2019년

데이터 사이언스

팀프로젝트

PROJECT.

지하철 이용과 교통사고 관계 분석

20182141 정현우

20182143 채현식

20182100 남정호

구분	내용
1.도입	가설 설정 배경
2.가설	가설 설명 가설로 인한 기대효과
3.이용자료 및 설명	데이터 이름 및 출처 데이터 관련 설명
4.분석 방법 및 결과	데이터 분석 과정 데이터 분석 결과
5.결론	최종결론 및 논의사항 향후 계획

지하철 이용과 교통사고 관계 분석

1. 도입

- 뉴스, 신문, 등 다양한 언론 매체와 이를 접한 대중들이 과거나 현재에나 심각한 문제로 받아들이고 있는 것이 있습니다. 바로, 교통사고입니다. 교통사고는 과속, 음주, 역주행, 등 다양한 원인으로 인해 본인의 과실이 아니라고 할지라도 언제든지 누구에게나 발생할 수 있는 사고입니다. 그것은 스포츠 스타에게도 예외는 아니었습니다. 최근, '호세 안토니오 레예스' 선수는 제한 속도 120km인 고속도로에서 시속 220km 이상의 속도로 질주하다 벽에 부딪히며 90m 이상을 튕겨 날아가게 되었고 결국 사망에 이르게 되었습니다. 이에 저희 팀은 교통사고 대다수가 도로 위에서 발생한다는 점에 집중하였고 평소에 도로 위를 움직이는 자가용 자동차나 대중교통인 버스를 이용하기보다는 도로 위가 아닌 철로 위에서 움직이는 대중교통인 지하철을 이용하게 된다면 교통사고의 빈도수가 낮아지지 않을까 라는 생각을 해보게 되었습니다.

<출처 : http://www.osen.co.kr/article/G1111160776>

2. 가설

- 도입부에서 언급하였듯이 평소에 도로 위를 움직이는 자가용 자동차나 대중교통인 버스를 이용하기보다는 도로 위가 아닌 철로 위에서 움직이는 대중교통인 지하철을 이용하게 된다면 교통사고의 빈도수가 낮아지지 않을까 라는 생각을 하였고 이에 저희 팀은 '인구대비 지하철 이용률이 높은 도시에서는 인구대비 교통사고 발생률이 낮을 것이다.'라는 가설을 세웠습니다.
- 이 가설이 틀리지 않다는 것을 증명한다면 지하철을 보다 활성화하는 것이 교통사고 발생 건수를 줄이는데 기여한다는 것을 기대할 수 있습니다.

3. 이용자료

- 데이터 명 : 2017_시도_시군구별_연령층별_교통사고.csv

  데이터 출처 : 공공데이터포탈

  https://www.data.go.kr/dataset/3038489/fileData.do

  데이터 형식 : .csv

  데이터 사용처 : 시도별 교통사고의 발생건수 파악 (data1)

- 데이터 명 : 시도별_대중교통이용횟수_1주_201906066154146.csv

  데이터 출처 : 통계청

  http://kosis.kr/statHtml/statHtml.do?orgId=116&tblId=DT_MLTM_5719&conn_path=I2

  데이터 형식 : .csv

  데이터 사용처 : 시도별 대중교통 이용 인구수 중 지하철 이용 인구수 파악 (data2)

- 데이터 명 : 대중교통_이용자유형별_이용인원_2017.csv

  데이터 출처 : 공공데이터포탈

  https://www.data.go.kr/dataset/3078089/fileData.do

  데이터 형식 : .csv

  데이터 사용처 : 시도별 대중교통 이용 인구수 파악 (data3)

- 데이터 명 : 201712_201712_주민등록인구및세대현황_월간.csv

  데이터 출처 : 공공데이터포탈

  https://www.data.go.kr/dataset/3033255/fileData.do

  데이터 형식 : .csv

  데이터 사용처 : 시도별 총 인구수 파악 (data4)

4. 분석방법 및 결과

1) 데이터를 가공하고 분석하는데 사용할 모듈들을 불러옵니다.

```
[50] from datascience import *
     import numpy as np
     import pandas as pd
     import matplotlib
     matplotlib.use('Agg', warn=False)
     %matplotlib inline
     import matplotlib.pyplot as plots
     plots.style.use('fivethirtyeight')
     # 일단 import
     %matplotlib inline
     #matplotlib에서 한글 사용법 출처
     #https://colab.research.google.com/github/nicewook/datascience_exercise/blob/master/korean_font_on_matplotlib.ipynb#scrollTo=xB0e99-Wy6k0
     import matplotlib.font_manager as fm  # 폰트 관련 용도
```

