**AI를 활용한 중고차 거래가 예측 프로젝트**

**목차**

**1. 주제 선정 및 목표**

**-** 주제 선정

- 목표

**2. 데이터 수집 및 분석**

**-** 데이터셋 선정 및 수집

- 데이터 전처리

- 데이터 분석 및 시각화

**3. 모델링 및 평가**

- 딥러닝 모델

- 머신러닝 모델

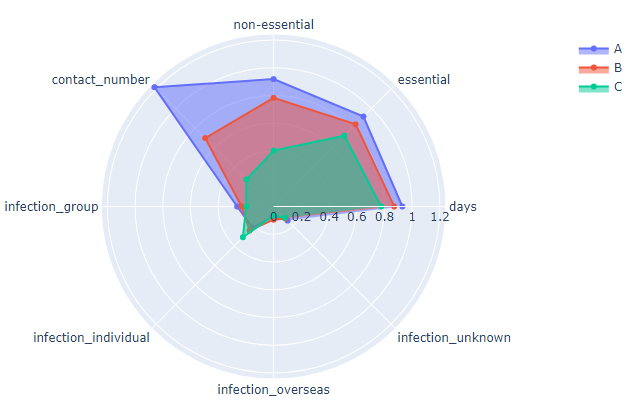
- 스태킹 앙상블 모델

**4. 웹 구현**

**1. 주제 선정 및 목표**

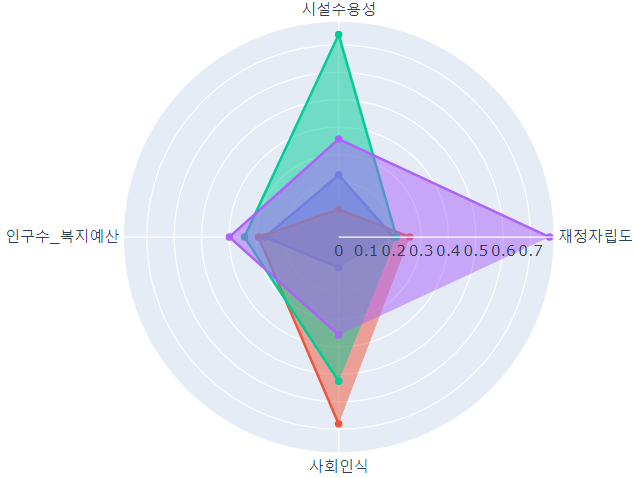
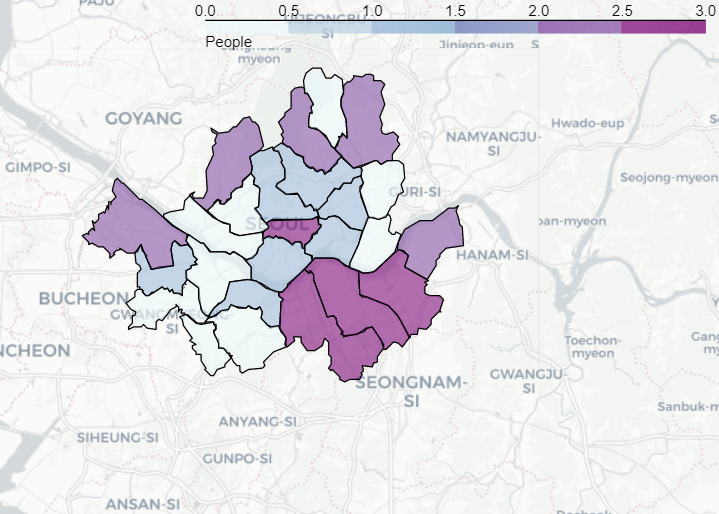
**1\_1. 주제 선정**

우리 조는 이 주제를 정하기 전에 다른 두 가지 주제에 대한 프로젝트를 진행하였고, 실패하였다. 첫 번째 주제는 코로나 감염 환자를 대상으로 한 클러스터링 및 시각화이고, 두 번째 주제는 장애인 복지시설 최적 위치 추천이었다. 다음은 전처리와 클러스터링을 마친 1차 주제의 결과를 시각화 한 것이다.



