금융과인공지능 2차과제

2021510010 이상민

1.

1-1) 아래 <표 1>과 <표 2>는 Valid Set을 사용한 특성 값을 나타낸 표이다.

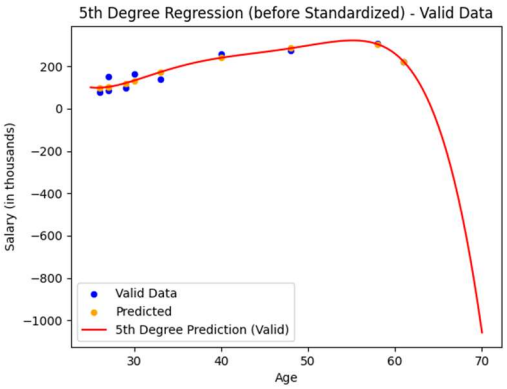
	X1	X2	X3	X4	X5
0	30	900	27000	810000	24300000
1	26	676	17576	456976	11881376
2	58	3364	195112	11316496	656356768
3	29	841	24389	707281	20511149
4	40	1600	64000	2560000	102400000
5	27	729	19683	531441	14348907
6	33	1089	35937	1185921	39135393
7	61	3721	226981	13845841	844596301
8	27	729	19683	531441	14348907
9	48	2304	110592	5308416	254803968
Mean	37.9	1595.3	74095.3	3725381	198268276.9
Std	13.287	1145.231	77969.77	4933503	303598889.3

<표 1. Z-Score 표준화 전 검증셋의 특성 값>

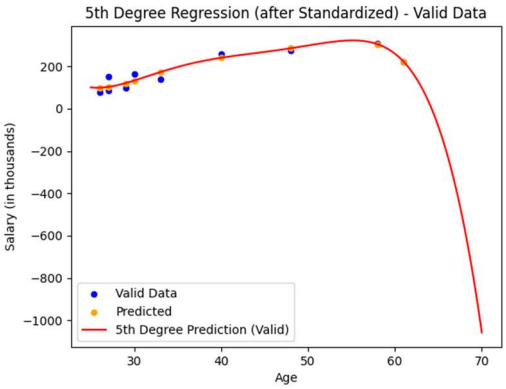
	X1	X2	X3	X4	X5
0	-0.5946	-0.6071	-0.6040	-0.5909	-0.5730
1	-0.8956	-0.8027	-0.7249	-0.6625	-0.6139
2	1.5128	1.5444	1.5521	1.5387	1.5089
3	-0.6698	-0.6586	-0.6375	-0.6118	-0.5855
4	0.1580	0.0041	-0.1295	-0.2362	-0.3158
5	-0.8204	-0.7564	-0.6979	-0.6474	-0.6058
6	-0.3688	-0.4421	-0.4894	-0.5147	-0.5242
7	1.7385	1.8561	1.9608	2.0514	2.1289
8	-0.8204	-0.7564	-0.6979	-0.6474	-0.6058
9	0.7601	0.6188	0.4681	0.3209	0.1862
Mean	0.0000	0.0000	0.0000	0.0000	0.0000
Std	1.0000	1.0000	1.0000	1.0000	1.0000

<표 2. Z-Score 표준화 후 검증셋의 특성 값>

1-2) 아래의 <그림 1>과 <그림 2>는 표준화 전후의 다항회귀분석 결과이다.



<그림 1. 표준화 전 선형회귀분석결과>



<그림 2. 표준화 후 선형회귀분석결과>

5차 다항 회귀 모형식 (표준화 전)은 다음과 같다:

$Y = 11588.0412 - 1520.1688X^1 + 77.9076X^2 - 1.9401X^3 + 0.0237X^4 - 0.0001X^5$

표준화 전의 RMSE는 24.5943으로 계산된다.

