과제 보고서

제목 : 파이썬(3차과제-모바일뱅킹)

과 목 명: 파이썬 통계분석
제출일자: 2022.11.17.
학 과: 정보통계학과
학 번: 2018015027
이 름: 김한탁

충북대학교
CHUNGBUK NATIONAL UNIVERSITY

*** 모바일뱅킹 자료**

자료) 모바일뱅킹

```
In [2]: data.head()
```

Out[2]:

	x1	x2	x3	x4	x5	x6	x7	x8	x9	x10	...	x16	x17	x18	x19	x20	x21	x22	x23	x24	y
0	4	6	6	6	5	7	7	7	4	4	...	5	7	4	4	6	4	7	7	7	2
1	5	6	5	4	6	7	2	7	6	6	...	6	6	4	6	7	6	5	4	5	4
2	5	7	6	3	2	7	3	4	5	5	...	7	7	5	5	5	3	5	4	4	3
3	4	3	3	3	4	5	5	6	6	5	...	7	7	7	6	6	7	3	5	5	2
4	7	7	7	2	1	2	1	3	3	3	...	6	5	3	2	2	2	6	6	6	1

1. 모바일뱅킹 자료를 train-set, test-set으로 8:2의 비율로 나누어 분석을 진행하라.

답안)

1) 코드 및 결과

```
In [3]: #[1] data set : train-set, test-set 분류
        from sklearn.model_selection import train_test_split
        from sklearn.linear_model import LinearRegression
        from sklearn.metrics import mean_squared_error , r2_score

        ## 훈련용, 테스트용 자료 분류 : random
        y_target=data['y']
        x_data=data.drop(['y'],axis=1,inplace=False)
```

```
In [4]: x_train, x_test, y_train, y_test= train_test_split(x_data, y_target, test_size=0.2, random_state=156)
        print('훈련 데이터의 크기 :',x_train)
        print('테스트 데이터의 크기 :',x_test)
```

```
훈련 데이터의 크기 :     x1  x2  x3  x4  x5  x6  x7  x8  x9  x10  ...  x15  x16  x17  x18  x19  #
16   7   5   5   5   6   7   4   5   6   5  ...    7    6    6    6    5
31   6   7   7   4   7   7   7   7   7   7  ...    7    7    7    7    7
18   6   6   6   5   1   7   4   4   5   5  ...    7    1    1    1    2
30   7   6   7   4   3   7   7   3   7   5  ...    7    7    7    4    4
54   5   5   7   4   3   5   4   7   7   4  ...    7    7    6    3    3
20   3   3   3   3   7   7   6   7   6   4  ...    5    5    5    5    5
15   5   6   3   5   2   6   5   6   6   4  ...    6    5    5    6    5
32   6   6   6   4   2   7   3   5   2   3  ...    5    5    5    3    4
50   6   6   6   6   6   6   7   7   7   7  ...    7    6    6    7    7
33   7   7   5   2   3   7   3   4   7   3  ...    7    7    7    3    3
55   6   7   6   2   5   7   6   7   6   6  ...    7    6    6    4    5
21   7   7   7   5   5   7   7   5   7   7  ...    7    7    5    5    5
59   5   6   5   4   6   4   2   7   6   6  ...    7    6    6    4    6
8    6   5   6   5   2   5   3   3   4   5  ...    7    3    3    1    2
38   4   7   7   2   4   7   7   7   7   7  ...    7    7    6    4    4
45   5   7   6   4   4   7   6   6   6   6  ...    7    5    5    5    6
47   7   7   7   7   1   7   7   4   7   7  ...    7    4    4    5    4
1    5   6   5   4   6   7   2   7   6   6  ...    7    6    6    4    6
```

2) 결과해석

문제에 따라 모바일뱅킹 자료를 랜덤으로 추출하여, train-set과 test-set에 할당하였다.

랜덤으로 추출한 80%의 변수와 종속변수 자료 데이터를 각각 x_train과 y_train에 할당하였고, 나머지는 x_test와 y_test에 할당하였다.

2. train-set을 이용해 24개 독립변수와 종속변수(y)의 다중회귀모형 분석을 수행하라. (분석 절차대로 상세히)

답안)

1) train-set의 변수들에 대한 행렬산점도와 상관계수

1-1) 코드 및 결과

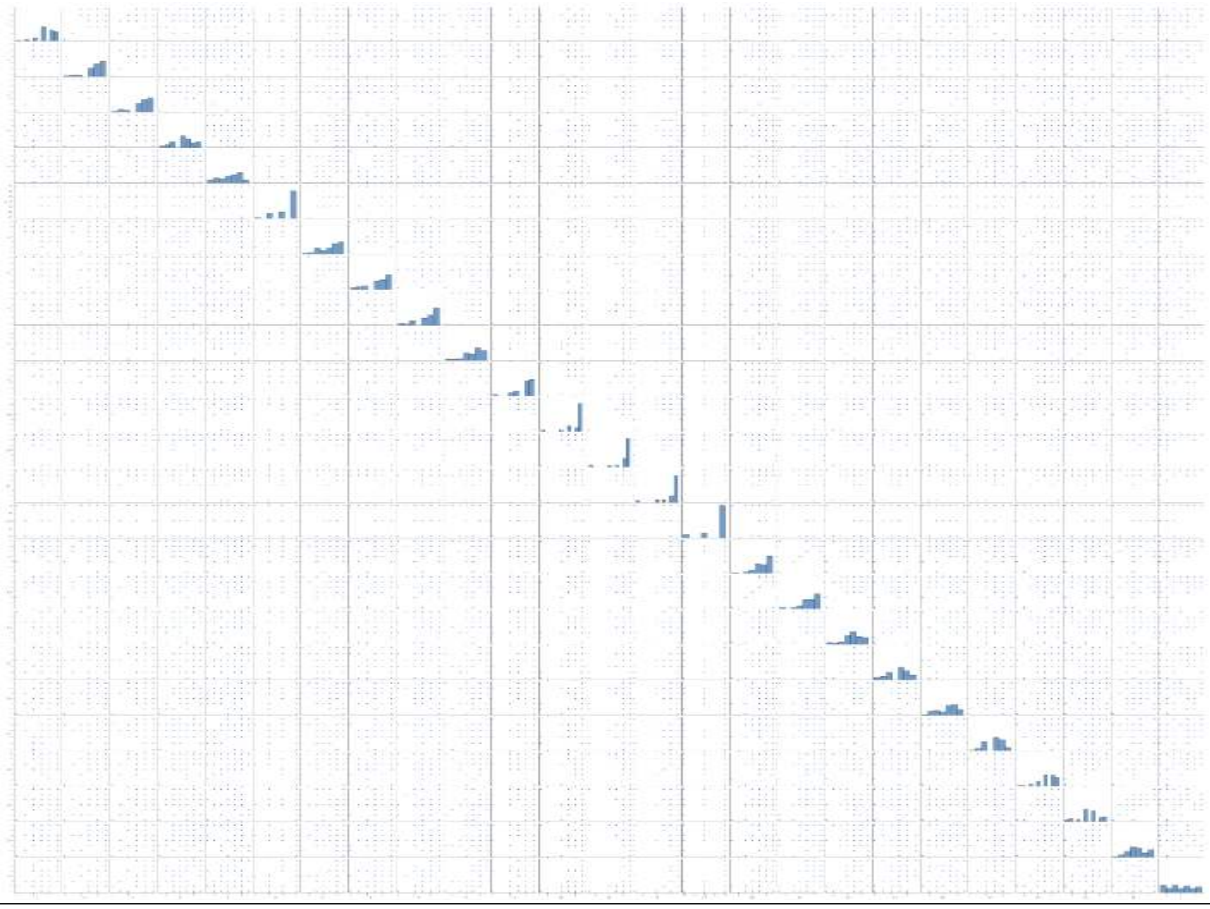
In [5]:

```
# 행렬산점도와 상관계수
data_train=x_train.join(y_train)
sns.pairplot(data = data_train)
data_train.corr()
```

