머신러닝을 활용한 모빌리티 선택 예측 모형 연구

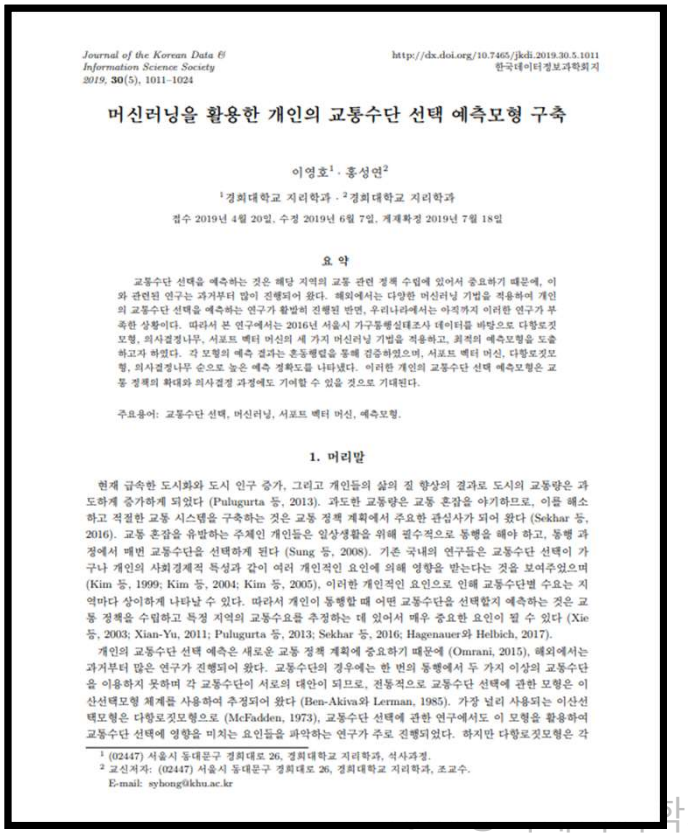
- 2021년 여객교통시설물 이용 실태조사를 기반으로-

2019110263 정형민
2020102510 차승민
2023101856 김재민

2024 경희대지리학과 공간정보 X 주소정보 X 파이썬 프로그래밍 부트캠프

논문 소개

「머신러닝을 활용한 개인의 교통수단 선택 예측모형 구축」(2019, 이영호 · 홍성연)



Journal of the Korean Data & Information Science Society
2019, **30**(5), 1011–1024

http://dx.doi.org/10.7465/jkdi.2019.30.5.1011
한국데이터정보과학회지

머신러닝을 활용한 개인의 교통수단 선택 예측모형 구축

이영호[1] · 홍성연[2]

[1]경희대학교 지리학과 · [2]경희대학교 지리학과

접수 2019년 4월 20일, 수정 2019년 6월 7일, 게재확정 2019년 7월 18일

요 약

교통수단 선택을 예측하는 것은 해당 지역의 교통 관련 정책 수립에 있어서 중요하기 때문에, 이와 관련된 연구는 과거부터 많이 진행되어 왔다. 해외에서는 다양한 머신러닝 기법을 적용하여 개인의 교통수단 선택을 예측하는 연구가 활발히 진행된 반면, 우리나라에서는 아직까지 이러한 연구가 부족한 상황이다. 따라서 본 연구에서는 2016년 서울시 가구통행실태조사 데이터를 바탕으로 다항로짓모형, 의사결정나무, 서포트 벡터 머신의 세 가지 머신러닝 기법을 적용하고, 최적의 예측모형을 도출하고자 하였다. 각 모형의 예측 결과는 혼동행렬을 통해 검증하였으며, 서포트 벡터 머신, 다항로짓모형, 의사결정나무 순으로 높은 예측 정확도를 나타냈다. 이러한 개인의 교통수단 선택 예측모형은 교통 정책의 확대와 의사결정 과정에도 기여할 수 있을 것으로 기대된다.

주요용어: 교통수단 선택, 머신러닝, 서포트 벡터 머신, 예측모형.

