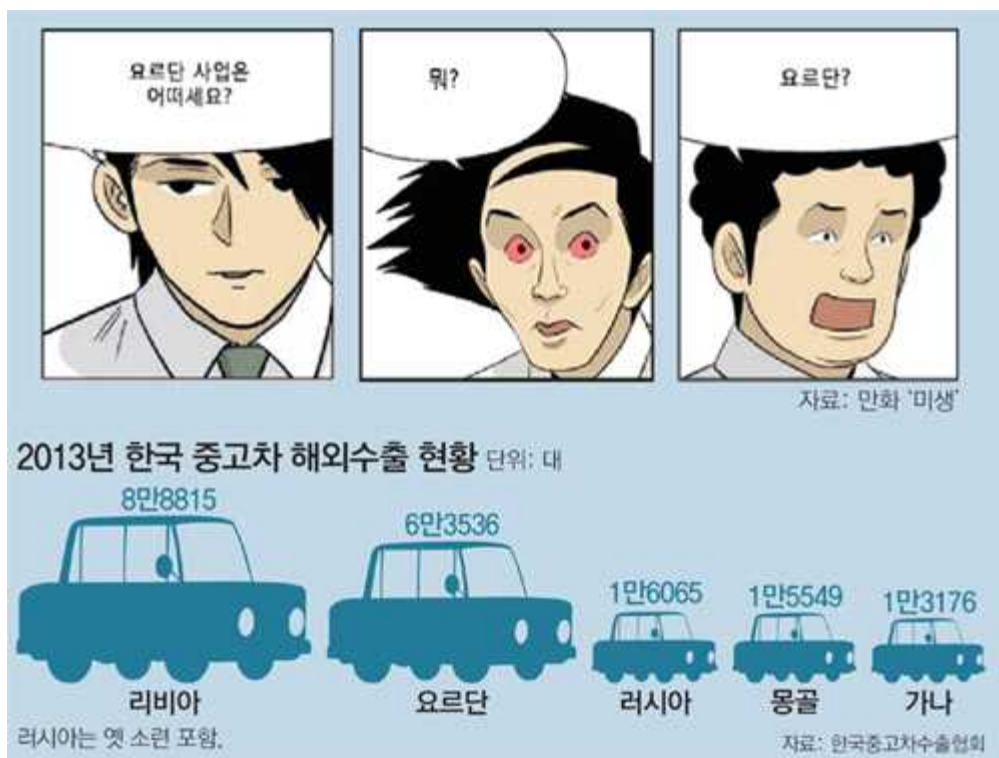


실속있게 중고차를 사고 팔기위한 중고차 가격 예측 모델



- 웹툰 미생 中

국내에서의 중고차 매매뿐만 아니라, 해외로의 중고차 수입 및 수출을 더욱더 실속있게 하기 위한 중고차 가격 예측 모델을 만들어 보려고 한다. 예측 모델을 이용하면 앞으로 중고차 매매에 있어서 딜러의 도움 없이 이러한 과정을 진행을 할 수 있을지에 대해 확인해 보겠다.

목차

1. 개요 및 분석동기	3 p
1.1. 개요	
1.2. 분석동기	
1.3. 사용도구	
2. 데이터 수집	4 p
2.1. 데이터	
2.2. 가격에 대한 가설	
2.3. 데이터 도수 분포	
2.4. 데이터 처리 과정	
3. 분석 과정	9 p
3.1. 데이터 상관관계 분석	
3.2. 예측 결과에 영향이 없는 변수 제거	
3.3. 학습데이터 / 테스트 데이터 생성	
3.4. 학습 과정	
4. 분석 결과의 정확도 분석	13 p
4.1. 정확도 분석	
4.2. 오차율 분석	
5. 분석 결과의 시사점 및 결론	14 p

1. 개요 및 분석동기

1.1. 개요

- 중고차 거래 게시판의 중고차 데이터를 기반으로 Machine Learning을 사용하여 중고차 가격을 예측한다.
- 중고차 가격에 영향을 미치는 변수를 분석한다.
- 결과로부터 가설에서 언급한 변수와 실제로 영향을 미치는 변수들을 비교한다.

1.2. 분석동기

- 데이터를 쉽게 구할 수 있다.
- 변수를 분석하여 다양한 데이터 전처리를 적용해볼 수 있다.
- 가격 예측의 오차범위를 구해볼 수 있고, 오차범위를 줄이기 위한 다양한 방법을 적용해볼 수 있다.
- 중고차 매매 건수가 해마다 증가하고 있는 추세로 온라인 중고차 시장에 합리적으로 중고차 매매를 할 수 있도록 도움을 줄 수 있는 데이터 분석이 필요하다고 판단하였다.