Out [5]:

	x1	x2	x3	x4	x5	x6	x7	x8	x9	x10	...	
x1	1.000000	0.517710	0.339055	0.292255	-0.131226	0.056203	-0.037430	-0.393731	-0.032933	0.130907	...	0.15
x2	0.517710	1.000000	0.465474	0.019204	-0.088396	0.178321	0.066557	0.010677	-0.014468	0.126276	...	0.18
x3	0.339055	0.465474	1.000000	0.016594	-0.039562	0.025328	0.152480	0.086923	0.038383	0.336627	...	0.26
x4	0.292255	0.019204	0.016594	1.000000	-0.114553	0.042391	0.110744	-0.103935	0.001222	0.270925	...	0.08
x5	-0.131226	-0.088396	-0.039562	-0.114553	1.000000	0.063929	0.308588	0.248809	0.123789	0.185542	...	0.35
x6	0.056203	0.178321	0.025328	0.042391	0.063929	1.000000	0.210579	-0.188165	-0.025654	0.038924	...	0.08
x7	-0.037430	0.066557	0.152480	0.110744	0.308588	0.210579	1.000000	0.299043	0.363202	0.216877	...	0.14
x8	-0.393731	0.010677	0.086923	-0.103935	0.248809	-0.188165	0.299043	1.000000	0.323982	0.285195	...	0.06
x9	-0.032933	-0.014468	0.038383	0.001222	0.123789	-0.025654	0.363202	0.323982	1.000000	0.439674	...	0.21
x10	0.130907	0.126276	0.336627	0.270925	0.185542	0.038924	0.216877	0.285195	0.439674	1.000000	...	0.17
x11	-0.266868	-0.202439	-0.142908	0.036119	0.151798	-0.009364	-0.003182	0.130159	0.168530	0.025965	...	0.15
x12	0.159140	0.293633	-0.050945	0.198739	0.113388	0.306570	0.027145	-0.172201	-0.061571	0.365125	...	-0.00
x13	0.034091	0.111494	-0.191251	0.053534	0.136805	0.467474	0.083217	-0.194290	0.042237	0.341998	...	-0.07
x14	0.082753	0.094770	-0.039074	-0.027942	0.245705	0.476580	0.081062	-0.151196	-0.006226	0.386076	...	-0.03
x15	0.232643	0.347774	0.192062	0.095285	0.051212	0.271473	0.045328	-0.190921	0.161181	0.300103	...	0.17
x16	0.157534	0.189037	0.268269	0.083595	0.354361	0.081972	0.141692	0.069943	0.218287	0.172384	...	1.00
x17	0.029752	0.207676	0.304688	0.042306	0.195666	0.085715	0.015925	0.284510	0.298121	0.369464	...	0.75
x18	-0.096600	-0.061888	-0.196089	0.151542	0.332913	0.108244	0.259591	0.202132	0.059053	0.256260	...	0.26
x19	0.006606	0.020958	-0.217105	0.248700	0.395320	0.167842	0.099658	0.053812	0.005025	0.262932	...	0.30
x20	0.206181	0.180097	-0.067454	0.335441	0.286968	-0.061721	-0.131486	-0.088030	-0.268889	0.055063	...	0.17
x21	0.228563	-0.013465	-0.173307	0.076499	0.410702	0.011207	0.075332	0.134643	0.002570	0.228333	...	-0.07
x22	0.231102	0.209946	0.489818	0.205580	0.324326	-0.138906	0.161230	0.126658	-0.071161	-0.005722	...	0.42
x23	0.116311	0.317936	0.433793	0.077659	-0.012088	-0.250236	0.094409	0.292652	0.045506	0.205766	...	0.22
x24	0.135595	0.264751	0.386312	0.162070	-0.060024	-0.245899	0.037219	0.133611	0.049121	0.198353	...	-0.00
y	0.291595	0.383455	0.361498	0.246732	0.330323	0.369964	0.433775	0.211981	0.421432	0.714587	...	0.49

