선형회귀, 로지스틱회귀 분석 보고서

정보융합학부 2021204051 우지윤

2023.11.19

목차

1. 일상생활에서 볼 수 있는 Odds 사례를 한 가지 찾아보고 설명하시오.

승산(Odds)은 성공 확률 P 에 대해, 성공 대비 실패의 확률 비율로 정의한다. (Odds = p/(1-p)) 여기서 P=1 인 경우, Odds 는 무한대의 값을 가진다. P 가 0 인 경우, Odds 는 0 의 값을 가진다. 즉, 승산(Odds)은 Y 가 1 의 값을 가질 확률과 Y 가 0 의 값을 가질 확률의 비율을 의미한다.

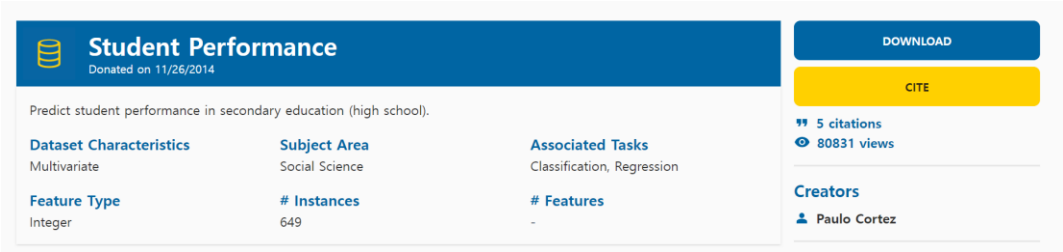
간단한 예를 들자면, 1 이 나올 주사위 횟수 1 회, 1 이 나오지 않을 주사위 횟수 5 회인 경우에 승산은 1:5 가 된다. 승산에 대해 조금 더 자세하게 설명하면, 동일 집단 내에서 어떤 사건이 발생할 확률과 발생하지 않은 확률의 비교 값이다. 즉, 일어날 확률이 일어나지 않을 확률의 몇 배인가를 보여주는 수치이다. 주사위를 던져 1 이 나올 확률은 1/6 이고, 나오지 않을 확률은 5/6 인 경우 승산은 (1/6)/(5/6)=1/5 가 되어 0.2 배가 된다.

	감염	미감염
백신 접종	10	94
백신 미접종	33	78
합계	43	172

위는 2021 년 국내 의학 연구소에서 코로나 19 백신을 대상으로 이중맹검(Double blind)을 실시한 결과를 그린 표이다. 의약 통계학에서는 후향적 연구, 또는 환자-대조군 연구라는 것을 실시한다고 한다. 환자들과 병에 걸리지 않은 대조군을 일정 숫자만큼 뽑아 백신 접종자와 미접종자가 얼마나 있는지를 살펴보는 것이다. 그 결과, 위와 같은 결과가 얻어졌다. 환자-대조군 연구에서는 위험도나 상대위험도를 계산하는 대신 오즈비를 사용하는데, 환자군에서 감염 오즈(Odds)는 (10/43)/(33/43)=10/33=0.303 이고, 대조군에서 감염 오즈(Odds)는 (94/172)/(78/172)=94/78=0.25 이다. 즉, 백신 접종자는 코로나 19 에 감염될 오즈가 미접종자의 25%에 불과하다는 것을 알 수 있다.

2-1. 선형회귀분석 - Student Performance Dataset

UCI Data Repository 사이트에서 선형회귀(Linear Regression)에 적합한 데이터를 선정하기 위하여 Task 를 'Regression'으로, Attribute 와 Instance 의 개수를 '10 개에서 1,000 개 사이'라는 검색 조건을 설정하여 검색하였다. 검색하여 나온 여러 데이터 중, 'Student Performance' 데이터를 사용하기로 하였다.



필요한 라이브러리를 import 하고, 데이터(df)를 불러와 데이터의 첫 5 행(df.head())과 마지막 2 행(df.tail(2))을 확인하였다. 본 데이터는 395 개의 행과 33 개의 열로 이루어져 있으며(len(df), df.shape), 수치형과 문자열형의 열 데이터가 혼합되어 있고(df.info()), 수치형 데이터의 통계량(df.describe())을 확인할 수 있었다. 그 후, 데이터의 고유값 수를 파악하였고(df.nunique()) 결측치가 없다는 것을(df.isnull().sum()) 확인하였다.

