SH의 학습노트 구독하기

SH의 학습노트

[Python]변수선택법 실습(2) - 전진선택법/후진소거법/단계적선택법/MAPE 모델 성능 평가 (변수선택법 실습(1)에 전처리과정 존재)

2020. 6. 19. 14:44 · ML in Python/Python

*아래 학습은 Fastcampus의 "머신러닝 A-Z까지"라는 인터넷 강의에서 실습한 내용을 복습하며 학습 과정을 공유하고자 복기한 내용입니다.

실습에 사용될 데이터 : Toyota Corolla Data (Toyota Corolla **모델 차 가격/기능 데이터**)

ToyotaCorolla.csv
0.21MB

회귀분석을 할 때 다중공선성이 발생하면, 데이터 분석의 신뢰성이나 예측 정확도를 떨어뜨린다. 이러한 문제를 하기 위한 방법 중 하나로 데이터 선정/전처리 과정에서 "**변수선택**"이

SH의 학습노트 구독하기

변수 선택법(Variable Selection)은

1. **전진선택법**(Forward Selection)

2. **후진소거법**(Backward Elimination)

3. **단계적선택법**(Stepwise Selection)

이 있다.

이 변수 선택법들을 알아가기 위해 Python을 통한 실습을 진행해보자. 이전 전치리과정과 모델 확인 과정은

이전게시물 : 변수선택법(1)에 존재한다. 학습이 목적이라면 보고 오는 것이 좋다.

link : https://todayisbetterthanyesterday.tistory.com/9

[Python]변수선택법 실습(1) - 변수선택법 실습 이전단계, 불필요한 …

todayisbetterthanyesterday.tistory.com

0. 변수선택법 (전체 경우의 수를 찾는 방법)

```
# 변수선택을 통해 형성한 모델의 AIC를 구하는 함수
# AIC가 낮을 수록 모델이 좋다고 평가된다.

def processSubset(X,y,feature_set):
    model = sm.OLS(y,X[list(feature_set)]) # Modeling
    regr = model.fit() # model fitting
    AIC = regr.aic # model's AIC
    return {"model" : regr, "AIC" : AIC}

print(processSubset(X = train_x, y = train_y, feature_set = feature_columns[0:5]))
```

```
{'model': <statsmodels.regression.linear_model.RegressionResultsWrapper object at 0x00000230B700EE08>, 'AIC': 19071.920536897833}
```

print(processSubset(X = train_x, y = train_y, feature_set = feature_columns[0:5]))

SH의 학습노트 구독하기

```
# 전체 변수의 AIC test

processSubset(X=train_x, y=train_y, feature_set = feature_columns)
```

```
{'model': <statsmodels.regression.linear_model.RegressionResultsWrapper at 0x230b700e148>,
 'AIC': 17001.91610144188}
```

전체 변수 모델의 AIC

```
import time
import itertools

# getBest : 가장 낮은 AIC를 가지는 모델을 선택하고 저장하는 함수

def getBest(X,y,k):
    tic = time.time()       # 시작 시간
    results = []            # 결과 저장 공간
    for combo in
itertools.combinations(X.columns.difference(['const'],k) :
                # 각 변수 조합을 고려한 경우의수

        combo = (list(combo)+['const'])
        # 상수항을 추가하여 combo를 결성

        results.append(processSubset(X,y,feature_set = combo)) # 모델링된
것을 저장

        # 만약 k=2이면 여기서 두가지 변수만 뽑아서 경우의 수를 분석하여
        # 저장 후 그 중 AIC가 가장 낮은 모델을 선택하도록 함

    models = pd.DataFrame(results) # 데이터프레임으로 모델결과 변환
    best_model = models.loc[models['AIC'].argmin()] # argmin은 최소값의
인덱스를 뽑는 함수
    toc = time.time()        # 종료 시간
    print("Processed", models.shape[0], "models on", k, "predictors
in",(toc - tic),"seconds.")

    return best_model

print(getBest(X=train_x, y = train_y, k=2))
```

```
Processed  630 models on 2 predictors in 1.4959793090820312 seconds.
model    <statsmodels.regression.linear_model.Regressio...
AIC                                              17484.3
Name: 211, dtype: object
```

print(getBest(X=train_x, y = train_y, k=2))

위의 함수는 **전체 변수의 가능한 조합을 모두 확인**하는 함수이다. 좋은 변수를 선택하여 모델을 만들 수 있겠지만, 문제는 **변수의 총 수와 k가 증가할때마다 시간이 기하급수적으로** [illegible] 다. 그렇기 때문에 "**변수를 선택하는 방법**"을 선정해야한다.