25 rows × 25 columns



2) train-set의 독립변수와 종속변수에 대한 다중회귀모형 분석

2-1) 코드 및 결과

```
In [6]: #[2] train-set을 이용한 다중회귀모형 분석
data_train=x_train.join(y_train)
fit_train=smf.ols('y_train~x_train', data=data_train).fit()
print(fit_train.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                y_train   R-squared:                       0.946
Model:                            OLS   Adj. R-squared:                  0.889
Method:                 Least Squares   F-statistic:                     16.73
Date:                Wed, 16 Nov 2022   Prob (F-statistic):           1.54e-09
Time:                        04:09:41   Log-Likelihood:                -32.390
No. Observations:                  48   AIC:                             114.8
Df Residuals:                      23   BIC:                             161.6
Df Model:                          24
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept    -15.4133      1.933     -7.975      0.000     -19.411     -11.415
x_train[0]     0.3103      0.180      1.724      0.098      -0.062       0.683
x_train[1]     0.1116      0.167      0.670      0.510      -0.233       0.456
x_train[2]     0.1695      0.161      1.051      0.304      -0.164       0.503
x_train[3]     0.0927      0.096      0.971      0.342      -0.105       0.290
x_train[4]    -0.0220      0.107     -0.207      0.838      -0.242       0.198
x_train[5]     0.2111      0.174      1.215      0.237      -0.148       0.570
x_train[6]     0.2464      0.101      2.435      0.023       0.037       0.456
x_train[7]     0.2657      0.130      2.049      0.052      -0.002       0.534
x_train[8]     0.1821      0.112      1.626      0.117      -0.050       0.414
x_train[9]     0.3311      0.146      2.265      0.033       0.029       0.633
x_train[10]    0.0896      0.109      0.824      0.418      -0.135       0.315
x_train[11]    0.0082      0.172      0.048      0.962      -0.347       0.363
x_train[12]   -0.0119      0.255     -0.047      0.963      -0.540       0.516
x_train[13]    0.5521      0.196      2.819      0.010       0.147       0.957
x_train[14]    0.3016      0.231      1.304      0.205      -0.177       0.780
x_train[15]    0.3446      0.159      2.163      0.041       0.015       0.674
x_train[16]    0.0852      0.165      0.517      0.610      -0.256       0.427
x_train[17]   -0.0593      0.138     -0.431      0.671      -0.344       0.226
x_train[18]    0.1873      0.139      1.350      0.190      -0.100       0.474
x_train[19]    0.0730      0.114      0.639      0.529      -0.163       0.310
x_train[20]   -0.0768      0.175     -0.439      0.665      -0.438       0.285
x_train[21]   -0.0694      0.167     -0.416      0.681      -0.415       0.276
x_train[22]    0.0335      0.180      0.186      0.854      -0.340       0.407
x_train[23]   -0.0543      0.157     -0.347      0.732      -0.378       0.270
==============================================================================
Omnibus:                        0.203   Durbin-Watson:                   1.649
Prob(Omnibus):                  0.903   Jarque-Bera (JB):                0.151
Skew:                          -0.124   Prob(JB):                        0.927
Kurtosis:                       2.880   Cond. No.                         535.
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

2-2) 결과해석

train-set이 독립변수(24개)와 종속변수에 대한 다중회귀모형 분석 결과는 아래와 같다.

우선 결정계수(R^2)가 0.946으로, 전반적인 만족도(y)에 대하여 약 95% 정도가 독립변수(x1, x2, …, x24)들에 의해 설명된다고 볼 수 있으며, 주어진 데이터에 모형이 잘 적합 되었을 것으로 예측된다.

또한 수정 결정계수(R_a^2)도 0.889로 높은 수치를 나타내어, 다른 모형들과 적합도를 비교하였을 때, 나쁘지 않을 것으로 판단된다.

다음으로 회귀모형의 가정과 관련된 통계량을 살펴보자. Durbin-Watson 통계량이 1.649로 오차의 독립성 가정이 위배되지 않는 것으로 보인다.

마지막으로 각 독립변수의 회귀계수를 살펴보면 x1, x7, x8, x10, x14, x16의 변수들을 제외한 나머지 변수들의 회귀계수들이 유의하지 않음을 확인할 수 있다. 이와 같은 사실은 모형에서 종속변수에 대한 대부분 독립변수의 영향력이 유의하지 않을 수 있다는 것을 의미한다. 유의한 변수 중 x14(β_{14}=0.521)가 영향력이 가장 높은 것으로 보인다.

3) VIF을 이용한 다중공선성 탐색

3-1) 코드 및 결과

In [7]:
```
# vif-통계량 계산
from patsy import dmatrices
from statsmodels.stats.outliers_influence import variance_inflation_factor
%matplotlib inline

y, X = dmatrices('y_train ~ x_train', data=data_train, return_type = 'dataframe')
vif = pd.DataFrame()
vif["VIF Factor"] = [variance_inflation_factor(X.values, i) for i in range(X.shape[1])]
vif["features"] = X.columns
vif
```

Out[7]:

	VIF Factor	features
0	380.548499	Intercept
1	4.537335	x_train[0]
2	4.083489	x_train[1]
3	4.186943	x_train[2]
4	2.321462	x_train[3]
5	3.312389	x_train[4]
6	1.919857	x_train[5]
7	2.878041	x_train[6]
8	3.458910	x_train[7]
9	2.323554	x_train[8]
10	4.419832	x_train[9]
11	1.827305	x_train[10]
12	5.851873	x_train[11]
13	11.268665	x_train[12]
14	7.609632	x_train[13]
15	2.032990	x_train[14]
16	4.736765	x_train[15]
17	6.182553	x_train[16]
18	4.180754	x_train[17]
19	3.546463	x_train[18]
20	3.569490	x_train[19]
21	3.956998	x_train[20]
22	5.435916	x_train[21]
23	7.741240	x_train[22]
24	6.182362	x_train[23]

3-2) 결과해석

위의 VIF 통계량을 통해 독립변수 x13이 확연히 다중공선성을 띠는 것을 확인할 수 있고, 상관된 변수는 x14, x17, x22, x23, x24 중 하나거나 혹은 그 이상일 것으로 판단된다. 그러나 그 외의 다른 변수들은 작은 VIF 통계량으로 다중공선성이 있다고 판단하기 어렵다.

3. 다중공선성을 제거하기 위해 변수선택법(후진법)을 사용하고, 적합모형을 이용해 test-set에 대해 예측하여 PRESS(예측 MSE)를 정리하라

답안)
1) train-set에 대한 변수선택법(후진선택법)

1-1) 코드 및 결과

```
In [8]: #[3] 다중공선성 제거를 위한 변수선택방법(후진법)
y=y_train
X=x_train

## 변수선택법(backward)
def backward(X, y, level, verbose=False): #후진선택법
    included=list(X.columns)    #선택된 변수를 저장할 리스트
    while True:
        changed=False
        if  (len(included)==1):
            model=sm.OLS(y, sm.add_constant(pd.DataFrame(X[included]))).fit()
        else:
            model=sm.OLS(y, sm.add_constant(pd.DataFrame(X[included]))).fit()
        pval=model.pvalues.iloc[1:]
        worst_pval=pval.max()

        if worst_pval > level:  #유의수준과 p-value를 비교해서 작으면 해당 변수를 모형에 포함
            changed = True
            worst_X=pval.idxmax()
            included.remove(worst_X)

            if verbose:
                print('DROP{:20} with p-val{:25}'.format(worst_X, worst_pval))
        if not changed:
            break
    return included      #최종 선택 변수 출력

backward(X, y, 0.05, verbose=True)    #데이터의 반응변수와 데이터 이름 입력
```

```
DROPx13                 with p-val      0.9630699109477205
DROPx12                 with p-val      0.9772612198623444
DROPx23                 with p-val      0.8503686725439825
DROPx5                  with p-val      0.7532873603658539
DROPx24                 with p-val      0.7523915161127827
DROPx18                 with p-val      0.7277908594148736
DROPx20                 with p-val      0.5721923050242743
DROPx22                 with p-val      0.7132427510644663
DROPx17                 with p-val      0.4812385663050699
DROPx21                 with p-val      0.3282472530987046
DROPx11                 with p-val      0.3230131264766204
DROPx3                  with p-val      0.1573376580253688
DROPx19                 with p-val     0.24186006370674112
DROPx6                  with p-val     0.11146612230802971
DROPx9                  with p-val     0.07205455568391747
DROPx4                  with p-val     0.08502718061941424
DROPx2                  with p-val     0.12068507022293182
```