데이터 첫 5행

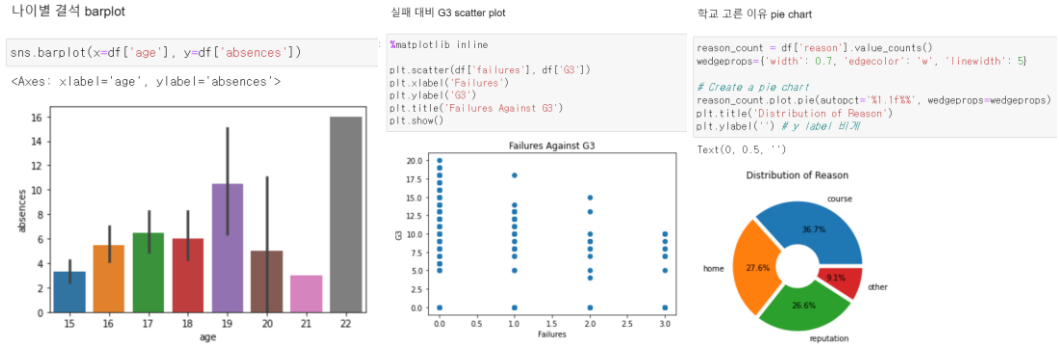
```
df.head()
```

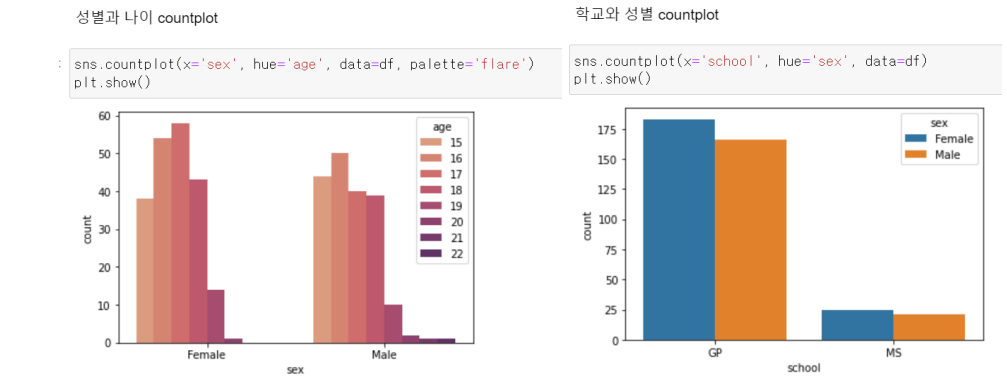
	school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	Fjob	...	famrel	freetime	goout	Dalc	Walc	health	absences	G1	G2	G3
0	GP	F	18	U	GT3	A	4	4	at_home	teacher	...	4	3	4	1	1	3	6	5	6	6
1	GP	F	17	U	GT3	T	1	1	at_home	other	...	5	3	3	1	1	3	4	5	5	6
2	GP	F	15	U	LE3	T	1	1	at_home	other	...	4	3	2	2	3	3	10	7	8	10
3	GP	F	15	U	GT3	T	4	2	health	services	...	3	2	2	1	1	5	2	15	14	15
4	GP	F	16	U	GT3	T	3	3	other	other	...	4	3	2	1	2	5	4	6	10	10

5 rows × 33 columns

데이터의 내재적인 특징을 가시화하기 위하여 강의 시간에 배운 bar plot, count plot, scatter plot, pie chart, histogram, cat plot, correlation matrix 를 사용하여 데이터 특징을 인지할 수 있도록 가시화하였다. 여러 변수를 살펴보고 시각화 과정에 적용하고자 하였다. 나이별 결석 횟수의 bar plot, 건강 상태별 공부 시간의 bar plot, 엄마 직업과

아빠 직업의 scatter plot, 실패 대비 G3 의 scatter plot, 근무일과 주일 알코올 소비량의 scatter plot 을 확인하였다. 등교 시간의 pie chart, 학교를 고른 이유의 pie chart, 학생 보호자의 pie chart, 가족 관계 품질의 histogram, 하교 후 자유시간의 histogram, 학교의 count plot, 성별의 count plot, 학교와 성별의 count plot, 성별과 나이의 count plot 도 확인하였다. 그리고 학교와 성별, 나이의 cat plot 과 거주지의 count plot 을 확인하였다.





우리는 모델을 개발할 때, 범용적으로 사용할 수 있는 모델을 만들려고 한다. Unseen data 에 대하여 예측 또는 분류하는 것이 목표이며, 여기서 unseen data 란 우리가 만든 모델이 처음 보는 데이터셋 또는 학습해 보지 않은 데이터를 의미한다. 모델이 실제로

적용되었을 때는 처음 보는 데이터에 대하여 예측 또는 분류를 수행할 것이기 때문에, 그만큼 모델의 unseen data 에 대한 성능이 좋아야 한다. 그래서 dataset 을 나누지 않고 전부 학습에 사용해 버린다면 해당 데이터셋에 대해서만 성능이 좋은 모델이 될 것이고, 개발한 모델에 대한 성능을 점검하지 못할 뿐만 아니라 실제로 모델이 적용되었을 때 좋은 성능을 기대하지 못할 것이다. 따라서 우리가 보유한 데이터셋을 어떻게 나누고 사용할지 결정하는 것은 매우 중요하다.

그럼 어떻게 데이터셋을 분할시키는 게 좋을까? 가장 기본적인 분할법은 전체 데이터셋을 train set 과 test set 으로 나누는 것이다. 이는 train set 을 이용하여 모델을 학습시키고, test set 을 이용하여 모델의 성능을 평가하기 위함이다. 그러나 단순히 train set 과 test set 으로만 나누게 된다면 모델의 성능 검정을 한 번 밖에 할 수 없고, test set 에 대한 결과를 토대로 모델을 수정하게 된다면 과적합(Overfitting)이 발생할 가능성이 커진다. 띠라서 우리는 train set 을 한 번 더 train set 과 validation set 으로 나눈다. 대다수의 경우에는 데이터셋을 train, validation, test set 을 각 6:2:2 의 비율로 나누어 사용한다.

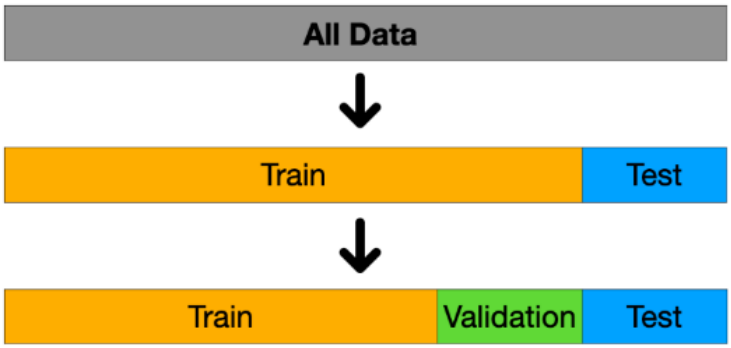


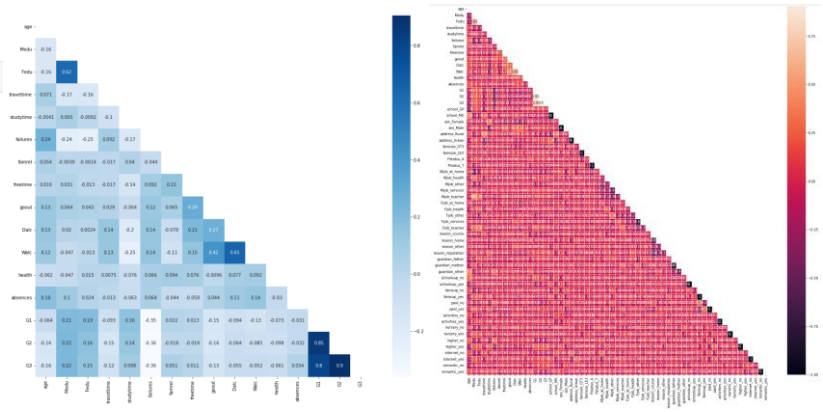
그림 1. 데이터 분할

모델 구축 전, 우선 데이터의 컬럼을 확인하고(df.columns) 컬럼의 자료형이 int64 가 아니라면, categorical 리스트에 추가하는 반복문을 사용하였다. 그 결과, 총 17 개의 컬럼이 리스트에 추가된 것을 알 수 있었다. 이 컬럼들을 제외한 수치형 데이터 컬럼 간의 heatmap 을 확인하였으며, 상관계수를 이용하여 clustermap 또한 확인할 수 있었다. get_dummies 를 활용하여 위 categorical 리스트의 변수들의 더미(지표) 변수를 생성하여 최종 데이터를 확인하였다. 그리고 최종 데이터의 상관계수와 전체 컬럼의 히트맵을 도출하였다.