1. 머리말

현재 급속한 도시화와 도시 인구 증가, 그리고 개인들의 삶의 질 향상의 결과로 도시의 교통량은 과도하게 증가하게 되었다 (Pulugurta 등, 2013). 과도한 교통량은 교통 혼잡을 야기하므로, 이를 해소하고 적절한 교통 시스템을 구축하는 것은 교통 정책 계획에서 주요한 관심사가 되어 왔다 (Sekhar 등, 2016). 교통 혼잡을 유발하는 주체인 개인들은 일상생활을 위해 필수적으로 통행을 해야 하고, 통행 과정에서 매번 교통수단을 선택하게 된다 (Sung 등, 2008). 기존 국내의 연구들은 교통수단 선택이 가구나 개인의 사회경제적 특성과 같이 여러 개인적인 요인에 의해 영향을 받는다는 것을 보여주었으며 (Kim 등, 1999; Kim 등, 2004; Kim 등, 2005), 이러한 개인적인 요인으로 인해 교통수단별 수요는 지역마다 상이하게 나타날 수 있다. 따라서 개인이 통행할 때 어떤 교통수단을 선택할지 예측하는 것은 교통 정책을 수립하고 특정 지역의 교통수요를 추정하는 데 있어서 매우 중요한 요인이 될 수 있다 (Xie 등, 2003; Xian-Yu, 2011; Pulugurta 등, 2013; Sekhar 등, 2016; Hagenauer와 Helbich, 2017).

개인의 교통수단 선택 예측은 새로운 교통 정책 계획에 중요하기 때문에 (Omrani, 2015), 해외에서는 과거부터 많은 연구가 진행되어 왔다. 교통수단의 경우에는 한 번의 통행에서 두 가지 이상의 교통수단을 이용하지 못하며 각 교통수단이 서로의 대안이 되므로, 전통적으로 교통수단 선택에 관한 모형은 이산선택모형 체계를 사용하여 추정되어 왔다 (Ben-Akiva와 Lerman, 1985). 가장 널리 사용되는 이산선택모형은 다항로짓모형으로 (McFadden, 1973), 교통수단 선택에 관한 연구에서도 이 모형을 활용하여 교통수단 선택에 영향을 미치는 요인들을 파악하는 연구가 주로 진행되었다. 하지만 다항로짓모형은 각

[1] (02447) 서울시 동대문구 경희대로 26, 경희대학교 지리학과, 석사과정.
[2] 교신저자: (02447) 서울시 동대문구 경희대로 26, 경희대학교 지리학과, 조교수.
E-mail: syhong@khu.ac.kr

