머신러닝 프로젝트

ML_PJT 1조

김정빈 박건우 윤정옥

목차

주제

데이터 소개

주제선정이유

세부진행방안

가치제공

2030세대의 주문건수 예측

2030세대의 인구수와 주문건수 상관관계 예측
=> 효율적인 경영전략을 제공

데이터 소개

'delivery' 데이터

	광역시 도명	시군 구명	날짜	시간대별 시간	강수 유형명	습도 값	강수량 값	기온 값	풍속 값	바람강도 유형명	...	회_배달 건수	치킨_배 달건수	피자_배 달건수	아시안/양식_ 배달건수	중식_배 달건수	족발/보쌈_배 달건수	야식_배 달건수	찜탕_배 달건수	도시락_배 달건수	패스트푸드_ 배달건수
0	강원도	강릉 시	2019-08-02	19	없음	83	0.0	25.4	0.3	약	...	0	0	0	0	0	0	0	0	0	0
1	강원도	강릉 시	2019-08-03	15	없음	83	0.0	25.7	0.6	약	...	0	0	0	0	0	0	0	0	0	0
2	강원도	강릉 시	2019-08-03	16	없음	83	0.0	25.4	0.2	약	...	0	0	0	0	0	0	0	0	0	0
3	강원도	강릉 시	2019-08-04	11	없음	86	0.0	27.3	1.2	약	...	0	0	0	0	0	0	0	0	0	0
4	강원도	강릉 시	2019-08-04	14	없음	89	0.0	26.2	1.1	약	...	0	0	0	0	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
34626	충청북도	충주 시	2020-07-31	19	없음	98	0.0	23.5	0.6	약	...	1	18	0	0	5	0	0	0	0	0
34627	충청북도	충주 시	2020-07-31	20	없음	97	0.0	23.6	0.3	약	...	0	13	0	0	3	0	0	0	0	0
34628	충청북도	충주 시	2020-07-31	21	없음	94	0.0	24.6	0.5	약	...	0	16	0	0	2	0	0	0	0	0
34629	충청북도	충주 시	2020-07-31	22	없음	93	0.0	25.5	1.0	약	...	0	10	0	0	0	0	0	0	0	0
34630	충청북도	충주 시	2020-07-31	23	없음	84	0.0	26.7	0.7	약	...	0	6	0	0	0	0	0	0	0	0

380741 rows × 27 columns

자료출처 : KT 빅데이터 플랫폼

데이터 소개

'population' 데이터

	행정구역(동읍면)별(1)	행정구역(동읍면)별(2)	항목	계	20 - 24세	25 - 29세	30 - 34세	35 - 39세	65 - 69세	70 - 74세	75 - 79세	80 - 84세	85 - 89세	90 - 94세	95 - 99세	100+
0	전국	소계	총인구수 (명)	51696216.0	3541061.0	3217367.0	3517868	4016272.0	2237345.0	1781229.0	1457890	909130.0	416164.0	141488.0	34844	17562.0
1	전국	소계	남자인구수 (명)	25827594.0	1877127.0	1682988.0	1806754	2045265.0	1072395.0	806680.0	600607	319391.0	113221.0	32695.0	7658	4137.0
2	전국	소계	여자인구수 (명)	25868622.0	1663934.0	1534379.0	1711114	1971007.0	1164950.0	974549.0	857283	589739.0	302943.0	108793.0	27186	13425.0
3	서울특별시	소계	총인구수 (명)	9930616.0	690728.0	751973.0	803507	817467.0	448956.0	350580.0	251961	141649.0	66067.0	24153.0	7058	5475.0
4	서울특별시	소계	남자인구수 (명)	4876789.0	347534.0	372249.0	402358	410076.0	211568.0	163766.0	112076	54033.0	19595.0	6146.0	1900	1406.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
841	제주특별자치도	제주시	남자인구수 (명)	235977.0	17377.0	13118.0	15084	18350.0	8474.0	6782.0	4941	2737.0	854.0	226.0	53	17.0
842	제주특별자치도	제주시	여자인구수 (명)	234688.0	15261.0	12245.0	14687	18062.0	9265.0	7877.0	7178	5649.0	3122.0	1387.0	460	137.0
843	제주특별자치도	서귀포시	총인구수 (명)	170932.0	10505.0	8067.0	9120	11606.0	8686.0	7460.0	6456	4521.0	1855.0	733.0	242	77.0
844	제주특별자치도	서귀포시	남자인구수 (명)	86568.0	5600.0	4247.0	4693	6082.0	4237.0	3441.0	2611	1494.0	370.0	103.0	29	9.0
845	제주특별자치도	서귀포시	여자인구수 (명)	84364.0	4905.0	3820.0	4427	5524.0	4449.0	4019.0	3845	3027.0	1485.0	630.0	213	68.0