그 결과

```
categorical_features
```

```
['school',
 'sex',
 'address',
 'famsize',
 'Pstatus',
 'Mjob',
 'Fjob',
 'reason',
 'guardian',
 'schoolsup',
 'famsup',
 'paid',
 'activities',
 'nursery',
 'higher',
 'internet',
 'romantic']
```



그다음, 선형회귀 모델을 구축하기 위하여 G3 컬럼을 제외한 컬럼들을 따로 features 리스트에 추가하였다. 선형회귀 모델은 종속변수 Y 를 독립변수 X 들 간의 선형결합으로 설명하고자 하는 모델을 의미한다. 이 데이터의 X 는 데이터의 features 컬럼의 values 이고, y 는 데이터의 G3 컬럼의 values 이다. Train:validate:test data 의 비율을 49:21:30 으로 설정하여 선형회귀 모델을 구축하였다. 선형회귀 모델의 예측 결과를 알아보기 위하여 MAE(Mean Absolute Error)와 MSE(Mean Squared Error)를 측정하였다. 검증 데이터에 대한 모델의 성능을 측정한 결과, MAE 는 1.63, MSE 는 7.13, R^2 는 0.77 이라는 값을 보유하고 있었다. 테스트 데이터에 대한 모델의 성능을 측정한 결과, MAE 는 1.19, MSE 는 3.05, R^2 는 0.86 이라는 값을 보유한 것을 알 수 있었다.

검증 데이터 성능 측정 결과

```
# Make predictions on the validation set
y_val_pred = model.predict(X_val)

# Evaluate the model on the validation set
mse_val = mean_squared_error(y_val, y_val_pred)
mae_val = mean_absolute_error(y_val, y_val_pred)
r2_val = r2_score(y_val, y_val_pred)

print(f"Mean Squared Error on Validation Set: {mse_val}")
print(f"Mean Absolute Error on Validation Set: {mae_val}")
print(f"R^2 Score on Validation Set: {r2_val}")
```

```
Mean Squared Error on Validation Set: 7.139595979667572
Mean Absolute Error on Validation Set: 1.6307593655873494
R^2 Score on Validation Set: 0.7675274766796651
```

테스트 데이터 성능 측정 결과

```
# Make predictions on the test set
y_test_pred = model.predict(X_test)

# Evaluate the model on the test set
mse_test = mean_squared_error(y_test, y_test_pred)
mae_test = mean_absolute_error(y_test, y_test_pred)
r2_test = r2_score(y_test, y_test_pred)

print(f"Mean Squared Error on Test Set: {mse_test}")
print(f"Mean Absolute Error on Validation Set: {mae_test}")
print(f"R^2 Score on Test Set: {r2_test}")
```

```
Mean Squared Error on Test Set: 3.0500000284777746
Mean Absolute Error on Validation Set: 1.1910129123263888
R^2 Score on Test Set: 0.8581955143710422
```

회귀모델에는 그 모델이 잘 학습했는지 확인하기 위한 평가 지표들이 4 가지 있다. MAE, MSE, R-MSE, R-Squared 가 그 지표들이다. 그 중에서 나는 MAE 와 MSE 를 위 모델의 성능을 측정하기 위해 사용하였다. MAE(Mean Absolute Error)는 모델의 예측값과 실제값의 차이의 절댓값의 평균을 나타낸다. MAE 는 절대값을 취하기 때문에 해석에

용이하며 가장 직관적으로 파악할 수 있는 지표이지만, 모델이 실제보다 낮은 값으로 예측하는지(Underestimate) 실제보다 높은 값으로 예측하는지(Overestimate) 알 수 없다는 특징이 있다. MSE(Mean Squared Error)는 제곱을 하기 때문에 MAE 와는 다르게 모델의 예측값과 실제값의 차이의 제곱의 합을 계산한다. MSE 는 제곱을 하기에 이상치(outlier)에 더 민감하다는 특징이 있다.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \qquad MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

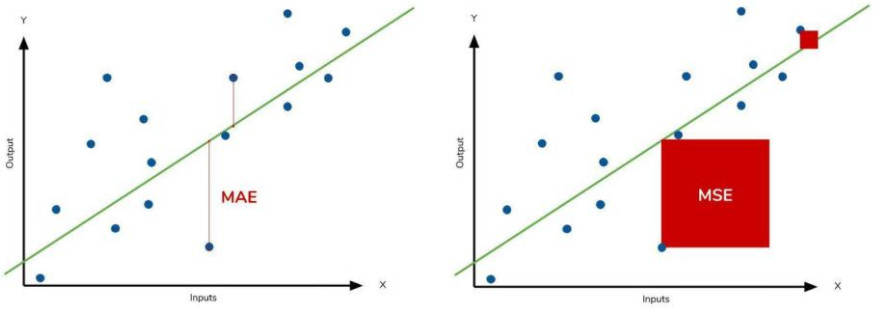
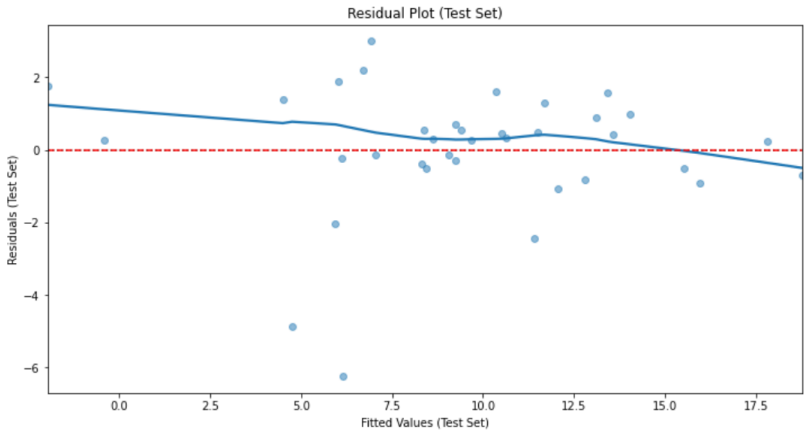
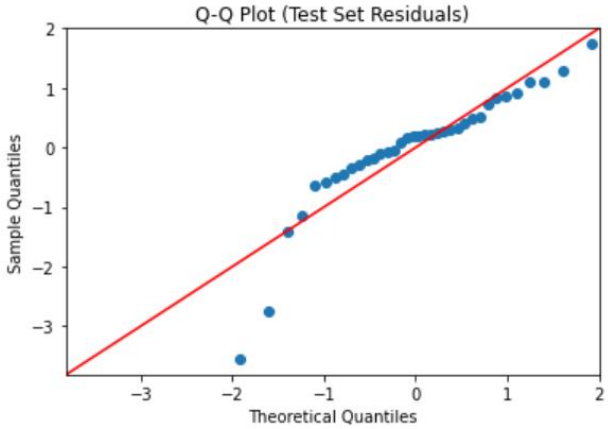


