**랜덤 포레스트**

<공부한 내용>

정형데이터를 다루는데 가장 뛰어난 성과를 내는 알고리즘을 앙상블 학습이라고 하는데, 랜덤 포레스트는 앙상블 학습의 대표적인 방법이다. 결정트리를 랜덤하게 만들어서, 이들의 예측을 통해 최종적인 예측을 하는 모델이다.

배깅을 사용하는데, 이는 bootstrap aggregating의 약자로, 부트스트랩으로 작은 결정나무들을 만들고, 이를 결합시키는 방법이다. 여기서 부트스트랩의 가장 큰 특징은 데이터에서 샘플링을 할 때 복원추출을 한다는 점이다.

기본적으로 분류모델의 경우 전체 특성 개수의 제곱근만큼의 특성을 선택 하고, 회귀모델의 경우 전체특성을 사용한다.

- 분류: max\_features=sqrt(n\_features)

- 회귀: max\_features=n\_features

랜덤 포레스트의 장점

1. 분류(classification), 회귀(Regression)문제에 모두 사용 가능함.

2. 결측치(Missing value)를 다루기 쉽다.

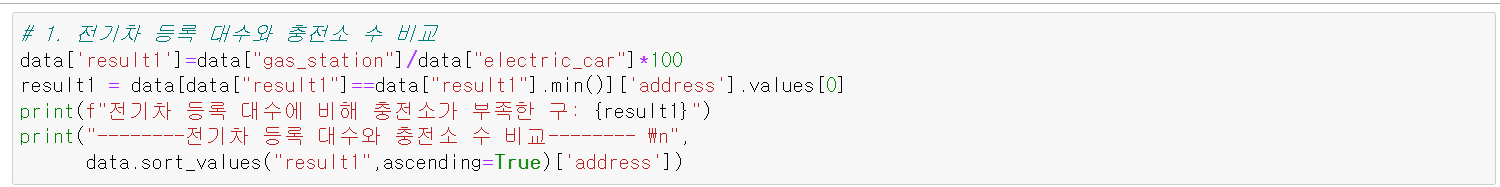
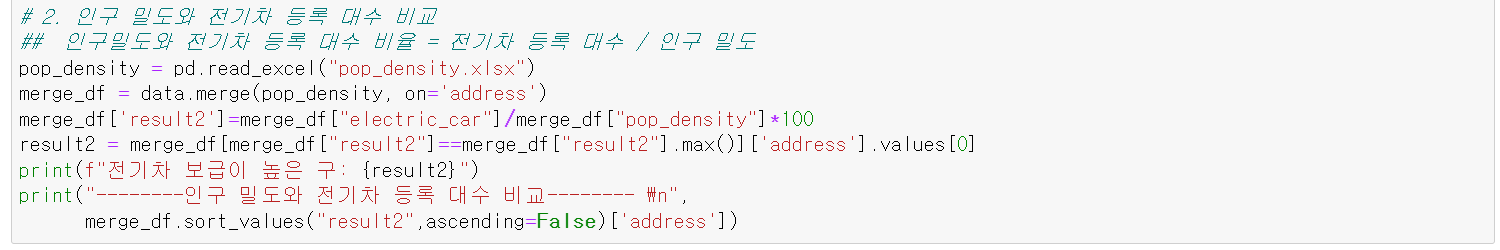
3. 대용량데이터 처리가 쉽다.

4. 과적합(Overfitting)문제를 해결해준다.

5. 특성중요도(Feature importance)를 구할 수 있다.

