Data Analysis Case Study (1)
- Optimization Problems

ESC 2024 Summer Special Session 1차
발제자: 유채원

Contents

1

Introduction to Mathematical Optimization

Why do we need optimization in data analysis?

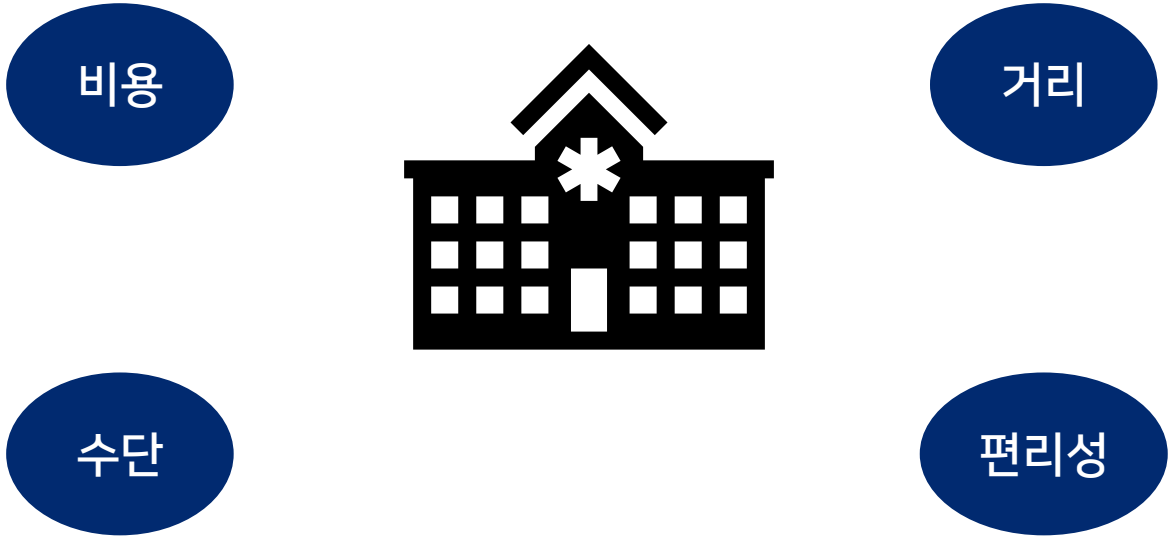
Mathematical Optimization

데이터 분석에 최적화가 왜 배워야 하는가?

버스 운영 확대를 위한 최적 노선을 제안해본다면…

노인들을 위한 시설의 최적 입지를 선정하고 싶다면…

여러 제한된 조건에서 물류를 서울에서 부산까지 보내고 싶다면…



현실에서는 고려할 게 너무 많아~~!!ㅠㅠ

Mathematical Optimization

수학적 최적화란 무엇인가?

- 주어진 제약 조건을 만족하는 선택 가능한 대안 중에서 의사결정자의 목표를 최대한으로 달성하는 최적의 대안을 선택하는 일련의 과정을 모사하는 수리 모형

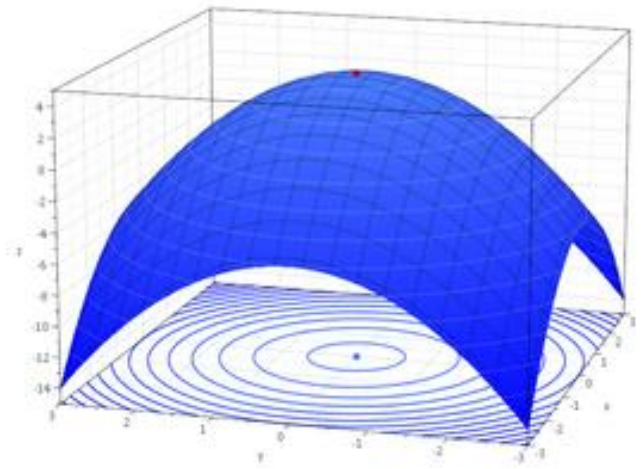
인공지능, 또는 데이터와 관련될 때 이 목표는 보통 추정값과 실제값의 error를 줄이는 것

3가지 조건

- 결정변수(decision variable)
- 목적함수(objective function)
- 제약조건(constraint)

Minimize $f_0(x)$

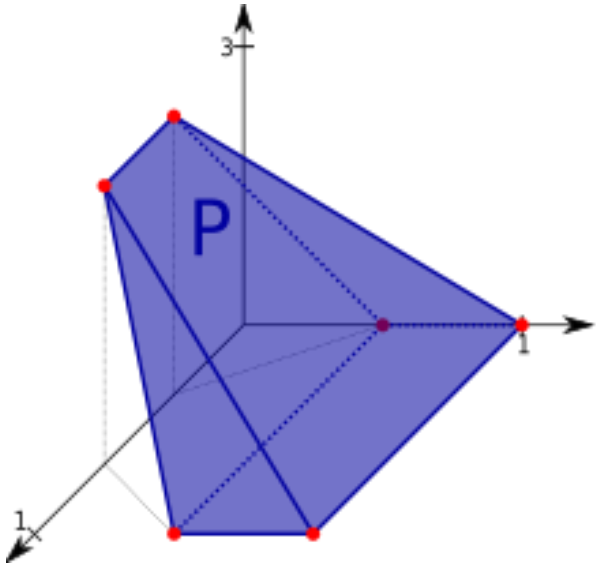
Subject to $f_i(x) \leq b_i,\ i = 1,2,\cdots,m \qquad \boldsymbol{x} \in \mathbb{R}^n$



Mathematical Optimization

최적화 모형의 예시

1. 선형 계획법
2. 정수 계획법
3. 비선형 계획법



선형 계획법(Linear Programming)

- 목적함수와 제약조건을 구성하는 함수와 결정변수 사이의 관계가 모두 선형 관계식
- 결정변수는 연속형(continuous) 변수

표준형

$$\begin{aligned} \text{Minimize} \quad & c_1x_1 + c_2x_2 + \cdots + c_nx_n \\ \text{Subject to} \quad & a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1 \\ & a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1 \\ & x_1, x_2, \cdots, x_n \geq 0 \quad \text{Non-negativity} \end{aligned}$$

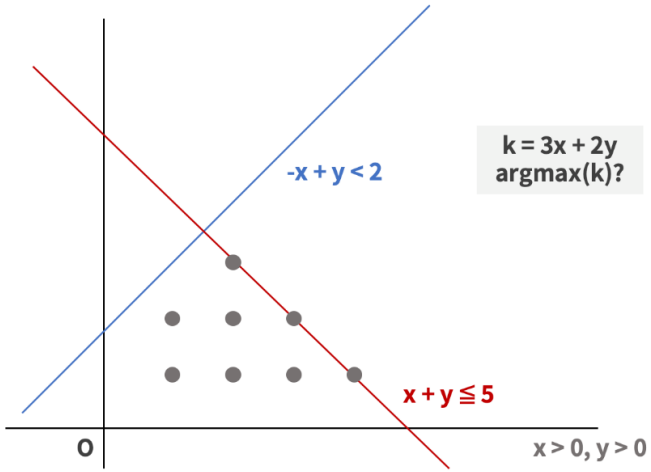
모든 제약조건은 등호(=) 제약 조건

해법 1. Simplex Method 2. Interior Point Method 3. Solver

Mathematical Optimization

최적화 모형의 예시

1. 선형 계획법
2. 정수 계획법
3. 비선형 계획법



정수 계획법(Integer Programming)

