소상공인 관련 연구 현안문제 해결

Whiskey조

화학공학과 22 송지윤
환경공학부 20 양현주
인공지능학과 21 이상민
수학과 19 황태연

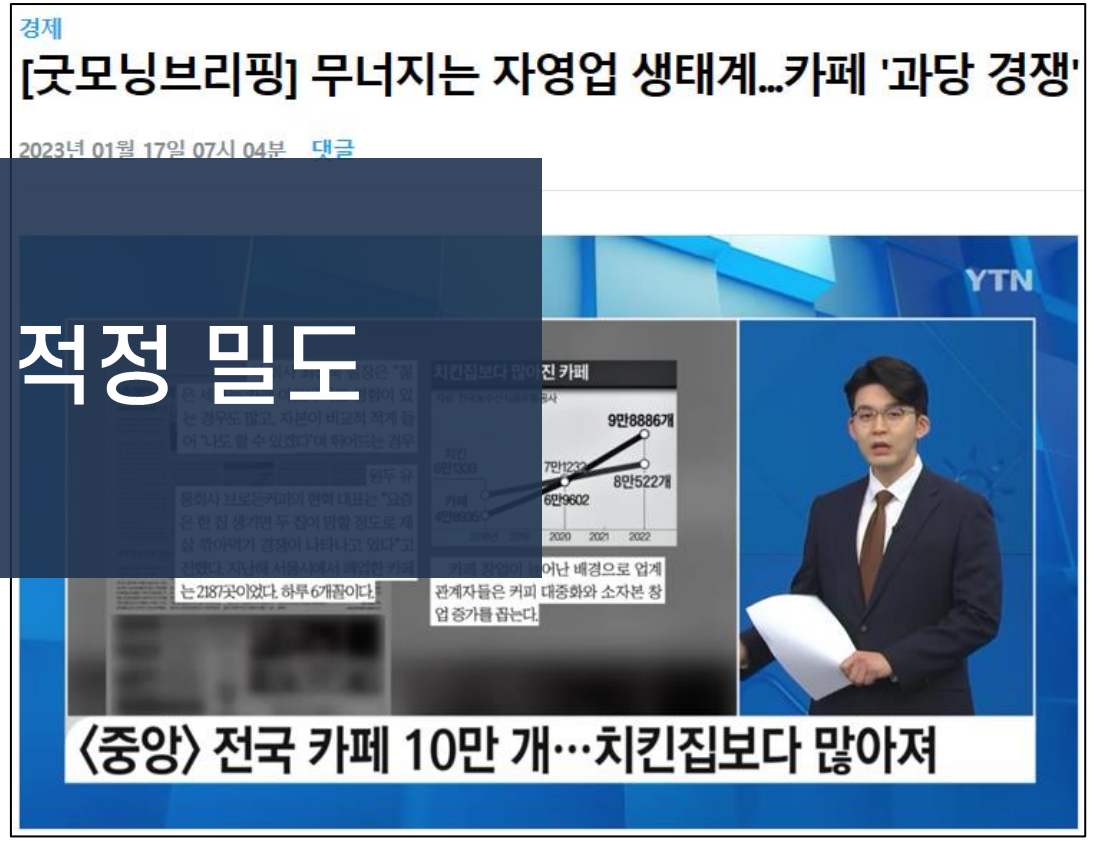
목차 A table of contents

Part 1

현안 문제

1 현안 문제

“ 급증하는 카페, 이대로 괜찮은가? ”