846 rows × 16 columns

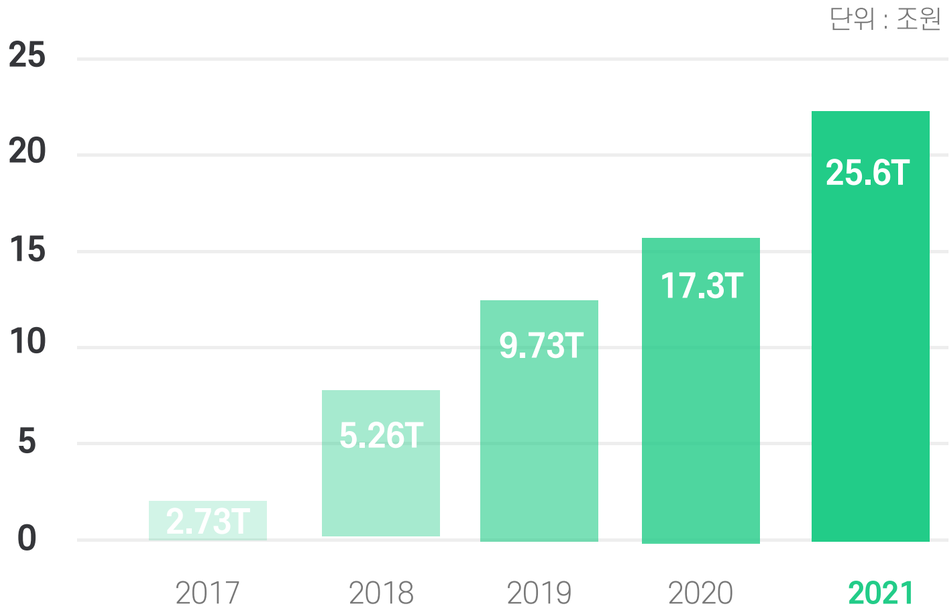
자료출처 : 제로베이스

온라인음식배달업 현황

국내 상황

- 매년 75% 성장
- 최근 2년간 연평균 성장률 163.9%
- 배달업 종사자 42만 3천명

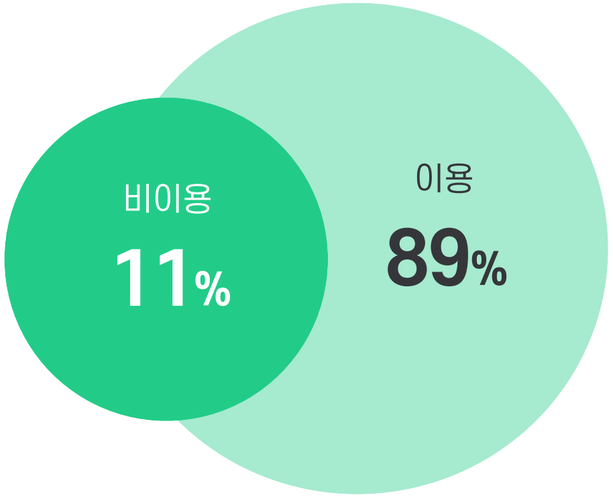
연도별 거래액



자료출처 : 통계청

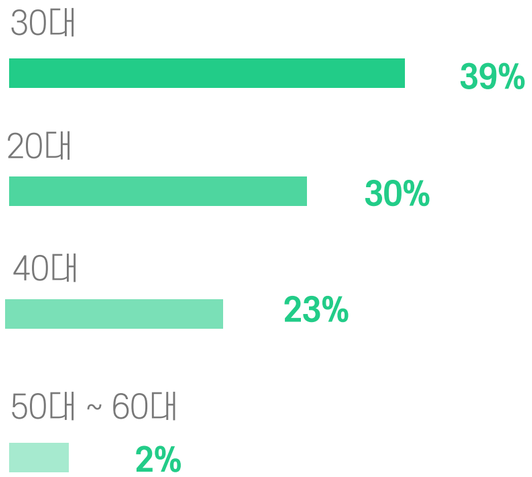
Q. 1개월 내 배달음식 주문경험

마케팅 • 빅데이터 분석 전문기관 나이스(NICE)는 온라인(모바일) 설문조사 패널 플랫폼 N플러스패널에서 자체 수행한 소비자 조사의 분석 결과를 발표했다.



자료출처 : 나이스(NICE)

연령대별 결제금액 비율



자료출처 : 하나은행
2020.01.01 ~ 2020.12.31

설문조사기관
나이스(NICE), 하나은행

설문대상
전국, 만 20세 이상 소비자

조사결과 분석
나이스(NICE), 하나은행

배달업계 성장부터 가치제공까지

기존의 배달업계 연구와 달리 연령대 요소로 인해 선택한 업종에 관해
심층적 분석을 진행 및 연령대 요소 중 2030세대에 집중해 실제로
연령과 음식 카테고리의 상관관계 파악

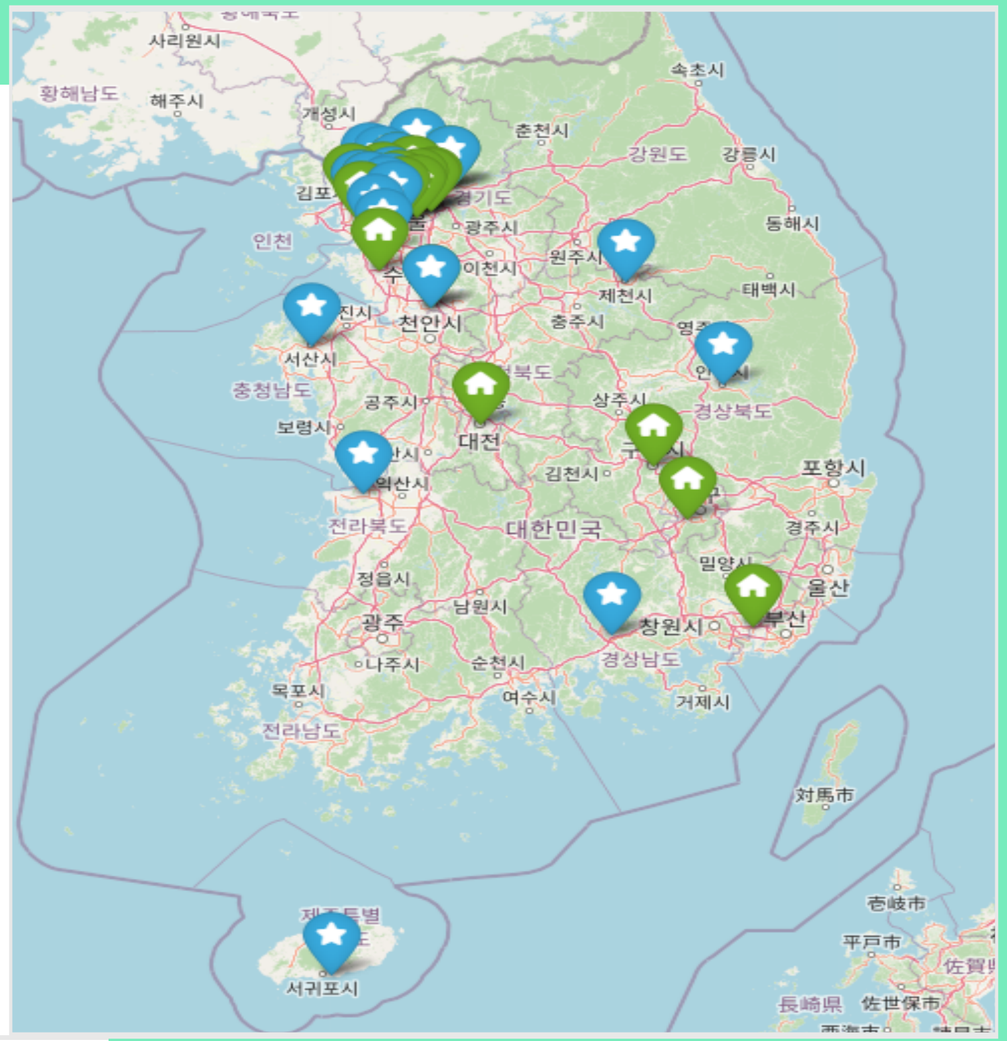
#1-1
데이터 merge

- delivery 데이터 '배달건수_합계' 컬럼추가
- population 데이터 연령대별 분류
- key(id)를 생성하여 두 데이터 merge

```
0   광역시도명          203 non-null    object
1   시군구명           203 non-null    object
2   도시락_배달건수      203 non-null    int64
3   돈까스/일식_배달건수  203 non-null    int64
4   배달건수_합계       203 non-null    int64
5   분식_배달건수       203 non-null    int64
6   아시안/양식_배달건수  203 non-null    int64
7   야식_배달건수       203 non-null    int64
8   족발/보쌈_배달건수    203 non-null    int64
9   중식_배달건수       203 non-null    int64
10  찜탕_배달건수       203 non-null    int64
11  치킨_배달건수       203 non-null    int64
12  카페/디저트_배달건수  203 non-null    int64
13  패스트푸드_배달건수   203 non-null    int64
14  피자_배달건수       203 non-null    int64
15  한식_배달건수       203 non-null    int64
16  회_배달건수         203 non-null    int64
17  ID              203 non-null    object
18  20 - 39세남자     202 non-null    float64
19  20 - 39세여자     202 non-null    float64
20  20 - 39세합계     202 non-null    float64
21  65세이상남자        202 non-null    float64
22  65세이상여자        202 non-null    float64
23  65세이상합계        202 non-null    float64
24  인구수합계          202 non-null    float64
25  2030비율        202 non-null    float64
26  고령층비율          202 non-null    float64
```

