creative commons

COMMONS DEED

저작자표시-비영리-변경금지 2.0 대한민국

이용자는 아래의 조건을 따르는 경우에 한하여 자유롭게

- 이 저작물을 복제, 배포, 전송, 전시, 공연 및 방송할 수 있습니다.

다음과 같은 조건을 따라야 합니다:

저작자표시. 귀하는 원저작자를 표시하여야 합니다.

비영리. 귀하는 이 저작물을 영리 목적으로 이용할 수 없습니다.

변경금지. 귀하는 이 저작물을 개작, 변형 또는 가공할 수 없습니다.

- 귀하는, 이 저작물의 재이용이나 배포의 경우, 이 저작물에 적용된 이용허락조건을 명확하게 나타내어야 합니다.
- 저작권자로부터 별도의 허가를 받으면 이러한 조건들은 적용되지 않습니다.

저작권법에 따른 이용자의 권리는 위의 내용에 의하여 영향을 받지 않습니다.

이것은 이용허락규약(Legal Code)을 이해하기 쉽게 요약한 것입니다.

Disclaimer

dCollection

이 학 석 사 학 위 논 문

결측치 대치를 활용한 신용데이터 분석방법에 관한 연구

2022년 1월

서울대학교 대학원
통계학과
임성현

결측치 대치를 활용한
신용데이터 분석방법에 관한 연구

지도교수 장 원 철

이 논문을 이학석사학위논문으로 제출함
2021년 10월

서울대학교 대학원
통계학과
임 성 현

임성현의 석사학위논문을 인준함
2022년 1월

위 원 장 임 요 한

부 위 원 장 장 원 철

위 원 정 성 규

국문초록

임성현
통계학과
대학원
서울대학교

결측이 발생한 데이터를 모두 제거한 후 완전한 데이터만을 활용해 분석하는 것을 listwise deletion이라고 한다. 이는 편의성 등으로 인해 빈번하게 채택되는 결측에 대한 대응법이지만, 정보의 손실과 편향된 결과를 초래할 가능성이 크다. 이에 결측 대치(imputation) 기법 활용이 listwise deletion에 비해 이점이 있는가에 대해 연구하고자 한다.

본 논문에서는 머신러닝 훈련 과정에서 완전한 데이터만을 활용하는 것과 결측을 대치한 데이터를 활용하는 것을 비교해 통계적 결측 대치가 listwise deletion보다 더 나은 것을 보이고자 한다. 이를 적절하게 논의하기 위해 결측 패턴, 머신러닝 기법, 계급 불균형 처리 등을 포함하여 내용을 전개하고, 은행의 고객 데이터를 활용해 연체 여부를 예측하는 분류 문제에서 결측 대치 기법의 효과가 있는지를 확인한다.

주요어 : 결측 패턴, 통계적 결측 대치, 머신러닝을 이용한 분류, 계급 불균형
Student Number : 2020 − 20361

Contents

1 서론 **1**

2 분석 방법 **4**

2.1 결측치 4

2.2 통계적 결측 대치 12

2.3 Oversampling 18

2.4 머신러닝 기법 22

2.5 Performance measure 24

3 사례 연구 **25**

3.1 데이터 25

3.2 분석 29

3.2.1 전처리 29

3.2.2 진행 과정 31

4 분석 결과 **34**

5 결론 **41**

Abstract **46**

Chapter 1

서론

대출을 다루는 은행과 카드사 등과 같은 기업들에 있어 고객의 대출을 승인할지 결정하는 것은 그들의 주요 과제 중 하나이다. 기업들은 명백하게 대출을 상환할 것 같은 고객들의 대출 신청만을 승인하고 싶어 할 것이다. 이에 고객의 정보를 통해 신청자의 연체 여부를 예측하는 분석이 자연스레 필요하게 된다. 기업이 보유한 신청자의 정보는 다르겠지만 일반적으로 대출액, 연 소득, 직장과 근속 연수 등을 포함한 정보로 이는 그들의 신용을 나타내는 지표로 볼 수 있다. 따라서, 본 연구에서는 이러한 자료를 신용 데이터라고 부르고 관련된 분류, 예측을 포함한 데이터 분석을 신용 데이터 분석 혹은 연체자 예측 분석으로 부를 것이다.

대부분 통계모형과 머신러닝모형은 결측이 없는(Complete, 완전한) 데이터가 필요하다. 이에 많은 통계적 분석은 데이터 수집 자체를 완전한 데이터로 하거나 결측이 있는 행을 제거하는 작업을 한 후 진행한다. 이때, 결측이 포함된 데이터에 대해 하나의 변수라도 결측이 발생하면 그 행, 즉 그 관측치의 정보를 모두 제거하는 방법을 listwise deletion이라고 한다. Listwise deletion은 편의성 등을 이유로 빈번하게 사용되는 방법이다. 하지만, 현실의 많은 데이터에는 결측치가 데이터 수집 단계 등에서 필연적으로 발생하기에 listwise deletion만을

활용한다면 고려할 수 있는 데이터의 개수가 굉장히 줄어들 것이다. 이렇게 감소하는 표본 수는 훈련 데이터의 감소로 직결되고 이는 머신러닝 알고리즘의 적합 과정에 영향을 줘 낮은 예측력의 원인이 될 가능성이 크다. 또한, Myers (2011)에 따르면 결측이 포함된 관측치를 모두 제거하는 방법으로 완전한 데이터를 형성할 때 결측에 대한 가정이 MCAR(Missing At Completely Random)이 아니라면 심한 편향이 발생한다. 결측의 가정에 대해서는 이후 장에서 자세히 논의한다.

본 연구에서는 머신러닝 기법을 사용할 때 결측치를 포함한 행을 모두 제거하지 않고 적절한 통계적 대치(Statistical imputation)를 통해 결측치를 대체하여 완전한 데이터를 만들고, 이 데이터를 통해 머신러닝 기법들을 적합 시킬 것이다. 결측을 포함한 행을 모두 제거해 완전한 데이터만을 추출한 방식으로도 머신러닝 기법들을 적합 시킨다. 각 방식에 대해 데이터 분할과 적합 등을 여러 번 반복하고 통계적인 검정을 통해 평가 데이터에 대한 AUC(Area Under the Curve)가 차이 나는지 관찰하여 예측의 관점에서 결측치 대치가 성능의 향상을 초래하는지 확인하는 연구의 방향을 잡는다.