표준화 후의 모형식은 다음과 같다:

$Y = 178.6 - 19161.9672X^1 + 84663.6696X^2 - 143504.7143X^3 + 110804.4930X^4 - 32724.5937X^5$

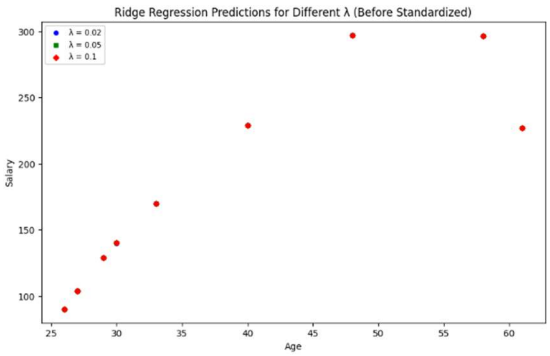
표준화 후의 RMSE도 동일하게 24.5943이다.

이 결과를 보면, 분석 결과인 RMSE는 같지만 모형식은 다르게 나타난다. 이는 정규화 과정이 변수들 간의 스케일 차이를 제거하고 균등하게 만들기 때문이다. 표준화를 하면 독립 변수들이 새로운 스케일 (평균 0, 표준편차 1)로 변환된다. 따라서, 회귀 분석에서 추정된 계수 역시 정규화 된 스케일에 맞춰서 조정된다.

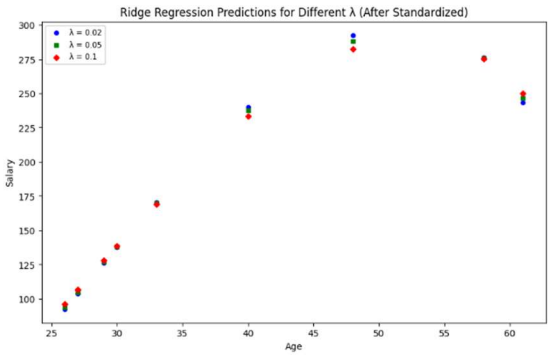
하지만, 예측 값과 RMSE는 동일하다. 이는 회귀 분석이 데이터의 선형 관계를 학습하는 과정이기 때문이다. 표준화 전후의 스케일이 다르더라도, 회귀 모델은 이를 반영하여 새로운 계수를 계산해 예측 값을 동일하게 만든다. RMSE는 실제 값과 예측 값 간의 차이를 측정하는 지표인데, 표준화 전과 후의 예측 값이 동일하기 때문에 RMSE 값도 동일하게 나온다.

결론적으로, 표준화는 변수의 분포를 변경할 뿐, 선형 관계 자체를 바꾸지 않기 때문에 회귀 모델의 예측 값과 RMSE는 변하지 않는다.

1-3) 아래의 <그림 3>, <그림 4>는 표준화 전후의 Ridge 회귀 분석 결과이다. <표 3>, <표 4>는 표준화 전후의 Ridge 모형 식의 가중치와 MSE를 나타낸다.



<그림 3. 표준화 전 Ridge 분석 결과>



<그림 4. 표준화 후 Ridge 분석 결과>

	λ	a	b1	b2	b3	b4	b5	MSE
0	0.02	-347.4003	-18.2240	4.1520	-0.1729	0.0030	0	660.2376
1	0.05	-434.3007	-7.2476	3.6109	-0.1599	0.0028	0	661.0497
2	0.10	-463.1154	-3.5532	3.4261	-0.1554	0.0028	0	661.3246

<표 3. 표준화 전 Ridge 가중치 및 MSE>

	λ	a	b1	b2	b3	b4	b5	MSE
0	0.02	178.6000	101.1917	79.5889	33.0009	-34.9933	-119.7662	745.9404
1	0.05	178.6000	110.6215	68.6418	17.6371	-39.0315	-98.5574	761.1457
2	0.1	178.6000	108.6648	61.9089	12.1350	-37.6943	-85.6615	790.1529

<표 3. 표준화 후 Ridge 가중치 및 MSE>

표준화 전 Ridge 회귀에서는 λ 값의 변화가 거의 반영되지 않는다. λ 값이 달라도 예측 값이 비슷하게 나타나는 문제가 있다. 이는 표준화되지 않은 데이터에서 특성들의 스케일이 달라, Ridge 회귀의 규제 페널티가 제대로 작동하지 않기 때문이다. 이로 인해 일부 가중치가 비정상적으로 커지거나 불안정해질 수 있다.