연도별 중고차 매매 건수



연도별 중고차 시장 규모 (단위=만대)	중고차 거래 온라인 플랫폼		
		특징	신뢰도 확보 방법
2010년	273	SK 엔카닷컴	국내 거래 매출 중 3분의1 등록 진단평가사가 사고 유무, 차량 등급 평가
2011년	325	중고나라	국내 최대 온라인 중고물품 매매 플랫폼
2012년	328	인중 중고차	5단계 검증 통해 '인중 딜러' 자격 부여
2013년	330	첫차	첫차를 사는 20~30대를 위한 중고차 가이드 하위 매물 필터링 프로그램 구축
2014년	340	신한카드 차투차	제휴할인쿠폰 제공으로 최대 200만원 할인 자체 트레이닝 영원팀 상담
2015년	366	KB 차차차	고객이 차를 판매할 때 매도가격 보장 빅데이터 분석으로 시세 제공
2016년	378		

*자료=국토교통부

1.3. 사용 도구

- 데이터 분석 및 처리에는 R과 Python, Excel을 사용하였다.
- 모델 학습 및 테스트에는 TensorFlow를 사용하였다.

2. 데이터 수집

2.1. 데이터

데이터는 장기간 동안 수집한 데이터가 필요하여 이번 프로젝트에 사용하기에 적합한 데이터를 탐색하다가 ebay 중고차 거래 게시글을 3개월 간 수집한 데이터를 사용하기로 하였다.

데이터 예시

dateCrawled	name	seller	offerType	price	abtest	vehicleType	yearOfReg	gearbox	powerPS	model
2016-03-24 10:58	A5_Sportb	privat	Angebot	18300	test	coupe	2011	Manual	190	
2016-03-14 12:52	Jeep_Gran	privat	Angebot	9800	test	suv	2004	Automatic	163	grand
2016-03-17 16:54	GOLF_4_1	privat	Angebot	1500	test	kleinwage	2001	Manual	75	golf
2016-03-31 17:25	Skoda_Fab	privat	Angebot	3600	test	kleinwage	2008	Manual	69	fabia
2016-04-04 17:36	BMW_316i	privat	Angebot	650	test	limousine	1995	Manual	102	3er

kilometer	monthOfReg	fuelType	brand	notRepairedDamage	dateCreated	nrOfPictures	postalCode	lastSeen
125000	5	diesel	audi	ja	2016-03-24 0:00	0	66954	2016-04-07 1:46
125000	8	diesel	jeep		2016-03-14 0:00	0	90480	2016-04-05 12:47
150000	6	benzin	volkswage	nein	2016-03-17 0:00	0	91074	2016-03-17 17:40
90000	7	diesel	skoda	nein	2016-03-31 0:00	0	60437	2016-04-06 10:17
150000	10	benzin	bmw	ja	2016-04-04 0:00	0	33775	2016-04-06 19:17

데이터 출처 : <https://gist.github.com/vladislaveme>

데이터의 변수 이름과 타입 및 설명

feature name	raw type	description
dateCrawled	string	데이터 수집 일시
name	string	게시글 제목
seller	string	판매자 종류 (private, public)
offerType	string	판매 방식
price	int	가격
abtest	string	튜닝 여부(test-튜닝, control-일반)
vehicleType	string	차종
yearOfRegistration	int	연식
gearbox	string	기어 방식(Automatic, Manual)
powerPS	int	마력
model	string	차 모델 이름
kilometer	int	주행거리
monthOfRegistration	int	차량 등록 월
fuelType	string	연료
brand	string	브랜드명
notRepairedDamage	string	수리했는지(yes, no)
dateCreated	string	작성일
nrOfPictures	int	광고 사진 개수
postalCode	string	지역번호 (=우편번호)
lastSeen	string	마지막 수정 일시

해당 데이터는 총 20개의 열로 되어 있으며 feature들을 처리하여 price를 예측하는 모델을 만들어 볼 예정이다.

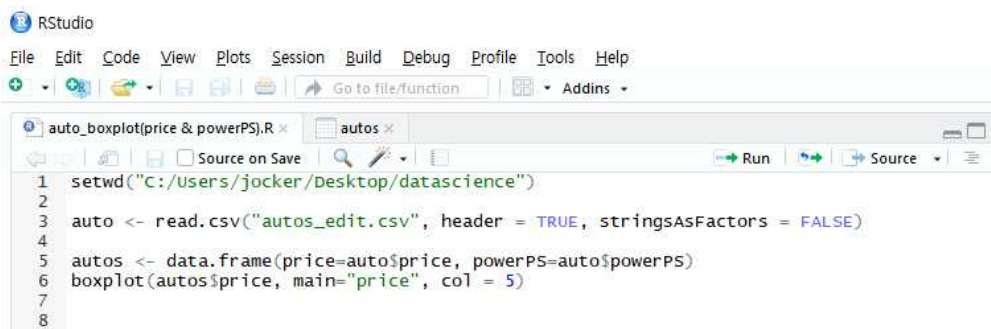
2.2. 가격에 대한 가설

데이터를 전 처리하기 전에 가격 예측에 중요한 데이터라고 생각되는 feature들을 가정하였다.