그림 2. MAE 와 MSE

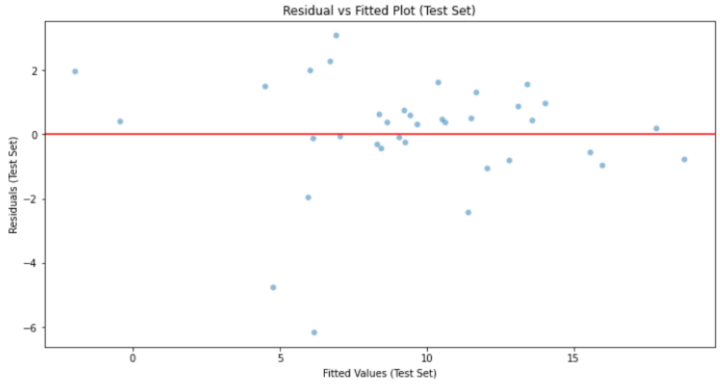
또한, 선형회귀 모델의 분석을 가시화하기 위하여 Residual Plot, Q-Q Plot, Residual vs Fitted Plot 을 도출했다. 첫 번째로, Residual Plot 은 회귀 분석에서 예측값과 실제값 간의 차이(=잔차)를 시각화한 것이다. 간단한 선형 회귀에서는 잔차를 각 데이터 포인트와 회귀선 간의 수직 거리로 계산하는데, 잔차 플롯은 이러한 잔차를 독립 변수 또는 예측값에 대해 그린 그래프이며 데이터 포인트가 모델로부터 얼마나 빗나가 있는지를 보여준다. 이상적으로 잔차는 y=0 인 수평선 주변에 무작위로 흩어져 있어야 한다[무작위 분포]. 잔차의 퍼짐은 독립 변수의 모든 수준에서 거의 일정해야 한다[등분산성]. 수평선에서 멀리 떨어진 포인트를 확인하고, 잔차의 이상값은 관측치나 모델에서의 오류를 나타낼 수 있다[이상값]. 잔차 플롯에서 선명한 곡률이나 굽힘이 있다면, 이는 모델이 적절하지 않을 수 있다는 신호이다[선형성]. 잔차의 정규성은 바로 잔차 플롯에서 확인되지는 않지만, 잔차의 히스토그램이나 Q-Q 플롯을 통해 확인할 수 있다[정규성]. 검증 데이터와 테스트 데이터에 대해 잔차 플롯을 그린 결과, 잔차는 수평선 주변에 무작위로 흩어져 있었고 등분산성과 선형성, 정규성을 잘 지키고 있음을 확인할 수 있었다.



두 번째로, Q-Q Plot(Quantile-Quantile Plot)은 주어진 데이터의 분포가 정규분포에 얼마나 가까운지를 시각적으로 평가하는 데 사용되는 플롯이다. 이는 정규분포의 분위수(Quantile)와 주어진 데이터의 분위수를 비교하여 두 분포 간 유사성을 확인한다. 정렬된 데이터의 분위수를 계산하고, 각 분위수에 해당하는 정규분포의 분위수를 계산하여 산점도로 나타낸다. 데이터의 산점도가 직선에서 벗어날수록 정규분포에 어긋나는 것을 의미하며, 산점도의 끝부분이 뾰족할수록 꼬리가 두꺼운 분포임을 나타낼 수 있다. 그리고 데이터가 정규 분포에 가까울수록 중간 부분이 평평하게 나타나며, 이 플롯을 통해 데이터가 얼마나 정규분포를 잘 따르는지 대칭성을 확인할 수 있다. 검증 데이터와 테스트 데이터에 대해 Q-Q 플롯을 그린 결과, 대부분의 데이터가 직선 근처에 있었으며 검증 데이터보다 테스트 데이터데 대한 모델 성능이 좋았음을 확인할 수 있었다.



세 번째로, Residual vs Fitted Plot 은 회귀분석에서 모델의 적합성을 시각적으로 평가하기 위해 사용된다. 이 플롯은 예측값과 잔차 간의 관계를 나타내어 모델에서 발생하는 패턴이나 비선형성을 확인하는 데 도움을 준다. 잔차가 예측값에 대해 무작위로 퍼져 있어야 하며, 잔차의 평균이 대략 0 의 값을 가져야 한다. 수평선 주변에 잔차들이 고르게 분포하면 모델이 평균적으로 일관된 예측을 하고 있다는 것을 나타낸다. 그리고 잔차들이 예측값 주변에 일정하게 퍼져 있어야 하며 즉, 잔차의 분산이 예측값에 따라 크게 변하지 않아야 한다. 검증 데이터와 테스트 데이터에 대해 위 플롯을 그린 결과, 데이터가 무작위로 퍼져 있었으며 대부분이 수평선 근처에 존재하고 있었다.



그 후, 모델에 대해 선형성은 유지한 채, Shrinkage Model(축소모델)을 활용하여 예측 정확도를 향상시키고자 하였다. 위 데이터에 대해 Lasso, Ridge, Elastic net 모델을 적용하였다. 검증 데이터에 대해 Lasso 모델의 MAE 는 1.52, MSE 는 7.23, R^2 는 0.77 의 값을, Ridge 모델의 MAE 는 1,63, MSE 는 7.13, R^2 는 0.77 의 값을, Elastic net 모델의 MAE 는 1.51, MSE 는 6.99, R^2 는 0.77 의 값을 가지는 것을 확인하였다. 테스트 데이터에 대해 Lasso 모델의 MAE 는 2.71, MSE 는 2.71, R^2 는 0.87 의 값을, Ridge 모델의 MAE 는 1.19, MSE 는 1.19, R^2 는 0.86 의 값을, Elastic net 모델의 MAE 는 1.1, MSE 는 1.1, R^2 는 0.87 의 값을 가지는 것을 확인하였다. Shrinkage model 을 활용함으로써 데이터에 대한 예측 정확도가 향상되었음을 직접 확인해 볼 수 있었다. 데이터와 모델에 대해 총 3 가지 알고리즘을 적용하고 예측 성능을 비교하였는데, 그 과정에서 각 알고리즘의 장단점도 확인할 수 있었다. 이 점은 뒤에서 로지스틱 회귀와 공통으로 다룰 내용이기에 넘어가겠다.