```
Out[8]: ['x1', 'x7', 'x8', 'x10', 'x14', 'x15', 'x16']
```

1-2) 결과해석

후진선택법에 의해 x1, x7, x8, x10, x14, x15, x16의 독립변수들을 제외한 나머지 변수들은 제외되었음을 확인할 수 있다. 즉 다중공선성을 갖지 않으면서, 종속변수를 잘 설명할 수 있는 최적의 모형에 x1, x7, x8, x10, x14, x15, x16의 독립변수들만 포함된다고 볼 수 있을 것이다.

2) 후진법에 의해 선택된 변수들에 대한 다중회귀모형

2-1) 코드 및 결과

```
In [9]: # 후진법에 의한 유의하지 않은 변수 생략
x_train2 = x_train.drop(['x13', 'x12', 'x23', 'x5', 'x24',
                         'x18', 'x20', 'x22', 'x17', 'x21',
                         'x11', 'x3', 'x19', 'x6', 'x9', 'x4', 'x2'],axis=1,inplace=False)
# = x_train.loc[:,['x1', 'x7', 'x8', 'x10', 'x14', 'x15', 'x16' ]]
data_train2=x_train.join(y_train)
fit_train2=smf.ols('y_train~x_train2', data=data_train2).fit()
print(fit_train2.summary())
```

OLS Regression Results

Dep. Variable:	y_train	R-squared:	0.910
Model:	OLS	Adj. R-squared:	0.894
Method:	Least Squares	F-statistic:	57.80
Date:	Sun, 13 Nov 2022	Prob (F-statistic):	6.40e-19
Time:	22:59:59	Log-Likelihood:	-44.559
No. Observations:	48	AIC:	105.1
Df Residuals:	40	BIC:	120.1
Df Model:	7		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	-13.9870	1.363	-10.261	0.000	-16.742	-11.232
x_train2[0]	0.3733	0.096	3.900	0.000	0.180	0.567
x_train2[1]	0.2740	0.062	4.392	0.000	0.148	0.400
x_train2[2]	0.2941	0.088	3.332	0.002	0.116	0.472
x_train2[3]	0.4575	0.086	5.346	0.000	0.285	0.631
x_train2[4]	0.5790	0.084	6.917	0.000	0.410	0.748
x_train2[5]	0.5334	0.185	2.884	0.006	0.160	0.907
x_train2[6]	0.5209	0.076	6.899	0.000	0.368	0.674

Omnibus:	0.038	Durbin-Watson:	1.681
Prob(Omnibus):	0.981	Jarque-Bera (JB):	0.076
Skew:	-0.046	Prob(JB):	0.963
Kurtosis:	2.828	Cond. No.	219.

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

2-2) 결과해석

train-set에서 후진법으로 선택된 변수 조합에 의한 다중회귀모형 분석 결과는 다음과 같다.

결정계수(R^2)가 0.910으로, 전반적인 만족도(y)에 대하여 약 91% 정도가 독립변수(x1, x7, x8, x10, x14, x15, x16)들에 의해 설명된다고 볼 수 있으며, 주어진 데이터에 모형이 잘 적합 되었을 것으로 예측된다.

수정 결정계수(R_a^2)도 0.894로, 문제 2에서 확인했던 모형(Full-model)의 수정 결정계수인 0.889보다도 큰 값을 가져, 모형이 적합도 측면에서 더 낫다고 판단할 수 있다.

다음 확인한 Durbin-Watson 통계량이 1.681으로 오차의 독립성 가정이 위배되지 않는 것으로 보인다.

마지막으로 각 독립변수의 회귀계수를 살펴보면 모든 변수의 회귀계수가 유의함을 확인할 수 있고, 그 중에서도 x14의 회귀계수가 0.5790으로 종속변수에 대한 영향력이 가장 클 것으로 보인다. 이외에도 x15, x16의 회귀계수를 통해, x14에 비등한 영향력을 가질 것으로 생각될 수 있다. 반면에 x7의 회귀계수는 0.2740으로 종속변수에 대한 가장 적은 영향력을 가진 것으로 보인다.

따라서 후진법의 결과에 따른 회귀모형에서, 전반적인 만족도(y)에 대해 거래내역의 비밀보장(x14), 안전한 거래를 위한 시스템 확보(x15), 빠른 접속 속도(x16)의 중요도가 높고, 가입절차의 편리성(x7)의 중요도가 가장 낮다고 설명될 수 있다.

3) test-set에 대한 PRESS(예측 MSE)

3-1) 코드 및 결과

In [11]:
```
# test-set을 예측하기 위해 학습모형에 없는 변수 제거
x_test2 = x_test.drop(['x13', 'x12', 'x23', 'x5', 'x24',
                       'x18', 'x20', 'x22', 'x17', 'x21',
                       'x11', 'x3', 'x19', 'x6', 'x9', 'x4', 'x2'],axis=1,inplace=False)
# = x_test2.loc[:,['x1', 'x7', 'x8', 'x10', 'x14', 'x15', 'x16' ]]
```

In [12]:
```
# Linear Regression OLS로 학습/예측평가 수행.
reg= LinearRegression()
reg.fit(x_train2 ,y_train )
y_preds= reg.predict(x_test2)
mse = mean_squared_error(y_test, y_preds)
rmse = np.sqrt(mse)
```

In [13]:
```
# MSE, RMSE, R^2, 설명분산점수(적합모형의 편향성 평가 : 0에 가까워야)
print('MSE : {0:.3f} , RMSE : {1:.3F}'.format(mse , rmse))
print('Variance score : {0:.3f}'.format(r2_score(y_test, y_preds)))
```