SH의 학습노트 구독하기

```
# 변수 선택에 따른 학습시간과 저장

models = pd.DataFrame(columns=["AIC","model"])
tic = time.time()
for i in range(1,4):
        models.loc[i] = getBest(X=train_x, y=train_y,k=i)
toc = time.time()
print("Total elapsed time:",(toc-tic),"seconds.")
```

```
Processed  36 models on 1 predictors in 0.06781911849975586 seconds.
Processed  630 models on 2 predictors in 1.148927927017212 seconds.
Processed  7140 models on 3 predictors in 12.886253356933594 seconds.
Total elapsed time: 14.36683964729309 seconds.
```

변수 조합 가능 경우의 수와 선별소요시간을 알려준다.

```
# 선택된 변수의 개수(1,2,3)별 가장낮은 AIC를 보유한 모델들이 들어있는 DF

models
```

	AIC	model
1	17744.411952	<statsmodels.regression.linear_model.Regressio...
2	17484.284528	<statsmodels.regression.linear_model.Regressio...
3	17347.522955	<statsmodels.regression.linear_model.Regressio...

models DataFrame

```
# 가장 AIC가 낮은 3번째 모델의 OLS결과를 출력

models.loc[3,"model"].summary()
```

SH의 학습노트 구독하기

OLS Regression Results

Dep. Variable:	Price	R-squared:	0.852
Model:	OLS	Adj. R-squared:	0.851
Method:	Least Squares	F-statistic:	1919.
Date:	Tue, 17 Mar 2020	Prob (F-statistic):	0.00
Time:	14:41:49	Log-Likelihood:	-8669.8
No. Observations:	1005	AIC:	1.735e+04
Df Residuals:	1001	BIC:	1.737e+04
Df Model:	3		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Automatic_airco	3728.2370	208.908	17.846	0.000	3318.289	4138.185
KM	-0.0158	0.001	-12.174	0.000	-0.018	-0.013
Mfg_Year	1586.9349	34.582	45.889	0.000	1519.074	1654.796
const	-3.162e+06	6.92e+04	-45.700	0.000	-3.3e+06	-3.03e+06

Omnibus:	150.836	Durbin-Watson:	2.032
Prob(Omnibus):	0.000	Jarque-Bera (JB):	1480.385
Skew:	0.326	Prob(JB):	0.00
Kurtosis:	8.910	Cond. No.	1.28e+08

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.28e+08. This might indicate that there are
strong multicollinearity or other numerical problems.

models.loc[3,"model"].summary()

```
# 모든 변수를 모델링한 것과 비교

print("full model Rsquared:","{:.5f}".format(fitted_full_model.rsquared))
print("full model AIC:","{:.5f}".format(fitted_full_model.aic))
print("full model MSE:","{:.5f}".format(fitted_full_model.mse_total))

print("selected model Rsquared:","
{:.5f}".format(models.loc[3,"model"].rsquared))
print("selected model AIC:","{:.5f}".format(models.loc[3,"model"].aic))
print("selected model MSE:","{:.5f}".format(models.loc[3,"model"].mse_total))
```

```
full model Rsquared:  0.90106
full model AIC:  17001.91610
full model MSE:  12310969.98808
selected model Rsquared:  0.85186
selected model AIC:  17347.52296
selected model MSE:  12310969.98808
```

full model vs selected model

1. 전진선택법

SH의 학습노트 구독하기

```
### 전진석택법(step=1)

def forward(X,y,predictors):

    # predictor - 현재 선택되어있는 변수
    # 데이터 변수들이 미리정의된 predictors에 있는지 없는지 확인 및 분류

    remaining_predictors = [p for p in X.columns.difference(['const']) if p
not in predictors]
    tic = time.time()
    results = []
    for p in remaining_predictors :
        results.append(processSubset(X=X,y=y,feature_set=predictors+[p]+
['const']))

    # 데이터프레임으로 변환
    models = pd.DataFrame(results)

    # AIC가 가장 낮은 것을 선택
    best_model = models.loc[models['AIC'].argmin()]
    toc = time.time()
    print("Processed ",models.shape[0]. "models on", len(predictors)+1,
"predictors in", (toc-tic))
    print("Selected predictors:",best_model["model"].model.exog_names,"AIC:
",best_model[0])
    return best_model

### 전진선택법 모델

def forward_model(X,y):

    Fmodels = pd.DataFrame(columns=["AIC","model"])
    tic = time.time()

    # 미리 정의된 데이터 변수
    predictors = []

    # 변수 1~10개 : 0-9 -> 1-10
    for i in range(1,len(X,columns.difference(['const']))+1):
        Forward_result = forward(X=X,y=y,predictors=predictors)
        if i > 1 :
            if Forward_result["AIC"] > Fmodel_before:
                break
        Fmodels.loc[i] = Forward_result
        predictors = Fmodels.loc[i]["model"].model.exog_names
        Fmodel_before = Fmodels.loc[i]["AIC"]
        predictors = [k for k in predictors if k != 'const']
    toc = time.time()
    print("Total elapsed time:",(toc-tic), "seconds.")

    return (Fmodels['model'][len(Fmodels['model'])])
```

```
Forward_best_model = forward_model(X=train_x, y=train_y)
```