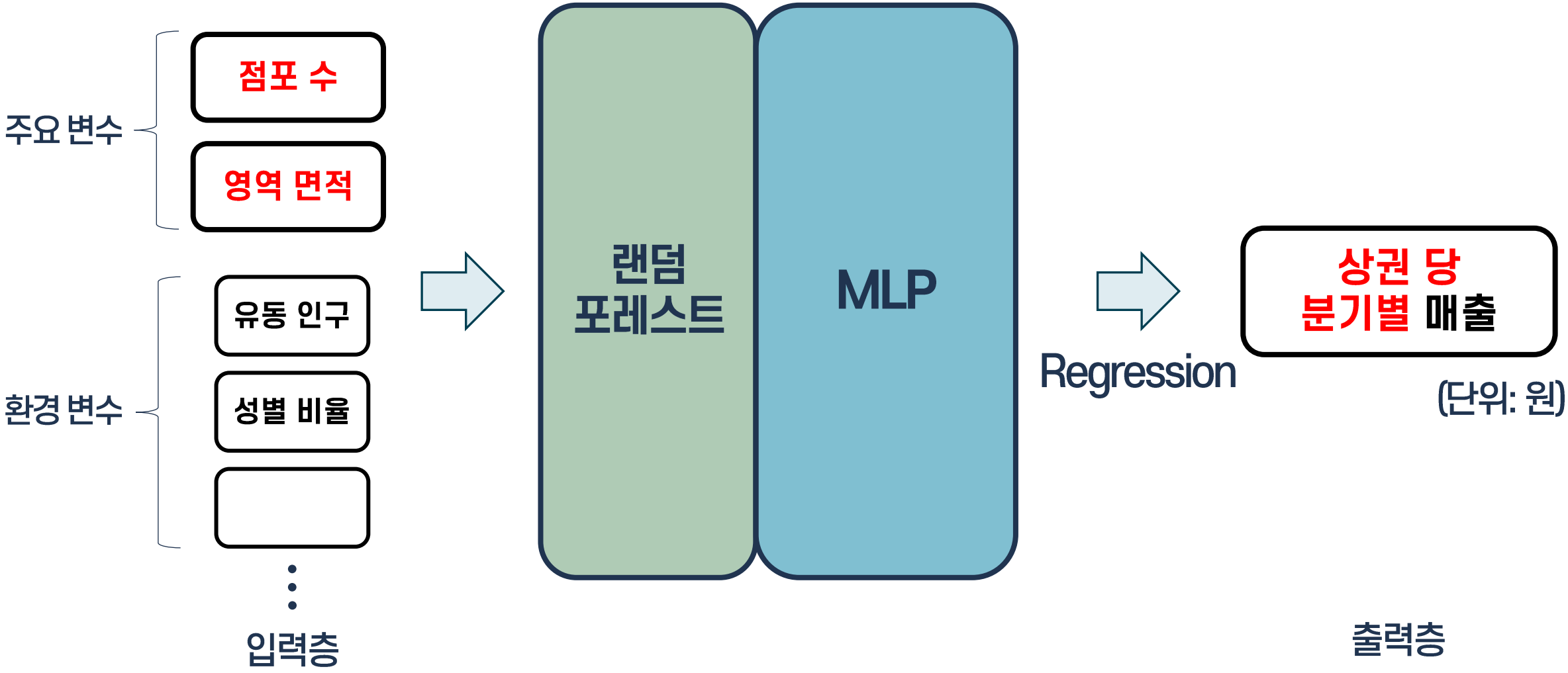


상권 내 카페의 적정 밀도

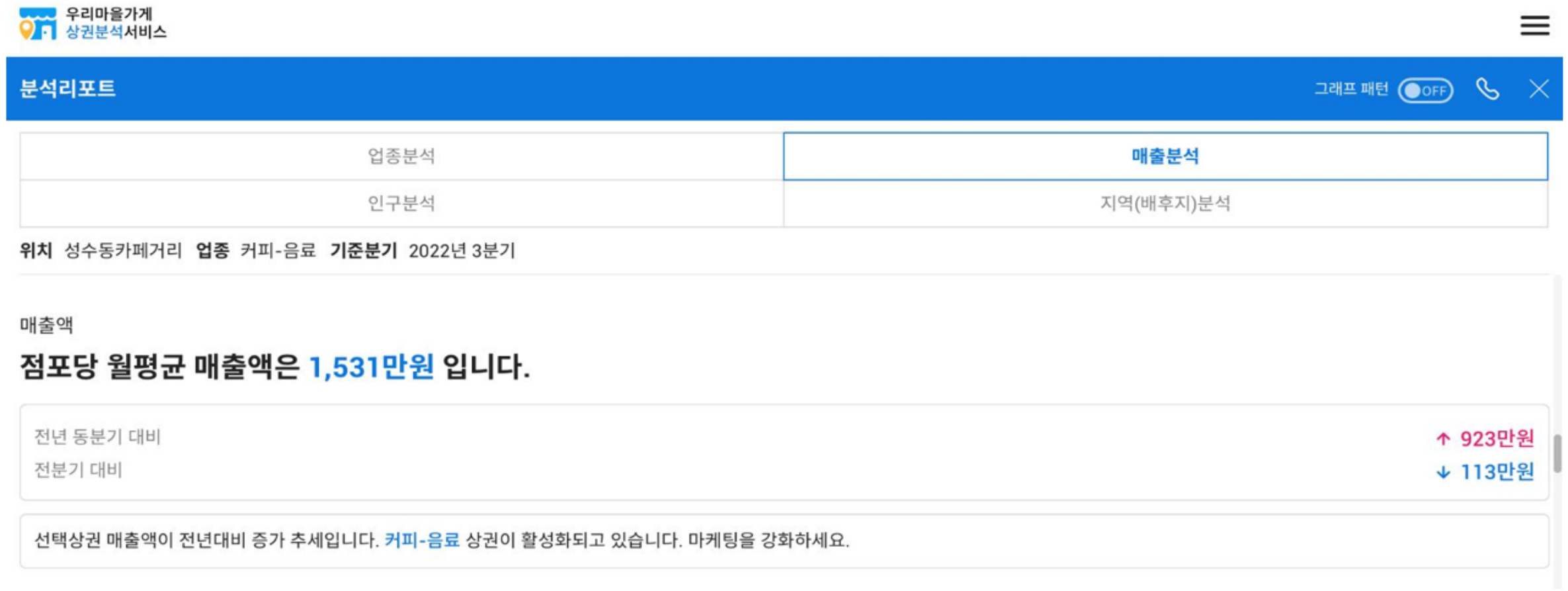
Part 2

모델 설계 및 구축

2 모델 설계 및 구축



2 모델 설계 및 구축



받은 데이터 중 당월 매출액(2,249,972,021) / 점포 수(49)
= 사이트 속 점포 당 월별 매출액(1,531만원) * 분기 당 개월 수(3)

2 모델 설계 및 구축

[1. 상관계수 분석]

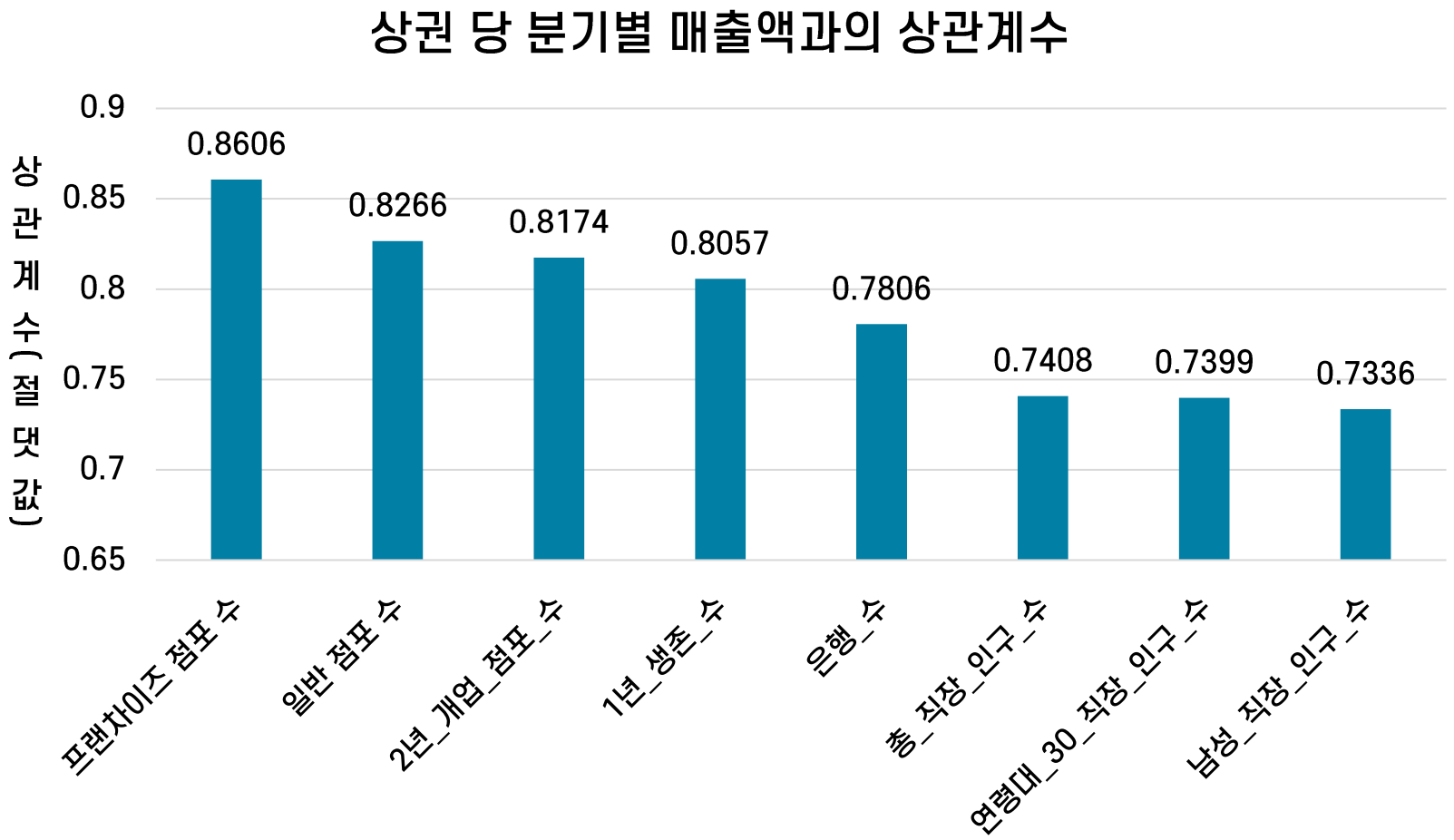
1. 점포 수
2. 영역 면적
3. 유동인구, 상존인구
    1. 전체 인구 수
    2. 인구 수 – 성별
    3. 인구 수 – 연령대
    4. 인구 수 – 시간대
    5. 인구 수 - 요일
4. 매출 비율
    1. 매출 비율 – 주중/주말
    2. 매출 비율 – 요일
    3. 매출 비율 – 시간대
    4. 매출 비율 – 성별
    5. 매출 비율 – 연령대
5. 프랜차이즈 점포 수
6. 개업/폐업률
7. 개업/폐업 점포 수
8. 상주인구, 직장 인구
    1. 전체 인구 수
    2. 인구 수 – 성별
    3. 인구 수 – 연령대
9. 상권 주변 시설 개수
10. 연차별 생존율
11. 연차별 생존 수

매출 금액은 제외!