- 선형 계획법의 표준형에 정수 제약조건 추가

표준형

$$\text{Minimize} \quad c_1x_1 + c_2x_2 + \cdots + c_nx_n$$

$$\text{Subject to} \quad a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1$$

$$a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1$$

$$\sout{x_1, x_2, \cdots, x_n \geq 0}$$

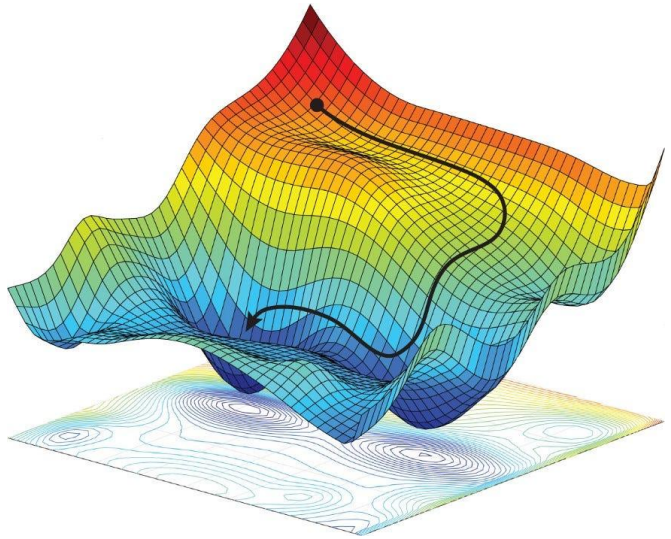
$$x_1, x_2, \cdots, x_n \in \{0,1\}$$

해법 1. Enumeration 2. Branch-and-Bound 3. Solver

Mathematical Optimization

최적화 모형의 예시

1. 선형 계획법
2. 정수 계획법
3. 비선형 계획법



비선형 계획법(Nonlinear Programming)

- 결정변수가 가질 수 있는 값을 인수분해로 표현

해법

1. Gradient Descent : 기울기를 활용하여 해의 개선 방향 결정

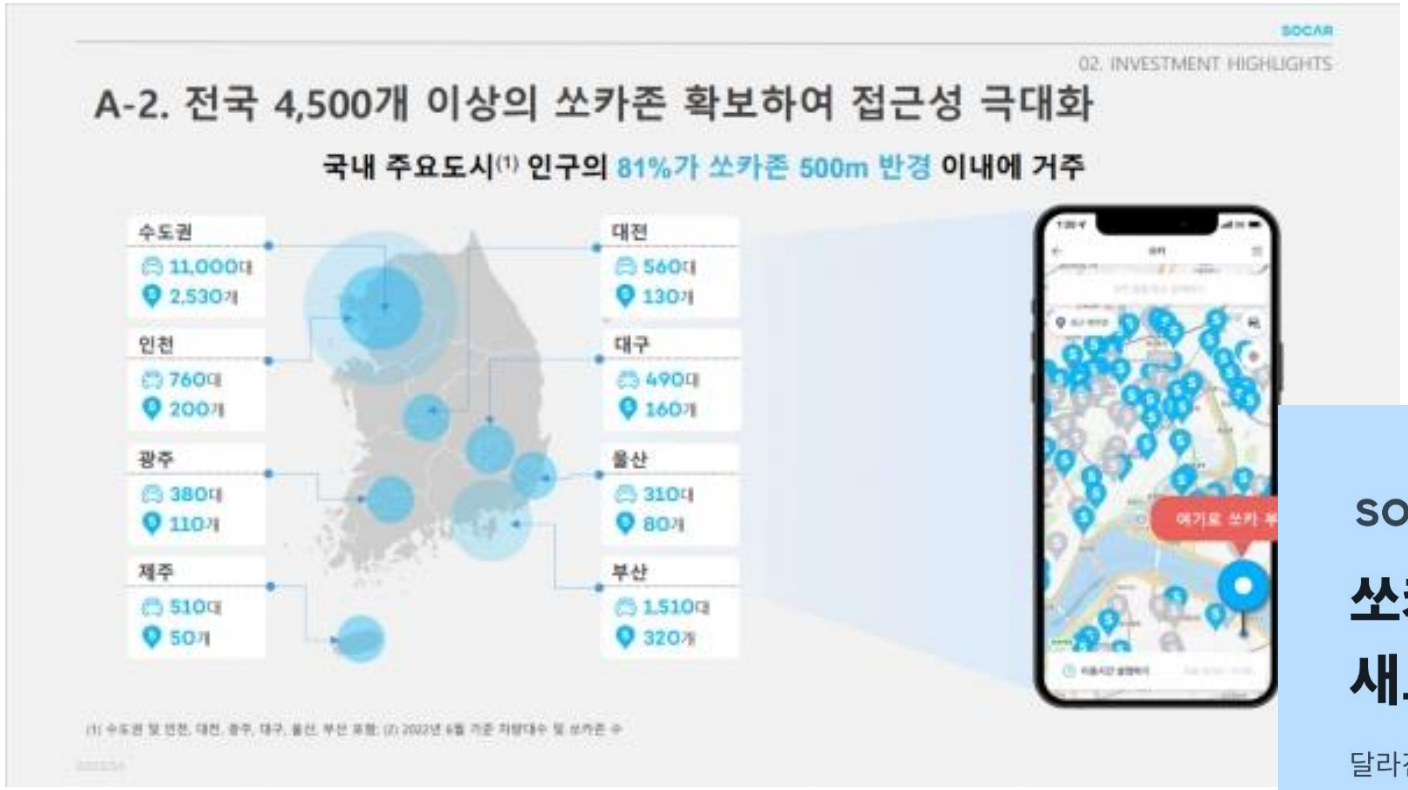
$$x_{k+1} = x_k - \alpha \nabla f_0(x_k)$$