2) 가설이 틀리지 않다는 것을 증명하기 위해서 필요한 데이터는 인구대비 지하철 이용률과 인구대비 교통사고 발생률입니다. 우선, 인구대비 교통사고 발생률을 구해야합니다. 그러기 위해서 필요한 데이터는 시도별 교통사고 발생건수에 대한 데이터와 시도별 총 인구수에 대한 데이터입니다. 그 중 시도별 교통사고 발생 건수에 대한 데이터를 trafficAcident2017에 넣어둡니다.

```
[5]  #https://www.data.go.kr/dataset/3038489/fileData.do
     #2017년_시도_시군구별_연령층별_교통사고.csv
     #해당 데이터를 시도별 교통사고 발생건수로 사용하기 위해 시도를 기준으로 groupby한 후
     #군구별로 구분 되어 있는 데이터를 sum으로 합쳐줌
     trafficAcident2017=pd.read_csv("20182143-data1.csv",encoding='euc-kr')
     trafficAcident2017=trafficAcident2017.groupby("시도").sum()[["발생건수"]]
     trafficAcident2017
```

시도	발생건수
강원	8316
경기	50627
경남	11742
경북	13896
광주	7499
대구	12970
대전	7767
부산	11753
서울	38625
세종	746
울산	4265
인천	7719
전남	9770
전북	7748
제주	4378
충남	9241
충북	9273

3) 다음으로 시도별 지하철 이용 인구수를 구해야합니다. 그러기 위해서 시도별 대중교통 이용 인구수를 활용하는데 대중교통 이용인구수 중 지하철 이용인구수 퍼센트에 대한 데이터를 publicTransit2017에 넣어둡니다.

```
[6]  #http://kosis.kr/statHtml/statHtml.do?orgId=116&tblId=DT_MLTM_5719&conn_path=I2
     #시도별_대중교통이용횟수_1주_20190606154146.csv
     #시도별 대중교통 이용하는 사람들중에서 대중교통별 이용하는 인구 퍼센트로 이용하기 위해 따로 데이터를 가공할 필요는 없으나
     #데이터 활용의 편의성을 위해 각각의 column을 rename시켜줌
     publicTransit2017=pd.read_csv("20182143-data2.csv", encoding="euc-kr")
     #encoding="enu-kr"은 유니코드를 한국말로 읽어오기 위해 활용
     publicTransit2017=publicTransit2017[["구분(1)","2017","2017.1"]].rename(columns={"구분(1)":"도시","2017":"시내버스","2017.1":"지하철"}).drop([0,1])
     publicTransit2017=publicTransit2017.set_index("도시")
     publicTransit2017
```

도시	시내버스	지하철
서울	32.9	67.1
부산	59.2	40.8
인천	54.1	45.9
대구	62.1	37.9
대전	79.9	20.1
광주	87.0	13.0
울산	98.0	2.0
경기	62.3	37.7
강원	98.2	1.8
충북	100.0	0.0
충남	94.2	5.8
전북	100.0	0.0
전남	100.0	0.0
경북	96.1	3.9
경남	96.2	3.8
제주	100.0	0.0
세종	100.0	0.0

