

Maryland 교통사고 데이터 분석

빅데이터플랫폼
곽민지

1.preprocessing

1. Plan

- 자의적으로 가중치를 정해서 INJURY 변수를 정하고, 이것의 평균값에 따라 군집화를 여러 번 반복함으로써, INJURY값을 되도록이면 잘 나누는 경우를 만들도록 군집화의 변수들을 선정한다.
- 교통 사고의 경우 특정 도로 상황 (도로 인프라, 환경 등)에서 특정 돌발 변수 (충돌 방법, 충돌 위치, 당시의 차량 움직임)가 만날 때 이루어진다. 따라서 상황변수와 돌발변수를 선정하여 각각에 대해 2회의 클러스터링을 실시한다.
- 각 클러스터링 결과를 가지고 decision tree를 그려 유형 분류의 feature를 분석한 후
- Visulization 및 척도 분석을 통해 각 사고 유형의 frequency와 INJURY 정도를 파악하여 정책 대책을 수립한다.

2. Def return_df(df):

- 불필요한 id 데이터 삭제, person type은 D로 동일하므로 삭제 `df = df.drop([' PERSON_TYPE ' , ' PERSON_ID ' , ' VEHICLE_ID '], axis=1)`
- area_direction 처리: 1시~12시 방향이던 변수를 앞,뒤,좌,우 방향인 변수로 변환하여, 개수를 줄였다.
- INJURY column 생성: INJ_SEVER_CODE (범주형) 에 따라 가중치를 부여하였다. No injury = 0점 / Fatal injury=30점 / 나머지 부상 = 15점을 부여하였다.
- Time->month로 변환하였다. Time 정보 중 month만을 추출하여 MONTH 컬럼에 추가하였고, hour등의 변수는 그 시간대의 통행량과 사고 발생량이 비례한다는 사실은 너무나 상식적이기 때문에 불필요하다고 생각하여 제외하였다.
- MONTH와 AGE는 계절별, 연령대별로 그룹화해서 간단하게 정리하였다.

3. Def preprocess(df, col_list):

- 숫자 데이터를 제외한 나머지 데이터를 더미화한후, Y/N 단 두가지로 나뉘지는 경우에는 N케이스는 삭제하였다(Y케이스와 동일한 상관관계)
- 숫자 데이터(AGE,MONTH,INJURY)는 정규화하였다.
- col_list에 해당하는 column들만 df에서 추출하되, MONTH, AGE, INJURY는 항상 포함하도록 리턴한다.

4. Def k_means(p_df):

- p_df에 해당하는 레코드들을 k-means clustering하고 clustering 결과 label 컬럼을 p_df에 붙여서 리턴함

5. Def show_analysis(result_df):

- k_means 결과 리턴된 df를 result_df로 받아 각 kmeans_label에 따라 groupby 하여 각 컬럼들의 mean값을 구해서 df로 리턴하는 함수.

2. 1차 클러스터링

1차 클러스터링 - 상황 변수들 :

' JUNCTION_CODE ', ' BODY_TYPE_CODE ', ' RD_COND_CODE ', ' RD_DIV_CODE ', ' SURF_COI

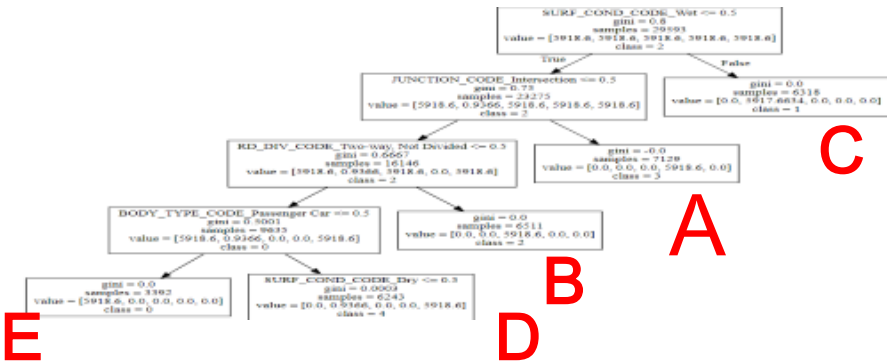
1) Accident table : 교차로 분기점 정보, 도로 상태 정보, 도로 분기 정보, 표면 상태 정보

2) Vehicle table: 차량 종류

(3) 제외 변수 lane_code, light_code, weather_code : 추가로 포함시켜 클러스터링했을 때 보다 이 세 변수를 제외했을 시 클러스터링하면 군집 간의 INJURY 값의 차이가 확연하였다. 즉, 이 세 변수들은 INJURY 값에 따른 분류를 더 어렵게 하므로 제외하였다.

