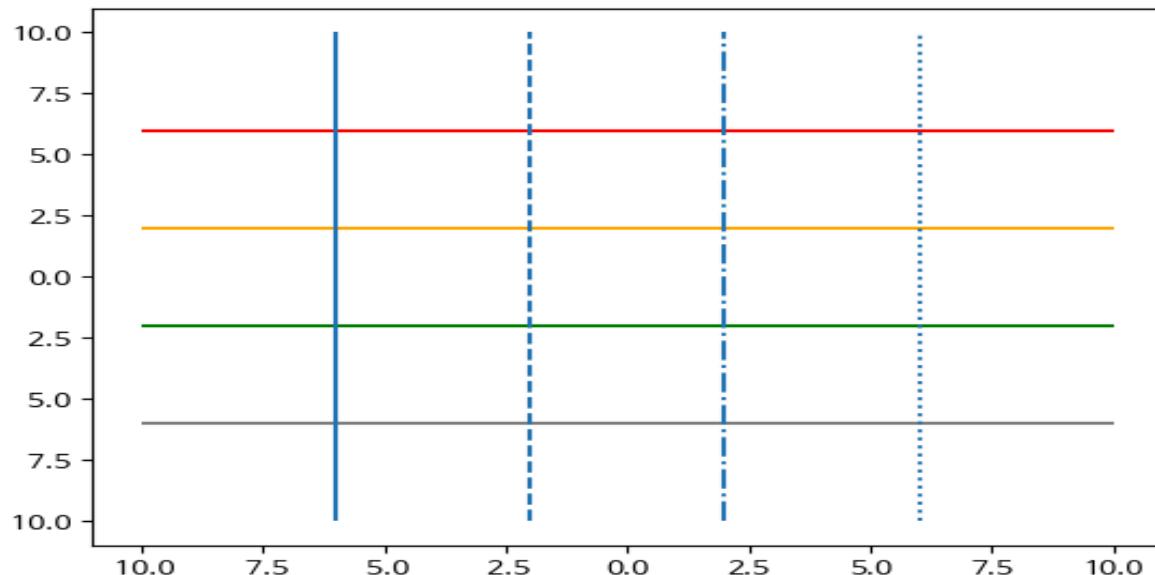
#### 수직선-vlines

```
plt.hlines(x, ymin, ymax, colors=None, linestyles='solid')
```

🤖 : 해당 그래프는 독자적인 상황이 아니라 다른 시계열에 표시선으로 들어가는 경우기 때문에 따로 실습하지 않도록 할게요

~

```
import matplotlib.pyplot as plt
plt.hlines(-6, -10, 10, color='grey')
plt.hlines(-2, -10, 10, color='green')
plt.hlines(2, -10, 10, color='orange')
plt.hlines(6, -10, 10, color='red')
plt.vlines(-6, -10, 10, linestyles='solid')
plt.vlines(-2, -10, 10, linestyles='dashed')
plt.vlines(2, -10, 10, linestyles='dashdot')
plt.vlines(6, -10, 10, linestyles='dotted')
plt.show()
```



## 4.2 꺾은선 그래프(시계열)

꺾은선 그래프는 시간의 변화에 따라 값이 지속적으로 변화할 때 유용한 그래프입니다. 값을 점으로 표기하고 점들을 선으로 이어 나타내며 X축이 시간, Y축이 값의 의미합니다.

### 💬 Chat Prompt

제시한 데이터를 시계열 데이터로 만든 뒤 꺾은선 그래프 시각화를 수행

- x축: 입사년도
- y축: 직원수 (합계)
- 라벨: 대학 구분
- 제목: 대학 구분에 따른 연도별 직원 수

### 🤖 GPT 출력: Message

입사년도에 따른 직원 수의 합계를 대학 구분별로 나누어 시계열 데이터로 만든 후, 꺾은선 그래프로 시각화했습니다. 그래프의 제목은 “대학 구분에 따른 연도별 직원 수”이며, 이를 통해 각 대학 구분에 따라 특정 입사년도에 입사한 직원 수의 변화를 확인할 수 있습니다.

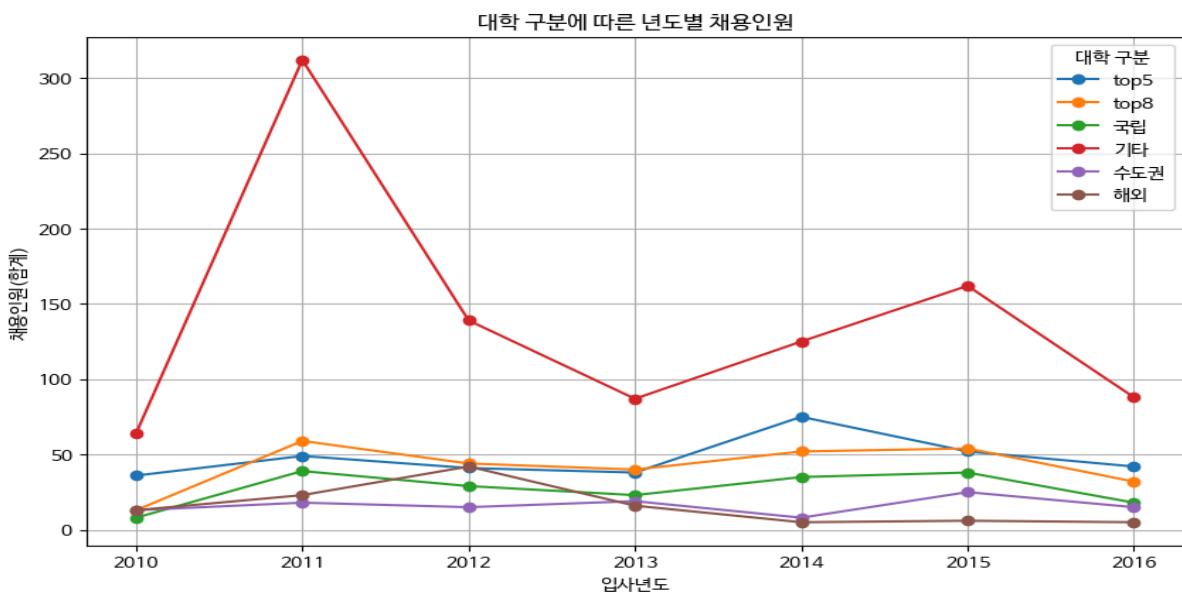
### 💡 GPT 출력: Code

```
# 입사년도별로 직원 수의 합계를 대학 구분으로 구분하여 계산
grouped_data = data.groupby(['입사년도', '대학 구분'])['ID'].count().unstack()

# 시계열 데이터를 꺾은선 그래프로 시각화
plt.figure(figsize=(10, 6))

grouped_data.plot(kind='line', marker='o', ax=plt.gca())

plt.title('대학 구분에 따른 연도별 직원 수')
plt.xlabel('입사년도')
plt.ylabel('직원 수 (합계)')
plt.legend(title='대학 구분')
plt.grid(True)
plt.show()
```



### ⚠️ 주의사항

- 데이터에서 x축이 되는 컬럼의 연속성이 없다면 꺾은선 그래프가 끊길 수 있음

### 💬 Chat Prompt

제시한 데이터를 시계열 데이터로 만든 뒤 꺾은선 그래프 시각화를 수행

- x축: 근속연수
- y축: 직업 만족도 (평균)

- 라벨: 성별
- 제목: 근속연수별 직업 만족도(평균)

### 🤖 GPT 출력: Message

근속연수에 따른 직업 만족도의 평균을 성별로 구분하여 시계열 데이터로 만들어 꺾은선 그래프로 시각화 했습니다. 그래프의 제목은 “근속연수별 직업 만족도(평균)”이며, 이를 통해 각 성별에 따라 근속연수에 따른 직업 만족도의 변화를 확인할 수 있습니다.

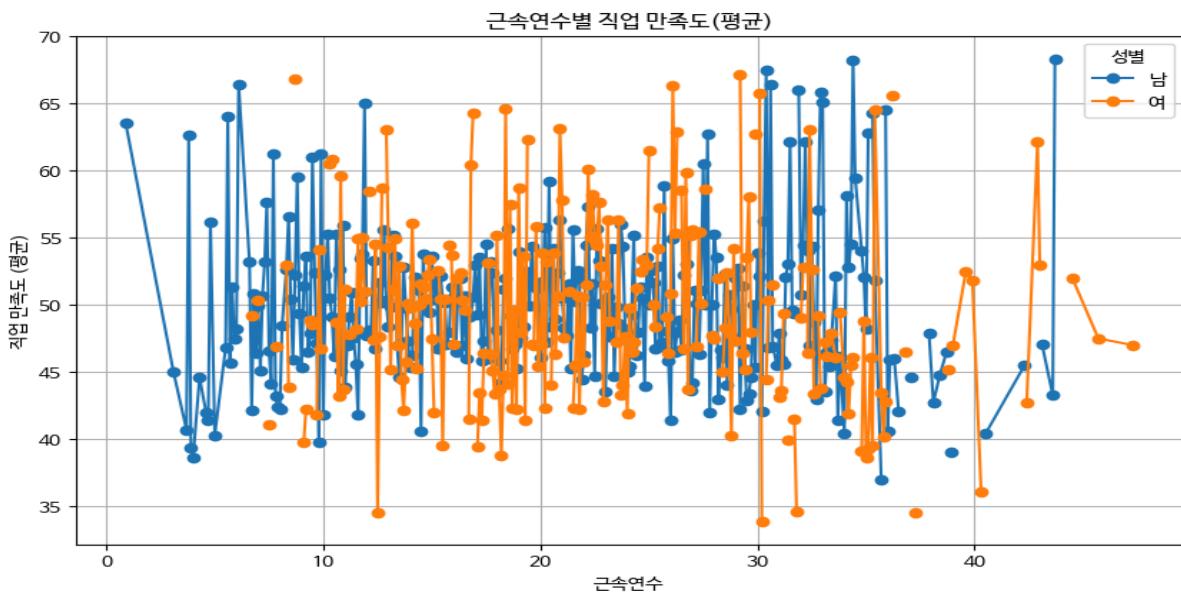
### 🤖 GPT 출력: Code

```
# 근속연수별로 직업 만족도의 평균을 성별로 구분하여 계산
grouped_data = data.groupby(['근속연수', '성별'])['직업 만족도'].mean().unstack()

# 시계열 데이터를 꺾은선 그래프로 시각화
plt.figure(figsize=(10, 6))

grouped_data.plot(kind='line', marker='o', ax=plt.gca())

plt.title('근속연수별 직업 만족도(평균)')
plt.xlabel('근속연수')
plt.ylabel('직업 만족도 (평균)')
plt.legend(title='성별')
plt.grid(True)
plt.show()
```



🤖 : 성별로 라벨 구분을 하였는데, 해당 성별에 근속연수가 연속적이지 않은 경우 꺾은선이 끊어질 수 있습니다.

## 5. 상관관계 시각화

상관관계란 2개의 변수 간의 선형관계를 표현하는 통계적 측도입니다. 3번 항목에서 산점도를 통해 변수 사이의 데이터 분포를 살펴보고 관계의 유형과 강도를 판단해 보았습니다. 이번 상관관계 시각화에서는 이를 수치화하여 데이터 간 관계를 표현합니다.

### 💡 상관관계 해석

- 상관관계는 -1~1 사이의 숫자 값으로 출력됨
- 아래 상관관계 판단 기준 표에 따라 해석

범위	설명	범위	설명
$r \leq -0.8$	강한 음의 상관관계	$0.8 \leq r$	강한 양의 상관관계
$-0.8 < r \leq -0.6$	음의 상관관계	$0.6 \leq r < 0.8$	양의 상관관계
$-0.6 < r \leq -0.4$	약한 음의 상관관계	$0.4 \leq r < 0.6$	약한 양의 상관관계
$-0.4 < r \leq 0$	거의 상관 없음	$0 \leq r < 0.4$	거의 상관 없음

### 💬 Chat Prompt

수치형 변수들에 대해 상관관계 행렬을 만들고 이를 시각화해줘

### 🤖 GPT 출력: Message

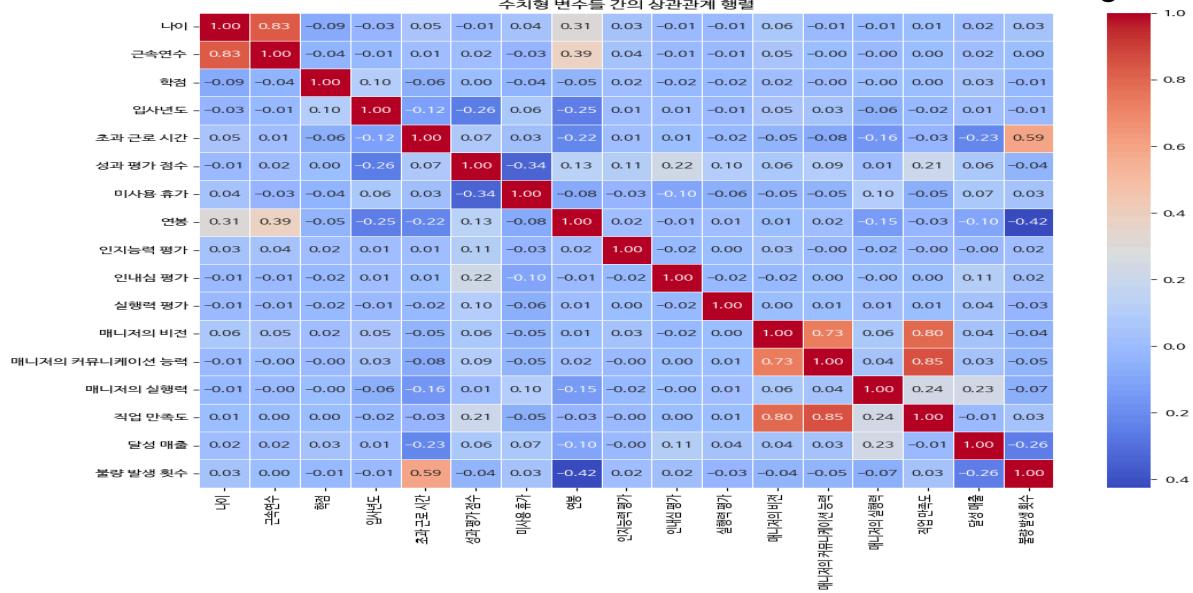
수치형 변수들 간의 상관관계 행렬을 시각화한 결과입니다. 이 히트맵(heatmap)을 통해 각 변수들 간의 상관관계를 직관적으로 파악할 수 있습니다. 상관계수는 -1에서 1 사이의 값을 가지며, 1에 가까울수록 강한 양의 상관관계, -1에 가까울수록 강한 음의 상관관계를 의미합니다. 0에 가까운 값은 상관관계가 거의 없음을 나타냅니다.

### 🤖 GPT 출력: Code

```
import seaborn as sns

# 수치형 변수들에 대해 상관관계 행렬 생성
correlation_matrix = data[numerical_columns].corr()

# 상관관계 행렬 시각화
plt.figure(figsize=(12, 10))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt='.2f',
            linewidths=0.5)
plt.title('수치형 변수들 간의 상관관계 행렬')
plt.show()
```



💡 : 목적이 되는 종속변수와 이를 설명하기 위한 독립변수간의 상관관계가 높은 것은 매우 좋은 형상입니다. 하지만 독립변수끼리 강한 상관관계가 있다면 다중공선성을 의심해야 합니다.

## 💡 다중공선성

회귀분석에서 설명변수간의 강한 선형관계가 있을 때 발생하는 문제로 설명변수끼리 상호작용하거나 중복적으로 영향을 미치는 상황을 말합니다. 다중공선성 문제가 있을 시 결과해석 및 회귀계수 추정이 불안정해지는 문제를 야기합니다.

