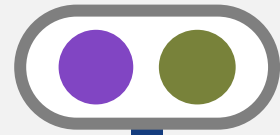


개인 분석
프로젝트

지하철 승차인원 예측



C O N T E N T S



1

프로젝트 개요 및 목적



2

분석과정 및 결과



3

활용 방안 및 기대효과

1

프로젝트 개요 및 목적

“ 수송 인원 총 27억 2,625만명, 하루 평균 746만 9,180명 ”

[2019년 서울 지하철 이용객 현황]

출처 : 서울교통공사

1

프로젝트 개요 및 목적

[2019년 서울 지하철 호선별 수송 실적 비교]

출처 : 서울교통공사

(단위 : 천명, %)

구 분	총인원 (일평균)	전년대비	증감 (일평균)	수송점유율	비고 (공사 운영구간)
1호선	172,369 (472)	101.8%	3,014 (8)	6.3%	서울역~청량리
2호선	811,959 (2,224)	100.7%	5,398 (15)	29.8%	전 구간
3호선	329,660 (903)	101.5%	4,829 (13)	12.1%	지축~오금
4호선	326,793 (895)	100.3%	896 (2)	12.0%	당고개~남태령
5호선	333,836 (914)	101.6%	5,308 (15)	12.2%	전 구간
6호선	204,575 (560)	100.6%	1,288 (4)	7.5%	전 구간
7호선	380,142 (1,041)	101.0%	3,882 (11)	13.9%	전 구간
8호선	112,086 (307)	104.1%	4,336 (12)	4.1%	전 구간
9호선	54,825 (150)	-	-	2.0%	언주~중앙보훈병원
합계	2,726,250 (7,469)	-	28,983 (79)	100%	

2019년도 수송인원 일평균이 가장 높은 호선은 **2호선**,
가장 낮은 호선은 **9호선**. 이에 따른 수송 점유율 확보.

[일평균 이용객수 상위 / 하위 3개역]

출처 : 서울교통공사



동일한 호선간에도 승차인원의 편차가 큼.

1 프로젝트 개요 및 목적

지하철 정원 관련 주요 수치



1량 당 정원	160명
1편성 정원	1600명 내외 1~4호선:10량, 기타 노선:6~8량
1량 좌석 수	54석
1량 손잡이 수	76개
출입문	8개(양쪽 각 4개)
1량 규격	19.50m x 2.92m(길이x폭) 실내 기준
적정 중량	20톤
최대 적재 중량	30톤

[지하철 정원 관련 주요 수치]

출처 : 서울교통공사

“ 지하철 1량 당
탑승 적정 인원은 160명 ”

“ 일평균 이용량이 가장 많은 2호선의 일간
평균 이용인원 2,225,000명 ”

“ 일간 2호선의 평일 운행회수 976회 ”

“ 1편성 당 평균 이용인원 약 2,280명 ”
(적정 인원 : 1600명 내외)

출처 : 서울시 지하철 운행현황 통계

적정 인원을 초과하는 이용객으로 인한
지하철 이용의 불편함

1

프로젝트 개요 및 목적

“

서민의 발이라 불리는 지하철,

**일자별 노선의 승차인원 예측해 지하철 운행의 유동적인
증/감축을 통한 시민들의 지하철 이용의 불편함을 해소하고자 함**

”

“

지하철 승차인원 예측을 위한 프로젝트 진행

”

진행 방향 : 2016~2018년도의 승차인원 데이터를 활용한 2019년도 승차인원 예측을 통한 모델링

2

분석과정 및 결과

데이터
수집

전처리

EDA

변수 선택

파생변수
생성

모델 선택

모델
최적화

모델 평가

주요 분석 데이터는 공공데이터인

“서울시 지하철호선별 역별 승차 인원 정보”의 2016 ~ 2019년도 데이터 활용



[교통]

서울시 지하철호선별 역별 승하차 인원 정보

교통카드(선불교통카드 및 1회용 교통카드)를 이용한 지하철호선별 역별(서울교통공사, 한국철도공...
수정일자: 2020-06-28 제공기관: 서울특별시 제공부서: 도시교통실 교통기획관 교통정책과

SHEET

OpenAPI

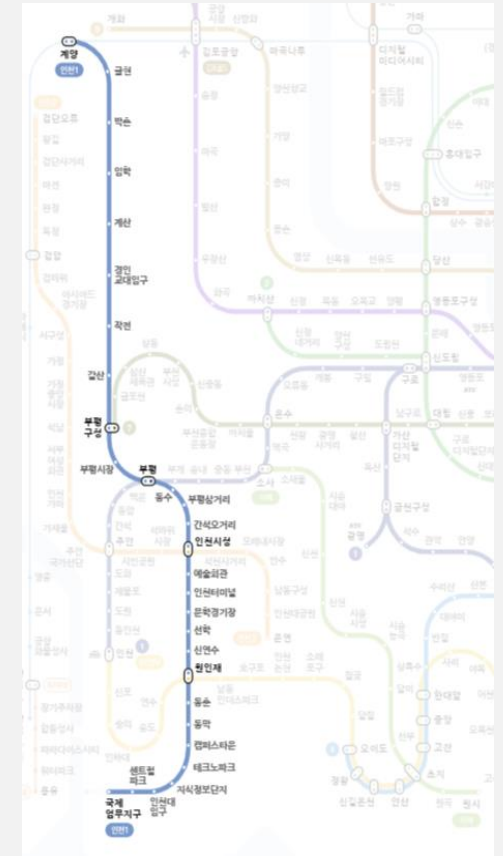
FILE

공공 데이터

출처: 서울 열린 데이터 광장

샘플 데이터를 통해 각 일자별 호선명, 역명, 승차총승객수,
하차총승객수, 등록일자 등의 정보가 포함되었음을 확인

	사용일자	호선명	역명	승차총승객수	하차총승객수	등록일자
0	20200905	우이신설선	4.19민주묘지	1682	1530	20200908
1	20200905	경원선	가봉	3846	3742	20200908
2	20200905	8호선	가락시장	3942	3958	20200908
3	20200905	3호선	가락시장	4174	3958	20200908
4	20200905	7호선	가산디지털단지	10065	9427	20200908

인천 1호선 등의 주요 서울의 노선이 아닌 호선은 분석에서 제외
→ 1~9호선을 주요 분석 호선으로 설정

2

분석과정 및 결과

지하철 승차인원의 영향을 줄 수 있다고 판단되는 일자별
기상 데이터(기온, 강수량, 풍속), 휴일정보 데이터 등의 외부 데이터 활용



데이터
수집

전처리

EDA

변수 선택

파생변수
생성

모델 선택

모델
최적화

모델 평가

Home > 기후통계분석 > 통계분석 > 기온분석

기온분석

그래프 분포도

■ 자료설명

지점별로 기온의 시계열 분석을 확인합니다.
일, 월, 연의 평균기온, 최저기온, 최고기온을 각각 조회할 수 있습니다.

* '지역/지점'의 '지역'은 전국 및 광역 단위의 평균 제공(1973년~) (전국 및 광역별 평균에 사용된 지점은 전국 평균산출에 사용되는 45개 지점이며, 제주도는 제주시와 서귀포시 자료임)

• 자료구분: 일 • 자료형태: 기본 • 기간: 20200819 ~ 20200917

• 지역/지점: 서울 선택

> 검색

CSV 다운로드 Excel 다운로드

출처 : 기상청 기상자료개방포털

오픈API 상세

XML **특일 정보**

(천문우주정보)국경일정보, 공휴일정보, 기념일정보, 24절기정보, 잡절정보를 조회하는 서비스입니다.

11 관심

f t u URL 복사

활용신청

출처 : 공공데이터 포털

2

분석과정 및 결과

데이터
수집

전처리

EDA

변수 선택

파생변수
생성

모델 선택

모델
최적화

모델 평가

서울시 지하철호선별 역별 승차 인원 데이터

각 연도의 월별 승차인원 데이터를
각 연도별 데이터로 통합

```
[4] # 데이터 로드
subway_2016 = pd.DataFrame()
subway_2017 = pd.DataFrame()
subway_2018 = pd.DataFrame()
subway_2019 = pd.DataFrame()
subway_2020 = pd.DataFrame()

for f in subway_2016_list:
    tmp = pd.read_csv(f)
    subway_2016 = subway_2016.append(tmp)

for f in subway_2017_list:
    tmp = pd.read_csv(f)
    subway_2017 = subway_2017.append(tmp)

for f in subway_2018_list:
    tmp = pd.read_csv(f)
    subway_2018 = subway_2018.append(tmp)

for f in subway_2019_list:
    tmp = pd.read_csv(f)
    subway_2019 = subway_2019.append(tmp)

for f in subway_2020_list:
    tmp = pd.read_csv(f)
    subway_2020 = subway_2020.append(tmp)
```

기상 데이터, 휴일정보 데이터

기온, 강수량, 풍속, 휴일정보 데이터 로드

```
[5] # 데이터 로드
holiday_table = pd.read_csv("holiday_table.csv", index_col=0)
holiday_table_2020 = pd.read_csv("holiday_table_2020.csv", index_col=0)
rainfall = pd.read_csv('rainfall.csv')
rainfall_2020 = pd.read_csv('rainfall_2020.csv')
temperature = pd.read_csv('temperature.csv')
temperature_2020 = pd.read_csv("temperature_2020.csv")
wind = pd.read_csv("wind.csv")
wind_2020 = pd.read_csv("wind_2020.csv")

temperature_2020 = temperature_2020.drop(temperature_2020[temperature_2020['일시'].isnull()].index)
temperature_2020 = temperature_2020.drop(['ㄹ지점번호', '지점명', 'ㄹ최고기온시각', '최저기온시각'], axis=1)
temperature_2020.columns = ['일시', '평균기온', '최고기온', '최저기온']
temperature_2020['일시'] = pd.to_datetime(temperature_2020['일시'], format='%Y-%m-%d')
temperature_2020['일시'] = temperature_2020['일시'].map(lambda x: x.strftime("%Y%m%d%H%M%S"))

holiday_table = pd.concat([holiday_table, holiday_table_2020], axis=0)
rainfall = pd.concat([rainfall, rainfall_2020], axis=0)
temperature = pd.concat([temperature, temperature_2020], axis=0)
wind = pd.concat([wind, wind_2020], axis=0)
```