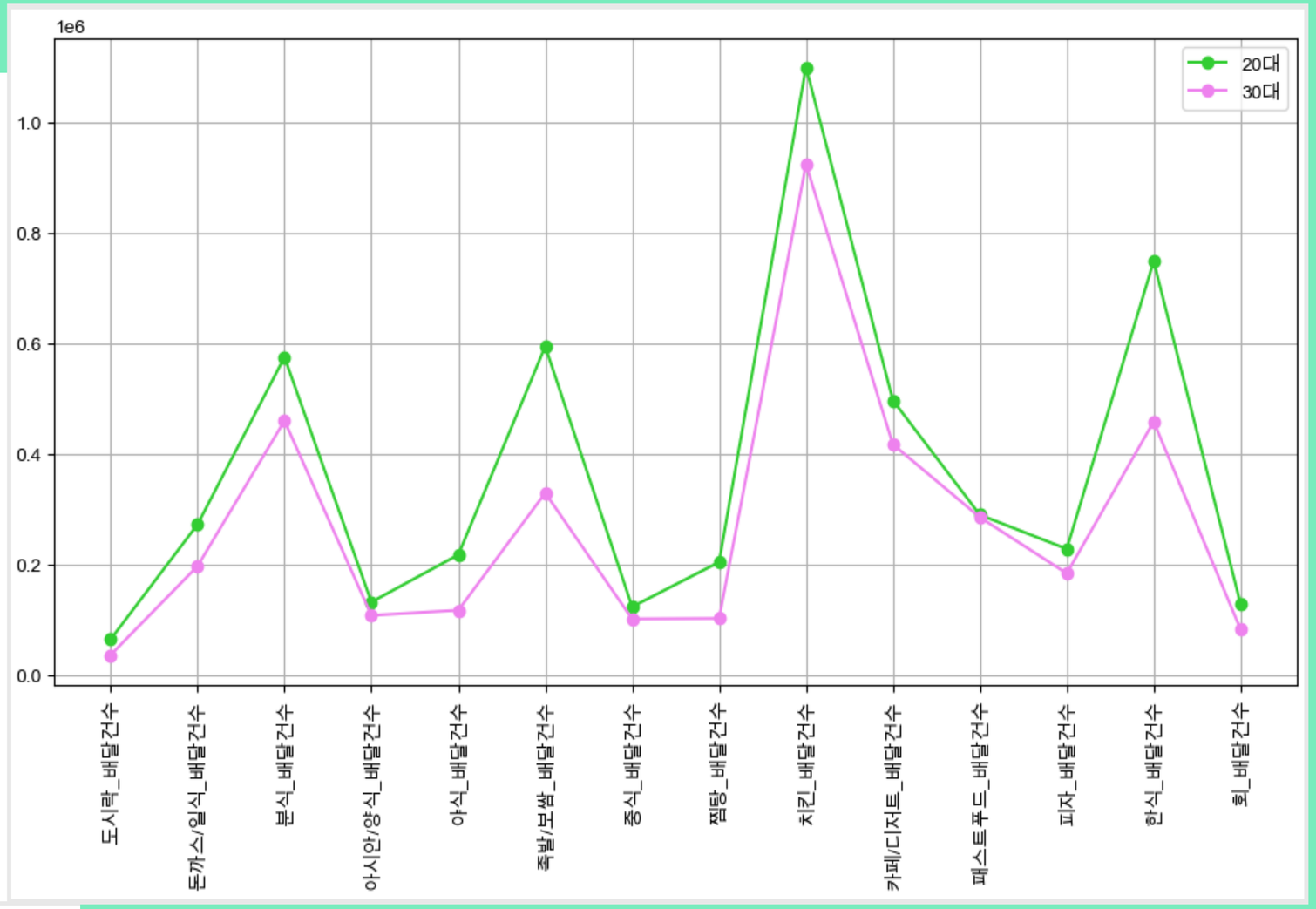
#1-2
분포도 확인

- 2030비율 상위 20개 지역
- 배달건수 상위 20개 지역



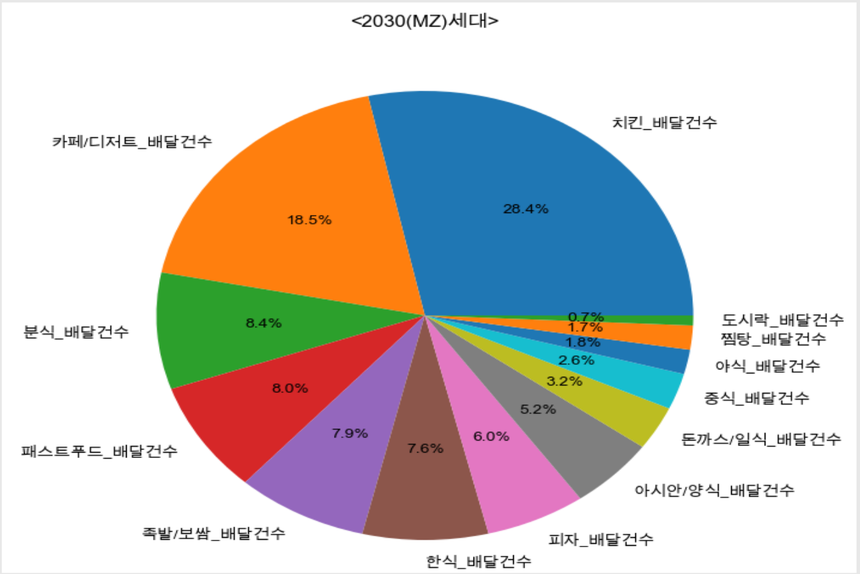
#2-1
20대, 30대 주문 건수

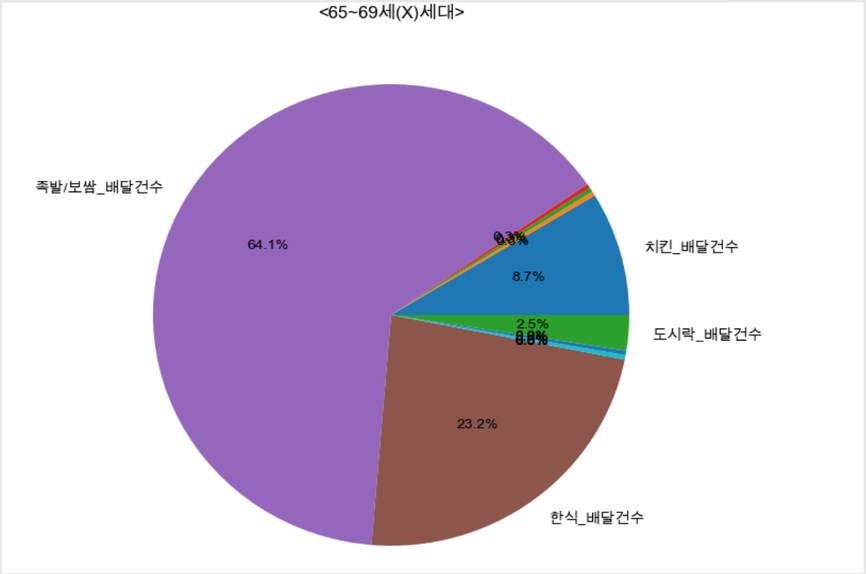
- 배달카테고리별 20대와 30대 주문 건수
- 전반적으로 20대가 30대보다 많이 주문