## 💬 Chat Prompt

상관관계 행렬을 통해 강한 상관관계가 있는 변수쌍을 알려줘

## 🤖 GPT 출력: Message

상관관계 행렬을 통해 강한 상관관계가 있는 변수쌍은 다음과 같습니다:

- 나이와 근속연수: 상관계수 0.83 (양의 상관관계)
- 매니저의 비전과 매니저의 커뮤니케이션 능력: 상관계수 0.73 (양의 상관관계)
- 매니저의 비전과 직업 만족도: 상관계수 0.80 (양의 상관관계)
- 매니저의 커뮤니케이션 능력과 직업 만족도: 상관계수 0.85 (양의 상관관계)

이 변수쌍들은 0.7 이상의 상관계수를 가지며, 이는 두 변수 간에 강한 상관관계가 있음을 의미합니다.

## 🤖 GPT 출력: Code

```
# 상관관계 행렬에서 절대값이 0.7 이상인 강한 상관관계를 가진 변수쌍을 찾기
strong_correlation_pairs = correlation_matrix.unstack().reset_index()
strong_correlation_pairs.columns = ['Variable1', 'Variable2',
                                    'Correlation']
strong_correlation_pairs['abs_Correlation'] =
strong_correlation_pairs['Correlation'].abs()
strong_correlation_pairs =
```

```
strong_correlation_pairs[strong_correlation_pairs['abs_Correlation'] >=
0.7]

# 중복 제거
strong_correlation_pairs =
strong_correlation_pairs[strong_correlation_pairs['Variable1'] != 
strong_correlation_pairs['Variable2']]
strong_correlation_pairs =
strong_correlation_pairs.drop_duplicates(subset=['abs_Correlation'])

print(strong_correlation_pairs)
```

	Variable1	Variable2	Correlation	abs_Correlation
1	나이	근속연수	0.827689	0.827689
199	매니저의 비전	매니저의 커뮤니케이션 능력	0.734693	0.734693
201	매니저의 비전	직업 만족도	0.799167	0.799167
218	매니저의 커뮤니케이션 능력	직업 만족도	0.852373	0.852373

## 💡 다중공선성 해결 방법

- 상관관계 분석 후 변수 제거: 상관관계가 높은 변수쌍 중 하나를 제거하거나 의미있는 변수 하나만을 선택
- 변수 결합: 두 변수를 하나의 변수로 결합 (PCA, 차원축소)
- VIF 분석: 각 독립 변수에 대해 VIF를 계산하여 높은 변수를 제거 (일반적으로 10 이상)

## 📌 더 알아보기: 머신러닝과 다중공선성

특정 인자를 예측 또는 해석하기 위해서 통계적 방법론을 사용하거나 머신러닝을 사용할 수 있습니다. 그 중 통계적 방법론은 엄격한 제약들이 있고, 계산식에 의존하기에 다중공선성 문제가 크게 작용합니다. 반면 머신러닝 기법은 제약 보다는 성능을 중요시하며, 특히 좋은 성능을 내는 트리기반 모델들은 인자끼리의 상관관계가 높아도 패턴을 찾아낼 뿐이라 다중공선성이 크게 작용하지 않습니다. 따라서 자신이 어떤 분석을 수행할 것인지에 따라 해당 과정을 포함시키면 됩니다.

## 💕 [Bonus] 치트키

👤 : 앞서 EDA 자체는 GPT에게 매직 키워드처럼 작용한다고 하였습니다. 이번에는 이를 활용해서 우리가 수행한 과정을 한번에 수행해 보도록 하겠습니다.

### 💬 Chat Prompt

주어진 데이터셋에 대해 일변량 EDA 수행해줘

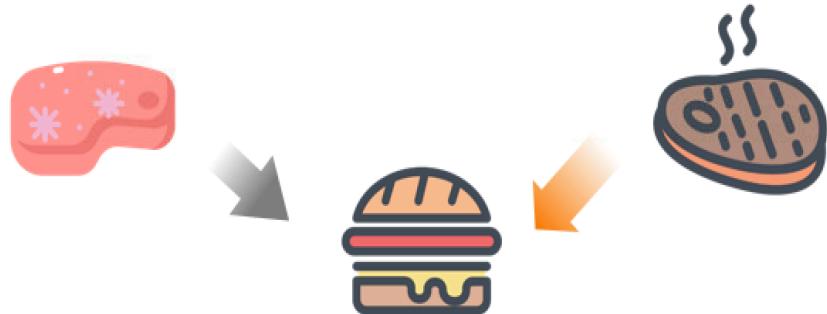
### 💬 Chat Prompt

주어진 데이터셋에 대해 다변량 EDA 수행해줘

# 🔥 Warm Up 🔥

---

## 좋은 재료를 통한 맛의 향상



## 목적을 위한 정확한 준비



## 👀 Definition

데이터 전처리란 수행하고자 하는 분석에 적합하게 데이터를 가공하는 작업을 의미

- 현업에서 발생하는 데이터를 포함하여 분석을 위해 수집한 데이터는 분석 모델에 적합하지 않은 경우가 많음
- 결측값 혹은 이상치가 있는 데이터들에 대한 처리가 가장 많은 시간이 소요됨

## 📍 데이터 전처리의 종류

구분	내용
데이터 클리닝	결측치 처리, 이상치 확인 및 제거
데이터 통합	다양한 데이터 피쳐의 결합
데이터 변환	스케일링
데이터 축소	변수 축소, 선택

# 데이터 전처리

앞서 데이터 전처리 과정 수행을 위해 시각화와 연계하여 데이터를 확인해야 한다고 하였습니다. 이번 장에서는 이전 확인하였던 olist 구매기록 데이터를 통해 데이터 전처리를 수행하도록 하겠습니다.

## 한국어 패치 다운로드

-  Korea matplotlib
-  나눔고딕 폰트

## Chat Prompt

가상환경에 한국어를 사용할 수 있도록 whl 파일과 ttf 파일을 제공하였으니, 아래 과정을 수행하며 한국어 환경 설정 수행

- Code interpreter로 제공한 파일들을 다운로드 및 수행
- 가상환경에 설치 및 한국어를 사용할 수 있도록 설정
- 설치 완료후 테스트 그래프 제시
- 모든 설명은 한국어로 진행

## GPT 출력: Code

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

# 폰트 파일 경로
font_path =
'/Users/sigi/sigirace.github.io/_posts/kang_lecture/gpt_ada/data/NanumBarunGothic.ttf'

# 폰트 등록
fm.fontManager.addfont(font_path)

# 한글 폰트 설정
plt.rc('font', family='NanumBarunGothic')
```

 : 앞서 저장하였던 데이터를 GPT에 업로드 해주세요!

## Chat Prompt

제시한 데이터는 이커머스회사 olist의 구매기록 데이터이다. 이 데이터의 컬럼을 살펴보고 설명해라.  
단, 모든 과정에서 설명시 한국어를 사용하고, ace\_tools는 사용하지 말아라

## GPT 출력: Message

제공된 이커머스 회사 Olist의 구매기록 데이터에는 다음과 같은 컬럼들이 포함되어 있습니다:

1. `order_id`: 주문을 식별하기 위한 고유 ID입니다.
2. `customer_id`: 고객을 식별하기 위한 고유 ID입니다.
3. `order_status`: 주문의 상태를 나타냅니다 (예: 완료, 취소 등).
4. `order_purchase_timestamp`: 주문이 이루어진 시간입니다.
5. `product_category`: 제품의 카테고리를 나타냅니다 (예: 전자제품, 의류 등).
6. `payment_method`: 결제 방법을 나타냅니다 (예: 신용카드, 계좌이체 등).
7. `installments`: 할부 횟수를 나타냅니다. 고객이 결제를 몇 번에 나누어 했는지를 나타냅니다.
8. `item_price_usd`: 제품의 가격을 미국 달러로 나타낸 값입니다.
9. `shipping_cost_usd`: 배송 비용을 미국 달러로 나타낸 값입니다.
10. `installment_divide`: 할부로 나눈 결제 금액을 나타냅니다.
11. `monthly_price`: 할부 결제 시 매월 지불해야 하는 금액을 나타냅니다.

이 컬럼들은 각 주문에 대한 세부 정보를 제공하며, 이를 통해 고객 행동 분석, 판매 트렌드 분석 및 기타 비즈니스 인사이트를 도출할 수 있습니다. 추가적인 분석이 필요하시면 말씀해 주세요!

#### GPT 출력: Code

```
import pandas as pd

# 데이터 불러오기
file_path = './data/olist_order_dataset_processed.csv'
data = pd.read_csv(file_path)

# 데이터의 컬럼을 살펴보기
columns_info = data.columns
columns_info
```

```
Index(['order_id', 'customer_id', 'order_status',
       'order_purchase_timestamp',
       'product_category', 'payment_method', 'installments',
       'item_price_usd',
       'shipping_cost_usd', 'installment_divide', 'monthly_price'],
      dtype='object')
```

 : 실습을 위해 앞서 데이터프레임에 함수를 적용하여 생성하였던 변수 2개를 삭제하겠습니다.

#### Chat Prompt

```
installment_divide, monthly_price 두 컬럼은 삭제해줘
```

#### GPT 출력: Message

`installment_divide`와 `monthly_price` 두 컬럼을 성공적으로 삭제하였습니다. 이제 남아 있는 컬럼들은 다음과 같습니다:

1. `order_id`
2. `customer_id`
3. `order_status`
4. `order_purchase_timestamp`
5. `product_category`
6. `payment_method`
7. `installments`
8. `item_price_usd`
9. `shipping_cost_usd`

이제 이 데이터셋을 통해 다양한 분석을 진행할 수 있습니다. 추가로 필요한 작업이 있으면 알려주세요.

#### 🤖 GPT 출력: Code

```
# 'installment_divide', 'monthly_price' 컬럼 삭제
data = data.drop(columns=['installment_divide', 'monthly_price'])

# 삭제 후 컬럼 목록 확인
data.columns
```

```
# 'installment_divide', 'monthly_price' 컬럼 삭제
data = data.drop(columns=['installment_divide', 'monthly_price'])

# 삭제 후 컬럼 목록 확인
data.columns
```

## 1. 결측치 탐색 및 처리

데이터를 적재, 수집하는 과정에서 결측치가 발생할 수 있습니다. 결측치는 데이터 분석시 오류를 발생시키는 주요 원인이기 때문에 전처리 과정을 거쳐야 합니다.

👤 : 현재 데이터 셋에서 결측치를 확인해 보도록 하겠습니다.

#### 💬 Chat Prompt

데이터 셋에서 결측치를 확인해주고 컬럼별 결측치의 비율을 알려줘

#### 🤖 GPT 출력: Message

데이터 셋에서 컬럼별 결측치 개수와 결측치 비율은 다음과 같습니다:

- item\_price\_usd: 결측치 99개, 결측치 비율 9.93%
- shipping\_cost\_usd: 결측치 99개, 결측치 비율 9.93%

나머지 컬럼들인 order\_id, customer\_id, order\_status, order\_purchase\_timestamp, product\_category, payment\_method, installments 컬럼에는 결측치가 없습니다.

즉, item\_price\_usd와 shipping\_cost\_usd 컬럼에만 결측치가 존재하며, 두 컬럼 모두 결측치 비율이 약 9.93%입니다. 추가로 조치가 필요하시다면 말씀해 주세요

### GPT 출력: Code

```
# 각 컬럼별 결측치 개수 확인
missing_data = data.isnull().sum()

# 각 컬럼별 결측치 비율 계산
missing_data_percentage = (missing_data / len(data)) * 100

# 결측치 개수와 비율을 데이터프레임으로 정리
missing_data_summary = pd.DataFrame({
    '결측치 개수': missing_data,
    '결측치 비율 (%)': missing_data_percentage
})

print(missing_data_summary)
```

	결측치 개수	결측치 비율 (%)
order_id	0	0.000000
customer_id	0	0.000000
order_status	0	0.000000
order_purchase_timestamp	0	0.000000
product_category	0	0.000000
payment_method	0	0.000000
installments	0	0.000000
item_price_usd	99	9.929789
shipping_cost_usd	99	9.929789

## 1.2 결측치 처리

결측치를 탐색하여 비율을 확인하였을 때, 일반적으로 5% 이하일 경우 제거하고 5% 이상일 경우 평균, 최빈값 등으로 대체하는 방법을 사용합니다. 이는 데이터의 수가 줄어드는 것을 방지하기 위함입니다. 본 예제에서는 학습을 위해 두 방법 모두 사용하도록 하겠습니다.

 : 데이터 셋에서 item\_price\_usd는 주요 컬럼이기에 결측치를 대체하고 shipping\_cost\_usd는 제거하도록 하겠습니다.

### Chat Prompt

shipping\_cost\_usd 컬럼의 결측치는 삭제해줘

#### 🤖 GPT 출력: Message

shipping\_cost\_usd 컬럼에서 결측치가 있는 행을 모두 삭제한 후, 데이터셋의 크기는 (898, 9)로 변경되었습니다. 이제 이 데이터셋은 shipping\_cost\_usd 컬럼에 결측치가 없는 상태입니다.

추가로 필요한 작업이 있으면 알려주세요.

#### 🤖 GPT 출력: Code

```
# shipping_cost_usd 컬럼에서 결측치가 있는 행을 삭제
data_cleaned = data.dropna(subset=['shipping_cost_usd'])

# 삭제 후 데이터프레임의 크기 확인
data_cleaned.shape
```

(898, 9)

#### 💬 Chat Prompt

item\_price\_usd의 결측치는 평균으로 대체해줘

#### 🤖 GPT 출력: Message

item\_price\_usd 컬럼의 결측치를 해당 컬럼의 평균값으로 대체했습니다. 대체 후에는 더 이상 결측치가 없습니다. 이로써 데이터셋의 결측치 처리가 완료되었습니다.

다른 작업이나 추가 요청이 있으시면 말씀해 주세요.

#### 🤖 GPT 출력: Code

```
# item_price_usd 컬럼의 결측치를 평균 값으로 대체
mean_item_price = data_cleaned['item_price_usd'].mean()
data_cleaned['item_price_usd'].fillna(mean_item_price, inplace=True)