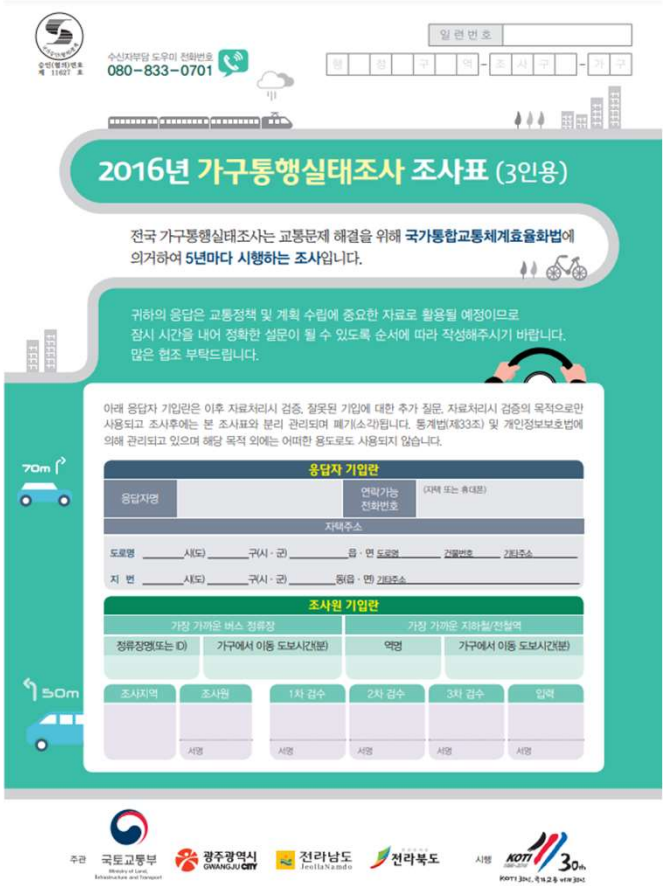
교통수단의 선택은 여러 개인적인 요인의 영향을 받음

⇩

개인이 통행할 때 어떤 교통수단을 이용할지 예측하는 모형 구축

교통 정책의 수립과 특정 지역 교통수요 추정에서 중요
<u>특히 서울 및 수도권에 인구가 집중되어 활발한 연구 필요!</u>

논문 소개



가구통행실태조사(2016)

<논문에서 사용한 변수들>

버스 정류장 접근성	정기 교육
지하철역 접근성	직업 형태
가족 구성원 수	통행 날짜 (월)
집의 형태	통행 날짜 (일)
월간 수입	통행 목적
자동차 소유 여부	최종 목적지
태어난 연도	통행 거리
성별	통행시간
운전면허증 소유여부	

변수에 따라 어떤 교통수단을 선택할 지 알 수 있다.
But 변수가 선택에 얼마나 영향을 주는 지 알 수 없다.

변수가 결과에 얼마나 영향을 주는지 알아내고 싶음

프로젝트 소개

가구통행실태조사(2016) ⇨ **개인**통행실태조사(2021)

<입력변수, X>

성별	자전거 및 킥보드
연령(만)	면허증 보유여부
총 가구원 수	통행수
주거 유형	통행 목적
월평균 소득	총 통행 시간
승용차	소요시간
오토바이(이륜차)	

<목표변수, y>

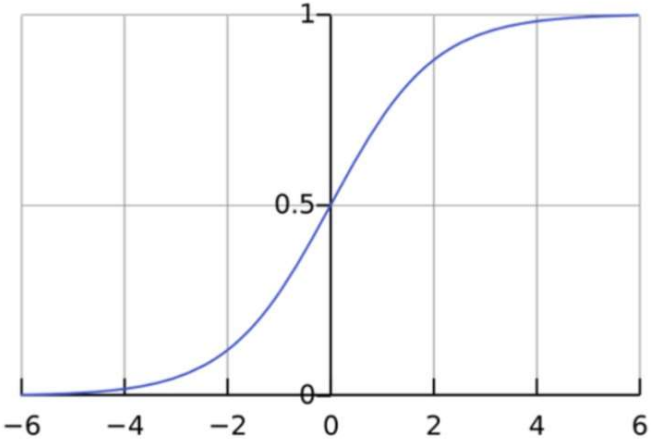
이동 수단

<분석 방법(머신러닝 기법)>

다항 로지스틱 회귀분석
랜덤 포레스트
서포트 벡터 모형

<분석방법>

※ 머신러닝 지도학습 – 분류(classification)



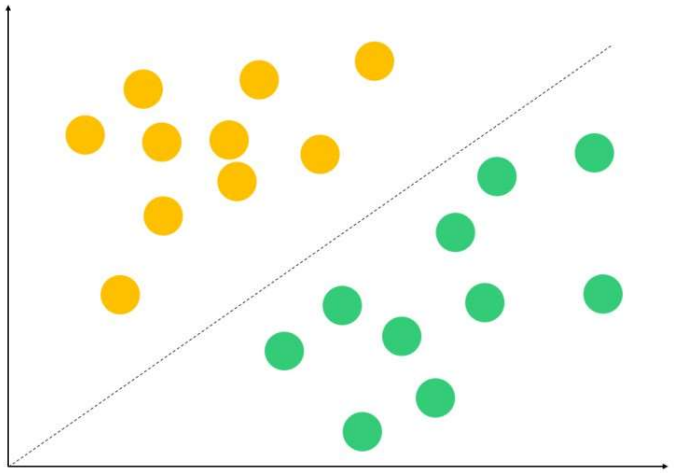
✓ **다항 로지스틱 회귀모형(Multinomial Logistic Regression)**