반면, 표준화 후 Ridge 회귀에서는 λ 값에 따른 예측 결과의 차이가 명확하게 나타난다. 표준화 과정으로 모든 특성의 스케일이 동일해지면서, Ridge 회귀의 페널티가 효과적으로 반영된다. λ 값이 작을수록 모델은 복잡해져 데이터를 더 정확하게 맞추려 하지만, 과적합의 위험이 커질 수 있다. λ 값이 커질수록 모델은 단순해지며, 과적합이 줄어들고 일반화 성능이 개선된다.

예를 들어, λ가 0.02일 때는 페널티가 약해 모델이 데이터를 최대한 맞추려 하여 복잡성이 높아진다. 이 경우, 과적합 위험이 높아질 수 있다. λ가 0.05일 때는 중간 수준의 페널티가 적용되어 모델의 복잡성을 적절히 유지하면서도 일반화 성능을 확보할 수 있다. λ가 0.1일 때는 모델이 단순해져, 데이터의 세세한 변동을 잘 반영하지 못할 수 있지만, 더 나은 일반화 성능을 보여줄 가능성이 높아진다.

결론적으로, 표준화된 Ridge 회귀는 λ 값의 변화에 따른 모델의 복잡도 조절이 잘 이루어지며, 과적합을 방지하면서 일반화 성능을 높이는 데 효과적이다.

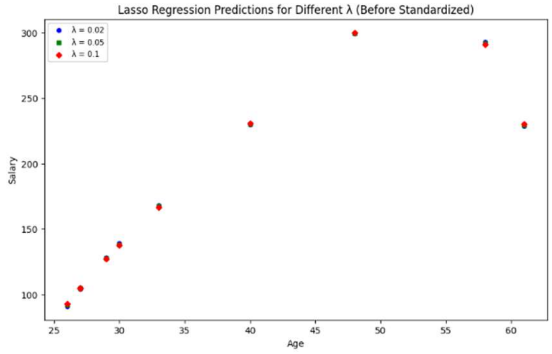
1-4) 아래의 <그림 5>, <그림 6>은 표준화 전후의 Lasso 회귀 분석 결과이다. <표 5>, <표 6>은 표준화 전후의 Lasso 모형 식의 가중치와 MSE를 나타낸다.

	λ	a	b1	b2	b3	b4	b5	MSE
0	0.02	-1292.0227	114.5852	-3.0754	0.0174	0.0006	0	672.8596
1	0.05	-1139.3795	99.6661	-2.5588	0.0102	0.0006	0	673.3503
2	0.1	-884.2766	74.7247	-1.6947	-0.0018	0.0006	0	675.1824

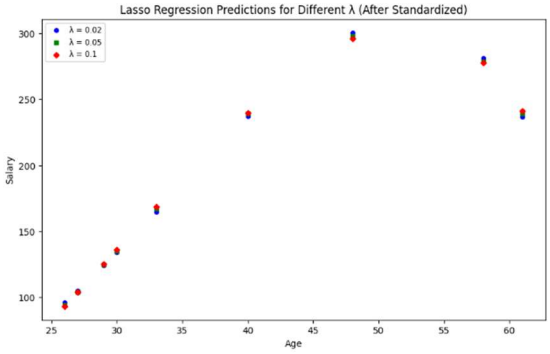
<표 5. 표준화 전 Lasso 가중치 및 MSE>

	λ	a	b1	b2	b3	b4	b5	MSE
0	0.02	178.6000	0.0000	122.2832	185.7951	0	-251.7098	714.8818
1	0.05	178.6000	0.0000	237.0009	0	0	-179.5160	724.5913
2	0.1	178.6000	43.7115	174.0969	0	0	-159.6705	731.7556