2

분석과정 및 결과

데이터
수집

전처리

EDA

변수 선택

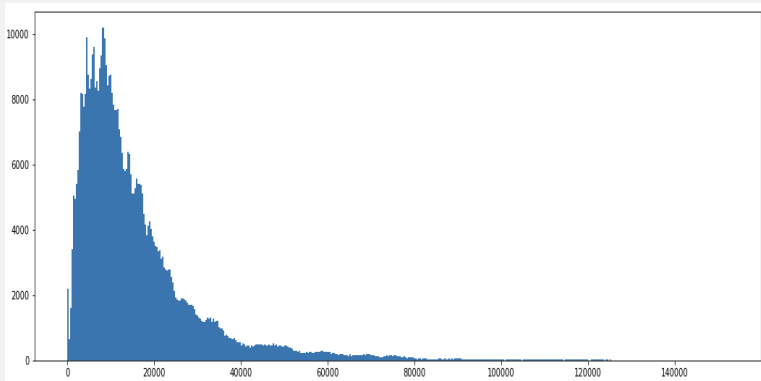
파생변수
생성

모델 선택

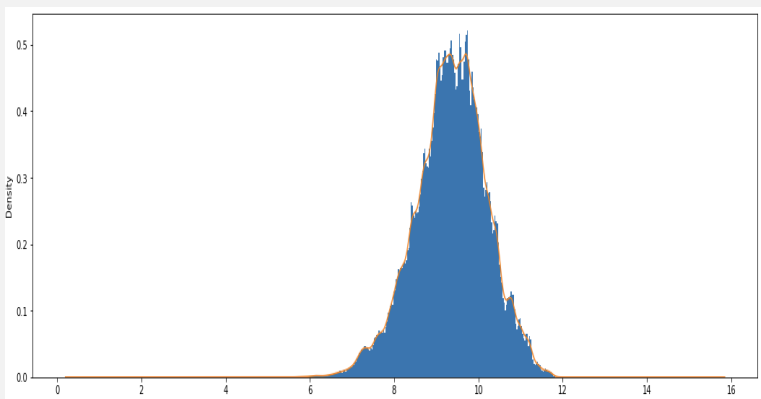
모델
최적화

모델 평가

타깃 변수(승차 총 승객수) 확인

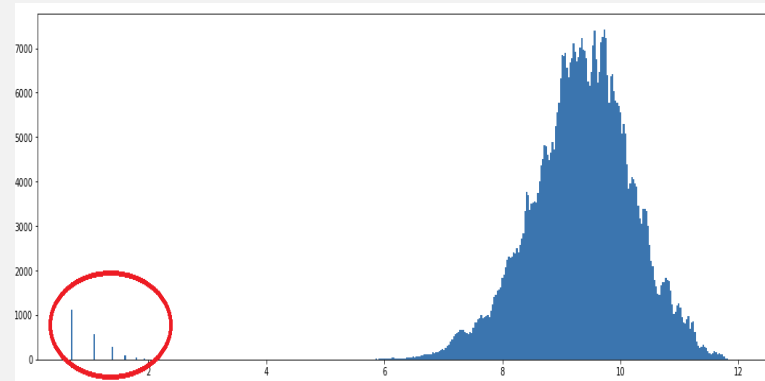


타깃변수 분포 확인 결과
오른쪽으로 꼬리가 늘어진 분포를 지님을 확인



이상치 제거 후 타겟의 밀도함수를 통한 타깃변수의 정규분포 근사 확인

로그 변환



로그 변환 후 이상치 탐색

	사용일자	노선명	역명	승차총승객수
44	2016-01-01	3호선	충무로	1.0
142	2016-01-01	6호선	연신내	3.0
347	2016-01-02	3호선	충무로	2.0
633	2016-01-03	3호선	충무로	2.0
772	2016-01-03	6호선	연신내	1.0
...
522019	2020-09-28	6호선	연신내	1.0
522221	2020-09-29	3호선	충무로	1.0
522325	2020-09-29	6호선	연신내	1.0
522562	2020-09-30	6호선	신내	1.0
522596	2020-09-30	6호선	연신내	1.0

충무로 / 연신내 등에서 승차승객이 한자리 수인 이상치 발견

이상치 제거

2

분석과정 및 결과

데이터
수집

전처리

EDA

변수 선택

파생변수
생성

모델 선택

모델
최적화

모델 평가

외부 데이터(기온, 강수량, 풍속, 휴일 정보)

날짜지점번호	지점명	일시	평균기온(℃)	최고기온(℃)	최저기온(℃)	최저기온시각
0	\t108 서울	2020-01-01	-2.2	0.3	14.57	-6.5
1	\t108 서울	2020-01-02	1.0	3.8	15.00	-0.7
2	\t108 서울	2020-01-03	-0.1	4.6	15.47	-3.4
3	\t108 서울	2020-01-04	1.2	6.1	14.50	-2.8
4	\t108 서울	2020-01-05	1.3	6.6	14.53	-3.2
...
298	\t NaN	NaN	NaN	NaN	NaN	NaN
299	\t NaN	NaN	NaN	NaN	NaN	NaN
300	\t NaN	NaN	NaN	NaN	NaN	NaN
301	\t NaN	NaN	NaN	NaN	NaN	NaN
302	\t NaN	NaN	NaN	NaN	NaN	NaN

날짜지점번호	지점명	일시	강수량(mm)	1시간최다강수량(mm)	1시간최다강수량시각
0	\t108 서울	2016-01-01	NaN	NaN	NaN
1	\t108 서울	2016-01-02	NaN	NaN	NaN
2	\t108 서울	2016-01-03	NaN	NaN	NaN
3	\t108 서울	2016-01-04	NaN	NaN	NaN
4	\t108 서울	2016-01-05	NaN	NaN	NaN
...
296	\t108 서울	2020-10-23	NaN	NaN	NaN
297	\t NaN	NaN	NaN	NaN	NaN
298	\t NaN	NaN	NaN	NaN	NaN
299	\t NaN	NaN	NaN	NaN	NaN
300	\t NaN	NaN	NaN	NaN	NaN

날짜지점번호	지점명	일시	평균풍속(m/s)	최대풍속(m/s)	최대풍속방향(deg)	최대풍속시각	최대순간풍속(m/s)	최대순간풍속방향(deg)	최대순간풍속시각
0	108 서울	2016-01-01	1.6	3.5	270.0	15:17	5.6	280.0	16:07
1	108 서울	2016-01-02	2.0	4.5	330.0	15:30	6.9	320.0	15:12
2	108 서울	2016-01-03	1.8	4.0	330.0	12:38	5.7	340.0	23:38
3	108 서울	2016-01-04	3.1	5.1	50.0	13:38	8.6	320.0	12:52
4	108 서울	2016-01-05	2.3	4.6	360.0	12:21	7.3	360.0	12:19
...
294	\t108 서울	2020-10-21	1.9	3.8	160.0	15:11	6.2	180.0	15:04
295	\t108 서울	2020-10-22	2.8	5.7	330.0	20:28	10.4	340.0	20:20
296	\t108 서울	2020-10-23	3.0	5.5	290.0	18:21	10.0	270.0	13:57
297	\t NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
298	\t NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

	dateKind	dateName	isHoliday	locdate	seq
0	1	신정	Y	20160101	1
1	1	설날	Y	20160207	1
2	1	설날	Y	20160208	1
3	1	설날	Y	20160209	1
4	1	대체공휴일	Y	20160210	1
...
12	1	추석	Y	20201001	1
13	1	추석	Y	20201002	1
14	1	개천절	Y	20201003	1
15	1	한글날	Y	20201009	1
16	1	기독탄신일	Y	20201225	1

주요 변수 추출 & 변수명 정리

	일시	평균기온	최고기온	최저기온
0	20160101000000	1.2	4.0	-3.3
1	20160102000000	5.7	9.5	1.0
2	20160103000000	6.5	9.4	5.1
3	20160104000000	2.0	5.3	-2.5
4	20160105000000	-2.7	1.5	-4.8
...
292	20201019000000	15.1	21.3	9.8
293	20201020000000	15.2	21.7	9.9
294	20201021000000	14.1	17.3	11.8
295	20201022000000	13.5	18.2	8.4
296	20201023000000	8.6	13.3	5.4

	일시	강수량
0	2016-01-01	NaN
1	2016-01-02	NaN
2	2016-01-03	NaN
3	2016-01-04	NaN
4	2016-01-05	NaN
...
296	2020-10-23	NaN

	일시	평균풍속	최대풍속
0	2016-01-01	1.6	3.5
1	2016-01-02	2.0	4.5
2	2016-01-03	1.8	4.0
3	2016-01-04	3.1	5.1
4	2016-01-05	2.3	4.6
...
294	2020-10-21	1.9	3.8
295	2020-10-22	2.8	5.7
296	2020-10-23	3.0	5.5

	날짜명	휴일여부	locdate
0	신청	Y	20160101
1	설날	Y	20160207
2	설날	Y	20160208
3	설날	Y	20160209
4	대체공휴일	Y	20160210
...
12	추석	Y	20201001
13	추석	Y	20201002
14	개천절	Y	20201003
15	한글날	Y	20201009
16	기독탄신일	Y	20201225

2

분석과정 및 결과

데이터
수집

전처리

EDA

변수 선택

파생변수
생성

모델 선택

모델
최적화

모델 평가

외부 데이터(기온, 강수량, 풍속, 휴일 정보)

	일시	평균풍속	최대풍속
0	2016-01-01	1.6	3.5
1	2016-01-02	2.0	4.5
2	2016-01-03	1.8	4.0
3	2016-01-04	3.1	5.1
4	2016-01-05	2.3	4.6
...
294	2020-10-21	1.9	3.8
295	2020-10-22	2.8	5.7
296	2020-10-23	3.0	5.5
297	NaN	NaN	NaN
298	NaN	NaN	NaN

기온, 강수량, 풍속 데이터 말미에 원인 모를
결측치 존재 -> 제거

	일시	평균풍속	최대풍속
652	20171014000000	NaN	NaN
704	20171205000000	NaN	NaN
705	20171206000000	NaN	3.5

2017년 10월 14일, 12월 5일, 6일의 풍속데이터 결측치
-> 최대풍속, 평균풍속의 중앙값으로 대체

	일시	강수량
0	2016-01-01	NaN
1	2016-01-02	NaN
2	2016-01-03	NaN
3	2016-01-04	NaN
4	2016-01-05	NaN
...
296	2020-10-23	NaN

강수량 데이터의 NaN 값-> 비가 오지 않았음을 의미
-> 0으로 대체

2

분석과정 및 결과

데이터
수집

전처리

EDA

변수 선택

파생변수
생성

모델 선택

모델
최적화

모델 평가

각 데이터의 시간 정보 통일

locdate날짜명휴일여부				일시평균기온최고기온최저기온				
0	20160101000000	신정	Y	0	20160101000000	1.2	4.0	-3.3
1	20160207000000	설날	Y	1	20160102000000	5.7	9.5	1.0
2	20160208000000	설날	Y	2	20160103000000	6.5	9.4	5.1
3	20160209000000	설날	Y	3	20160104000000	2.0	5.3	-2.5
4	20160210000000	대체공휴일	Y	4	20160105000000	-2.7	1.5	-4.8
...
12	20201001000000	추석	Y	292	20201019000000	15.1	21.3	9.8
13	20201002000000	추석	Y	293	20201020000000	15.2	21.7	9.9
14	20201003000000	개천절	Y	294	20201021000000	14.1	17.3	11.8
15	20201009000000	한글날	Y	295	20201022000000	13.5	18.2	8.4
16	20201225000000	기독탄신일	Y	296	20201023000000	8.6	13.3	5.4
일시강수량				일시평균풍속최대풍속				
0	20160101000000		0.0	0	20160101000000		1.6	3.5
0	20200101000000		0.1	1	20160102000000		2.0	4.5
1	20160102000000		0.0	2	20160103000000		1.8	4.0
1	20200102000000		0.0	3	20160104000000		3.1	5.1
2	20160103000000		0.0	4	20160105000000		2.3	4.6
...
294	20201021000000		0.0	292	20201019000000		1.7	3.4
295	20161022000000		0.0	293	20201020000000		1.6	3.1
295	20201022000000		0.0	294	20201021000000		1.9	3.8
296	20161023000000		2.5	295	20201022000000		2.8	5.7
296	20201023000000		0.0	296	20201023000000		3.0	5.5

각 데이터의 시간 정보를
yyyyMMddHHmmss 꼴로 통일

2

분석과정 및 결과

데이터
수집

전처리

EDA

변수 선택

파생변수
생성

모델 선택

모델
최적화

모델 평가

데이터 통합

[260] # 지하철 데이터 통합

subway_total = pd.concat([subway_2016, subway_2017, subway_2018, subway_2019, subway_2020])

subway_total = subway_total.drop(['하차총승객수', '등록일자'], axis=1)

추가데이터 합치기

subway_total_all = subway_total.merge(holiday_table, how = 'left', left_on = '사용일자', right_on = 'locdate')

subway_total_all = subway_total_all.merge(temperature, how = 'left', left_on = '사용일자', right_on = '일시')

subway_total_all = subway_total_all.merge(rainfall, how = 'left', left_on = '사용일자', right_on = '일시')

subway_total_all = subway_total_all.merge(wind, how = 'left', left_on = '사용일자', right_on = '일시')