활동량이 많아 접촉자 수가 많고 확진판정까지 오랜 시간이 걸린 유형 A, 여전히 활동량은 많지만 접촉자 수는 적은 유형 B, 활동량이 적고 필수적인 경로만 이동했으나 개인 감염이 많은 유형 C로 분류되었다. 하지만 정부의 방침상 환자들의 경로를 비롯한 추가 정보를 공개하지 않아 이 이상 분석을 진행할 수 없었다.

2차 주제는 보건복지부의 ‘장애인 등록현황’ 데이터와 ‘장애인 복지시설 현황’ 데이터로 진행하였다.



장애인들의 수요를 알기 위해 설문조사를 비롯한 사전조사를 실시하였으나, 정보를 찾을 수 없었고 해당 지역에 사는 장애인구를 통해 수요를 대체하였다. 수요 대비 시설이 부족한 곳들을 선별하는데 까지는 성공했지만, 장애 복지 시설들이 어떤 기준으로 세워지는 지를 파악할 수 없었다. 따라서 위치 추천을 하더라도 평가 기준이 모호해졌다.

위의 두 번의 실패를 경험한 뒤 새로운 주제를 선정할 때 우선 고려한 것은 다음 세 가지이다.

첫째, 주제에 활용할 수 있는 양질의 데이터를 충분히 수집할 수 있는가?

둘째, 결과에 영향을 주는 요인들과 평가기준이 명확한가?

셋째, 프로젝트를 통하여 나온 결과의 타당성을 입증할 수 있는가?

위 세가지 조건을 만족하는 것들 중에 우리 조는 중고차 시세 예측 서비스를 주제로 선택하였다.

**1\_2. 목표**

자신의 차를 중고로 팔려고 하는 사람들에게 데이터에 의한 예상 가격을 제공하여 판매자가 중고차 딜러들이 제시한 값과 비교하여 더 나은 선택을 할 수 있도록 서비스를 제공하는 것이 목표이다.

**2. 프로젝트 제작 과정**

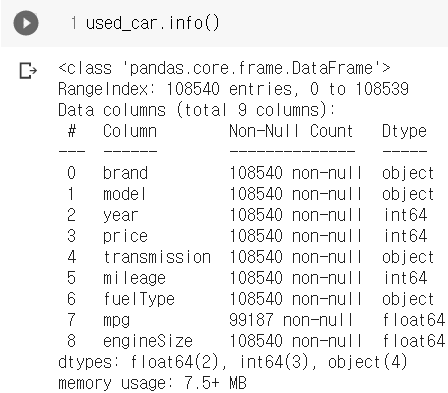
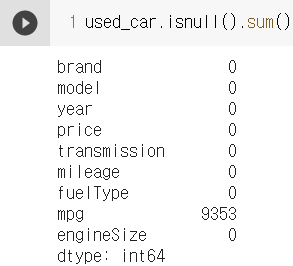
**2\_1. 데이터셋 선정 및 수집**

데이터셋 후보 : Kaggle(Used Cars Dataset, 100000UK Used Car Data set), 웹 크롤링

Used Cars Dataset의 경우 Row수는 42만개로 충분했으나, 대체불가능한 Null 값을 가진 Column들이 너무 많아서 제외하였다. 웹 크롤링의 경우 ‘보배드림’의 중고차 매물 정보를 수집했으나 전처리 결과 2천건의 데이터밖에 남지 않았다. 따라서 양질의 데이터를 가장 많이 보유하고 있는 UK Used Car 데이터를 사용하기로 결정했다.

**2\_2. 데이터 분석 및 시각화**

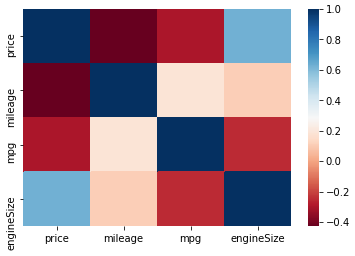
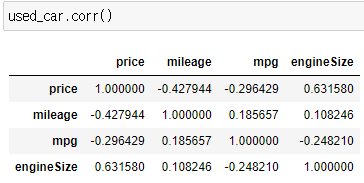
데이터 분석에는 python의 numpy와 pandas라이브러리를 사용하였고, 시각화 툴로는 plotly, seaborn을 사용하였다.