* 위에서 선정한 변수 + MONTH + AGE + INJURY 를 가지고 클러스터링을 실시, 총 5개 군집으로 분류하였다.

* Cluster 번호를 y로, 나머지 변수들을 X로 지정하여 decision tree 분석 결과 surface condition -> junction code -> road division ->body type 순으로 분류되었다



	A	B	C	D	E
INJURY의 평균	0.098471	0.095454	0.087909	0.075857	0.068249
Surface Condition	대부분 dry	대부분 dry	99% Wet	대부분 dry	대부분 dry
Junction	Intersection				
Road division		Two-way,not divided			
Body type				Passenger car	
대분류 기준 요약	Surface dry, Intersection의 사고	Surface dry, Non-intersection two-way, not divided	99 % Surface wet	Surface dry, Passenger car	Surface dry+ 주로 Non-intersection two-way,not divided가 아 니며 passenger car가 아 님

3. 2차 클러스터링

2차 클러스터링 – 돌발 변수들 :

'COLLISION_TYPE_CODE', 'AREA_DAMAGED_CODE_MAIN', 'HIT_AND_RUN_FLAG', 'MOVEMENT_CODE'

앞서 상황 변수들에 의해 5개 군집으로 분류한 데이터들을 다시 3개의 군집으로 클러스터링 한다.

총 $5 * 3 = 15$ 개의 군집으로 사고 유형이 분류하였으며, 각 5개 군집과 그 소그룹들에 대해 decision tree를 그렸다(5번).

그리고 각 군집을 INJURY 평균값에 따라 -H / -MH / -ML / -L 4가지 중 하나로 소분류 하였다.

이 때 H(0.08이상) MH(0.065-0.08) ML(0.05-0.065) L(0.05미만)

그 결과 통합되는 군집이 있기 때문에 총 14개의 군집이 나온다.

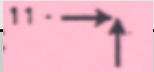


위에서 그린 5개의 decision tree를 이용하여 소분류 기준을 대략적으로 정리하면 다음과 같다.

또한 14개의 군집을 INJURY 평균에 따라 나열하면 오른쪽 리스트와 같은 순서가 된다.

위 군집일수록 INJURY 정도가 약하고, 아래 군집일수록 INJURY 정도가 크다.

당초 상황 변수만을 가지고 한 5개 분류와는 달리, 돌발 변수가 개입한 2차 클러스터링 이후 부상 정도의 순위가 뒤바뀌는 것을 알 수 있다.

```
[('E-L', 0.050749318801089918),
('C-L', 0.060623556581986142),
('D-ML', 0.069698759598346136),
('E-ML', 0.073463268365817097),
('A-ML', 0.075595727198027943),
('B-ML', 0.079672131147540987),
('B-MH', 0.083005366726296964),
('D-MH', 0.083158263305322125),
('E-MH', 0.085918854415274457),
('A-MH', 0.086446280991735541),
('C-MH', 0.095042796005706129),
('C-H', 0.10319685922602355),
('A-H', 0.12071354347073086),
('B-H', 0.1223185759926974)]
```

	A		B		C		D	E	
대분류 기준	Surface dry, Intersection의 사고		Surface dry, Non-intersection two-way, not divided		Surface 99% wet		Surface dry, Passenger car	Surface dry, A,B,D에 속하지 않음	
소분류 기준: 부상 증가 요인	A-H	Collision_type : "same movement angle" 	B-H	Collision_type: "single vehicle"	C-H		별 차이가 없다.	E-M	area_damaged: 'front'
소분류 기준: 부상 감소 요인	A-M		B-M		C-L	(Collision_type: 'same direction rear end') + (area_damaged: 'back' Or movement: 'slowing stoping') 		E-L	area_damaged: 'back' + Collision_type: 'same direction rear end' 

4. 평가 척도 제시

* 세가지 평가 척도를 이용하여 사고 유형을 비교해본다.

1) avg(INJURY) : 각 사고 유형에 대한 부상 정도의 순수한 측정이 가능하다.