2 모델 설계 및 구축

선별 기준

1. 상관계수 높은 순
2. 대표성을 나타냄
3. 유의미한 다양성



2 모델 설계 및 구축

✓ 선별 기준

1. 상관계수 높은 순
2. 대표성을 나타냄
3. 유의미한 다양성

19년도 20년도 발달상권에서
상권 인근 유동인구는 외식업종
매출액에 영향을 줌.

분류	index	select		상관계수	절댓값
매출	563	FRI_SELNG_AMT	금요일_매출_금액	0.993452	0.993452
	557	MDWK_SELNG_AMT	주중_매출_금액	0.990914	0.990914
	561	WED_SELNG_AMT	수요일_매출_금액	0.987734	0.987734
	572	ML_SELNG_AMT	남성_매출_금액	0.987638	0.987638
	576	AGRDE_30_SELNG_AMT	연령대_30_매출_금액	0.9821	0.9821
	569	TMZON_14_17_SELNG_AMT	시간대_14_17_매출_금액	0.977955	0.977955
	552	AGRDE_20_SELNG_RATE	연령대_20_매출_비율	0.245684	0.245684
	534	MDWK_SELNG_RATE	주중_매출_비율	0.195651	0.195651
	550	FML_SELNG_RATE	여성_매출_비율	0.186984	0.186984
	544	TMZON_06_11_SELNG_RATE	시간대_06_11_매출 비율	0.183231	0.183231
	540	FRI_SELNG_RATE	금요일_매출_비율	0.179432	0.179432
	533	THSMON_SELNG_CO	당월_판매_건수	0.965551	0.965551
	603	STOR_CO_x	점포_수	0.856524	0.856524
유동인구	16	TMZON_4_FLPOP_CO_x	시간대_4_유동인구_수	0.541474	0.541474
	8	AGRDE_20_FLPOP_CO_x	연령대_20_유동인구_수	0.477429	0.477429
	23	FRI_FLPOP_CO_x	금요일_유동인구_수	0.38966	0.38966
	5	ML_FLPOP_CO_x	남성_유동인구_수	0.376432	0.376432
	4	TOT_FLPOP_CO_x	총_유동인구_수	0.353343	0.353343
상존인구	624	TMZON_4_FLPOP_CO_y	시간대_4_상존인구_수	0.661925	0.661925
	616	AGRDE_20_FLPOP_CO_y	연령대_20_상존인구_수	0.623099	0.623099
	631	FRI_FLPOP_CO_y	금요일_상존인구_수	0.583916	0.583916
	613	ML_FLPOP_CO_y	남성_상존인구_수	0.575586	0.575586
	612	TOT_FLPOP_CO_y	총_상존인구_수	0.556088	0.556088

2 모델 설계 및 구축

✓ 선별 기준

1. 상관계수 높은 순
2. 대표성을 나타냄
3. 유의미한 다양성

상주인구는 20년 골목상권에서 유의미함.

상주인구	1162	AGRDE_30_REPOP_CO	연령대_30_상주인구_수	0.165837	0.165837
	1159	FML_REPOP_CO	여성_상주인구_수	0.087702	0.087702
	1157	TOT_REPOP_CO	총_상주인구_수	0.082845	0.082845
직장 인구 수	1205	TOT_WRC_POPLTN_CO	총_직장_인구_수	0.740832	0.740832
	1210	AGRDE_30_WRC_POPLTN_CO	연령대_30_직장_인구_수	0.739927	0.739927
	1206	ML_WRC_POPLTN_CO	남성_직장_인구_수	0.733593	0.733593
시설	1231	BANK_CO	은행_수	0.780643	0.780643
	1229	VIATR_FCLTY_CO	집객시설_수	0.715297	0.715297
연차별 생존	1145	YEAR_2_OPBIZ_STOR_CO_x	2년_개업_점포_수	0.817426	0.817426
	1141	YEAR_1_BEING_CO_x	1년_생존_수	0.805663	0.805663
	1140	YEAR_1_BEING_RT_x	1년_생존_율	0.197306	0.197306
신생기업 생존	1204	YEAR_5_OPBIZ_STOR_CO_y	5년_개업_점포_수	0.826491	0.826491
	1197	YEAR_3_BEING_CO_y	3년_생존_수	0.769061	0.769061
	1190	YEAR_1_BEING_RT_y	1년_생존_율	0.194096	0.194096
영업개월	1184	TOT_BSN_MT	총_영업개월(10년전기준)	0.890791	0.890791
	1188	TOT_STOR_CO_90	총_점포_수(1990년기준)	0.885853	0.885853
	1186	SALE_MT_AVG_90	평균영업개월(1990년기준)	0.10027	0.10027
개폐업	606	SIMILR_INDUTY_STOR_CO	총_점포_수	0.883539	0.883539
	611	FRC_STOR_CO	프렌차이즈_점포_수	0.860648	0.860648
	605	STOR_CO_y	일반_점포_수	0.826598	0.826598
	608	OPBIZ_STOR_CO	개업_점포_수	0.64276	0.64276
	610	CLSBIZ_STOR_CO	폐업_점포_수	0.634718	0.634718
	607	OPBIZ_RT	개업_율	-0.01245	0.012449
	609	CLSBIZ_RT	폐업_율	0.000572	0.000572

2 모델 설계 및 구축

[2. 모델 설계]

1) 데이터 전처리

테스트 데이터는 2022년 3분기 데이터 이용

Data Preprocessing

```
train_df = pd.read_csv('train_df.csv')
test_df = pd.read_csv('test_df.csv')
```

```
# Data Shuffle
train_df = train_df.sample(frac=1, random_state=0).reset_index(drop=True)
test_df
```

2 모델 설계 및 구축

[2. 모델 설계]

1) 데이터 전처리

```
# 필요한 데이터만 collect
L = [530, 603, 1255, 1256, 550, 551, 555, 552, 556, 524,
     4, 8, 12, 16, 13, 25, 1177, 1157, 1211, 1212,
     1210, 1209, 1213, 1208, 1205, 534, 544, 540, 23, 5,
     1159, 1162, 1206, 1229, 1204, 1197, 606, 611, 608, 610,
     1240, 1234, 1248, 1247, 1230, 1233, 1243, 1239, 1232] # 49개의 변수

L.append(532)

train = train_data[:, L]
test = test_data[:, L]
```

2 모델 설계 및 구축

[2. 모델 설계]

1) 데이터 전처리

```
# Data Cleaning
train = np.array(train, dtype=float)
is_row_nan = np.isnan(train).any(axis=1)
train = train[np.logical_not(is_row_nan), :]

test = np.array(test, dtype=float)
is_row_nan = np.isnan(test).any(axis=1)
test = test[np.logical_not(is_row_nan), :]

print(train.shape, test.shape)
```

2 모델 설계 및 구축

[2. 모델 설계]

1) 데이터 전처리

```
# Nan 데이터를 제거한 유효 데이터 비율
valid_train = train.shape[0] / 17294
valid_test = test.shape[0] / 1216
print(valid_train, valid_test)

# 90% train, 10% valid
split_point = int(train.shape[0]*0.9)
print('split point:', split_point)
```

2 모델 설계 및 구축

[2. 모델 설계]

1) 데이터 전처리

```
"""
# Data Scaling (가우스 랭크)
scaler = QuantileTransformer(output_distribution='normal')
train_scaled = scaler.fit_transform(train)
test_scaled = scaler.transform(test)
"""

# Data Scaling
scaler = StandardScaler()
scaler.fit(train)
train_scaled = scaler.transform(train)
test_scaled = scaler.transform(test)
```

2 모델 설계 및 구축

[2. 모델 설계]