- 로지스틱 회귀분석과 유사하지만 종속변수가 두 개의 범주로 제한되지 않음 -> 여러 범주형 변수를 분류 가능함

✓ **서포트 벡터 머신(Support Vector Machine)**

- 지도학습의 분류문제에 주로 사용, 다양한 차원 경계에서 분리를 진행하여 최적의 경계를 찾는 것을 목표

- **초평면**: 데이터를 두 개의 클래스로 나누는 선 또는 면
- **마진**: 초평면과 가장 가까운 데이터 포인트 사이의 거리. SVM은 이를 최대화하려 함
- **서포트 벡터**: 마진에 가장 가까이 있는 데이터 포인트

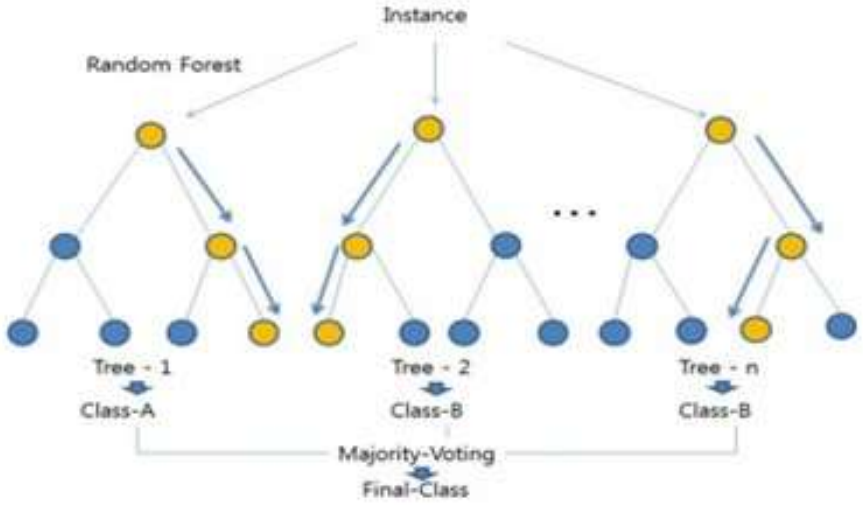


<분석방법>

※ 머신러닝 기법

✓ **랜덤 포레스트(Random Forest)**

- 여러 의사결정나무를 조합한 지도학습 알고리즘 모델
- 과적합 해소, 분산 감소 -> 정확도 상승
- 계산 비용 높음, 규칙이 많아 추론 로직 설명 어려움
- -> 복잡한 비선형 관계를 설명가능하게 하는 방식이 존재

