ADX Project 1

중고차 시세 예측

Team 3. 이윤영, 김동현(19), 김동현(20), 서현우

DEPARTMENT OF
INDUSTRIAL ENGINERRING

목차

중고차 시장 현황

2024년 국내 중고차 시장은 약 380만대 규모이다.
그 중 B2C(기업과 소비자 간 거래)만 연 250만대 규모이다.(1)

[국내 중고차 시장의 문제점(500명 대상, 복수응답)](2)

구분	소비자
허위 매물	400 (79.8%)
불투명한 중고차 가격정보	71 (70.5%)
중고차 성능에 대한 낮은 신뢰도	296 (59.1%)
중고차 매물 비교정보 부족	259 (51.7%)
판매업자의 강매행위	189 (37.7%)

일반 소비자는 기업에 비해 "정보의 비대칭성"에 놓일 수 밖에 없다.
따라서 정확하고 공정한 중고차 가격의 측정이 필수적이다.

변수 파악

Y-DATA PROFILE을 활용한 DATA SUMMARY 확보

Overview | Alerts 134 | Reproduction

Dataset statistics

Number of variables	126
Number of observations	36794
Missing cells	8444
Missing cells (%)	0.2%
Duplicate rows	0
Duplicate rows (%)	0.0%
Total size in memory	35.4 MiB
Average record size in memory	1008.0 B

Variable types

Text	3
Numeric	14
Categorical	108
Boolean	1

변수 파악

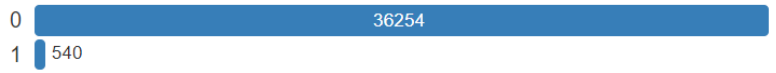
Y-DATA PROFILE을 활용한 DATA SUMMARY 확보

RIGHT_WHEEL_HOUSE_SHEETING

Categorical

IMBALANCE

Distinct	2
Distinct (%)	< 0.1%
Missing	0
Missing (%)	0.0%
Memory size	287.6 KiB



FRONT_RIGHT_FENDER has constant value "" Constant

FRONT_LEFT_DOOR has constant value "" Constant

BACK_RIGHT_DOOR has constant value ""

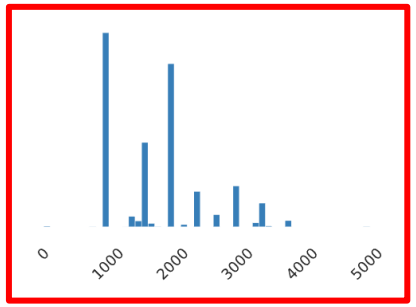
FRONT_PANNEL has constant value ""

EXHA

Real number (ℝ)

HIGH CORRELATION

Distinct	72	**Minimum**	0
Distinct (%)	0.2%	**Maximum**	5038
Missing	0	**Zeros**	3
Missing (%)	0.0%	**Zeros (%)**	< 0.1%
Infinite	0	**Negative**	0
Infinite (%)	0.0%	**Negative (%)**	0.0%
Mean	1784.0622	**Memory size**	287.6 KiB



추구한 방향성

- 변수 처리

임의로 변수를 제거하는 것은 위험하다고 판단했기에, 합칠 수 있는 변수들은 합쳐서 고려하고자 했음. 또한 중요성이 큰 변수들은 파생변수로 다루었음.

- Price 결측치 처리

Shipping Price > Nc grade price > New carprice로 이어지는 가격이 중요하다고 판단했고, 여기에 있는 결측 치를 어떻게 해결할 것인지 중점을 뒀음.

- 최종 목표

10% MAPE를 목표로 중고차 시세를 예측할 수 있는 모델을 만들고자 함.