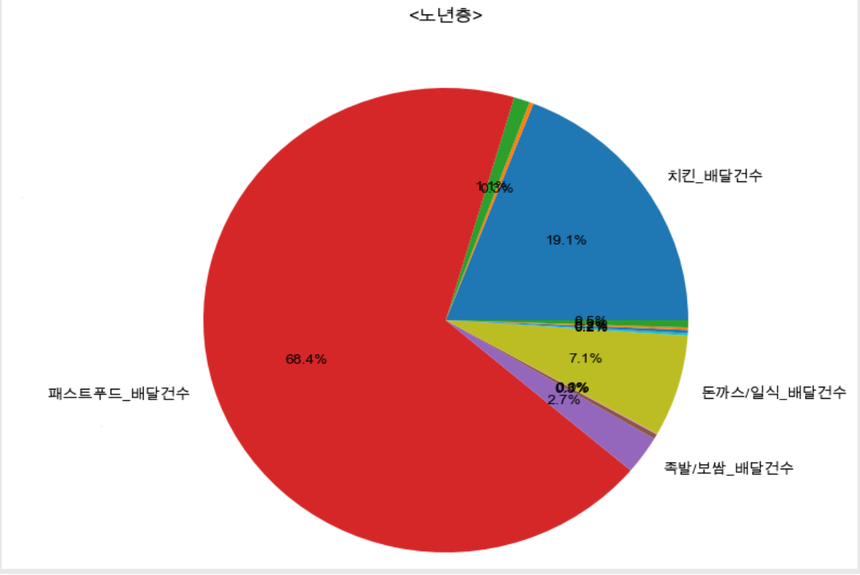


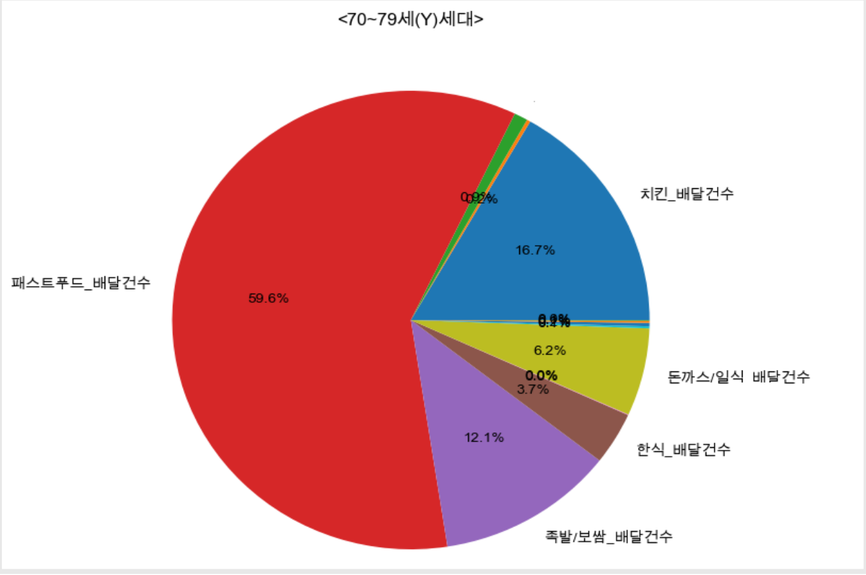
#2-2
세대별 주문 건수

- MZ세대 : 1981년 ~ 1996년생
- X세대 : 1960년대 ~ 1970년대생
- Y세대 : 1946년 ~ 1965년생
- 노년층 : 1946년 이전생