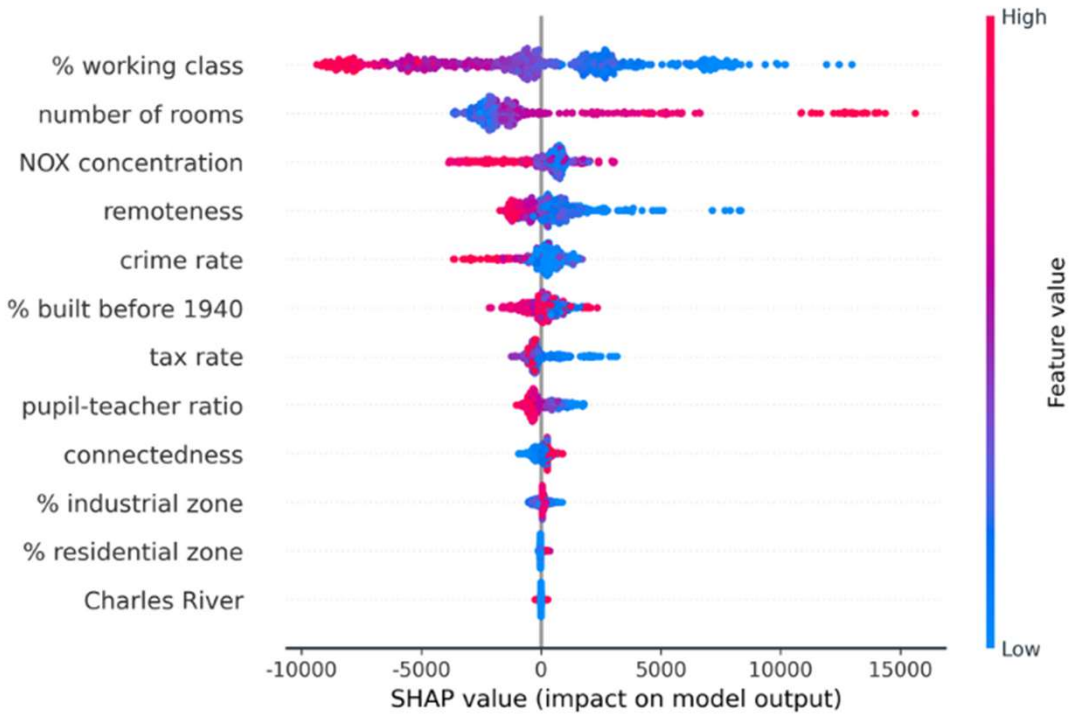


<분석방법>

※설명가능한 인공지능(SHAP)

- **SHAP value (SHapley Additive exPlanations)**

전체 결과에 각 feature들이 얼마나 영향을 미쳤는지 수치로 표현 가능



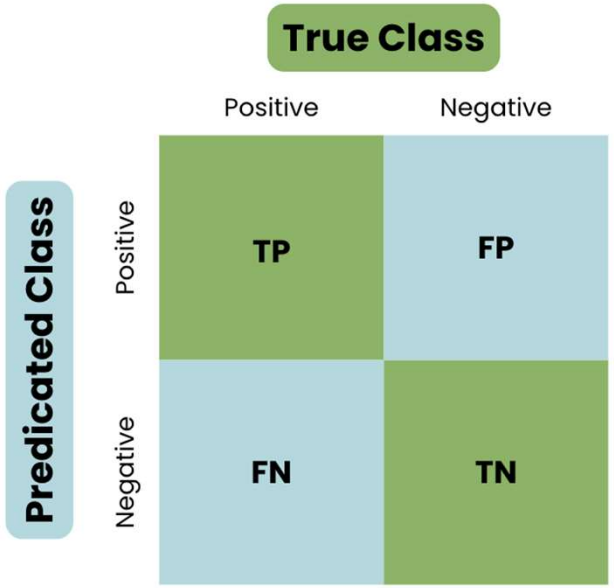
<분류의 결과 – 혼동 행렬>

- **정확도**

전체에서 예측한 것들 중 올바른 예측을 얼마나 했는 지에 관한 지표

- **정밀도**

분류된 것들 중 실제로 맞춘 개수에 대한 지표



<코드> _ 전처리

```python
# 범주형 데이터 카테고리화
mob_df['license'] = mob_df['license'].astype('category')
mob_df['move_type'] = mob_df['move_type'].astype('category')
mob_df['mobility'] = mob_df['mobility'].astype('category')
```

```python
# 결과
sex              category
age                 int64
pop_family        float64
house_type       category
income           category
car               float64
motorcycle        float64
bike              float64
license          category
depart           category
destination      category
move_type        category
total_time        float64
mobility         category
mobility_time     float64
```

<코드> _ 전처리2

```
# 범주형 아닌 데이터 정규화
from sklearn import preprocessing
fining_df = mob_df[['age','pop_family','car','motorcycle','bike','total_time','mobility_time']]
normalized_data = scaler.fit_transform(fining_df)
```

```
# one-hot-encoding (범주화)
catagory_df = mob_df[['sex','house_type','income','license','move_type']]
one_hot_encoded_df = pd.get_dummies(catagory_df)
```

```
#train, test set 만들기
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.3,random_state=5)
```