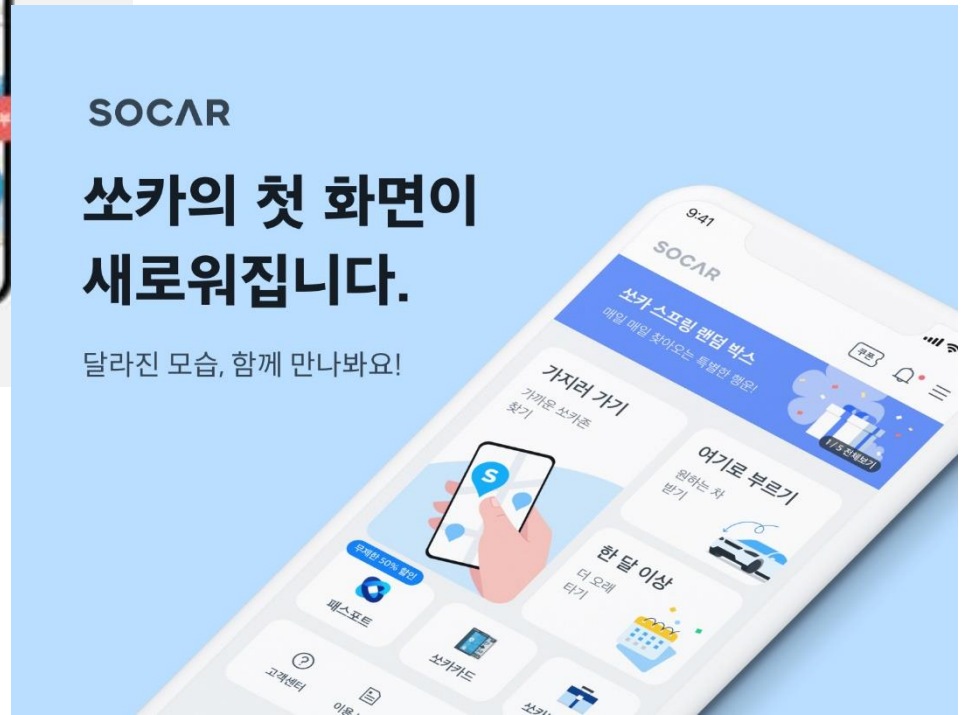
2. Newton-Raphson Method : 가장 직관적인 이동량을 채택하여 해를 개선

$$x_{k+1} = x_k - \frac{f_0(x_k)}{f'(x_k)}$$

2
Application to the case of *Socar*

Case Study - 쏘카



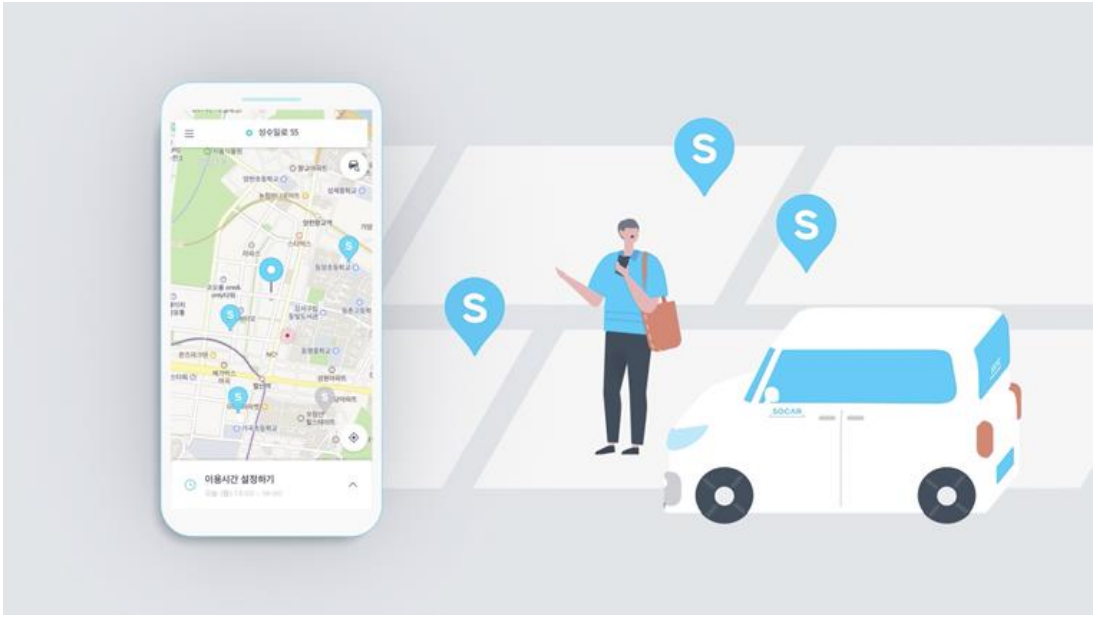


예약 테트리스 프로젝트(What)

쏘카에서 차량을 예약하기 위해서는 가까운 쏘카존에서 차량을 배정받아야 합니다.

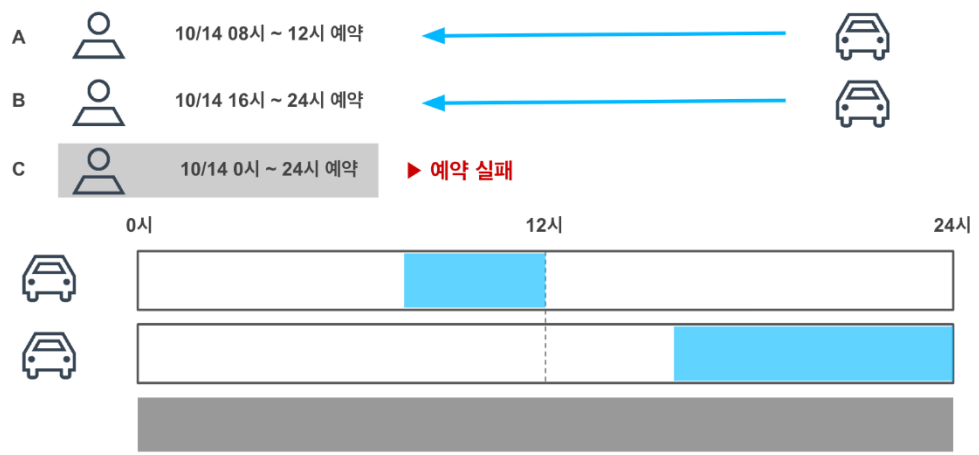
간단하게 하나의 존에 한 종류의 차량이 여러 개 있다면, 어떤 차량이 예약한 사용자에게 먼저 배정되어야 할까요?

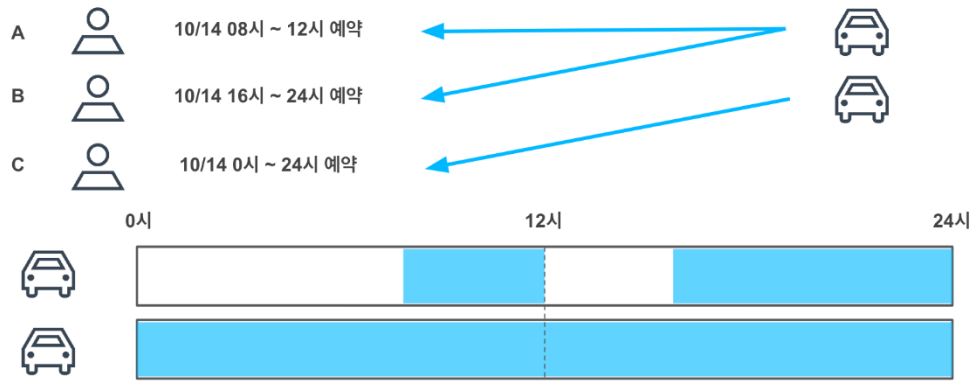
Socar의 예약 테트리스 프로젝트



예약 테트리스 프로젝트 (What)

예약 테트리스 프로젝트란?