	사용일자	노선명	역명	승차총승객수	날짜명	휴일여부	평균기온	최고기온	최저기온	강수량	index	평균풍속	최대풍속	year	month	day	week	date
0	2016-01-01	4호선	동작	1213.0	신정	Y	1.2	4.0	-3.3	0.0	0	1.6	3.5	2016	1	1	53	Friday
1	2016-01-01	4호선	동작	1213.0	신정	Y	1.2	4.0	-3.3	0.0	0	1.6	3.5	2016	1	1	53	Friday
2	2016-01-01	4호선	이촌	3365.0	신정	Y	1.2	4.0	-3.3	0.0	0	1.6	3.5	2016	1	1	53	Friday
3	2016-01-01	4호선	이촌	3365.0	신정	Y	1.2	4.0	-3.3	0.0	0	1.6	3.5	2016	1	1	53	Friday
4	2016-01-01	4호선	신용산	5746.0	신정	Y	1.2	4.0	-3.3	0.0	0	1.6	3.5	2016	1	1	53	Friday
...
1328780	2020-09-30	2호선	독성	5557.0	주석	Y	19.6	25.7	16.2	38.3	273	1.7	4.1	2020	9	30	40	Wednesday
1328781	2020-09-30	2호선	한양대	1870.0	주석	Y	19.6	25.7	16.2	38.3	273	1.7	4.1	2020	9	30	40	Wednesday
1328782	2020-09-30	2호선	한양대	1870.0	주석	Y	19.6	25.7	16.2	38.3	273	1.7	4.1	2020	9	30	40	Wednesday
1328783	2020-09-30	2호선	왕십리(성동구청)	6817.0	주석	Y	19.6	25.7	16.2	38.3	273	1.7	4.1	2020	9	30	40	Wednesday
1328784	2020-09-30	2호선	왕십리(성동구청)	6817.0	주석	Y	19.6	25.7	16.2	38.3	273	1.7	4.1	2020	9	30	40	Wednesday

각 연도별 지하철 승차 인원 데이터
+
외부 데이터 통합

연도별, 월별, 주차별, 요일별 분석을 위해
시간 데이터로부터 각 데이터 추출

휴일이 아닌 일자의 휴일여부 -> N으로 변경

토요일, 일요일의 휴일여부 -> Y로 변경

2

분석과정 및 결과

데이터
수집

전처리

EDA

변수 선택

파생변수
생성

모델 선택

모델
최적화

모델 평가

라벨 인코딩(Label Encoding)

휴일여부	date	노선명	역명
0	1	1	287
1	2	1	179
1	2	1	268
1	2	1	269
1	2	1	270
...
1	4	9	20
1	4	9	138
1	4	9	182
1	4	9	40
0	4	9	24

휴일여부, 요일, 노선명, 역명 등 문자열로 주어진 데이터들을
모델에 활용하기 위한 숫자형태로 바꾸어 주는 라벨 인코딩 진행

=> 데이터 사이즈를 고려해 라벨 인코딩이 더 적합하다고 판단

2

분석과정 및 결과

데이터
수집

전처리

EDA

변수 선택

파생변수
생성

모델 선택

모델
최적화

모델 평가

분석용 데이터

	노선명	역명	승차총승객수	휴일여부	평균기온	강수량	최대풍속	year	month	day	date	누적휴일	명절여부	환승노선	승차총승객수_log
308777	1	287	28296.0	0	0.034650	-0.272244	-1.595552	2018	1	14	1	0	0	0	10.250511
282858	1	179	15201.0	1	1.478218	0.328971	-0.776419	2018	9	20	2	1	0	1	9.629182
282859	1	268	29126.0	1	1.478218	0.328971	-0.776419	2018	9	20	2	1	0	0	10.279421
282860	1	269	28728.0	1	1.478218	0.328971	-0.776419	2018	9	20	2	1	0	2	10.265662
282861	1	270	19847.0	1	1.478218	0.328971	-0.776419	2018	9	20	2	1	0	0	9.895859
...
476384	9	20	9107.0	1	0.323363	-0.272244	1.107585	2020	6	23	4	1	0	2	9.116908
476385	9	138	1269.0	1	0.323363	-0.272244	1.107585	2020	6	23	4	1	0	0	7.146772
476386	9	182	16558.0	1	0.323363	-0.272244	1.107585	2020	6	23	4	1	0	0	9.714685
476375	9	40	2801.0	1	0.323363	-0.272244	1.107585	2020	6	23	4	1	0	0	7.938089
31601	9	24	3311.0	0	-0.114364	-0.272244	0.288453	2016	6	5	4	0	0	0	8.105308

520610 rows x 15 columns

520610행
15열

	노선명	역명	승차총승객수	휴일여부	평균기온	강수량	최대풍속	year	month	day	date	누적휴일	명절여부	환승노선	승차총승객수_log
count	520610.000000	520610.000000	520610.000000	520610.000000	5.206100e+05	5.206100e+05	5.206100e+05	520610.000000	520610.000000	520610.000000	520610.000000	520610.000000	520610.000000	520610.000000	520610.000000
mean	5.026653	156.765508	16158.036123	0.322764	-2.615782e-16	1.823595e-14	2.111770e-16	2017.894385	5.683433	14.841982	2.999422	0.697386	0.027087	0.279945	9.348408
std	2.281421	91.032001	14729.559297	0.467534	1.000001e+00	1.000001e+00	1.000001e+00	1.373147	3.449799	8.948887	2.000434	1.247472	0.162338	0.511433	0.853744
min	1.000000	0.000000	60.000000	0.000000	-2.647593e+00	-2.722442e-01	-2.414684e+00	2016.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	4.110874
25%	3.000000	80.000000	6754.000000	0.000000	-8.501180e-01	-2.722442e-01	-6.945061e-01	2017.000000	3.000000	7.000000	1.000000	0.000000	0.000000	0.000000	8.818038
50%	5.000000	157.000000	11844.000000	0.000000	1.184697e-01	-2.722442e-01	-1.211135e-01	2018.000000	6.000000	15.000000	3.000000	0.000000	0.000000	0.000000	9.379661
75%	7.000000	236.000000	20264.000000	1.000000	8.821638e-01	-2.481956e-01	6.161056e-01	2019.000000	9.000000	22.000000	5.000000	1.000000	0.000000	0.000000	9.916651
max	9.000000	315.000000	152285.000000	1.000000	1.869378e+00	1.131117e+01	5.940466e+00	2020.000000	11.000000	30.000000	6.000000	10.000000	1.000000	2.000000	11.933516