```
MSE : 2.111 , RMSE : 1.453
Variance score : 0.146
```

3-2) 결과해석

train-set으로 적합된 회귀모형이 x1, x7, x8, x10, x14, x15, x16의 독립변수만을 포함하는 모형이므로, 모형을 적용하여 예측값을 구하기 위해선 test-set의 데이터에서 또한 모형에 포함된 독립변수를 제외하고 나머지 변수들을 제거하여야 한다. 위의 코드에서 x_test2가 불필요한 독립변수를 제거한 데이터이다.

결과적으로 test-set에 대한 예측 MSE(PRESS) 값은 2.111, RMSE 값은 1.453으로 계산됨을 알 수 있다.

또한 오차의 편향을 나타내는 설명분산점수(Variance score)(0.146)와 결정계수(0.910)가 차이가 0.764로 편향이 크지 않은 것으로 판단된다.

4) 다중공선성을 제거하기 위해 PCA(주성분분석)을 수행하라. 적당한 주성분의 수를 찾고(scree-plot) 주성분 식을 정리하여 각 주성분의 적절히 변수 이름으로 정한 후 주성분들을 독립변수로 하여 종속변수(y)에 대해 다중회귀모형 분석을 수행하고 모형의 적합성(다중공선성, 잔차독립성, R^2 등)과 변수의 중요도(회귀계수)를 상세히 해석하라.

답안)

1) train-set의 전처리 과정(reindex)

1-1) 코드 및 결과

In [14]:
```
# [4]
# train-set reindex
data_train3=data_train.reset_index().drop(['index'], axis=1, inplace=False)
data_train3.head()
```

Out[14]:

	x1	x2	x3	x4	x5	x6	x7	x8	x9	x10	...	x16	x17	x18	x19	x20	x21	x22	x23	x24	y
0	7	5	5	5	6	7	4	5	6	5	...	6	6	6	5	6	5	6	2	3	3
1	6	7	7	4	7	7	7	7	7	7	...	7	7	7	7	7	7	7	3	3	7
2	6	6	6	5	1	7	4	4	5	5	...	1	1	1	2	3	4	2	1	4	2
3	7	6	7	4	3	7	7	3	7	5	...	7	7	4	4	6	5	6	7	7	5
4	5	5	7	4	3	5	4	7	7	4	...	7	6	3	3	3	3	7	7	7	1

5 rows × 25 columns

2) PCA분석

2-1) 코드 및 결과

In [15]:
```
# 다중공선성 제거를 위한 PCA수행
# X-Y변수 추출
from sklearn.preprocessing import StandardScaler  # 표준화 패키지 라이브러리
x = x_train
# 표준화(Z)
x = StandardScaler().fit_transform(x) # x객체에 x를 표준화한 데이터를 저장
```

In [16]:
```
# PCA모듈설치 및 PCA분석+주성분의 회귀계수를 데이터프레임으로 구성
from sklearn.decomposition import PCA
pca = PCA(n_components=9)   # 주성분을 몇개로 할지 결정
printcipalComponents = pca.fit_transform(x)
principalDf = pd.DataFrame(data=printcipalComponents, columns =
                           ['PC1', 'PC2', 'PC3', 'PC4', 'PC5', 'PC6','PC7', 'PC8', 'PC9'])
principalDf.head()   # 주성분들의 점수
```

Out[16]:

	PC1	PC2	PC3	PC4	PC5	PC6	PC7	PC8	PC9
0	-0.837903	0.811865	0.302896	0.865092	-1.575929	0.308363	0.834821	0.228170	0.838689
1	-3.670319	-1.544701	1.601641	0.069209	-0.515734	-0.540594	1.262401	0.995586	0.550466
2	2.185209	4.964951	-3.834424	-1.667829	1.435644	1.194361	0.945531	0.690479	-1.255352
3	0.331399	-2.148657	-1.427858	-0.685616	-0.945581	0.427035	0.676515	-0.047651	-0.105584
4	6.908030	-3.458597	1.707233	-0.419003	-1.543667	1.428070	0.173123	1.088632	-0.152547

2-2) 결과해석
PCA 분석과정에서 주성분의 개수를 임의로 9개로 정하였고(n_components=9), 그 결과로 기존의 독립변수(24개)의 값들이 주성분값으로 변환되어 출력되었다.

3) 각 주성분의 설명력과 누적설명력

3-1) 코드 및 결과

```
In [19]: # 각 주성분의 설명력
         print(pca.explained_variance_ratio_)
          # 누적설명력
         print(sum(pca.explained_variance_ratio_))
```

```
[0.19171385 0.16141056 0.11159085 0.08837884 0.06746586 0.05277534
 0.04847686 0.04263884 0.0390553 ]
0.8035063162224182
```

3-2) 결과해석

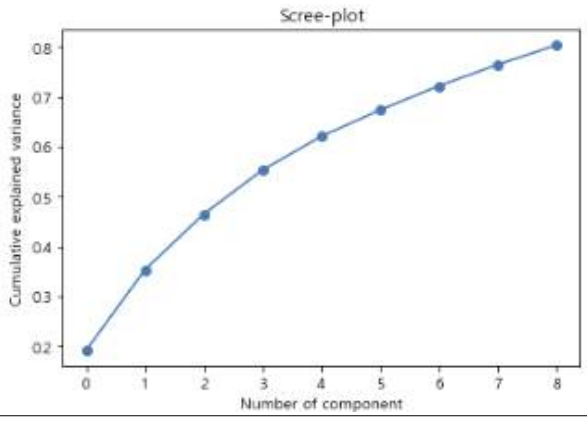
9개의 주성분의 설명력은 위와 같다.
주성분의 설명력 중 가장 높은 값은 약 0.19이고, 가장 낮은 값은 약 0.04 정도로 나타난 것을 볼 수 있다.
또한 주성분들의 누적설명력이 약 0.80으로 나타난 것을 통해 전체 변동의 80%정도를 설명한다는 것을 알 수 있다. 그러나 주성분의 누적설명력에서 제7 주성분과 제8 주성분, 제9 주성분의 설명력이 비슷하게 작게 나타나는 것을 통해, 주성분의 수를 줄이는 것이 나을 것으로 판단된다.

4) screen-plot

4-1) 코드 및 결과

```
In [19]: # screen-plot
         import numpy as np
         import matplotlib.pylab as plt
         exp_var_cum = np.cumsum(pca.explained_variance_ratio_)
         plt.title('Scree-plot')
         plt.xlabel('Number of component')
         plt.ylabel('Cumulative explained variance')
         plt.plot(exp_var_cum, 'o-')
```