<코드>

|  |
| --- |
| *import pandas as pd*  *import numpy as np*  *from sklearn.model\_selection import train\_test\_split*  **# train\_set와 test\_set로 나누기**  *data = pd.read\_excel("new\_join.xlsx")*  *data\_select = data[["generation","generation\_per","generation\_sum","toilet","electric\_car","급속충전기","완속충전기"]]*  *target\_data = data['gas\_station']*  *train\_input, test\_input, train\_target, test\_target = train\_test\_split(data\_select, target\_data, test\_size=0.2, random\_state=42)*  *from sklearn.model\_selection import cross\_val\_score*  *from sklearn.ensemble import RandomForestRegressor*  *rf = RandomForestRegressor(n\_estimators=100, random\_state=42)*  *rf.fit(train\_input, train\_target)*  **# 변수 중요도**  *importances = rf.feature\_importances\_*  *feature\_names = data\_select.columns*  *importance\_data = pd.DataFrame(zip(feature\_names, importances))*  *importance\_data.sort\_values(1,ascending=False)*  **# 변수중요도를 활용한 결과 도출**  *# 1. 전기차 등록 대수와 충전소 수 비교*  *data['result1']=data["gas\_station"]/data["electric\_car"]\*100*  *result1 = data[data["result1"]==data["result1"].min()]['address'].values[0]*  *print(f"전기차 등록 대수에 비해 충전소가 부족한 구: {result1}")*  *print("--------전기차 등록 대수와 충전소 수 비교-------- \n",*  *data.sort\_values("result1",ascending=True)['address'])*  **# 2. 인구 밀도와 전기차 등록 대수 비교**  *## 인구밀도와 전기차 등록 대수 비율 = 전기차 등록 대수 / 인구 밀도*  *pop\_density = pd.read\_excel("pop\_density.xlsx")*  *merge\_df = data.merge(pop\_density, on='address')*  *merge\_df['result2']=merge\_df["electric\_car"]/merge\_df["pop\_density"]\*100*  *result2 = merge\_df[merge\_df["result2"]==merge\_df["result2"].max()]['address'].values[0]*  *print(f"전기차 등록 대수에 비해 충전소가 부족한 구: {result2}")*  *print("--------인구 밀도와 전기차 등록 대수 비교-------- \n",*  *merge\_df.sort\_values("result2",ascending=False)['address'])* |



<결과>

1) 변수 중요도

electric\_car - toilet - generation\_per - generation\_sum - 급속충전기 - 완속충전기 – generation 순서대로 높았음.

→ 랜덤포레스트의 결과 gas\_station에 가장 큰 영향력을 미치는 변수는 electric\_car

2) gas\_station에 가장 큰 영향력을 미치는 변수는 electric\_car이었기 때문에 electric\_car을 활용한 방안을 고려하여 두가지 비교를 했음.

|  |
| --- |
| 1. 전기차 등록 대수와 충전소 수 비교 = 충전소가 적은 구 찾기  : 구별로 전기차 등록 대수와 충전소 수를 비교하여, 전기차 등록 대수에 비해 충전소 수가 적은 구 추출.  → 충전소/전기차수가 적을수록 전기차 충전 인프라가 부족할 것이라고 예상할 수 있기 때문이다.  결과를 확인해봤을 때, 가장 낮았던 구는 서대문구였고, 차례로  강남구 – 구로구 – 서초구 – 강동구 – 마포구 – 중구 – 성동구 – 영등포구 – 용산구 –  동작구 - 종로구 – 송파구 – 노원구 – 강서구 – 중랑구 – 은평구 – 관악구 – 금천구 –  양천구 – 강북구 – 광진구 – 성북구 – 도봉구 – 동대문구였음.  \* 서대문구, 강남구, 구로구 등 상위에 있는 구들은 이미 전기차를 보유하고 있는 사람들이 충전소 이용에 어려움을 겪을 수 있을 거라고 생각된다. 그래서 충전소의 수를 늘려야 한다. |

|  |
| --- |
| 2. 인구 밀도와 전기차 등록 대수 비교 = 전기차 등록 대수가 높은 구 찾기  : 구별로 인구 밀도와 전기차 등록 대수를 비교하여 높은 구 추출.  → 전기차 등록 대수 비율이 높을수록 해당 지역은 상대적으로 전기차 등록이 많은 상태이므로 충전소가 필요하다고 예상할 수 있기 때문이다.  결과를 확인해봤을 때, 가장 높았던 구는 강남구였고, 차례로  서초구 – 구로구 – 영등포구 – 강서구 – 송파구 – 종로구 – 용산구 – 성동구 – 중구 –  마포구 – 강동구 – 노원구 – 은평구 – 성북구 – 관악구 – 도봉구 – 강북구 – 양천구 –  서대문구 – 중랑구 – 동작구 – 광진구 – 금천구 – 동대문구였음.  \* 강남구, 서초구, 구로구 등 상위에 있는 구들은 전기차 비율이 많기 때문에 해당 구에 충전소 수요가 높을 것이라고 예상할 수 있다. 그렇기 때문에 충전소 수를 늘려야 한다. |

종합적으로 보았을 때 충전소 수가 적은 구와 전기차 등록 대수가 모두 상위 5개 구안에 들었던 곳은 강남구, 서초구, 구로구였다. 그 중에서도 강남구는 두번째로 충전소가 적고, 가장 전기차 등록 대수가 높았기 때문에 충전소 보급이 가장 시급한 곳은 강남구라고 생각한다.