요약 통계량

2

분석과정 및 결과

데이터
수집

전처리

EDA

변수 선택

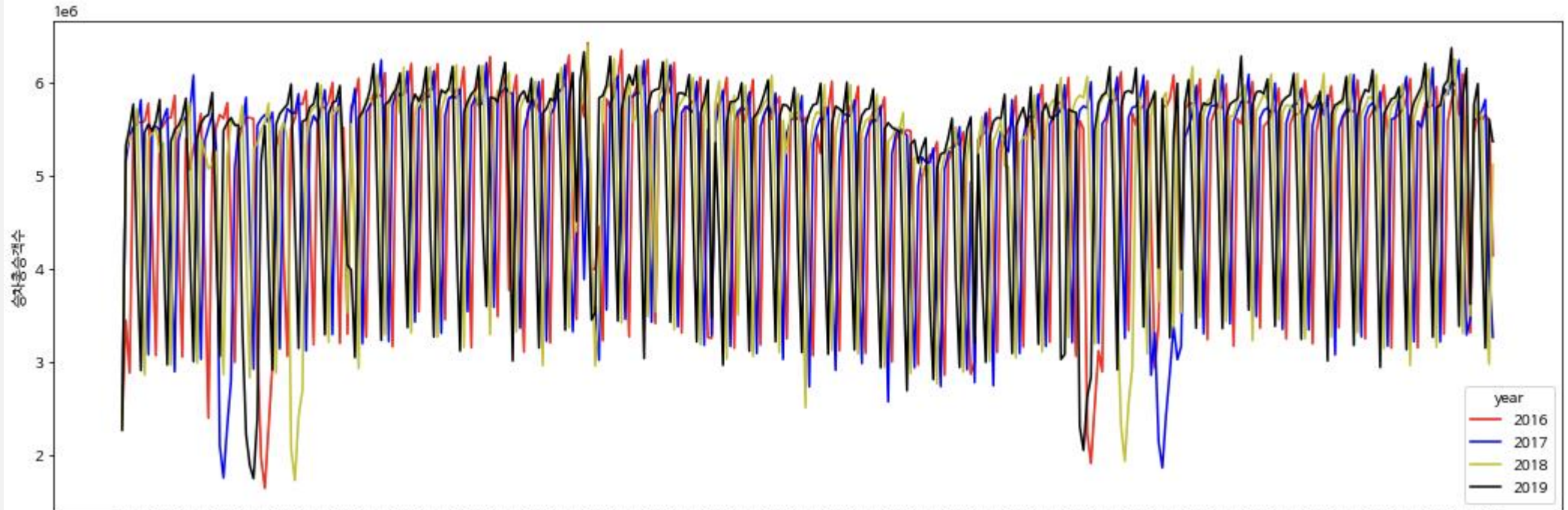
파생변수
생성

모델 선택

모델
최적화

모델 평가

2016 ~ 2019년도 데이터 승차인원의 전반적인 추세



4개 연도의 지하철 탑승객 추세는 비슷비슷함

=> 이를 바탕으로 예측 대상인 2019년도 데이터에 가까운 2018년도 데이터를 바탕으로 EDA 진행

2

분석과정 및 결과

데이터
수집

전처리

EDA

변수 선택

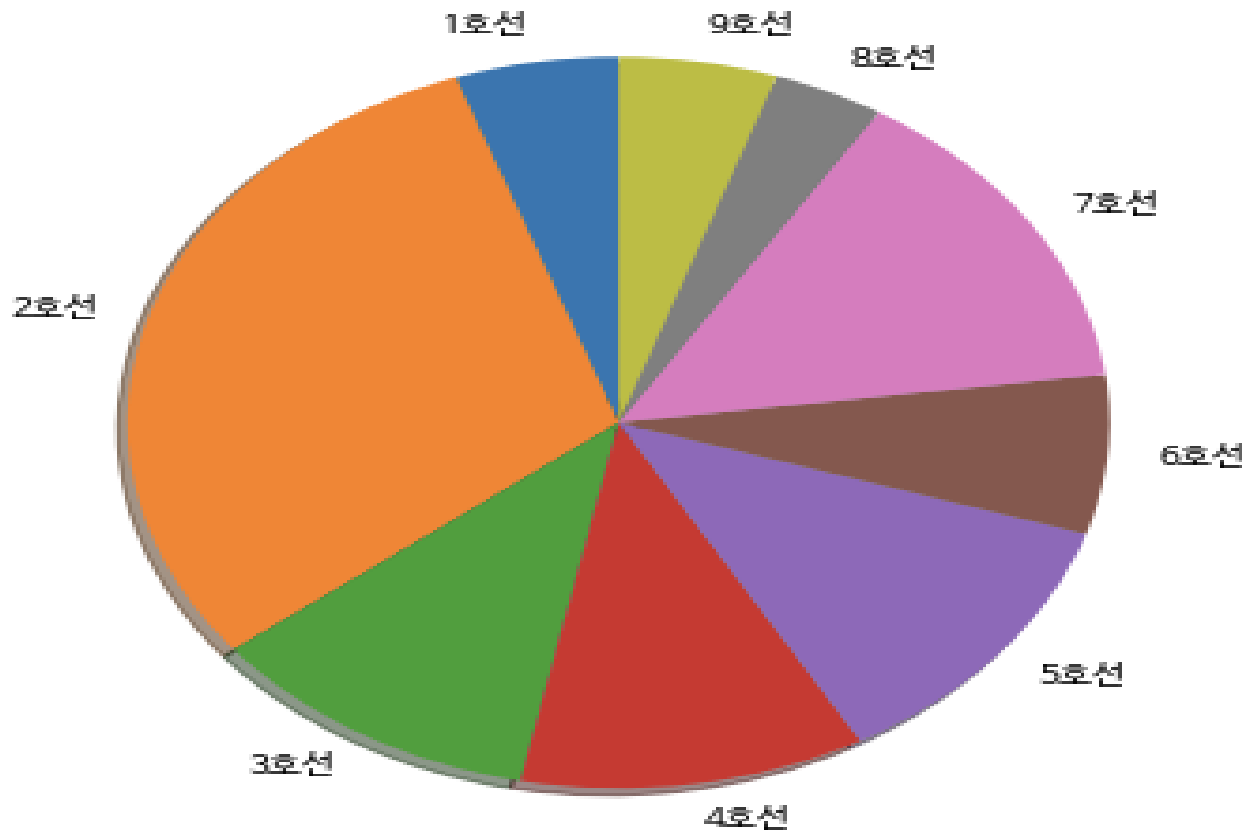
파생변수
생성

모델 선택

모델
최적화

모델 평가

호선별 탑승 인원



2호선의 탑승인원이 가장 많고 그 뒤로 7호선, 4호선이 많았음

2

분석과정 및 결과

데이터
수집

전처리

EDA

변수 선택

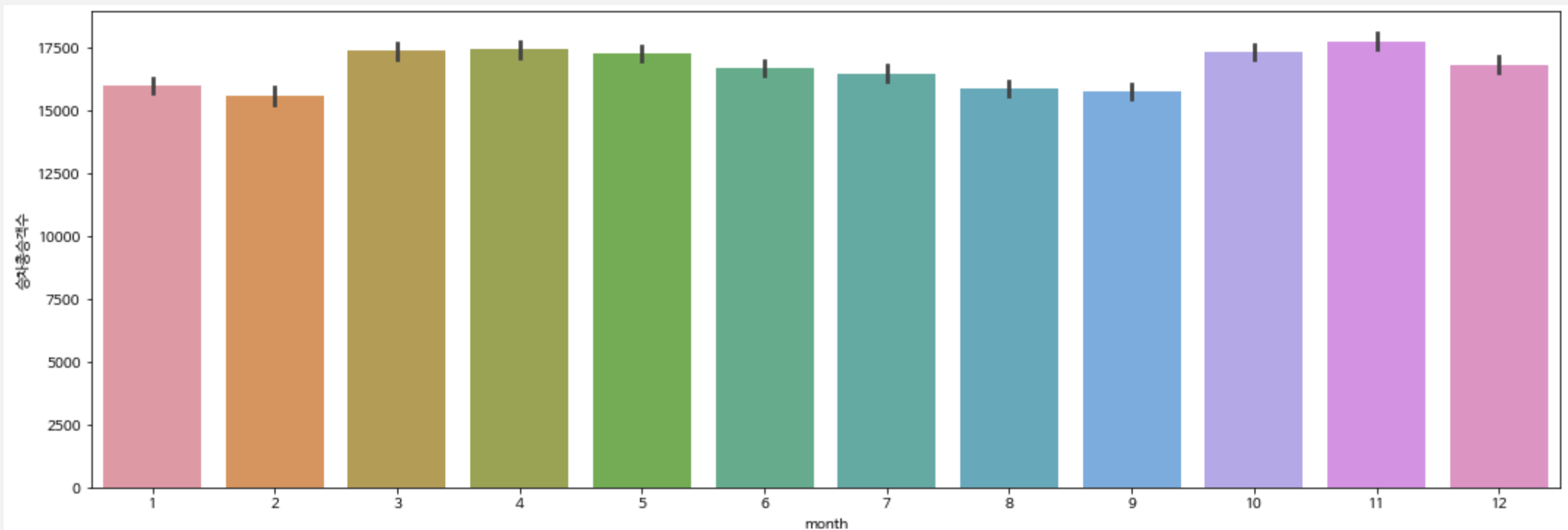
파생변수
생성

모델 선택

모델
최적화

모델 평가

월별 탑승인원



11월이 가장 많고 2월, 9월의 탑승객이 상대적으로 적음

2

분석과정 및 결과

데이터
수집

전처리

EDA

변수 선택

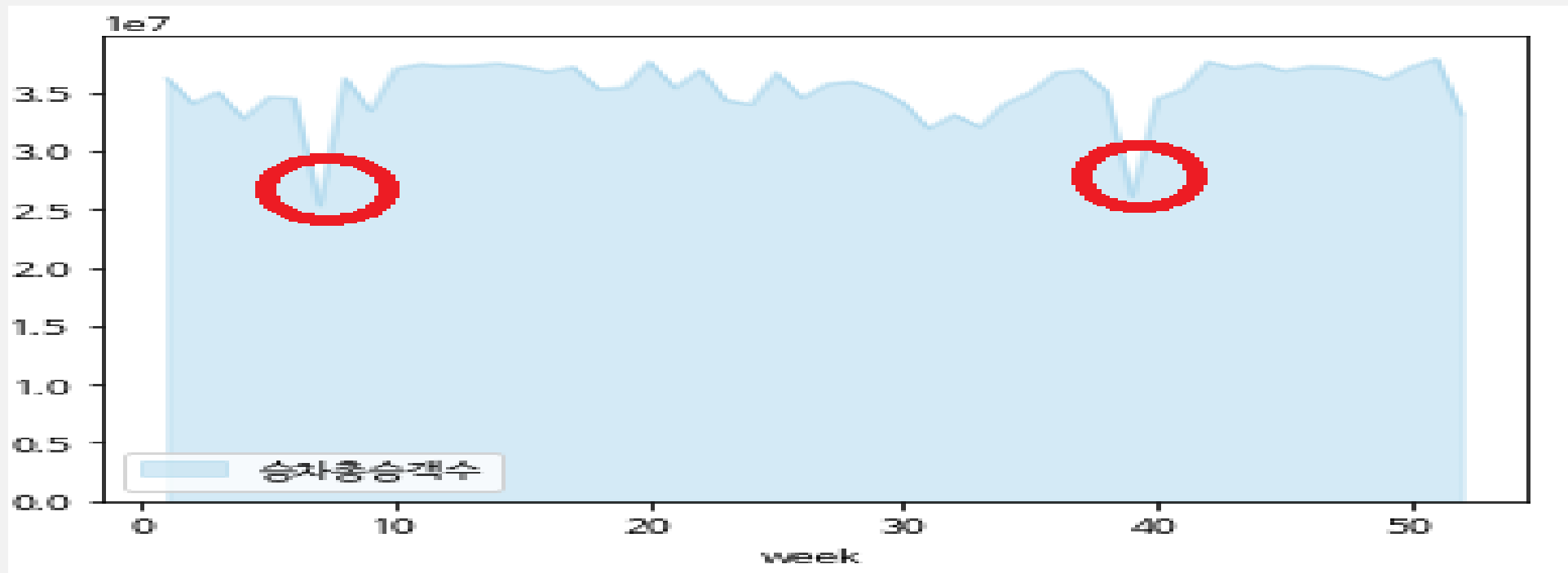
파생변수
생성

모델 선택

모델
최적화

모델 평가

주차별 탑승인원



18년도 기준 7주차, 39주차의 승차인원이 명절(설날, 추석)의 영향으로 감소함

2

분석과정 및 결과

데이터
수집

전처리

EDA

변수 선택

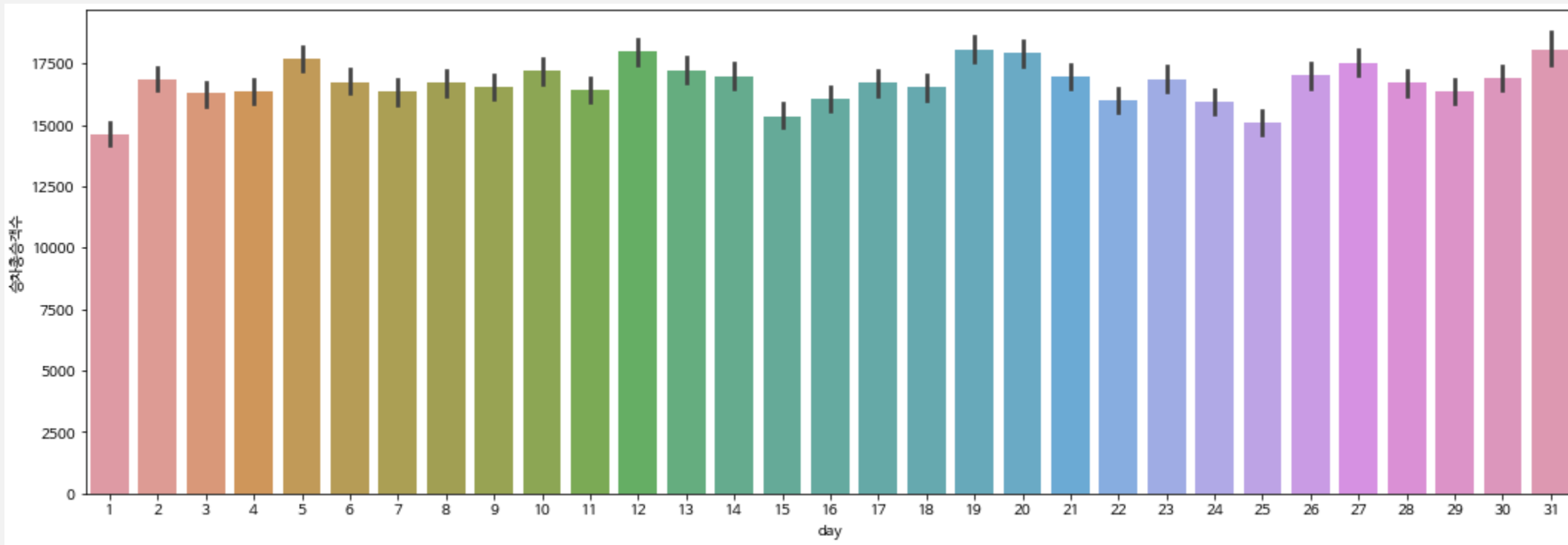
파생변수
생성

모델 선택

모델
최적화

모델 평가

일자별 탑승인원



일자별 탑승인원 확인 결과 승차인원이 주기적으로 감소했다가 증가했음을 확인

2

분석과정 및 결과

데이터
수집

전처리

EDA

변수 선택

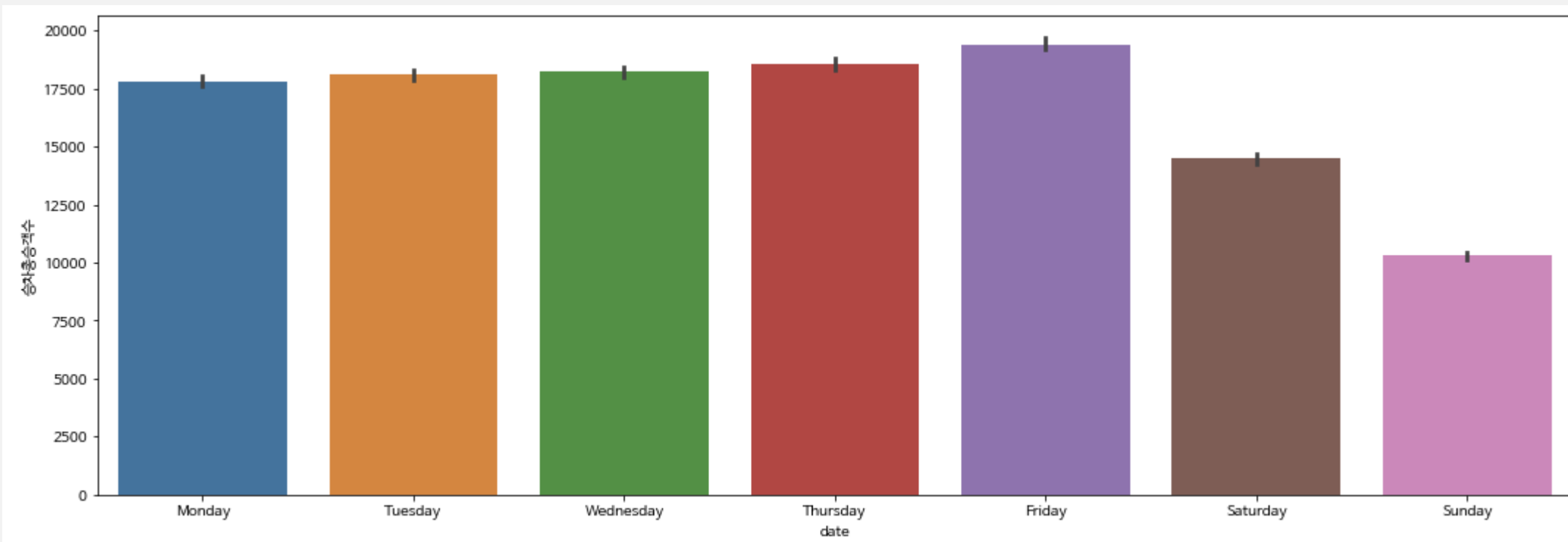
파생변수
생성

모델 선택

모델
최적화

모델 평가

요일별 탑승 인원



금요일의 탑승인원이 가장 많고 주말인 토요일, 일요일에 승차인원이 적었음

2