```
Out[19]: [<matplotlib.lines.Line2D at 0x217b49396a0>]
```



4-2) 결과해석

위의 그래프는 주성분의 수에 따른 누적설명력의 screen-plot이다.
제6 주성분 이후의 주성분들의 기울기가 매우 완만하고, 기울기의 변화(설명력)에 거의 변화가 없음을 확인할 수 있다. 따라서 7, 8, 9 주성분을 제거하는 것이 적절해 보인다.

5) 주성분을 6개로 하여, PCA 수행

5-1) 코드 및 결과

```
In [20]: # 주성분을 6개로 정하고 다시 PCA분석
from sklearn.decomposition import PCA
pca = PCA(n_components=6)
printcipalComponents = pca.fit_transform(x)
principalDf = pd.DataFrame(data=printcipalComponents, columns =
                           ['PC1', 'PC2', 'PC3', 'PC4', 'PC5', 'PC6'])
print(pca.components_)
```

```
[[-0.08392858 -0.09850562  0.02637049 -0.10926036 -0.20092747 -0.20950002
  -0.10341146  0.01607888 -0.05870581 -0.23933288 -0.03846139 -0.36312297
  -0.37490634 -0.37337137 -0.23407892 -0.12888063 -0.13317555 -0.28169232
  -0.33503535 -0.1950908  -0.24020019  0.00386219  0.10286908  0.11953275]
 [-0.15354794 -0.22660633 -0.33202132 -0.11395419 -0.1047735   0.09901248
  -0.11976288 -0.17405484 -0.11350375 -0.18651917  0.05608839  0.08299899
   0.18520819  0.13852587 -0.04841884 -0.28773736 -0.30543556 -0.04798712
  -0.03701855 -0.14164856  0.02022364 -0.3715784  -0.40920362 -0.34225481]
 [-0.38861488 -0.33453663 -0.21341829 -0.09538713  0.29770726 -0.12441625
   0.17408843  0.36828537  0.21949617  0.03257178  0.28306148 -0.19783973
  -0.12861776 -0.13030148 -0.17840737  0.14249046  0.13568553  0.26885602
   0.17706535 -0.06275386  0.09064959  0.00639346 -0.04651505 -0.14766172]
 [ 0.0769858  -0.1052393  -0.22178386  0.18768423  0.10678517 -0.2094188
  -0.21996444 -0.14020803 -0.41071639 -0.25281813 -0.06316885 -0.00320815
  -0.11313677 -0.1165692  -0.21017259 -0.05591458 -0.15476136  0.19529274
   0.24112187  0.48302845  0.23364934  0.22200882  0.05120252  0.07029553]
 [-0.06867593 -0.03751513 -0.04347825  0.03657941 -0.01126084 -0.1864464
   0.19135432  0.29852536  0.11479485  0.2877334  -0.31467252  0.12185852
   0.09509878  0.08479084 -0.17705735 -0.47038096 -0.27428337  0.03783133
  -0.08678779 -0.01525148  0.30824421 -0.17080108  0.20897169  0.31474344]
 [ 0.16775146 -0.15780626 -0.12899457  0.61313055 -0.34640946 -0.22193935
  -0.10051652 -0.11214127  0.28741957  0.27774251  0.32193206 -0.01824204
  -0.0639445  -0.19003982  0.16537255 -0.03795506 -0.03011493  0.01978227
   0.07887393 -0.00079511 -0.02199369 -0.12889469 -0.0687798   0.01968032]]
```

5-2) 결과해석

주성분의 수를 6개로 줄인 후의 PCA 분석 결과이다.

또한 주성분을 기존의 독립변수(24개)들의 선형결합으로 나타내었을 때, 독립변수의 계수(pca.components)들을 보여줌으로써 주성분에 미치는 독립변수의 영향력을 확인할 수 있다. 이처럼 계수를 확인하여, 주성분의 성격을 특징지을 수 있을 것이다.

제1 주성분(PC1)부터 독립변수들의 계수를 살펴보면, x12, x13, x14, x19가 높은 것으로 나타났다.

다음으로 제2 주성분(PC2)에서 독립변수 계수는 x3, x22, x23, x24이 높은 것으로 나타났으며, 제3 주성분(PC3)에서 독립변수 계수는 x1, x2, x5, x8이, 제4 주성분(PC4)에서는 x9, x10, x19, x20, x21이 높은 것으로 나타났다.

그리고 제5 주성분(PC5)에서는 x8, x10, x11, x16, x17, x21이 높았고, 마지막으로 제6 주성분(PC6)은 4, x5, x16가 높은 것으로 나타났다.

6) 누적설명력

6-1) 코드 및 결과

```
In [21]: # 누적설명력
print(sum(pca.explained_variance_ratio_))
```

```
0.6733353138978313
```

6-2) 결과해석

약 0.7 정도로 전체 변동의 70% 정도를 설명한다.

주성분의 수를 줄이기 전의 주성분의 누적 설명력에 비해 10%정도의 설명력이 감소하였으나, 전체 변동을 설명하는 데 무리가 없을 것으로 판단된다.

7) 주성분의 성격에 따른 변수 이름 정의

7-1) 코드 및 결과