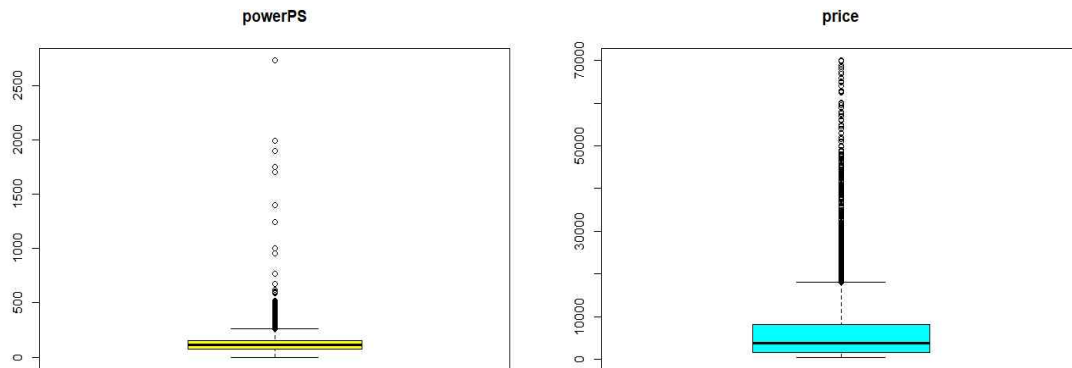
- 마력이 높을수록 가격이 높을 것이다
마력은 자동차의 성능에 중요한 지표이므로 사람들이 가성비를 볼 때 가장 많이 따져볼 것이다.
- 지붕이 열리는 차(cabrio)가 고정형인 차보다 가격이 높을 것이다.
지붕이 열리는 차를 고급차로 인식되는 경우가 많아 차종 중에서도 cabrio인 차의 가격대를 높게 설정할 것이라고 예상된다.
- 수리한 경력이 있는 차의 가격이 낮을 것이다.
중고차를 구매할 때 가장 많이 살펴볼 것이 어느 곳에 어떤 결함이 있는지 여부이다. 판매자가 판매하는 차량이 결함이 있으면 일부러 가격을 낮추므로 가격에 많은 영향을 줄 것으로 예상된다.
- 주행거리가 높을수록 가격이 낮을 것이다.
주행거리 역시 중고차를 구매할 때 많이 보는 지표로, 운행량을 통해 대략적인 차의 상태를 알 수 있어 가격에 적지 않은 영향을 줄 거라 예상된다.
- 연식이 오래된 차량일수록 가격이 낮을 것이다.
연식이 오래된 차량일수록 차량에 결함이 생길 확률이 높아지고, 차량의 부품 또한 낡아 있을 가능성 때문에 가격에 적지 않은 영향을 줄 거라 예상된다.

2.3. 데이터 도수 분포

도수 분포는 R의 boxplot 함수를 사용하여 가격과 마력에 해당하는 Box Plot을 그려서 확인해봤다.



```
1 setwd("C:/Users/jocker/Desktop/datascience")
2
3 auto <- read.csv("autos_edit.csv", header = TRUE, stringsAsFactors = FALSE)
4
5 autos <- data.frame(price=auto$price, powerPS=auto$powerPS)
6 boxplot(autos$price, main="price", col = 5)
7
8
```



2.4. 데이터 처리과정

feature name	raw type	processing
dateCrawled	string	열 삭제 (필요없음)
name	string	열 삭제 (brand, model로 대체)
seller	string	열 삭제 (값이 1종류 밖에 없음)
offerType	string	열 삭제 (값이 1종류 밖에 없음)
price	int	정규화
abtest	string	one-hot encoding
vehicleType	string	결측치 처리, one-hot encoding
yearOfRegistration	int	정규화
gearbox	string	one-hot encoding
powerPS	int	결측치 처리, 이상치 처리, 정규화
model	string	결측치 처리, one-hot encoding
kilometer	int	정규화
monthOfRegistration	int	열 삭제
fuelType	string	one-hot encoding
brand	string	one-hot encoding
notRepairedDamage	string	1 or 0으로 변환
dateCreated	string	seconds로 변환
nrOfPictures	int	열 삭제 (필요없음)
postalCode	string	열 삭제 (필요없음)
lastSeen	string	열 삭제 (필요없음)