4) 앞서 언급했듯이 시도별 지하철 이용 인구수를 구해야합니다. 시도별 대중교통 이용 인구수를 활용할 것이기 때문에 시도별 대중교통 이용 인구수에 대한 데이터를 psUser2017에 넣어둡니다.

```
[7]  #https://www.data.go.kr/dataset/3078089/fileData.do
     #대중교통_이용자유형별_이용인원_2017_.csv
     #시도별 대중교통 이용 인구수로 이용하기 위해 시도별로 groupby하려 했으나
     #,와 -가 문자열로 취급되기 때문에 각각의 column을 다른 배열에 저장하고
     #replace를 이용하여 ,는 삭제하고 -는 0으로 처리한 후 sum하여 시도별 대중교통 이용 인구수 데이터를 만들어냄
     psUser2017=pd.read_csv("20182143-data3.csv", encoding="euc-kr")
     psUser2017=psUser2017.rename(columns={"구분":"도시"}).set_index("도시")

     #각각의 column을 다른 배열에 저장
     person=psUser2017["일반인"]
     child=psUser2017["어린이"]
     teenager=psUser2017["청소년"]
     senior=psUser2017["기타"]
     #rowNum은 다 같으니 일반인으로 count
     rowNum=psUser2017["일반인"].count()

     #,를 제거한 뒤 저장할 배열 생성
     personNoComma=make_array()
     teenagerNoComma=make_array()
     childNoComma=make_array()
     seniorNoComma=make_array()
     sum=make_array()

     #replace를 이용하여 ,가 발견될 시 이를 제거(rowNum횟수만큼 반복)
     for i in np.arange(rowNum):
       personNoComma=np.append(personNoComma, person[i].replace(",",""))
       childNoComma=np.append(childNoComma, child[i].replace(",",""))
       teenagerNoComma=np.append(teenagerNoComma, teenager[i].replace(",",""))
       seniorNoComma=np.append(seniorNoComma,senior[i].replace(",","").replace("-","0"))

     #분류된 각각의 column의 값을 sum 배열에 저장(rowNum횟수만큼 반복)
     for i in np.arange(rowNum):
       sum=np.append(sum, (int)(personNoComma[i])+(int)(childNoComma[i])+(int)(teenagerNoComma[i])+(int)(seniorNoComma[i]))

     psUser2017=pd.DataFrame(sum,index=psUser2017.index, columns=["대중교통 이용 인구수"])
     psUser2017
```

도시	대중교통 이용 인구수
서울	4423933.0
부산	952394.0
대구	547568.0
인천	775463.0
광주	245970.0
대전	272079.0
울산	152499.0
세종	22677.0
경기	2981638.0
강원	85734.0
충북	117371.0
충남	148613.0
전북	118447.0
전남	120123.0
경북	165815.0
경남	253772.0
제주	63008.0

5) psUser2017에 넣어둔 시도별 대중교통 이용 인구수 데이터와 publicTransit2017에 넣어둔 대중교통 이용인구수 중 지하철 이용인구수 퍼센트 데이터를 이용하여 구하고자했던 시도별 지하철 이용 인구수에 대한 데이터를 subwayPopulation2017에 넣어둡니다.

```
#시도별 지하철 이용 인구수로 이용하기 위해
#위에서 구한 시도별 대중교통 이용 인구수와 시도별 대중교통별 이용 인구수 퍼센트를 곱함

#인구수는 소수가 아니므로 반올림
pop=make_array()
#join을 이용하여 행을 맞춤
subwayPopulation2017=psUser2017.join(publicTransit2017)
for i in np.arange(rowNum):
  pop=np.append(pop,np.round(subwayPopulation2017["대중교통 이용 인구수"][i]*(float)(subwayPopulation2017["지하철"][i])/100,0))

subwayPopulation2017=pd.DataFrame(pop,index=psUser2017.index, columns=["지하철 이용 인구수"])
subwayPopulation2017
```

도시	지하철 이용 인구수
서울	2968459.0
부산	388577.0
대구	207528.0
인천	355938.0
광주	31976.0
대전	54688.0
울산	3050.0
세종	0.0
경기	1124078.0
강원	1543.0
충북	0.0
충남	8620.0
전북	0.0
전남	0.0
경북	6467.0
경남	9643.0
제주	0.0

6) 마지막으로 필요한 데이터는 시도별 총 인구수입니다. 시도별 총 인구수에 대한 데이터를 cityPoplulation2017에 넣어둡니다.

[9]
```
#https://www.data.go.kr/dataset/3033255/fileData.do
#201712_201712_주민등록인구및세대현황_월간.csv
#시도별 총인구수로 이용하기 위해 전국에 대한 데이터를 drop시키고
#,는 문자열로 취급되기 때문에 replace를 이용하여 제거해준다
totalPop=make_array()
cityPopulation2017=pd.read_csv("20182143-data4.csv", encoding="euc-kr")
#불필요한 데이터를 drop
cityPopulation2017=cityPopulation2017.drop(0)

#replace를 이용하여 ,제거(rowNum의 횟수만큼)
for i in np.arange(rowNum):
  totalPop=np.append(totalPop, (int)(cityPopulation2017["2017년12월_총인구수"].iloc[i].replace(",","")))

cityPopulation2017=pd.DataFrame(totalPop, index=psUser2017.index, columns=["2017년 총 인구수"])
cityPopulation2017
```