# 결측치 대체 후 결측치 확인
data_cleaned['item_price_usd'].isnull().sum()
```

## 2. 이상치 탐색 및 처리

### 💡 이상치의 종류

(1) 관측값의 형식과 다른 형식의 값으로 표시된 결측치	(2) 관측값의 형식과 같은 형식의 값으로 표시된 결측치	(3) 자료 수집의 오류로 발생한 이상치	(4) 다른 관측치들과는 현저히 차이나는 실제 관측치
몸무게	성적	거실 온도	일자별 게임시간
60.0	90	22.4	2
55.5	85	22.3	1
ERROR	999	2345	17
70.5	100	22.1	4

- 이상치는 측정의 변동성, 실험의 오류, 측정 장비의 이상 등으로 발생할 수 있음
- 이상치는 분석 모델의 성능을 떨어뜨리거나 분석 결과에 악영향을 줄 수 있음 ↗ 제거 혹은 처리 필요

### 2.1 이상치 탐색

👤 : 먼저 데이터의 요약 통계 정보를 통해 이상치를 확인 해보겠습니다.

#### 💬 Chat Prompt

제시한 데이터의 요약 통계정보를 통해 이상치가 있는지 확인해줘

#### 🤖 GPT 출력: Message

데이터의 요약 통계 정보를 통해 몇 가지 이상치가 의심되는 값을 확인할 수 있습니다:

##### 1. `installments` (할부 횟수):

- 최대 값이 999로, 일반적인 할부 횟수를 고려할 때 비정상적으로 큰 값입니다. 일반적인 할부는 12개월, 24개월 정도이므로 999는 이상치일 가능성이 높습니다.
- 평균은 3.62로, 표준편차가 31.68로 매우 큽니다. 이는 몇몇 값이 전체 데이터를 왜곡할 가능성이 있다는 것을 시사합니다.

##### 2. `item_price_usd` (제품 가격):

- 최소 값이 \$1.805, 최대 값이 \$9999.000입니다. 일부 제품의 경우 높은 가격이 있을 수 있지만, \$9999는 상당히 높은 가격으로, 특정 데이터 입력 오류나 극단값일 가능성이 있습니다.
- 표준편차가 \$333.61로, 평균 \$29.10에 비해 상당히 큽니다. 이는 일부 극단적인 높은 가격이 전체 데이터에 영향을 미쳤을 수 있음을 나타냅니다.

##### 3. `shipping_cost_usd` (배송 비용):

- 최소 값이 -\$9.595로, 배송 비용이 음수인 경우는 비정상적입니다. 이는 데이터 입력 오류일 가능성이 있습니다.
- 최대 값은 \$99999로, 이는 비정상적으로 큰 값입니다. 일반적인 배송비로 보기 어려우므로, 이상치일 가능성이 높습니다.

- 표준편차가 \$3336.87로, 평균 \$115.49에 비해 매우 큽니다. 이는 몇몇 값이 데이터의 평균에 상당한 영향을 미치고 있음을 나타냅니다.

이러한 이상치들은 분석 전에 처리해야 할 가능성이 있습니다. 데이터의 특정 값을 추가적으로 조사하거나, 이상치를 처리하는 방법을 적용하여 분석의 정확성을 높일 수 있습니다. 추가적인 분석이나 특정 데이터 처리 방법이 필요하시면 알려주세요!

### GPT 출력: Code

```
# 데이터의 요약 통계 정보 확인
summary_stats = data.describe()

# 요약 통계 정보 출력
print(summary_stats)
```

	installments	item_price_usd	shipping_cost_usd
count	997.00000	898.00000	898.00000
mean	3.623872	29.099161	115.493469
std	31.681427	333.614822	3336.873052
min	0.000000	1.805000	-9.595000
25%	1.000000	13.110000	2.464300
50%	1.000000	17.098100	3.064700
75%	4.000000	20.898100	5.169425
max	999.00000	9999.000000	99999.000000

 : 이를 좀 더 시각화하여 판단하기 쉽도록 해보겠습니다.

### Chat Prompt

수치형 컬럼들에 대해서 개별적으로 box plot 시각화 수행

단, box plot은 개별적으로 수행하되, 하나의 그리드에 표현

### GPT 출력: Message

위의 그래프는 각 수치형 컬럼에 대한 개별적인 박스 플롯을 보여줍니다. 각 플롯은 컬럼 내 값들의 분포를 시각적으로 표현하며, 이상치를 쉽게 식별할 수 있습니다. 특히 `installments`, `shipping_cost_usd`, 그리고 `item_price_usd` 컬럼에서 이상치가 두드러지게 나타나는 것을 확인할 수 있습니다.

이러한 시각화를 통해 데이터의 특이점이나 이상치를 더 명확하게 파악할 수 있습니다. 추가적인 분석이나 질문이 있으시면 말씀해 주세요!

### GPT 출력: Code

```

import matplotlib.pyplot as plt

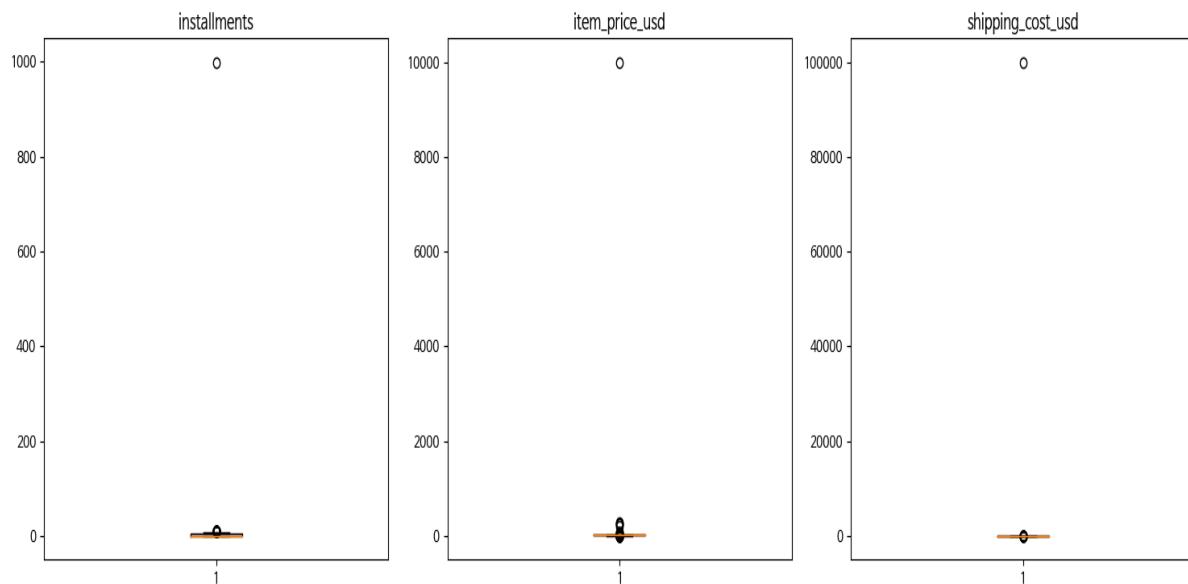
# 수치형 컬럼 선택
numeric_columns = ['installments', 'item_price_usd', 'shipping_cost_usd']

# 박스 플롯 그리기
plt.figure(figsize=(15, 10))

for i, column in enumerate(numeric_columns, 1):
    plt.subplot(3, 2, i) # 3x2 그리드로 배치
    plt.boxplot(data[column].dropna()) # NaN 값 제거 후 박스 플롯 그리기
    plt.title(column)

plt.tight_layout()
plt.show()

```



🤖 : 극단적인 값이 있어 데이터를 파악하기 어려워 보입니다. 상위, 하위 3% 정도 제거하고 데이터를 확인해보겠습니다.

#### 💬 Chat Prompt

제시한 데이터에서 각 컬럼별 상위, 하위 3% 정도 극단적인 값을 제거해줘

#### 🤖 GPT 출력: Message

각 수치형 컬럼에서 상위, 하위 3%의 극단적인 값을 제거한 후의 요약 통계 정보를 확인했습니다. 이 과정을 통해 데이터에서 일부 이상치가 제거되었으며, 통계값들이 더 안정적인 범위로 수렴한 것을 볼 수 있습니다.

예를 들어:

- **installments**: 최대 값이 10으로 줄어들어, 비정상적으로 큰 값이 제거되었습니다.
- **item\_price\_usd**: 최대 값이 \$34.01로 줄어들었으며, 평균과 표준편차가 줄어들어 안정

적인 분포를 보입니다.

- `shipping_cost_usd`: 최대 값이 \$9.69로 줄어들어, 극단적인 높은 배송 비용이 제거되었습니다.

이제 데이터를 활용한 추가 분석이 가능합니다. 다른 분석 작업이나 추가 요청 사항이 있으시면 알려주세요!

### GPT 출력: Code

```
# 각 컬럼별 상위, 하위 3% 극단적인 값을 제거

def remove_extremes(df, columns, percentile=0.03):
    for column in columns:
        lower_bound = df[column].quantile(percentile)
        upper_bound = df[column].quantile(1 - percentile)
        df = df[(df[column] >= lower_bound) & (df[column] <= upper_bound)]
    return df

# 수치형 컬럼에서 상위, 하위 3% 제거
cleaned_data = remove_extremes(data, numeric_columns)

# 결과 확인
print(cleaned_data.describe())
```

	installments	item_price_usd	shipping_cost_usd
count	731.00000	731.00000	731.00000
mean	2.704514	16.681761	3.927773
std	2.853063	6.765804	2.084227
min	0.00000	3.781000	1.485800
25%	1.00000	15.198100	2.601100
50%	1.00000	17.098100	3.100800
75%	4.00000	20.881000	5.123350
max	10.00000	34.008100	9.688100

### Chat Prompt

수치형 컬럼들에 대해서 개별적으로 box plot 시각화 수행

단, box plot은 개별적으로 수행하되, 하나의 그리드에 표현

### GPT 출력: Message

위의 박스 플롯들은 극단적인 값을 제거한 후 각 수치형 컬럼의 분포를 보여줍니다. 이상치가 제거되면서 플롯이 더 안정적인 형태를 띠고 있으며, 이전보다 데이터의 중심 경향이 더 명확하게 드러납니다.

이제 데이터 분석이 더 신뢰할 수 있는 범위 내에서 진행될 수 있으며, 추가적으로 궁금한 사항이 있거나 다른 작업이 필요하시면 말씀해 주세요!

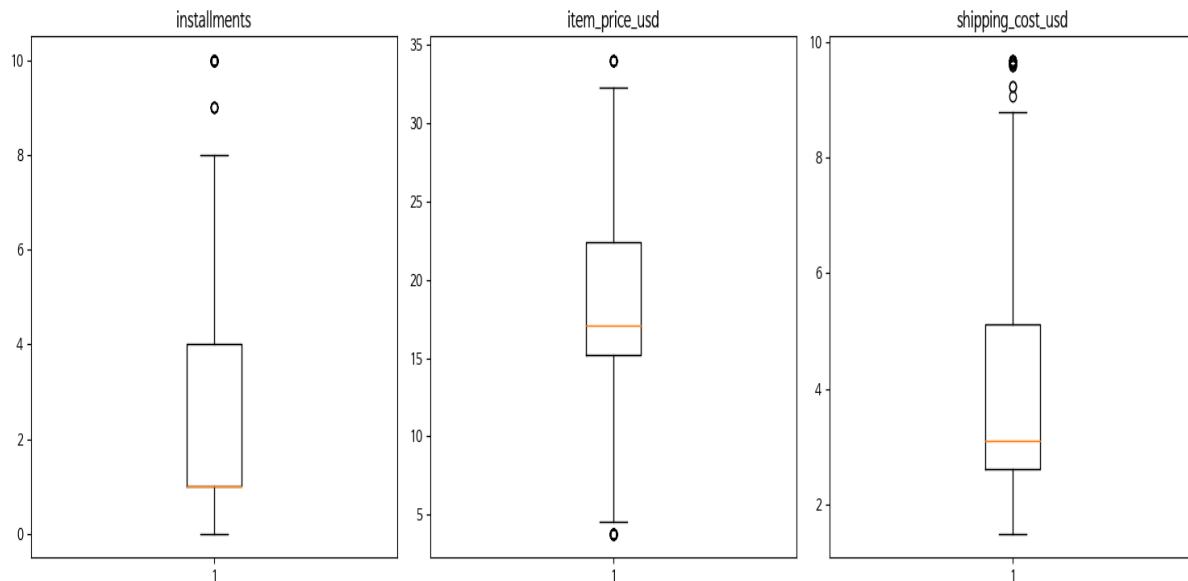
### 🤖 GPT 출력: Code

```
# 극단값 제거 후 수치형 컬럼들에 대한 박스 플롯 다시 그리기

plt.figure(figsize=(15, 10))

for i, column in enumerate(numeric_columns, 1):
    plt.subplot(3, 2, i) # 3x2 그리드로 배치
    plt.boxplot(cleaned_data[column].dropna()) # NaN 값 제거 후 박스 플롯 그리기
    plt.title(column)

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



🤖 : 다른 변수에 비해 `shipping_cost_usd`는 outlier가 많은 것으로 보입니다. 이를 IQR 방식을 통해 제거해보겠습니다.

### 💬 Chat Prompt

`shipping_cost_usd` 컬럼에 대해 IQR 방식으로 이상치를 탐색해주고, 이상치의 비율과 상한, 하한에 대해 알려줘

### 🤖 GPT 출력: Message

`shipping_cost_usd` 컬럼에 대해 IQR 방식을 사용하여 이상치를 탐색한 결과는 다음과 같습니다:

- 하한 (Lower Bound): 약 -1.17
- 상한 (Upper Bound): 약 8.89
- 이상치 비율: 전체 데이터의 약 2.09%

즉, `shipping_cost_usd` 값이 -1.17 미만이거나 8.89 초과인 경우가 이상치로 간주됩니다. 이상치는 전체 데이터의 약 2.09%를 차지하고 있습니다.

추가로 궁금한 사항이나 요청이 있으시면 알려주세요.

#### GPT 출력: Code

```
# IQR 방식으로 이상치 탐색
Q1 = data_cleaned['shipping_cost_usd'].quantile(0.25)
Q3 = data_cleaned['shipping_cost_usd'].quantile(0.75)
IQR = Q3 - Q1

lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

# 이상치 탐색
outliers = data_cleaned[(data_cleaned['shipping_cost_usd'] < lower_bound) | (data_cleaned['shipping_cost_usd'] > upper_bound)]

# 이상치 비율 계산
outliers_ratio = len(outliers) / len(data_cleaned) * 100

lower_bound, upper_bound, outliers_ratio
```

(-1.171587499999998, 8.8889125, 2.0935960591133003)

## 2.2 이상치 처리

이상치를 탐색한 후 제거하거나 평균값, 최빈값 등으로 대체하는 처리 과정이 필요합니다. 이는 데이터 분석가가 도메인 전문가와 협의를 통해 수행하여야 합니다. 본 실습에서는 다양한 처리 방법을 학습하기 위해 임의로 진행하도록 하겠습니다.

 : 데이터에 대해 하한값 미만은 제거하고 상한값 초과 값은 상한값으로 대체하겠습니다.

#### Chat Prompt

`shipping_cost_usd` 컬럼에서 하한값 미만은 제거해줘

#### GPT 출력: Message

`shipping_cost_usd` 컬럼에서 하한값(-1.17) 미만의 데이터를 제거한 후, 데이터셋의 크기는 (812, 9)로 유지되었습니다. 이는 제거된 데이터가 극히 적었음을 의미합니다.

이제 데이터셋이 이상치 제거를 통해 정제되었습니다. 추가로 필요하신 작업이 있으면 말씀해 주세요.

#### GPT 출력: Code

```
# item_price_usd 컬럼에서 하한값 이하의 데이터 제거
cleaned_data = cleaned_data[cleaned_data['item_price_usd'] > lower_bound]

# 제거 후 데이터 확인
print(cleaned_data.describe())
```

	installments	item_price_usd	shipping_cost_usd
count	656.00000	656.00000	656.00000
mean	2.862805	18.045913	4.052822
std	2.924404	5.722692	2.142393
min	0.000000	6.840000	1.485800
25%	1.000000	15.198100	2.732200
50%	2.000000	17.098100	3.114100
75%	5.000000	20.900000	5.169900
max	10.000000	34.008100	9.688100

### 📌 더 알아보기: 데이터가 변하지 않은 이유

- IQR로 구한 하한값보다 데이터의 최소값이 더 크기 때문에 제거되지 않음
- 비율로 연산하기 때문에 -값이 나올 수 있음

### 💬 Chat Prompt

shipping\_cost\_usd 컬럼에서 상한값 초과는 상한값으로 대체해줘

### 🤖 GPT 출력: Message

shipping\_cost\_usd 컬럼에서 상한값(8.89)을 초과하는 값을 상한값으로 대체했습니다. 대체 후의 shipping\_cost\_usd 컬럼의 통계 정보를 보면, 최대값이 상한값으로 조정된 것을 확인할 수 있습니다.