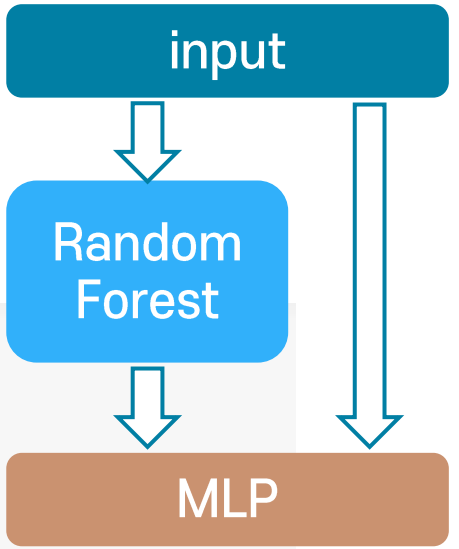
1) 데이터 전처리

```
# Split data into training and test sets
train_data = train_scaled[:split_point, :]
val_data = train_scaled[split_point:, :]
test_data = test_scaled

X_train, y_train = train_data[:, :-1], train_data[:, -1]
X_val, y_val = val_data[:, :-1], val_data[:, -1]
X_test, y_test = test_data[:, :-1], test_data[:, -1]

X_train
```

2 모델 설계 및 구축

[2. 모델 설계]

2) 랜덤 포레스트 (Stacking)




```
from sklearn.ensemble import RandomForestRegressor

# Train the Random Forest regression model
rf = RandomForestRegressor(n_estimators=500, max_leaf_nodes = 16,
                           random_state=0)
rf.fit(X_train, y_train)

# Get the Random Forest predictions on the training set
rf_train_preds = rf.predict(X_train)

# Add the Random Forest predictions as a new feature in the training data
X_train_stacked = np.column_stack((X_train, rf_train_preds))
```

2 모델 설계 및 구축

[2. 모델 설계]

3) MLP 모델

```
model = models.Sequential()
model.add(layers.Dense(64, activation = 'relu'))
model.add(layers.Dense(32, activation = 'relu'))
model.add(layers.Dense(16, activation = 'relu'))
model.add(layers.Dense(8, activation = 'relu'))
model.add(layers.Dense(4, activation = 'relu'))
model.add(layers.Dense(1))

# Compile the model
model.compile(optimizer='adam', loss='mean_squared_error')
```

2 모델 설계 및 구축

[2. 모델 설계]

3) MLP 모델

```
# Get the Random Forest predictions on the val set
rf_val_preds = rf.predict(X_val)

# Add the Random Forest predictions as a new feature in the val data
X_val_stacked = np.column_stack((X_val, rf_val_preds))

# Get the Random Forest predictions on the test set
rf_test_preds = rf.predict(X_test)

# Add the Random Forest predictions as a new feature in the test data
X_test_stacked = np.column_stack((X_test, rf_test_preds))
```

2 모델 설계 및 구축

[2. 모델 설계]

3) MLP 모델

```
# Train the model
history = model.fit(X_train_stacked, y_train, epochs=200,
                    validation_data=(X_val_stacked, y_val))

# Evaluate the model on the test data
model.evaluate(X_test_stacked, y_test)
```

[2. 모델 설계]

4) 모델 성능 지표 출력

```
# 모델 성능 지표
import matplotlib.pyplot as plt

plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.legend(['Train', 'Val'], loc = 'upper left')
plt.show()
```

2 모델 설계 및 구축

[2. 모델 설계]

4) 모델 성능 지표 출력

```
import matplotlib.pyplot as plt

predictions = model.predict(X_test_stacked)
plt.plot(predictions, y_test, 'o', label='Ground truth vs Predictions')
plt.plot((0, 15), (0, 15))

plt.xlabel('Predictions')
plt.ylabel('Ground truth')
plt.legend()
plt.show()
```

[2. 모델 설계]

4) 모델 성능 지표 출력

```
# 로그 스케일로 그래프 출력
plt.plot(np.log(predictions), np.log(y_test), 'o',
         label='Ground truth vs Predictions')
plt.plot((-5, 2), (-5, 2))
plt.xlabel('Predictions')
plt.ylabel('Ground truth')
plt.legend()
plt.show()
```

2 모델 설계 및 구축

중소벤처기업부

2019년 보도 자료

2017년 월 평균 영업이익:

2,690,000

&

2017년 영업이익률: **15.8%**

2017년 월 평균 영업이익: **2,690,000**

2017년 대비 2022년 물가상승률: **10.33%**

⟶ 2,690,000 X 1.1033 = **2,967,877**

매출액(기준) = 영업이익 X 100% / 영업이익률(%)

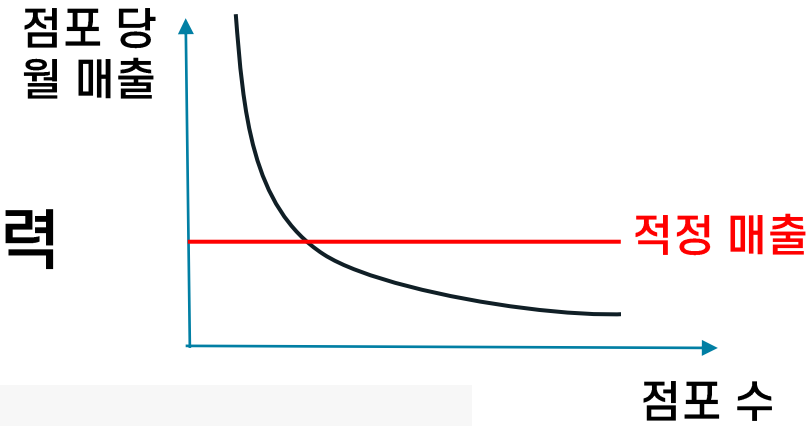
= **2,967,877** X 100% / **15.8%**

= **18,784,031**

2 모델 설계 및 구축

[3. 결과 출력]

1) 점포 수 – 점포 당 월 매출 그래프 및 적정 한계선 출력




```
def Result(num):
    # num을 선택할 때에는 nan 데이터가 포함되어 있지 않은 데이터인지 확인해야 한다.
    Xy = test_df.loc[num][L]
    N = 1000
    Xy = np.tile(Xy, (N, 1))
    Xy[:, 1] = np.arange(1, N+1)

    Xy = scaler.transform(Xy)

    X = Xy[:, :-1]
    y = Xy[:, -1]
```

2 모델 설계 및 구축

[3. 결과 출력]

1) 점포 수 – 점포 당 월 매출 그래프 및 적정 한계선 출력

```
    # Get the Random Forest predictions on the set
    rf_preds = rf.predict(X)

    # Add the Random Forest predictions as a new feature in the data
    X_stacked = np.column_stack((X, rf_preds))
    pred = model.predict(X_stacked)
    pred = np.tile(pred, (1, len(L)))
    pred = scaler.inverse_transform(pred)
    # 분기별 매출에서 3달 나눔
    result = pred[:, -1].reshape(N)/(3*np.arange(1, N+1))
    return result
```

2 모델 설계 및 구축

[3. 결과 출력]

1) 점포 수 – 점포 당 월 매출 그래프 및 적정 한계선 출력

```
def IMG(num, M = 5, N = 300):
    result = Result(num)
    plt.xlim(M, N)
    plt.ylim(0,3e7)

    standard = 18784031 ################## STANDARD ####################

    plt.axhline(standard, color='red')
    plt.axvline(test_df.loc[num][603], color='green')

    plt.xlabel('Number of stores')
    plt.ylabel('Expected revenue per store')
```

2 모델 설계 및 구축

[3. 결과 출력]