도시	2017년 총 인구수
서울	9857426.0
부산	3470653.0
대구	2475231.0
인천	2948542.0
광주	1463770.0
대전	1502227.0
울산	1165132.0
세종	280100.0
경기	12873895.0
강원	1550142.0
충북	1594432.0
충남	2116770.0
전북	1854607.0
전남	1896424.0
경북	2691706.0
경남	3380404.0
제주	657083.0

7) 가설이 틀리지 않았다는 것을 증명하기 위해서 필요한 데이터인 인구대비 지하철 이용률과 인구대비 교통사고 발생률에 대한 데이터를 모두 구하였습니다. 인구대비 지하철 이용률은 subwayPopulation2017에 넣어둔 시도별 지하철 이용 인구수에 대한 데이터와 cityPoplulation2017에 넣어둔 시도별 총 인구수의 나눗셈 연산으로 구할 수 있습니다. 또한, 인구대비 교통사고 발생률은 trafficAcident2017에 넣어둔 시도별 교통사고 발생 건수에 대한 데이터와 cityPoplulation2017에 넣어둔 시도별 총 인구수의 나눗셈 연산으로 구할 수 있습니다.

```
[7]  #인구수 대비 지하철 이용비율과 인구수 대비 사고 비율로 이용하기 위해
     #join을 이용하여 하나의 dataframe에 데이터를 취합함
     proportion2017=cityPopulation2017.join([subwayPopulation2017,trafficAcident2017])
     proportion2017["인구 대비 지하철 이용 인구수"]=proportion2017["지하철 이용 인구수"]/proportion2017["2017년 총 인구수"]
     proportion2017["인구 대비 사고 발생 건수"]=proportion2017["발생건수"]/proportion2017["2017년 총 인구수"]
     proportion2017
```

도시	2017년 총 인구수	지하철 이용 인구수	발생건수	인구 대비 지하철 이용 인구수	인구 대비 사고 발생 건수
서울	9857426.0	2968459.0	38625	0.301139	0.003918
부산	3470653.0	388577.0	11753	0.111961	0.003386
대구	2475231.0	207528.0	12970	0.083842	0.005240
인천	2948542.0	355938.0	7719	0.120717	0.002618
광주	1463770.0	31976.0	7499	0.021845	0.005123
대전	1502227.0	54688.0	7767	0.036405	0.005170
울산	1165132.0	3050.0	4265	0.002618	0.003661
세종	280100.0	0.0	746	0.000000	0.002663
경기	12873895.0	1124078.0	50627	0.087315	0.003933
강원	1550142.0	1543.0	8316	0.000995	0.005365
충북	1594432.0	0.0	9273	0.000000	0.005816
충남	2116770.0	8620.0	9241	0.004072	0.004366
전북	1854607.0	0.0	7748	0.000000	0.004178
전남	1896424.0	0.0	9770	0.000000	0.005152
경북	2691706.0	6467.0	13896	0.002403	0.005163
경남	3380404.0	9643.0	11742	0.002853	0.003474
제주	657083.0	0.0	4378	0.000000	0.006663

8) 각각의 데이터에 대한 스케일이 제각각이기 때문에 정규화를 통해 각각의 데이터에 대한 스케일을 맞춰줍니다. 또한, 정규화에 대한 결과가 음수로 나올 경우 분석과정에서 비교가 어려움으로 각각의 데이터의 최솟값의 절댓값을 추가로 더해주어 해당 경우가 발생하지 않도록 하였습니다.

```
[57]  #스케일을 맞춰주기 위해 정규화를 진행 (Z=(X-μ)/σ)
      def normalization(array):
        retArray=make_array()
        #배열의 크기
        arrSize=array.size
        #배열의 평균
        arrMean=np.mean(array)
        #배열의 표준편차
        arrStd=np.std(array)
        #배열의 크기만큼 정규화 진행
        for i in np.arange(arrSize):
          z=(array[i]-arrMean)/arrStd
          retArray=np.append(retArray, z)
        return retArray
```

```
[9]  #스케일을 맞춰주기 위해 정규화를 진행
     #음수가 나오면 비교가 어려우므로 최솟값의 절댓값을 더함
     temp=proportion2017[["인구 대비 지하철 이용 인구수","인구 대비 사고 발생 건수"]]
     popZ=normalization(temp["인구 대비 지하철 이용 인구수"])
     popZ=popZ+np.abs(popZ.min())
     accidentZ=normalization(temp["인구 대비 사고 발생 건수"])
     accidentZ=accidentZ+np.abs(accidentZ.min())

     #barh를 편하게 쓰기위해 데이터사이언스의 Table로 만들어 준다.
     proportionZ2017=Table().with_columns("도시",psUser2017.index,"2017 인구 대비 지하철 이용 인구수 비율의 Z",popZ, "2017 인구 대비 사고 발생 건수 비율의 Z",accidentZ)
     #barh로 표현했을때 눈에 보기 쉽게 sorting를 진행
     proportionZ2017=proportionZ2017.sort("2017 인구 대비 지하철 이용 인구수 비율의 Z", descending=True)
     proportionZ2017.show()
```