사례 연구(Case Study)를 진행할 데이터는 연체 여부가 포함된 은행의 대출자에 대한 신용 데이터이며 머신러닝 기법을 통해 고객이 연체자인지를 예측하는 분류 문제를 다룰 것이다. 머신러닝을 이용한 분류 문제에서 중요한 이슈 중 하나가 분류하고자 하는 반응변수의 계급(class) 간 분포가 불균형한 경우이다. 다시 말해, 반응변수에서 특정 계급의 비율이 훨씬 높은 경우이다. 신용 데이터의 특성상 연체자보다는 상환한 사람의 비율이 높을 것이므로 계급의 관점에서 불균형적인 데이터일 가능성이 크다. 많은 머신러닝 기법들의 모델 훈련 과정이 언급한 계급 불균형 문제가 심하면 잘 이루어지지 않고 이에 따라 매우 낮은 예측 성능을 보이게 된다. 좋지 않은 성능을 보이는 기법에 결측치 대치를 통해 약간의 성능 향상을 보이더라도 이는 큰 의미를 갖지 않는다. 예를 들어, 적합한 머신러닝 기법을 통해 얻은 검정 AUC(Test AUC)가 0.55라고 가정하자. 이는 임의 추측(Random guess)보다 미세하게 좋은, 실효성이 굉장히 떨어지는 결과

이기에 대치를 통해 0.58의 test AUC를 보이더라도 이는 대치에 따라 의미 있는 예측력의 상승을 보였다고 하기 힘들다. 따라서, 계급 불균형에 따른 과소 적합 문제를 처리하는 방법들을 각 데이터에 적용한 후 머신러닝 기법들에 훈련한 결과를 통해 의미 있는 비교를 진행한다.

종합적으로 결측 패턴(Missing pattern), 통계적 결측 대치(Statistical Imputation), 머신러닝을 이용한 분류 문제(Machine learning for classification), 계급 불균형 문제 처리(Class imbalance handling) 등을 연구하고, 결측이 존재하는 대출지원자의 상환 여부 데이터에 대한 사례 연구를 통해 결과를 확인한다.

Chapter 2

분석 방법

이 장에서는 본 논문의 전반적인 내용을 다루기 위해 요구되는 개념들을 소개한다. 이는 데이터 소개와 앞선 장에서 언급한 결측의 가정에 대한 분류, 분류를 위한 머신러닝 기법들에 관한 소개, 계급 불균형을 다루는 방법들에 관한 소개 등을 포함한다.

2.1 결측치

Rubin (1976) 에 따르면 결측을 세 가지로 분류할 수 있고 그 분류에 관해 설명한다.

Y라는 데이터 행렬을 생각하자. 편의상 Y는 $(n \times p)$ 차원이라 하자. 이때, Y를 관측값이 존재하는 부분과 아닌 부분으로 (Y_{obs}, Y_{mis})로 분할해 표현하자. Y의 (i, j)원소가 관측이 되면 1, 아니면 0의 값을 갖는 r_{ij}을 원소로 가지는 $(n \times p)$ 행렬 R과 미지의 모수인 ξ를 고려하자. 정의된 이 notation들을 이용하여 결측 가정을 표현할 수 있다.

- MCAR(Missing At Completely Random) : 결측의 발생이 데이터와 관련 없이, 모든 데이터의 각 점에 대해 결측이 일어날 확률이 같은 패턴을

의미한다. 즉, 다음과 같다.

$$P(R|Y_{mis}, Y_{obs}, \xi) = P(R|\xi) \tag{2.1}$$

MCAR은 통계적인 검정을 통해 성립 여부를 확인할 수 있는 유일한 결측 가정이다. R, SPSS 등을 통해 Little's MCAR Test를 진행해 확인할 수 있다. 하지만, 실제 데이터가 MCAR 가정을 만족하는 경우는 거의 없고, MCAR 가정을 만족하면 listwise deletion을 해도 편향(bias)의 증가 문제가 발생하지 않는다. 본 연구에서 진행할 listwise deletion과 imputation 비교에 대한 의미가 줄어들며 실현 가능성이 작기에 MCAR 가정은 진행하지 않으며, test 역시 추가로 진행하지 않는다.

- MAR(Missing At Random) : 결측치가 관측된 값(observed variables)에 의해 설명이 된다는 가정으로써, 관측값으로 조건부를 준다면 결측이 일어날 확률이 모든 데이터에 같다는 의미이다. 다음과 같이 표현된다.

$$P(R|Y_{mis}, Y_{obs}, \xi) = P(R|Y_{mis}, \xi) \tag{2.2}$$

  실제로 Multiple Imputation 등을 활용할 때 대부분 MAR 가정에서 출발한다. Pedersen, Mikkelsen, Cronin-Fenton, Kristensen, Pham, Pedersen, and Petersen (2017)에서 많은 변수가 포함된 Multiple Imputation을 활용할 때 다른 가정에 비해 MAR이 그럴듯한 가정이라고 말하고 있다.

- MNAR(Missing Not At Random) : MCAR 도 MAR도 아니며, 결측이 관측되지 않은 값에 의존한다는 가정이다. 다루기가 까다로우며 (2.1) , (2.2)와 같이 $P(R|Y_{mis}, Y_{obs}, \xi)$ 을 특정한 식으로 표현할 수 없는 경우를 말한다. Non-ignorable이라고도 불린다. Misztal (2013)에서 MNAR인 경우, imputation이 아닌 더 복잡한 방법을 통해 결측을 다뤄야 한다는 내용을 포함하고 있다. Selection models나 pattern-mixture models를 활용해야 하고, 자세한 내용은 Allison (2001) 등을 참고하라고 언급하고 있다.

이해를 돕기 위한 예시를 van Ginkel, Linting, Rippe, and van der Voort (2020)에서 다음과 같이 언급하고 있다. 설문지에 설문하는 상황에서, MCAR은 설문자들이 우연히 질문에 응하지 않아 발생하는 결측이다. 성별에 관한 정보가 모두 관측되고, 여성이 남성보다 소득에 관련된 질문을 더 빈번하게 응답하지 않았을 때 소득에 대한 결측은 MAR로 볼 수 있다. 고소득층이 저소득층보다 소득에 관한 질문을 더 많이 응하지 않았으면 소득에 대한 결측은 MNAR일 것이다. Misztal (2013)에서 제시한 데이터 예시는 다음과 같으며, 저자가 예시를 위해 만든 데이터로써 각 변수 $x_1, ..., x_6$이 특정 의미를 가진 것은 아니다.

Table 2.1: Complete dataset

x_1	x_2	x_3	x_4	x_5	x_6	y
36	702	7	850	76.34	11	w
32	5000	7	1100	241.49	35	w
31	5000	13	1340.2	292.58	24	w
35	7710	24	2446	356.53	35	w
32	7679	4	1750	370.88	35	w
23	2800	3	560	205.5	18	w
28	1147	4	676.9	114.91	12	w
20	1300	5	1950	77.36	23	w
36	1320	93	1011.3	142.24	11	w
31	1612	22	594	93.94	23	w
46	2776	58	1034.3	272.4	12	s
41	2199	95	1752.2	154.7	18	s
23	4300	8	1450	195.31	35	s
40	3018	137	2416.2	137.08	35	s
24	4950	20	1038.9	224.83	35	s
50	1000	34	1163.4	98.13	12	s
42	4742	231	1293	215.39	35	s
32	3268	105	1940	375.15	10	s
41	3000	214	1539.1	350.55	10	s
43	3000	71	1433.7	323.28	11	s