<표 6. 표준화 후 Lasso 가중치 및 MSE>



<그림 5. 표준화 전 Lasso 분석 결과>

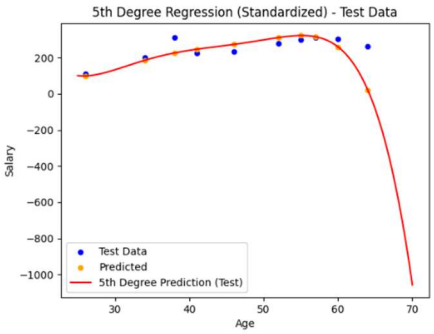


<그림 6. 표준화 후 Lasso 분석 결과>

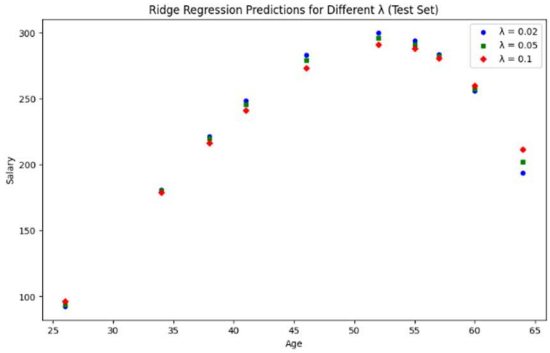
표준화 전과 표준화 후의 Lasso 회귀 분석 결과를 비교하고 있다. 표준화 전 Lasso 회귀에서는 λ 값의 변화가 거의 반영되지 않는 것을 확인할 수 있다. 이는 데이터의 각 특성이 서로 다른 스케일을 가지고 있기 때문에, 규제 페널티가 제대로 작동하지 않는 상황을 나타낸다. 이로 인해 모델이 특정 특성에 과도하게 의존하며, Lasso 회귀의 중요한 특징인 일부 계수를 0으로 만드는 기능이 제대로 작동하지 않는다. 결국 모든 특성의 계수가 비정상적으로 큰 값을 가지게 되어, 모델이 과적합될 가능성이 높아진다.

반면, 표준화 후 Lasso 회귀 분석에서는 λ 값의 변화에 따른 예측 결과의 차이가 명확하게 나타난다. 이는 표준화 과정을 통해 모든 특성의 스케일이 동일하게 되었기 때문에, Lasso 회귀의 규제 페널티가 효과적으로 작용하기 때문이다. 표준화된 데이터에서는 Lasso 회귀가 불필요한 특성의 계수를 0으로 만들고, 중요한 특성만을 선택하는 특징 선택 기능을 발휘할 수 있다. 이로 인해 과적합이 줄어들고, 일반화 성능이 개선되는 결과를 얻을 수 있다.

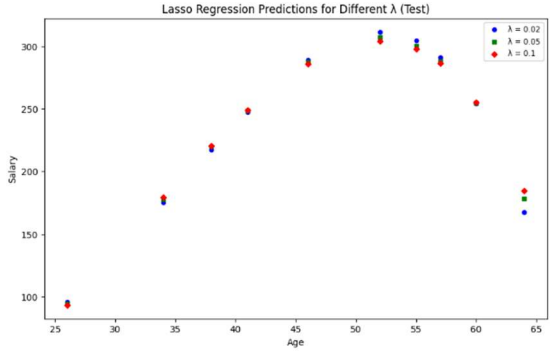
1-5) 아래의 <그림 7>, <그림 8>, <그림 9>는 Test Set에 대한 각 회귀분석 결과이다. <표 7>은 각 모형에대한 성과 비교이다.



<그림 7. Test Set의 선형회귀분석결과>



<그림 8. Test Set의 Ridge 회귀 분석 결과>



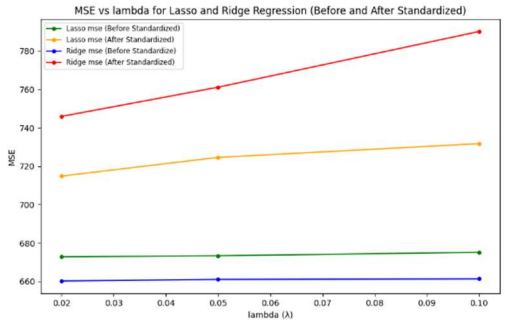
<그림 9. Test Set의 Lasso 회귀 분석 결과>

Model	Vanilla	Lasso			Ridge		
λ	-	0.02	0.05	0.1	0.02	0.05	0.1
MSE (Valid)	604.8817	714.8818	724.5913	731.7556	745.9404	761.1457	790.1529
MSE(Test)	7077.8072	2628.7355	2346.0771	2201.8218	2010.1731	1864.6012	1743.8413