동일한 사고 부위 병합하기

팬더, 휠, 필러 등 동일 부위에 대한 교환/판금/용접인 경우 변수를 하나로 통합했습니다.

```
parts_to_combine = [
    ('FRONT_LEFT_FENDER', 'FRONT_RIGHT_FENDER', 'FENDER_REPLACE'),
    ('FRONT_LEFT_DOOR', 'FRONT_RIGHT_DOOR', 'DOOR_REPLACE'),
    ('BACK_LEFT_DOOR', 'BACK_RIGHT_DOOR', 'BACK_DOOR_REPLACE'),
    ('LEFT_STEP', 'RIGHT_STEP', 'STEP_REPLACE'),
    ('LEFT_FILER_A', 'RIGHT_FILER_A', 'A_FILER_REPLACE'),
    ('LEFT_FILER_B', 'RIGHT_FILER_B', 'B_FILER_REPLACE'),
    ('LEFT_FILER_C', 'RIGHT_FILER_C', 'C_FILER_REPLACE'),
    ('LEFT_REAR_FENDER', 'RIGHT_REAR_FENDER', 'REAR_FENDER_REPLACE'),
    ('LEFT_INSIDE_PANEL', 'RIGHT_INSIDE_PANEL', 'INSIDE_PANEL_REPLACE'),
    ('LEFT_WHEEL_HOUSE', 'RIGHT_WHEEL_HOUSE', 'WHEEL_HOUSE_REPLACE'),
    ('LEFT_INSIDE_WHEEL_HOUSE', 'RIGHT_INSIDE_WHEEL_HOUSE', 'INSIDE_WHEEL_HOUSE_REPLACE'),
    ('LEFT_REAR_WHEEL_HOUSE', 'RIGHT_REAR_WHEEL_HOUSE', 'REAR_WHEEL_HOUSE_REPLACE'),
    ('LEFT_QUARTER', 'RIGHT_QUARTER', 'QUARTER_REPLACE'),
    ('LEFT_SIDE_PANEL', 'RIGHT_SIDE_PANEL', 'SIDE_PANEL_REPLACE'),
    ('LEFT_REAR_CORNER_PANEL', 'RIGHT_REAR_CORNER_PANEL', 'REAR_CORNER_PANEL_REPLACE'),
    ('LEFT_CORNER_PANEL', 'RIGHT_CORNER_PANEL', 'CORNER_PANEL_REPLACE'),
    ('LEFT_SKIRT_PANEL', 'RIGHT_SKIRT_PANEL', 'SKIRT_PANEL_REPLACE'),
    ('LEFT_INSIDE_SHEETING', 'RIGHT_INSIDE_SHEETING', 'INSIDE_SHEETING_REPLACE'),
    ('LEFT_WHEEL_HOUSE_SHEETING', 'RIGHT_WHEEL_HOUSE_SHEETING', 'WHEEL_HOUSE_SHEETING_REPLACE'),
    ('LEFT_REAR_INSIDE_PANEL_SHEETING', 'RIGHT_REAR_INSIDE_PANEL_SHEETING', 'REAR_INSIDE_PANEL_SHEETING_REPLACE'),
    ('LEFT_REAR_WHEEL_HOUSE_SHEETING', 'RIGHT_REAR_WHEEL_HOUSE_SHEETING', 'REAR_WHEEL_HOUSE_SHEETING_REPLACE'),
    ('LEFT_SIDE_PANEL_SHEETING', 'RIGHT_SIDE_PANEL_SHEETING', 'SIDE_PANEL_SHEETING_REPLACE'),
]
```

단일 값만 존재하는 열 삭제

값이 (0)만 존재하는 열은 중고차 가격을 예측하는데 무의미할 것으로 판단했습니다.

```
# 한 column에 값이 하나 뿐이면은 drop 함
columns_to_drop = [col for col in df2.columns if df2[col].nunique() == 1]
df2.drop(columns_to_drop, axis=1, inplace=True)
```

```
columns_to_drop
```

```
['FRONT_PANNEL',
 'BACK_PANEL1',
 'DASH_PANEL',
 'SHEET_PANEL',
 'SIDE_MEMBER_FRAME2',
 'JOINCAR',
 'NOTAVAILABLE',
 'MF_KEY',
 'SUNLOOPDUAL',
 'ETC',
 'EPS',
 'A_FILER_REPLACE',
 'B_FILER_REPLACE',
 'WHEEL_HOUSE_REPLACE',
 'QUARTER_REPLACE',
 'REAR_CORNER_PANEL_REPLACE',
 'CORNER_PANEL_REPLACE',
 'SKIRT_PANEL_REPLACE',
 'SIDE_PANEL_SHEETING_REPLACE']
```

가격 변수 결측치 제거

출고가, 신차등급가격 그리고 신차금액 데이터 중에서 0 또는 1인 데이터를 Nan 값으로 대체하였습니다.

```
# 극단적인 값 제거
df2['SHIPPING_PRICE'] = df2['SHIPPING_PRICE'].replace({'': np.nan, 0: np.nan})
df2['NC_GRADE_PRICE'] = df2['NC_GRADE_PRICE'].replace({'': np.nan, 0: np.nan})
```

```
##  NEWCARPRIC 극단적인 값 제거
df2['NEWCARPRIC'] = df2['NEWCARPRIC'].replace([0, 1], np.nan)
```