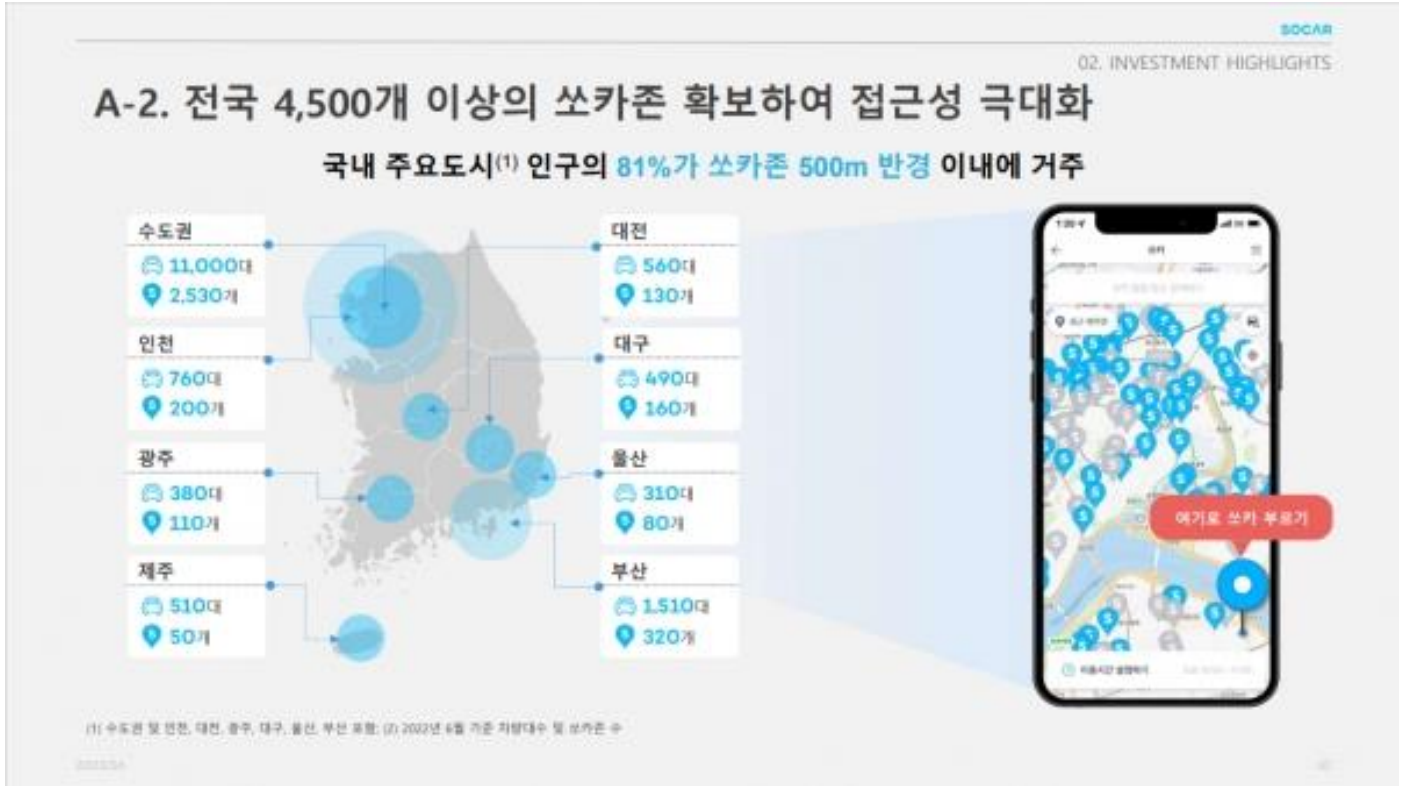


어떻게 배정되느냐는 얼마나 많은 고객들이 쏘카를 이용할 수 있는지 결정합니다.

그렇다면 예약 시간을 길게 설정한 고객들부터 차량을 배정하면 되는 것 아닐까요?

예약 테트리스 프로젝트 (Why)

이 프로젝트의 목적은 무엇인가?



1. 그동안은 차량 배정이 수작업으로 진행
   - 현재 15000대 이상의 차량
   - 4500개 이상의 쏘카존
   - 90일 전부터 예약 가능

2. 업무효율, 운영효율을 모두 잡을 수 있음!

예약 테트리스 프로젝트 (How)

수학적 모델링을 이용 – Optimization Problem Solving

최적화 문제란? (제약 조건이 있는) 목적함수를 최대화, 최소화시키는 최적의 해, 또는 최적의 해에 근접한 값을 찾는 문제

구분		장점	단점	예시
알고리즘	문제에 맞춤인 알고리즘 개발	최적해가 보장됨	개발이 어려움	Dijkstra's Algorithm
	휴리스틱 알고리즘 사용	간편하게 근사최적해를 구할 수 있음	구한 해가 최적인지 보장할 수 없음	Genetic Algorithm Ant Colony Algorithm
수학적으로 모델링 후 해찾기 도구 솔버(Solver) 사용		매우 빠른 속도로 최적해 또는 최적에 매우 근접한 해를 도출할 수 있음	문제가 복잡한 경우 연산 리소스가 매우 많이 필요함	Linear Programming Integer Programming

예약 테트리스 프로젝트 (How)

최적화 문제 구분

구분		장점	단점	예시
알고리즘	문제에 맞춤인 알고리즘 개발	최적해가 보장됨	개발이 어려움	Dijkstra's Algorithm
	휴리스틱 알고리즘 사용	간편하게 근사최적해를 구할 수 있음	구한 해가 최적인지 보장할 수 없음	Genetic Algorithm Ant Colony Algorithm
수학적으로 모델링 후 해찾기 도구 솔버(Solver) 사용		매우 빠른 속도로 최적해 또는 최적에 매우 근접한 해를 도출할 수 있음	문제가 복잡한 경우 연산 리소스가 매우 많이 필요함	Linear Programming Integer Programming

Ex) Dijkstra's Algorithm

https://www.w3schools.com/dsa/dsa_algo_graphs_dijkstra.php

- 그래프에서 최단 경로를 구하는 방법
- 한 시작점에서 nearest neighbor를 탐색
- Negative edge 있을 때에는 Bellman-Ford 방법을 이용

예약 테트리스 프로젝트 (How)

최적화 문제 구분

구분		장점	단점	예시
알고리즘	문제에 맞춤인 알고리즘 개발	최적해가 보장됨	개발이 어려움	Dijkstra's Algorithm
	휴리스틱 알고리즘 사용	간편하게 근사최적해를 구할 수 있음	구한 해가 최적인지 보장할 수 없음	Genetic Algorithm Ant Colony Algorithm
수학적으로 모델링 후 해찾기 도구 솔버(Solver) 사용		매우 빠른 속도로 최적해 또는 최적에 매우 근접한 해를 도출할 수 있음	문제가 복잡한 경우 연산 리소스가 매우 많이 필요함	Linear Programming Integer Programming

Ex) Genetic Algorithm

- 비선형 또는 계산 불가능한 문제에 많이 사용

1. 문제에 대한 초기 가능한 해(염색체)들을 선택
2. 점차적으로 변형하면서 좋은 해를 선택(진화)

https://leedakyeong.tistory.com/entry/Optimization-%EC%B5%9C%EC%A0%81%ED%99%94-%EC%95%8C%EA%B3%A0%EB%A6%AC%EC%A6%98-GA%EB%9E%80

예약 테트리스 프로젝트 (How)