분석과정 및 결과

데이터
수집

전처리



EDA

변수 선택

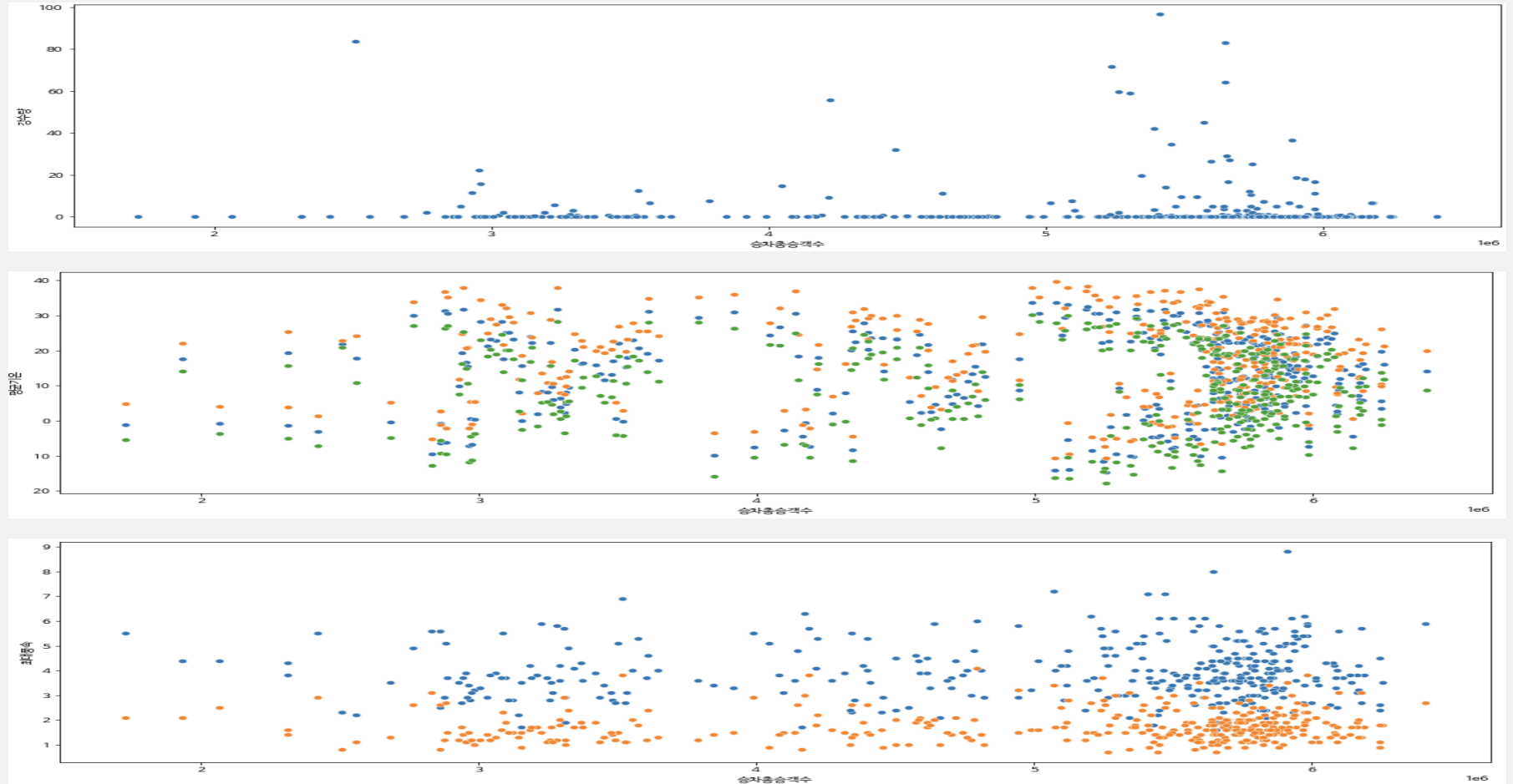
파생변수
생성

모델 선택

모델
최적화

모델 평가

기온정보(강수량, 기온, 풍속)에 따른 탑승 인원



기온 데이터와 승차인원 간의 선형관계성은 찾아보지 못하였음

2

분석과정 및 결과

데이터
수집

전처리



EDA

변수 선택

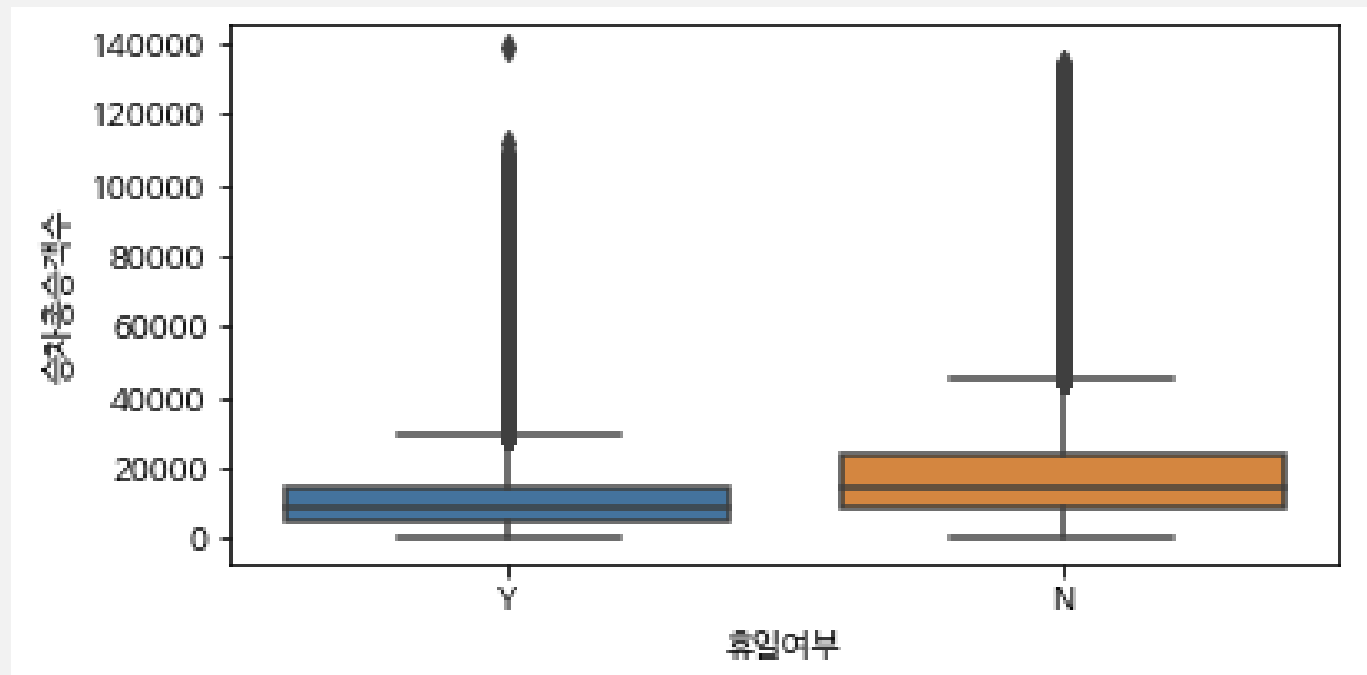
파생변수
생성

모델 선택

모델
최적화

모델 평가

휴일별 탑승인원



휴일이 아닌 날(N)에 승차인원이 휴일인 날(Y)에 비해 많았음

2

분석과정 및 결과

데이터
수집

전처리

EDA



변수 선택

파생변수
생성

모델 선택

모델
최적화

모델 평가

변수 유의성 검증

변수 year, ANOVA 검정 P-value : 0.1757
집단 간 평균의 차이 X(유의미 X)

변수 month, ANOVA 검정 P-value : 0.0000
집단 간 평균의 차이 0(유의미)

변수 day, ANOVA 검정 P-value : 0.0000
집단 간 평균의 차이 0(유의미)

변수 week, ANOVA 검정 P-value : 0.0000
집단 간 평균의 차이 0(유의미)

변수 노선명, ANOVA 검정 P-value : 0.0000
집단 간 평균의 차이 0(유의미)

변수 휴일여부, ANOVA 검정 P-value : 0.0000
집단 간 평균의 차이 0(유의미)

Scipy 패키지의 f_oneway, ttest_ind 라이브러리를 이용한
ANOVA, 사후검정, T-test로 승차인원예측을 위한
각 변수 별 유의성 확인 및 유의미한 변수 선택

=> 월, 일자, 주차, 요일, 호선, 역, 휴일 여부, 기온 데이터

2

분석과정 및 결과

데이터
수집

전처리

EDA

변수 선택

파생변수
생성

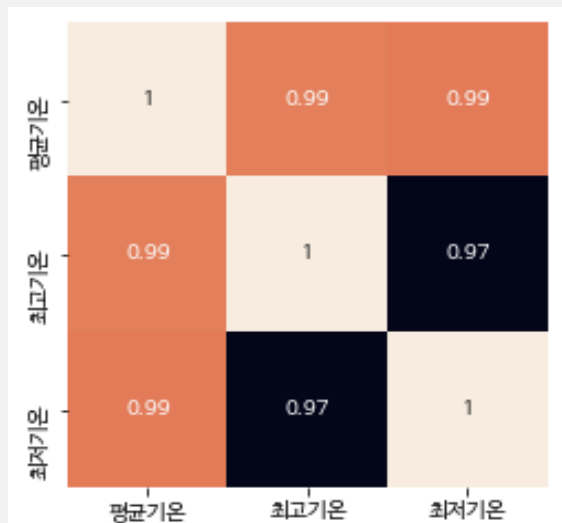
모델 선택

모델
최적화

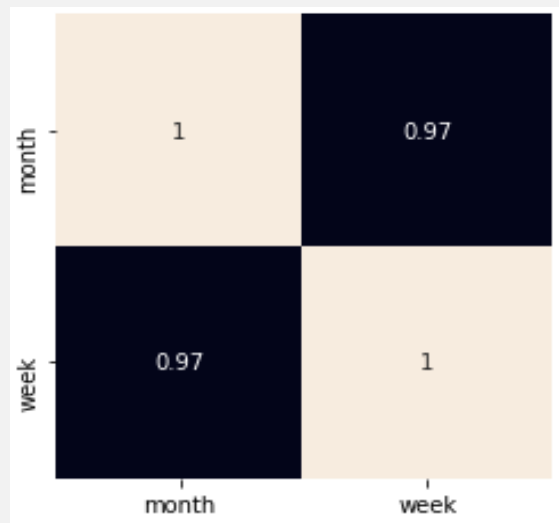
모델 평가

다중공선성(Multicollinearity)