Table 2.2: MCAR dataset

x_1	x_2	x_3	x_4	x_5	x_6	y
36	702	7	NA	76.34	11	w
32	5000	7	1100	241.49	35	w
31	5000	13	1340.2	292.58	24	w
35	7710	NA	2446	356.53	35	w
32	7679	4	1750	NA	35	w
23	NA	3	560	205.5	18	w
28	1147	NA	676.9	114.91	12	w
20	1300	5	1950	77.36	23	w
36	1320	93	1011.3	142.24	11	w
31	1612	22	594	93.94	NA	w
46	2776	58	NA	272.4	12	s
41	NA	95	1752.2	154.7	18	s
23	4300	8	1450	195.31	35	s
40	3018	137	2416.2	137.08	NA	s
24	4950	20	1038.9	224.83	35	s
50	1000	34	1163.4	98.13	12	s
42	4742	231	1293	215.39	35	s
NA	3268	105	1940	375.15	10	s
41	3000	214	1539.1	350.55	NA	s
43	3000	71	1433.7	NA	11	s

Table 2.3: MAR dataset

x_1	x_2	x_3	x_4	x_5	x_6	y
36	702	7	850	76.34	11	w
32	NA	7	1100	241.49	35	w
31	NA	13	1340.2	NA	24	w
35	7710	24	2446	356.53	35	w
NA	7679	4	1750	370.88	35	w
23	2800	3	NA	205.5	18	w
28	1147	4	676.9	114.91	12	w
20	1300	5	NA	NA	23	w
36	1320	93	1011.3	142.24	11	w
31	1612	22	594	93.94	NA	w
46	2776	58	1034.3	272.4	12	s
NA	2199	95	1752.2	154.7	18	s
23	4300	8	1450	195.31	35	s
40	3018	NA	2416.2	137.08	35	s
NA	NA	20	1038.9	224.83	35	s
50	1000	34	1163.4	98.13	12	s
42	4742	231	1293	215.39	35	s
32	3268	105	1940	375.15	10	s
41	3000	214	1539.1	350.55	10	s
43	3000	71	1433.7	323.28	11	s

Table 2.4: MNAR dataset

x_1	x_2	x_3	x_4	x_5	x_6	y
36	702	7	850	76.34	11	w
32	NA	7	1100	241.49	35	w
31	NA	13	1340.2	292.58	24	w
35	NA	24	2446	356.53	35	w
32	NA	4	1750	370.88	35	w
23	NA	3	560	205.5	18	w
28	1147	4	676.9	114.91	12	w
20	1300	5	1950	77.36	23	w
36	NA	93	1011.3	142.24	11	w
31	NA	22	594	93.94	23	w
46	2776	58	1034.3	272.4	12	s
NA	2199	95	1752.2	154.7	18	s
NA	4300	8	1450	195.31	35	s
NA	3018	137	2416.2	137.08	35	s
NA	4950	20	1038.9	224.83	35	s
50	1000	34	1163.4	98.13	12	s
42	4742	231	1293	215.39	35	s
NA	3268	105	1940	375.15	10	s
41	3000	214	1539.1	350.55	10	s
43	3000	71	1433.7	323.28	11	s

그렇지만, 통계적 검정을 통해 결측이 MAR인지 MNAR인지 구분해 낼 수는 없다. Zhu (2014), Li (2013)등에서 이를 제시하고 있고, 결측 시뮬레이션을 다루는 많은 논문은 데이터 수집 과정, 연구의 방향 등 상황에 맞는 적절한 가정을 한 채 진행한다. 이에 본 연구에서는 Lo, Siah, and Wong (2019) 등을 참고하여 MAR 가정을 진행하였다. Jaeger (2006) , Potthoff, Tudor, Pieper, and Hasselblad (2006)등에서 특정 조건 하 MAR 가정 설정에 관해 연구한 바 있다.

대치 방법의 성능을 검증하는 사례 연구를 위해서는 참값을 알고 있는 원데이터에서 임의로 결측을 발생시키는 과정이 필요하다. 결측의 발생은 R의 ampute 함수를 이용한 패키지인 missCompare 등을 활용하며, MAR 가정과 원데이터의 결측률을 고려해 결측이 포함된 데이터를 준비하였다. 더 자세한 진행 과정에 대한 논의는 3장 2절에서 다룬다.

2.2 통계적 결측 대치

본 연구에서 고려하는 대치 기법들은 결과(Output)로 결측이 없는 데이터 자체가 얻어지는 기법들로서 다음과 같다.

- 평균/최빈값

  연속형 변수에는 그 변수의 평균, 범주형 변수에는 그 변수의 최빈값(mode)을 대입하는 간단한 기법이며, 편의성으로 인해 자주 이용되는 결측 대치 기법이다.

  j번째 열에 s개가 관측되고, $p-s$개가 결측치일 때, 관측된 값 행들의 집합을 N_j라 하자. ($|N_j| = s$)식 (2.1)에서 정의한 r_{ij}를 고려하고, $r_{ij} = 0$인 y_{ij} 에 관측값의 평균을 대입한다. 식으로 표현하면 다음과 같다.

  $$\hat{y}_{ij} = \sum_{k \in N_j} \frac{y_{kj}}{s} \tag{2.3}$$

  유사하게, j번째 열이 범주형일 때는, 관측값의 최빈값을 대입한다.

- 중앙값/최빈값

  평균과 비슷하게 연속형 변수에는 중앙값을, 범주형 변수에는 최빈값을 대입하는 대치 방법이다.

- KNN

  K-nearest algorithm을 통해 결측 대치를 진행하는 방법이다. 근방(neighborhoods)을 정의할 때는 유클리디안 거리(Euclidean distance)를 사용한다. KNN imputation 기능을 제공하는 다양한 패키지가 존재하며, 본 연구에서 사용한 R 패키지 bnstruct에서는 연속형 변수는 근방의 중앙값을, 범주형 변수는 근방의 최빈값을 대입한다.

- missForest

  Stekhoven and Bühlmann (2012)에서 제시한 missForest에 대한 기법에 대한 소개는 다음과 같다.

  먼저 논리 전개를 위한 notation을 다음과 같이 제시한다. $(X_1, ..., X_p)$으로 이루어진 $n \times p$ 행렬 X 에서, 임의의 s번째 변수를 X_s라 하고 이 중 결측이 발생한 index를 $i_{mis}^{(s)} \subseteq \{1, ..., n\}$ 라 하자. $i_{obs}^{(s)} = \{1, ..., n\} \setminus i_{mis}^{(s)}$ 라 하면, 다음과 같이 X를 분할해 표현할 수 있다.

  X_s에서 결측이 발생하지 않고 관측된 값을 $y_{obs}^{(s)}$, X_s에서 결측에 해당하는 부분을 $y_{mis}^{(s)}$라 하자. 또한, s번째 변수가 아닌 변수들에 대해 $i_{obs}^{(s)} = \{1, ..., n\} \setminus i_{mis}^{(s)}$에 해당하는 $x_{obs}^{(s)}$, $i_{mis}^{(s)}$에 해당하는 $x_{mis}^{(s)}$ 로 총 네 개의 부분으로 분할할 수 있다. 명백하게, index $i_{obs}^{(s)}$와 $i_{mis}^{(s)}$는 X_s를 기준으로 할당되므로 $x_{obs}^{(s)}$에는 결측이 존재할 수 있고, $x_{mis}^{(s)}$에도 관측치가 존재한다. 이해를 돕기 위해 Misztal (2013)에서 제시한 $s = 6$인 자료를 Table 2.1로 구성하였다. Figure 2.1과 마찬가지로 자체적인 자료로써, 각 $x_1, ...x_6$는 역시 특정 의미를 지닌 변수가 아니다.