1) 점포 수 – 점포 당 월 매출 그래프 및 적정 한계선 출력

```
    x = np.arange(N)
    plt.plot(x, result[:N])

    # Find the difference between the two y values
    y_diff = result - standard
    y_abs_diff = abs(y_diff)

    intersection_x = np.argmin(y_abs_diff)
    print(intersection_x)
```

2 모델 설계 및 구축

[3. 결과 출력]

1) 점포 수 – 점포 당 월 매출 그래프 및 적정 한계선 출력

```
# Plot the two functions and the intersection point
x = int(intersection_x)
y = int(result[int(intersection_x)])

t = test_df.loc[num][603]
t = int(t)
```

2 모델 설계 및 구축

[3. 결과 출력]

1) 점포 수 – 점포 당 월 매출 그래프 및 적정 한계선 출력

```
    plt.scatter(t, int(result[t]), color='green', label='now')
    plt.annotate(f'({t}, {int(result[t])})', (t, int(result[t])), textcoords='offse
                 arrowprops=dict(arrowstyle='->', connectionstyle='arc3,rad=0.2', c

    plt.scatter(x, y, color='red', label='pred')

    plt.annotate(f'({x}, {y})', (x, y), textcoords='offset points', xytext=(50, 15)
            arrowprops=dict(arrowstyle='->', connectionstyle='arc3,rad=0.2', color=

    plt.legend()

    # Show the plot
    plt.show()
```

2 모델 설계 및 구축

[3. 결과 출력]

1) 점포 수 – 점포 당 월 매출 그래프 및 적정 한계선 출력

```
def Analysis(num):
    try:
        result = Result(num)
        standard = 18784031 ################## STANDARD ####################

        y_diff = result - standard
        y_abs_diff = abs(y_diff)
        intersection_x = np.argmin(y_abs_diff)
    except:
        return 0

    print(num)
    return int(intersection_x) - test_df.loc[num][603]
```

2 모델 설계 및 구축

[3. 결과 출력]

2) 과당 경쟁 및 집적 효과 맵 출력

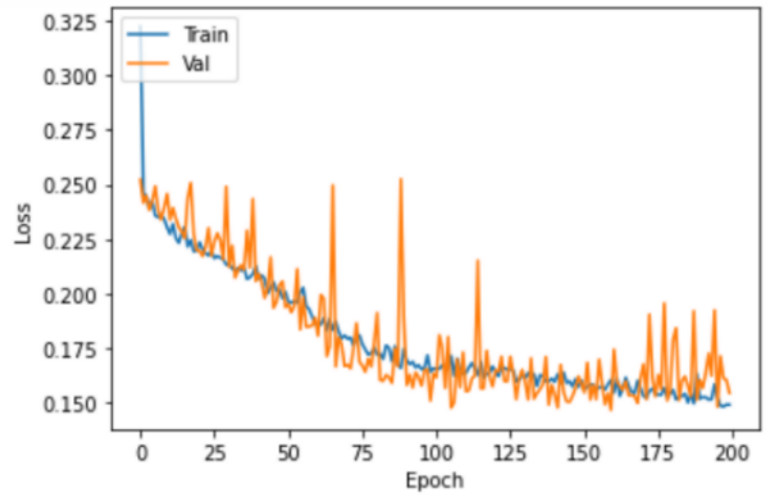
```
xy = test_df.iloc[:, [2, 3, 1251, 1255, 1256]]

# assign a color to each code
colors_list = ['red' if Analysis(idx) <= -5
               else 'yellow' if Analysis(idx) <= 5
               else 'green' for idx in xy.index]

fig = plt.figure(figsize=(12, 8))

# plot the points with different colors based on the code
plt.scatter(xy['X_VALUE'], xy['Y_VALUE'], c=colors_list, s=1)

plt.show()
```