• 불필요한 Feature 제거

- 값이 1종류 밖에 없거나 당연히 불필요하다고 생각되는 feature는 제거하였다.
- dateCrawled, postalCode, lastSeen : 가격과 연관이 없을 것으로 생각하여 제거하였다.
- name : 게시글 제목으로 차종, 브랜드 등의 정보가 포함되어 있지만 이미 해당 Feature가 존재하므로 삭제해도 무방하다.
- seller, offerType, nrOfPictures : 판매자 종류, 판매 방식에 대한 값은 1종류 밖에 없어 무의미하여 제거한다. 광고 사진 개수는 크롤링이 제대로 되지 않았는지 모든 값이 0이므로

제거하였다.

- monthOfRegistration : 우선 결측치가 많고 비슷한 Feature인 연식에 비해 가격에 미치는 영향이 적을 것으로

• 이상치 처리

Box Plot에서 관측되었던 이상치를 처리한다.

- 가격 : 특정 값들이 오차범위 밖에 있지만, 밀집해있어 데이터 부족에 의한 것으로 판단하여 따로 이상치 처리를 하지 않기로 하였다.

- powerPS(마력) : 중고차 목록에 슈퍼카가 존재하지 않는다는 사실로부터 약 750마력이 넘는 차량의 데이터를 이상치로 추정하였다. 이상치를 가진 대부분의 데이터가 뒤에 의미 없는 숫자 한자리가 더 있거나 빠져있는 경우가 있어 name 필드에 적힌 마력을 참고하거나 브랜드명, 모델명 등으로 검색하여 정확한 값을 채웠다. 추가로 소형차 또는 전기차 이외의 마력이 5이하인 데이터는 정확하지 않은 데이터로 보고 제거하였다.

• 결측치 처리

필드 별 결측치 개수

vehicleType	powerPS	model	monthOfRegistration	fuelType	notRepairedDamage	nrOfPictures
291	1191	288	1070	850	2644	20001

- 차종, 모델 : 1차로 name 필드에 적힌 차종/모델을 참고하였고 나머지는 제거하였다.

- powerPS(마력) : 마력이 0으로 표기된 차량의 경우 모두를 결측치로 처리하여 데이터를 제거하면 안 된다고 판단이 되었다. 마력이 표기된 차량 중에서 마력이 0으로 표기된 차량과 이름이 같고 동시에 연식이 같은 차량의 데이터를 참고하여 마력의 값을 채워 넣었다. 차량의 이름을 잘 알아보지 못하거나, 참고할 만한 차량의 데이터가 없는 400여대의 차량은 제거하였다.

- monthOfRegistration(출시월) : 각각의 출시 월에 대한 데이터를 따로 구하기 힘들고 비슷한 필드인 연식에 비해 가격에 주는 영향도 적다고 생각되어 출시월 필드를 제거하였다.

- 연료 : 아래 표와 같이 차종 별로 잘 쓰이는 연료형식으로 채워 넣었다.

차량 유형	연료 유형
SUV, Kombi, Bus, 리무진 (중·대형차)	디젤
Cabrio, 쿠페, 소형차 (소형차)	가솔린

- notRepairedDamage(수리여부) : 중고 차량의 가격을 결정하는 요인에 있어서 사고차량인지 여부가 매우 중요한 특징이라고 생각하였다. 그렇기 때문에 사고 차량인지 여부를 판별하는 데이터에 결측치가 있는 경우, 해당 결측치를 단순히 삭제하는 것이 아닌 무사고 상태로 채워넣기로 결정하였다. 무사고 차량의 경우 사고유무에 대해 따로 표기를 해야 할 필요성을 느끼지 못해 사고유무를 적지 않았을 가능성이 매우 높기 때문에 그러한 판단을 하였

으며, 반대로 중고 차량을 판매 할 때 사고가 난 차량의 경우, 차량의 주인은 구매자들에게 사고가 난 차량이라는 사실을 어떠한 방법으로든 알려야 차량을 판매하는데 있어서 문제가 생기는 것을 방지 할 수 있다고 생각했기 때문이다. 따라서 사고유무에 대해 따로 표기가 안 되어있는 차량은 무사고차량으로 표기하여 처리하였다.

- nrOfPictures(광고사진 개수) : 전체 데이터가 0으로 되어 있어 광고사진 개수 필드를 제거하였다.