- INJURY 변수 : 앞서 정의한 INJURY 변수들의 평균 값, INJ_SEVER_CODE의 범주 마다 서로 다른 가중치를 부여함

No injury = 0점 / Fatal injury=30점 / 나머지 부상 = 15점

(이유: 분석의 편의상 나머지 변수는 절반의 점수로 책정하였다. 본 연구의 취지는 치명상을 입히는 사고에 대한 분석이

fatal injury의 샘플이 너무 적기 때문에 분석이 어렵다고 판단하여, 부상을 입히는 교통 사고 전반에 대한

분석으로 관점을 넓혀서 경상의 경우에도 절반의 점수를 부여하였다.)

- 앞서 비교해 본 바 있다.

2) FREQUENCY : 해당 사고 유형의 레코드 개수를 count

- injury 정도가 심각한 유형이 대체로 빈도수도 높은 편이었다.

(B-H,A-H,C-H)

3) avg(INJURY) X FREQUENCY

- 고위험군의 중요성을 측정하기 위한 지표이다.

- 대체로 injury 정도가 심각한 유형이 계산 결과 높았다. (A-H,B-H,C-MH)

4) damage_score: DAMGE_CODE 변수를 그 심각도에 따라 숫자로 환산하여 비교해보았다.

Destroyed=30 / Disabled=15 / Functional=10 / Superfical=5 / No Damage=0

- Destoryed의 경우가 나머지 damage에 비해 압도적으로 인명에 손실을 끼치기 때문에 큰 점수를 부여하였다.

- INJURY에 비해서 변수의 편중 정도가 적기 때문에 비교에 용이

- INJURY 분석결과와 결과가 크게 다르지 않으나, C-H의 경우 인명 피해에 비해 차가 많이 손실되었음을 알 수 있다.

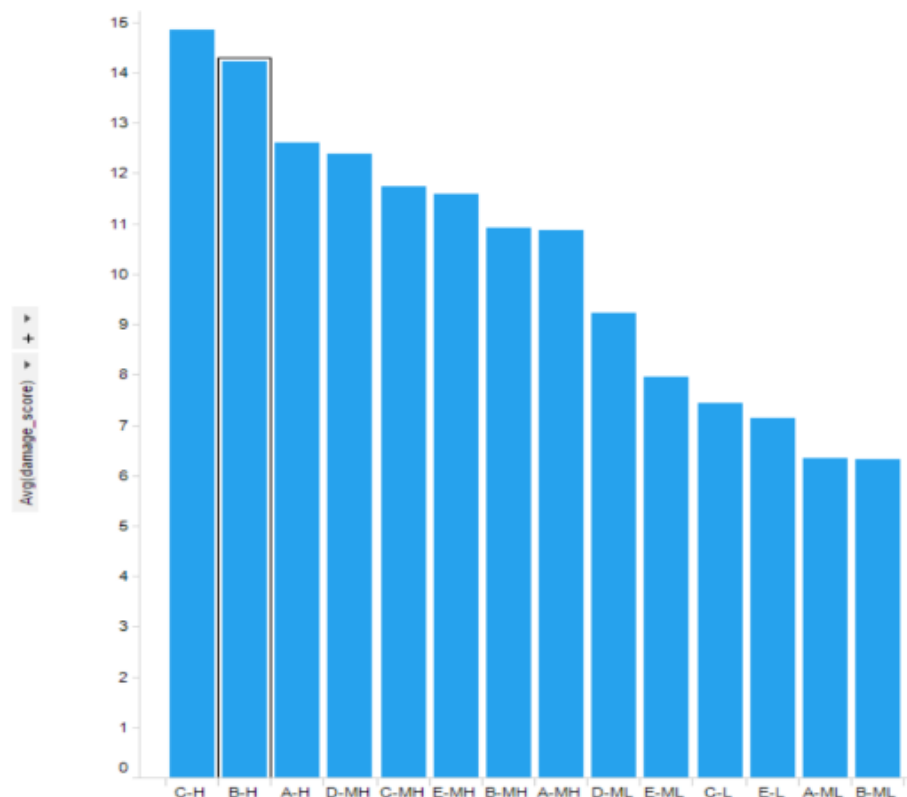
#avg(INJURY)

```
[('E-L', 0.050749318801089918),
('C-L', 0.060623556581986142),
('D-ML', 0.069698759598346136),
('E-ML', 0.073463268365817097),
('A-ML', 0.075595727198027943),
('B-ML', 0.079672131147540987),
('B-MH', 0.083005366726296964),
('D-MH', 0.083158263305322125),
('E-MH', 0.085918854415274457),
('A-MH', 0.086446280991735541),
('C-MH', 0.095042796005706129),
('C-H', 0.10319685922602355),
('A-H', 0.12071354347073086),
('B-H', 0.1223185759926974)]
```