Table 2.5: missForest notation

x_1	x_2	x_3	x_4	x_5	x_6	y
36	702	7	NA	76.34	11	w
32	5000	7	1100	241.49	35	w
31	5000	13	1340.2	292.58	24	w
35	7710	NA	2446	356.53	35	w
32	7679	4	1750	NA	35	w
23	NA	3	560	205.5	18	w
28	1147	NA	676.9	114.91	12	w
20	1300	5	1950	77.36	23	w
36	1320	93	1011.3	142.24	11	w
31	1612	22	594	93.94	NA	w
46	2776	58	NA	272.4	12	s
41	NA	95	1752.2	154.7	18	s
23	4300	8	1450	195.31	35	s
40	3018	137	2416.2	137.08	NA	s
24	4950	20	1038.9	224.83	35	s
50	1000	34	1163.4	98.13	12	s
42	4742	231	1293	215.39	35	s
NA	3268	105	1940	375.15	10	s
41	3000	214	1539.1	350.55	NA	s
43	3000	71	1433.7	NA	11	s

y_{mis} x_{mis} y_{obs} x_{obs}

missForest의 진행은 다음과 같이 요약된다.

1. 결측률을 기준으로 $X_s, s = 1, ..., p$을 오름차순으로 정렬하고, 이때 열의 집합을 k라 하며, 평균 대치 등을 이용해 결측치에 대한 초기값 (initial guess)을 준다.

2. (a) 기존의 imputation을 진행하여 얻어진 X를 X_{old}^{imp}라 하자. 첫 번째 X_{old}^{imp}는 mean imputation을 진행한 X이다. 이 후 $s \in k$ 에 대해 $y_{obs}^{(s)} \sim x_{obs}^{(s)}$의 Random Forest을 적합한다.

   (b) 적합한 Random Forest와 $x_{mis}^{(s)}$ 통해 $y_{mis}^{(s)}$을 예측하고, $y_{mis}^{(s)}$의 예측치을 통해 imputed matrix을 갱신한다. 이 갱신한 X를 X_{new}^{imp}로 표기한다.

   (c) N은 연속형 변수 열의 집합, F는 범주형 변수 열의 집합이라 하자. 즉, $N \subset \{1, ..., p\}$ 이고, $F = \{1, ..., p\} \setminus N$ 이다. X_{old}^{imp}와 X_{new}^{imp}에 대해 j번째 변수를 각각 X_{old}^{j}와 X_{new}^{j}로 표기하면,
   연속형 변수의 차이는

$$\frac{\sum_{j \in N} (X_{new}^{j} - X_{old}^{j})^2}{\sum_{j \in N} (X_{new}^{j})^2} \tag{2.4}$$

   범주형 변수의 차이는

$$\frac{\sum_{j \in F} \sum_{i=1}^{n} I_{X_{new}^{j} \neq X_{old}^{j}}}{\#NA} \tag{2.5}$$

   로 정의된다. 여기서 $\#NA$는 범주형 변수에 포함된 결측치의 수이다. 위와 같이 정의된 imputed matrix와 기존의 matrix에 대한 변수간의 차이(difference between the newly imputed data matrix and the previous one)가 연속형, 범주형 변수 모두 증가하면 반복을 멈추고, 최종적인 imputed matrix X^{imp}을 생성한다.

이를 포함한 더 자세한 설명은 Stekhoven and Bühlmann (2012) 와 Misztal (2013)를 참고하면 된다.

- FAMD

Audigier, Husson, and Josse (2016) 와 Josse and Husson (2016)를 참고해 작성한 기법에 대한 소개는 다음과 같다. R의 missMDA 패키지에 변수의 종류(연속형, 이산형)에 따라 다른 imputation 방법이 구현되어 있다. 본 연구는 mixed type data를 활용할 예정이므로, FAMD(Factor Analysis for Mixed Data) 기법을 사용하며, 이에 대한 논의에 필요한 notation은 다음과 같다.

$(x_1, ..., x_{K_1})$가 연속형 변수이고 $(x_{K_1+1}, ..., x_p)$가 더미 변수(dummy variables)인 $n \times p$행렬 X를 생각하자. 처음 K_1개는 연속형 변수이고, 나머지 K_2개는 범주형 변수를 나타내는 더미 변수의 개수일 때, $q_k = k$번째 범주형 변수의 카테고리 수라 하면, $K = K_1 + K_2$에 대해 $p = K_1 + \sum_{k=K_1+1}^{K}(q_k - 1)$ 로 표현할 수 있다. $\widehat{\sigma}_{x_j}$를 연속형 변수인 x_j의 표준편차, p_j을 j번째 범주형 변수에 속하는 각 개체의 비율(proportion of individuals that takes category j)이며, $D^{-1/2}$ 는 $diag(\widehat{\sigma}^2_{x_1}, \widehat{\sigma}^2_{x_2}, ..., p_{K_1+1}, ..., p_j, ..., p_p)$ 이다. M은 각 행이 X의 각 열의 평균인 행렬이고, 식 (2.1)에서 활용한 R 행렬과 같이 결측 여부에 대한 지시행렬을 W로 표현한다.

PCA와 유사하게 FAMD는 다음과 같은 norm을 최소화하는 것으로 정의된다.

$$\|(X - M)D^{-1/2} - U\Lambda^{1/2}V^t\|^2 \tag{2.6}$$

이를 통해 FAMD는 $(X-M)D^{-1/2}$의 least square관점에서 가장 좋은 계수 $S(S < (p-K_2))$로의 근사(best low rank approximation of $(X-M)D^{-1/2}$

in the least sqaure sense)를 제공한다. 해(solution)는 $(X-M)D^{-1/2}$의 특이값 분해(SVD, Singular Value Decomposition)로부터, $\widehat{U}_{I\times S}, \widehat{\Lambda}_{S\times S}, \widehat{V}_{J\times S}$가 얻어지고, $\widehat{\Lambda}_{S\times S}$는 가장 큰 S개의 특이값 $\widehat{\lambda}_1, ..., \widehat{\lambda}_S$ 를 대각원소로 갖는 대각행렬이다.

이러한 FAMD를 통한 imputation을 iterative FAMD이라 부르며 다음과 같이 진행된다. 편의상 본 논문에서는 FAMD로 지칭하였다.