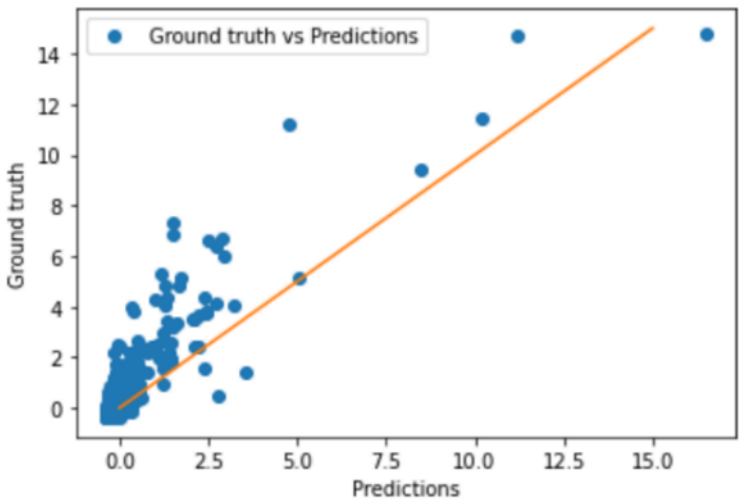
Part 3

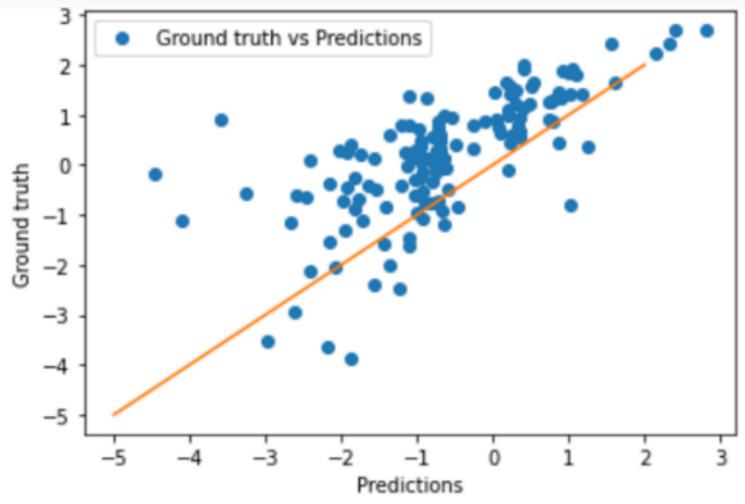
모델 결과

3 모델 결과

주요 변수만



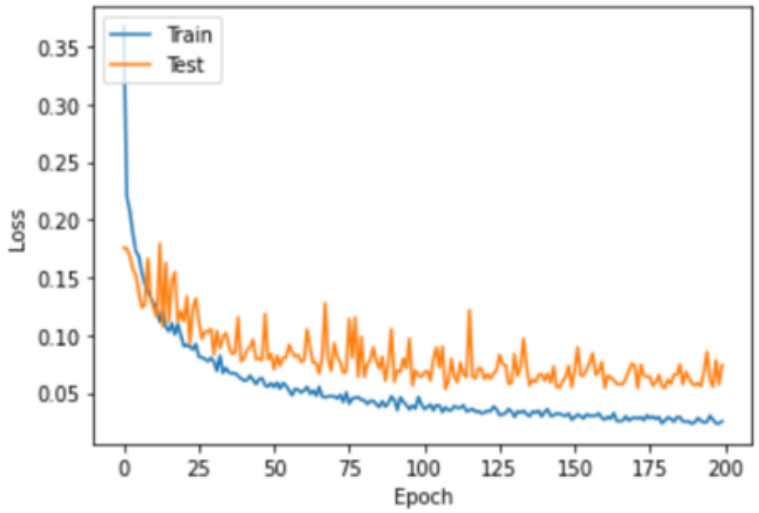


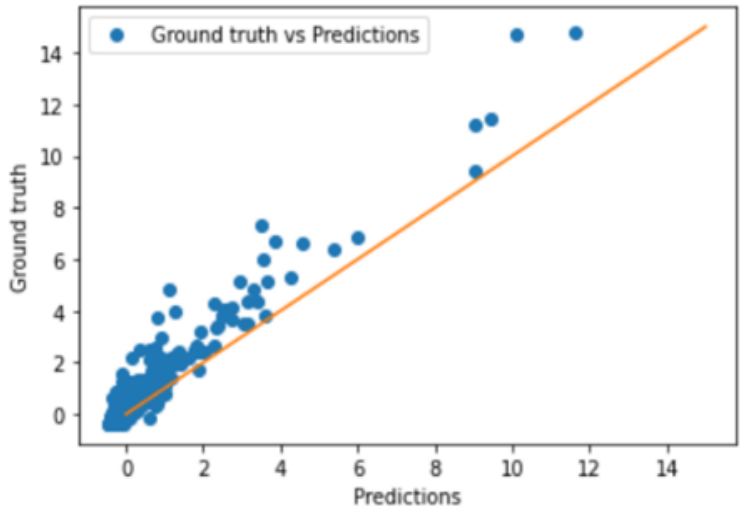


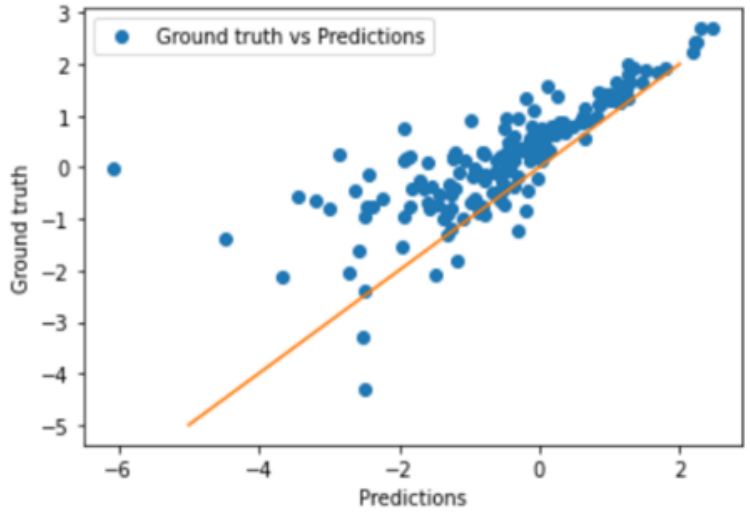
1회	2회	3회	평균
0.3865	0.4332	0.3742	0.3980

3 모델 결과

상관 계수 고려 X



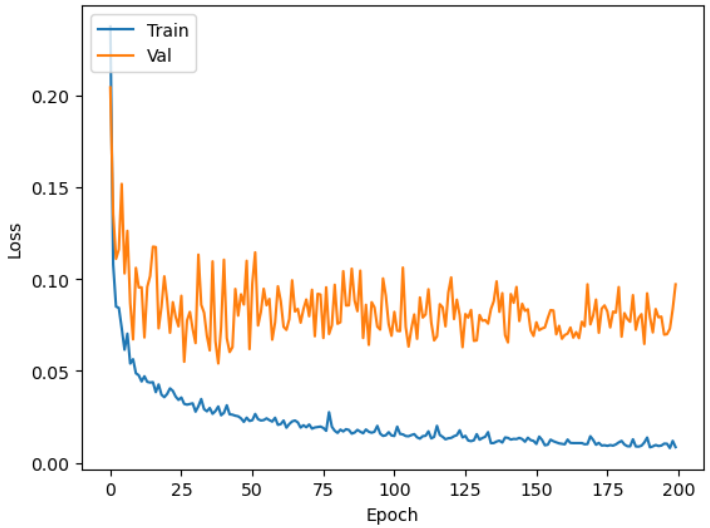


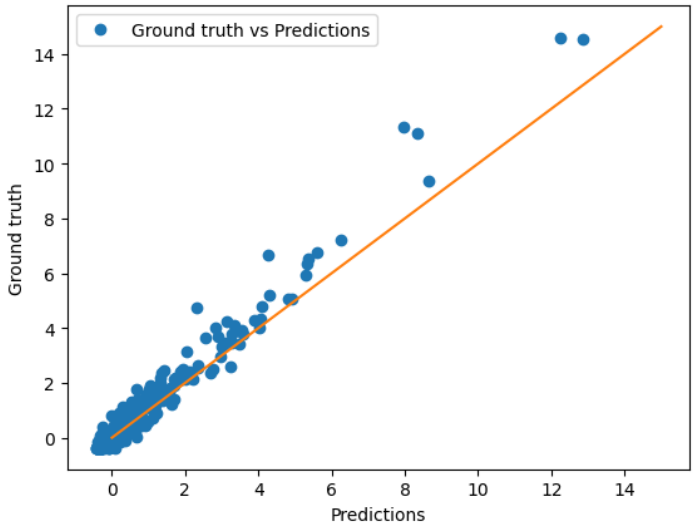


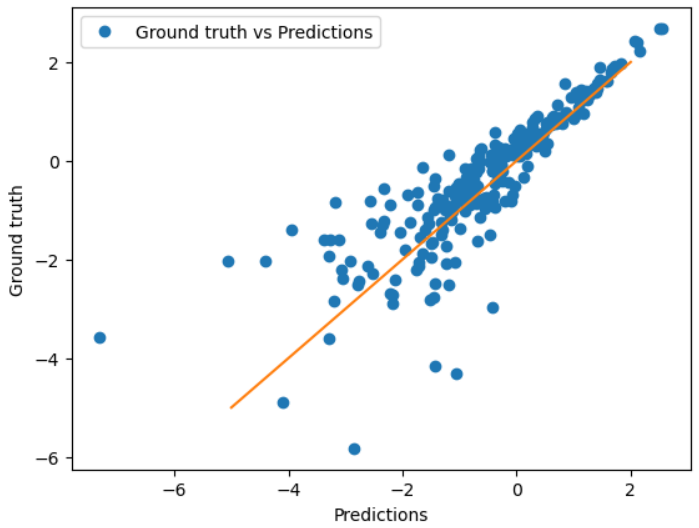
1회	2회	3회	평균
0.1909	0.1859	0.0751	0.1506

3 모델 결과

최종 결과







1회	2회	3회	평균
0.0709	0.0819	0.0711	0.0746

3 모델 결과

중소벤처기업부

2019년 보도 자료

2017년 월 평균 영업이익:

2,690,000

&

2017년 영업이익률: **15.8%**

2017년 월 평균 영업이익: **2,690,000**

2017년 대비 2022년 물가상승률: **10.33%**

⟶ 2,690,000 X 1.1033 = **2,967,877**

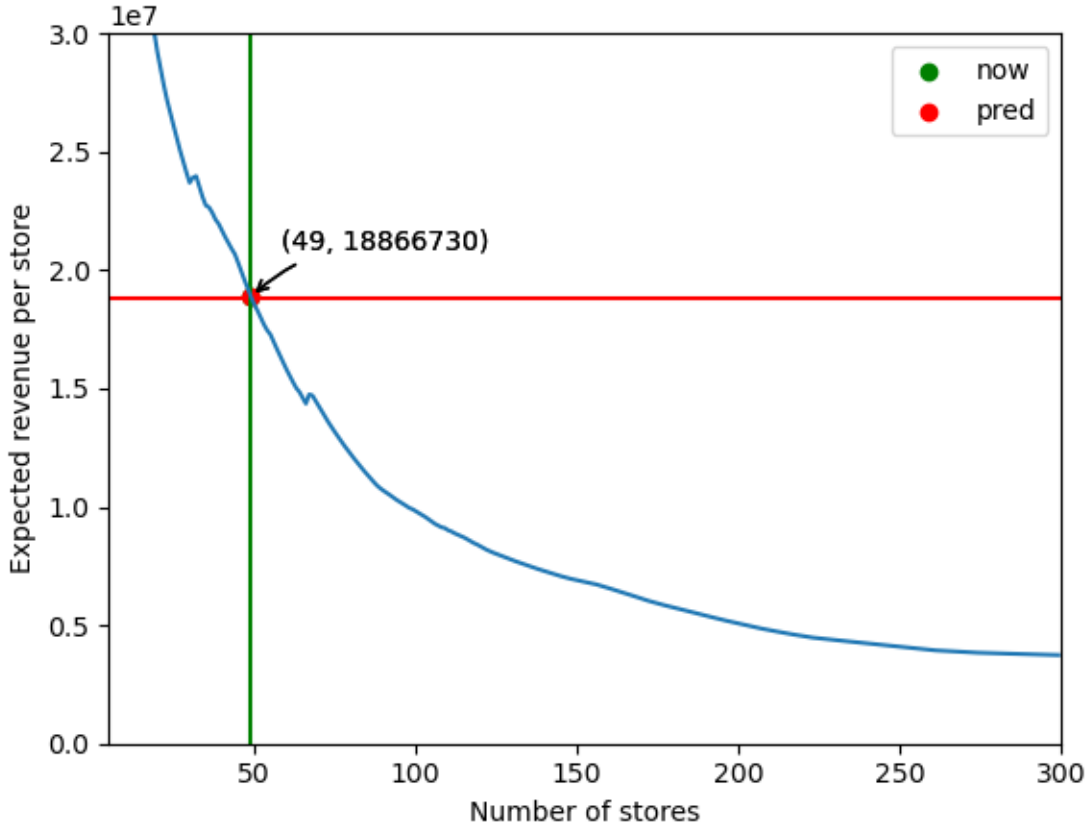
매출액(기준) = 영업이익 X 100% / 영업이익률(%)

= **2,967,877** X 100% / **15.8%**

= **18,784,031**

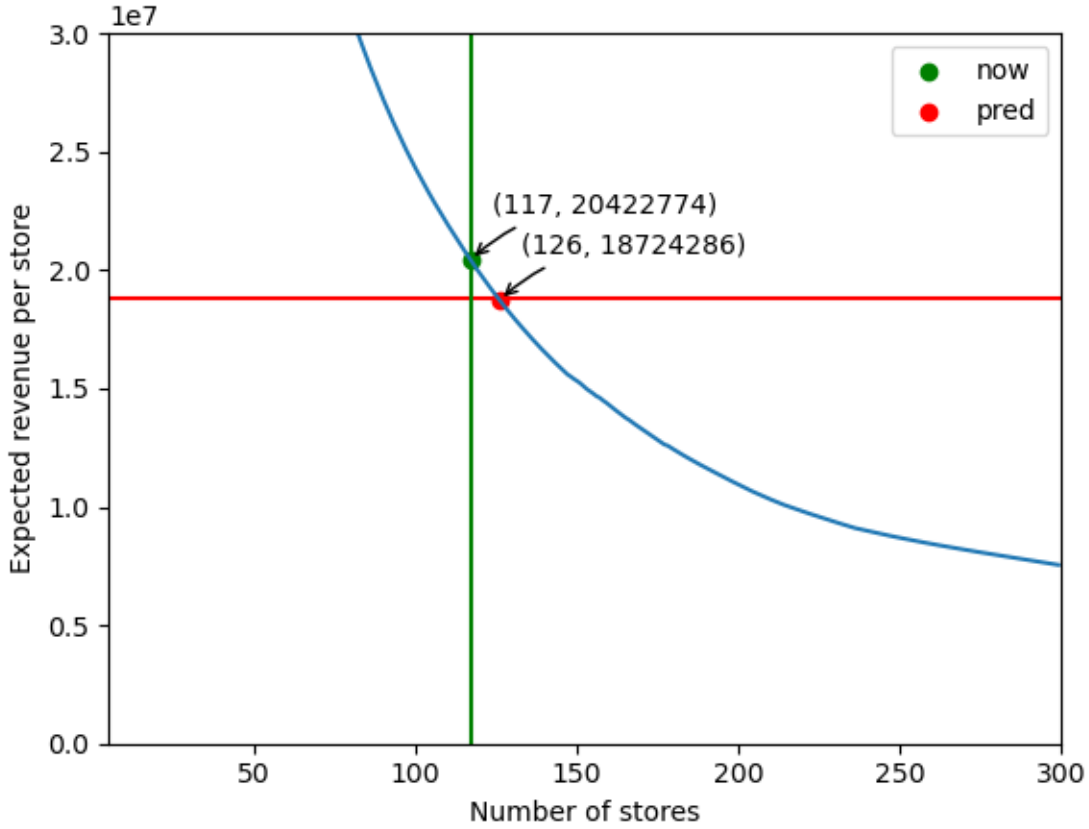
3 모델 결과

```
IMG(130) # 성수동카페거리; 2110131; 골목상권
```



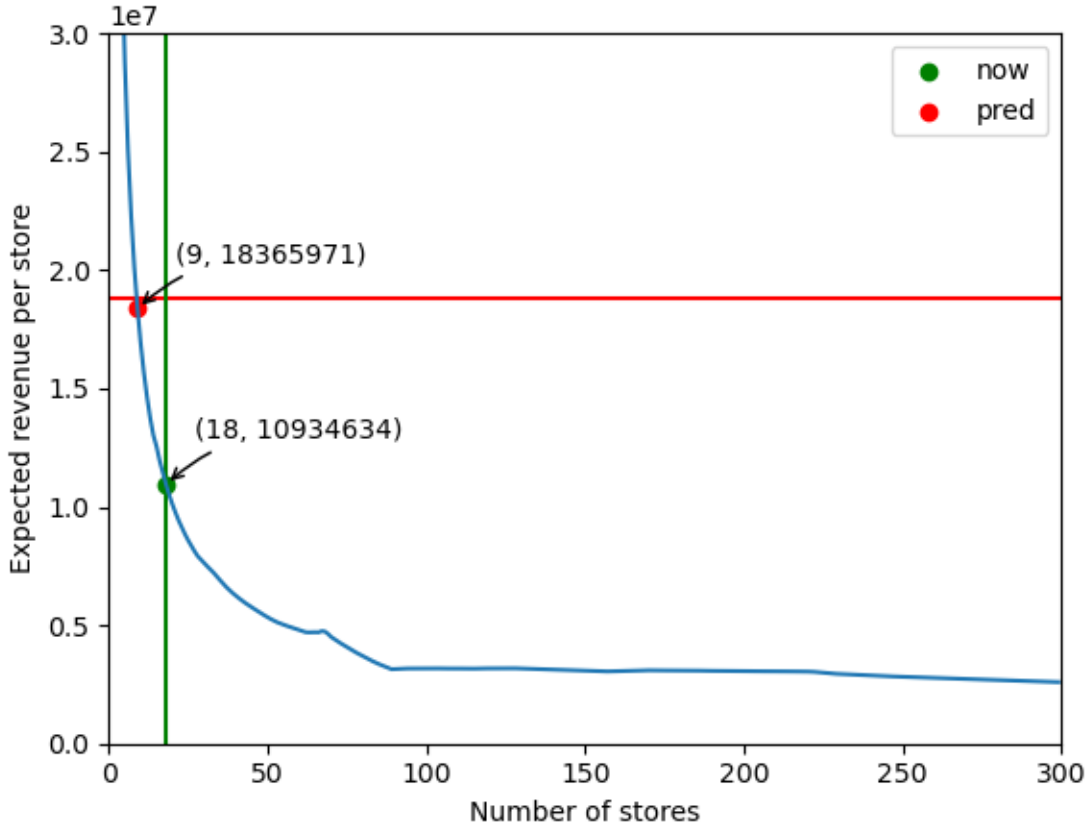
3 모델 결과

```
IMG(1275) # 가로수길; 2120186; 발달상권
```