Lasso Regression:
Mean Squared Error on Validation Set: 7.236331708211757
Mean Absolute Error on Validation Set: 1.520381698967571
R^2 Score on Validation Set: 0.764377662744263

Ridge Regression:
Mean Squared Error on Validation Set: 7.136912974774876
Mean Absolute Error on Validation Set: 1.6303815139559505
R^2 Score on Validation Set: 0.7676148380540709

Elastic Net Regression:
Mean Squared Error on Validation Set: 6.99578351808165
Mean Absolute Error on Validation Set: 1.5142032858069643
R^2 Score on Validation Set: 0.7722101570337073

Lasso Regression:
Mean Squared Error on Test Set: 2.714262961473256
Mean Absolute Error on Test Set: 2.714262961473256
R^2 Score on Test Set: 0.873805029665674

Ridge Regression:
Mean Squared Error on Test Set: 1.1906832796588616
Mean Absolute Error on Test Set: 1.1906832796588616
R^2 Score on Test Set: 0.85818495738555

Elastic Net Regression:
Mean Squared Error on Test Set: 1.1026597750925318
Mean Absolute Error on Test Set: 1.1026597750925318
R^2 Score on Test Set: 0.8701507207694

$$minimize \sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 + \lambda\sum_{j=1}^{p}|\beta_j|$$

Lasso 수식

$$minimize \sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 + \lambda\sum_{j=1}^{p}\beta_j^2$$

Ridge 수식

$$minimize \sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 + \lambda_1\sum_{j=1}^{p}\beta_j^2 + \lambda_2\sum_{j=1}^{p}|\beta_j|$$

ElasticNet 수식

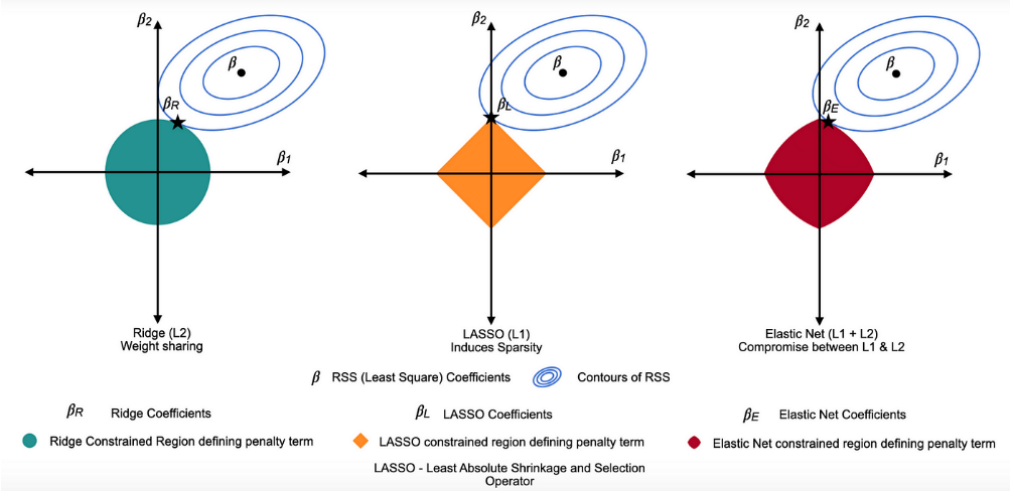
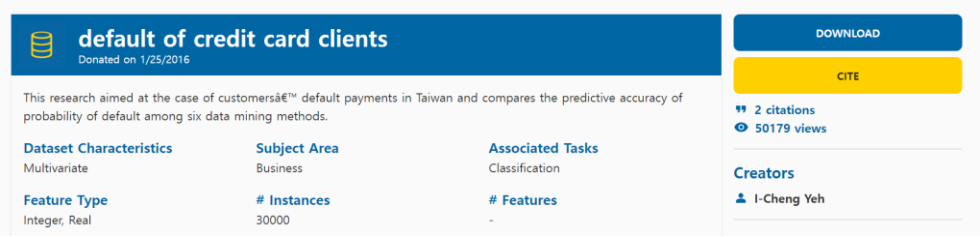


그림 3. 축소 모델

2-2. 로지스틱 회귀 분석 - Default of Credit Card Clients

UCI Data Repository 사이트에서 로지스틱 회귀(Logistic Regression)에 적합한 데이터를 선정하기 위하여 Task 를 'Regression'으로, Attribute 와 Instance 의 개수를 '10 개에서 1.000 개 사이'라는 검색 조건을 설정하여 검색하였다. 검색하여 나온 데이터 중, 'Default of Credit Card Clients' 데이터를 사용하기로 하였다.



필요한 라이브러리를 import 하고, 데이터를 불러와 데이터의 첫 5 행과 마지막 2 행을 확인하였다. 본 데이터는 30,000 개의 행과 25 개의 열로 이루어져 있으며, 열 데이터가 수치형으로만 이루어져 있었고, 수치형 데이터의 통계량을 확인할 수 있었다. 그 후, 데이터의 고유값 수를 파악하였고 분석에 필요하지 않은 한 개의 열을 삭제하였다. 그리고 35 개의 중복되는 열을 삭제하였고, 결측치가 없다는 것을 확인하였다.

```
df.tail(2)
```