최적화 문제 구분

구분		장점	단점	예시
알고리즘	문제에 맞춤인 알고리즘 개발	최적해가 보장됨	개발이 어려움	Dijkstra's Algorithm
	휴리스틱 알고리즘 사용	간편하게 근사최적해를 구할 수 있음	구한 해가 최적인지 보장할 수 없음	Genetic Algorithm Ant Colony Algorithm
수학적으로 모델링 후 해찾기 도구 솔버(Solver) 사용		매우 빠른 속도로 최적해 또는 최적에 매우 근접한 해를 도출할 수 있음	문제가 복잡한 경우 연산 리소스가 매우 많이 필요함	Linear Programming Integer Programming

예약 테트리스 프로젝트에서 선택한 방법:

수학적 모델링 후 Solver 이용

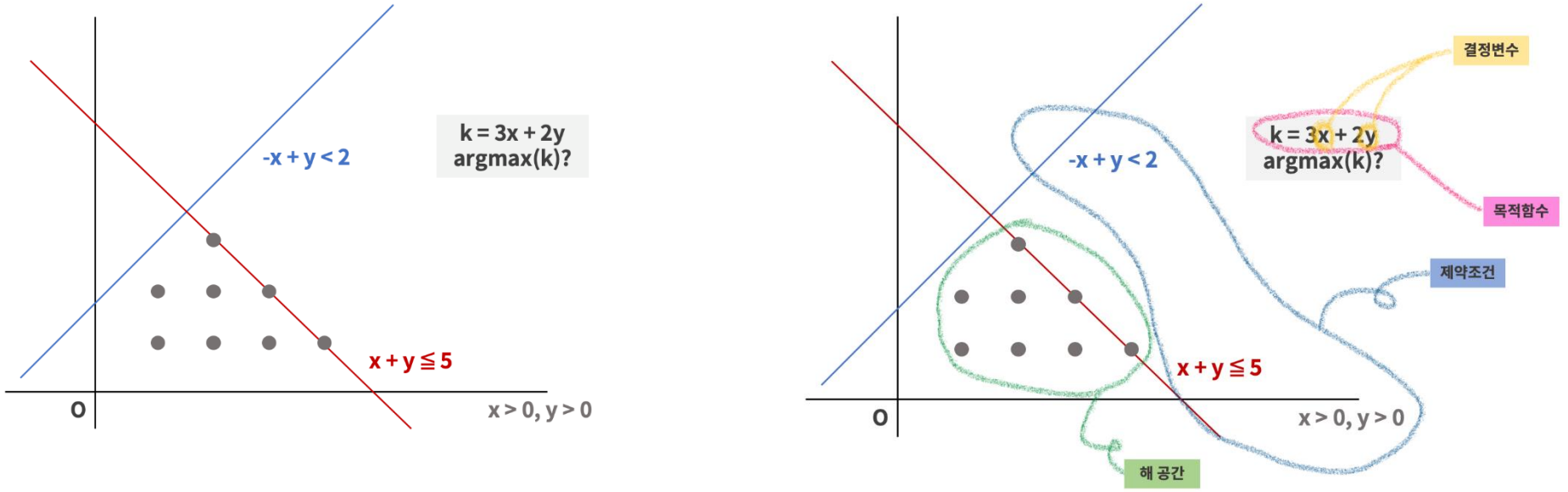
1. 최적에 가까운 해가 필요
2. 알고리즘을 직접 개발하는 것에 비해 비용,시간적인 효율↑
3. Solver로는 Google OR Tools 이용

예약 테트리스 프로젝트 (How)

1. 수학적 모델링

정수 계획법 (Integer Programming): 결정 변수가 주어진 정수 조건을 만족시키면서 목적함수를 최적화

결정 변수: 목적 함수를 결정하는 변수 (아래 예시에서 x,y가 결정 변수)



정수 선형 계획법의 예시

예약 테트리스 프로젝트 (How)

1. 수학적 모델링

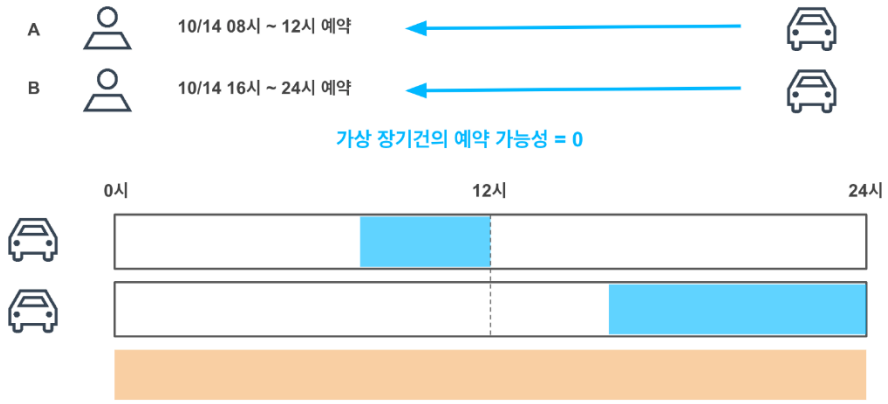
목적 함수를 어떻게 설정할 것인가?

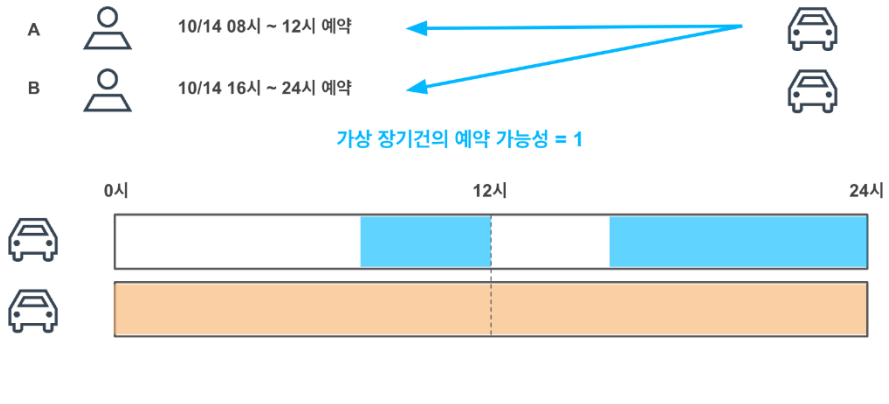
어떤 차량 배치가 더 '최적'인지를 비교하려면 최적화 지표를 정의해야 합니다. → '가상 예약' 정의 도입

실제 예약: 실제로 고객이 이용 중이거나 이용 예정인 예약

가상 예약: 실제 (장기) 예약이 가능한 여유 공간의 가상의 예약

→ 가상 예약의 예약 가능성이 클수록 최적이다!





다른 예약과의 선후 관계를 고려하지 않아 빠르게 최적값 찾기 가능

예약 테트리스 프로젝트 (How)

1. 수학적 모델링

수식으로 어떻게 표현할 것인가?

결정 변수: 예약에 차량의 배정 여부(0 or 1) Integer

목적 함수: 가상 예약 건수의 최대화

제약 조건: 1. 모든 예약은 단 하나의 차량에 배정되어야 한다.

2. 차량이 고정되어야 하는 예약은 배정된 차량이 변경되면 안 된다.

3. 임의의 서로 다른 두 개의 실제 예약은 같은 차량에 배정된 경우 서로 겹칠 수 없다.