이로써 해당 컬럼의 이상치 처리 작업이 완료되었습니다. 추가로 궁금한 점이나 요청이 있으면 말씀해 주세요.

### 🤖 GPT 출력: Code

```
# shipping_cost_usd 컬럼에서 상한값 초과하는 데이터를 상한값으로 대체
data_cleaned['shipping_cost_usd'] =
data_cleaned['shipping_cost_usd'].apply(
    lambda x: upper_bound if x > upper_bound else x
)

# 상한값 초과 데이터 대체 후 데이터 확인
print(data_cleaned.describe())
```

	installments	item_price_usd	shipping_cost_usd
count	812.000000	812.000000	812.000000
mean	2.695813	18.040812	3.911310
std	2.849679	7.608777	2.036154
min	0.000000	3.781000	1.485800
25%	1.000000	15.198100	2.601100
50%	1.000000	17.098100	3.101750
75%	4.000000	22.420000	5.116225
max	10.000000	34.008100	8.888912

👤 : 결측치와 이상치의 처리 결과를 확인해보겠습니다.

#### 💬 Chat Prompt

결측치 처리와 이상치 처리 결과를 요약해줘

#### 🤖 GPT 출력: Message

결측치와 이상치 처리 과정을 요약하면 다음과 같습니다:

##### 1. 결측치 처리:

- `shipping\_cost\_usd` 컬럼:
  - 결측치가 있는 모든 행을 삭제하였습니다. 이로 인해 데이터셋의 크기가 감소했습니다.
- `item\_price\_usd` 컬럼:
  - 결측치를 해당 컬럼의 평균값으로 대체하였습니다.

##### 2. 이상치 처리:

- 상위 및 하위 3% 극단적인 값 제거:
  - 모든 수치형 컬럼(`installments`, `item\_price\_usd`, `shipping\_cost\_usd`)에서 상위 및 하위 3%의 극단적인 값을 제거하였습니다.
- `shipping\_cost\_usd` 컬럼의 IQR 방식 이상치 처리:
  - 하한값: -1.17 미만인 값을 제거하였습니다.
  - 상한값: 8.89를 초과하는 값을 상한값으로 대체하였습니다.

이로써 데이터셋은 결측치와 이상치가 적절히 처리된 상태입니다. 이렇게 정제된 데이터는 보다 신뢰할 수 있는 분석 결과를 도출하는 데 유용할 것입니다.

추가로 궁금한 점이나 요청이 있으시면 알려주세요.

## 3. Feature Engineering

Feature Engineering은 데이터 분석 및 머신러닝 모델링 과정에서 매우 중요한 단계로, 데이터를 바탕으로 새로운 특성(변수)을 생성하거나 기존 특성을 변형하여 분석의 성능을 향상시키는 과정을 말합니다.

#### 💡 Feature Engineering의 주요 활동

- **기존 특성 조합:** 여러 특성을 조합하여 새로운 특성을 만듭니다. 예를 들어, 가격과 수량을 곱하여 총 금액(total amount) 특성을 생성할 수 있습니다.
- **날짜/시간 특성 변환:** 날짜 및 시간을 연도, 월, 요일, 시간 등으로 변환해 모델에 유용한 정보를 추가합니다.
- **비율 계산:** 두 특성 간의 비율을 계산하여 새로운 특성을 만듭니다. 예를 들어, 매출과 비용을 이용해 이익률을 생성할 수 있습니다.

### 3.1 날짜 파생변수 생성

날짜는 Feature Engineering에서 가장 쉽게 사용할 수 있는 데이터입니다. 주말, 주중, 오전, 오후, 공휴일 등 만들어 낼 수 있는 경우의 수가 매우 많기 때문입니다. 따라서 분석하고자 하는 도메인에 맞추어 이를 활용하는 것을 추천드립니다.

#### Chat Prompt

is\_weekend 컬럼을 생성하고 이 값은 order\_purchase\_timestamp를 통해 주중, 주말로 분리하여 지정해줘

#### GPT 출력: Message

order\_purchase\_timestamp를 기준으로 주중과 주말을 분리하여 is\_weekend 컬럼을 생성했습니다. 이 컬럼은 주문이 주말에 발생했을 경우 True, 주중에 발생했을 경우 False 값을 가집니다.

예시로 첫 몇 개의 데이터에서 주중(False)과 주말(True)이 잘 구분된 것을 확인할 수 있습니다. 추가로 작업이 필요하시면 알려주세요.

#### GPT 출력: Code

```
# is_weekend 컬럼 생성 (주말: True, 주중: False)
data_cleaned['is_weekend'] =
    data_cleaned['order_purchase_timestamp'].dt.weekday >= 5

# 주중과 주말을 'Weekday'와 'Weekend'로 구분하여 저장
data_cleaned['is_weekend'] = data_cleaned['is_weekend'].apply(lambda x:
    'Weekend' if x else 'Weekday')

# 결과 확인
data_cleaned[['order_purchase_timestamp', 'is_weekend']].head()
```

	order_purchase_timestamp	is_weekend
0	2017-01-23 18:29:00	False
1	2017-01-30 10:47:00	False
2	2017-01-31 17:40:00	False
3	2017-10-01 20:04:00	True
4	2017-10-01 09:15:00	True

#### Chat Prompt

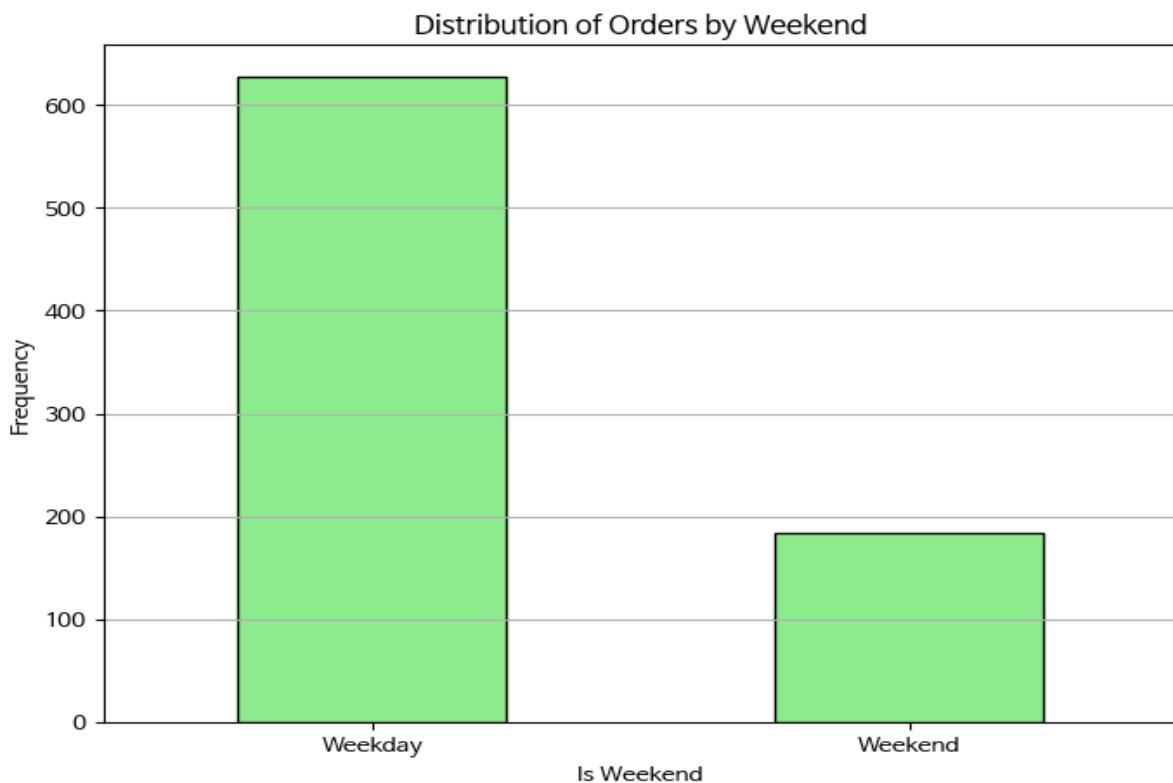
is\_weekend을 막대 그래프로 시각화해줘

### 🤖 GPT 출력: Message

is\_weekend 컬럼의 분포를 보여주는 막대그래프입니다. 주중(Weekday)과 주말(Weekend)로 나누어 각각의 주문 빈도를 확인할 수 있습니다. 이를 통해 주중과 주말의 주문 패턴을 비교할 수 있습니다. 추가로 궁금한 사항이 있으면 알려주세요.

### 🤖 GPT 출력: Code

```
# is_weekend 컬럼의 분포를 막대그래프로 시각화
plt.figure(figsize=(8, 6))
data_cleaned['is_weekend'].value_counts().plot(kind='bar',
color='lightgreen', edgecolor='black')
plt.title('Distribution of Orders by Weekend')
plt.xlabel('Is Weekend')
plt.ylabel('Frequency')
plt.xticks(ticks=[0, 1], labels=['Weekday', 'Weekend'], rotation=0)
plt.grid(axis='y')
plt.show()
```



### 3.2 추가 파생변수 생성

🤖 : 일단 추가 파생변수를 만들어 놓으면 손해는 보지 않습니다. 데이터 분석의 목적과 방법이 명확치 않으면 GPT를 이용하는 것도 아주 좋은 방법입니다.

 Chat Prompt

데이터를 보고 추가할 수 있는 파생변수를 제안해줘

 GPT 출력: Message

1. `order_hour`:

- 설명: 주문이 이루어진 시각(시간)입니다. `order_purchase_timestamp`에서 시간을 추출하여 생성할 수 있습니다.

- 활용: 특정 시간대에 주문이 집중되는지를 분석할 수 있습니다.

2. `order_day`:

- 설명: 주문이 이루어진 요일입니다. `order_purchase_timestamp`에서 요일을 추출하여 생성할 수 있습니다.

- 활용: 요일별 주문 패턴을 분석할 수 있습니다.

3. `order_month`:

- 설명: 주문이 이루어진 월입니다. `order_purchase_timestamp`에서 월을 추출하여 생성할 수 있습니다.

- 활용: 월별 매출 트렌드를 분석할 수 있습니다.

4. `total_cost_usd`:

- 설명: 제품 가격과 배송비를 합한 총 비용입니다. `item_price_usd`와 `shipping_cost_usd`를 더하여 생성할 수 있습니다.

- 활용: 고객이 실제로 지불한 총 금액을 분석할 수 있습니다.

5. `is_high_value_order`:

- 설명: 특정 기준 금액 이상인 주문을 “고가 주문”으로 간주하는 플래그 변수입니다.

- 활용: 고가 주문을 식별하고, 이러한 주문에 대한 특별한 분석이나 마케팅 전략을 수립할 수 있습니다.

6. `days_since_order`:

- 설명: 현재 시점으로부터 주문이 이루어진 날짜까지의 일수를 계산한 변수입니다.

- 활용: 고객의 최신 주문 시점을 파악하고, 재구매를 유도하는 데 사용할 수 있습니다.

7. `product_category_group`:

- 설명: 제품 카테고리를 대분류로 그룹화한 변수입니다. 예를 들어, 비슷한 제품 카테고리들을 하나의 그룹으로 묶을 수 있습니다.

- 활용: 제품의 대분류에 따른 구매 패턴을 분석할 수 있습니다.

8. `payment_installment_type`:

- 설명: 할부 횟수에 따른 결제 유형을 나누는 변수입니다. 예를 들어, `installments`가 1이면 “일시불”, 2 이상이면 “할부”로 구분할 수 있습니다.

- 활용: 할부와 일시불 고객의 행동 차이를 분석할 수 있습니다.

 : 할부 유형을 구분하여 향후 분석에 활용하는 것도 좋은 아이디어일 것 같습니다.

 Chat Prompt

`installments`의 분포를 시각화해줘

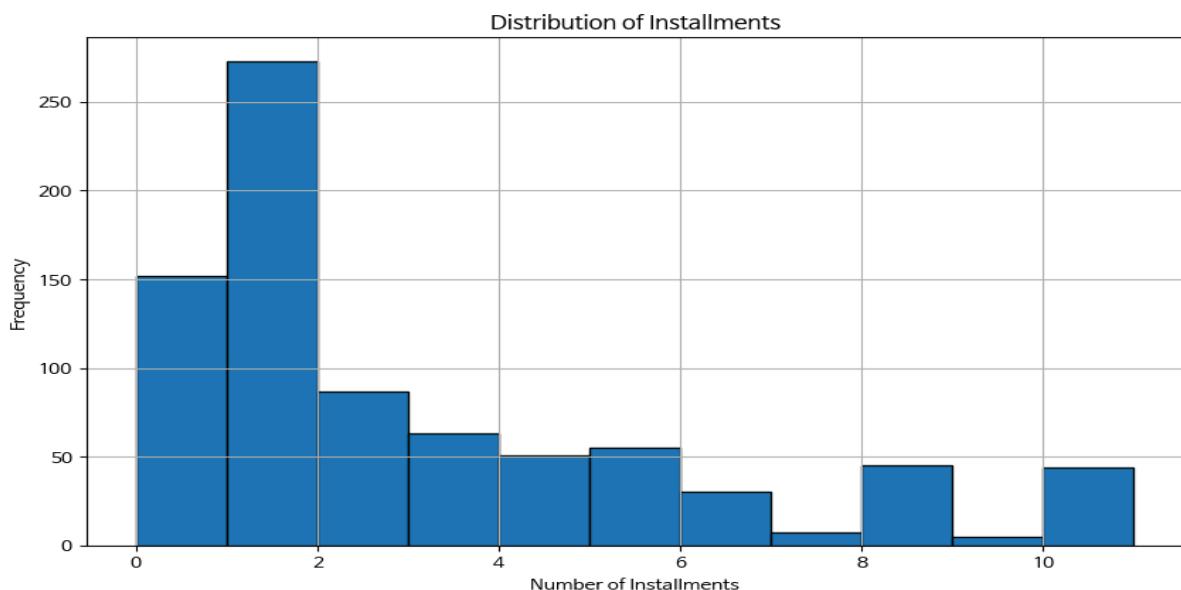
 GPT 출력: Message

그래프는 `installments` 컬럼의 분포를 나타냅니다. 각 할부 횟수에 따른 빈도를 확인할 수 있으며, 이를 통해 할부가 얼마나 자주 사용되는지, 어떤 할부 횟수가 가장 많이 사용되는지 파악할 수 있습니다. 추가로 분석하거나 시각화할 부분이 있으면 말씀해 주세요.

### 🤖 GPT 출력: Code

```
import matplotlib.pyplot as plt

# installments 컬럼의 분포 시각화
plt.figure(figsize=(10, 6))
plt.hist(data_cleaned['installments'],
         bins=range(int(data_cleaned['installments'].min()),
                    int(data_cleaned['installments'].max()) + 2), edgecolor='black')
plt.title('Distribution of Installments')
plt.xlabel('Number of Installments')
plt.ylabel('Frequency')
plt.grid(True)
plt.show()
```



👤 : 할부 개월수에 따라 유형을 3등분하여 0:일시불, 1~3:일반, 4~: 장기 로 구분하도록 하겠습니다.

### 💬 Chat Prompt

할부 개월수에 따라 유형을 구분하는 컬럼을 생성해줘  
 - 0개월: 일시불  
 - 1~3개월: 단기  
 - 4개월~: 장기

### 🤖 GPT 출력: Message

`installments` 컬럼을 기준으로 할부 개월수에 따라 “일시불”, “단기”, “장기”로 구분하는 `installment_duration_type` 컬럼을 생성했습니다. 이제 이 컬럼을 사용해 할부 유형에 따른 분석을 진행할 수 있습니다. 추가로 필요한 작업이 있으시면 말씀해 주세요.