1. MissForest와 유사하게 첫 추측을 통해 결측치에 대한 초기값을 설정하여 $\widehat{X}^0$를 얻고, D^0, M^0 를 계산한다.
2. (a) ℓ번째 단계에서, 앞서 논의한대로 $(X^{\ell-1} - M^{\ell-1})(D^{\ell-1})^{-1/2}$의 SVD를 통해 $\widehat{U}^\ell, \widehat{V}^\ell, (\widehat{\Lambda}^\ell)^{1/2}$ 를 얻는다.
   (b) 이 때 첫 번째로 얻은 best low rank S는 유지한 채, $\widehat{X}^\ell_{n\times p} = (\widehat{U}^\ell_{n\times S}\ (\widehat{\Lambda}^\ell_{S\times S})^{1/2}(\widehat{V}^\ell_{p\times S})^t)((D^{\ell-1})_{p\times p})^{1/2} + M^{\ell-1}_{n\times p}$ 를 통해 $\widehat{X}^\ell_{n\times p}$를 계산한다.
   (c) 지시행렬 W와 함께 $X^\ell = W*X+(1-W)*\widehat{X}^\ell$ 을 통해 imputed matrix을 만들고, D^ℓ, M^ℓ도 갱신된다.
3. 수렴조건 $\sum_{ij}(\hat{x_{ij}}^\ell - \hat{x_{ij}}^{\ell-1})^2 < \epsilon$을 만족할 때까지 2.(a), 2.(b), 2.(c)를 반복한다. 여기서 ϵ은 10^{-6} 등을 고려한다.

FAMD의 논의한 내용을 포함한 자세한 설명은 Audigier, Husson, and Josse (2016)와 Josse and Husson (2016)를 참고하면 된다.

2.3 Oversampling

Oversampling이란 소수 class의 데이터를 복제 혹은 새롭게 합성하여 데이터의 균형을 맞추는 기법이다.

2.4에서 소개할 예정인 XGBoost는 scale_pos_weight와 같은 자체적인 불균형 해결을 위한 모수를 보유하기도 한다. 하지만, 대부분의 머신러닝 기법들은 훈련 데이터의 class 중요도를 같게 생각하는 균형 데이터를 가정한다.Nnamoko and Korkontzelos (2020)에서 이에 관련한 연구가 진행되었으며, 계급 불균형에 따른 적합 문제를 방지하기 위해 사용하는 많은 기법이 알려져 있다. 이를 크게 다수 class의 데이터를 줄이는 Under Sampling 기법과 Over sampling 기법으로 나눈다. Oversampling 기법으로는 Random Oversampling, SMOTE 등의 기법이 사용되며, Undersampling 기법으로는 Random Undersampling, TOMEK link를 이용한 undersampling 등이 있다.(Guo, Yin, Dong, Yang, and Zhou, 2008) Undersampling은 훈련 데이터를 감소시켜 적합 과정에 악영향을 줄 가능성이 존재해 표본 수가 undersampling을 하기엔 충분하지 않다고 보고 본 연구에서는 고려하지 않았다. Oversampling은 더 적은 class(minority class)에 속하는 관측치를 증폭시키는 방법으로 이해할 수 있으며 좀 더 자세한 설명은 다음과 같다.

- Random Oversampling

  단순히 적은 class의 관측치를 랜덤하게 골라 증폭시키는 방법이다. 가장 간단한 oversampling 기법으로 소수 class의 데이터를 복제하는 방법이다. 다시 말해, 소수 class의 데이터를 복원 추출하여 훈련 데이터에 추가하는 방법이다. 이에 데이터에 따라 과적합(Overfitting) 문제가 발생할 수 있지만 아무런 가정 없이 적용해 효과를 낼 수 있다는 장점이 있다.

- SMOTE(Synthetic Minority Oversampling Technique)

  SMOTE는 합성된 소수 class의 데이터를 만드는 (creating synthetic examples) oversampling 기법이다. 즉, Random oversampling처럼 원 데이터와 일치하는 데이터가 아닌 합성된 데이터가 훈련 자료에 추가된다. 소수 class 데이터의 K개의 최근접 이웃(K minority class nearest neighbors)들을 활용하는 방식이며 소수 class를 어떤 비율만큼 증폭시킬지에 따라 고려하는 이웃의 수는 달라진다. 이를 포함한 기법의 자세한 설명과 코드 구성법을 Chawla, Bowyer, Hall, and Kegelmeyer (2002)에서 소개하고 있다. 이에 대한 리뷰를 포함한 Brandt and Lanzén (2021)이 제시한 SMOTE의 synthetic examples 생성 과정은 다음과 같다.

  1. 먼저 최근접 이웃의 수 k와 synthetic observations의 수 N를 결정한다. 생성 과정 설명에선 문자 표기를 위해 최근접 이웃의 수는 소문자 k로 나타낸다.
  2. x_i, $i = 1, ..., n_s$을 소수 class에 속한 데이터라 하고, x_i의 집합을 A라 한다.
  3. 유클리디안 거리(Euclidean distance)를 기준으로 A의 모든 원소 중 x_i의 k근접이웃을 찾는다.(일반적으로 $k = 5$) x_i의 k근접이웃의 집합을 S_{ik}라 한다.
  4. S_{ik}에서 N번의 랜덤복원추출을 통해 $x_{ij}, j = 1, ..., N$을 얻는다.
  5. $[0, 1]$ 로부터 임의로 생성한 λ와 주어진 x_{ij}에 대해 synthetic observation $x_k = x_i + \lambda(x_i - x_{ij})$의 관계로부터 생성한다. 이를 모든 x_{ij}에 대해 진행한다.

SMOTE의 데이터 생성에 대한 이해를 돕는 사진을 포함한 자세한 논의를 Brandt and Lanzén (2021)에서 제시하고 있으니 Chawla, Bowyer, Hall, and Kegelmeyer (2002)와 함께 참고하면 된다.

- ADASYN(Adaptive synthetic sampling approach for imbalanced learning)

  ADASYN 역시 최근접 이웃의 개념을 사용한다. 소수 class에 속하는 x_i에 대해 유클리디안 거리에 근거한 K개의 이웃을 찾고, 다음과 같이 r_i를 계산한다. SMOTE와 마찬가지로 Brandt and Lanzén (2021)에 제시된 ADASYN의 알고리즘은 다음과 같은 절차에 의해 진행된다.

  1. 최근접 이웃의 수 k와 synthetic observation을 생성한 후에 원하는 class 균형의 수준(the desired level of class balance after generating the synthetic data) β을 정한다. SMOTE와 같이 최근접 이웃의 수는 k로 표기하고, 완전한 균형 데이터를 희망하면 $\beta = 1$로 설정한다.
  2. n_ℓ을 다수 class에 속한 관측치의 수, n_s을 소수 class에 속한 관측치의 수라 한다. 이 때, $G = (n_\ell - n_s) \times \beta$로 계산한다.
  3. x_i, $i = 1, ..., n_s$을 소수 class에 속한 데이터라 하고, x_i의 집합을 A라 한다.
  4. SMOTE의 synthetic observation 생성과정 3. 과 같은 정의를 한다.
  5. x_i의 k개 이웃 중 다수 class에 속하는 관측치의 수 $\triangle_i$, $i = 1, ..., n_s$를 계산하고, r_i는 $\triangle_i/k$, $i = 1, ..., n_s$로 정의된다.
  6. $\hat{r_i} = r_i / \sum_{i=1}^{n_s} r_i$ 로 r_i를 표준화 해준다. $\sum_{i=1}^{n_s} \hat{r_i} = 1$ 로써, density distribution이다.