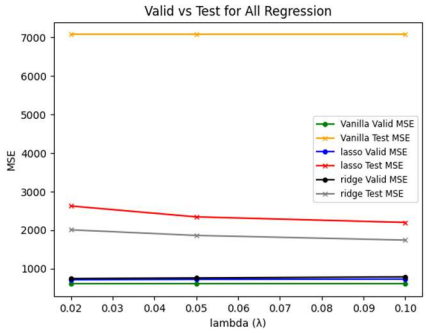
<표 7. 모형 성과 비교>

위의 데이터들을 바탕으로 각 모형에 대한 MSE 비교를 쉽게 하기 위해 표준화 전후의 Ridge와 Lasso의 MSE와 표준화 후의

valid와 test data의 각 모형의 MSE를 <그림 10>, <그림 11>로 시각화 하였다.



<그림 10. 표준화 전후의 Ridge, Lasso MSE 비교>



<그림 11. 표준화 후 모든 모형에 대한 MSE 비교>

먼저, 표준화 전 Lasso와 Ridge 회귀에서는 λ 값이 변해도 MSE 값이 거의 변하지 않는다. 이는 데이터의 특성들이 서로 다른 스케일을 가지고 있어서, L1(Lasso)이나 L2(Ridge) 정규화가 제대로 작동하지 않기 때문이다. 이로 인해 가중치가 불안정해지고, 모델이 과적합되기 쉽다.

반면, 표준화 후에는 λ 값이 커질수록 MSE가 조금씩 증가하는 것을 볼 수 있다. 표준화로 모든 특성의 스케일이 같아지면서 정규화 페널티가 잘 적용되기 때문이다. λ 값이 작을 때는 모델이 복잡해져 데이터를 더 잘 맞추려고 하지만, λ 값이 커질수록 모델이 단순해져 과적합이 줄어드는 경향을 보인다.

검증셋과 테스트셋의 MSE 값을 비교해 보면, Vanilla 모델은 검증셋에서 가장 낮은 MSE(604.88)를 보이지만, 테스트셋에서는 MSE 값이 크게 증가(7077.81)한다. 이는 Vanilla 모델이 과적합되어 새로운 데이터에서는 성능이 떨어지는 것을 의미한다.

Lasso 회귀에서는 λ 값이 커질수록 검증셋의 MSE 값은 조금씩 증가하지만, 테스트셋에서는 MSE 값이 감소하는 경향을 보인다. 이는 Lasso가 불필요한 특성의 가중치를 0으로 만들어, 중요한 특성만 남겨두기 때문이다. 이로 인해 과적합이 줄어들고 일반화 성능이 개선된다.

Ridge 회귀에서는 λ 값이 증가할수록 검증셋의 MSE는 증가하지만, 테스트셋에서는 MSE 값이 감소한다. 특히, λ 값이 0.1일 때 테스트셋에서 가장 낮은 MSE(1743.84)를 기록한다. 이는 Ridge 모델이 모든 특성의 가중치를 균형 있게 줄이며, 과적합을 잘 억제하기 때문이다.

종합적으로 보면, 검증셋에서는 Vanilla 모델이 가장 낮은 MSE 값을 보이지만, 과적합 문제로 인해 테스트셋에서는 성능이 떨어진다. 반면, Lasso와 Ridge 모델은 정규화를 통해 과적합을 방지하고, 특히 Ridge 모델이 테스트셋에서 가장 좋은 성능을 보여준다. 따라서, 일반화 성능을 고려할 때 RIDGE 모델이 가장 적합한 선택이라고 할 수 있다.

2.

2-1) 로지스틱 회귀 분석에서 디폴트(채무불이행)를 양성(1)으로 정의하고, 비디폴트(정상 상환)를 음성(0)으로 정의한 결과와, 디폴트를 음성으로 정의한 결과를 비교하였다.