SH의 학습노트 구독하기

```
Processed  36 models on 1 predictors in 0.07032036781311035
Selected predictors: ['Mfg_Year', 'const']  AIC: <statsmodels.regression.linear_model.RegressionResultsWrapper object at 0x0000026A
AA69FC08>
Processed  35 models on 2 predictors in 0.06881427764892578
Selected predictors: ['Mfg_Year', 'Automatic_airco', 'const']  AIC: <statsmodels.regression.linear_model.RegressionResultsWrapper o
bject at 0x0000026AAA6A9448>
Processed  34 models on 3 predictors in 0.074798583984375
Selected predictors: ['Mfg_Year', 'Automatic_airco', 'KM', 'const']  AIC: <statsmodels.regression.linear_model.RegressionResultsWra
pper object at 0x0000026AAA695F48>
Processed  33 models on 4 predictors in 0.07679438591003418
Selected predictors: ['Mfg_Year', 'Automatic_airco', 'KM', 'Weight', 'const']  AIC: <statsmodels.regression.linear_model.Regression
ResultsWrapper object at 0x0000026AA9679DC8>
Processed  32 models on 5 predictors in 0.07779335975646973
Selected predictors: ['Mfg_Year', 'Automatic_airco', 'KM', 'Weight', 'HP', 'const']  AIC: <statsmodels.regression.linear_model.Regr
essionResultsWrapper object at 0x0000026AAA665C48>
Processed  31 models on 6 predictors in 0.06683850288391113
Selected predictors: ['Mfg_Year', 'Automatic_airco', 'KM', 'Weight', 'HP', 'Powered_Windows', 'const']  AIC: <statsmodels.regressio
n.linear_model.RegressionResultsWrapper object at 0x0000026AA9687D88>
Processed  30 models on 7 predictors in 0.0718083381652832
Selected predictors: ['Mfg_Year', 'Automatic_airco', 'KM', 'Weight', 'HP', 'Powered_Windows', 'BOVAG_Guarantee', 'const']  AIC: <st
atsmodels.regression.linear_model.RegressionResultsWrapper object at 0x0000026AAA6B4A48>
Processed  29 models on 8 predictors in 0.06183338165283203
Selected predictors: ['Mfg_Year', 'Automatic_airco', 'KM', 'Weight', 'HP', 'Powered_Windows', 'BOVAG_Guarantee', 'Guarantee_Perio
d', 'const']  AIC: <statsmodels.regression.linear_model.RegressionResultsWrapper object at 0x0000026AAA690208>
Processed  28 models on 9 predictors in 0.05884099006652832
Selected predictors: ['Mfg_Year', 'Automatic_airco', 'KM', 'Weight', 'HP', 'Powered_Windows', 'BOVAG_Guarantee', 'Guarantee_Perio
d', 'Sport_Model', 'const']  AIC: <statsmodels.regression.linear_model.RegressionResultsWrapper object at 0x0000026AAA674B08>
Processed  27 models on 10 predictors in 0.05086493492126465
Selected predictors: ['Mfg_Year', 'Automatic_airco', 'KM', 'Weight', 'HP', 'Powered_Windows', 'BOVAG_Guarantee', 'Guarantee_Perio
d', 'Sport_Model', 'Quarterly_Tax', 'const']  AIC: <statsmodels.regression.linear_model.RegressionResultsWrapper object at 0x000002
6AA9683CC8>
```

변수를 계속 추가하며 AIC가 증가하는 경우가 생기면, 이전 모델을 선택하는 학습과정을 진행한다.

```
Forward_best_model.aic
```

```
16931.423078614705
```

전진선택법 AIC

```
Forward_best_model.summary()
```

SH의 학습노트 구독하기

OLS Regression Results

Dep. Variable:	Price	R-squared:	0.913
Model:	OLS	Adj. R-squared:	0.911
Method:	Least Squares	F-statistic:	430.2
Date:	Fri, 19 Jun 2020	Prob (F-statistic):	0.00
Time:	13:40:36	Log-Likelihood:	-8440.7
No. Observations:	1005	AIC:	1.693e+04
Df Residuals:	980	BIC:	1.705e+04
Df Model:	24		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Mfg_Year	1085.3547	128.927	8.418	0.000	832.350	1338.359
Automatic_airco	2153.8579	181.656	11.857	0.000	1797.379	2510.337
KM	-0.0176	0.001	-13.476	0.000	-0.020	-0.015
Weight	15.2183	1.344	11.324	0.000	12.581	17.856
HP	17.7670	3.569	4.978	0.000	10.763	24.771
Powered_Windows	447.7987	144.318	3.103	0.002	164.591	731.006
BOVAG_Guarantee	500.2034	132.596	3.772	0.000	239.999	760.408
Guarantee_Period	70.6432	15.236	4.637	0.000	40.745	100.542
Sport_Model	342.1842	87.579	3.907	0.000	170.320	514.049
Quarterly_Tax	13.1931	1.829	7.213	0.000	9.604	16.782
Petrol	2364.8527	374.446	6.316	0.000	1630.044	3099.661
Tow_Bar	-245.6601	79.388	-3.094	0.002	-401.450	-89.871
Backseat_Divider	-371.1265	123.808	-2.998	0.003	-614.086	-128.167
Mfr_Guarantee	213.7289	75.902	2.816	0.005	64.781	362.677
Metallic_Rim	257.6785	94.246	2.734	0.006	72.731	442.626
Airco	247.9995	89.804	2.762	0.006	71.770	424.229
ABS	-305.2334	104.103	-2.932	0.003	-509.524	-100.943
Diesel	996.9856	360.576	2.765	0.006	289.396	1704.576
Age_08_04	-22.3608	10.775	-2.075	0.038	-43.505	-1.217
Automatic	308.1617	159.519	1.932	0.054	-4.876	621.199
CD_Player	227.6148	100.891	2.256	0.024	29.627	425.602
Boardcomputer	-220.5754	119.962	-1.839	0.066	-455.987	14.837
Central_Lock	-228.2779	142.448	-1.603	0.109	-507.816	51.261
Airbag_1	324.3694	222.848	1.456	0.146	-112.945	761.684
const	-2.18e+06	2.58e+05	-8.440	0.000	-2.69e+06	-1.67e+06