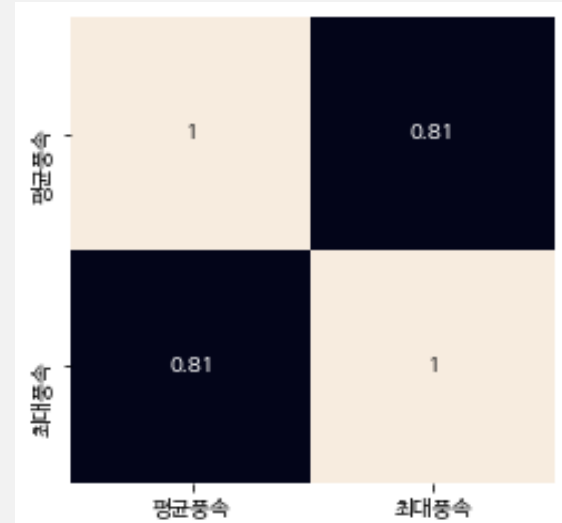
=> 독립변수들 간의 상관관계가 높아 종속변수를 예측에 부정적인 영향을 미치는 다중공선성 제거를 위해 상관관계가 높은 변수탐색



평균기온, 최고기온, 최저기온 간의
상관관계가 높음



월, 주차 간의 상관관계가 높음



평균풍속, 최대풍속 간의
상관관계가 높음

=> 상관관계가 높은 변수들 중 평균기온, 월, 최대풍속 데이터만을 분석에 활용

2


분석과정 및 결과

데이터
수집

전처리

EDA

변수 선택

 파생변수
생성

모델 선택

모델
최적화

모델 평가

파생변수 생성

=> 선택한 변수를 바탕으로 승차인원 예측에 도움을 줄 수 있는 파생변수 생성

변수 누적휴일, ANOVA 검정 P-value : 0.0000
집단 간 평균의 차이 0(유의미)

변수 명절여부, ANOVA 검정 P-value : 0.0000
집단 간 평균의 차이 0(유의미)

변수 환승노선, ANOVA 검정 P-value : 0.0000
집단 간 평균의 차이 0(유의미)

가설 1. 연속된 휴일 수가 승차인원
예측에 영향을 줄 것이다.

가설2. 명절 여부가 승차인원
예측에 영향을 줄 것이다.

가설3. 환승 노선의 수가 승차인원
예측에 영향을 줄 것이다.

=> 세 가지 가설 모두 유의함을 확인하며 새로운 파생변수 생성

2

분석과정 및 결과

데이터
수집

전처리

EDA

변수 선택

파생변수
생성

모델 선택

모델
최적화

모델 평가

최종 분석 활용 변수

노선명 역명 휴일여부 평균기온 강수량 최대풍속 month day date 누적휴일 명절여부 환승노선

=> 분석에 활용한 독립변수

승차총승객수_log

=> 분석에 활용한 종속변수

=> 기온, 일자, 노선, 역 등을 바탕으로 로그 변환한 승차총승객수 예측 후 지수화를 통한 최종 예측값 산출

2

분석과정 및 결과

데이터
수집

전처리

EDA

변수 선택

파생변수
생성

모델 선택

모델
최적화

모델 평가

모델 선택을 위한 데이터 분리 및 지표 설정

학습 / 검증데이터 분리

```
x_train, x_valid, y_train, y_valid = train_test_split(subway_train, label_train, test_size=0.2,
                                                    random_state=1122, shuffle=True)
```

예측 성능의 평가를 위해 7:3비율로 학습 데이터셋 분리 => Hold Out 검증 기법

```
print("학습 데이터 셋 크기 : {}".format(x_train.shape))
print("검증 데이터 셋 크기 : {}".format(x_valid.shape))
```

```
학습 데이터 셋 크기 : (263041, 396)
검증 데이터 셋 크기 : (65761, 396)
```

약 26만 개의 학습 데이터

약 7만 개의 검증 데이터

$$\frac{1}{n} \sum_{i=1}^n (|y_i - \hat{y}_i|)$$

모델의 평가 기준을 MAE(Mean Absolute Error)로 설정

=> 실제 승차인원과 예측 승차인원 차의 평균을 직관적으로 확인하고자 함.

2

분석과정 및 결과

데이터
수집

전처리

EDA

변수 선택

파생변수
생성

모델 선택

모델
최적화

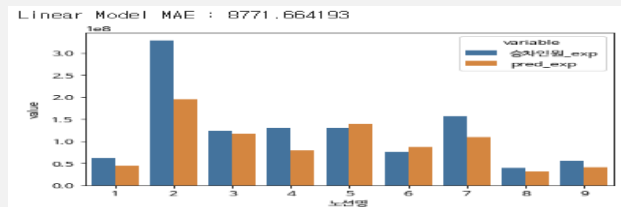
모델 평가

모델 선택(Model Selection)

=> 학습 데이터를 학습한 각 모델의 검증 데이터 예측값의 MAE 값 확인

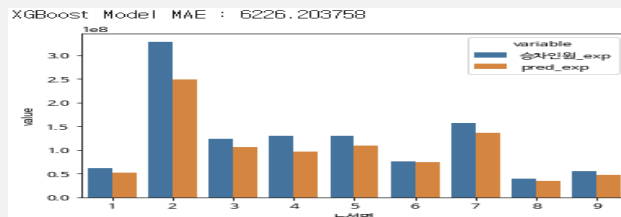
1. LinearRegression Model

MAE : 8771.66



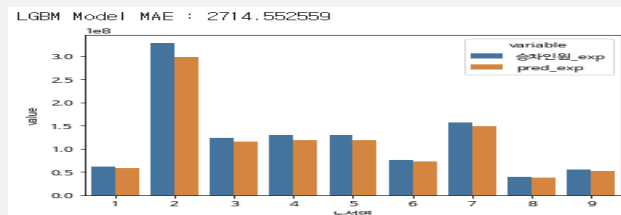
2. XGBoost Regression Model

MAE : 6226.20



3. LightGBM Regression Model

MAE : 2714.55



=> 가장 좋은 예측 성능을 보여준 LGBMRegressor Model을 최종 예측모델로 선택

2

분석과정 및 결과

데이터
수집

전처리

EDA

변수 선택

파생변수
생성

모델 선택

모델
최적화

모델 평가

하이퍼 파라미터(Hyper Parameter) 튜닝

```

from sklearn.model_selection import GridSearchCV

params = {'learning_rate': [0.001, 0.03, 0.01, 0.05, 0.1, 0.2],
          'max_depth': [3, 5, 7, 10]}

n_splits = 5

kf = KFold(n_splits = n_splits, shuffle=True, random_state=1110)

lgbm_grid = GridSearchCV(LGBMRegressor(tree_method='gpu_hist'), params, n_jobs=-1, scoring='neg_mean_absolute_error', cv=kf)

lgbm_grid.fit(subway_train, label_train)

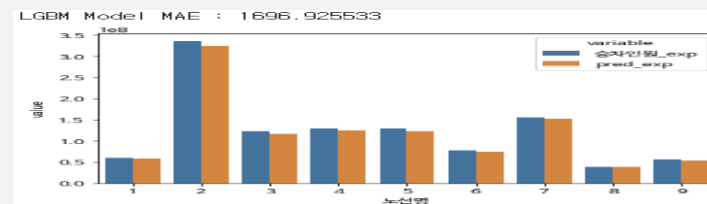
Grid_cv = pd.DataFrame(lgbm_grid.best_params_, index=[0])

```

⇒ 선택한 LGBM Model을 최적화 시키기 위해
GridSearchCV 구현 및 파라미터 튜닝

주요 파라미터 중 learning_rate, max_depth
두 개의 주요 파라미터 튜닝

learning_rate	max_depth
0.2	10



⇒ GridSearchCV를 이용한 learning_rate, max_depth 파라미터 튜닝 결과 각각 0.2, 10일때 가장 좋은 성능을 나타내며 Hold-out 검증을 이용하여 확인한 모델의 CV(Cross Validation) MAE값은 1696.96

2

분석과정 및 결과

데이터
수집

전처리

EDA

변수 선택

파생변수
생성

모델 선택

모델
최적화

모델 평가

과적합(Over fitting) 방지

1. 교차 검증(Cross Validation)

⇒ K-Folds 교차 검증을 통한 과적합 방지

2. Early Stopping Rounds 조절

⇒ 일정 수준 이상 성능이 증가되지 않을 때 학습 중지

3. Out-of-fold(OOF) Ensemble

⇒ 각 Fold에 대한 예측 값을 앙상블하는 기법

```

val_scores = []
oof_pred = np.zeros((subway_test.shape[0], ))

for i, (trn_idx, val_idx) in enumerate(kf.split(subway_train, label_train)):
    x_train, x_valid = subway_train.iloc[trn_idx], subway_train.iloc[val_idx]
    y_train, y_valid = label_train.iloc[trn_idx], label_train[val_idx]

    model = LGBMRegressor(n_estimators=20000, learning_rate=0.2, max_depth=10, verbose=1000, tree_method='gpu_hist', random_state=10)
    evals = [(x_train, y_train), (x_valid, y_valid)]

    model.fit(x_train, y_train, eval_metric='mae', eval_set = evals, early_stopping_rounds=500, verbose=500)
    pred = model.predict(x_valid)

    df.append(valid)
    oof_pred += model.predict(subway_test) / n_splits
    val_mae = mean_absolute_error(y_valid, pred)

    val_scores.append(val_mae)

print(f"{i+1} folder validation score : {val_mae:.4f}\n\n")

```

2

분석과정 및 결과

데이터
수집

전처리

EDA

변수 선택

파생변수
생성

모델 선택

모델
최적화

모델 평가

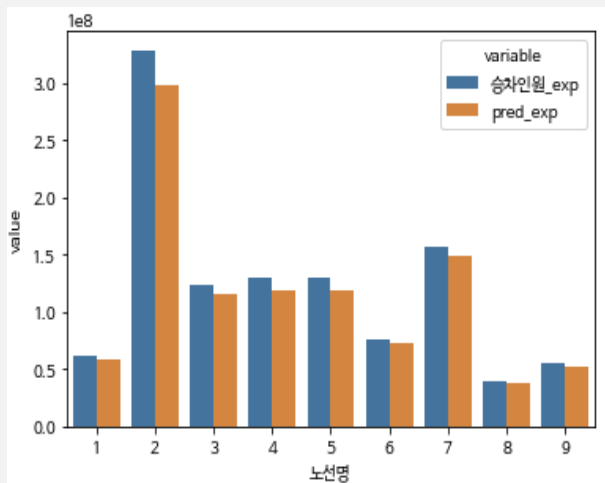
모델 평가(Model Evaluation)