먼저, 디폴트를 양성으로 정의했을 때의 절편은 -6.1808로 나타났으며, 가중치는 [0.1418, 0.0041, -0.0012, 0.0107]로 계산되었다. 반대로, 디폴트를 음성으로 정의했을 때의 절편은 6.5647로 바뀌었고, 가중치는 [-0.1395, -0.0041, 0.0011, -0.0113]로 나타났다. 이는 양성과 음성 클래스를 정의하는 방식이 반대가 되면, 절편과 가중치의 부호 또한 반대로 변하기 때문이다.

절편이 -6.1808에서 6.5647로 변한 것은 모델이 예측할 목표 클래스가 반대가 되었기 때문이다. 즉, 디폴트를 양성으로 정의할 경우, 모델은 디폴트 확률을 예측하며 이때 절편이 음수로 나타난다. 반면, 비디폴트를 양성으로 정의할 경우에는 절편이 양수가 된다. 이는 모델이 예측의 기준을 반대로 설정했기 때문이다.

가중치의 부호가 반대가 되는 것도 동일한 원리이다. 예를 들어, 디폴트를 양성으로 정의할 때 나이의 가중치가 0.1418이라면, 이는 나이가 많을수록 디폴트 가능성이 증가함을 나타낸다. 반대로, 비디폴트를 양성으로 정의했을 때 나이의 가중치가 -0.1395인 것은 나이가 많을수록 비디폴트 가능성이 줄어든다는 의미이다. 따라서, 가중치의 부호가 반대가 되는 것은 모델이 목표 클래스를 반대로 설정했기 때문이다.

하지만 양성과 음성 클래스를 어떻게 정의하더라도, 예측 확률 자체는 동일하게 유지된다. 예를 들어, 디폴트를 양성으로 정의했을 때 어떤 샘플의 예측 확률이 [0.7, 0.3]이라면, 비디폴트를 양성으로 정의했을 때는 [0.3, 0.7]로 나타난다. 이는 예측 확률이 서로 보완적인 관계에 있기 때문이다.

결론적으로, 절편과 가중치의 부호 변화는 양성과 음성 클래스를 어떻게 정의하느냐에 따른 차이일 뿐이며, 모델의 예측 확률 자체에는 영향을 미치지 않는다. 따라서, 로지스틱 회귀 모델에서 양성과 음성 클래스를 바꾸어도 모델의 예측 성능은 동일하게 유지되기 때문이다.

2-2) 아래 <표 8>, <표 9>, <표 10>은 0.25, 0.2, 0.15의 Z 값으로 계산된 혼동행렬이다. <표 11>은 디폴트 예측을 위해 각 z값으로 계산된 결과값을 비교한 표이다.

Z= 0.25	Predict No Default	Predict Default
Outcome No Default	0	17.88370521
Outcome Default	0	82.11629479

<표 8. Z = 0.25 혼동행렬>

Z = 0.20	Predict No Default	Predict Default
Outcome No Default	0	17.88370521
Outcome Default	0	82.11629479

<표 9. Z= 0.2 혼동행렬>

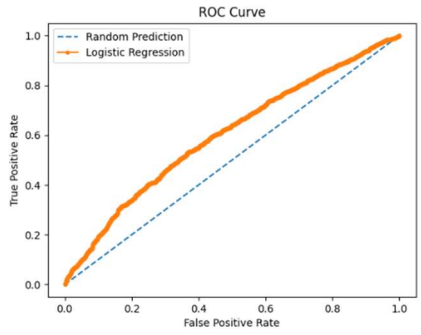
Z = 0.15	Predict No Default	Predict Default
Outcome No Default	0	17.88370521
Outcome Default	0	82.11629479

<표 10. Z = 0.15 혼동행렬>

	0.25	0.2	0.15
Accuracy	0.821163	0.821163	0.821163
True Pos Rate	1	1	1
True Neg Rate	0	0	0
False Pos Rate	1	1	1
Precision	0.821163	0.821163	0.821163
F-Score	0.901801	0.901801	0.901801

<표 11. 혼동행렬관련 비율>

2-3) <그림 12>는 디폴트 예측을 위해 계산된 ROC 그래프이다.