4. 임의의 실제 예약과 가상 예약은 같은 차량에 배정된 경우 서로 겹칠 수 없다.

예약 테트리스 프로젝트 (How)

1. 수학적 모델링

R: 실제 예약 r의 집합

F: 가상 예약 f의 집합

V: 차량 v의 집합

S_r, S_f: 실제 예약 r / 가상 예약 f의 대여시작시간

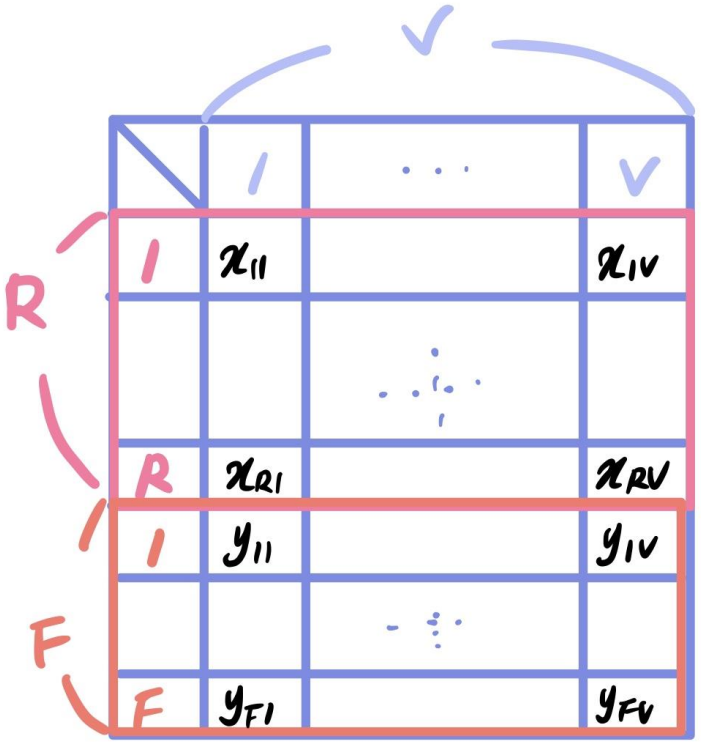
E_r, E_f: 실제 예약 r / 가상 예약 f의 대여종료시간

T_{rv}: 실제 예약 r의 차량 v 고정 배정 여부 ($T_{rv} \in \{0, 1\}$)

- 결정변수

x_{rv}: 실제 예약 r의 챠량 v 배정 여부 ($x_{rv} \in \{0, 1\}$)

y_{fv}: 가상 예약 f의 챠량 v 배정 여부 ($f_{rv} \in \{0, 1\}$)



예약 테트리스 프로젝트 (How)

1. 수학적 모델링

- 목적함수

$$\text{Maximize } \sum_v^V \sum_f^F y_{fv}$$

- 제약조건

모든 예약은 단 하나의 차량에 배정되어야 한다

$$\sum_v^V x_{rv} = 1 \, (\forall \, r \in R)$$

차량이 고정되어야하는 예약은 배정된 차량이 변경되면 안된다

$$x_{rv} \geq T_{rv} \, (\forall \, r \in R, \forall \, v \in V)$$

임의의 서로 다른 두 개의 실제예약은 같은 차량에 배정된 경우 서로 겹칠 수 없다

$$\text{if } S_{r_1} \leq S_{r_2} \leq E_{r_1},$$
$$x_{r_1 v} + x_{r_2 v} \leq 1 \, (\forall \, r_1, r_2 \in R, \forall \, v \in V)$$

임의의 가상예약과 실제예약은 같은 차량에 배정된 경우 서로 겹칠 수 없다

$$\text{if } S_r \leq S_f \leq E_r \text{ or } S_f \leq S_r \leq E_f,$$
$$x_{rv} + x_{fv} \leq 1 \, (\forall \, r \in R, f \in F)$$

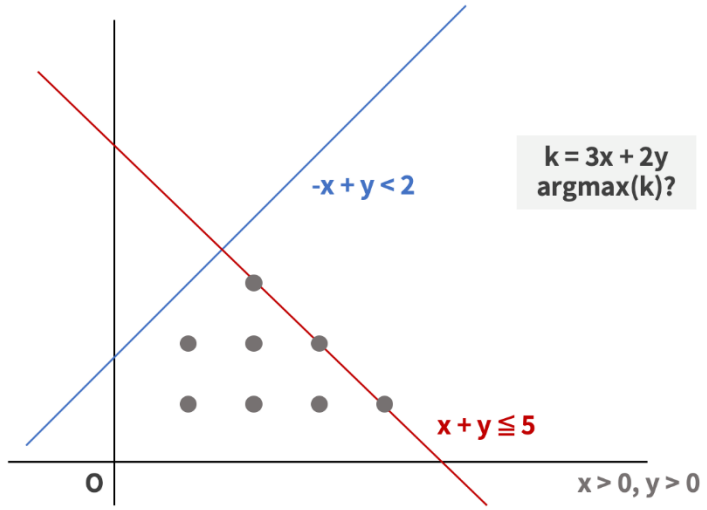
	Car A	Car B
실제 1	1	0
실제 2	1	0
가상 1	0	1

→ maximize Σ

예약 테트리스 프로젝트 (How)

2. 최적화 모델 구현하기

Google OR-Tools Solver – 무료로 이용 가능, 커스텀 설정 지원




```python
from ortools.sat.python import cp_model


model = cp_model.CpModel()

x = model.NewIntVar(1, 100, "x") # 적당히 큰 최대값 설정
y = model.NewIntVar(1, 100, "y") # 적당히 큰 최대값 설정

k = 3 * x + 2 * y

model.Add(x + y <= 5)
model.Add(-x + y < 2)

model.Maximize(k)

solver = cp_model.CpSolver()
status = solver.Solve(model)

if status == cp_model.OPTIMAL or status == cp_model.FEASIBLE:
    print(f'k = {solver.Value(k)}')
    print(f'x = {solver.Value(x)}')
    print(f'y = {solver.Value(y)}')
else:
    print('Solution Not Found')
```

예약 테트리스 프로젝트 (How)

3. 최적화 모델 배포 (참고만)

최적화를 진행할 서버: 최적화 서버

기존 서버: 쏘카 서비스 서버

고려할 점

1. 예약이 새로 들어올 때마다 다시 최적화를 진행할 것인가?
    - 수작업으로 할 때도 1시간에 한 번씩 업데이트
    - 예약을 일정 간격으로 정리하는 Batch 방식

2. 서버끼리 어떻게 데이터를 주고 받을 것인가?
    - API 형식 -> 연산 시간이 오래 걸림
    - 메시지 형식 -> Pull & Push