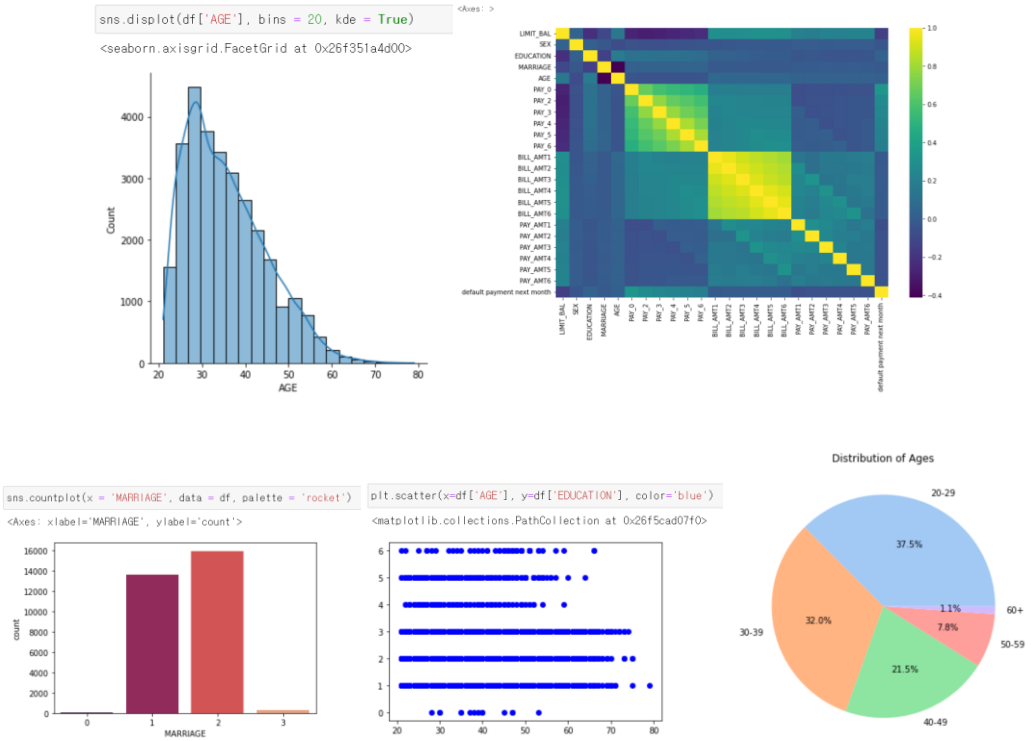
	ID	LIMIT_BAL	SEX	EDUCATION	MARRIAGE	AGE	PAY_0	PAY_2	PAY_3	PAY_4	...	BILL_AMT4	BILL_AMT5	BILL_AMT6	PAY_AMT1	PAY_AMT
29998	29999	80000	1	3	1	41	1	-1	0	0	...	52774	11855	48944	85900	340
29999	30000	50000	1	2	1	46	0	0	0	0	...	36535	32428	15313	2078	180

2 rows × 25 columns

분포 플롯(distribution plot)은 데이터가 정규화되어 있는지 살펴보기 위해 사용된다. 데이터의 AGE 컬럼과 BILL_AMT4, Pay_AMT6 컬럼을 대상으로 displot 을 그린 결과, 데이터가 정규화되어 있지 않다는 것이 명확하다는 걸 알 수 있었다. 이 값들은 정확성을 위해 scaled 되어야 할 필요가 있다. 이 점은 추후에 다루겠다. 그 후, 데이터셋의 변수 간 관계를 살펴보는 상관관계 히트맵을 시각화하여 살펴보았다.

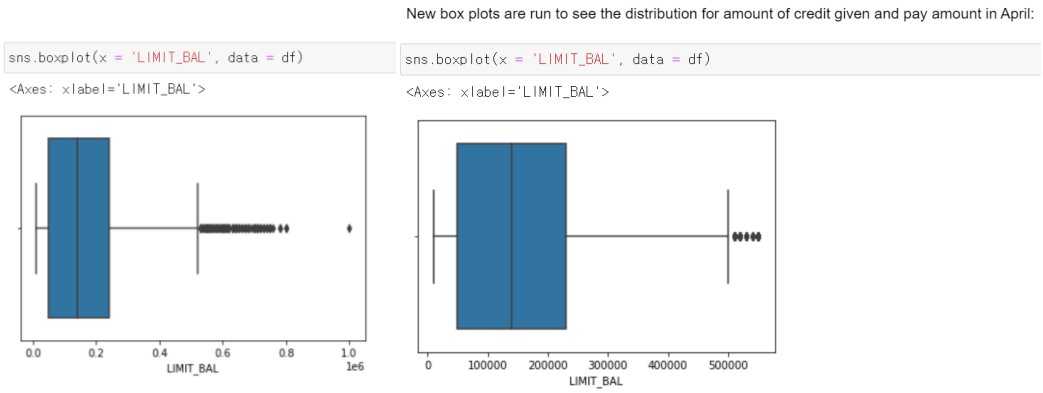
데이터의 내재적인 특징을 가시화하기 위하여 강의시간에 배운 bar plot, count plot, scatter plot, pie chart, histogram, correlation matrix 를 사용하여 데이터 특징을

인지할 수 있도록 하였다. 성별, 교육, 결혼 상태 변수에 대한 count plot 을 그려 분포를 확인하였다. 성별과 결혼 간의 bar plot, 나이와 한도 잔액의 bar plot, 나이와 교육의 scatter plot, 결혼과 교육의 scatter plot 을 확인하였다. 교육에 대한 분포를 보기 위한 pie chart, 나이 분포의 pie chart, 나이 histogram, 나이대의 histogram 을 그려보았다.



위 분포 플롯들로부터, 데이터에 outlier 가 있을 수 있음을 알 수 있었다. Outlier 가 있는지 확인해 보기 위해 box plot 을 그려 확인하였다. 데이터의 한도 잔액 컬럼과 4 월 지불 금액 컬럼의 박스 플롯을 그려보았다. 그 결과 0.6(600,000)을 초과하는 크레딧 금액과 50,000 을 초과하는 4 월 지불 금액에 대해 상당히 많은 outlier 가 존재하고 있음을 확인할 수 있다. 이 케이스는 부유한 개인이나 사업 목적으로 인한 것이며, 발생하는 경우가 거의 없다. 이러한 이상치 중 일부를 제거하기 위해 한도 잔액을 550,000 으로 제한하고, 4 월 지불 금액을 50,000 으로 제한하는 조건을 설정하여 데이터(df)를 수정하였다. 수정된 데이터는 29,327 개의 행이 있었으며, 원래 데이터의

2.24%가 제거된 것임을 알 수 있다. 이에 대해 동일한 컬럼에 대해 box plot 을 그려 확인해 본 결과, 전보다 이상치가 많이 제거되었음을 확인하였다.