범주형 변수 처리하기

라벨 인코딩을 통해 범주형 데이터를 수치형 변수로 변환했습니다.

```
def encode_MISSNM(x):
  if x == 'A/T':
    return 0
  elif x == 'M/T':
    return 1
  else: return 2

def encode_FUELNM(x):
  if x == 'LPG': return 0
  elif x == '가솔린': return 1
  elif x == 'Hybrid': return 2
  elif x == '디젤': return 3
  elif x == '겸용': return 4
  elif x == '전기': return 5
  else: return 1 # 최빈값이 가솔린이므로 nan값이 들어올 경우 1을 리턴

def encode_COLOR(x):
  key_vals = {'A':0, 'B':1, 'C':2, 'D':3, 'F': 4}
  return key_vals[x]
```

```
def encode_USEUSENM(x):
  if x == '자가': return 0
  elif x == '렌트': return 1
  elif x == '업무': return 2
  elif x == '리스': return 3
  elif x == '사업': return 4
  else: return 1 # 최빈값이 렌트이므로 nan값이 들어올 경우 1을 리턴

def encode_OWNECLASNM(x):
  if x == '법인': return 0
  elif x == '개인': return 1
  elif x == '법인상품': return 2
  elif x == '상품용': return 3
  elif x == '재외국인': return 4
  elif x == '개인사업': return 5
  elif x == '종교단체': return 6
  else: return 0 # 최빈값이 법인이므로 nan값이 들어올 경우 0을 리턴

def encode_INNEEXPOCLASCD_YN(x):
  if x == 'X': return 0
  else: return 1

def encode_YEARCHK(x):
  if x == 'N': return 0
  else: return 1
```

파생변수 두 가지를 추가

중고차 도메인에 대한 이해를 바탕으로 파생변수 두 가지를 생성했습니다.

- real_used : 실사용 일 수
- CRITICAL_ACCIDENT : 감가에 치명적인 사고

```
#파생변수 추가
SUCCYMD_datetime = pd.to_datetime(df_test['SUCCYMD'], format='%Y%m%d', errors='coerce')
CARREGIYMD_datetime = pd.to_datetime(df_test['CARREGIYMD'], format='%Y%m%d', errors='coerce')

real_used = SUCCYMD_datetime - CARREGIYMD_datetime
df_test['real_used'] = real_used.dt.days
```

```
# 파생변수 하나 더 추가 Critical Accident
critical = ['TRUNK_FLOOR', 'TRUNK_FLOOR_SHEETING', 'C_FILER_REPLACE', 'REAR_FENDER_REPLACE']
df_test['CRITICAL_ACCIDENT'] = df_test[critical].apply(lambda row: (row > 0).any(), axis=1).astype(int)
df_test.columns
```

가격 변수 결측치 대체

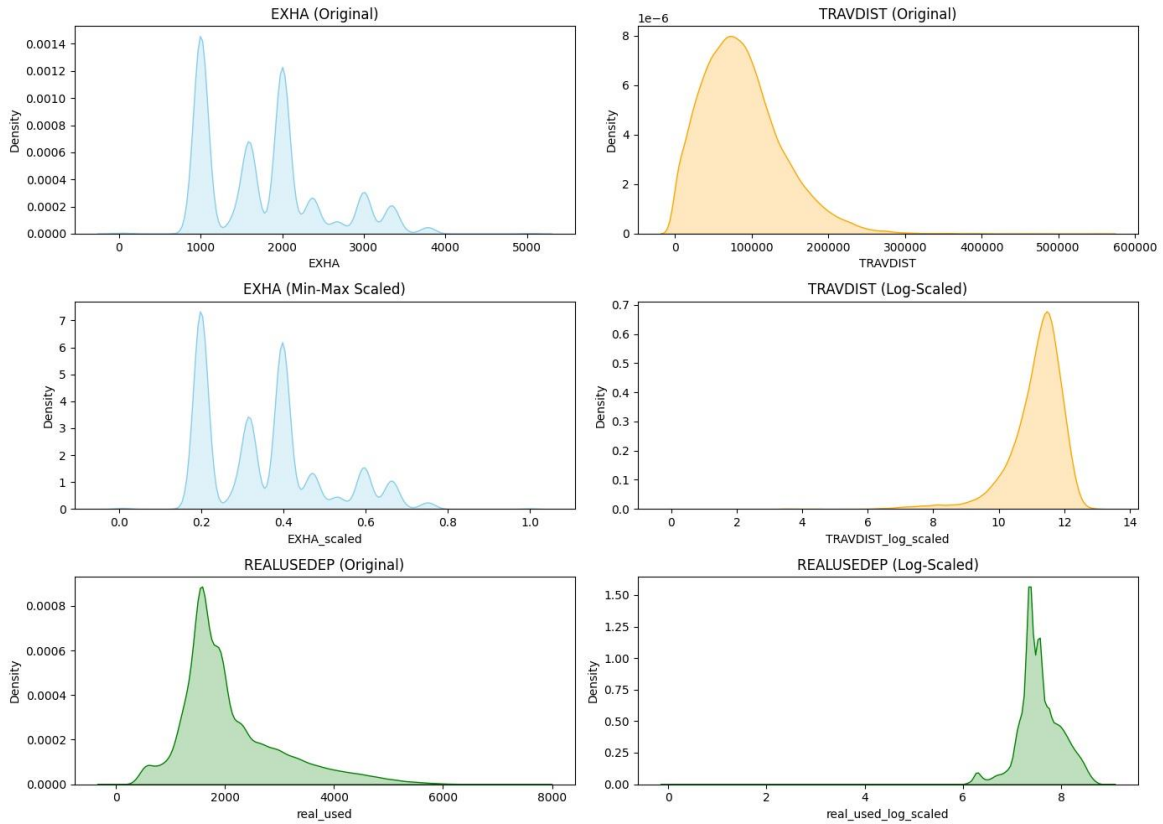
3.3에서 제거한 0 또는 1의 출고가, 신차등급가격 그리고 신차금액에 대하여
랜덤 포레스트 모델을 이용해 값을 예측하고, 결측치를 대체했습니다.