```
In [22]:
# 주성분의 적절한 이름을 붙여 새로운 변수 생성
data_train3['security']=principalDf['PC1']      # 보안(security) : x12, x13, x14, x19
data_train3['introduction']=principalDf['PC2'] # (시스템의)도입(introduction): (x3), x22, x23, x24
data_train3['promotion']=principalDf['PC3']     # 홍보활동(marketing):x1, x2, (x5), x8
data_train3['variety']=principalDf['PC4']       # 다양성(variety) :x9, (x10), (x19), x20, (x21)
data_train3['utility']=principalDf['PC5']       # 유용성(utility) : x8, x10, x11, x16, x17, x21
data_train3['education']=principalDf['PC6']     # 교육(education): x4, x5, x18
data_train3.head()
```

Out [22]:

	x4	x5	x6	x7	x8	x9	x10	...	x22	x23	x24	y	security	introduction	promotion	variety	utility	education
5	6	7	4	5	6	5	...	6	2	3	3	-0.837903	0.811865	0.302896	0.865092	-1.575929	0.308363
'	4	7	7	7	7	7	7	...	7	3	3	7	-3.670319	-1.544701	1.601641	0.069209	-0.515734	-0.540594
5	1	7	4	4	5	5	...	2	1	4	2	2.185209	4.964951	-3.834424	-1.667829	1.435644	1.194361
'	4	3	7	7	3	7	5	...	6	7	7	5	0.331399	-2.148657	-1.427858	-0.685616	-0.945581	0.427035
'	4	3	5	4	7	7	4	...	7	7	7	1	6.908030	-3.458597	1.707233	-0.419003	-1.543667	1.428070

7-2) 결과해석

5)에서 나타난 각 주성분을 선형결합식으로 나타내었을 때, 계수들을 통해 주성분에 대한 영향력이 큰 독립변수들을 확인하였다. 위의 과정은 영향력이 큰 독립변수들을 위주로 주성분의 성격을 특징짓는 과정이다.

제1 주성분(PC1)은 정보보안에 관한 변수들(x12, x13, x14, x19)의 영향력이 크므로 'security(보안)'로 표현되었다. 다음으로 제2 주성분(PC2)은 새로운 시스템의 도입에 관한 변수들(x22, x23, x24)의 영향력이 주로 큰 것으로 나타나 'introduction(도입)'으로 표현되었다. 제3 주성분(PC3)은 마케팅과 서비스에 대한 인식에 관련된 변수들(x1, x2, x5)의 영향력이 크게 나타나 'marketing(홍보활동)'으로 표현되었다.

제4 주성분(PC4)과 제5 주성분(PC5)은 비슷한 성격을 가진 것으로 보인다. 하지만 제4 주성분은 다양한 서비스와 관련된 변수(x19, x20, x21)의 영향력으로 'variety(다양성)'로 표현하였고, 제5 주성분은 편리함과 관련된 변수(x8, x10, x11, x16, x17, x21)의 영향력으로 'utility(유용성)'로 표현되었다. 마지막으로 제6 주성분은 사용자 교육에 대한 변수(x4)의 영향력이 압도적으로 높아 'education(교육)'으로 표현되었다.

8) 6개의 주성분을 독립변수로 하는 다중회귀모형 분석(변수 y_train1은 train_set의 y의 표준화 값)

8-1) 코드 및 결과

```
In [40]:
fit_pca=smf.ols('y_train1 ~ security + introduction + promotion + variety + utility + education'
                , data=data_train4).fit()
print(fit_pca.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                y_train1   R-squared:                       0.902
Model:                            OLS   Adj. R-squared:                  0.887
Method:                 Least Squares   F-statistic:                     62.67
Date:                Thu, 17 Nov 2022   Prob (F-statistic):           4.44e-19
Time:                        06:03:36   Log-Likelihood:                -12.441
No. Observations:                  48   AIC:                             38.88
Df Residuals:                      41   BIC:                             51.98
Df Model:                           6
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept   6.939e-17      0.049   1.42e-15      1.000      -0.099       0.099
security      -0.3543      0.023    -15.521      0.000      -0.400      -0.308
introduction  -0.2130      0.025     -8.561      0.000      -0.263      -0.163
promotion     -0.0006      0.030     -0.019      0.985      -0.061       0.060
variety       -0.2601      0.034     -7.737      0.000      -0.328      -0.192
utility       -0.0095      0.038     -0.247      0.806      -0.087       0.068
education      0.0599      0.044      1.376      0.176      -0.028       0.148
==============================================================================
Omnibus:                        0.124   Durbin-Watson:                   2.058
Prob(Omnibus):                  0.940   Jarque-Bera (JB):                0.315
Skew:                          -0.073   Prob(JB):                        0.854
Kurtosis:                       2.631   Cond. No.                         2.15
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

8-2) 결과해석

train-set에서 PCA로 결정된 주성분(독립변수)과 종속변수(y)에 대한 다중회귀모형 분석은 다음과 같다.

결정계수(R^2)가 0.902으로, 전반적인 만족도(y)에 대하여 약 90% 정도가 주성분들에 의해 설명된다고 볼 수 있으며, 주어진 데이터에 모형이 잘 적합 되었을 것으로 예측된다.

수정 결정계수(R_a^2)도 0.887로, 문제 2에서 확인했던 모형(Full-model)의 수정 결정계수인 0.889보다 미세하게 작으나 그 차이가 유의미하다고 보기 어렵다고 판단된다.

다음 확인한 Durbin-Watson 통계량이 2.058로 오차(잔차)의 독립성 가정이 위배되지 않는 것으로 보인다.

각 주성분(독립변수)의 회귀계수를 살펴보면, 변수 promotion(제3 주성분), utility(제5 주성분), education(제6 주성분)이 유의하지 않음을 확인할 수 있다. 그러므로 3개의 변수를 제외하여 새로운 회귀모형을 적합하는 것이 적절해 보인다. 유의한 주성분에서 중요도는 'security', 'variety', 'introduction' 순서로 나타났다.

마지막으로 주성분(독립변수)들 간에는 서로 독립이므로, 다중공선성은 존재하지 않으며 또한 Cond.No(상태수)가 2.15로 매우 작아, 마찬가지로 다중공선성이 존재하지 않을 것으로 예측할 수 있다.

9) 회귀계수가 유의하지 않은 독립변수(주성분)을 제거한 다중회귀모형 분석(변수 y_train1은 train_set의 y의 표준화 값)

9-1) 코드 및 결과

```
In [41]: fit_pca=smf.ols('y_train1 ~ security + introduction + variety'
                         , data=data_train4).fit()
         print(fit_pca.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:               y_train1   R-squared:                       0.897
Model:                            OLS   Adj. R-squared:                  0.890
Method:                 Least Squares   F-statistic:                     127.7
Date:                Thu, 17 Nov 2022   Prob (F-statistic):           9.81e-22
Time:                        06:04:35   Log-Likelihood:                -13.558
No. Observations:                  48   AIC:                             35.12
Df Residuals:                      44   BIC:                             42.60
Df Model:                           3
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept   6.939e-17      0.048   1.43e-15      1.000      -0.098       0.098
security      -0.3543      0.023    -15.709      0.000      -0.400      -0.309
introduction  -0.2130      0.025     -8.665      0.000      -0.263      -0.163
variety       -0.2601      0.033     -7.830      0.000      -0.327      -0.193
==============================================================================
Omnibus:                        3.329   Durbin-Watson:                   2.154
Prob(Omnibus):                  0.189   Jarque-Bera (JB):                1.772
Skew:                          -0.160   Prob(JB):                        0.412
Kurtosis:                       2.115   Cond. No.                         2.15
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

9-2) 결과해석

회귀계수가 유의하지 않은 독립변수(주성분)을 제거한 다중회귀모형 분석은 다음과 같다.

결정계수(R^2)가 0.897으로, 전반적인 만족도(y)에 대하여 약 90% 정도가 주성분들에 의해 설명된다고 볼 수 있으며, 주어진 데이터에 모형이 잘 적합 되었을 것으로 예측된다.

수정 결정계수(R_a^2)도 0.890로, 문제 2에서 확인했던 모형(Full-model)의 수정 결정계수인 0.889보다 미세하게 크나 그 차이가 유의미하다고 보기 어렵다고 판단된다.

다음 확인한 Durbin-Watson 통계량이 2.154로 오차(잔차)의 독립성 가정이 위배되지 않는 것으로 보인다.

각 주성분(독립변수)의 회귀계수를 살펴보면, 모든 회귀계수가 유의하고, -0.3543으로 'security'가 가장 영향력이 커 중요한 변수라는 것을 알 수 있다. 또한 'variety', 'introduction' 순서로 중요함을 확인할 수 있다.

마지막으로 주성분(독립변수)들 간에는 서로 독립이므로, 다중공선성은 존재하지 않으며 또한 Cond.No(상태수)가 2.154로 매우 작아, 마찬가지로 다중공선성이 존재하지 않을 것으로 예측할 수 있다.

5. test-set 20%에 대해 train-set의 모형을 적용하여 예측하고, PRESS(prediction MSE)를 계산하여 변수선택법의 prediction MSE와 비교하여 보아라.

답안)
1) test-set의 독립변수(x_test)와 종속변수(y_test)의 표준화

1-1) 코드 및 결과