불필요한 Column을 삭제하였다. DataSet의 컬럼은 brand, model, year, transmission(구동방식), mileage(주행거리), fuelType, mpg(miles per gallon), enginzeSize(1000cc)이고 전체 rows 수는 108540이다. null값을 가진 mpg 컬럼의 경우 Outlier 값들을 삭제해 준 뒤 해당 Column의 평균값으로 대체해줄 것이다. 또한, year의 경우 현재 int형 값이므로 회귀계수 계산시 너무 큰 영향을 끼칠 수 있으므로, str타입으로 바꿔준 뒤 One-Hot-Encoding을 진행할 것이다.

**2\_2\_1. Column간의 상관관계 분석**

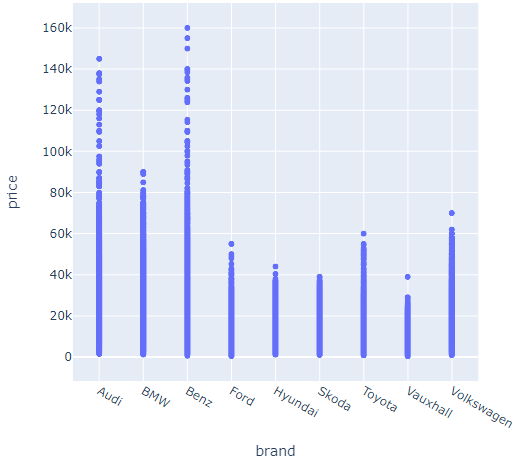
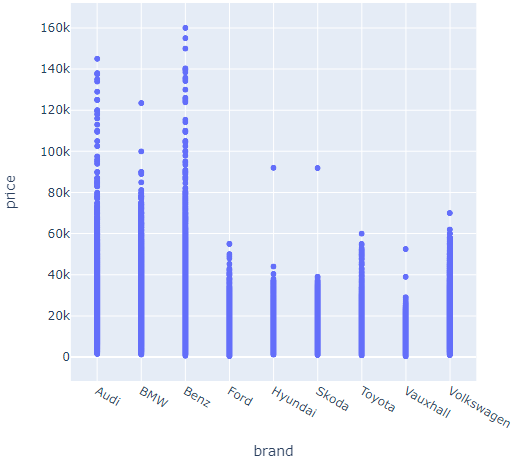
회귀 분석 시 feature간 상관관계가 매우 높을 경우 분산이 커지면서 오류가 급격히 증가하는 다중공선성(multi-collinearity)문제가 발생할 수 있다. 따라서 feature간 상관계수를 Heatmap으로 나타내었다.

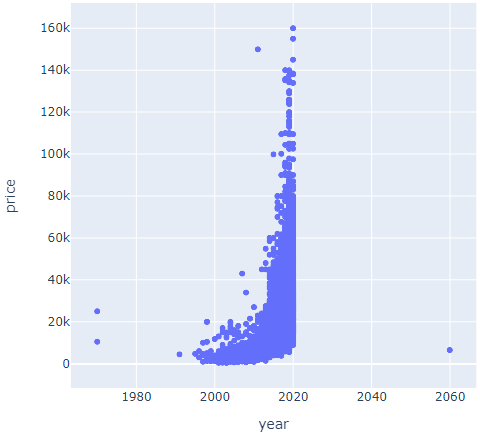
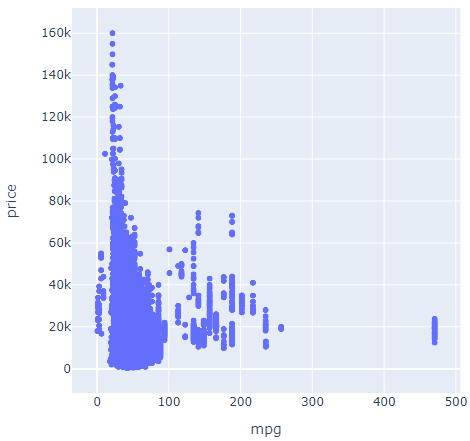


feature들간 상관관계가 존재하지 않는다. 단, target값인 price와 engineSize간의 약한 상관관계가 존재하므로, engineSize 컬럼 값에 회귀분석 시 영향을 끼칠 수 있는 Outlier가 있는지 확인하고, 삭제해 줄 필요가 있다.