#FREQUENCY

```
[('E-L', 1468),
('C-L', 1732),
('D-ML', 3386),
('E-ML', 667),
('A-ML', 1217),
('B-ML', 1525),
('B-MH', 2795),
('D-MH', 2856),
('E-MH', 1257),
('A-MH', 3025),
('C-MH', 2804),
('C-H', 1783),
('A-H', 2887),
('B-H', 2191)]
```

final_group	injury*freq
E-ML	49
E-L	74.5
A-ML	92
C-L	105
E-MH	108
B-ML	121.5
C-H	184
B-MH	232
D-ML	236
D-MH	237.5
A-MH	261.5
C-MH	266.5
B-H	268
A-H	348.5



6. 유형 선정 및 정책 제안

1그룹) 부상 고위험 그룹 : A-H그룹 + B-H그룹 (고위험군 중에서도 고위험군으로 엄선된 군집일 뿐만 아니라, 빈도수 역시 높은 편이라서 매우 중요한 집단으로 보았다.)

2그룹) 안전 그룹: C-L그룹 + E-L그룹 (이들 그룹은 C, E그룹 내부에서도 눈에 띄게 부상 정도가 감소한 그룹이기에, 유의미한 결과를 도출할 수 있는 집단으로 보았다.)

1. 군집별 대표 특성을 통한 대안 제시

(1) **A-H**: intersection(교차로) + same movement angle이 위험하다

-> spotfire 분석 결과, right turn시 수직 방향으로 충돌하여 정면이 손상되는 사고 유형으로 생각된다. intersection에서 신호 체계 정비 및 시야를 방해하는 장애물 제거 등을 통해 안전하게 수직 방향의 turn을 하도록 도로를 정비해야 한다.

(2) **B-H**: two-way, not divided(분리대가 없는 이차선 도로) + single vehicle(혼자 사고남) 조건이 위험하다

-> 이러한 도로 유형의 경우 다른 차량 충돌하는 것이 아니라 차 한대 혼자서 사고를 내는 경우가 잦고 위험하다는 뜻이다. 따라서 해당 도로 유형에서 차가 아닌 기타 장애물에 충돌할 확률이 높으며 위험성이 높음을 의미하므로, 기타 장애물에 대한 점검 및 해당 도로 유형에서의 시야 확보를 위한 대책이 필요하다.

(3) **C-L**: wet한 경우, same direction rear end + back이 damaged + stoping slowing movement에서 부상이 감소함

-> wet한 상황에서 차가 멈추지 않고 일정한 속도로 달리다가 앞방향이 손상되는 경우 부상이 비약적으로 높아졌다(C-H의 12시방향 damage) 따라서 비가 오는 날에는 오지 않는 날에 비해 차의 손상의 방향과 주행 속도가 부상에 엄청난 임팩트를 준다는 사실을 인지해야한다. 비가 오는 날에 과속 단속을 집중시키고, 앞좌석의 안전을 위한 에어백의 민감성이 증대되어야 할 것이다.

Etc) **D**: intersection도, two-way not divide도 아닌 경우 passenger car를 탔을 때가 그렇지 않을 때에 비해(E그룹) 부상의 정도가 크다

6. 유형 선정 및 정책 제안

1그룹) 부상 고위험 그룹(고위험군) : A-H그룹 + B-H그룹

2그룹) 안전 그룹(저위험군): C-L그룹 + E-L그룹

2. 기타 변수 분석: 고위험군(A-H,B-H)에서 높은 지표를 나타내고, 저위험군(C-L,E-L)에서 낮은 지표를 나타내는 변수들 그리고 그 반대 케이스의 변수들을 선정하여 부상을 높이는 변수, 부상을 낮추는 변수로 놓고 봤다.(오른쪽 코드 참고, boonsuk이라는 dataframe은 각 군집별로 col의 평균을 정리해 놓은 테이블이다.)

(1) 고위험군에 상대적으로 음주운전자가 많았다 ->음주운전 단속 강화

(2) 저위험군은 거의 100% equipment misuse가 없었으며, 자살한 misuse의 경우 고위험군에 몰려있었다

(3) 저위험군에서 뺑소니 사고의 경우는 거의 없었으며, 고위험군에 많았다

(4) Movement 가 accelerating(가속 중인 상태)의 비중이 고위험군이 저위험군에 비해 현저히 높았다.