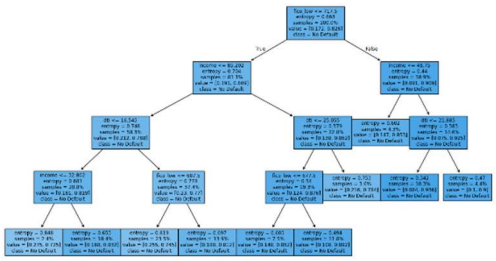


<그림 12>

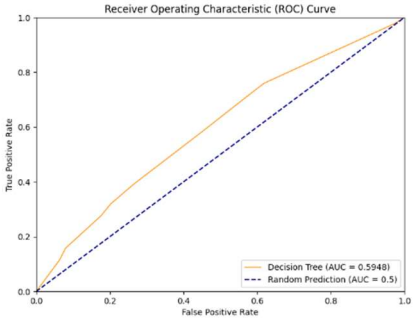
계산된 AUC는 0.6019로 여전히 0.6020과 거의 같은 값을 갖는 것을 확인할 수 있다.

3.

아래 <그림 13>은 렌딩클럽 자료를 바탕으로 제작된 Decision Tree이다. <그림 14>는 Decision Tree를 바탕으로 계산된 ROC Curve이며, <표 12>는 이때 계산된 혼동행렬이다.



<그림 13. Decision Tree>



<그림 14. ROC Curve>

Z = 0.9	Predict No Default	Predict Default
Outcome No Default	9.4320	72.6842
Outcome Default	1.1325	16.7512

<표 12. 혼동행렬>

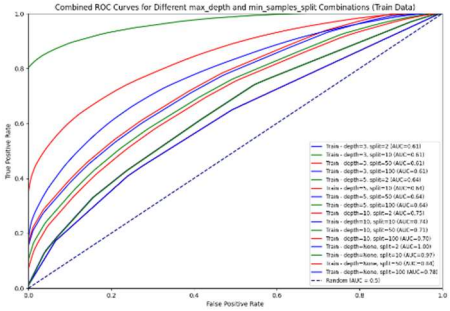
렌딩클럽 데이터를 사용하여 의사결정 트리 모델을 만들고, Z-값을 0.9로 설정했을 때의 결과를 분석해보면 다음과 같다. Z-값이 0.9라는 것은 디폴트를 예측할 확률이 90% 이상일 때만 디폴트로 예측하는 것을 의미한다. 이로 인해 모델은 대부분의 사례를 디폴트가 아니라고 판단하여, 실제 디폴트인 경우를 많이 놓쳤다. 혼동행렬에서도 모델이 많은 사례에서 False Positive(실제로는 디폴트가 아니지만 디폴트로 예측한 경우)가 많이 발생했음을 볼 수 있다.

ROC 곡선에서는 AUC 값이 0.5948로, 이는 모델의 예측 성능이 랜덤 추정(AUC = 0.5)보다는 약간 우수하지만, 충분히 좋은 성능을 보여주지는 않는다. 따라서 이 모델은 예측력이 낮으며, 개선이 필요함을 의미한다.

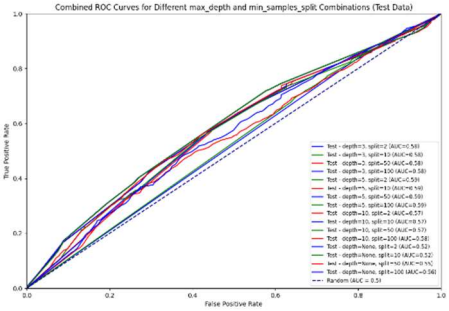
의사결정 트리 구조를 보면, DTI(부채 대비 소득 비율)가 낮고 FICO 점수가 높은 대출자를 디폴트가 발생하지 않는 고객으로 판단하는 경향이 있다. 즉, 소득 대비 부채 비율이 낮고 신용 점수가 높은 경우 대출이 승인될 가능성이 높아짐을 나타낸다.