### 🤖 GPT 출력: Code

```
# 'installment_duration_type' 컬럼 생성: 할부 개월수에 따라 유형 구분
def categorize_installments(installments):
    if installments == 0:
        return '일시불'
    elif 1 <= installments <= 3:
        return '단기'
    else:
        return '장기'

data_cleaned['installment_duration_type'] =
data_cleaned['installments'].apply(categorize_installments)

# 새로운 컬럼 확인
print(data_cleaned[['installments', 'installment_duration_type']].head())
```

installments	installment_duration_type
0	0
1	0
2	1
3	1
4	1

### 💬 Chat Prompt

`installment_duration_type`을 막대그래프로 시각화해줘

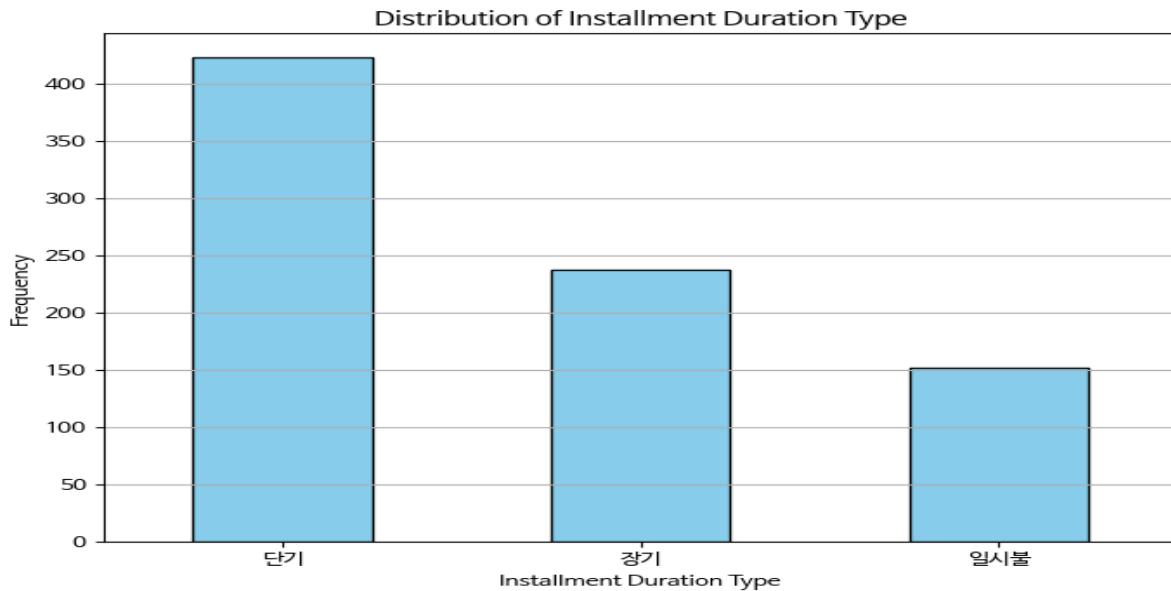
### 🤖 GPT 출력: Message

`installment_duration_type` 컬럼의 분포를 보여주는 막대그래프입니다. 할부 유형이 “일시불”, “단기”, “장기”로 구분되어 각각의 빈도를 확인할 수 있습니다. 이 시각화를 통해 어느 유형의 할부가 더 많이 사용되는지 알 수 있습니다. 추가로 분석하거나 궁금한 점이 있으시면 말씀해 주세요.

### 🤖 GPT 출력: Code

```
# installment_duration_type 컬럼의 분포를 막대그래프로 시각화
plt.figure(figsize=(8, 6))
data_cleaned['installment_duration_type'].value_counts().plot(kind='bar',
```

```
color='skyblue', edgecolor='black')
plt.title('Distribution of Installment Duration Type')
plt.xlabel('Installment Duration Type')
plt.ylabel('Frequency')
plt.xticks(rotation=0)
plt.grid(axis='y')
plt.show()
```



# 🔥 Warm Up 🔥



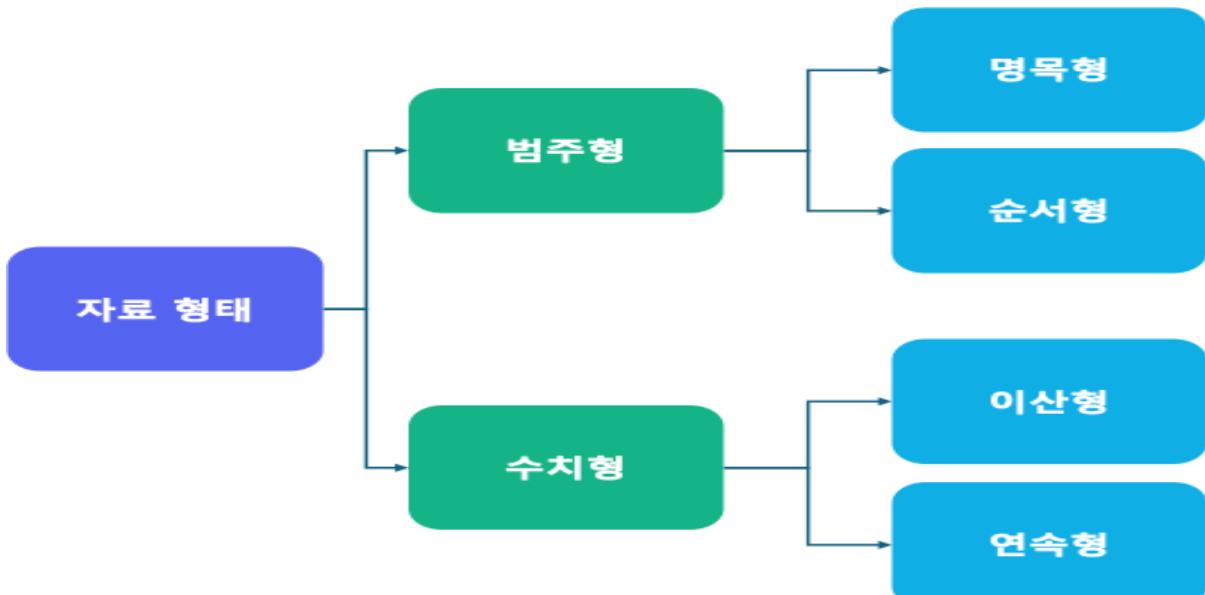
앞서 파이썬 기초, 시각화 EDA, 전처리 과정을 통해 데이터 분석을 위한 준비를 모두 끝내었습니다. 데이터 분석을 통해 인사이트를 얻을 준비가 되셨나요? 고급 데이터 분석 맛보기 시간에서 경험했던 막막했던 부분들이 이제는 어떻게 달라졌는지 같이 비교하는 시간이 되었으면 좋겠습니다.

## ✓ 들어가기전에!

솔직히... 현업으로 돌아가셔서 어떤 분석을 적용하실지 많은 고민을 해봤지만 결론은 모르겠습니다. 따라서 다양한 분석 유형을 준비하였으니 보유하신 데이터에 맞는 유형을 사용하시면 됩니다. 분석 유형을 선정하실 때 가장 중요한 것은 보유한 데이터가 어떤 자료형태인지 확인하는 것이기에 **2.5 데이터의 자료형태 확인하기**를 복습하도록 할게요!

### 2.5 데이터의 자료형태 확인하기

- 데이터의 자료형태는 **시각화**, **전처리** 그리고 **분석** 시 모두 중요하게 적용되는 개념
- 각 형태에 따라 다른 기법이 사용됨



분류	상세 분류	설명	예시
범주형		몇 개의 범주로 나누어진 자료를 의미	
명목형		순서 관계가 없는 자료	성별(남성, 여성), 혈액형(A형, B형, O형, AB형 등)
순서형		순서 관계가 존재하는 자료	교육 수준(고졸, 대졸, 석사 등), 등급(1등급, 2등급 등)
수치형		이산형과 연속형으로 이루어진 자료를 의미	
이산형		구분된 값을 가짐	출산 횟수, 방문 횟수
연속형		연속적인 값을 가짐	신장, 체중, 온도

 데이터 분석 실습

🔗 한국어 패치 다운로드

-  Korea matplotlib
-  나눔고딕 폰트

💬 Chat Prompt

가상환경에 한국어를 사용할 수 있도록 whl 파일과 ttf 파일을 제공하였으니, 아래 과정을 수행하며 한국어 환경 설정 수행

- Code interpreter로 제공한 파일들을 다운로드 및 수행
- 가상환경에 설치 및 한국어를 사용할 수 있도록 설정
- 설치 완료후 테스트 그래프 제시
- 모든 설명은 한국어로 진행

🤖 GPT 출력: Code

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

# 폰트 파일 경로
font_path =
'/Users/sigi/sigirace.github.io/_posts/kang_lecture/gpt_ada/data/NanumBarunGothic.ttf'

# 폰트 등록
fm.fontManager.addfont(font_path)

# 한글 폰트 설정
plt.rc('font', family='NanumBarunGothic')
```

 : 앞서 저장하였던 데이터를 GPT에 업로드 해주세요!

💬 Chat Prompt

제시한 데이터는 이커머스회사 olist의 구매기록 데이터를 전처리한 것이다. 이 데이터의 컬럼을 살펴보고 설명해라. 단, 모든 과정에서 설명시 한국어를 사용하고, ace\_tools는 사용하지 말아라.

🤖 GPT 출력: Message

주어진 데이터는 이커머스 회사 Olist의 구매 기록 데이터입니다. 데이터의 컬럼(열)과 그 설명은 다음과 같습니다:

1. **order\_id**: 각 주문의 고유 식별자입니다. 이 값은 주문을 식별하는 데 사용됩니다.
2. **customer\_id**: 각 고객의 고유 식별자입니다. 이를 통해 고객을 식별할 수 있습니다.
3. **order\_status**: 주문의 상태를 나타냅니다. 예를 들어, 'delivered'는 주문이 배송되었음을 의미합니다.
4. **order\_purchase\_timestamp**: 주문이 이루어진 날짜와 시간을 나타냅니다.
5. **product\_category**: 주문된 상품의 카테고리를 나타냅니다. 예를 들어, 'furniture\_decor'은 가구와 장식품 카테고리입니다.
6. **payment\_method**: 고객이 사용한 결제 방법을 나타냅니다. 예를 들어, 'boleto'는 브라질에서 널리 사용되는 결제 방식 중 하나입니다.
7. **installments**: 할부 횟수를 나타냅니다. 값이 0이면 일시불 결제, 1 이상이면 해당 횟수로 나누어 결제됩니다.
8. **item\_price\_usd**: 주문된 상품의 가격을 미국 달러(USD)로 나타냅니다.
9. **shipping\_cost\_usd**: 배송 비용을 미국 달러(USD)로 나타냅니다.
10. **is\_weekend**: 주문이 주말에 이루어졌는지를 나타냅니다. 주말이면 'True', 평일이면 'False'입니다.
11. **installment\_duration\_type**: 할부 기간의 유형을 나타냅니다. '일시불'이나 '단기' 등의 값이 들어갑니다.

이 데이터는 고객의 주문 기록, 결제 방식, 배송 관련 정보 등을 포함하고 있어, 이커머스 비즈니스 분석에 유용하게 사용될 수 있습니다. 추가적인 분석이나 문의가 필요하시면 말씀해 주세요.

### GPT 출력: Code

```
import pandas as pd

# 데이터 로드
file_path = '/mnt/data/cleaned_olist_order_data.csv'
data = pd.read_csv(file_path)

# 데이터의 첫 몇 행과 컬럼 정보 확인
data_head = data.head()
data_columns = data.columns

data_head, data_columns
```

	order_id	customer_id	\	
0	76c6e866289321a7c93b82b54852dc33	f54a9f0e6b351c431402b8461ea51999		
1	9559e0b00c67d159a266152428f85615	c3498e41d1549d56c4cbde7cbfdabeb3		
2	35aa6b03b34d8d0a5c1eaae2bb5fcf76	18f5e4ab94b6b5fdcf32dade57defe50		
3	80dabd6ee5b0e0d7d5672d511ca8cc7e	fbea4ed949b70f597425d14164f82851		
4	3389e2333066b728a245423a73f3f82c	45c155ac404d868fd8f1d07e8cd916c3		
	order_status	order_purchase_timestamp	product_category	payment_method
\				
0	delivered	2017-01-23 18:29:00	furniture_decor	boleto
1	delivered	2017-01-30 10:47:00	auto	credit_card
2	delivered	2017-01-31 17:40:00	auto	credit_card
3	delivered	2017-10-01 20:04:00	health_beauty	credit_card
4	delivered	2017-10-01 09:15:00	auto	boleto

```
installments    item_price_usd    shipping_cost_usd    is_weekend    \
0              0                3.781000            3.0495      False
1              0                28.101000           3.4200      False
2              1                9.498100            2.7588      False
3              1                17.098100           3.1141      True
4              1                30.305832            2.8690      True