• 데이터 전처리 적용

위에서 정한 내용을 바탕으로 데이터를 추가한 csv 파일을 DataFrame으로 읽어, 사용하지 않는 데이터나 열을 삭제하였다. 그 결과 [20001, 20] 크기의 데이터가 [18623, 11]으로 줄어들었다.

```
import pandas as pd
from sklearn.ensemble import RandomForestClassifier

csv = pd.read_csv('autos.csv', encoding='latin-1')
print(">> raw csv: ", len(csv), len(csv.columns.values))

# 필요없는 Feature를 제거하고, 결측치를 채운 csv 로드
csv = pd.read_csv('autos-missing_value_processed.csv')
print(">> missing value processed csv: ", len(csv), len(csv.columns.values))

# 남은 결측치를 가진 행 제거
csv = csv.dropna()
print(">> csv after drop na: ", len(csv), len(csv.columns.values))

# 마력(powerPS)이 5 미만의 행 제거
csv = csv[csv['powerPS'] > 5]
print(">> csv after drop min irregular powerPS: ", len(csv), len(csv.columns.values))

>> raw csv: 20001 20
>> missing value processed csv: 20001 11
>> csv after drop na: 18958 11
>> csv after drop min irregular powerPS: 18623 11
```

• Int 형 변수 정규화

price, powerPS, kilometer, yearOfRegistration 등의 변수는 minmax scale로 0~1 사이의 값으로 정규화하여 나타낸다.

```
# 정규화
cdata['price'] = minmax_scale(cdata['price'].astype(float))
cdata['powerPS'] = minmax_scale(cdata['powerPS'].astype(float))
cdata['kilometer'] = minmax_scale(cdata['kilometer'].astype(float))
cdata['yearOfRegistration'] = minmax_scale(cdata['yearOfRegistration'].astype(float))
#cdata['monthOfRegistration'] = minmax_scale(cdata['monthOfRegistration'].astype(float))
```

• Yes/No 변수 1 또는 0으로 변환

notRepairedDamage는 Yes(ja)이면 1, No(nein)이면 0으로 변환하였다.


```
# boolean 변환
cdata['notRepairedDamage'] = cdata['notRepairedDamage'].mask(cdata['notRepairedDamage'] == 'ja', 1)
                        .mask(cdata['notRepairedDamage'] == 'nein', 0)
```

- 범주형 변수 One-hot encoding

범주형 변수는 각각의 값에 대한 필드를 만들고 0, 1로 값을 표시하는 One-hot encoding을 사용하였다. pd.get_dummies로 새로운 필드를 만들어 기존 데이터에 join시키고 기존의 범주형 변수의 필드를 삭제하는 방식으로 구현했다. 처리 결과 [18623, 11]의 데이터는 다시 [18623 56]으로 필드 개수가 많이 늘어났다.

```
# one-hot encoding
cdata = cdata.join(pd.get_dummies(cdata['gearbox'], prefix='gearbox'))
                .join(pd.get_dummies(cdata['fuelType'], prefix='fuelType'))
                .join(pd.get_dummies(cdata['brand'], prefix='brand'))

cdata = cdata.drop('gearbox', 1)
                .drop('fuelType', 1)
                .drop('brand', 1)
print(">> csv after onehot: ", len(cdata), len(cdata.columns.values))
```