도시	2017 인구 대비 지하철 이용 인구수 비율의 Z	2017 인구 대비 사고 발생 건수 비율의 Z
서울	3.94916	1.19658
인천	1.58308	0
부산	1.46826	0.707104
경기	1.14505	1.20962
대구	1.09951	2.41257
대전	0.477412	2.34854
광주	0.286476	2.30506
충남	0.0534036	1.6081
경남	0.0374094	0.787297
울산	0.034329	0.959342
경북	0.0315074	2.34136
강원	0.0130536	2.52736
제주	0	3.72178
전남	0	2.33149
전북	0	1.43521
충북	0	2.94251
세종	0	0.0418015

9) 데이터를 가공하여 얻은 결과를 바탕으로 분석하기 위해 barh그래프를 활용하고자 하였으나 한글이 깨지는 문제점이 발생하여 matplotlib에서 폰트를 따로 다운로드하여 적용시킴으로서 해당 문제점을 해결하였습니다.

```
[59]  #matplotlib에서 한글이 깨져서 따로 다운로드 해서 적용
      #한글 나눔 고딕 인스톨
      !apt-get update -qq
      !apt-get install fonts-nanum* -qq
```

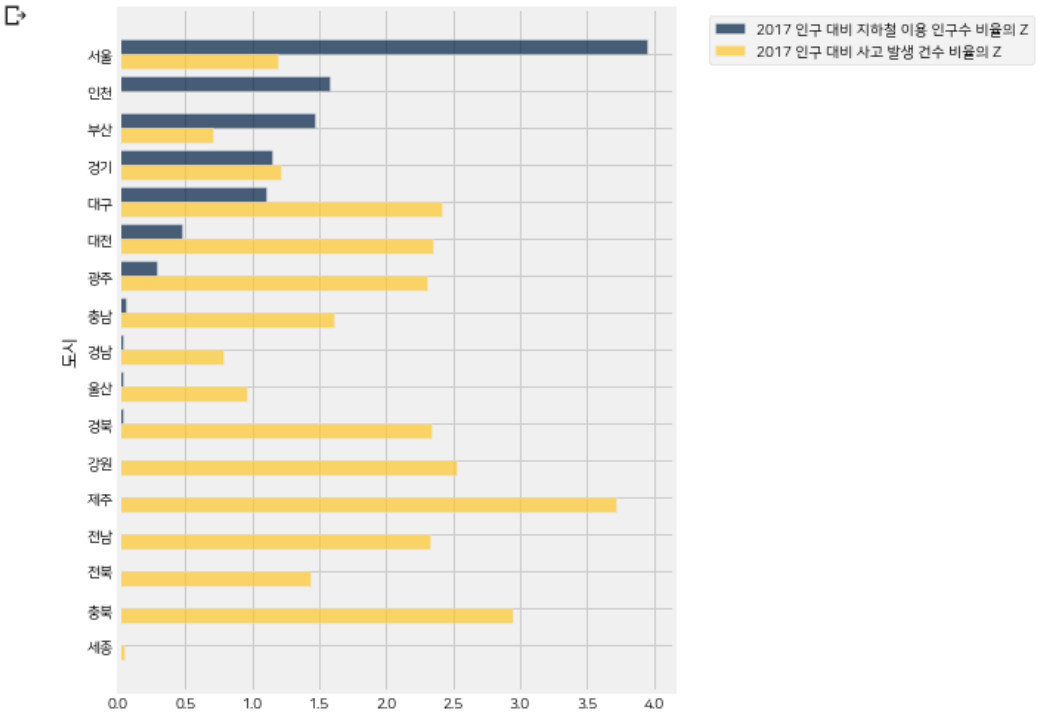
```
[60]  #설치된 나눔폰트들을 출력
      sys_font=fm.findSystemFonts()
      nanum_font = [f for f in sys_font if 'Nanum' in f]
      nanum_font
```

```
['/usr/share/fonts/truetype/nanum/NanumBarunGothic.ttf',
 '/usr/share/fonts/truetype/nanum/NanumGothicCoding-Bold.ttf',
 '/usr/share/fonts/truetype/nanum/NanumSquareEB.ttf',
 '/usr/share/fonts/truetype/nanum/NanumGothicExtraBold.ttf',
 '/usr/share/fonts/truetype/nanum/NanumSquareL.ttf',
 '/usr/share/fonts/truetype/nanum/NanumMyeongjoExtraBold.ttf',
 '/usr/share/fonts/truetype/nanum/NanumMyeongjo.ttf',
 '/usr/share/fonts/truetype/nanum/NanumMyeongjoBold.ttf',
 '/usr/share/fonts/truetype/nanum/NanumBrush.ttf',
 '/usr/share/fonts/truetype/nanum/NanumGothicEcoBold.ttf',
 '/usr/share/fonts/truetype/nanum/NanumPen.ttf',
 '/usr/share/fonts/truetype/nanum/NanumGothicCoding.ttf',
 '/usr/share/fonts/truetype/nanum/NanumBarunGothicBold.ttf',
 '/usr/share/fonts/truetype/nanum/NanumBarunGothicLight.ttf',
 '/usr/share/fonts/truetype/nanum/NanumMyeongjoEco.ttf',