- NEWCARPRIC
- NC_GRADE_PRICE
- SHIPPING_PRICE

```
### NEWCARPRIC, 랜덤 포레스트로 가격 예측해서 대체하기
import pandas as pd
from sklearn.ensemble import RandomForestRegressor
known_newcarpric = df_test.dropna(subset=['NEWCARPRIC'])
unknown_newcarpric = df_test[df_test['NEWCARPRIC'].isna()]

features_columns = known_newcarpric.select_dtypes(include=[np.number]).columns.drop('NEWCARPRIC').tolist()
features_columns = [col for col in features_columns if col not in ["GOODNO", "CARNM", "CHASNO", "SHIPPING_PRICE", "NC_GRADE_PRICE", "SUCCPRIC"]]

features = known_newcarpric[features_columns]
target = known_newcarpric['NEWCARPRIC']



features_unknown = unknown_newcarpric[features_columns]
predicted_newcarpric = model_1.predict(features_unknown)


df_test.loc[df_test['NEWCARPRIC'].isna(), 'NEWCARPRIC'] = predicted_newcarpric
```

수치형 데이터 정규화

수치형 데이터를 정규화 하여 모델의 성능을 향상시키고자,
각각의 분포를 고려하여 minmax scaler 또는 log scaler를 사용했습니다.