3. 분석 과정

3.1. 데이터 상관관계 분석

price에 미치는 영향이 적은 변수를 분석하기 위해 상관관계 분석을 실시하였다. 상관관계 분석에 사용한 데이터는 One-hot encoding이 처리된 데이터의 경우 필드 개수가 너무 많아져 내용을 확인하기 어려워진다. 따라서 원본 데이터를 사용하되, 범주형의 경우 LabelEncoding을 하여 진행하였다.

```
from sklearn import preprocessing, svm
from sklearn.preprocessing import StandardScaler, Normalizer

cor_csv = csv.copy()
columns = list(cor_csv.columns)
labels = ['abtest', 'vehicleType', 'gearbox', 'model', 'fuelType', 'brand', 'notRepairedDamage']
les= {}
for l in labels:
    les[l] = preprocessing.LabelEncoder()
    les[l].fit(cor_csv[l])
    tr = les[l].transform(cor_csv[l])
    cor_csv.loc[:, l + '_label'] = pd.Series(tr, index=cor_csv.index)

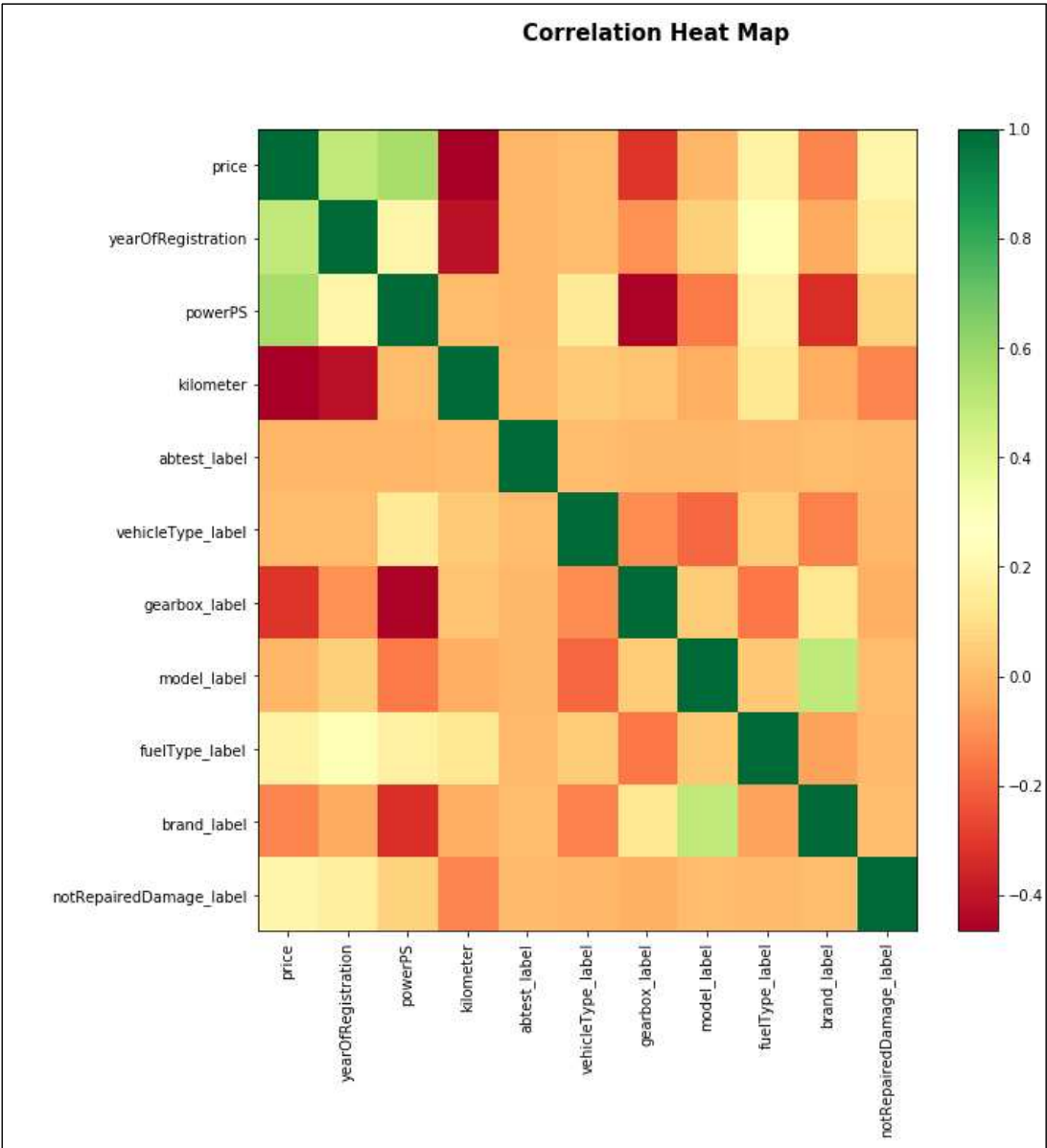
cor_csv = cor_csv[columns+[x+'_label' for x in labels]]
cor_csv.corr()
```