    installment_duration_type
0                  일시불
1                  일시불
2                  단기
...
4                  단기 ,
Index(['order_id', 'customer_id', 'order_status',
       'order_purchase_timestamp',
       'product_category', 'payment_method', 'installments',
       'item_price_usd',
       'shipping_cost_usd', 'is_weekend', 'installment_duration_type'],
      dtype='object'))
```

## 1. EDA

👨‍💻 : 앞서 배워본 치트키를 사용하여 일변량, 다변량 EDA를 수행하겠습니다.

### 1.1 일변량 EDA

- 하나의 컬럼에 대해서만 분석함

💬 Chat Prompt

제시한 데이터로 일변량 EDA 수행해줘

🤖 GPT 출력: Message

일변량 탐색적 데이터 분석(EDA)의 결과는 다음과 같습니다:

#### 1. 기초 통계량

- 주문 상태(order\_status): 대부분의 주문이 "delivered" 상태로 완료되었습니다.
- 결제 방법(payment\_method): 대부분의 고객이 "credit\_card"를 결제 방법으로 사용했습니다.
- 할부 횟수(installments): 할부 횟수는 0(일시불)에서 10회까지 다양하며, 평균적으로 2.7회입니다.
- 상품 가격(item\_price\_usd): 상품 가격의 평균은 약 \$18.04이며, 가장 낮은 가격은 \$3.78, 가장 높은 가격은 \$34.01입니다.
- 배송비(shipping\_cost\_usd): 배송비의 평균은 약 \$3.91이며, 가장 낮은 배송비는 \$1.49, 가장 높은 배송비는 \$8.88입니다.
- 주말 구매 여부(is\_weekend): 데이터에서 대부분의 주문은 평일에 이루어진 것으로 보입니다.
- 할부 기간 유형(installment\_duration\_type): '단기' 할부가 가장 흔하게 사용되었습니다.

#### 2. 시각화 결과

- 주문 상태 분포: 거의 모든 주문이 'delivered' 상태로 완료되었습니다.

- 결제 방법 분포: 'credit\_card'가 가장 많이 사용된 결제 방법입니다.
- 할부 횟수 분포: 대부분의 고객이 1회 할부 또는 일시불을 선택했습니다.
- 상품 가격 및 배송비 분포: 가격과 배송비 모두 정규분포와 유사한 형태를 보이며, 특정 범위 내에 많이 집중되어 있습니다.
- 주말 구매 여부: 대부분의 구매가 평일에 이루어졌음을 알 수 있습니다.
- 할부 기간 유형 분포: '단기' 할부가 가장 일반적입니다.

이 분석 결과는 데이터의 전반적인 분포와 패턴을 이해하는 데 도움이 되며, 추가적인 분석이 필요할 경우 이를 기반으로 심층적인 분석을 진행할 수 있습니다. 추가적으로 궁금한 점이나 더 분석하고 싶은 부분이 있으면 말씀해 주세요.

## 1.2 다변량 EDA

- 두 개 이상의 컬럼을 종합하여 인사이트를 도출

### 💬 Chat Prompt

제시한 데이터로 다변량 EDA 수행해줘

### 🤖 GPT 출력: Message

다변량 탐색적 데이터 분석(EDA)의 결과는 다음과 같습니다:

#### 1. 주문 상태별 결제 방법 분포

- 결제 방법과 주문 상태: 대부분의 주문이 'credit\_card'로 결제되었으며, 'delivered' 상태인 경우가 대부분입니다. 다른 결제 방법들도 대부분 'delivered' 상태에 해당하지만, 'boleto' 등의 결제 방법은 상대적으로 적은 비율을 차지하고 있습니다.

#### 2. 할부 횟수와 상품 가격의 관계

- 할부와 상품 가격: 할부 횟수가 증가할수록 일부 상품의 가격이 높은 경향을 보입니다. 그러나 가격이 높다고 해서 할부 횟수가 반드시 늘어나는 것은 아니며, 다양한 결제 방식에서 이와 같은 경향을 확인할 수 있습니다.

#### 3. 상품 가격과 배송비의 관계

- 상품 가격과 배송비: 상품 가격이 증가함에 따라 배송비도 약간 증가하는 경향이 있지만, 매우 강한 상관관계는 없습니다. 주말 구매 여부에 따라 이 관계가 달라지지 않으며, 대부분의 주말 구매가 평일 구매와 비슷한 패턴을 보입니다.

#### 4. 상관 행렬

- 상관 관계: 할부 횟수, 상품 가격, 배송비 간의 상관관계를 분석한 결과, 이들 변수 간에는 강한 상관관계가 존재하지 않습니다. 이는 각 변수가 독립적으로 움직일 가능성이 크다는 것을 의미합니다.

이러한 다변량 분석을 통해 변수들 간의 관계를 이해할 수 있으며, 추가적인 분석이나 모델링의 방향을 설정하는 데 도움이 됩니다. 추가로 분석하고 싶은 부분이 있으면 알려주세요!

## 2. 분석 주제 선정

### 2.1 아이디어 도출

 : 수행한 EDA를 기반으로 과제를 도출해봅니다.

## Chat Prompt

이 데이터를 기반으로 비즈니스 인사이트를 도출하고 싶어 분석할만한 과제를 제안해줘

## GPT 출력: Message

이 데이터를 기반으로 비즈니스 인사이트를 도출할 수 있는 다양한 분석 과제를 제안드리겠습니다. 이러한 과제를 통해 이커머스 비즈니스의 효율성을 높이고, 고객 경험을 개선할 수 있는 전략적 결정을 내릴 수 있습니다.

### 1. 고객 세분화 및 맞춤형 마케팅 전략 수립

- 목표: 고객의 구매 패턴, 결제 방법, 상품 카테고리 선호도 등을 분석하여 고객을 세분화합니다. 각 고객군에 맞춤형 마케팅 캠페인을 기획하여 매출을 극대화할 수 있는 전략을 수립할 수 있습니다.
- 분석 과제:
  - 고객의 구매 빈도, 평균 구매 금액, 선호 결제 수단을 분석합니다.
  - 고객군별로 자주 구매하는 상품 카테고리와 결제 방식을 파악합니다.
  - 세그먼트별 마케팅 메시지와 프로모션 전략을 제안합니다.

### 2. 결제 방법 및 할부 옵션의 효율성 분석

- 목표: 각 결제 방법과 할부 옵션이 매출에 미치는 영향을 분석하여, 더 나은 결제 환경을 제공하고 결제 과정에서의 이탈을 줄이는 방법을 모색합니다.
- 분석 과제:
  - 결제 방법별 구매 완료율과 평균 주문 금액을 분석합니다.
  - 할부 옵션별로 고객이 어떻게 반응하는지, 그리고 이 옵션이 매출에 미치는 영향을 분석합니다.
  - 특정 결제 방법 또는 할부 옵션이 더 자주 사용되는 특정 상품 카테고리를 파악합니다.

### 3. 배송비와 고객 만족도 간의 관계 분석

- 목표: 배송비가 고객의 구매 결정과 만족도에 미치는 영향을 분석하여, 최적의 배송비 정책을 제안합니다.
- 분석 과제:
  - 상품 가격 대비 배송비의 비율이 높은 경우 고객의 구매 결정에 어떤 영향을 미치는지 분석합니다.
  - 배송비가 무료일 때와 유료일 때의 구매 패턴을 비교합니다.
  - 배송비와 고객 리뷰나 재구매율 사이의 상관관계를 분석합니다.

### 4. 주문 시간대와 구매 패턴 분석

- 목표: 고객의 주문 시간대에 따른 구매 패턴을 분석하여, 시간대별로 최적화된 프로모션 전략을 수립합니다.
- 분석 과제:
  - 주중과 주말, 그리고 특정 시간대(예: 낮 vs 저녁)에 이루어진 주문 패턴을 분석합니다.
  - 특정 시간대에 구매가 집중되는 상품 카테고리를 파악합니다.
  - 시간대별로 매출이 높은 경우 해당 시간대에 맞춤형 프로모션을 기획합니다.

### 5. 재구매율 및 고객 충성도 분석

- 목표: 고객의 재구매 패턴을 분석하여 충성도 높은 고객군을 식별하고, 이를 유지하기 위한 전략을 마련합니다.
- 분석 과제:
  - 특정 결제 방법이나 상품 카테고리에서의 재구매율을 분석합니다.
  - 첫 구매 이후 일정 기간 내에 재구매하는 고객군의 특성을 파악합니다.
  - 충성도 높은 고객에게 특별 혜택을 제공하는 프로그램을 기획합니다.

이와 같은 분석 과제를 통해 얻은 인사이트를 바탕으로, 이커머스 비즈니스의 다양한 측면을 개선하고 더 나은 성과를 달성할 수 있습니다. 각 과제에 대한 추가적인 방향이나 세부적인 분석 방법이 필요하다면 언제든지 도와드리겠습니다.

## 2.2 아디아 구체화

💡 : 프롬프트를 고도화하여 분석 과제 도출을 구체화 해보겠습니다.

아래에는 작업을 수행하기 위해 부여된 Role, 이전에 수행한 Pre-work 그리고 수행해야될 Todo가 있습니다. 요청을 적절하게 완료하는 응답을 작성하세요.

### [Role]

이커머스 회사 olist에서 근무하는 데이터 분석가

### [Pre-work]

주어진 데이터로부터 일변량, 다변량 EDA를 수행함

### [Todo]

EDA 수행결과로부터 인사이트를 도출하기 위한 과제를 선정하여 리포트를 작성한다. 단, 리포트에는 목표, 보유 데이터 중 활용할 수 있는 데이터, 명확한 분석 방법 및 알고리즘이 포함되어야 함

### [리포트 작성 예시]

#### 1. 태양광 발전량과 발전 패널의 연식간의 상관관계 분석

목표: 발전 패널의 연식이 태양광 발전량에 주는 영향을 분석하여 교체 및 유지보수 전략을 마련합니다.

활용 데이터(보유)

- `install_date`: 발전기 설치 일자
- `gen`: 발전량
- `date`: 일자

분석 방법

- 설치일자로부터 연식을 도출하여 연식과 발전량간의 단순 회귀 분석 수행
- 일자별 발전량을 활용해 시계열 데이터 분석 수행

#### 2. 리조트 객실 취소 인자 분석

목표: 리조트의 객실 취소에 영향을 주는 인자들을 분석하여 오버부킹 전략을 마련합니다.

활용 데이터(보유)

- `is_canceled`: 예약 취소 여부
- `price`: 객실 가격
- `room_type`: 객실 타입
- 등등

추가 데이터(필요)

- 주요 행사, 공공 API 호출 필요
- 등등

분석 방법

- 취소 여부에 대한 트리 모델을 학습 후 `feature importance`를 도출함
- 취소 여부인 범주형 변수와 수치형 변수 간의 t-검정 수행 후 가설에 대한 채택/기각 선택

Let's go!

## 리포트: Olist 이커머스 데이터 분석 및 인사이트 도출

### 1. 결제 방법에 따른 고객 이탈 분석

목표: 결제 방법에 따른 고객의 구매 완료율을 분석하여 이탈이 빈번한 결제 방법을 파악하고, 이를 개선하여 고객 이탈을 줄이기 위한 전략을 마련합니다.

#### 활용 데이터(보유)

- `order_status`: 주문 상태 (e.g., 'delivered', 'canceled')
- `payment_method`: 결제 방법 (e.g., 'credit\_card', 'boleto')
- `installments`: 할부 횟수

#### 분석 방법 및 알고리즘

- 결제 방법별 이탈률 분석: 결제 방법에 따라 주문이 'canceled'로 끝나는 비율을 계산하여, 어떤 결제 방법이 높은 이탈률을 보이는지 파악합니다.
- 이탈 원인 탐색: 이탈률이 높은 결제 방법에 대해 추가적인 탐색을 수행, 할부 횟수나 상품 가격이 이탈에 영향을 미치는지 확인합니다.
- 로지스틱 회귀 분석\*\*: 결제 방법, 할부 횟수, 상품 가격 등을 독립 변수로 설정하여 이탈 여부를 종속 변수로 하는 로지스틱 회귀 모델을 구축합니다. 이를 통해 결제 방법과 다른 변수들 간의 관계를 정량적으로 평가합니다.

### 2. 주말 구매 패턴 분석

목표: 주말과 평일의 구매 패턴을 비교 분석하여, 주말에 맞춤형 마케팅 캠페인을 기획하고 매출을 증대 시킬 전략을 도출합니다.

#### 활용 데이터(보유)

- `order_purchase_timestamp`: 주문 일시
- `is_weekend`: 주말 구매 여부 (Boolean)
- `product_category`: 상품 카테고리
- `item_price_usd`: 상품 가격
- `payment_method`: 결제 방법

#### 분석 방법 및 알고리즘

- 주말 vs 평일 구매 분석: 주말과 평일에 이루어진 주문 건수를 비교하고, 시간대별 주문 패턴을 시계열 분석을 통해 파악합니다.
- 상품 카테고리별 선호도 분석: 주말과 평일에 어떤 상품 카테고리가 더 많이 판매되는지 분석합니다. 이를 통해 주말 특화 상품을 제안할 수 있습니다.
- 주말 맞춤형 마케팅 캠페인 제안: 주말 구매 고객을 타겟으로 하는 할인 프로모션이나 이벤트 전략을 수립합니다.

### 3. 배송비와 고객 재구매율 간의 상관관계 분석

목표: 배송비가 고객의 재구매율에 미치는 영향을 분석하여, 최적의 배송비 정책을 마련함으로써 고객 충성도를 높이고 재구매를 촉진합니다.

#### 활용 데이터(보유)

- `item_price_usd`: 상품 가격
- `shipping_cost_usd`: 배송비
- `order_status`: 주문 상태
- `customer_id`: 고객 식별자

#### 분석 방법 및 알고리즘

- 배송비와 재구매율의 상관관계 분석: 고객별로 재구매 여부를 확인하고, 배송비 수준에 따라 재구매율이 어떻게 변하는지 분석합니다.
- 군집 분석: 배송비와 상품 가격을 기준으로 고객을 군집화하여, 각 군집의 재구매 패턴을 파악합니다.

- 회귀 분석: 배송비와 재구매율 간의 관계를 정량적으로 평가하기 위해 회귀 분석을 수행합니다. 이를 통해 배송비 정책이 재구매에 미치는 영향을 평가하고 최적의 배송비 정책을 제안합니다.

이러한 분석을 통해 도출된 인사이트는 Olist의 전략적 의사결정에 기여할 수 있으며, 고객 만족도 및 매출 증대에 중요한 역할을 할 것입니다. 필요한 추가 분석이나 데이터를 요청해 주시면 더욱 구체적인 전략을 제시할 수 있습니다.

### Learn!

상황과 역할 그리고 예시를 통해 프롬프트를 작성하면 일반적인 질문보다 구체화된 답변을 받을 수 있습니다.

### 꿀팁

GPT는 확률적으로 답변하기 때문에 여러분 질문을 수행하여 원하는 주제를 선택할 수 있습니다.

## 3. 분석 실습

 : 위에서 열심히 프롬프트 작성을 통해 과제를 도출하였으나... 실습을 위해 아래의 과제로 분석을 수행하겠습니다....

### 3.1 범주형 변수간의 관계 분석

- 범주형 변수간의 관계를 분석하기 위해서는 카이제곱 검정을 수행하여 유의확률을 통해 가설을 채택/기각 합니다.

### 분석과제

상품 유형(product\_category)과 결제 방식(payment\_method)간의 차이가 있는지 파악

### Chat Prompt

상품 유형(product\_category)과 결제 방식(payment\_method)간의 차이가 있는지 파악

- 가설 도출
- 카이제곱 분석 수행
- 유의확률에 따라 관계 도출

### GPT 출력: Message

상품 유형(product\_category)과 결제 방식(payment\_method) 간의 관계를 파악하기 위해 아래와 같은 과정을 진행하겠습니다.

#### 1. 가설 도출

먼저, 두 변수 간의 관계를 분석하기 위해 가설을 설정합니다.

- 귀무가설(H0): 상품 유형(product\_category)과 결제 방식(payment\_method) 간에는 관계가 없다. 즉, 결제 방식은 상품 유형에 따라 달라지지 않는다.
- 대립가설(H1): 상품 유형(product\_category)과 결제 방식(payment\_method) 간에는 관계가 있다. 즉, 결제 방식은 상품 유형에 따라 달라진다.

#### 2. 카이제곱 분석 수행

두 범주형 변수 간의 독립성을 검정하기 위해 카이제곱 검정을 수행합니다. 이를 통해 상품 유형과 결제 방식 간의 관계가 통계적으로 유의미한지 확인할 수 있습니다.

이제 이를 코드로 수행하겠습니다.

### 3. 카이제곱 분석 결과 해석

카이제곱 검정의 결과는 다음과 같습니다:

- 카이제곱 통계량(Chi-Square Statistic): 63.98
- p-value(유의확률): 0.061
- 자유도(Degrees of Freedom, dof): 48

### 4. 유의확률에 따른 관계 도출

- 유의확률(p-value): p-value는 0.061로, 일반적인 유의수준 0.05보다 크기 때문에 귀무가설을 기각하지 못합니다.
- 결론: 이 결과는 상품 유형(product\_category)과 결제 방식(payment\_method) 간에 통계적으로 유의미한 관계가 없다는 것을 의미합니다. 즉, 이 데이터에서는 상품 유형에 따라 결제 방식이 달라지지 않는다고 볼 수 있습니다.

따라서, 상품 유형과 결제 방식 간의 차이는 통계적으로 유의미하지 않으며, 두 변수 간에 독립적인 관계를 가질 가능성이 높습니다.

#### GPT 출력: Code

```
from scipy.stats import chi2_contingency

# 교차표 생성: product_category와 payment_method
crosstab = pd.crosstab(data['product_category'], data['payment_method'])

# 카이제곱 검정 수행
chi2, p, dof, expected = chi2_contingency(crosstab)

chi2, p, dof, expected
```

```
(63.97918687574001,
 0.06118147058250798,
 48,
 array([[1.55615764e+01, 6.17475369e+01, 9.97536946e-01, 2.69334975e+00],
        [5.95566502e+00, 2.36317734e+01, 3.81773399e-01, 1.03078818e+00],
        [4.80295567e+00, 1.90578818e+01, 3.07881773e-01, 8.31280788e-01],
        [3.84236453e+00, 1.52463054e+01, 2.46305419e-01, 6.65024631e-01],
        [1.15270936e+00, 4.57389163e+00, 7.38916256e-02, 1.99507389e-01],
        [1.92118227e+00, 7.62315271e+00, 1.23152709e-01, 3.32512315e-01],
        [5.76354680e-01, 2.28694581e+00, 3.69458128e-02, 9.97536946e-02],
        [3.55418719e+01, 1.41028325e+02, 2.27832512e+00, 6.15147783e+00],
        [4.66847291e+01, 1.85242611e+02, 2.99261084e+00, 8.08004926e+00],
        [3.84236453e-01, 1.52463054e+00, 2.46305419e-02, 6.65024631e-02],
        [9.60591133e-01, 3.81157635e+00, 6.15763547e-02, 1.66256158e-01],
```