7. $g_i = \hat{r}_i \times G$ 를 계산한다. 이는 각 관측치 x_i에 대해 몇 개의 synthetic observation이 생성되어야 하는지에 대한 값이다. g_i가 정수가 아닌 경우에는 버림한 정수값을 이용한다. (Huang, 2015)
8. S_{ik}에서 g_i개를 랜덤 복원 추출해 $x_{ij}, j = 1, ..., g_i$라 한다.
9. $[0, 1]$ 로부터 임의로 생성한 λ와 주어진 x_{ij}에 대해 synthetic observation $x_k = x_i + \lambda(x_i - x_{ij})$의 관계로부터 생성한다.

자세한 설명은 He, Bai, Garcia, and Li (2008)와 Brandt and Lanzén (2021)의 리뷰를 참고하면 된다.

SMOTE와 ADASYN은 유사한 방법이지만 각 소수 class 데이터 당 생성하는 synthetic examples 수에서 차이가 있다. SMOTE의 경우는 미리 정해진 N 값을 이용해 각 소수 class에 대해 데이터를 합성하지만, ADASYN은 ADASYN의 생성과정 6.와 7.에서 처럼 density distribution을 이용해 모든 데이터마다 같은 수의 합성을 하지 않는다.

SMOTE와 ADASYN은 Python의 imblearn 패키지에 구현되어 있으며, SMOTE()와 ADASYN()을 통해 활용하면 된다.

2.4 머신러닝 기법

본 연구에서 활용하는 머신러닝 기법에 대한 설명을 진행한다. 더 자세한 논의는 각 기법에 언급한 참고논문 등을 참고하면 된다.

- Random Forest

  많은 트리를 붓스트랩 표본(bootstrap sample)을 통해 형성하고 소프트 보팅(soft voting)에 의해 최종 모형을 만드는 배깅(Bagging) 기반의 앙상블(Ensemble) 머신러닝 기법으로 Leo Breiman에 의해 제안되었다. (Breiman, 2001)

- AdaBoost

  Weak learners 를 결합해 강력한 분류기를 형성하는 방식으로 잘못 분류한 관측에 해당하는 가중치는 증가시키고 맞게 분류된 관측치에 해당하는 가중치는 감소하는 방식으로 갱신하며 Freund and Schapire에 의해 제안되었다. (Freund and Schapire, 1997)

- Gradient Boosting Machine(GBM)

  전체 앙상블과 관련된 손실 함수의 음의 기울기와 최대 상관 관계가 있는 기본 학습기(base learners)를 새롭게 구성하는 것이 기본적인 아이디어이다.(Natekin and Knoll, 2013) Gradient Descent를 통해 손실함수를 최소화하는 방향으로 가중치를 업데이트하며 학습한다.

  XGBoost는 GBM을 바탕으로 개발되었고, 병렬 연산(parallel computing)을 통해 빠른 학습이 가능해지며 등장했다. (Chen and Guestrin, 2016)

또한, GBM과 XGBoost는 균형 트리 분할(level-wise tree growth)를 사용하지만, LightGBM은 리프 중심 분할(leaf-wise tree growth)을 통해 XGBoost보다도 연산 속도, 메모리 소모 측면에서 이점을 가진다. (Ke, Meng, Finley, Wang, Chen, Ma, Ye, and Liu, 2017) 본 연구에서는 XGBoost와 LightGBM을 모두 활용한다.

2.5 Performance measure

짝 자료(paired data)에 대해, paired data가 형성하는 값의 차이에 대한 평균이 0인지 0보다 큰지를 검정한다고 하자. 예를 들어, paired 되어 있는 A와 B가 있고 $D_i = A_i - B_i$와 D_i의 평균을 μ_d 라 하면, 검정하고자 하는 가설은

$$H_0 : \mu_d = 0$$

$$H_1 : \mu_d > 0$$

로 표현할 수 있다.

본 연구에서는 imputation을 통해 예측력의 향상이 있는지에 관해 관심이 있으므로, 단측검정을 진행한다. 검정 방법은 정규성 가정을 이용하는 Paired t-test와 가정이 필요 없는 permutation paired test, 비모수적인 검정인 Wilcoxon signed-rank test 등을 활용할 수 있다. Paired data가 아닌 경우에는 A의 평균 μ_A 와 μ_B에 대해 다음과 같은 가설을 검정한다.

$$H_0 : \mu_A = \mu_B$$

$$H_1 : \mu_A > \mu_B$$

이때는 일반적인 t-test와 permutation test 그리고 Wilcoxon rank sum test 를 활용할 수 있다.

Chapter 3

사례 연구

3.1 데이터

본 연구는 결측 대치에 따른 예측성능의 향상이 있는지 보는 것에 목적이 있으므로, 결측이 포함된 데이터를 통해 사례 연구를 진행한다. 다시 말해, 원 데이터 선정에서 완전한 데이터를 고려하지 않을 것이다. 또한, 관측치 대비 결측치의 비율을 결측률(Missing rate)이라 할 때, 결측률이 매우 높은 변수에 통계적 대치를 사용하게 된다면 이 대치 값들이 원 데이터의 분포를 반영한다고 보기 힘들며 예측성능의 향상을 기대하기도 어려울 것이다. 50% 이상의 결측률을 보이는 변수에 대한 대치 결과는 의심스러울 것이라는 Scheffer (2002)를 참고하여 높은 결측률의 기준을 50%로 잡았다.

위와 같은 논의에 따라 결측을 포함하되 높은 결측률을 가지는 변수가 너무 많지 않은 데이터로서 본 연구의 데이터를 선정했다. 이에 Kaggle에 공개된 상태인 은행의 연체 여부 예측 데이터(https://www.kaggle.com/zaurbegiev/my-dataset)를 이용한 사례 연구를 진행할 것이다. 본 데이터는 주로 자체적인 데이터 분석 대회에 사용되는 Kaggle 데이터의 특성상 훈련 자료와 평가 자료가 분리되어 있다. 평가 자료는 연체 여부를 알 수 없으므로 머신러닝 적합에 사

용할 수 없어 훈련 자료를 연구에 사용할 원 데이터로 삼고 자체적인 데이터 분할을 한다.

데이터는 100,514개의 관측치와 19개의 변수로 이루어져 있다. 관측치 중 마지막 514개의 모든 값은 결측으로 유효한 관측치가 아니므로 실제 관측치는 100,000개이다. 반응변수 Loan.Status는 연체자면 'Charged off', 연체자가 아닌 대출을 상환한 고객이면 'Fully paid'의 값을 갖는 이진 범주형 변수이다. 데이터는 22,639명의 연체 고객(Charged off)과 77,361명의 상환 고객(Fully paid) 정보를 가지고 있다. 19개의 변수 중 분류(예측)하고자 하는 반응변수 연체 여부(Loan.Status)를 제외한 설명 변수들과 그 의미는 다음과 같다. 또한, 각 변수의 결측률까지 살펴본다.