2019년도 승차인원 예측 MAE : 1111.7672

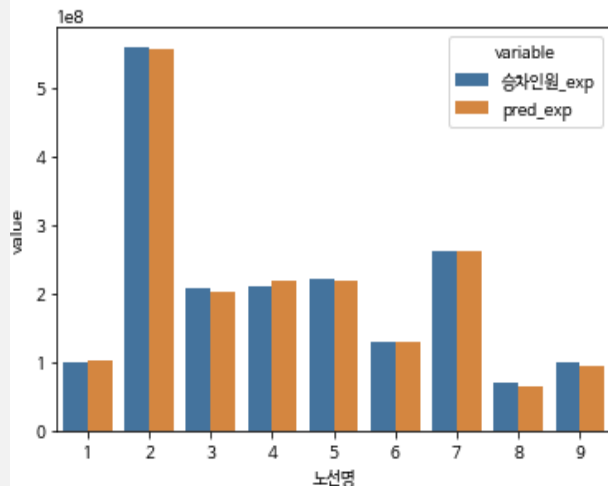
최종 예측데이터(2019년도 승차인원)에 대한
최종 MAE값은 1111.7672임을 확인

=> 실제 승차인원과 약 1100명정도 차이나는 값으로 예측

〈 모델 최적화 전 〉



〈 모델 최적화 후 〉

예측 성능 시각화 결과 최적화 후의 모델이
승차 인원을 더 잘 예측하는 모습

2

분석과정 및 결과

데이터
수집

전처리

EDA

변수 선택

파생변수
생성

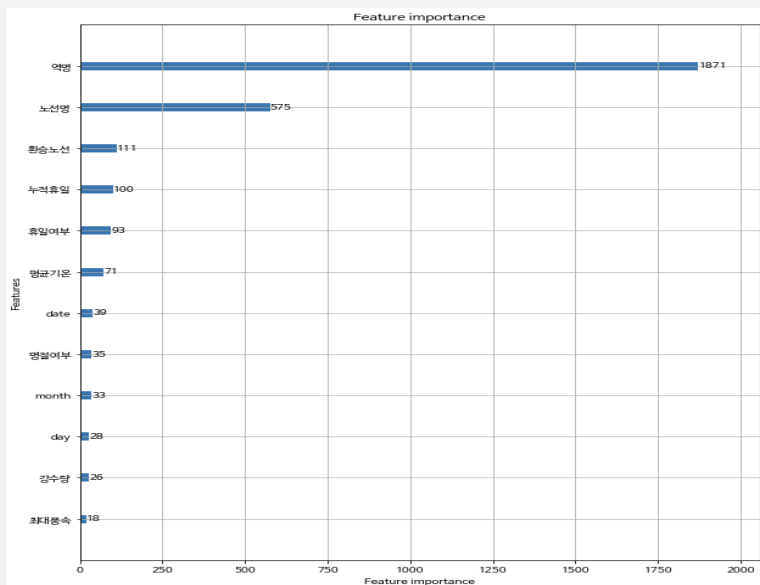
모델 선택

모델
최적화

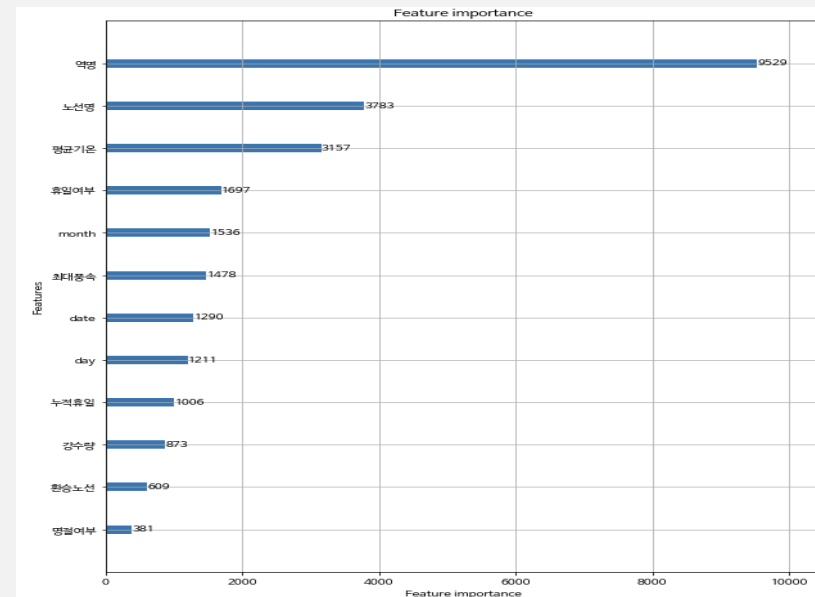
모델 평가

변수 중요도(Feature importance) 변화 확인

〈 파라미터 튜닝 전 〉



〈 파라미터 튜닝 후 〉



=> 이를 통하여 지하철 승차인원을 예측함에 있어 단순히 역, 노선 등의 정보 뿐만 아니라 기온, 휴일여부, 최대 풍속, 월, 일 데이터 등의 다양한 변수 조합을 이용하여 예측 결과를 산출해 냄을 확인

3

활용방안 및 기대효과

“ 정확한 데이터(기온, 강수량, 풍속)를 기반으로 한 승차인원 예측 ”

승차인원	
실제	106582.00
예측	107510.36

⇒ 특정 일자의 실제 기상정보를 바탕으로 잠실역 승차인원 예측 결과 실제 승차인원과 약 900명 정도의 차이를 보이는 인원 수 예측

부정확한 기상 정보

승차인원	
실제	106582.00
예측	105360.97

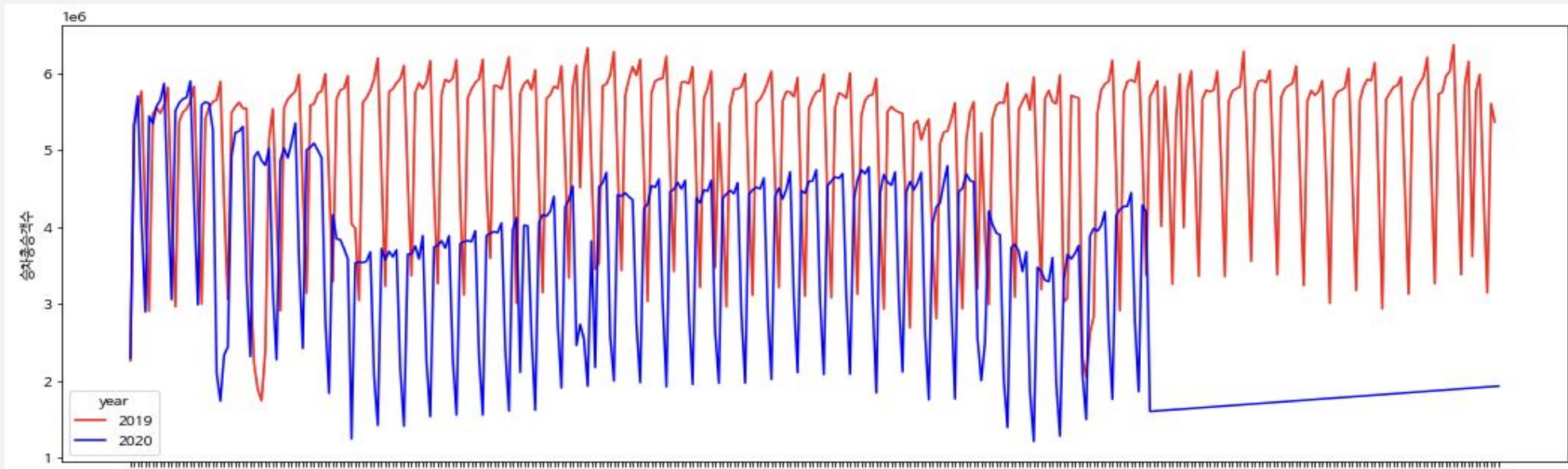
⇒ 특정 일자의 부정확한 기상 정보를 바탕으로 잠실역 승차인원 예측 결과 실제 승차인원과 약 1200명 정도의 차이를 보이는 인원 수 예측

=> 정확한 기상 예측 데이터와 함께 승차인원을 예측함으로써 지하철 탑승 수요 파악 가능

3

활용방안 및 기대효과

“ 변수 추가로 인한 승차인원 변화에 대응 ”

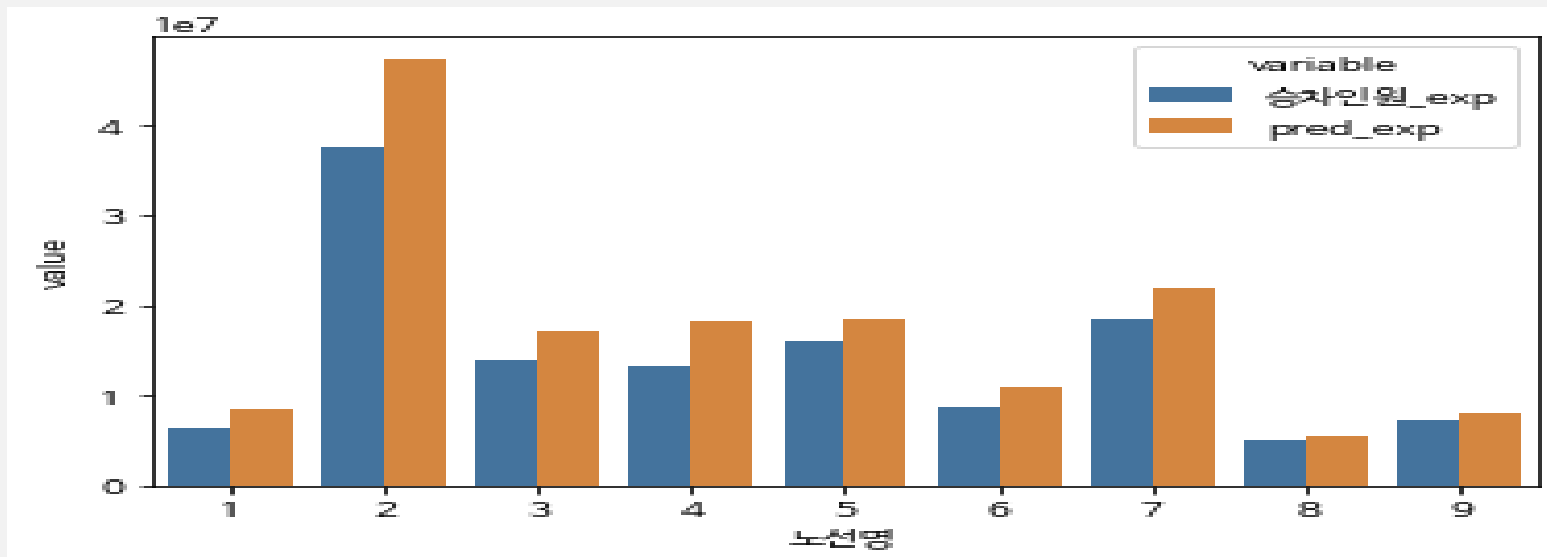


코로나19로 인해 2019년도의 기존 승차인원에 비해 2020년의 승차인원이 확 줄어든 모습

3

활용방안 및 기대효과

“ 변수 추가로 인한 승차인원 변화에 대응 ”