**2\_2\_2 데이터의 결측값과 이상값 확인과 처리**

* target값인 price와 feature간의 산점도를 그려보고 outlier가 있는지 확인하였다.

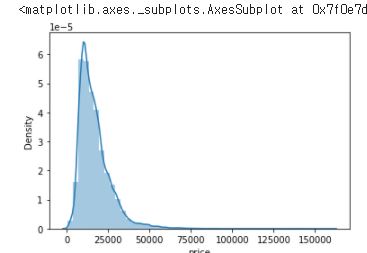
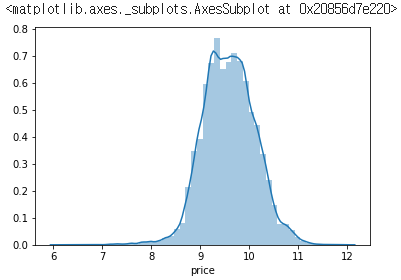


* bmw, 현대, skoda , vauxhall의 이상치들이 발견되었고 이를 제거했다.
* 나머지 column에 대해서도 이상치가 있는지 확인하고 제거해주었다.

year가 현재시점을 넘어선 값과 mpg값이 평균값에 비해 월등히 높은 값들도 삭제하였다.

**2\_2\_3. Target 데이터 로그변환**

선형 회귀 모델의 경우 target값이 정규분포 형태일 경우 성능에 긍정적인 영향을 미치기 때문에 target인 price의 분포를 우선 확인하였다. 왜곡된 분포를 따르고 있고 비싼 가격과 싼 가격의 격차가 큰 점, price column이 다른 column들에 비하여 값이 너무 큰 점이 있었기 때문에 log변환을 적용하여 정규분포에 비슷한 형태로 바꿔주었다.

**3. 모델링 및 평가**

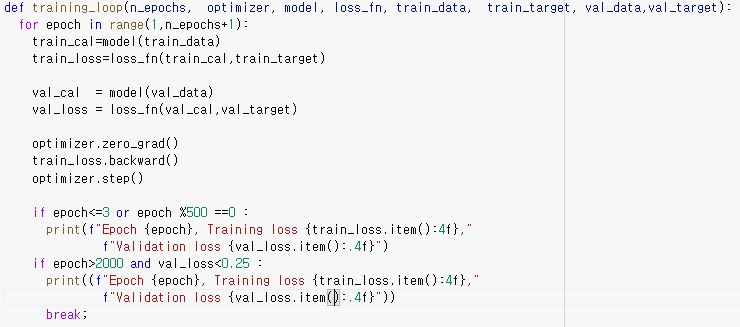
예측 모델은 머신러닝과 딥러닝을 모델링 한 뒤, 성능이 좋은 것을 채택하였다. 딥러닝은 파이토치를, 머신러닝은 Scikit-learn 프레임워크를 사용하였다.

1. **딥러닝 모델**

코드 주소 : https://colab.research.google.com/drive/1WdLvZsw3Qi-n4w00skskLajL\_IH1\_lK0#scrollTo=X92datag8dky

pytorch를 사용하여 딥러닝 모델을 구축하였다.

train set : validation set은 8:2로 나누었고, test set은 train set의 10000개를 사용하였다



optimizer은 AdamW, loss function은 Mean Squared Error를 사용하였다. 세가지 모델을 실험해보았는데 4개의 층을 가진 input x 50 x 50 x 1 ,2개의 층을 가진 input x 100 x 1의 sequential model과 하나의 층을 가진 input x 1 linear model을 사용했다. 4개의 층을 가진 모델의 validation loss가 0.05까지 내려가서 실제 사용 후부로 생각하고 MSE값과 RMSE값을 구했다.



평균 오차는 3015 파운드 ≓ 440만원이었다. 나머지 모델들도 반복횟수를 늘려도 trainning loss와 validation loss가 0.25보다 작으며 종료하도록 했다. Linear model은 validation loss가 0.2478, 2층짜리 모델은 validation loss가 0.29가 나왔다.