모델을 구축하기 전, 성별/교육/결혼 컬럼의 값에 원-핫 인코딩(One-Hot Encoding)을 적용하고, default 컬럼을 제외한 데이터의 모든 변수를 StandardScaler 를 통해 scaled 되도록 하였다. 스케일된 데이터를 데이터프레임으로 변환한 df_feat 을 구축하였고, features 는 df_feat 의 column 이다. 이 데이터의 X 는 데이터(df_feat)의 features 컬럼의 values 이고, y 는 데이터의 default payment next month 컬럼의 values 이다. 최종 데이터(df_feat)의 로지스틱 회귀 모델을 구축하기 위하여 train:validate:test data 의 비율을 49:21:30 으로 설정하였다. 검증 데이터와 테스트 데이터에 대해 각각 confusion matrix 를 만들고, classification report 를 통해 precision, recall, f1-score, support, accuracy, macro avg, weighted avg 를 구하였다. 검증 데이터와 테스트 데이터에 대한 로지스틱 회귀 모델의 정확도는 각각 0.805, 0.806 이었다. 그리고 로지스틱 회귀 모델에 대한 accuracy 는 0.806, cross-validation accuracy 는 0.81, precision 은 0.7, recall 은 0.23, F1 score 는 0.35 였다.

검증 데이터

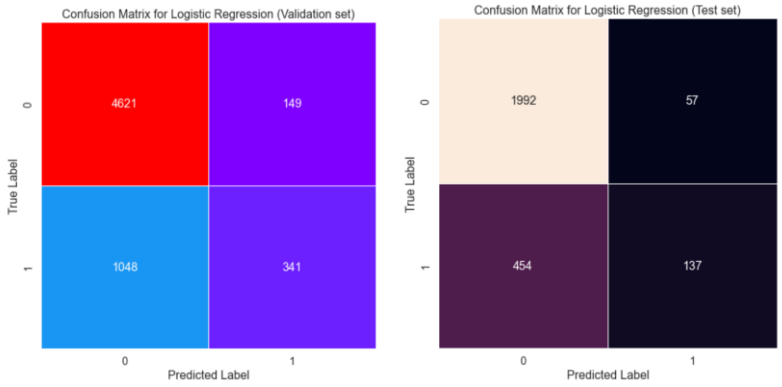
```
In [99]: print(classification_report(y_val, y_val_pred))
```

	precision	recall	f1-score	support
0	0.82	0.97	0.89	4770
1	0.70	0.25	0.36	1389
accuracy			0.81	6159
macro avg	0.76	0.61	0.62	6159
weighted avg	0.79	0.81	0.77	6159

테스트 데이터

```
In [59]: print(classification_report(y_test, predictions))
```

	precision	recall	f1-score	support
0	0.81	0.97	0.89	2049
1	0.71	0.23	0.35	591
accuracy			0.81	2640
macro avg	0.76	0.60	0.62	2640
weighted avg	0.79	0.81	0.77	2640



Classification 을 할 수 있는 머신러닝(ML) 알고리즘을 작성하였다면, 해당 모델이 얼마나 잘 작동하는지 통계적으로 확인해보아야 한다. 그 척도로 accuracy, precision, recall, f1 score 가 있다. Accuracy(정확도)는 올바르게 예측된 데이터의 수를 전체 데이터의 수로 나누는 것이다. Recall(재현율)은 실제로 참(True)인 데이터를 모델이 참이라고 인식한 데이터의 수이다. Precision(정밀도)은 모델이 참으로 예측한 데이터 중 실제로 참인 데이터의 수이다. 여기서 정밀도와 재현율은 서로 trade-off 되는 관계가 있다. F1 score 는 precision 과 recall 의 조화평균으로, 둘을 조합하여 하나의 통계치를 반환하는 지표이다. Precision 과 recall 이 0 에 가까울수록 f1 score 도 동일하게 낮은 값을 갖도록 하기 위해 일반적인 평균이 아닌 조화 평균을 계산한다. 이 지표들 중 어떤 것을 사용해야 할지는 전적으로 모델의 맥락에 달려 있다. ML 로 해결하려는 문제가 무엇인지를 정확하게 알면 알수록 위 지표들 중 어떤 것을 사용해야 할지를 결정할 수 있다. 그리고 confusion matrix 는 훈련을 통한 예측 성능을 측정하기 위해 예측 값과 실제 값을 비교하기 위한 표이다. 아래의 표에서 TP 와 TN 은 실제 값을 맞게 예측한 부분이며, FP 와 FN 은 실제 값과 다르게 예측한 부분을 나타낸다.

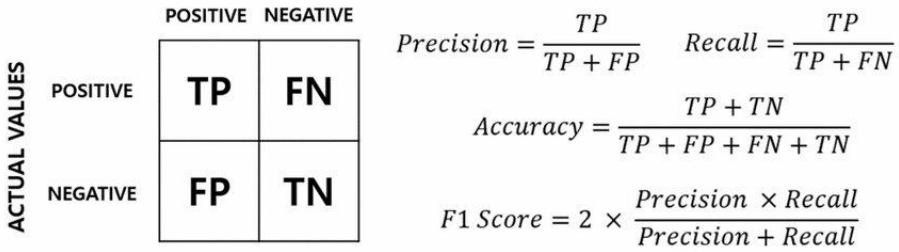


그림 4. Precision, Recall, Accuracy, F1 score 4 가지 평가 척도

그 후, Lasso, Ridge, Elastic net 총 3 가지의 축소 모델 알고리즘을 적용하고 예측 성능을 비교하였다. Lasso 를 적용한 결과, 검증 데이터에 대한 accuracy 는 0.8, precision 은 0.7, recall 은 0.2, F1 score 는 0.4 이었다. 테스트 데이터에 대한 accuracy 는 0.8, precision 은 0.7, recall 은 0.2, F1 score 는 0.3 이었다. Ridge 를 적용한 결과, 검증 데이터에 대한 accuracy 는 0.8, precision 은 0.7, recall 은 0.2, F1 score 는 0.3 이었다. 테스트 데이터에 대한 accuracy 는 0.8, precision 은 0.7, recall 은 0.2, F1 score 는 0.35 였다. Elastic net 를 적용한 결과, 검증 데이터에 대한 accuracy 는 0.8, precision 은 0.7, recall 은 0.2, F1 score 는 0.4 이었다. 테스트 데이터에 대한 accuracy 는 0.8, precision 은 0.7, recall 은 0.2, F1 score 는 0.34 였다.

```
Validation Set Evaluation - Logistic Regression with Lasso Regularization

Accuracy: 0.8057
Precision: 0.6959
Recall: 0.2455
F1 Score: 0.3630


Test Set Evaluation - Logistic Regression with Lasso Regularization

Accuracy: 0.8064
Precision: 0.7062
Recall: 0.2318
F1 Score: 0.3490


Validation Set Evaluation - Logistic Regression with Ridge Regularization

Accuracy: 0.8057
Precision: 0.6959
Recall: 0.2455
F1 Score: 0.3630


Validation Set Evaluation - Logistic Regression with Elastic Net Regularization

Accuracy: 0.8058
Precision: 0.6965
Recall: 0.2462
F1 Score: 0.3638
```