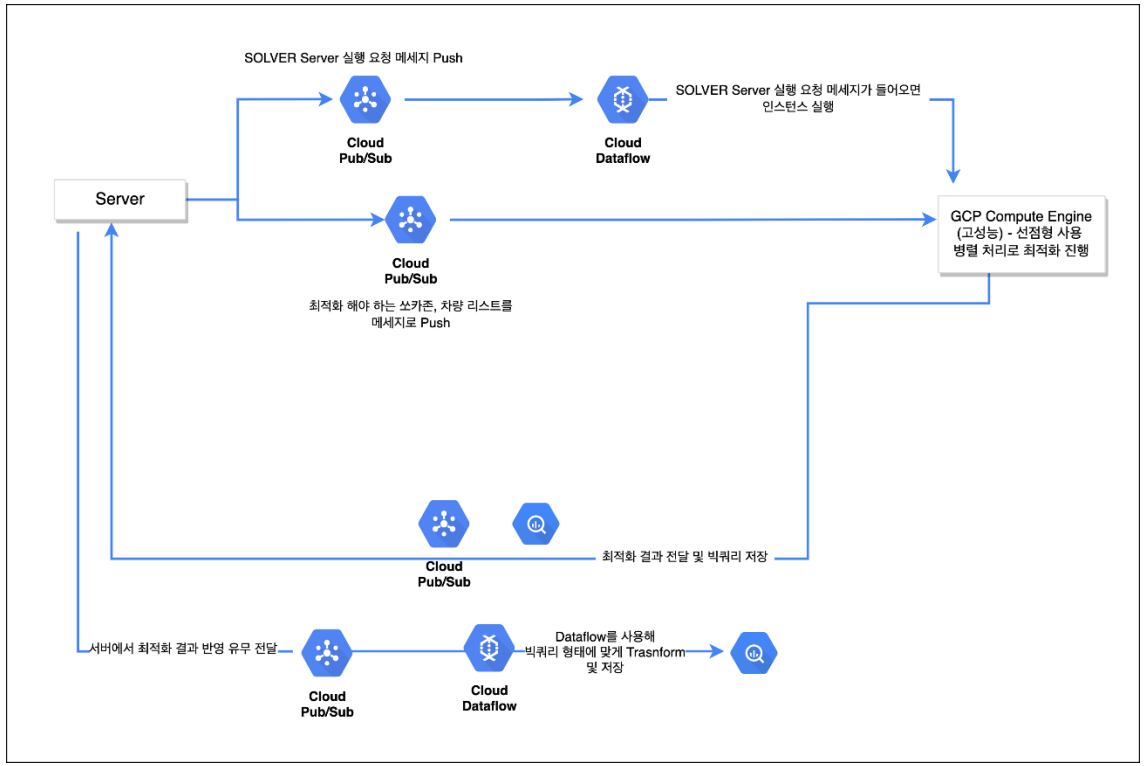
예약 테트리스 프로젝트 (How)

3. 최적화 모델 배포 (참고만)

최적화를 진행할 서버: 최적화 서버

기존 서버: 쏘카 서비스 서버

상호작용

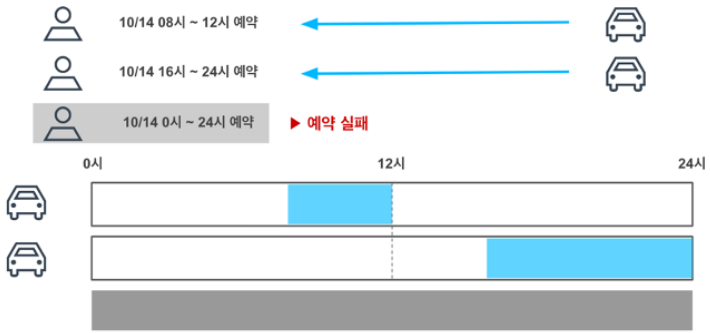


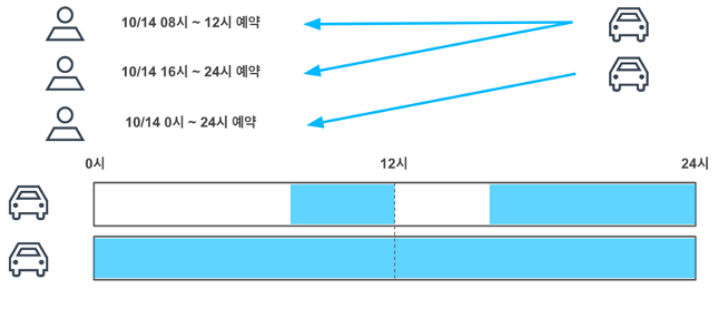
순서 요약

1. 쏘카 서비스 서버에서 최적화 서버를 띄우는 메시지 보냄
2. 쏘카 서비스 서버는 띄워진 최적화 서버에 데이터 전송(Push)
3. 최적화 서버에서 메시지 받고(Pull) Ray로 최적화를 병렬로 실행
4. 최적화 결과를 쏘카 서비스 서버로 보냄
5. 쏘카 서비스 서버에서 결과를 데이터베이스에 저장하고 성공 여부를 최적화 서버에 전송
6. 실행되지 않을 때(특정 조건을 만족하면) 최적화 서버 종료
7. 선점형 인스턴스가 구글에 의해 회수된다면 Shutdown Script가 다시 최적화 서버 실행

예약 테트리스 프로젝트 성과

사용한 차량 대비 차량이 얼마나 점유되었는지를 비교하면 성과를 알 수 있습니다.





차량사용비율

(=예약에 사용된 차량 / 전체 운영차량)

$$\frac{2대}{2대} = 1.0$$

$$\frac{2대}{2대} = 1.0$$

가동률

(=예약에 점유된 시간 / 전체 판매가능 시간)

$$\frac{4시간 + 8시간}{24시간 \times 2대} = 0.25$$

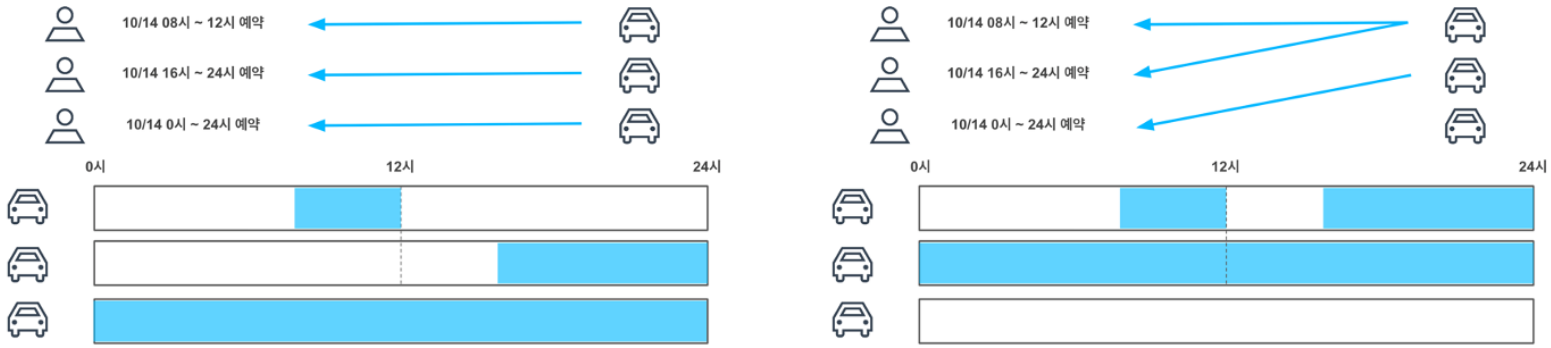
$<$

$$\frac{4시간 + 8시간 + 24시간}{24시간 \times 2대} = 0.75$$

같은 대수의 차량으로 더 많은 예약을 수용

예약 테트리스 프로젝트 성과

사용한 차량 대비 차량이 얼마나 점유되었는지를 비교하면 성과를 알 수 있습니다.