Omnibus:	72.129	Durbin-Watson:	2.021
Prob(Omnibus):	0.000	Jarque-Bera (JB):	280.121
Skew:	0.211	Prob(JB):	1.49e-61

전진선택법 모델 OLS 결과

2. 후진소거법

```
### 후진소거법(step=1)
def backward(X,y,predictors):
```

```
    tic = time.time()
    results = []

    # 데이터 변수들이 미리 정의된 predictors 조합 확인

    for combo in itertools.combinations(predictors, len(predictors) - 1):
        results.append(processSubset(X=X,y=y,feature_set=list(combo)+
['const']))
    models = pd.DataFrame(results)

    # 가장 낮은 AIC를 가진 모델을 선택
    best_model = models.loc[models['AIC'].argmin()]
    toc = time.time()

    print("Processed ",models.shape[0], "models on", len(predictors) - 1,
"predictors in",(toc-tic))
    print("Selected predictors:",best_model['model'].model.exog_names,'
AIC:',best_model[0])
    return best_model

def backward_model(X,y) :
    Bmodels = pd.DataFrame(columns=["AIC","model"], index =
range(1,len(X.columns))
    tic = time.time()
    predictors = X.columns.difference(['const'])
    Bmodel_before = processSubset(X,y,predictors)['AIC']
    while (len(predictors) > 1):
        Backward_result = backward(X=train_x, y= train_y,
predictors=predictors)
        if Backward_result['AIC'] > Bmodel_before :
                break
        Bmodels.loc[len(predictors) -1] = Backward_result
        predictors = Bmodel.loc[len(predictors) - 1]
['model'].model.exog_names
        Bmodel_before = Backward_result["AIC"]
        predictors = [k for k in predictors if k != 'const']

    toc = time.time()
    print("Total elapsed time:",(toc-tic),"seconds.")
    return (Bmodels["model"].dropna().iloc[0]
```