변수명	종류	변수 설명
Loan.ID	범주형	대출 지원 시 각 대출에 부여되는 ID
Customer.ID	범주형	대출 지원자의 ID
Current.Loan.Amount	연속형	현재의 대출 금액
Term	범주형	단기/장기 대출 여부
Credit.Score	연속형	CB사 등에서 얻은 대출 지원자의 신용 점수
Annual.Income	연속형	지원자의 연 수입
Years.in.current.job	순서형	지원자의 현 직장에서의 근속연수
Home.Ownership	범주형	주거지 분류(자가, 임대 등)
PURPOSE	범주형	대출 지원 목적(사업, 교육 등)
Monthly debt	연속형	매달 상환해야 하는 채무
Years.of.Credit.History	연속형	신용 거래 기록이 시작된 이래 경과한 시간
Months.since.last.delinquent	순서형	최근 체납 이후에 흐른 달(month)
Number.of.Open.Accounts	순서형	거래할 수 있는 계좌의 수
Number.of.Credit.Problems	순서형	자체 선정 신용 문제에 지원자가 몇 개나 해당하는지 표기한 변수
Current.Credit.Balance	연속형	현재의 은행 계좌에 저축된 돈(credit balance)
Maximum.Open.Credit	연속형	이전까지 사전 승인된 Credit의 최댓값.
Bankruptcies	연속형	지원자가 지금까지 겪은 파산의 수.
Tax.Liens	연속형	체납 등을 원인으로 담보 등이 걸린 경우(tax liens)의 수

Table 3.1: 데이터 변수 설명

변수명	종류	결측률(%)
Loan.ID	범주형	0.000
Customer.ID	범주형	0.000
Current.Loan.Amount	연속형	11.484
Term	범주형	0.000
Credit.Score	연속형	19.154
Annual.Income	연속형	19.154
Years.in.current.job	순서형	4.222
Home.Ownership	범주형	0.000
PURPOSE	범주형	0.000
Monthly debt	연속형	0.000
Years.of.Credit.History	연속형	0.000
Months.since.last.delinquent	순서형	53.141
Number.of.Open.Accounts	순서형	0.000
Number.of.Credit.Problems	순서형	0.000
Current.Credit.Balance	연속형	0.000
Maximum.Open.Credit	연속형	0.002
Bankruptcies	연속형	0.204
Tax.Liens	연속형	0.010

Table 3.2: 변수의 결측률

높은 결측률을 보이는 Months.since.last.delinquent 변수가 존재함을 확인할 수 있다. 이는 데이터 소개에서 언급한 높은 결측률을 가지는 변수이고, 이 변수는 분석에서 제외한다. 또한, Loan.ID와 Customer.ID는 개인을 구분하는 용도로, 신용에 관한 정보를 포함하지 않으므로 설명변수로써의 의미가 없기에 역시 제외한다. 종합하면 19개의 변수 중 3개의 변수를 제거해 분석에 사용하는 설명변수 15개와 반응변수 1개로 총 16개의 변수가 분석에 사용된다.

3.2 분석

3.2.1 전처리

가장 선행해서 진행하는 전처리는 의미 없는 데이터와 변수 제거이다. 데이터 소개에서 언급한 100,001번째부터 100,514번까지의 데이터는 모든 변수가 결측인 관측치임에 포함할 이유가 없으므로 모두 제거해 총 데이터를 100,000개로 한다. Loan.ID와 Customer.ID도 앞서 언급한 이유로부터 분석에서는 제외한다.

현재의 대출액을 나타내는 Current.Loan.Amount에는 많은 값이 9,999,999로 대입되어 있다. 이 값이 특정한 의미를 나타낼 가능성도 물론 존재하지만, 구체적인 파악이 불가능하고 제외한 변수의 평균이 312,313.6이라는 점을 통해 결측치로 보는 것이 더 합당하다고 판단 후 결측 처리한다. 현재 직장에서의 근속 연수를 나타내는 Years.in.current.job에도 결측의 의미를 담은 'n/a'라는 값이 상당 부분 포함되어 있어 이 값들은 결측으로 처리한다. 앞서 제시한 표의 결측률은 이러한 처리가 진행된 후에 관측된 값이다.

대출의 목적을 나타내는 Purpose 변수는 총 16개의 범주로 이루어진 범주형 변수이다. 이 중 같은 의미를 갖지만 다른 범주로 포함된 'other'와 'Other' 범주는 Other 하나로 합친다. 'Renewable_energy' 라는 범주는 총 100,000명의 고객 중 10명의 고객의 대출 지원 목적에만 해당한다. 이 범주를 그대로 유지하게 되

면 임의로 훈련 검증 자료를 나누는 과정에서 특정 자료로만 포함이 될 가능성이 존재하여 프로그램의 적합 과정에서 오류를 발생시킬 수 있다. 이러한 문제를 방지하기 위해 이 역시 'Other' 범주로 합친다. 결과적으로 Purpose 변수는 14개의 범주를 가지게 된다.

범주형 변수가 포함된 데이터를 머신러닝을 이용해 적합하기 위해 하는 대표적인 전처리가 one-hot encoding이다. 보유한 데이터 단기/장기 대출을 의미하는 Term 변수를 예로 들자. 두 가지 범주가 존재하므로 one-hot encoding으로 인해 Term_1과 Term_2라는 변수가 생성된다. 각 관측치는 단기 대출자면 Term_1은 0, Term_2는 1의 값을 가지고 장기 대출자면 Term_1은 1, Term_2는 0의 값을 가지게 된다. 고려하는 머신러닝 기법 중 XGBoost의 경우는 범주형 변수를 자체적으로 인식하지 않아 one-hot encoding이 필요하다. 이에 다른 기법들은 필수적이지 않지만, encoding 후의 적합 역시 잘 진행되므로 일치성을 위해 모든 기법에 적용하도록 한다.

3.2.2 진행 과정

사례 연구에 관한 구성을 다음과 같이 구체화한다. 결측 대치를 한 데이터를 통한 훈련 과정이 완전한 데이터만을 추출한 훈련 과정보다 머신러닝 기법에 있어 더 나은 예측 성능을 보이는지 관측하기 위해 다음과 같은 절차로 진행한다. 이는 Lo, Siah, and Wong (2019)와 Lo, Siah, and Wong (2019)의 Supplementary Materials를 참고하였다.

우선, 원 데이터(Original dataset)에서 결측이 없는 완전한 데이터만을 추출한다. 이 데이터를 Lo, Siah, and Wong (2019)와 같이 gold-standard 데이터라 칭한다. 이 완전한 데이터를 7:3의 비율로 훈련 데이터와 평가 데이터로 나눈다. 훈련 데이터에 적절한 결측 가정과 원 데이터에 존재했던 결측률을 이용해 훈련 데이터에 결측을 발생시킨다.(Generate Missingness, Introduce Missingness) 결측에 관련된 가정은 2.1절에서 다뤘다.