 '/usr/share/fonts/truetype/nanum/NanumBarunpenR.ttf',
 '/usr/share/fonts/truetype/nanum/NanumGothicEcoExtraBold.ttf',
 '/usr/share/fonts/truetype/nanum/NanumGothic.ttf',
 '/usr/share/fonts/truetype/nanum/NanumSquareRoundR.ttf',
 '/usr/share/fonts/truetype/nanum/NanumGothicEco.ttf',
 '/usr/share/fonts/truetype/nanum/NanumSquareRoundEB.ttf',
 '/usr/share/fonts/truetype/nanum/NanumSquareB.ttf',
 '/usr/share/fonts/truetype/nanum/NanumGothicLight.ttf',
 '/usr/share/fonts/truetype/nanum/NanumBarunpenB.ttf',
 '/usr/share/fonts/truetype/nanum/NanumSquareRoundL.ttf',
 '/usr/share/fonts/truetype/nanum/NanumSquareR.ttf',
 '/usr/share/fonts/truetype/nanum/NanumBarunGothicUltraLight.ttf',
 '/usr/share/fonts/truetype/nanum/NanumMyeongjoEcoExtraBold.ttf',
 '/usr/share/fonts/truetype/nanum/NanumGothicBold.ttf',
 '/usr/share/fonts/truetype/nanum/NanumSquareRoundB.ttf',
 '/usr/share/fonts/truetype/nanum/NanumMyeongjoEcoBold.ttf']
```

```
[61]  #이들중 한가지를 선택하여 사용
      path = '/usr/share/fonts/truetype/nanum/NanumSquareRoundB.ttf'
      font_name = fm.FontProperties(fname=path, size=10).get_name()
      plots.rc('font', family=font_name)
      fm._rebuild()
```

10) 데이터를 가공하여 얻은 결과를 barh그래프로 표현하여 비교해보았습니다.

```
[13]  #barh를 이용하여 그래프를 그린다.
      proportionZ2017.barh(0)
```



5. 결론

- 분석 결과 얻게 된 표와 barh 그래프를 살펴보았을 때 저희 팀이 세운 가설인 '인구대비 지하철 이용률이 높은 도시에서는 인구대비 교통사고 발생률이 낮을 것이다.'는 채택될 수 있다는 결론을 내렸습니다.
- 그러나 대구와 같이 이례적인 지역이 존재하였습니다. 이에 대해 저희 팀이 논의해본 결과 가설과 분석과정에서 교통사고 발생 건수에 영향을 줄 수 있는 여러 가지 요인들을 고려하지 못하였기 때문이라고 생각하였습니다. 따라서, 교통사고 발생 건수에도 교통량, 도로의 제한속도, 도로의 차선 개수, 등과 같은 요인들을 고려해야 하였다고 논의하였습니다.
- 이에 향후 팀 프로젝트에서는 논의 결과를 고려하여 각 시도별 지하철이 생기기 전과 후를 비교하되 부가적인 원인이 될 수 있는 교통량, 인구수 같은 것들도 고려하여 계획을 세워야 한다고 결론을 내렸습니다.