1. **머신러닝 모델**

- **평가 방식**

Scikit-learn을 사용하여 머신러닝 모델을 구축하고 학습/평가하였다. train/test set을 두 가지 방식으로 나누어 진행하였다. 첫번째는 train\_test\_split()함수로 7:3비율로 나누어 진행했고, 두번째는 cross\_val\_score()함수를 통해 K-Fold 교차 검증 방식을 이용하였다. 최종적으로는 교차 검증 방식을 채택했다. 전자는 Overfitting이 발생할 우려가 있었고 train,test 셋을 나눌 때 랜덤하게 나누지만 고르게 데이터가 나뉘지 않을 가능성이 있다고 판단했다. 스태킹을 위한 최종모델에서는 GridSearchCV API를 사용하여 하이퍼 파라미터 튜닝과 교차검증을 한 번에 실시하였다.

**- 방법1**

- 하이퍼 파라미터 튜닝을 하기 전 결과 : 왼쪽부터 일반선형회귀, 릿지, 라소



- 하이퍼 파라미터 튜닝 후 결과 : 일반선형회귀, 릿지(alpha=0.001), 라소(alpha=0.0001)

****

- 릿지와 라소모델을 결합해서 예측한 결과 : 성능이 높은 Ridge에 가중치를 두었다.

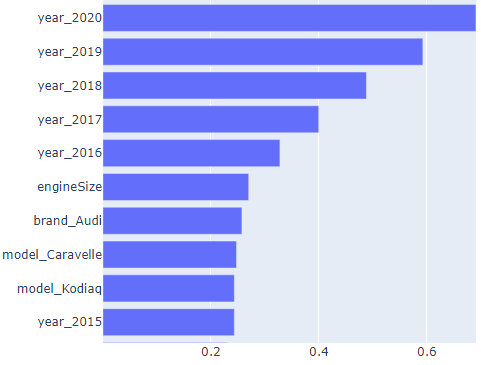
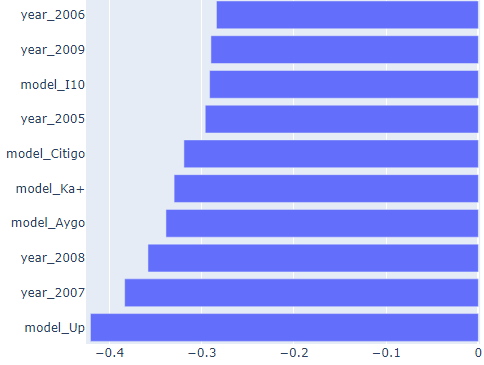


**- 방법 2 - GridSearchCV를 통해 K-Fold 교차검증과 파라미터 튜닝 진행**

- 결과

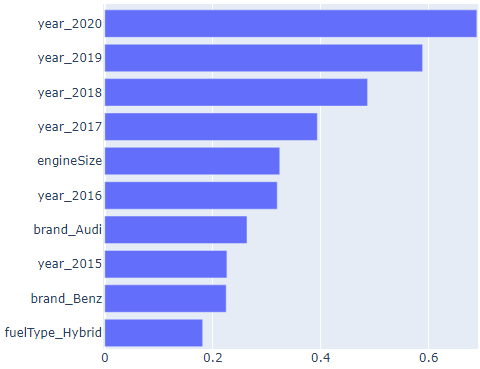
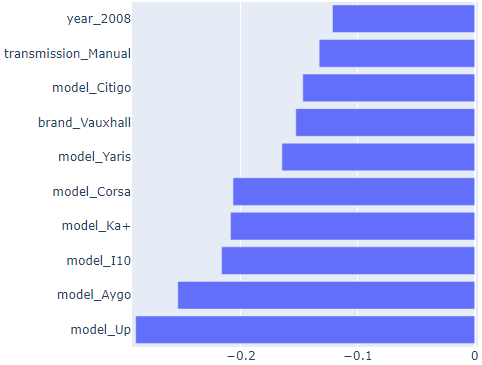


* Ridge 모델에서 Price에 영향을 주는 회귀 계수



🡪 연식과 engineSize, 그리고 특정 모델들이 Price에 큰 영향을 미치는 것을 알 수 있었다.