코드 _ 머신러닝

```python
# SVM 모델
from sklearn import svm
model_svm = svm.SVC(kernel='rbf', gamma='scale')
model_svm.fit(X_train,y_train.values.ravel())
y_pred = model_svm.predict(X_test)
```

```python
# radomforest 모델
from sklearn.ensemble import RandomForestClassifier
rf_classifier = RandomForestClassifier(random_state = 5)
re_classifier.fit(X_train, y_train)
y_pred = rf_classifier.predict(X_test)
```

```python
# multinomial logistic regression
from sklearn.lenear_model import LogisticRegression
model_logistic = LogisticRegression(solver = 'saga', max_iter=2000)
model_logistic.fit(X_train, y_train)
y_pred = model_logistic.predict(X_test)
```

```python
# 정확도/정밀도 계산
정밀도 = np.diag(confusion_np) / confusion_df.sum()
정확도 = np.diag(confusion_np.sum() / confusion_df.sum().sum()
```

<분석결과 다항 Logistic>

	도보	승용차(직접)	승용차(타인)	시내버스	마을버스	시외버스	고속버스	기타버스	지하철	경전철	고속철도	일반철도	택시	소형화물차	대형화물차	자전거
0	26471	7841	212	0	0	154	0	0	0	5	13	21	0	0	0	0
1	5909	36171	135	22	0	487	0	1	0	102	10	44	0	0	0	0
2	4625	2826	576	0	0	464	0	0	0	3	5	6	0	0	0	0
3	59	115	63	14	0	118	0	0	0	1	1	0	0	0	0	0
4	207	563	49	0	0	96	0	0	0	3	0	1	0	0	0	0
5	2862	2868	590	0	1	960	0	0	0	5	4	3	0	0	0	0
6	16	32	20	4	0	25	0	0	0	0	0	0	0	0	0	0
7	13	35	10	1	0	68	0	0	0	0	0	0	0	0	0	0
8	638	593	33	0	0	21	0	0	0	1	0	0	0	0	0	0
9	76	1312	2	3	0	36	0	3	0	82	0	8	0	0	0	0
10	529	554	11	0	0	14	0	0	0	1	2	1	0	0	0	0
11	107	197	0	0	0	1	0	0	0	4	0	58	0	0	0	0
12	102	48	3	0	0	3	0	0	0	0	1	0	0	0	0	0
13	10	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	0	17	6	0	0	7	0	0	0	2	0	0	0	0	0	0
15	6	7	1	0	0	1	0	0	0	1	0	0	0	0	0	0

- 99,418 중에 64,334를 맞춤 (=정확도 약 64.7%)
- 정밀도를 정리한 표

승용차 🚗	68%
도보🚶	64%
일반철도🚆	41%
시외버스🚌	39%

☆소수 첫째 자리에서 반올림

<분석결과 SVM>

	도보	승용차(직접)	승용차(타인)	시내버스	마을버스	시외버스	고속버스	기타버스	지하철	경전철	고속철도	일반철도	택시	소형화물차	대형화물차	자전거
0	26209	8305	129	0	0	73	0	0	0	0	0	1	0	0	0	0
1	4855	37719	111	0	0	190	0	0	0	0	0	6	0	0	0	0
2	4990	2641	425	0	0	449	0	0	0	0	0	0	0	0	0	0
3	77	150	48	0	0	96	0	0	0	0	0	0	0	0	0	0
4	238	558	42	0	0	81	0	0	0	0	0	0	0	0	0	0
5	3163	2671	460	0	0	999	0	0	0	0	0	0	0	0	0	0
6	24	28	20	0	0	25	0	0	0	0	0	0	0	0	0	0
7	18	44	4	0	0	61	0	0	0	0	0	0	0	0	0	0
8	647	603	18	0	0	18	0	0	0	0	0	0	0	0	0	0
9	72	1437	2	0	0	11	0	0	0	0	0	0	0	0	0	0
10	513	585	9	0	0	5	0	0	0	0	0	0	0	0	0	0
11	104	248	0	0	0	1	0	0	0	0	0	14	0	0	0	0
12	107	48	0	0	0	2	0	0	0	0	0	0	0	0	0	0
13	11	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	5	19	1	0	0	7	0	0	0	0	0	0	0	0	0	0
15	6	8	1	0	0	1	0	0	0	0	0	0	0	0	0	0