3.2. 데이터 분석결과

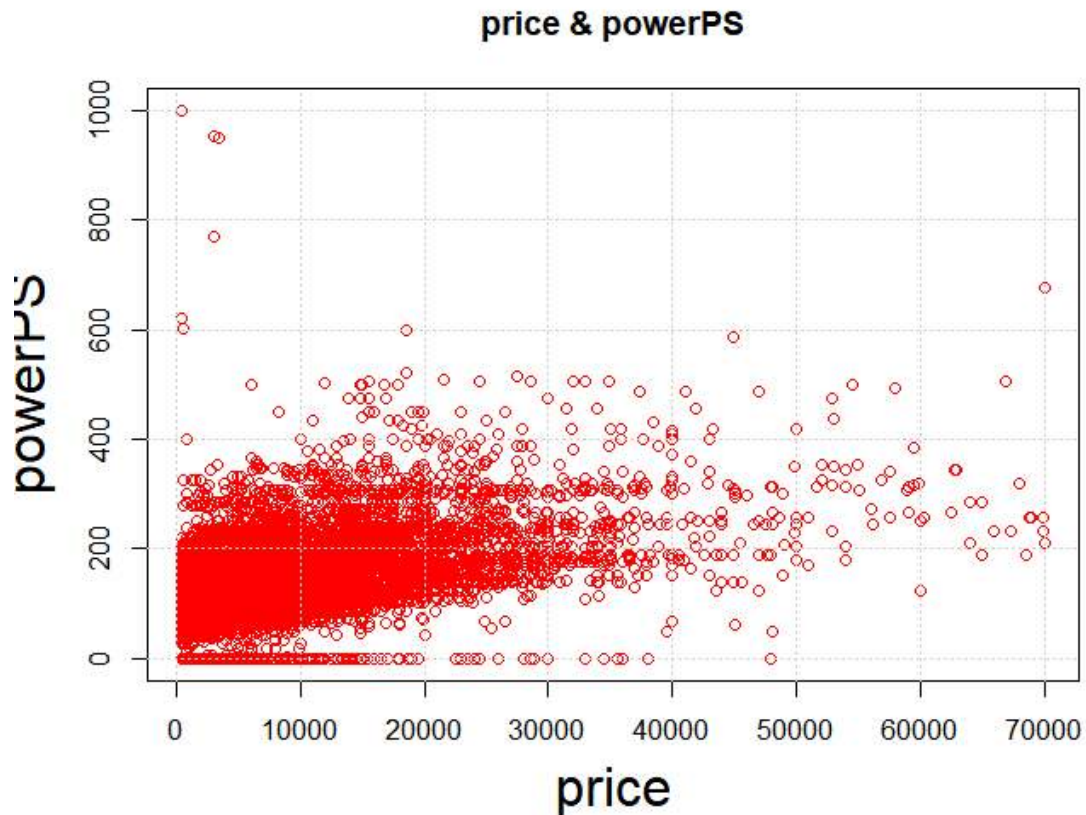
데이터 상관관계 분석을 통해 생긴 히트 맵을 이용하여 가격에 영향을 주는 변수들에 대해

알아보았다.

price에 영향을 미치는 변수	price에 영향을 미치지 않는 변수
yearOfRegistration(연식), powerPS(마력), kilometer(주행거리), gearbox(기어방식), brand, notRepairedDamage(수리경력), fuelType(연료)	abtest, vehicleType(차종), model(모델명)



추가로 price(가격)와 powerPS(마력) 데이터 간의 상관관계 분석을 위해 R의 plot 함수를 사용하여 산점도를 그려보았다.



3.3. 예측 결과에 영향이 없는 변수 제거

상관관계 분석 결과 영향이 적었던 변수 3개를 drop을 사용하여 제거한다. 그 결과 필드 개수가 [18623 56]에서 [18623 53]으로 줄어들었다.

```
cdata = cdata.drop('model', 1)##
        .drop('abtest', 1)##
        .drop('vehicleType', 1)
print(">> csv after drop: ", len(cdata), len(cdata.columns.values))
```

3.4. 학습데이터 / 테스트 데이터 생성

지금까지 처리한 결과를 새로운 csv 파일로 저장하였다. 저장할 때 18623의 60%인 11173개는 train용 csv 파일로 저장하고 나머지는 test용 csv 파일로 저장하였다.

```
# load data
xy = np.loadtxt(CSV_TRAIN_NAME, delimiter=',', dtype=np.float32)
x_data = xy[:, 1:]
y_data = xy[:, [0]]

NUM_OF_COL = np.size(x_data, axis=1)
```

3.5. 학습 과정

- 학습 데이터 로드

```
xy = np.loadtxt('autos_preprocessed.csv', delimiter=',', dtype=np.float32)
x_data = xy[:, 1:]
y_data = xy[:, [0]]

NUM_OF_COL = np.size(x_data, axis=1)
```

- tf.matmul을 사용하여 **선형 회귀 방식**으로 예측한다. 선형 회귀를 사용했기 때문에 optimizer도 다른 optimizer에 비해 비교적 빠르게 학습하는 **Gradient Descent**를 사용하였다.

```
# Hypothesis
hypothesis = tf.matmul(X, W) + b

# Simplified cost/loss function
cost = tf.reduce_mean(tf.square(hypothesis - Y))

# Minimize
optimizer = tf.train.GradientDescentOptimizer(learning_rate=0.01)
train = optimizer.minimize(cost)
```

4. 분석 결과의 정확도 분석

4.1. 정확도 분석

학습은 총 10만 epoch를 돌려 테스트 데이터를 맞춰 본 결과, 평균 95.70%의 유사도로 맞췄다.