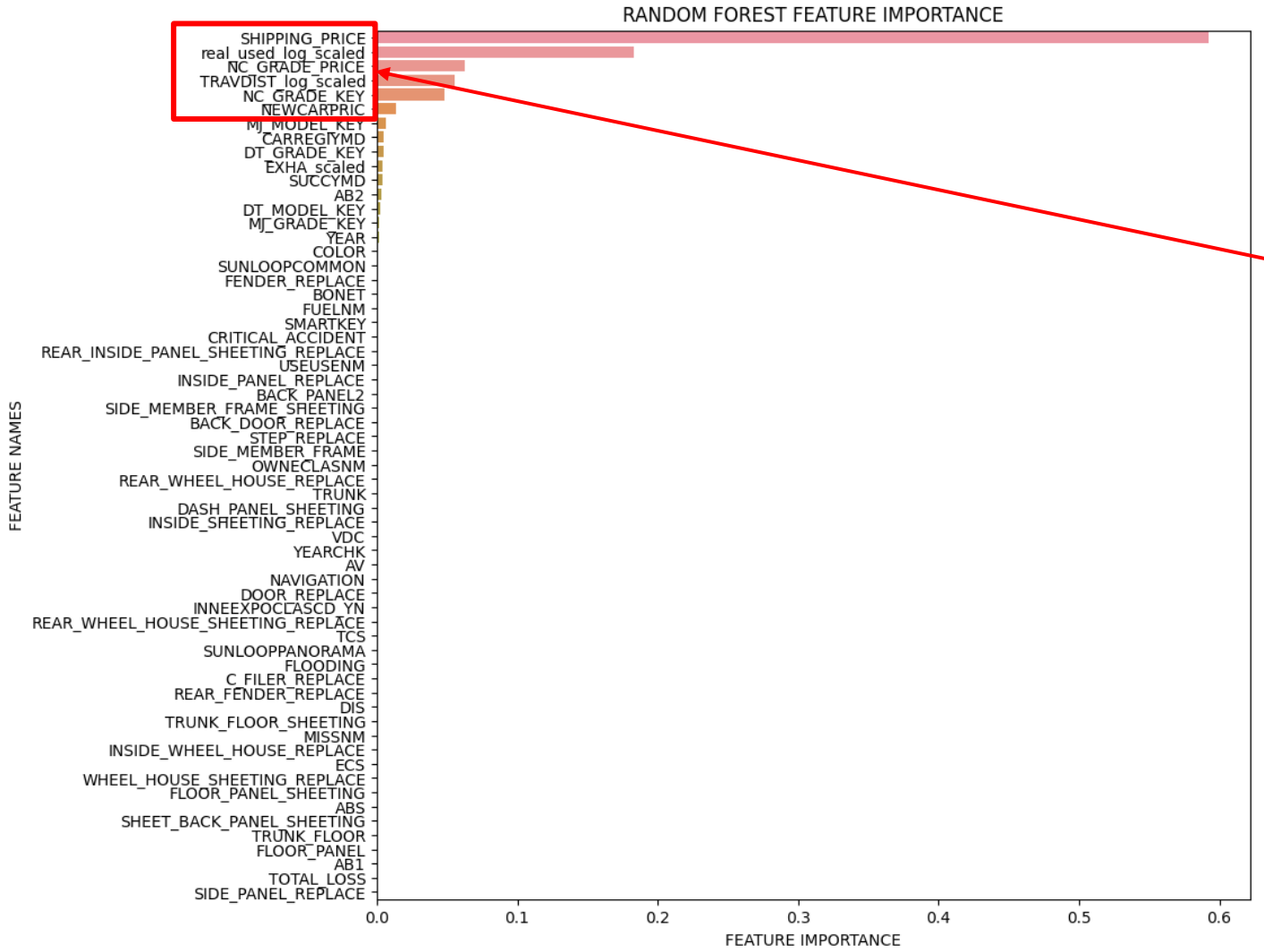


랜덤 포레스트 모델

최적의 하이퍼 파라미터를 선정하고 모델을 학습시켰습니다.
이때, 중요도가 가장 높은 변수는 출고가, 실사용 일수, 신차등급가격, 주행거리 등 이었습니다.

```
# 랜덤 포레스트 돌리기
for n_estimator in n_estimators:
    for max_depth in max_depths:

        rf = RandomForestRegressor(n_estimators=n_estimator, max_depth=max_depth, random_state=42)
        rf.fit(X_train, y_train)
        y_pred = rf.predict(X_val)

        # MSE 계산
        mse = mean_squared_error(y_val, y_pred)

        results[(n_estimator, max_depth)] = mse

        if mse < best_mse:
            best_mse = mse
            best_rf = rf
            best_hyperparameter = (n_estimator, max_depth)


print(f"Best Hyperparameters: {best_hyperparameter}, Best MSE: {best_mse}")
```

랜덤 포레스트 모델



① SHIPPING_PRICE
② real_used_log_scaled
③ NC_GRADE_PRICE
④ TRAVDIST_log_scaled
⑤ NEWCARTPRIC

Implementation 과정

- 데이터 불러오기
- 변수처리 (차원 축소, 라벨 인코딩, 파생변수 추가)
- 신차금액, 신차등급가격, 출고가 순으로 결측치 대체
- 정규화

Train/Test Prediction MAPE

Random Forest : 7.03%
Ada Boost : 10.31%
Light GBM : 7.87%

Implementation Prediction MAPE

Random Forest : 11.04%
Ada Boost : 12.68%
Light GBM : 10.97%

Conclusion

한계점

- **변수들 간의 상관 관계**

correlation이 높은 변수들이 있었고 이에 대해 처리를 하지 못했습니다.

AB1	is highly overall correlated with AB2 and 1 other fields
AB2	is highly overall correlated with AB1 and 1 other fields
ABS	is highly overall correlated with AB1 and 1 other fields
AV	is highly overall correlated with NAVIGATION
BACK_PANEL2	is highly overall correlated with TRUNK and 1 other fields
CARREGIYMD	is highly overall correlated with DT_GRADE_KEY and 5 other fields
DT_GRADE_KEY	is highly overall correlated with CARREGIYMD and 6 other fields

개선안

수치형 변수에 대해 PCA, 이진형 변수에 대해 MCA를 적용하여 차원을 축소하면서 high correlation 문제를 해결할 수 있습니다.

DEPARTMENT OF
INDUSTRIAL ENGINERRING

Q&A