(5) Lane이 acceleration lane인 비중이 저위험군이 고위험군에 비해 상당히 높았다.

->acceleration lane의 마련을 통해 안전하게 가속할 수 있는 여유를 준다

```
highindex={'makes higher':[], 'makes lower':[]}
for i in indexes:
    if boonsuk.loc[i, 'A-H']==boonsuk.loc[i].max() or boonsuk.loc[i, 'B-H']==boonsuk.loc[i].max():
        if boonsuk.loc[i, 'A-H']!=boonsuk.loc[i].min() and boonsuk.loc[i, 'B-H']!=boonsuk.loc[i].min():
            if not (boonsuk.loc[i, 'C-L']==boonsuk.loc[i].max() or boonsuk.loc[i, 'E-L']==boonsuk.loc[i].max()):
                highindex['makes higher'].append(i)
    if boonsuk.loc[i, 'A-H']==boonsuk.loc[i].min() or boonsuk.loc[i, 'B-H']==boonsuk.loc[i].min():
        if boonsuk.loc[i, 'A-H']!=boonsuk.loc[i].max() and boonsuk.loc[i, 'B-H']!=boonsuk.loc[i].max():
            if not (boonsuk.loc[i, 'C-L']==boonsuk.loc[i].min() or boonsuk.loc[i, 'E-L']==boonsuk.loc[i].min()):
                highindex['makes lower'].append(i)
```

```
lowindex={'makes higher':[], 'makes lower':[]}
for i in indexes:
    if boonsuk.loc[i, 'C-L']==boonsuk.loc[i].max() or boonsuk.loc[i, 'E-L']==boonsuk.loc[i].max():
        if boonsuk.loc[i, 'C-L']!=boonsuk.loc[i].min() and boonsuk.loc[i, 'E-L']!=boonsuk.loc[i].min():
            if not (boonsuk.loc[i, 'A-H']==boonsuk.loc[i].max() or boonsuk.loc[i, 'B-H']==boonsuk.loc[i].max()):
                lowindex['makes lower'].append(i)
    if boonsuk.loc[i, 'C-L']==boonsuk.loc[i].min() or boonsuk.loc[i, 'E-L']==boonsuk.loc[i].min():
        if boonsuk.loc[i, 'C-L']!=boonsuk.loc[i].max() and boonsuk.loc[i, 'E-L']!=boonsuk.loc[i].max():
            if not (boonsuk.loc[i, 'A-H']==boonsuk.loc[i].min() or boonsuk.loc[i, 'B-H']==boonsuk.loc[i].min()):
                lowindex['makes higher'].append(i)
```

```
high=[]
for h in highindex['makes higher']:
    if h in lowindex['makes higher']:
        high.append(h)
```

```
['CONDITION_CODE_Fatigued Fainted',
'CONDITION_CODE_Had Been Drinking',
'CONDITION_CODE_Other Handicaps',
'CONDITION_CODE_Using Drugs',
'EQUIP_PROB_CODE_Air Bag Failed',
'INJ_SEVER_CODE_Fatal Injury',
'INJ_SEVER_CODE_Incapacitating/Disabled Injury',
'INJ_SEVER_CODE_Possible Incapacitating Injury',
'LANE_CODE_Right Turn Lane',
'RD_COND_CODE_Loose Surface Material',
'RD_DIV_CODE_Two-way, Not Divided',
'SURF_COND_CODE_Ice',
'SURF_COND_CODE_Slush',
'SURF_COND_CODE_Snow',
'SURF_COND_CODE_Water (standing/moving)',
'WEATHER_CODE_Severe Winds',
'DAMAGE_CODE_Destroyed',
'HIT_AND_RUN_FLAG_Y',
'MOVEMENT_CODE_Accelerating',
'INJURY']
```

```
In [49]: low=[]
for h in highindex['makes lower']:
    if h in lowindex['makes lower']:
        low.append(h)

low
```

```
Out[49]: ['CONDITION_CODE_Apparently Normal',
'EQUIP_PROB_CODE_No Misuse',
'INJ_SEVER_CODE_No Injury',
'COLLISION_TYPE_CODE_Same Direction Rear End',
'LANE_CODE_Acceleration Lane',
'LANE_CODE_Crossover Area',
'RD_DIV_CODE_Two-way, Divided, Positive Median Barrier',
'MOVEMENT_CODE_Slowing or Stopping',
'MONTH']
```

끝!