```
In [25]: [5]
from sklearn.preprocessing import StandardScaler  # 표준화 패키지 라이브러리
# 표준화(Z)
x_test1 = pd.DataFrame(data=x_test)
x_test1 = StandardScaler().fit_transform(x)
y_train1= pd.DataFrame(data=y_train)
y_train1= StandardScaler().fit_transform(y_train1)
y_test1=pd.DataFrame(data=y_test)
y_test1=StandardScaler().fit_transform(y_test1)
```

1-2) 결과해석

test-set의 독립변수 값을 train-set 모형에 적용하기 위해선, test-set의 독립변수들을 주성분 값으로 변환하여야 한다. 따라서 주성분 값을 구하기 위해서 독립변수를 표준화하는 과정이 필요하다. 또한 test-set의 독립변수 값들을 표준화 했기 때문에, 종속변수 또한 표준화 해야 한다.

2) train-set 모형에 적용하기 위해 test-set의 독립변수(x) 데이터의 주성분 변환

1-1) 코드 및 결과

```
In [31]: x_train_PC1=data_train3.loc[:,['security','introduction','variety'] ]
x_test1=pd.DataFrame(data=x_test1)
test_PC1=x_test1.iloc[:, 0:24]*pca.components_[0]
test_PC2=x_test1.iloc[:, 0:24]*pca.components_[1]
test_PC3=x_test1.iloc[:, 0:24]*pca.components_[3]
x_test_PC=pd.DataFrame()
x_test_PC['security']=test_PC1.sum(axis=1)
x_test_PC['introduction']=test_PC2.sum(axis=1)
x_test_PC['variety']=test_PC3.sum(axis=1)
x_test_PC
```

Out[31]:

	security	introduction	variety
0	0.091358	1.031659	0.004888
1	-2.409337	-3.122707	0.428457
2	-1.975873	-0.265940	-0.684595
3	-0.426884	-1.208133	0.516020
4	1.268898	-0.076332	-0.480233
5	-1.865731	-1.314760	-0.411208
6	-0.111985	1.182930	-0.234353
7	0.314769	3.057397	-3.191346
8	-1.609181	-0.102345	0.841227
9	-0.978530	0.788280	0.253970
10	3.520971	-1.087748	1.202931
11	4.181526	1.117697	1.754242

1-2) 결과해석

train-set 모형에서 주성분 'security', 'introduction', 'variety'만을 포함하고 있다.
따라서 test-set을 모형에 적용하기 위해서, test-set의 독립변수들을 주성분의 형태로 변환시켜야 한다.
train-set의 주성분('security', 'introduction', 'variety')의 선형결합식의 계수와 test-set의 표준화 시킨 독립변수의 값들을 각각 곱하게 되면, 모형에 적용할 수 있는 test-set의 주성분 값을 구할 수 있다.

3) 모형 적용 및 예측 수행

3-1) 코드 및 결과

```
In [33]: # Linear Regression OLS로 학습/예측평가 수행.
reg= LinearRegression()
reg.fit(x_train_PC1, y_train1)
y_preds= reg.predict(x_test_PC)
mse = mean_squared_error(y_test1, y_preds)
rmse = np.sqrt(mse)

In [34]: # MSE, RMSE, R^2, 설명분산점수(적합모형의 편향성 평가 : 0에 가까워야)
print('MSE : {0:.3f} , RMSE : {1:.3F}'.format(mse , rmse))
print('Variance score : {0:.3f}'.format(r2_score(y_test1, y_preds)))

MSE : 0.271 , RMSE : 0.520
Variance score : 0.729
```

3-2) 결과해석

test-set에 대한 예측 MSE(PRESS) 값은 0.271, RMSE 값은 0.520로 계산됨을 알 수 있다.
또한 오차의 편향을 나타내는 설명분산점수(Variance score)(0.729)와 결정계수(0.897)가 차이가 0.168로 매우 작아, 편향이 거의 나타나지 않는 것을 확인할 수 있다.

4) 변수 선택법 예측 MSE와 비교

4-1) 코드 및 결과

*변수 선택법 예측 MSE

```
In [13]: # MSE, RMSE, R^2, 설명분산점수(적합모형의 편향성 평가 : 0에 가까워야)
print('MSE : {0:.3f} , RMSE : {1:.3F}'.format(mse , rmse))
print('Variance score : {0:.3f}'.format(r2_score(y_test, y_preds)))

MSE : 2.111 , RMSE : 1.453
Variance score : 0.146
```

*PCA 회귀모형 예측 MSE

```
In [34]: # MSE, RMSE, R^2, 설명분산점수(적합모형의 편향성 평가 : 0에 가까워야)
print('MSE : {0:.3f} , RMSE : {1:.3F}'.format(mse , rmse))
print('Variance score : {0:.3f}'.format(r2_score(y_test1, y_preds)))

MSE : 0.271 , RMSE : 0.520
Variance score : 0.729
```

4-2) 결과해석

PCA 수행 후, 회귀모형의 예측 MSE가 0.271로, 변수 선택법 모형의 예측 MSE(2.111)보다 1.84정도 작다. 따라서 PCA 적용 회귀모형이 전반적인 만족도(y)에 대하여 예측을 더 잘한다고 말할 수 있다.