- Lasso 모델에서 Price에 영향을 주는 회귀 계수

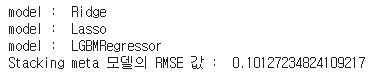


1. **스태킹 앙상블 모델**

회귀트리를 이용한 앙상블 모델인 LGBMRegressor로 다시 위 과정을 진행한 뒤, CV기반의 Stacking 기법을 활용해 최종 메타모델로 예측을 수행하였다. LightGBM Regressor의 최적 파라미터로는 {'learning\_rate': 0.05, 'max\_depth': 16, 'n\_estimators': 1000, 'subsample': 0.6}가 나왔고 RMSE값은 0.211이 나왔다.

스태킹 모델의 개별모델로는 Ridge, Lasso, LightGBM을 사용했고 최종 모델은 Lasso를 사용했다. 최종 스태킹 모델의 RMSE값은 약 0.101로 릿지, 라소 개별모델보다는 0.15정도의 향상이 있었고, LGBM앙상블 단일 모델보다 0.11 개선되었다. 최종 모델로 예측시 오차 금액 ≓ 184만원으로 예상된다.

* **결과**



**스태킹 앙상블 모델 구조**

예측값 #1

예측값 #2

개별모델

학습폴드 #1

학습폴드 #2

학습폴드 #3

학습폴드 #4

검증폴드 #5

예측값 #3

예측값 #4

예측값 #5

학습폴드 1,2,3,4 학습, 검증폴드 5로 예측한 값 저장

테스트 셋

#4

#5

학습폴드 1,2,3,4로 학습 후 테스트 셋으로 예측한 값 저장

메타 모델의

학습 데이터 생성

메타 모델의

테스트 데이터 생성

학습

예측

예측

#3

#2

#1

평균

5개 평균

개별 모델마다

5회 반복

Ridge

Ridge

Train\_Target

Lasso

LGBM

최종 메타 모델

Lasso

LGBM

Test

학습 피처 데이터 셋

학습 타겟

데이터 셋(원본)

테스트용 타겟 데이터 셋(원본)

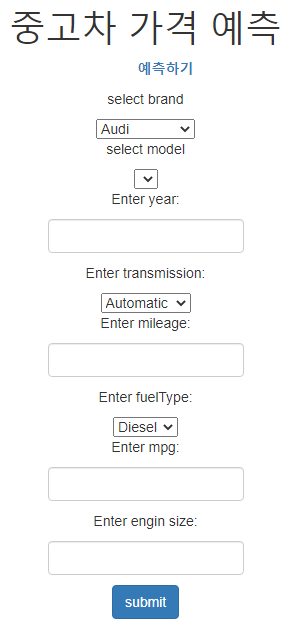
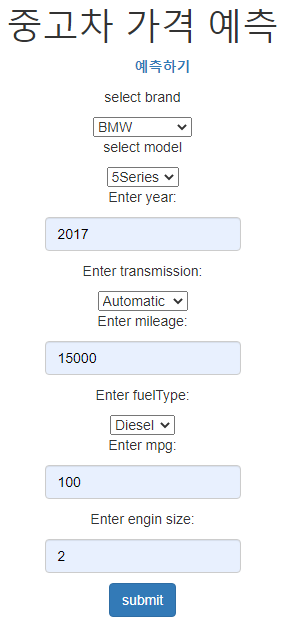
예측

평가

학습

테스트 데이터 셋

**4. 웹 구현**

입력과 결과 확인을 간편하게 할 수 있도록 예측할 자동차의 정보를 입력하고 예측 값을 띄워주는 웹페이지를 구현하였다. 파이썬으로 작성된 웹 프레임 워크인 Flask를 활용하여 구현하였다.

Carinfo.html 페이지에서 예측을 원하는 자동차의 정보를 입력하는 form이 생성된다.

Form에서 정보를 입력하고 예측하기를 누르면 입력한 정보들을 서버로 전송하여 머신 러닝 모델링 함수로 넘겨준다. 그 후 리턴 되는 예상 가격을 pred.html로 넘겨주어서 pred.html 페이지에 해당 예측 값이 보여 지도록 한다.