```
[1.92118227e+01, 7.62315271e+01, 1.23152709e+00, 3.32512315e+00],  
[9.60591133e-01, 3.81157635e+00, 6.15763547e-02, 1.66256158e-01],  
[5.76354680e+00, 2.28694581e+01, 3.69458128e-01, 9.97536946e-01],  
[3.26600985e+00, 1.29593596e+01, 2.09359606e-01, 5.65270936e-01],  
[5.95566502e+00, 2.36317734e+01, 3.81773399e-01, 1.03078818e+00],  
[3.45812808e+00, 1.37216749e+01, 2.21674877e-01, 5.98522167e-  
01]]))
```

### 💬 Chat Prompt

상품 유형(product\_category)과 결제 방식(payment\_method)간의 차이를 시각화로 보여줘

### 🤖 GPT 출력: Message

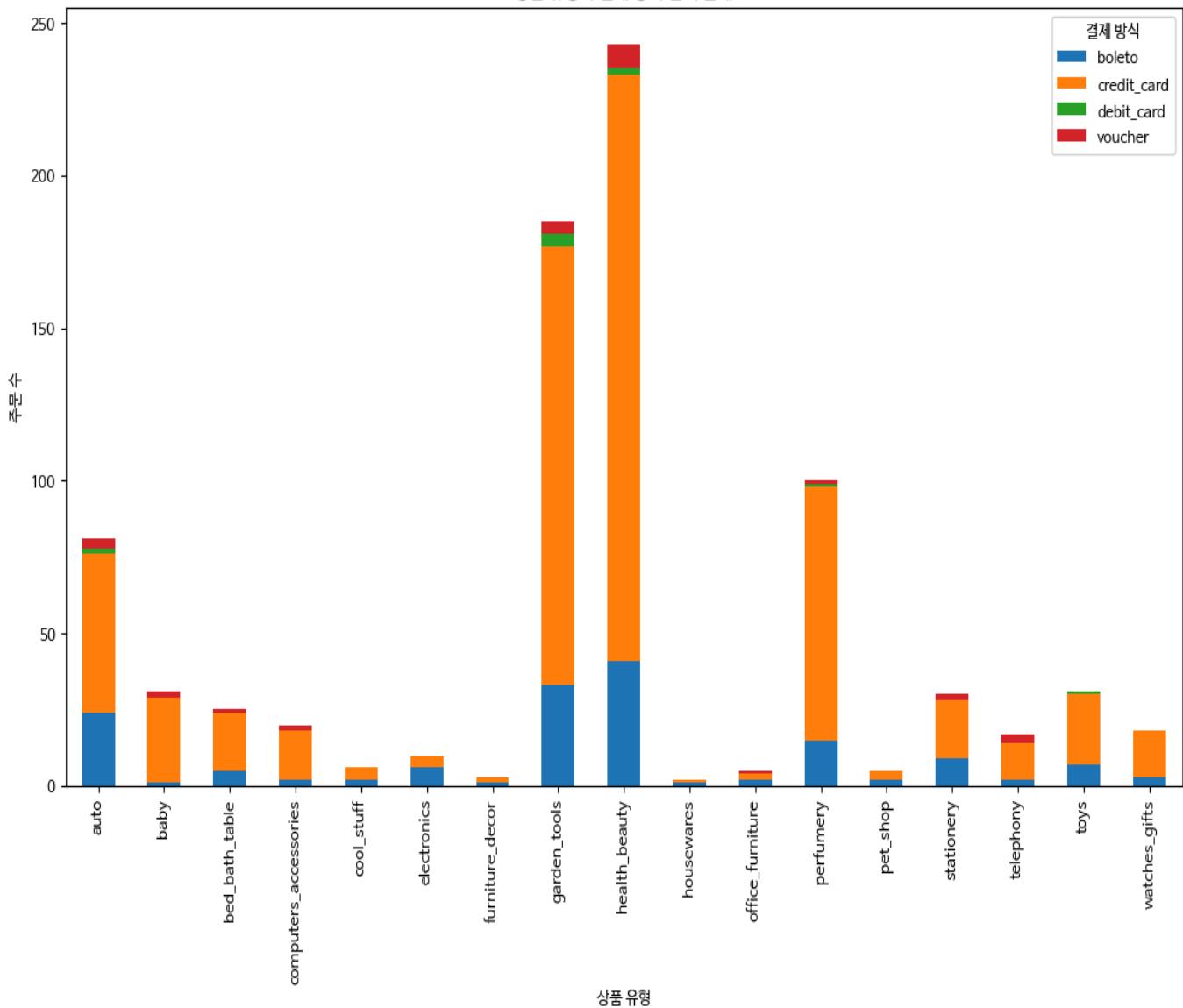
위의 시각화는 상품 유형(product\_category)과 결제 방식(payment\_method) 간의 관계를 보여주는 누적 막대 그래프입니다. 이 그래프를 통해 각 상품 유형별로 어떤 결제 방식이 주로 사용되는지 한눈에 파악할 수 있습니다.

각 막대는 특정 상품 유형에 대해 다양한 결제 방법의 비율을 나타내며, 결제 방식 간의 차이를 시각적으로 비교할 수 있습니다. 추가적인 분석이나 질문이 있으시면 말씀해 주세요.

### 🤖 GPT 출력: Code

```
import matplotlib.pyplot as plt  
  
# 교차표 생성: product_category와 payment_method  
crosstab = pd.crosstab(data['product_category'], data['payment_method'])  
  
# 시각화: Stacked Bar Chart  
crosstab.plot(kind='bar', stacked=True, figsize=(14, 8))  
plt.title('상품 유형과 결제 방식 간의 관계')  
plt.xlabel('상품 유형')  
plt.ylabel('주문 수')  
plt.legend(title='결제 방식')  
plt.show()
```

상품 유형과 결제 방식 간의 관계



## 💡 해석

분석 결과 유의미한 차이가 있지 않다고 하였으나, 유의수준을 덜 엄격하게 설정하면 차이가 있다고 볼 수 있습니다.

즉, 유의수준을 0.1로 설정할 경우, 상품 유형에 따라 결제 방식이 다를 수 있다는 결론을 내릴 수 있고, 이 관계를 바탕으로 추가적인 분석이나 비즈니스 전략을 검토해볼 수 있습니다.

## 📌 더 알아보기: 0.05 유의수준 vs 0.1 유의수준

### 1. 0.05 유의수준:

- 장점:** 더 엄격한 기준을 적용하여, 잘못된 결론을 내릴 가능성을 줄입니다. 연구 결과가 더 신뢰할 수 있게 됩니다.
- 단점:** 귀무가설을 기각하기 어려워질 수 있습니다. 즉, 실제로 관계가 있는 두 변수 사이에서도 관계를 발견하지 못할 가능성이 있습니다(2종 오류 가능성 증가).
- 과학 연구, 의학 연구 등에서 사용됩니다. 여기서는 오류의 비용이 높기 때문에 엄격한 기준이 요구됩니다.

### 2. 0.1 유의수준:

- 장점:** 더 관대한 기준을 적용하여, 실제로 관계가 있는 변수들 간의 관계를 더 잘 발견할 수 있습니다. 연구 결과가 더 포괄적일 수 있습니다.

- 단점: 귀무가설을 기각할 확률이 높아지기 때문에, 실제로 관계가 없는 두 변수 사이에 관계가 있다고 잘못 결론 내릴 가능성(1종 오류)이 증가합니다.
- 경제학, 사회학 연구 등에서 사용될 수 있습니다. 이 경우 연구의 초기 단계에서 더 많은 관계를 발견하기 위해 덜 엄격한 기준을 설정할 수 있습니다.

### 3.2 수치형 변수간의 관계 분석

- 수치형 변수간의 관계를 분석하기 위해서, 상관계수 분석 혹은 회귀 분석을 수행할 수 있습니다.

#### 💡 분석과제

상품 가격(item\_price\_usd)과 배송 비용(shipping\_cost\_usd)간의 관계 분석

#### 💬 Chat Prompt

상품 가격(item\_price\_usd)과 배송 비용(shipping\_cost\_usd)간의 관계를 분석함

- 가설 도출
- 회귀분석 수행
- 유의확률에 따라 관계 도출

#### 🤖 GPT 출력: Message

상품 가격(item\_price\_usd)과 배송 비용(shipping\_cost\_usd) 간의 관계 분석

#### 1. 가설 도출

먼저, 두 변수 간의 관계를 분석하기 위해 가설을 설정합니다.

- 귀무가설(H0): 상품 가격(item\_price\_usd)과 배송 비용(shipping\_cost\_usd) 간에는 선형적인 관계가 없다.
- 대립가설(H1): 상품 가격(item\_price\_usd)과 배송 비용(shipping\_cost\_usd) 간에는 선형적인 관계가 있다.

#### 2. 회귀분석 수행

상품 가격이 배송 비용에 미치는 영향을 파악하기 위해 단순 선형 회귀분석을 수행합니다. 독립 변수는 item\_price\_usd이고, 종속 변수는 shipping\_cost\_usd입니다.

이제 이를 코드로 수행하겠습니다.

#### 3. 회귀분석 결과 해석

- R-squared: 0.034 (3.4%)
- 이 값은 상품 가격(item\_price\_usd)이 배송 비용(shipping\_cost\_usd)의 변동성을 약 3.4% 설명한다는 의미입니다. 이는 매우 낮은 설명력으로, 상품 가격이 배송 비용에 미치는 영향이 크지 않음을 시사합니다.
- F-statistic: 28.78
- F-통계량은 모델이 유의미한지를 테스트하는 데 사용됩니다. 여기서 F-statistic의 p-value가 1.06e-07로 매우 작기 때문에, 모델이 통계적으로 유의미하다고 결론지을 수 있습니다.
- 회귀 계수(coef):

- 상수항(const): 3.0170
- 이는 기본 배송 비용이 \$3.0170임을 의미합니다.
- 상품 가격(item\_price\_usd): 0.0496
- 상품 가격이 \$1 증가할 때, 배송 비용은 약 \$0.0496 증가하는 것으로 나타났습니다.
- p-value:
- item\_price\_usd의 p-value는 0.000으로, 일반적인 유의수준 0.05보다 작습니다. 따라서 상품 가격과 배송 비용 간의 관계가 통계적으로 유의미하다는 결론을 내릴 수 있습니다.

#### 4. 결론 도출

- \*\*귀무가설(H0)\*\*는 기각됩니다. 즉, \*\*대립가설(H1)\*\*을 채택할 수 있습니다.
- 이 결과는 상품 가격(item\_price\_usd)과 배송 비용(shipping\_cost\_usd) 간에 통계적으로 유의미한 선형 관계가 있음을 나타냅니다.
- 그러나, R-squared 값이 매우 낮기 때문에, 상품 가격이 배송 비용의 변동을 설명하는 주요 요인이 아니라는 점도 중요하게 고려해야 합니다. 이는 다른 요인들이 배송 비용에 더 큰 영향을 미칠 수 있음을 시사합니다.

추가적인 분석이나 더 심도 있는 연구가 필요하다면 이를 바탕으로 진행할 수 있습니다.

#### 🤖 GPT 출력: Code

```
import statsmodels.api as sm

# 독립 변수 (상품 가격)
X = data['item_price_usd']
# 종속 변수 (배송 비용)
y = data['shipping_cost_usd']

# 상수항 추가
X = sm.add_constant(X)

# 회귀 분석 수행
model = sm.OLS(y, X).fit()

# 회귀 분석 결과 요약
model_summary = model.summary()
model_summary
```

#### OLS Regression Results

```
=====
Dep. Variable:      shipping_cost_usd    R-squared:
0.034
Model:                  OLS    Adj. R-squared:
0.033
Method:                Least Squares    F-statistic:
28.78
Date:      Wed, 21 Aug 2024    Prob (F-statistic):
1.06e-07
Time:      03:12:31    Log-Likelihood:
```

-1714.9  
 No. Observations: 812 AIC:  
 3434.  
 Df Residuals: 810 BIC:  
 3443.  
 Df Model: 1  
 Covariance Type: nonrobust  
 =====  
 =====

	coef	std err	t	P> t	[0.025 0.975]
const	3.0170	0.181	16.678	0.000	2.662 3.372
item_price_usd	0.0496	0.009	5.365	0.000	0.031 0.068

=====

Omnibus: 101.241 Durbin-Watson: 1.791  
 Prob(Omnibus): 0.000 Jarque-Bera (JB): 140.130  
 Skew: 1.017 Prob(JB): 3.73e-31  
 Kurtosis: 3.069 Cond. No. 50.5  
 =====

**Notes:**

[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

💡 : 상관계수도 고려해보겠습니다.

💬 **Chat Prompt**

상품 가격(item\_price\_usd)과 배송 비용(shipping\_cost\_usd)간의 상관관계를 상관계수로 분석 해줘

🤖 **GPT 출력: Message**

상품 가격(item\_price\_usd)과 배송 비용(shipping\_cost\_usd) 간의 상관계수는 0.185입니다.

해석:

- 상관계수 0.185는 두 변수 간에 약한 양의 선형 관계가 있음을 의미합니다. 즉, 상품 가격이 증가할 때 배송 비용도 약간 증가하는 경향이 있지만, 그 관계는 매우 약합니다.

- 이 값은 변수 간의 선형적 연관성이 크지 않음을 나타내므로, 다른 요인들이 배송 비용에 더 큰 영향을 미칠 수 있습니다.

### GPT 출력: Code

```
# 상품 가격과 배송 비용 간의 상관계수를 계산
correlation = data['item_price_usd'].corr(data['shipping_cost_usd'])
correlation
```

0.18523538792917865

### 해석

회귀분석을 통해 두 변수간 통계적으로 유의미한 선형 관계가 있음을 알 수 있었습니다. 그러나 설명력(R-squared)이 낮았고 상관계수 또한 낮았기에 아주 약한 관계임을 예상할 수 있습니다. 즉, 다른 요인이나 비선형적인 관계가 있음을 유추할 수 있습니다.

### 3.3 범주형과 수치형 변수간의 관계 분석

- 범주형과 수치형 변수간의 관계는 ANOVA 분석을 통해 알 수 있습니다.

### 분석과제

주말 여부(is\_weekend)에 따른 매출의 영향이 있는지 파악

### Chat Prompt

주말 여부(is\_weekend)에 따른 매출의 영향이 있는지 파악

- 가설 도출
- Anova 분석 수행
- 유의확률에 따라 관계 도출

### GPT 출력: Message

주말 여부(is\_weekend)에 따른 매출(item\_price\_usd)의 영향 분석

#### 1. 가설 도출

주말 여부에 따라 매출이 차이가 있는지 확인하기 위해 가설을 설정합니다.

- 귀무가설( $H_0$ ) : 주말 여부(is\_weekend)에 따른 매출(item\_price\_usd)에는 차이가 없다.
- 대립가설( $H_1$ ) : 주말 여부(is\_weekend)에 따른 매출(item\_price\_usd)에는 차이가 있다.

## 2. ANOVA 분석 수행

주말과 평일 간의 매출 차이를 분석하기 위해 일원분산분석(ANOVA)을 수행합니다. 이 방법은 두 그룹 간의 평균 차이가 통계적으로 유의미한지 판단하는 데 사용됩니다.

이제 이를 코드로 수행하겠습니다.

## 3. ANOVA 분석 결과 해석

- F-statistic: 0.207
- p-value: 0.649

## 4. 유의확률에 따른 관계 도출

- p-value는 0.649로, 일반적인 유의수준 0.05보다 훨씬 큽니다.
- 결론: 주말 여부(is\_weekend)에 따른 매출(item\_price\_usd)에는 통계적으로 유의미한 차이가 없다는 결론을 내릴 수 있습니다. 즉, 주말과 평일 간의 매출 차이는 통계적으로 의미가 없다고 판단됩니다.

이 결과에 따르면, 주말 여부가 매출에 미치는 영향은 미미하며, 두 그룹 간의 차이는 통계적으로 유의미하지 않습니다. 추가적으로 다른 요인을 고려하여 매출에 영향을 미칠 수 있는 요소를 분석하는 것이 필요할 수 있습니다.

### GPT 출력: Code