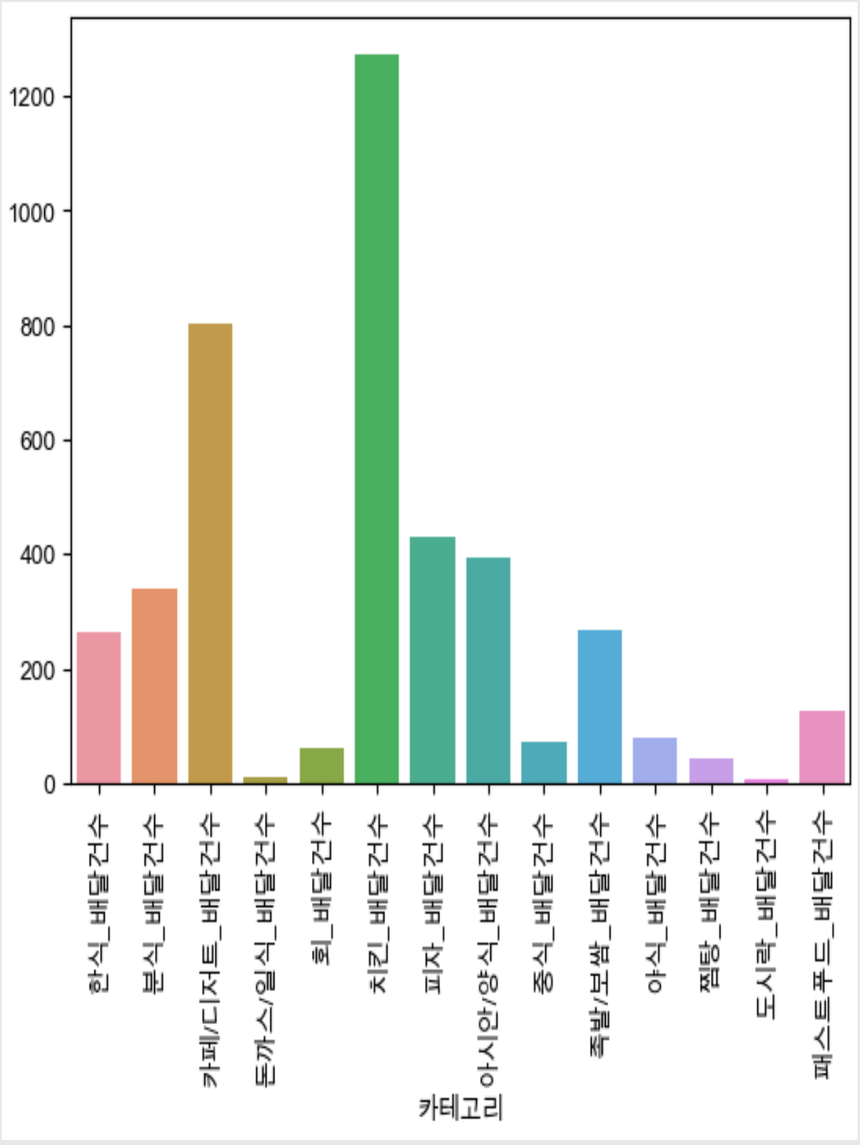


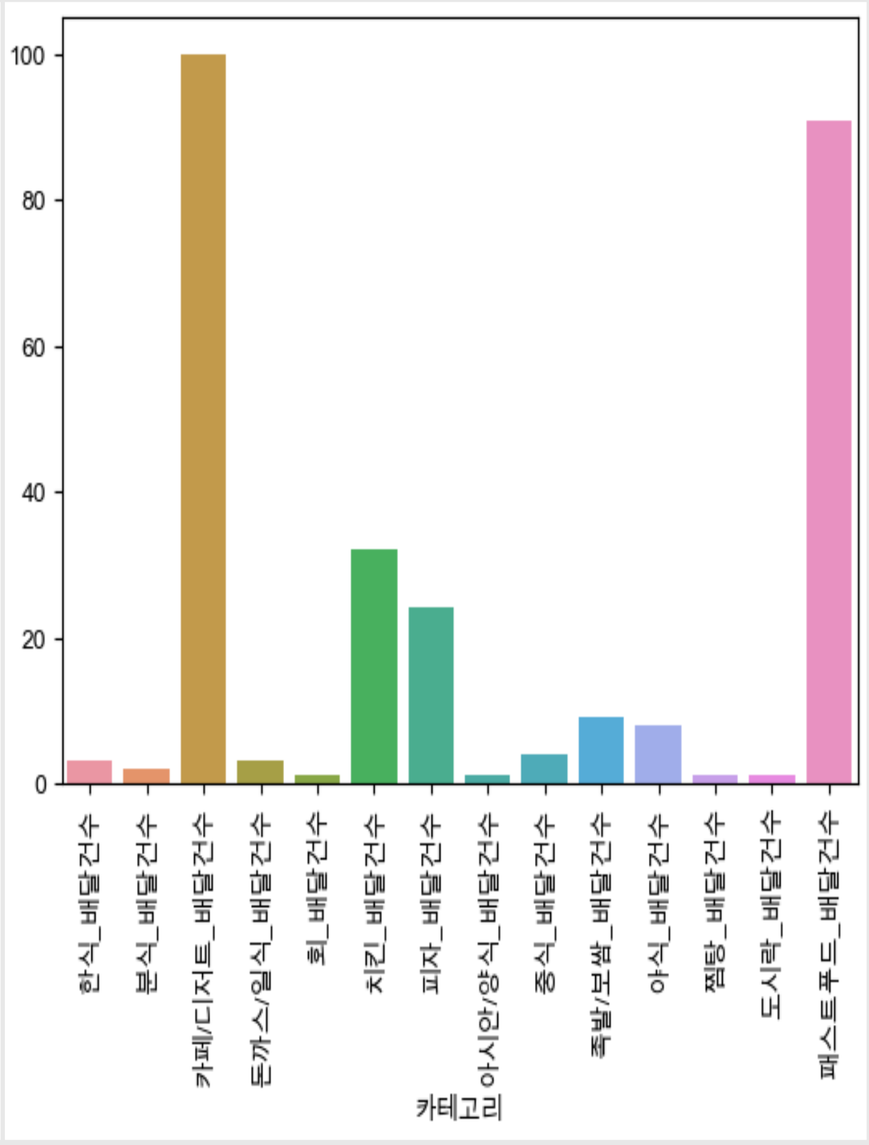




#2-3
세대별 휴일 주문 건수

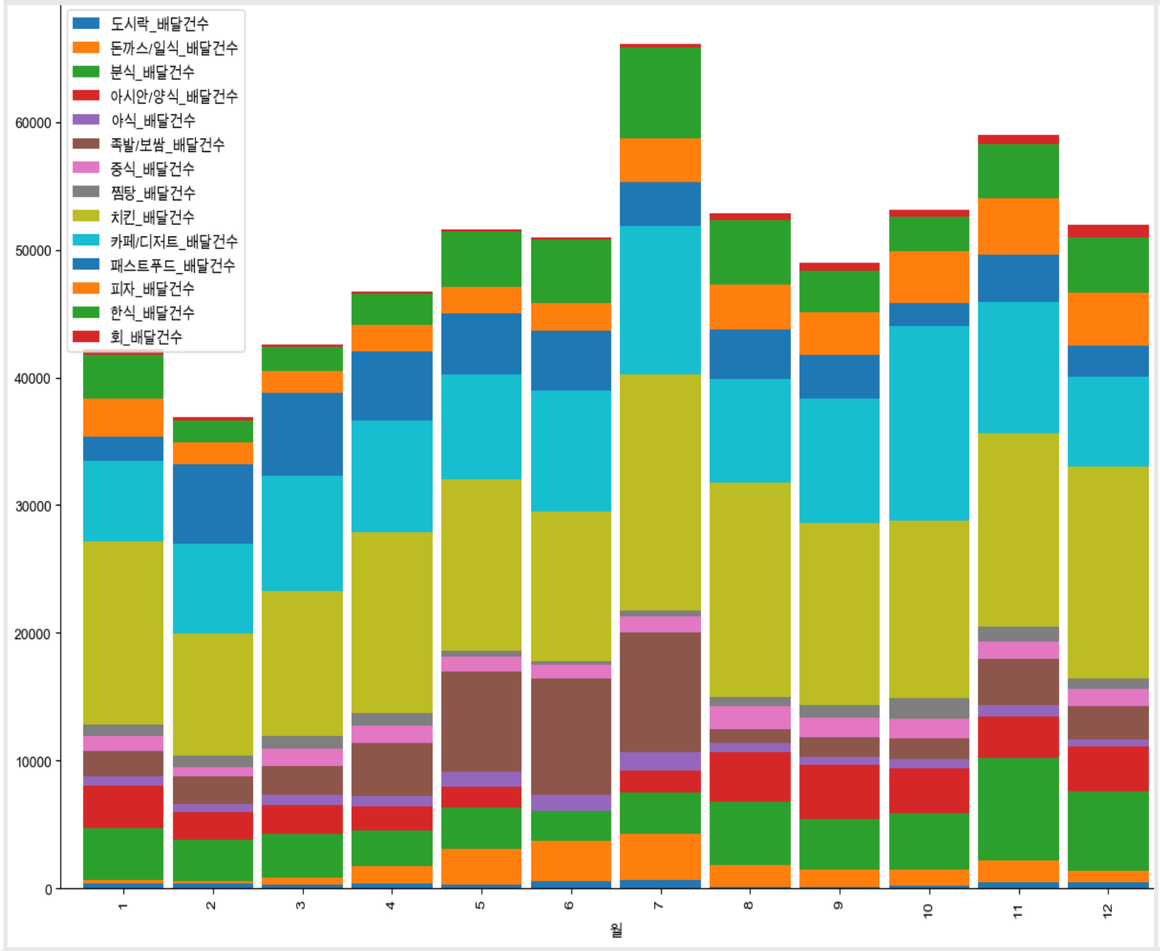
- 2030세대가 많은 지역 TOP10의
  크리스마스, 크리스마스 이브 , 신정
- 고령층(65세 이상)이 많은 지역 TOP10의
  설날, 추석 배달 카테고리