2020년도 9월 승차인원 예측 MAE : 3448.9860

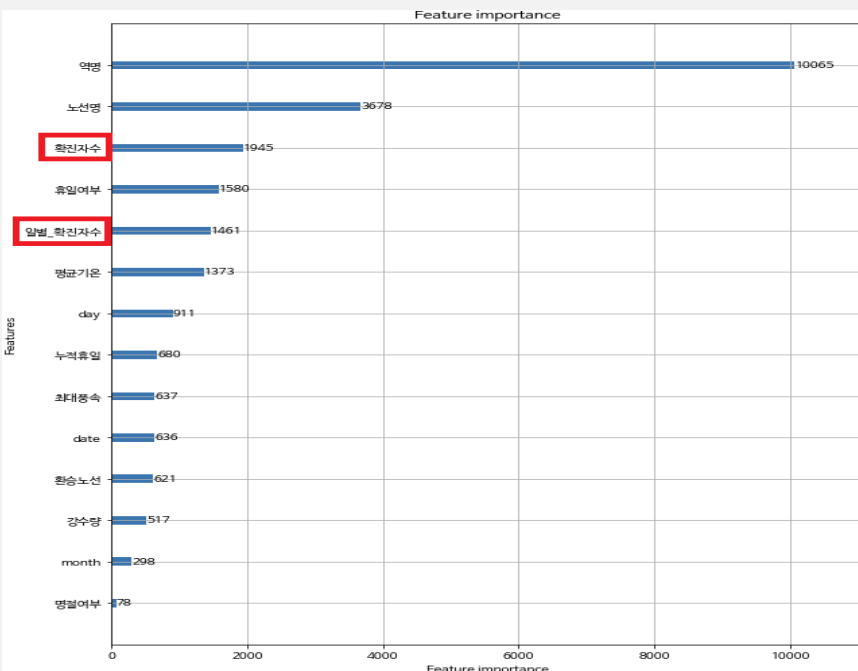
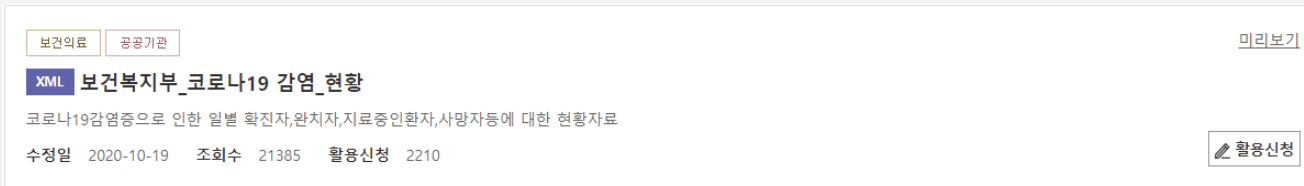
2016 ~ 2018년도의 데이터로 설계한 기존의 모델로 2020년도 9월의 승차 인원을 예측한 결과 MAE값은 3448.99, 기존 모델의 MAE가 1111.72임을 감안할때 엄청난 성능 감소

시각화 결과 코로나19로 인한 승차객 감소 현상을 모델이 반영하지 못함

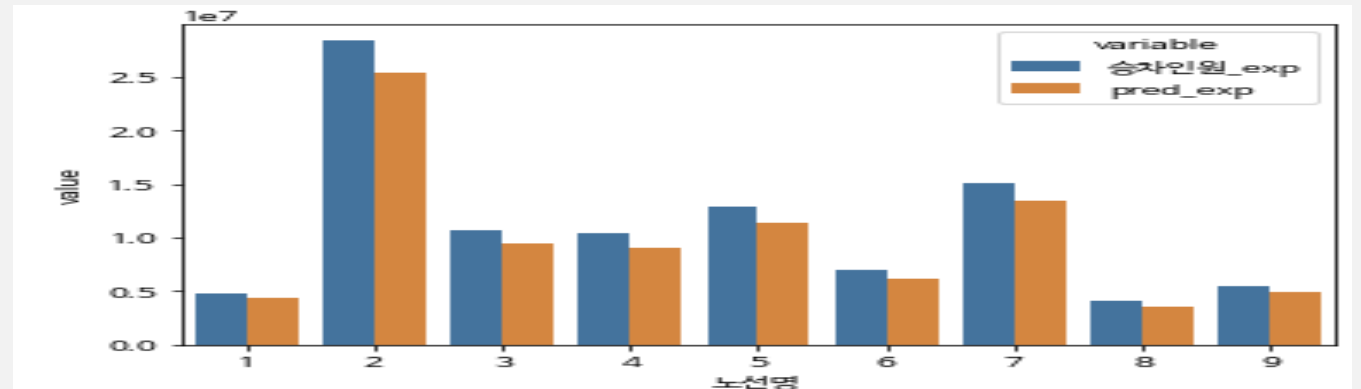
3

활용방안 및 기대효과

“ 변수 추가로 인한 승차인원 변화에 대응 ”



공공데이터 활용을 통한 누적확진자 수,
일자별 확진자 수 변수 추가



2020년도 9월 승차인원 예측 (With Covid Data) MAE : 1455.7566

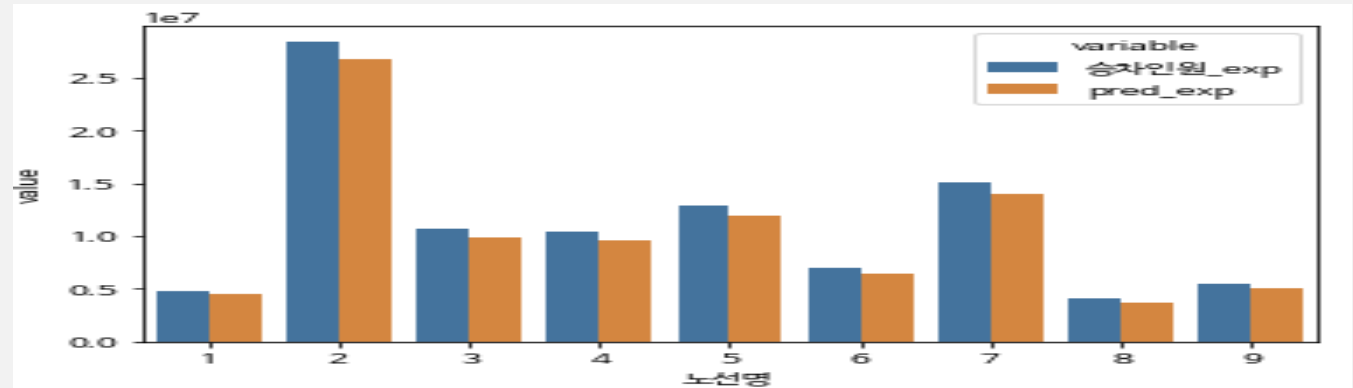
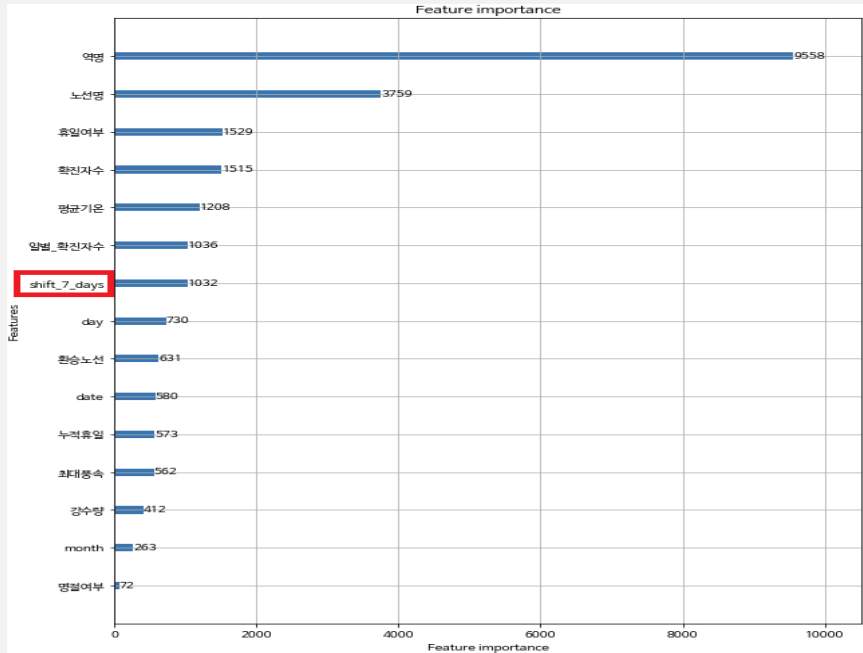
실제로 높은 변수 중요도를 보여주며
예측 성능을 향상시킴

3

활용방안 및 기대효과

“ 변수 추가로 인한 승차인원 변화에 대응 ”

바이러스 잠복기(평균 4~7일)을 고려한 일주일 전 확진자 수 변수 추가



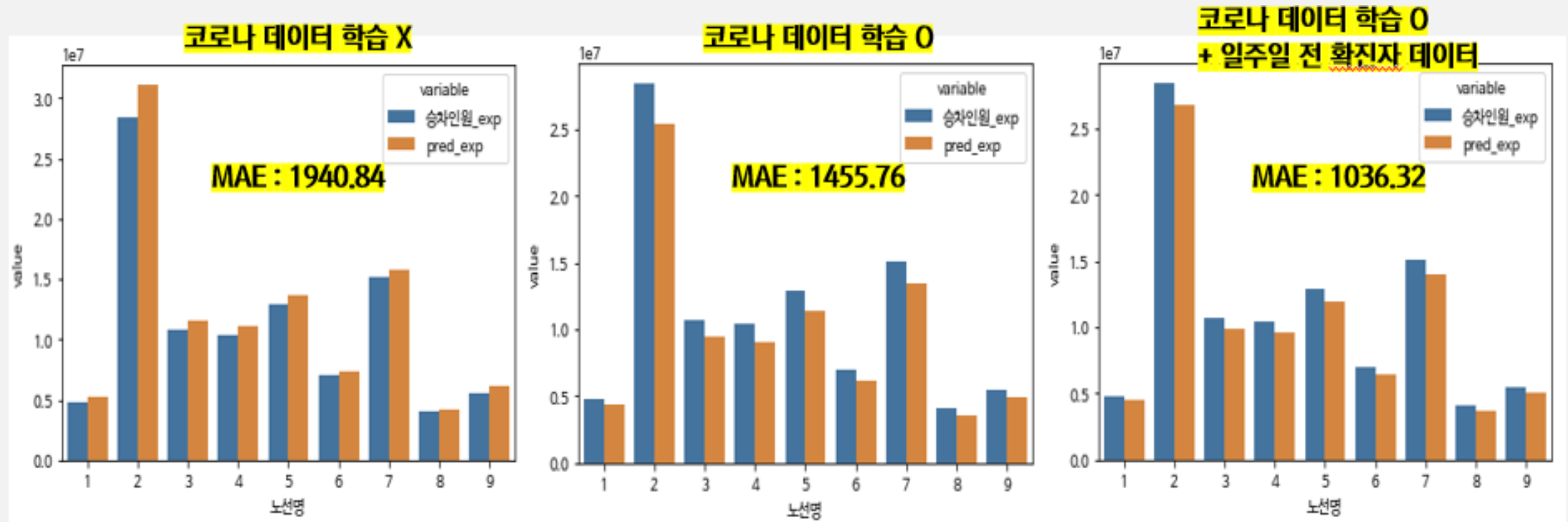
2020년도 9월 승차인원 예측 (With Covid Data + 일주일 전 확진자 데이터) MAE : 1036.3153

마찬가지로 한번 더 코로나19이후 승차인원에 예측 성능
향상에 기여함을 확인

3

활용방안 및 기대효과

“ 변수 추가로 인한 승차인원 변화에 대응 ”



=> 기존 설계 모델에 변수를 추가함으로써 코로나19이후 지하철 승차인원에 영향을 주는 요인을 확인하며 승차인원 예측 성능을 향상시킬 수 있음

“

지하철 승차에 영향을 주는 요인 탐색.

**승차인원 예측에 따른 효율적인 운행 인력 운용 및
유동적인 운행 증 / 감축.**

지하철 이용 불편 감소로 인한 대중교통 이용 장려.

”