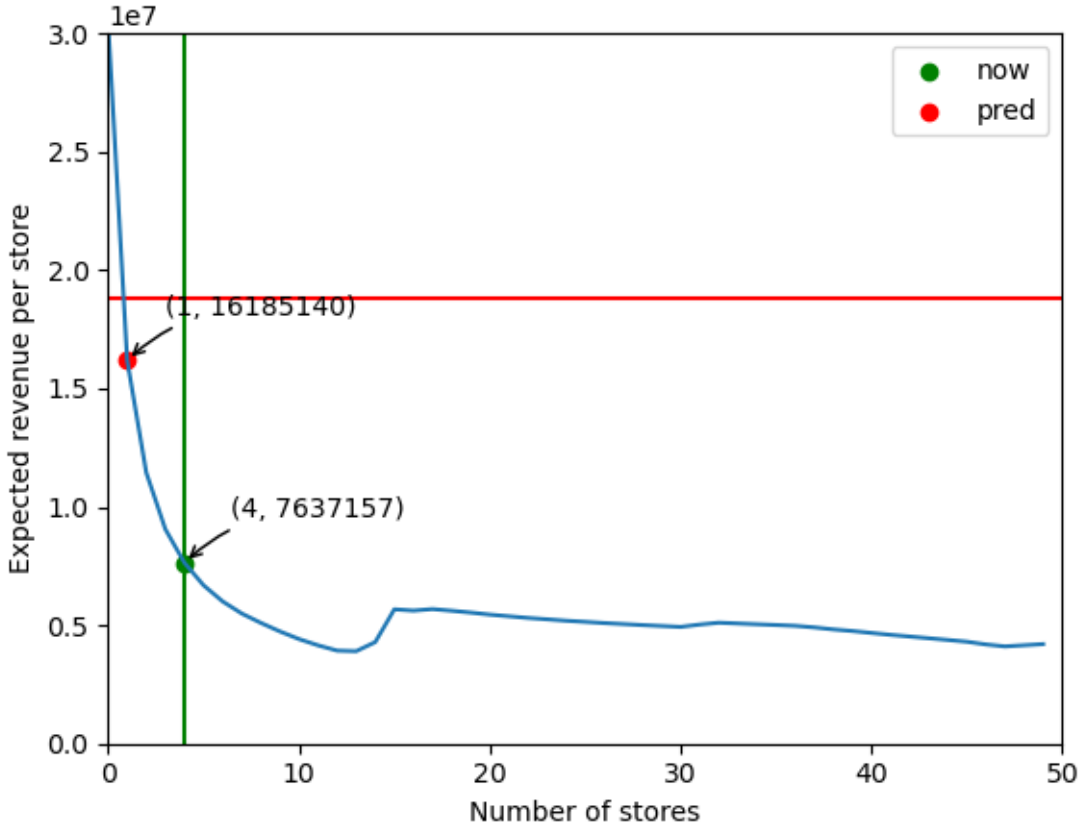
3 모델 결과

```
IMG(1102, 0) # 송파나루역 1번(송리단길); 2111013; 골목상권
```

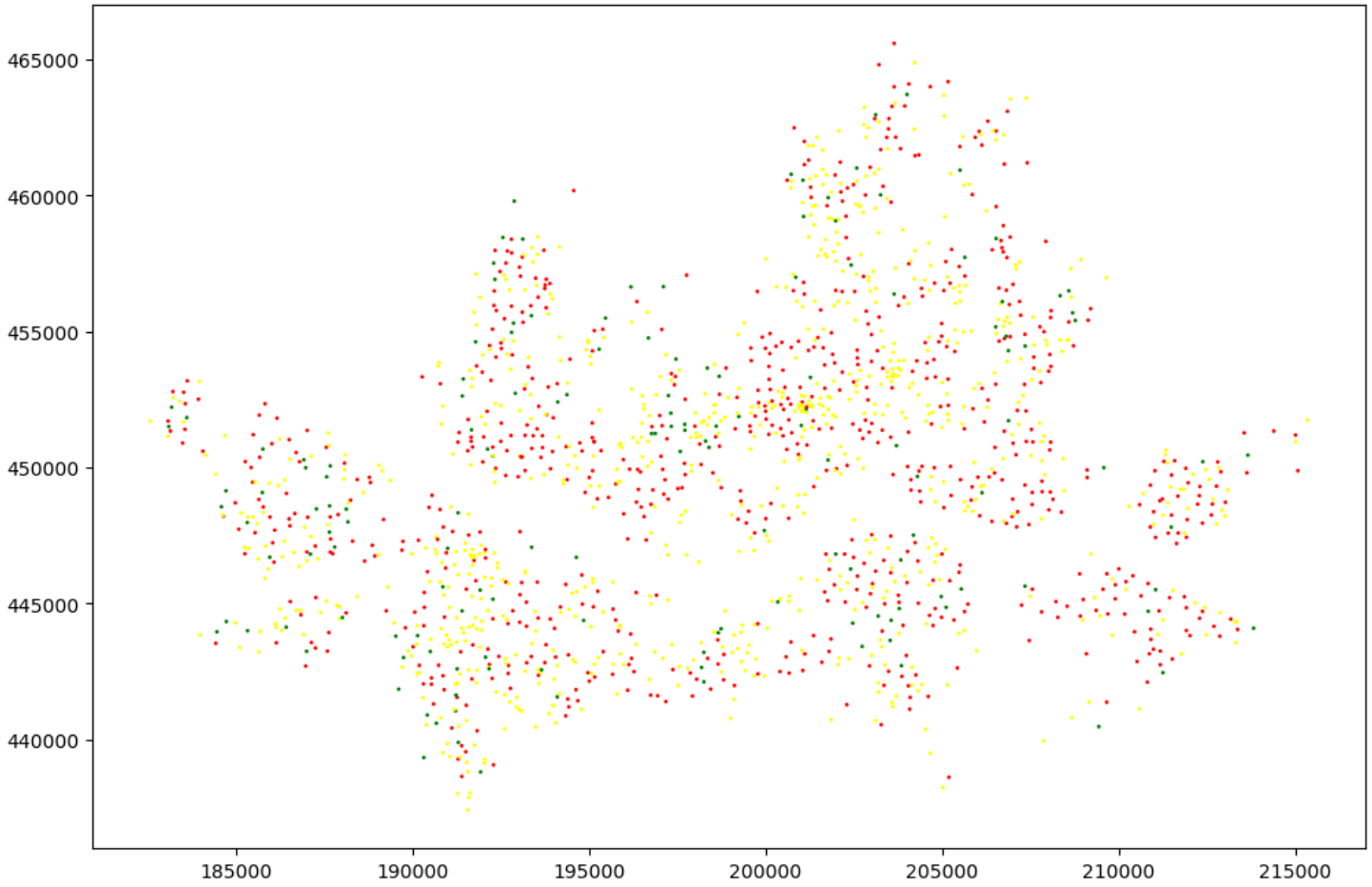


3 모델 결과

```
IMG(1, 0, 50) # 2110002;
```



3 모델 결과



경청해주셔서 감사합니다☺

Whiskey조
화학공학과 22 송지윤
환경공학부 20 양현주
인공지능학과 21 이상민
수학과 19 황태연