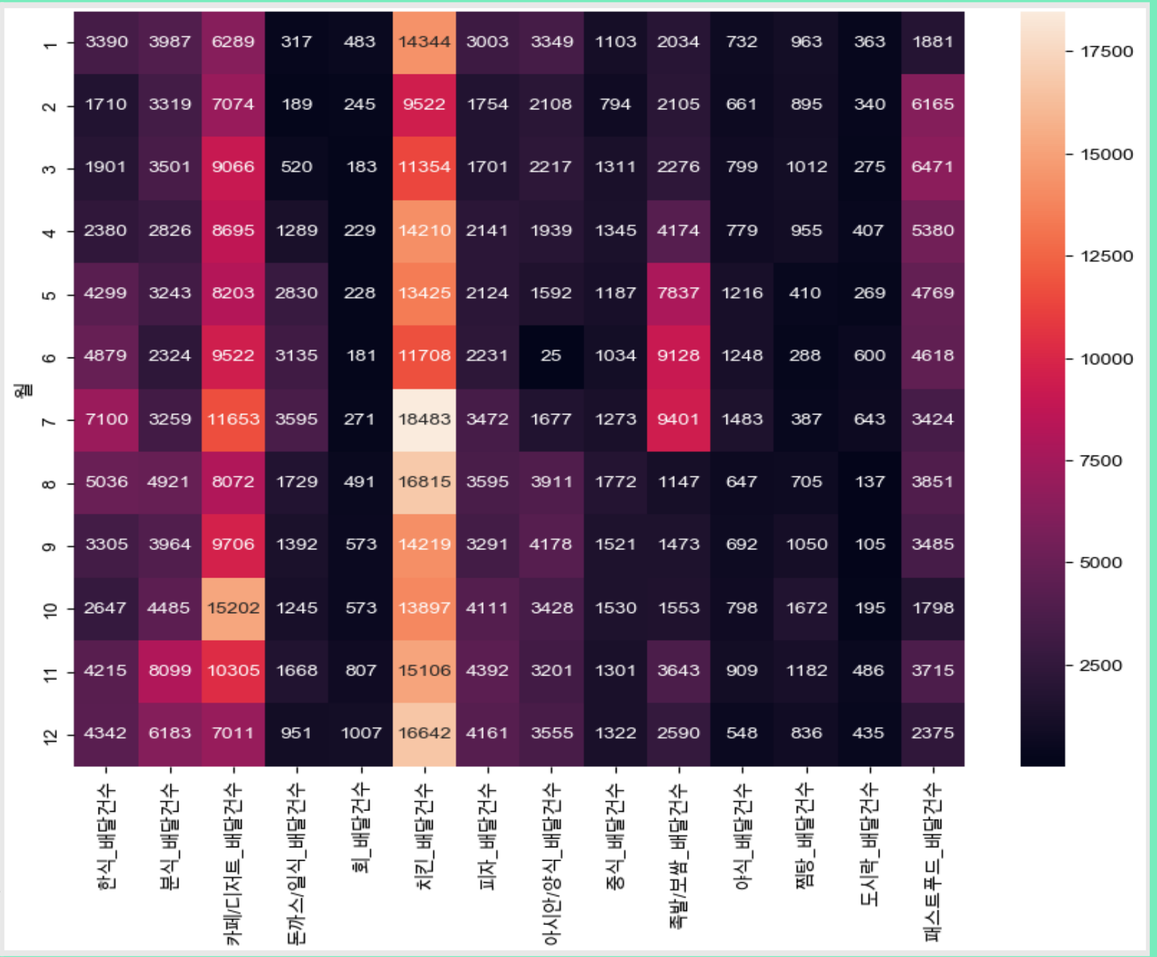




#2-4

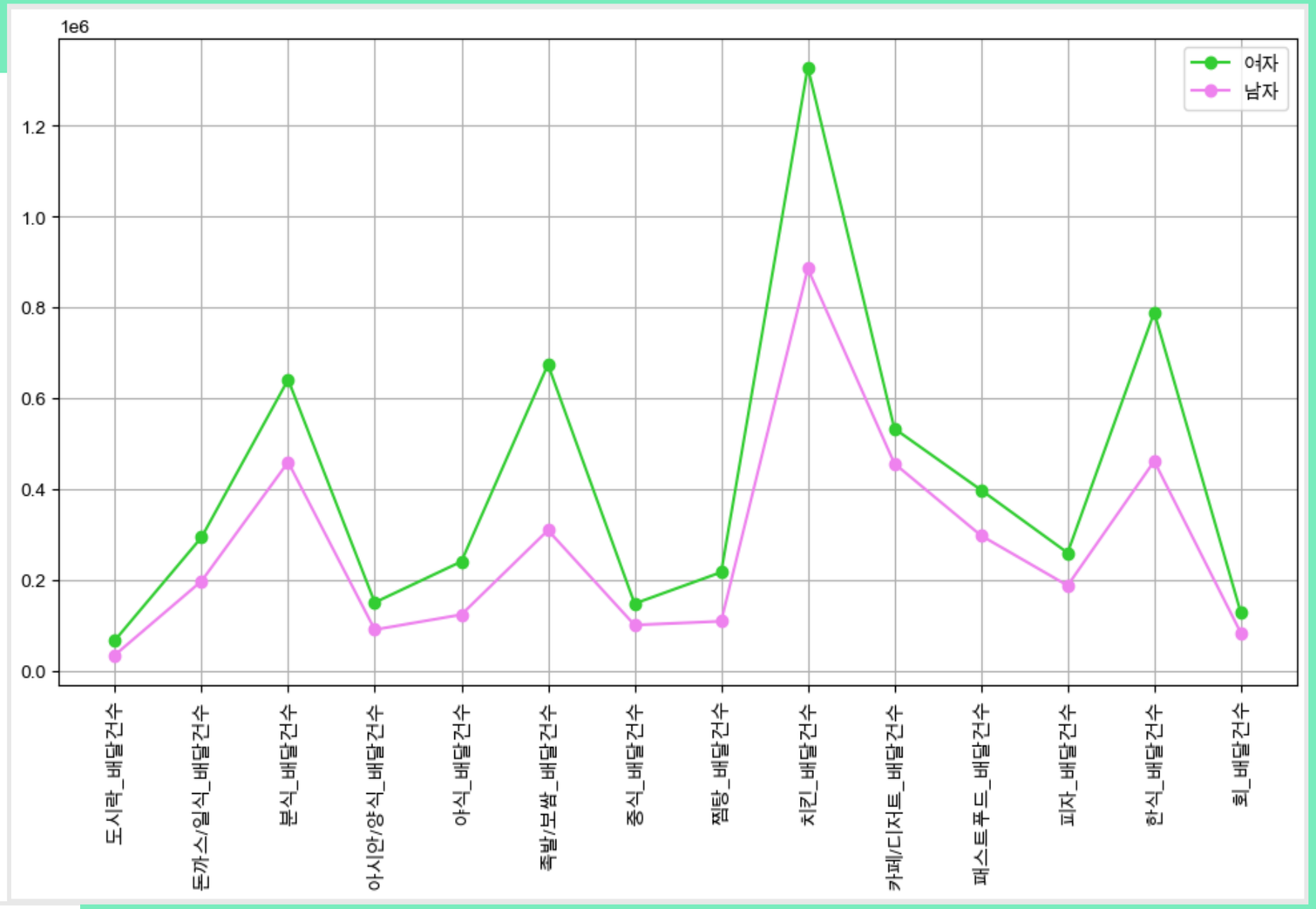
2030세대 월별 주문 건수

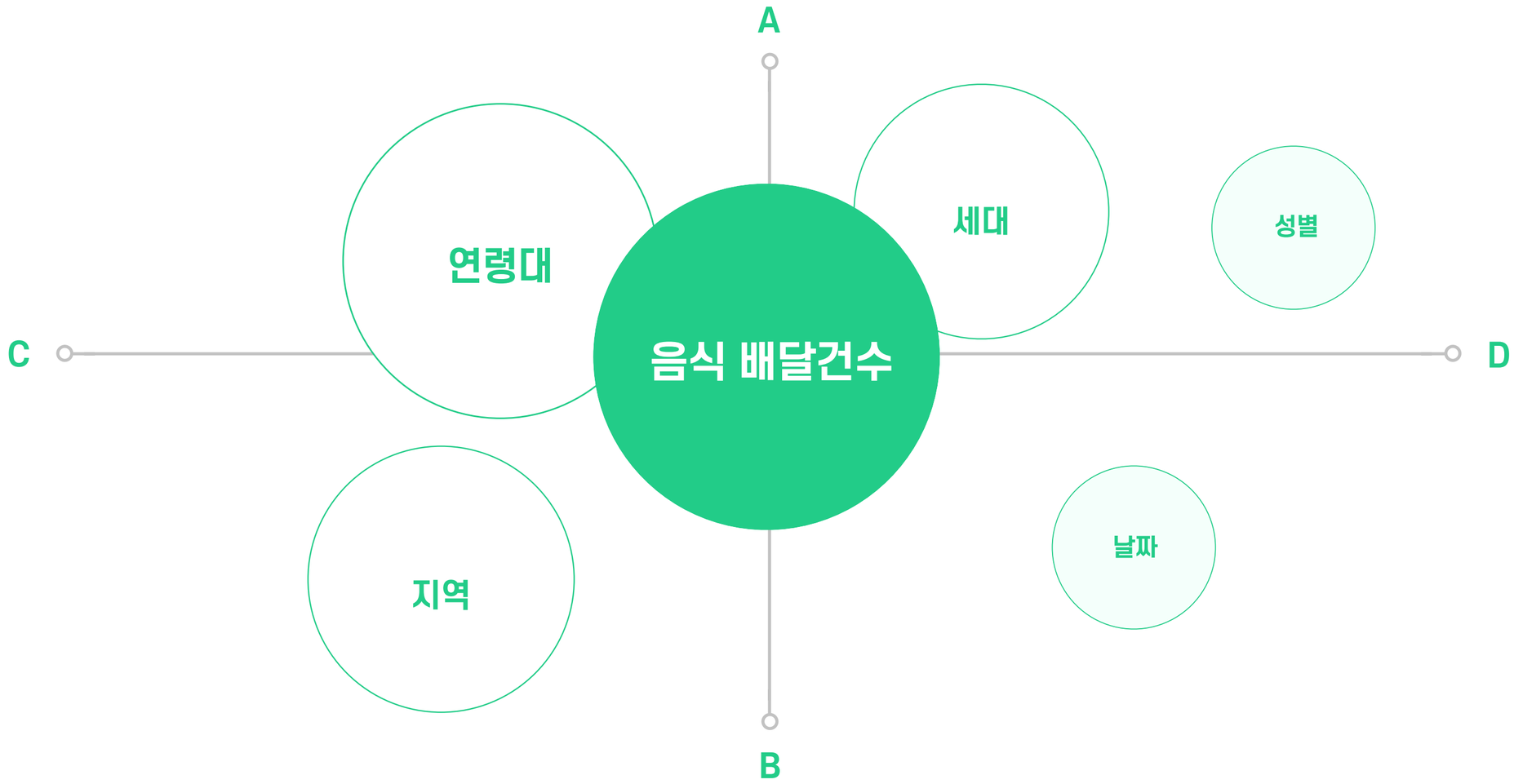




#2-5
2030세대 성별 주문 건수

- 배달카테고리별 남자와 여자 주문 건수
- 전반적으로 여자가 남자보다 많이 주문



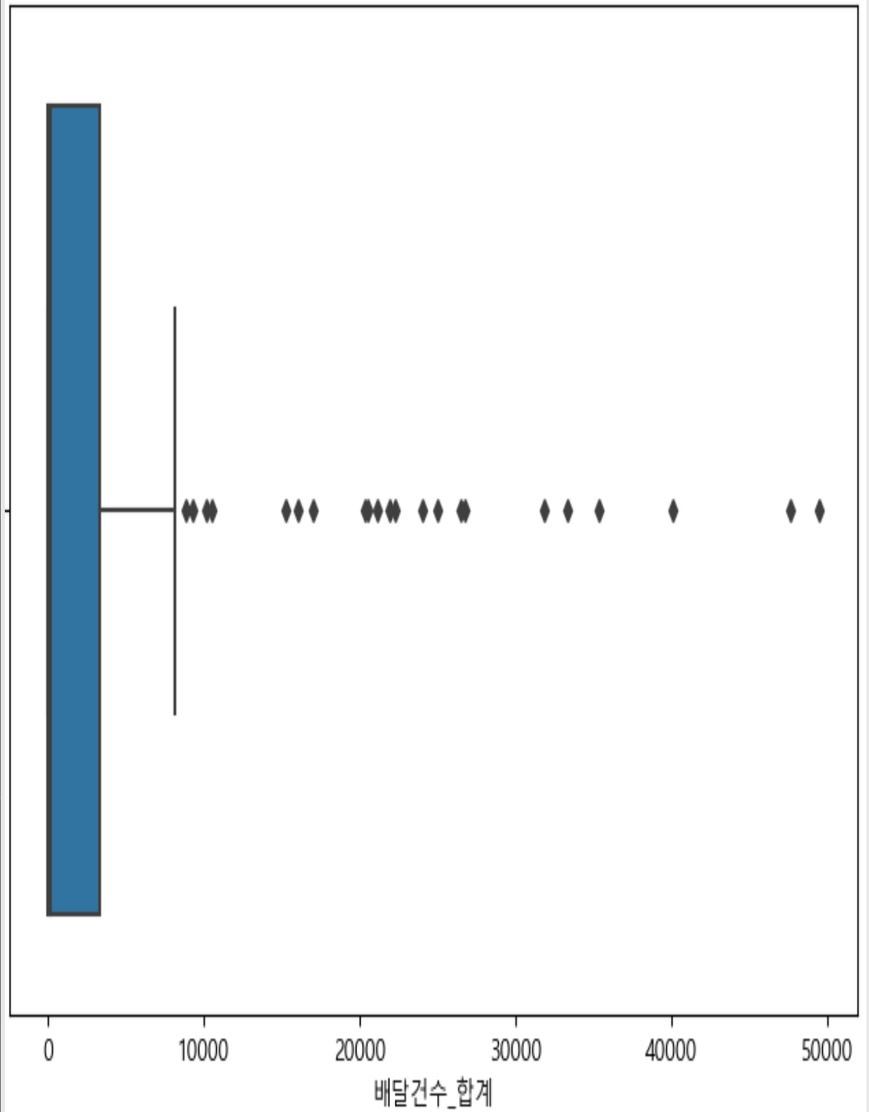

A
연령대
세대
성별
C
음식 배달건수
D
지역
날짜
B

#1-1
1차 이상치 제거

- BOX PLOT에서 유의미한 값들 계산
- 3사분위수와 1사분위수를 이용해 이상치제거

```
print('평균: ', de_test['배달건수_합계'].mean())
print('중앙값: ', de_test['배달건수_합계'].median())
print('최댓값: ', de_test['배달건수_합계'].max())
print('최솟값: ', de_test['배달건수_합계'].min())

q1 = de_test['배달건수_합계'].quantile(0.25)
q2 = de_test['배달건수_합계'].quantile(0.5)
q3 = de_test['배달건수_합계'].quantile(0.75)
q4 = de_test['배달건수_합계'].quantile(1)

iqr = q3-q1

iqr_3 = q3 + (1.5*iqr)
iqr_1 = q1 - (1.5*iqr)

print('3사분위수', iqr_3)