- 정확도 분석에 사용한 연산

```
# test
accuracy = tf.reduce_mean(1 - abs(hypothesis - Y))
```

- 정확도 출력 결과

```
prediction: [[0.1827354 ]
             [0.00615078]
             [0.05233002]
             ...
             [0.1904685 ]
             [0.08809946]
             [0.08904424]]
score: 95.70289850234985 %
```

- 위 예측값 (prediction)의 실제 값은 price의 max-value인 **69999**, min-value인 **500**을 사용하여 구할 수 있다.

```
SCALED_PRICE = PREDICTION * (MAX - MIN) + MIN
```

4.2. 오차율 분석

오차율은 직접 예측값과 실제 label을 비교하여 나타냈다. 상단의 for문에서 max_error를 구하는 부분은 예측값과 label이 가장 많이 차이나는 행의 예측값(p)과 label(l)을 출력한다. 하단의 diff는 배열 연산 후, 오차의 최솟값과 최댓값을 출력한다.

```
# 오차율 계산
max_error = 0
p = 0
l = 0
for i in range(np.size(x_test, 0)):
    error = abs(prediction[i] - y_test[i])
    if (max_error < error):
        p = prediction[i]
        l = y_test[i]
        max_error = error

print(p, l, max_error)
diff = abs(prediction - y_test)
print(diff.min(axis=0)[0], '~', diff.max(axis=0)[0])
```

출력 결과

```
[1.1586559] [0.13309544] [1.0255604]
4.786998e-07 ~ 1.0255604
```

5. 분석 결과의 시사점 및 결론

분석 기법으로는 종류를 나누거나 분류하는 것이 아닌 값(중고차의 가격)을 예측하는 것으로서 간단한 Gradient Decent 알고리즘을 사용하여 빠르게 학습시킬 수 있는 **선형 회귀 모델(Linear Regression Model)**을 만들어 사용하였다. Neural Network를 사용한다면 느리지만 MLP, Backpropagation로 더 정확한 예측이 가능할 것으로 전망한다.

상관관계 분석 과정에서 연식, 마력, 주행거리, 기어방식, 브랜드가 중고차 가격에 꽤 많은 영향을 미치는 것을 확인할 수 있었다. 수리경력과 연료에서는 약간의 영향(0.3)을 미치는 반면, 차종과 모델명은 가격에 큰 영향을 미치지 않는다는 것을 알 수 있었다.

앞에서 우리는 마력, 지붕이 열리는 차(차종), 수리경력 유무, 주행거리, 연식이 가격에 영향을 줄 것으로 예상했다. 마력과 수리경력 유무, 주행거리, 연식 등에 있어서는 예상한 대로 가격에 영향을 준다는 것이 나타났지만, 단순히 지붕이 열리는 차(차종)은 가격에 영향을 크게 미치지 않는다는 것으로 확인하였다. 지붕이 열리는 차의 데이터 수가 전체 데이터에서 차지하는 비율이 많지 않다는 점과 지붕이 열리는 차라고 하더라도 마력 작고 수리경력이 있으며, 연식이 오래된 경우에는 이러한 요소들에게 더 큰 영향을 받기 때문에 단순히 지붕이 열리는 차는 가격에 큰 영향을 미치지 않는다고 판단하였다.

Tensorflow를 통한 예측모델을 학습시킨 후 예측의 **정확도 분석**에서 약 95%로 가격 예측에 성공하였다. 이를 통해 딜러의 도움 없이 자동차를 판매하는 경우 자신의 자동차 데이터를 입력하여 현재 중고차 시장에서 얼마 정도의 가격으로 측정이 될 것인지 간단하고 쉽게 확인 및 예측을 해볼 수 있다고 생각한다. 또한 중고차를 구입하는 경우에는 자동차의 기본 스펙(영향을 미치는 요소들)을 입력하여 예상 견적을 뽑아내는 것이 가능하다고 판단된다.

하지만 **오차율 분석 과정**에서 예측 값과 실제 값이 크게 차이나는 경우가 확인되어 데이터에서 주어진 변수들이 가격을 정확히 설명하기 어렵다고 할 수 있다. 따라서 여기서 사용한 데이터 이외에 중고차 가격에 영향을 줄 것이라 추측되는 변수로 중고차의 상태에 대한 상세 데이터, 판매자의 중고차 판매경험, 현재 중고차 시장의 활성화 등의 추가로 여러 데이터들을 고려해야한다고 판단된다. 이에 단순히 개인이 중고차를 일대일로 거래를 하는 것보다 딜러가 이러한 참고 데이터 모형을 가지고 좀 더 객관적이고 공정한 가격으로 소비자에게 중고차 매매를 진행한다면 만족스럽고 실속있는 거래를 할 수 있을 거라 생각된다.