SH의 학습노트 구독하기

```
Backward_best_model = backward_model(X=train_x, y= train_y)
```

```
Processed  36 models on 35 predictors in 0.22747516632080078
Selected predictors: ['ABS', 'Age_08_04', 'Airbag_1', 'Airbag_2', 'Airco', 'Automatic', 'Automatic_airco', 'BOVAG_Guarantee', 'Back
seat_Divider', 'Boardcomputer', 'CD_Player', 'CNG', 'Central_Lock', 'Cylinders', 'Diesel', 'Doors', '
P', 'KM', 'Met_Color', 'Metallic_Rim', 'Mfg_Month', 'Mfg_Year', 'Mfr_Guarantee', 'Petrol', 'Power_Ste
arterly_Tax', 'Radio', 'Radio_cassette', 'Sport_Model', 'Tow_Bar', 'Weight', 'cc', 'const']  AIC: <statsmodels.regression.linear_mo
del.RegressionResultsWrapper object at 0x00000230CF44CD48>
Processed  35 models on 34 predictors in 0.17675542831420898
Selected predictors: ['ABS', 'Age_08_04', 'Airbag_1', 'Airbag_2', 'Airco', 'Automatic', 'Automatic_airco', 'BOVAG_Guarantee', 'Back
seat_Divider', 'Boardcomputer', 'CD_Player', 'CNG', 'Central_Lock', 'Cylinders', 'Diesel', 'Doors', 'Gears', 'Guarantee_Period', 'H
P', 'KM', 'Met_Color', 'Metallic_Rim', 'Mfg_Month', 'Mfg_Year', 'Mfr_Guarantee', 'Petrol', 'Powered_Windows', 'Quarterly_Tax', 'Rad
io', 'Radio_cassette', 'Sport_Model', 'Tow_Bar', 'Weight', 'cc', 'const']  AIC: <statsmodels.regression.linear_model.RegressionResu
ltsWrapper object at 0x00000230CF437908>
Processed  34 models on 33 predictors in 0.18745827674865723
Selected predictors: ['ABS', 'Age_08_04', 'Airbag_2', 'Airco', 'Automatic', 'Automatic_airco', 'BOVAG_Guarantee', 'Backseat_Divide
r', 'Boardcomputer', 'CD_Player', 'CNG', 'Central_Lock', 'Cylinders', 'Diesel', 'Doors', 'Gears', 'Guarantee_Period', 'HP', 'KM',
'Met_Color', 'Metallic_Rim', 'Mfg_Month', 'Mfg_Year', 'Mfr_Guarantee', 'Petrol', 'Powered_Windows', 'Quarterly_Tax', 'Radio', 'Radi
o_cassette', 'Sport_Model', 'Tow_Bar', 'Weight', 'cc', 'const']  AIC: <statsmodels.regression.linear_model.RegressionResultsWrapper
object at 0x00000230CF429A48>
Processed  33 models on 32 predictors in 0.17581558227539062
Selected predictors: ['ABS', 'Age_08_04', 'Airco', 'Automatic', 'Automatic_airco', 'BOVAG_Guarantee', 'Backseat_Divider', 'Boardcom
puter', 'CD_Player', 'CNG', 'Central_Lock', 'Cylinders', 'Diesel', 'Doors', 'Gears', 'Guarantee_Period', 'HP', 'KM', 'Met_Color',
'Metallic_Rim', 'Mfg_Month', 'Mfg_Year', 'Mfr_Guarantee', 'Petrol', 'Powered_Windows', 'Quarterly_Tax', 'Radio', 'Radio_cassette',
'Sport_Model', 'Tow_Bar', 'Weight', 'cc', 'const']  AIC: <statsmodels.regression.linear_model.RegressionResultsWrapper object at 0x
00000230CF4291C8>
Processed  32 models on 31 predictors in 0.22132134437561035
Selected predictors: ['ABS', 'Age_08_04', 'Airco', 'Automatic', 'Automatic_airco', 'BOVAG_Guarantee', 'Backseat_Divider', 'Boardcom
puter', 'CD_Player', 'CNG', 'Central_Lock', 'Cylinders', 'Diesel', 'Doors', 'Guarantee_Period', 'HP', 'KM', 'Met_Color', 'Metallic_
Rim', 'Mfg_Month', 'Mfg_Year', 'Mfr_Guarantee', 'Petrol', 'Powered_Windows', 'Quarterly_Tax', 'Radio', 'Radio_cassette', 'Sport_Mod
el', 'Tow_Bar', 'Weight', 'cc', 'const']  AIC: <statsmodels.regression.linear_model.RegressionResultsWrapper object at 0x00000230CF
43B548>
Processed  31 models on 30 predictors in 0.15282320976257324
Selected predictors: ['ABS', 'Age_08_04', 'Airco', 'Automatic', 'Automatic_airco', 'BOVAG_Guarantee', 'Backseat_Divider', 'Boardcom
puter', 'CD_Player', 'CNG', 'Central_Lock', 'Cylinders', 'Diesel', 'Doors', 'Guarantee_Period', 'HP', 'KM', 'Metallic_Rim', 'Mfg_Mo
nth', 'Mfg_Year', 'Mfr_Guarantee', 'Petrol', 'Powered_Windows', 'Quarterly_Tax', 'Radio', 'Radio_cassette', 'Sport_Model', 'Tow_Ba
r', 'Weight', 'cc', 'const']  AIC: <statsmodels.regression.linear_model.RegressionResultsWrapper object at 0x00000230CF443488>
```

SH의 학습노트 구독하기

전체 변수를 다 넣은 full모델부터 개수가 하나씩 줄며 AIC가 높아지면 그 변수는 제외하는 방식이다.

```
Backward_best_model.aic
```