print('1사분위수', iqr_1)
```

```
평균:  56235.58139534884
중앙값:  1068.5
최댓값:  2069540
최솟값:  1
3사분위수 53345.75
1사분위수 -31992.25
```

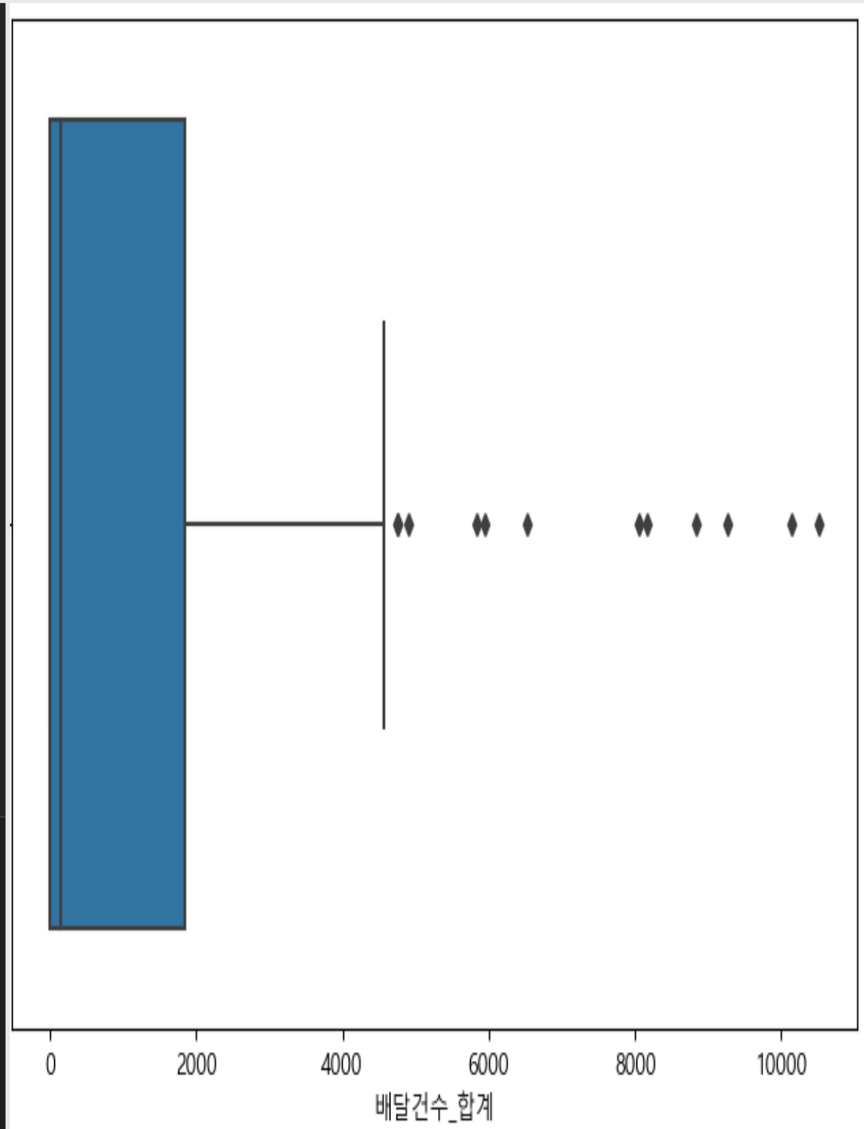


#1-2
2차 이상치 제거

- 1차 이상치를 제거 후에도 많은 이상치 존재
- 다시 3사분위수와 1사분위수를 이용해 이상치제거

```
print('평균: ', de_test_1['배달건수_합계'].mean())
print('중앙값: ', de_test_1['배달건수_합계'].median())
print('최댓값: ', de_test_1['배달건수_합계'].max())
print('최솟값: ', de_test_1['배달건수_합계'].min())

q1 = de_test_1['배달건수_합계'].quantile(0.25)
q2 = de_test_1['배달건수_합계'].quantile(0.5)
q3 = de_test_1['배달건수_합계'].quantile(0.75)
q4 = de_test_1['배달건수_합계'].quantile(1)

iqr = q3-q1

iqr_3 = q3 + (1.5*iqr)
iqr_1 = q1 - (1.5*iqr)

print('3사분위수', iqr_3)
print('1사분위수', iqr_1)
```

```
평균:  4626.95744680851
중앙값:  149.0
최댓값:  49479
최솟값:  1
3사분위수 8420.0
1사분위수 -5044.0
```