- 99,418 중에 65,366를 맞춤 (=정확도 약 65.7%)
- 정밀도를 정리한 표

승용차 🚗	68%
일반철도🚈	67%
도보🚶	64%
시외버스🚌	49%

☆소수 첫째 자리에서 반올림

<분석결과 Random Forest>

	도보	승용차(직접)	승용차(타인)	시내버스	마을버스	시외버스	고속버스	기타버스	지하철	경전철	고속철도	일반철도	택시	소형화물차	대형화물차	자전거
0	24646	6851	1680	9	46	1162	6	4	92	50	118	33	13	4	1	2
1	5819	34964	738	16	80	685	2	7	113	295	105	38	13	1	4	1
2	3368	1770	1744	15	24	1522	5	7	23	4	14	4	3	0	2	0
3	49	116	86	25	0	87	1	2	1	4	0	0	0	0	0	0
4	169	424	119	3	40	158	0	0	2	4	0	0	0	0	0	0
5	1881	1543	1351	25	30	2402	8	13	17	10	8	1	0	1	3	0
6	11	22	17	2	0	40	3	1	1	0	0	0	0	0	0	0
7	5	56	20	3	1	34	0	3	1	3	0	0	1	0	0	0
8	522	504	122	2	1	63	1	1	61	3	5	0	0	0	1	0
9	89	1084	11	1	4	17	1	1	2	310	1	1	0	0	0	0
10	475	458	64	0	0	44	0	0	3	6	58	2	2	0	0	0
11	76	157	3	0	0	4	0	0	2	4	0	121	0	0	0	0
12	101	35	11	0	0	3	0	0	0	0	2	0	5	0	0	0
13	12	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	0	18	2	0	1	9	0	0	0	1	0	0	0	0	1	0
15	5	10	0	0	0	0	0	0	0	0	0	0	0	0	0	1

- 99,418 중에 64,384를 맞춤 (=정확도 약 64.8%)
- 정밀도를 정리한 표

승용차 🚗	73%
도보🚶	66%
일반철도🚆	61%
시외버스🚌	39%

☆소수 첫째 자리에서 반올림

<결과 해석>

	정확도	정밀도 1등	정밀도 2등	정밀도 3등
다항로지스틱	64.7%	승용차	도보	일반철도
서포트 벡터	65.7%	승용차	일반철도	도보
랜덤 포레스트	64.8%	승용차	도보	일반철도

프로젝트: 서포트 벡터 머신, 랜덤 포레스트, 다항 로지스틱
논문: 서포트 벡터 머신, 다항 로지스틱, 의사결정 나무

- 논문과 유사하게 서포트 벡터 머신이 정확도가 가장 높게 나옴.
- 13개의 변수 중 승용차, 도보, 일반철도가 모든 케이스에서 정밀한 결과를 나타냄.
- 이는, 데이터의 분포가 특정 모빌리티에 많은 빈도를 보여 더 정확한 학습을 할 수 있어 정밀한 결과를 도출한 것으로 해석할 수 있음.
- 데이터의 비선형적 특성을 고려한 머신러닝이므로 다항 로지스틱보다 서포트 벡터 머신과 랜덤 포레스트가 정확도가 높게 나온 것으로 유추함.

한계점

- 3개의 모델의 정확도가 큰 차이를 보이지 않아, 해당 케이스에서 월등한 모델을 추리지 못한점
- 입력변수와 목표변수의 범주형 데이터의 종류가 많아, 학습의 시간이 오래 걸린 점
- 머신러닝에서 Sklearn 라이브러리를 사용함에, CPU만 사용하여 학습이 상대적으로 느려진 점. GPU를 사용하고 싶다면 tensorflow를 사용하여야 한다.

느낀점

- 생각보다 많은 머신러닝 알고리즘들이 라이브러리로 구현이 되어있다.
- 분류 문제를 다룰 시에 효율적인 변수 선택을 해야한다.