16986.47214565498

후진소거법 AIC

```
Backward_best_model.summary()
```

SH의 학습노트 구독하기

OLS Regression Results

Dep. Variable:	Price	R-squared:	0.900
Model:	OLS	Adj. R-squared:	0.898
Method:	Least Squares	F-statistic:	422.3
Date:	Tue, 17 Mar 2020	Prob (F-statistic):	0.00
Time:	17:22:41	Log-Likelihood:	-8471.2
No. Observations:	1005	AIC:	1.699e+04
Df Residuals:	983	BIC:	1.709e+04
Df Model:	21		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
ABS	-268.6265	104.984	-2.559	0.011	-474.644	-62.609
Age_08_04	-22.5415	10.946	-2.059	0.040	-44.023	-1.060
Airco	169.5263	91.126	1.860	0.063	-9.298	348.350
Automatic	376.4619	145.954	2.579	0.010	90.045	662.879
Automatic_airco	2661.2599	190.430	13.975	0.000	2287.563	3034.957
BOVAG_Guarantee	443.9018	133.050	3.336	0.001	182.806	704.997
Backseat_Divider	-364.0490	119.959	-3.035	0.002	-599.455	-128.643
CD_Player	183.5882	104.925	1.750	0.080	-22.315	389.491
CNG	-2362.4514	421.520	-5.605	0.000	-3189.633	-1535.270
Cylinders	-5.162e+05	6.16e+04	-8.377	0.000	-6.37e+05	-3.95e+05
Diesel	-1475.9059	298.201	-4.949	0.000	-2061.089	-890.723
Guarantee_Period	65.2507	13.792	4.731	0.000	38.185	92.316
HP	13.3830	3.715	3.602	0.000	6.093	20.673
KM	-0.0168	0.001	-12.640	0.000	-0.019	-0.014
Mfg_Year	1097.3546	130.736	8.394	0.000	840.801	1353.909
Mfr_Guarantee	207.6703	78.851	2.634	0.009	52.934	362.406
Powered_Windows	418.7317	85.732	4.884	0.000	250.493	586.971
Quarterly_Tax	16.6899	1.895	8.808	0.000	12.972	20.408
Radio_cassette	-175.8576	107.080	-1.642	0.101	-385.989	34.273
Sport_Model	344.5004	88.377	3.898	0.000	171.070	517.931
Tow_Bar	-148.0873	82.311	-1.799	0.072	-309.614	13.439
Weight	8.7320	1.176	7.425	0.000	6.424	11.040
const	-1.291e+05	1.54e+04	-8.377	0.000	-1.59e+05	-9.88e+04

Omnibus:	111.195	Durbin-Watson:	1.974
Prob(Omnibus):	0.000	Jarque-Bera (JB):	767.547
Skew:	0.211	Prob(JB):	2.13e-167
Kurtosis:	7.261	Cond. No.	2.60e+20

후진소거법으로 선택된 모델의 OLS 결과

3. 단계적선택법

```
def Stepwise_model(X,y):
    Stepmodels = pd.DataFrame(columns = ["AIC","model"])
    tic = time.time()
    predictors = []
```

```
    Smodel_before = processSubset(X,y,predictors + ['const'])['AIC']

    # 변수 1~10개 0-9 -> 1-10
    for i in range(1,len(X.columns.difference(['const']))+1) :
        Forward_result = forward(X=X,y=y,predictors = predictors) # constant
added
        print('forward')
        predictors = Stepmodels.loc[i]['model'].model.exog_names
        predictors = [k for k in predictors if k != 'const']
        Backward_result = backward(X=X,y=y,predictors = predictors)
        if Backward_result["AIC"] < Forward_result["AIC"]
            Stepmodels.loc[i] = Backward_result
            predictors = Stepmodels.loc[i]["model"].model.exog_names
            Smodel_before = Stepmodels.loc[i]["AIC"]
            predictors = [k for k in predictors k != "const"]
            print('backward')
        if Stepmodels.loc[i]["AIC"] > Smodel_before:
            break
        else :
            Smodel_before = Stepmodels.loc[i]["AIC"]
    toc = time.time()
    print("Total elapsed time:",(toc-tic),"seconds.")
    return (Stepmodels["model"][len(Stepmodels["model"])])
```