```
from scipy.stats import f_oneway

# 주말 여부에 따른 매출을 그룹화
weekend_sales = data[data['is_weekend'] == True]['item_price_usd']
weekday_sales = data[data['is_weekend'] == False]['item_price_usd']

# ANOVA 분석 수행
anova_result = f_oneway(weekend_sales, weekday_sales)
anova_result
```

```
F_onewayResult(statistic=0.20731798058076364, pvalue=0.6489997358230262)
```

### Chat Prompt

주말 여부(is\_weekend)에 따른 매출을 시각화해줘

### GPT 출력: Message

위의 그래프는 주말 여부에 따른 평균 매출을 시각화한 것입니다.

- 평일(0)과 주말(1)의 평균 매출을 비교할 수 있으며, 각 기간에 고객당 평균적으로 얼마의

매출이 발생했는지를 확인할 수 있습니다.

이 그래프를 통해 주말과 평일 중 어느 시기에 더 높은 평균 매출이 발생하는지 파악할 수 있으며, 이는 고객의 구매 패턴을 이해하는 데 도움이 됩니다.

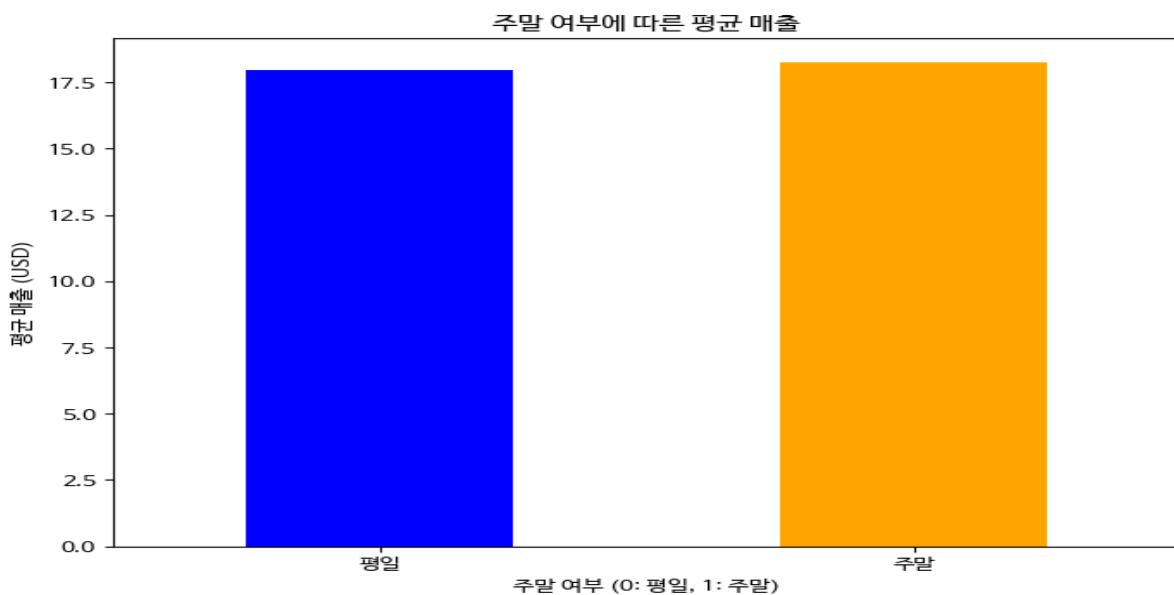
추가적으로 분석이 필요한 부분이 있으면 말씀해 주세요

### 💡 GPT 출력: Code

```
# 주말 여부에 따른 매출의 평균을 계산합니다.

# 주말 여부에 따른 매출 평균 계산
average_revenue_by_weekend = data.groupby('is_weekend')
['item_price_usd'].mean()

# 시각화
plt.figure(figsize=(8, 6))
average_revenue_by_weekend.plot(kind='bar', color=['blue', 'orange'])
plt.title('주말 여부에 따른 평균 매출')
plt.xlabel('주말 여부 (0: 평일, 1: 주말)')
plt.ylabel('평균 매출 (USD)')
plt.xticks([0, 1], labels=['평일', '주말'], rotation=0)
plt.show()
```



### 💡 해석

주중, 주말 여부에 따라 총매출은 차이가 있으나, 평균 매출은 차이가 없음을 알 수 있다. 이는 Olist가 이커머스 회사이기 때문에 주말의 영향이 적은 것이라고 해석 할 수 있다.

### 3.4 고객 세그먼트 분석

구매 데이터 기반 고객 세분화는 RFM분석을 수행합니다. 이는 고객별로 R(Recently, 얼마나 최근), F(Frequency, 얼마나 자주), M(Monetary, 얼마나 많이)에 대해 스코어링을 하고 이를 분류함으로 수행됩니다.

## 분석과제

고객 세그먼트 분석을 통해 세분화 전략 수립

### Chat Prompt

제시한 데이터를 통해 RFM 분석을 수행해라

- 각 customer\_id 별로 R,F,M을 5점 척도로 스코어링
- 스코어링 시에 설명 포함

### GPT 출력: Message

RFM 분석을 통해 각 고객별로 R(Recency), F(Frequency), M(Monetary) 스코어를 5점 척도로 계산하였습니다.

- **R\_Score:** 최근성 점수로, 최근에 구매한 고객일수록 높은 점수(5점)를 받습니다. 오래 전에 구매한 고객일수록 점수가 낮아집니다.
- **F\_Score:** 빈도 점수로, 구매 빈도가 높은 고객일수록 높은 점수(5점)를 받습니다. 구매 빈도가 낮은 고객일수록 점수가 낮아집니다.
- **M\_Score:** 금액 점수로, 총 구매 금액이 높은 고객일수록 높은 점수(5점)를 받습니다. 구매 금액이 적은 고객일수록 점수가 낮아집니다.

각 고객의 customer\_id에 대해 R, F, M 스코어를 계산하여 고객의 가치를 평가할 수 있으며, 이를 통해 마케팅 전략을 세울 수 있습니다.

추가적인 분석이나 다른 작업이 필요하시면 말씀해 주세요!

### GPT 출력: Code

```
# Recency (R): 마지막 구매 이후의 경과일
# Frequency (F): 특정 기간 동안의 구매 빈도
# Monetary Value (M): 총 구매 금액

# Recency 계산
data['order_purchase_timestamp'] =
pd.to_datetime(data['order_purchase_timestamp'])
last_purchase = data.groupby('customer_id')
['order_purchase_timestamp'].max().reset_index()
last_purchase.columns = ['customer_id', 'LastPurchaseDate']

# 분석 기준 날짜 (가장 최근 날짜로 설정)
now = data['order_purchase_timestamp'].max()

# Recency 계산
last_purchase['Recency'] = (now -
last_purchase['LastPurchaseDate']).dt.days

# Frequency 계산
```

```

frequency = data.groupby('customer_id')['order_id'].count().reset_index()
frequency.columns = ['customer_id', 'Frequency']

# Monetary 계산
monetary = data.groupby('customer_id')
['item_price_usd'].sum().reset_index()
monetary.columns = ['customer_id', 'Monetary']

# RFM 데이터를 합침
rfm = last_purchase.merge(frequency, on='customer_id')
rfm = rfm.merge(monetary, on='customer_id')

# R, F, M 각각을 5점 척도로 스코어링
rfm['R_Score'] = pd.qcut(rfm['Recency'], 5, labels=[5, 4, 3, 2, 1])
rfm['F_Score'] = pd.qcut(rfm['Frequency'].rank(method="first"), 5, labels=
[1, 2, 3, 4, 5])
rfm['M_Score'] = pd.qcut(rfm['Monetary'], 5, labels=[1, 2, 3, 4, 5])

# 결과를 확인하기 위해 RFM 데이터를 보여줍니다.
print(rfm[['customer_id', 'R_Score', 'F_Score', 'M_Score']].head())

```

	customer_id	R_Score	F_Score	M_Score
0	011a899e735ec86f00f26089cd8e09cf	3	1	2
1	0147fb5c91112d914a4f4a480a0f09e	1	1	5
2	015faf9ec36eccd957c0f2782aab09a4	2	5	5
3	0191e09197e54ad2690c3abd44723c7c	3	1	5
4	0270aff4097536e634627aba39320083	1	1	3

💡 : 스코어링한 RFM 스코어를 기준으로 고객 세분화 세그먼트를 설정합니다.

### 💬 Chat Prompt

RFM 스코어를 기준으로 고객 세분화를 해줘

- 세그먼트를 최대한 MECE하게 설계
- 각 세그먼트를 정의하고, 분류 기준을 설명한 후, 고객들을 세그먼트별로 분류
- 각 세그먼트에 특성에 어울리는 네이밍 부여
- 각 세그먼트에 몇명의 고객이 포함되었는지 표기

### 🤖 GPT 출력: Message

RFM 분석을 기반으로 고객을 다음과 같은 세그먼트로 세분화하였습니다:

1. 우수 고객: R, F, M 점수가 모두 높은 고객. 매우 가치 있는 고객들입니다. (57명)
2. 충성 고객: R(최근성)과 F(빈도) 점수가 높은 고객. 꾸준히 구매하는 충성 고객들입니다. (62명)
3. VIP 고객: R(최근성)과 M(금액) 점수가 높은 고객. 최근에 큰 금액을 소비한 고객들입니다. (81명)

4. 잠재 고객: R(최근성) 점수가 높은 고객. 최근에 구매했지만 빈도와 금액이 낮은 고객들입니다. (101명)

5. 위험 고객: R(최근성) 점수가 낮고, F(빈도)와 M(금액) 점수가 높은 고객. 과거에 가치가 높았지만 현재 이탈 위험이 있는 고객들입니다. (44명)

6. 이탈 고객: R(최근성) 점수가 낮고, F(빈도) 점수가 높은 고객. 최근 활동이 줄어든 고객들입니다. (72명)

7. 빈번 구매 고객: F(빈도) 점수가 높은 고객. 자주 구매하지만 금액이 적을 수 있는 고객들입니다. (65명)

8. 고가 구매 고객: M(금액) 점수가 높은 고객. 구매 빈도는 낮지만 큰 금액을 소비하는 고객들입니다. (76명)

9. 일반 고객: R, F, M 점수가 모두 중간 정도이거나 낮은 고객들. 특출나지 않은 일반 고객들입니다. (191명)

이 분류를 통해 각 세그먼트에 어떤 고객들이 포함되어 있는지 파악할 수 있으며, 각 세그먼트별로 맞춤형 마케팅 전략을 수립할 수 있습니다. 추가적으로 분석이 필요하거나 다른 도움이 필요하시면 말씀해 주세요!

: 프롬프트를 통해 세그먼트별 전략을 추천받아 봅니다.

### Chat Prompt

아래에는 작업을 수행하기 위해 부여된 Role, 이전에 수행한 Pre-work 그리고 수행해야될 Todo가 있습니다. 요청을 적절하게 완료하는 응답을 작성하세요.

#### [Role]

이커머스 회사 olist에서 근무하는 마케팅 팀 직원

#### [Pre-work]

고객 세그먼트를 수행함

#### [Todo]

각 고객 세그먼트별로 마케팅 전략을 수립하여 보고서 작성

Let's go!

### GPT 출력: Message

olist 마케팅 전략 보고서

작성자: 마케팅 팀

#### 1. 우수 고객 (57명)

- 특성: R, F, M 점수가 모두 높은 매우 가치 있는 고객.
- 마케팅 전략:
  - VIP 프로그램 도입: 전용 할인, 우선 배송, 맞춤형 서비스 제공.
  - 감사 이벤트: 정기적으로 감사 편지 및 특별 혜택 제공.
  - 리퍼럴 프로그램: 친구 추천 시 보상 제공.

#### 2. 충성 고객 (62명)

- 특성: 자주 구매하며, 오랜 기간에 걸쳐 꾸준히 활동한 고객.
- 마케팅 전략:
- 포인트 제도 강화: 구매 횟수에 따라 포인트 적립률 증가.
- 제품 리뷰 요청: 리뷰 작성 시 보상 제공.
- 재구매 유도 캠페인: 재구매 시 특별 할인 제공.

### 3. VIP 고객 (81명)

- 특성: 최근에 높은 금액을 소비한 고객.
- 마케팅 전략:
- 고가 상품 추천: 프리미엄 상품 및 신규 럭셔리 제품 추천.
- 맞춤형 서비스: 개인 쇼핑 도우미 서비스 제공.
- 독점 이벤트: 한정판 제품 또는 이벤트 초대.

### 4. 잠재 고객 (101명)

- 특성: 최근 구매는 했지만 빈도와 금액이 낮은 고객.
- 마케팅 전략:
- 제품 관심사 분석: 관심 제품에 대한 타겟 광고 제공.
- 웰컴 오퍼: 첫 구매 이후 재구매 시 추가 할인 제공.
- 뉴스레터 강화: 새로운 제품 및 할인 소식을 정기적으로 제공.

### 5. 위험 고객 (44명)

- 특성: 과거에 가치가 높았지만 현재 이탈 위험이 있는 고객.
- 마케팅 전략:
- 리인게이지먼트 캠페인: 개인화된 메일링을 통해 재참여 유도.
- 재구매 유도 할인: 장바구니에 남아있는 제품에 대한 할인 제공.
- 휴면 고객 이벤트: 특정 기간 내 구매 시 특별 혜택 제공.

### 6. 이탈 고객 (72명)

- 특성: 과거에는 자주 구매했지만 최근 활동이 감소한 고객.
- 마케팅 전략:
- 재참여 캠페인: 개인화된 혜택으로 고객의 관심을 다시 끌기.
- 긴급 할인 제공: 한정 기간 동안 특별 할인을 제공해 재방문 유도.
- 서비스 피드백 요청: 이탈 원인을 파악하기 위한 설문조사 시행.

### 7. 빈번 구매 고객 (65명)

- 특성: 구매 빈도가 높으나, 구매 금액이 적은 고객.
- 마케팅 전략:
- 구독 서비스 제안: 정기적인 구매를 위한 구독 서비스 도입.
- 번들 할인: 다수의 저가 상품을 한 번에 구매 시 할인 제공.
- 친구 추천 보상: 친구를 추천할 때마다 할인 혜택 제공.

### 8. 고가 구매 고객 (76명)

- 특성: 구매 빈도는 낮지만, 구매할 때마다 큰 금액을 소비하는 고객.
- 마케팅 전략:
- 맞춤형 제안: 프리미엄 상품 추천 및 맞춤형 서비스 제공.
- 한정판 제품 소개: 특별한 제품



고생하셨습니다.

sigirace.github.io