2.6 공통

회귀계수 축소법의 기본적인 공통 원리는 SSE 에 Penalty 를 더하여 함께 축소시키는 것을 목표로 한다. 기본적으로 다중선형회귀와 유사하나, 계수 축소법에서는 잔차에다가 회귀계수를 최소화하는 과정을 거친다. 회귀계수의 축소법으로는 Lasso, Ridge, Elastic net 총 3 가지가 있다. Lasso 는 f(b)에 회귀계수의 절댓값의 합을 대입한다. Lasso 의 beta(b)는 한 번에 구할 수 없으며, 라그랑지안(Lagrangian) 함수를 통해 최적화를 진행할 필요가 있다. Ridge 는 f(b)에 회귀계수의 제곱의 합을 대입하며, beta(b)는 결과적으로 라그랑지안 미분을 통해 구해진다. 다중공선성은 X 들 간의 강한 선형관계가 있을 때 발생하며, 이 경우 X 의 역행렬을 구할 수 없기에 Ridge 는 X 를 구할 수 있도록 강제로 작은 값을 대각행렬에 추가한다.

유사해 보이는 위 두 회귀계수 축소법은 두 식의 람다(lambda) 값이 모두 parameter 로, 크면 클수록 보다 많은 회귀계수를 0 으로 수렴시킨다. 두 방법 모두 SSE 를 희생해 계수를 축소하는 방법이다. 그리고 적절한 람다 값은 람다 값을 변형시키며 MSE 가 최소일 때의 람다를 탐색해 찾아내야 한다는 점에서 두 회귀모델의 공통점은 바로 “Penalized Regression(벌점 회귀) 혹은 Regularized Regression(규제화 회귀)”이다. 둘 간의 큰 차이는 Ridge 는 계수를 0 에 근사하도록 축소하나, Lasso 는 계수를 완벽하게 0 으로 축소시킨다는 점이다. 그렇기에 Ridge 의 경우 입력변수가 전반적으로 비슷한 수준으로 출력변수에 영향을 미치는 경우 사용되고, Lasso 의 경우 출력변수에 미치는 입력변수의 영향력 편차가 큰 경우에 사용된다.

Elastic net 모델은 Lasso 와 Ridge 의 하이브리드(hybrid) 모델이며, lambda_1, lambda_2 는 각각 Ridge 와 Lasso 의 속성에 대한 강도를 조절한다. 이를 통해 Ridge 의 정규화 속성과 Lasso 의 변수 축소 속성을 둘 다 갖는 모델이다. 상관관계가 있는 다수의 변수 중 하나를 무작위 선택하여 계수를 축소하는 Lasso 와 달리, Elastic net 모델은 상관성이 높은 다수의 변수를 모두 선택하거나 제거한다. 이러한 방식을 통해 group effect 를 유도하여, 다수의 변수 간에 상관관계가 존재할 때 유용하다. 보통은 성능이 Ridge 와 Elastic net 이 좋다고 하나, 이는 절대적이지 않다. 그렇기에 변수 선택법이든, 계수 축소법이든 모형을 단순화시킬 때 여러 방면으로 확인해 보는 작업이 필요하며, 이는

강의 시간에 교수님이 가르쳐주신 “No Free Lunch Theroem”과 통하는 속성이 있다고 생각한다.

Lasso, Ridge, Elastic net 모델 분석에서는 다중공선성(Multicollinearity) 문제와 과적합(Overfitting) 문제를 방지하기 위하여 아래와 같은 정규화 방식이 적용되었다. Ridge 회귀 모델은 L2 정규화, Lasso 모델은 L1 정규화, Elasticnet 모델은 Ridge 의 L2 와 Lasso 의 L1 정규화를 혼합한 모델이다. 여기서 ‘다중공선성(Multi-Collinearity)’ 이란 독립변수 X 들 간의 강한 상관관계가 나타나서 독립변수들이 독립적이지 않게 되는 문제가 발생하는 현상을 의미한다. 이 경우 coefficient 의 추정치가 부정확해지고, standard error 의 값이 커지게 된다.

Lasso 는 변수를 선택하여 일부 계수를 정확하게 0 으로 만들어 해당 변수를 모델에서 완전히 제외할 수 있고[변수 선택], 모델이 어떤 변수에 민감한지 해석하기 쉽다는 장점이 있다. 그러나 데이터가 많은 경우 모든 변수를 제외하는 Over-Regularization 문제가 존재하고, 데이터에 대한 작은 변화에도 모델이 크게 변할 수 있다는 stability(안정성) 문제가 존재한다.

Ridge 는 다중공선성 문제에 효과적으로 대처할 수 있으며, 변수를 0 으로 축소하지 않고 상관성이 높은 변수 간에 가중치를 유지하며 Lasso 에 비해 안정적인 결과를 제공한다는 장점이 있다. 그러나 Ridge 는 변수를 0 으로 축소하지 않기에 모든 변수를 유지하게 되고, 가중치가 0 이 되지 않기 때문에 해석이 어려울 수 있다는 단점이 있다.

Elastic net 은 L1(Lasso)와 L2(Ridge)의 규제를 조절할 수 있어 두 모델의 장점을 혼합하고, L1 규제로 인해 일부 변수를 선택할 수 있는 특성이 있다. 그러나 L1 비율과 L2 비율을 조절해야 하므로 두 개의 하이퍼파라미터를 조정해야 하며, L1 규제로 인해 해석이 어려울 수 있다는 단점이 존재한다.

종합적으로, Lasso 모델은 변수 선택이 중요하거나 해석이 필요한 경우에 유용한 모델이다. Ridge 모델은 다중공선성 문제를 해결하고자 할 때 유용하며, 변수 간 상관성을 유지하면서 모델을 안정화할 수 있다. Elastic net 모델은 L1 규제와 L2 규제의 혼합으로 두 규제의 장점을 결합하고자 할 때 유용한 모델이다.

Python 코드 구글 드라이브 공유 링크

아래 URL 로 웹 들어가시면 코드를 보실 수 있습니다! 감사합니다.

- 정보융합학부 2021204051 우지윤

1. 선형회귀

https://drive.google.com/file/d/10z0ORCj2RQQ1ldHzWrTFZycg3ab9W6QR/view?usp=sharing

2. 로지스틱 회귀

https://drive.google.com/file/d/11144714Rovp7GM_vwIh6Xjg0alkFZtFc/view?usp=sharing