SH의 학습노트 구독하기

```
Processed  21 models on 20 predictors in 0.07889747619628906
Selected predictors: ['Mfg_Year', 'Automatic_airco', 'KM', 'Weight', 'Powered_Windows', 'HP', 'Quarterly_Tax', 'Petrol', 'Guarantee
_Period', 'BOVAG_Guarantee', 'Sport_Model', 'Backseat_Divider', 'Mfr_Guarantee', 'CD_Player', 'Automatic', 'CNG', 'Tow_Bar', 'ABS',
'Mfg_Month', 'Airco', 'const']  AIC: <statsmodels.regression.linear_model.RegressionResultsWrapper object at 0x00000230CF476E08>
Processed  15 models on 22 predictors in 0.05292391777038574
Selected predictors: ['Mfg_Year', 'Automatic_airco', 'KM', 'Weight', 'Powered_Windows', 'HP', 'Quarterly_Tax', 'Petrol', 'Guarantee
_Period', 'BOVAG_Guarantee', 'Sport_Model', 'Backseat_Divider', 'Mfr_Guarantee', 'CD_Player', 'Automatic', 'CNG', 'Tow_Bar', 'ABS',
'Mfg_Month', 'Airco', 'Radio_cassette', 'Cylinders', 'const']  AIC: <statsmodels.regression.linear_model.RegressionResultsWrapper o
bject at 0x00000230B6F61408>
forward
Processed  22 models on 21 predictors in 0.08675932884216309
Selected predictors: ['Mfg_Year', 'Automatic_airco', 'KM', 'Weight', 'Powered_Windows', 'HP', 'Quarterly_Tax', 'Petrol', 'Guarantee
_Period', 'BOVAG_Guarantee', 'Sport_Model', 'Backseat_Divider', 'Mfr_Guarantee', 'CD_Player', 'Automatic', 'CNG', 'Tow_Bar', 'ABS',
'Mfg_Month', 'Airco', 'Radio_cassette', 'const']  AIC: <statsmodels.regression.linear_model.RegressionResultsWrapper object at 0x00
000230CF443AC8>
Processed  14 models on 23 predictors in 0.062087059020996094
Selected predictors: ['Mfg_Year', 'Automatic_airco', 'KM', 'Weight', 'Powered_Windows', 'HP', 'Quarterly_Tax', 'Petrol', 'Guarantee
_Period', 'BOVAG_Guarantee', 'Sport_Model', 'Backseat_Divider', 'Mfr_Guarantee', 'CD_Player', 'Automatic', 'CNG', 'Tow_Bar', 'ABS',
'Mfg_Month', 'Airco', 'Radio_cassette', 'Cylinders', 'Age_08_04', 'const']  AIC: <statsmodels.regression.linear_model.RegressionRes
ultsWrapper object at 0x00000230B6F8BB08>
forward
Processed  23 models on 22 predictors in 0.10551786422729492
Selected predictors: ['Mfg_Year', 'Automatic_airco', 'KM', 'Weight', 'Powered_Windows', 'HP', 'Quarterly_Tax', 'Petrol', 'Guarantee
_Period', 'BOVAG_Guarantee', 'Sport_Model', 'Backseat_Divider', 'Mfr_Guarantee', 'CD_Player', 'Automatic', 'CNG', 'Tow_Bar', 'ABS',
'Mfg_Month', 'Airco', 'Radio_cassette', 'Cylinders', 'const']  AIC: <statsmodels.regression.linear_model.RegressionResultsWrapper o
bject at 0x00000230B3EC3E08>
backward
Processed  14 models on 23 predictors in 0.0695044994354248
Selected predictors: ['Mfg_Year', 'Automatic_airco', 'KM', 'Weight', 'Powered_Windows', 'HP', 'Quarterly_Tax', 'Petrol', 'Guarantee
_Period', 'BOVAG_Guarantee', 'Sport_Model', 'Backseat_Divider', 'Mfr_Guarantee', 'CD_Player', 'Automatic', 'CNG', 'Tow_Bar', 'ABS',
'Mfg_Month', 'Airco', 'Radio_cassette', 'Cylinders', 'Age_08_04', 'const']  AIC: <statsmodels.regression.linear_model.RegressionRes
ultsWrapper object at 0x00000230CF443A48>
forward
Processed  23 models on 22 predictors in 0.09905004501342773
Selected predictors: ['Mfg_Year', 'Automatic_airco', 'KM', 'Weight', 'Powered_Windows', 'HP', 'Quarterly_Tax', 'Petrol', 'Guarantee
_Period', 'BOVAG_Guarantee', 'Sport_Model', 'Backseat_Divider', 'Mfr_Guarantee', 'CD_Player', 'Automatic', 'CNG', 'Tow_Bar', 'ABS',
'Mfg_Month', 'Airco', 'Radio_cassette', 'Cylinders', 'const']  AIC: <statsmodels.regression.linear_model.RegressionResultsWrapper o
bject at 0x00000230D04F4588>
backward
Processed  14 models on 23 predictors in 0.0680246353149414
Selected predictors: ['Mfg_Year', 'Automatic_airco', 'KM', 'Weight', 'Powered_Windows', 'HP', 'Quarterly_Tax', 'Petrol', 'Guarantee
_Period', 'BOVAG_Guarantee', 'Sport_Model', 'Backseat_Divider', 'Mfr_Guarantee', 'CD_Player', 'Automatic', 'CNG', 'Tow_Bar', 'ABS',
'Mfg_Month', 'Airco', 'Radio_cassette', 'Cylinders', 'Age_08_04', 'const']  AIC: <statsmodels.regression.linear_model.RegressionRes
ultsWrapper object at 0x00000230B3EC3888>
forward
Processed  23 models on 22 predictors in 0.08278298377990723
Selected predictors: ['Mfg_Year', 'Automatic_airco', 'KM', 'Weight', 'Powered_Windows', 'HP', 'Quarterly_Tax', 'Petrol', 'Guarantee
_Period', 'BOVAG_Guarantee', 'Sport_Model', 'Backseat_Divider', 'Mfr_Guarantee', 'CD_Player', 'Automatic', 'CNG', 'Tow_Bar', 'ABS',
'Mfg_Month', 'Airco', 'Radio_cassette', 'Cylinders', 'const']  AIC: <statsmodels.regression.linear_model.RegressionResultsWrapper o
bject at 0x00000230CF4046C8>
backward
```

forward와 backward를 AIC를 기준으로 비교하며 단계적 반복진행하는 학습을 통해 변수를 선택한다. (Stepwise)