4. 옵션문제

다음 <그림 15>, <그림 16>와 <표 13>은 트리의 최대 깊이 및 분할에 필요한 최소 샘플 수의 변경 효과를 테스트한 결과이다.



<그림 15. Train Set에 대한 테스트 결과>



<그림 16. Test Set에 대한 테스트 결과>

	최대 깊이	최소 샘플 수	train_auc	test_auc
0	5	2	0.6412	0.5925
1	5	50	0.6404	0.5924
2	5	100	0.6395	0.5919
3	5	10	0.6408	0.5917
4	3	100	0.6103	0.5829
5	3	50	0.6103	0.5829
6	3	10	0.6103	0.5829
7	3	2	0.6103	0.5829
8	10	100	0.6956	0.5808
9	10	50	0.7094	0.5720
10	10	10	0.7387	0.5698
11	10	2	0.7478	0.5663
12		100	0.7803	0.5604
13		50	0.8429	0.5468
14		10	0.9660	0.5240
15		2	1.0000	0.5187

<표 13. Train Set과 Test Set에 대한 AUC비교>

의사결정트리 모델 분석에서 최대 깊이(max_depth)와 분할에 필요한 최소 샘플 수(min_samples_split)가 모델 성능에 어떤 영향을 주는지 살펴보았다.

먼저, 트리의 최대 깊이 (max_depth)는 트리의 복잡성을 결정하는 중요한 요소이다. 깊이가 깊어질수록 모델이 데이터를 더 세부적으로 학습하게 된다. 분석 결과, max_depth 값이 증가할수록 훈련 데이터에서는 AUC 값이 상승하며, 데이터에 더 잘 맞는 경향을 보인다. 예를 들어, max_depth=10일 때 훈련 데이터의 AUC 값은 0.7478로 높지만, 테스트 데이터에서는 0.5663로 크게 떨어진다. 특히, max_depth=None (제한 없음)일 때는 훈련 데이터에서 AUC 값이 1.0으로 완벽한 예측을 보이지만, 테스트 데이터에서는 AUC 값이 0.5187로 매우 낮게 나타난다. 이는 트리의 깊이가 깊어질수록 과적합이 발생하며, 새로운 데이터에서는 성능이 저하된다는 것을 의미한다. 따라서, 최적의 깊이는 max_depth=5로 선택하는 것이 적절하다. 이 경우 훈련 데이터와 테스트 데이터 간의 AUC 차이가 적고, 일반화 성능이 안정적이기 때문이다.

다음으로, 분할에 필요한 최소 샘플 수 (min_samples_split)는 트리가 분할되기 위해 필요한 최소 샘플의 개수를 의미한다. 이 값이 작을수록 트리가 더 자주 분할되어 복잡한 모델이 만들어진다. 분석 결과, min_samples_split 값이 작아질수록 훈련 데이터의 AUC 값은 증가하지만, 테스트 데이터의 AUC 값은 감소하는 경향이 나타난다. 예를 들어, min_samples_split=2일 때 훈련 데이터의 AUC 값은 1.0으로 매우 높지만, 테스트 데이터의 AUC 값은 0.5187로 낮게 나타난다. 이는 분할 기준이 너무 낮으면 모델이 과적합되어, 새로운 데이터에서는 성능이 떨어진다는 것을 의미한다. 반대로, min_samples_split 값이 50 또는 100일 때는 훈련 데이터의 AUC 값이 다소 낮아지지만, 테스트 데이터에서는 안정적인 성능을 보인다. 특히, min_samples_split=50일 때는 과적합이 줄어들고 테스트셋에서의 AUC 값이 높아진다.

결론적으로, 최적의 하이퍼파라미터는 max_depth=5와 min_samples_split=50으로 선택하는 것이 좋다. 이 조합에서는 모델이 훈련 데이터에 과도하게 맞추지 않으면서도, 테스트 데이터에서 안정적이고 높은 성능을 보여준다. 이를 통해 일반화 성능이 개선되며, 새로운 데이터에서도 일관된 예측 결과를 기대할 수 있다.