차량사용비율

(=예약에 사용된 차량 / 전체 운영차량)

$$\frac{3대}{3대} = 1.0$$

$$\frac{2대}{3대} = 0.67$$

가동률

(=예약에 점유된 시간 / 전체 판매가능 시간)

$$\frac{4시간 + 8시간 + 24시간}{24시간 \times 3대} = 0.5$$

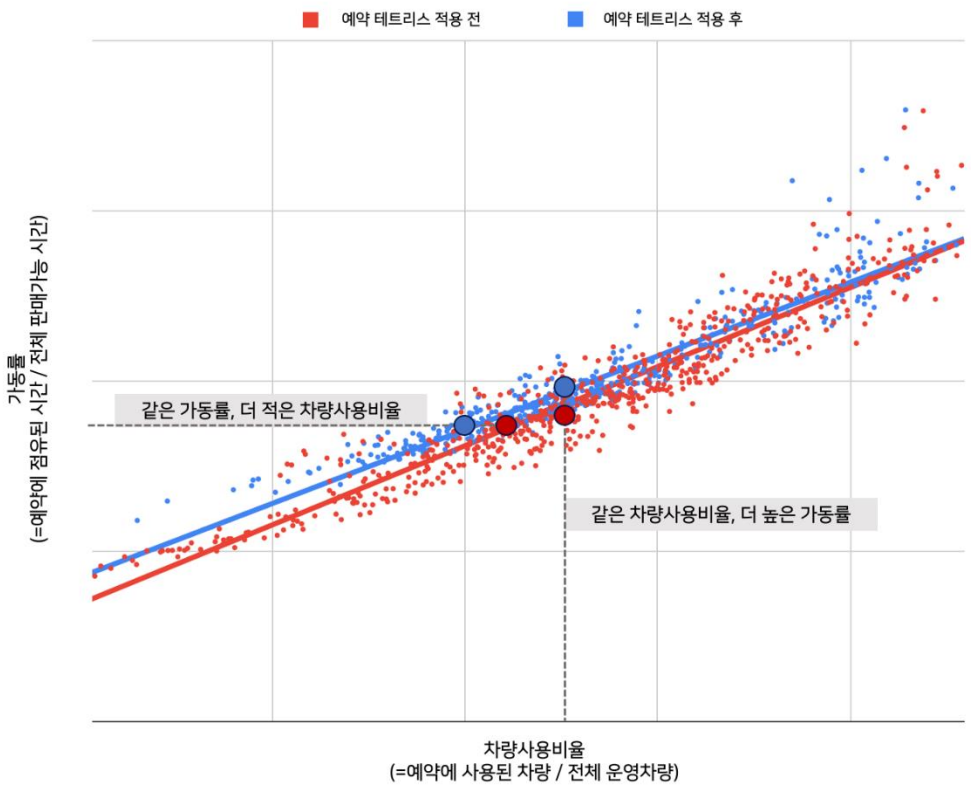
$$\frac{4시간 + 8시간 + 24시간}{24시간 \times 3대} = 0.5$$

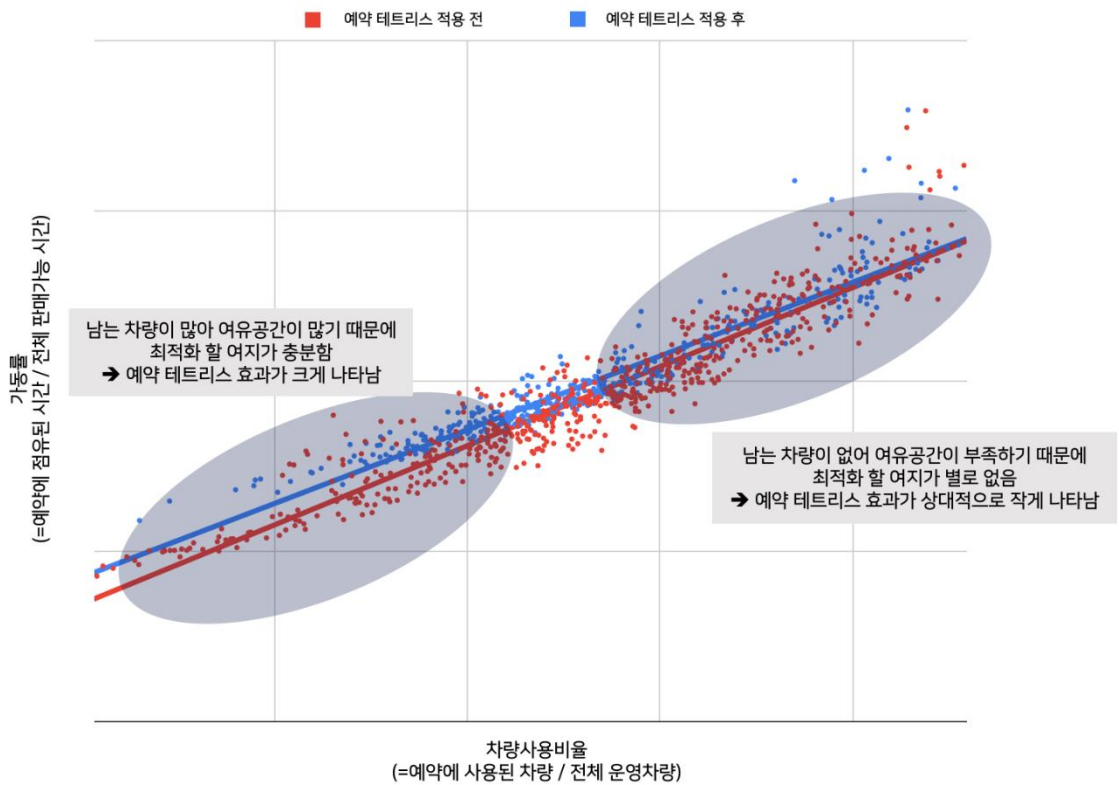
같은 예약량이 더 적은 차량 대수로도 수용

예약 테트리스 프로젝트 성과

사용한 차량 대비 차량이 얼마나 점유되었는지를 비교하면 성과를 알 수 있습니다.

성과 시각화를 통해 인사이를 도출하며 프로젝트를 마무리할 수 있습니다.





Reference

Socar Tech Blog: 쏘카 예약을 효율적으로 - 수학적 모델링을 활용한 쏘카 예약 테트리스

https://tech.socarcorp.kr/data/2022/06/10/reservation-tetris.html

기초 최적화 이론 강의: https://www.youtube.com/watch?v=mmqiHA1RiWE&t=2099s&ab_channel=DrWill

Dijkstra's Algorithm https://www.w3schools.com/dsa/dsa_algo_graphs_dijkstra.php

Genetic Algorithm https://leedakyeong.tistory.com/entry/Optimization-%EC%B5%9C%EC%A0%81%ED%99%94-%EC%95%8C%EA%B3%A0%EB%A6%AC%EC%A6%98-GA%EB%9E%80

감사합니다