SH의 학습노트 구독하기

```
Stepwise_best_model.aic
```

16986.472145654916

Stepwise 모델 AIC

4. 성능평가

```
# number of params
print(Forward_best_model.params.shape, Backward_best_model.params.shape,
Stepwise_best_model.params.shape)
```

(23,) (23,) (23,)

변수선택법에 따른 선택된 변수개수 (같다)

```
# 모델에 의해 예측된/추정된 값 = test_y
pred_y_full = fitted_full_model.predict(test_x)
pred_y_forward =
Forward_best_model.predict(test_x[Forward_best_model.model.exog_names])
pred_y_backward =
Backward_best_model.predict(test_x[Backward_best_model.model.exog_names])
pred_y_stepwise =
Stepwise_best_model.predict(test_x[Stepwise_best_model.model.exog_names])
```

```
# MSE, RMSE, MAE, MAPE 4가지 지표를 통해 예측성능을 확인할 예정

perf_mat = pd.DataFrame(columns=["ALL", "FORWARD", "BACKWARD",
"STEPWISE"],index =['MSE', 'RMSE','MAE', 'MAPE'])

# MAPE의 함수
def mean_absolute_percentage_error(y_true, y_pred):
    y_true, y_pred = np.array(y_true), np.array(y_pred)
    return np.mean(np.abs((y_true - y_pred) / y_true)) * 100
from sklearn import metrics  # 나머지는 sklearn에서 활용

# 성능지표
perf_mat.loc['MSE']['ALL'] = metrics.mean_squared_error(test_y,pred_y_full)
perf_mat.loc['MSE']['FORWARD'] =
metrics.mean_squared_error(test_y,pred_y_forward)
```

```
perf_mat.loc['MSE']['BACKWARD'] =
metrics.mean_squared_error(test_y,pred_y_backward)
perf_mat.loc['MSE']['STEPWISE'] =
metrics.mean_squared_error(test_y,pred_y_stepwise)

perf_mat.loc['RMSE']['ALL'] = np.sqrt(metrics.mean_squared_error(test_y,
pred_y_full))
perf_mat.loc['RMSE']['FORWARD'] = np.sqrt(metrics.mean_squared_error(test_y,
pred_y_forward))
perf_mat.loc['RMSE']['BACKWARD'] = np.sqrt(metrics.mean_squared_error(test_y,
pred_y_backward))
perf_mat.loc['RMSE']['STEPWISE'] = np.sqrt(metrics.mean_squared_error(test_y,
pred_y_stepwise))

perf_mat.loc['MAE']['ALL'] = metrics.mean_absolute_error(test_y, pred_y_full)
perf_mat.loc['MAE']['FORWARD'] = metrics.mean_absolute_error(test_y,
pred_y_forward)
perf_mat.loc['MAE']['BACKWARD'] = metrics.mean_absolute_error(test_y,
pred_y_backward)
perf_mat.loc['MAE']['STEPWISE'] = metrics.mean_absolute_error(test_y,
pred_y_stepwise)

perf_mat.loc['MAPE']['ALL'] = mean_absolute_percentage_error(test_y,
pred_y_full)
perf_mat.loc['MAPE']['FORWARD'] = mean_absolute_percentage_error(test_y,
pred_y_forward)
perf_mat.loc['MAPE']['BACKWARD'] = mean_absolute_percentage_error(test_y,
pred_y_backward)
perf_mat.loc['MAPE']['STEPWISE'] = mean_absolute_percentage_error(test_y,
pred_y_stepwise)

print(perf_mat)
```

SH의 학습노트 구독하기

	ALL	FORWARD	BACKWARD	STEPWISE
MSE	1.25986e+06	1.2825e+06	1.2825e+06	1.2825e+06
RMSE	1122.44	1132.47	1132.47	1132.47
MAE	808.762	812.329	812.329	812.329
MAPE	7.32996	7.32908	7.32908	7.32908

Full, Forward, Backward, Stepwise 네가지 예측오차 성능

위의 표를 보면 4가지 모두 모든 변수를 넣었을때 오차와 비슷하다는 것을 확인할 수 있다. 하지만, 모든 변수를 넣은 모델은 변수가 37개나 되기에, 학습의 효율성 측면에서 Full 변수 모델보다 효율적이다. 그리고 다중공선성 과적합과 같은 문제가 발생할 때, 변수를 줄이는 방법을 통해서 모델의 신뢰성을 높일 수 있을 것이다.

1 구독하기

TAG backward elimination, Forward selection, MAPE, MSE, rmse 택법, 변수선택법, 전진선택법, 후진소거법

SH의 학습노트 구독하기

관련글

[Python]회귀계수 축소법 실습 - Ridge,Lasso
2020.06.24

[Python]로지스틱회귀분석 실습
2020.06.20

[Python]변수선택법 실습(1) - 변수선택법 실...
2020.06.16

[Python]다중회귀분석 실습 - 모델해석과 다...
2020.06.13

댓글

이름

비밀번호

댓글을 입력해주세요.

비공개 댓글 남기기

티스토리

© 2018 TISTORY. All rights reserved.