#1-3

x_train, y_train

- x_train : key(ID), 연령대별 주문량 합계
- y_train : key(ID), 배달건수 합계

x_train

	ID	20 - 24세합계	25 - 29세합계	30 - 34세합계	35 - 39세합계
0	예천	1960.0	1625.0	1879.0	2236.0
1	예산	4411.0	3392.0	3352.0	3960.0
2	산청	1591.0	1249.0	1273.0	1448.0
3	태백	2666.0	1845.0	2204.0	3261.0
4	함평	1714.0	1258.0	1275.0	1453.0
...	...	...	...	...	...
101	의왕	11549.0	10712.0	10678.0	11937.0
102	대구 서구	14302.0	12699.0	11909.0	12520.0
103	김포	20941.0	18492.0	24864.0	35284.0
104	연천	3064.0	2937.0	2467.0	2478.0
105	대전 대덕	14957.0	11963.0	11691.0	13525.0

106 rows × 5 columns

y_train

	ID	배달건수_합계
0	예천	2
1	예산	2
2	산청	2
3	태백	2
4	함평	2
...	...	...
101	의왕	8163
102	대구 서구	8829
103	김포	9260
104	연천	10148
105	대전 대덕	10507

106 rows × 2 columns

#1-4

범주형, 수치형 변수 데이터 분리

- 범주형 변수 : ID
- 수치형 변수 : 연령대별 주문량 합계(4개)
- LabelEncoder와 MinMaxScaler 활용

```
# 범주형 변수 전처리
from sklearn.preprocessing import LabelEncoder
encoder = LabelEncoder()
x_train['ID'] = encoder.fit_transform(x_train['ID'])
```

```
# 수치형 변수 전처리 (MinMaxScaler)
from sklearn.preprocessing import MinMaxScaler
scaler  =  MinMaxScaler()
x_train = scaler.fit_transform(x_train)
```

```
# 데이터 분리
from sklearn.model_selection import train_test_split
X_TRAIN, X_TEST, Y_TRAIN, Y_TEST = train_test_split(x_train, y_train['배달건수_합계'], test_size = 0.2, random_state = 13)
```

#2-1
모델링 및 평가

- RandomForestRegressor : 1296.69
- XGBRegressor : 1531.65
- LinearRegression : 1361.26
- Ridge : 1216.90
- SVR : 1404.08

```
modelRF = RandomForestRegressor()
modelRF.fit(X_TRAIN, Y_TRAIN)
y_pred_RF = modelRF.predict(X_TEST)
mse_RF = mean_squared_error(Y_TEST, y_pred_RF)
rmse_RF = mean_squared_error(Y_TEST, y_pred_RF, squared=False)
print("MSE:", mse_RF)
print("RMSE:", rmse_RF)
```

```
MSE: 1681428.086595238
RMSE: 1296.6989190229 31
```

```
modelXGB = XGBRegressor()
modelXGB.fit(X_TRAIN, Y_TRAIN)
y_pred_XGB = modelXGB.predict(X_TEST)
mse_XGB = mean_squared_error(Y_TEST, y_pred_XGB)
rmse_XGB = mean_squared_error(Y_TEST, y_pred_XGB, squared=False)
print("MSE:", mse_XGB)
print("RMSE:", rmse_XGB)
```

```
MSE: 2345974.9813531092
RMSE: 1531.6575927253157
```

```
modelLR = LinearRegression()
modelLR.fit(X_TRAIN, Y_TRAIN)
y_pred_LR = modelLR.predict(X_TEST)
mse_LR = mean_squared_error(Y_TEST, y_pred_LR)
rmse_LR = mean_squared_error(Y_TEST, y_pred_LR, squared=False)
print("MSE:", mse_LR)
print("RMSE:", rmse_LR)
```

```
MSE: 1853054.22405812
RMSE: 1361.269342950953 4
```

```
modelRI = Ridge(alpha=10)
modelRI.fit(X_TRAIN, Y_TRAIN)
y_pred_RI = modelRI.predict(X_TEST)
mse_RI = mean_squared_error(Y_TEST, y_pred_RI)
rmse_RI = mean_squared_error(Y_TEST, y_pred_RI, squared=False)
print("MSE:", mse_RI)
print("RMSE:", rmse_RI)
```

```
MSE: 1480856.5291415437
RMSE: 1216.904486449755 7
```

```
modelSVR = SVR()
modelSVR.fit(X_TRAIN, Y_TRAIN)
y_pred_SVR = modelSVR.predict(X_TEST)
mse_SVR = mean_squared_error(Y_TEST, y_pred_SVR)
rmse_SVR = mean_squared_error(Y_TEST, y_pred_SVR, squared=False)
print("MSE:", mse_SVR)
print("RMSE:", rmse_SVR)
```

```
MSE: 1971462.4512320966
RMSE: 1404.0877647896 86
```

#2-2
GridSearchCV

- RMSE가 낮은 RandomForest, Ridge
  => 하이퍼파라미터 조정
- RandomForestRegressor : 1163.85
- 조정 후 모델링 결과 성능향상

```
from sklearn.model_selection import GridSearchCV

rf_model = RandomForestRegressor()

param_grid = {
    'n_estimators': [50, 100, 200, 300],
    'max_depth': [10, 20, 30, 40],
    'min_samples_split': [2, 4, 6, 8],
    'min_samples_leaf': [1, 2, 3, 4]
}

grid_search = GridSearchCV(estimator=rf_model, param_grid=param_grid, cv=5, n_jobs=-1)

grid_search.fit(x_train, y_train['배달건수_합계'])

print('Best Parameters:', grid_search.best_params_)
print('Best Score:', grid_search.best_score_)
```

```
Best Parameters: {'max_depth': 30, 'min_samples_leaf': 4, 'min_samples_split': 8, 'n_estimators': 50}
Best Score: -157890.3145359737
```

```
modelRF = RandomForestRegressor(max_depth= 30, min_samples_leaf=4, min_samples_split=8, n_estimators=50)
modelRF.fit(X_TRAIN, Y_TRAIN)
y_pred_RF_g = modelRF.predict(X_TEST)

mse_RF_g = mean_squared_error(Y_TEST, y_pred_RF_g)
rmse_RF_g = mean_squared_error(Y_TEST, y_pred_RF_g, squared=False)

print("MSE:", mse_RF_g)
print("RMSE:", rmse_RF_g)
```

```
MSE: 1354567.6648361846
RMSE: 1163.8589540129785
```

배달요식업 자영업자들에게 2030세대의
주문건수 예측을 가능하게 함으로써
효율적인 경영전략을 제공한다.

THANK YOU!

ML_PJT 1조

김정빈 박건우 윤정옥