이는 보유한 훈련데이터가 결측을 포함하고 있는 실제 상황과 유사하다. 이 단계에서 완전한 데이터만을 추출해 훈련한 경우와 통계적 대치를 한 데이터를 통해 훈련한 경우를 비교해 예측 성능의 차이가 있는지를 확인하는 것이 연구의 목표이다. 이렇게 준비한 완전한 훈련 데이터(Complete-case)와 통계적 대치를 진행한 데이터(Imputed train data)들에 각각 oversampling을 적용한 후 머신러닝 기법을 적합하여 미리 분할해둔 평가 데이터의 연체 여부를 예측하고 성능을 비교하게 된다. 평가 데이터는 class 불균형 해소, 결측 대치 등의 추가적인 전처리를 하지 않음을 다시 한번 상기한다.

결측 대치와 머신러닝 기법의 경우 2.2절과 2.4절에서 언급한 모든 방법을 진행해 Imputed train data와 Complete-case를 비교한다. 각 적합을 1번 반복하는 과정마다 oversampling 역시 매번 적용해줘야 한다. Oversampling은 1장에서 언급하였듯이 적절한 상황에서의 비교를 목적으로 한다. 이에 본 연구에서는 모든 oversampling을 다루기보단 주목적인 Imputation의 실험에 좀 더 집중하기 위해, 주어진 데이터에 대해 가장 좋은 적합 결과를 보이면서도 상대적으로 모수 조정을 간단하게 진행 후 적합이 가능한 Random Forest를 이용해 oversampling

기법을 결정하였다. 즉, gold-standard 데이터에서 훈련/평가 데이터 분할을 진행한 후 얻어진 훈련 데이터를 통해 Random Forest 적합을 하고, 이에 대한 test AUC를 경험적으로 관측한다. 이 결과를 통해 가장 적절한 하나의 기법만 공통으로 적용하려 한다.

시행하고자 하는 사례 연구와 유사하게 원 데이터에서 결측을 제거했던 gold-standard 데이터를 훈련/평가 데이터로 7:3의 비율로 나눈다. 각 oversampling 기법을 적용한 후 Random Forest를 적합해 test AUC를 100번 기록한 후 비교해 시각화한 그래프이다.

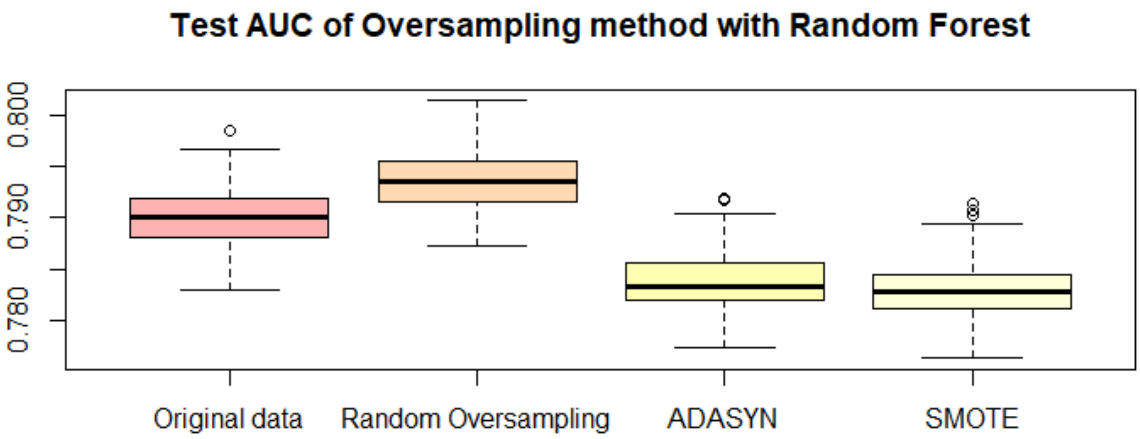

Figure 3.1: Oversampling 기법 비교

결과를 통해 ADASYN과 SMOTE는 오히려 oversampling을 하지 않은 경우보다 AUC가 낮게 형성되며 Random Oversampling만 약간의 효과를 보였음을 알 수 있다. 이는 약 10만 개의 데이터에서 3:1 정도로(77361:22639) 분포하는 계급 간 비율이 머신러닝 적합에 악영향을 줄 정도로 심각하지 않으며, 소수 class 자체의 정보를 활용하는 것이 synthetic examples를 추가해 훈련하는 것보다 적합 과정에 긍정적인 효과를 준다고 해석할 수 있다.

일반적으로 Random Oversampling이 가장 빈번하게 겪는 문제가 과적합(Overfitting) 문제인데, 현재 데이터는 데이터의 수에 비해 변수의 숫자가 많지 않기 때문에 이러한 문제가 발생하지 않은 것으로 보인다. 물론 이 결과를 통해 Random Oversampling이 SMOTE, ADASYN보다 나은 oversamling 기법이라고 할 수는 없으나, 간단한 적용으로의 이점이 있으며 본 연구에서 oversampling의 적용은 적절한 상황에서의 비교가 목적이기에 Random Oversampling을 적용한다.

사례 연구는 훈련/평가 데이터 분할, 통계적 대치, oversampling 등에 따른 변동성이 발생하기에 한 번의 적합으로 결과를 비교하는 것이 아닌 좀 더 로버스트(Robust)한 결과를 위해 100회 반복한다. 각 반복에서 얻어지는 test AUC를 기록하고 2.5절에서 언급한 방법으로 비교한다.

Chapter 4

분석 결과

3장 2절에서 논의한 사례 연구의 결과를 소개한다. 먼저, 통계적 검정을 시행하기 전에 100번의 반복에 관한 결과를 요약한 표이다. Gold-standard는 원데이터, Complete-case는 결측이 없는 행만 추출한 데이터, Mean - missForest는 Imputation을 진행한 데이터이다.

Table 4.1: 각 결측 대응에 대한 XGBoost의 AUC 요약

결측 보정	머신러닝	평균	표준편차	5%	50%	95%
Gold-standard	XGBOOST	0.7397	0.0035	0.7338	0.7397	0.7452
Complete-case	XGBOOST	0.735	0.0034	0.7295	0.735	0.7398
Mean	XGBOOST	0.7373	0.0033	0.7312	0.7373	0.7428
Median	XGBOOST	0.7366	0.0035	0.7309	0.7369	0.7423
KNN	XGBOOST	0.7371	0.0035	0.7307	0.7371	0.7425
FAMD	XGBOOST	0.7373	0.0035	0.7314	0.7376	0.7429
missForest	XGBOOST	0.7346	0.0033	0.7294	0.7347	0.7401

Table 4.2: 각 결측 대응에 대한 Random Forest의 AUC 요약

결측 보정	머신러닝	평균	표준편차	5%	50%	95%
Gold-standard	Random Forest	0.7937	0.0030	0.7883	0.7938	0.7981
Complete-case	Random Forest	0.7587	0.0035	0.7526	0.7587	0.7643
Mean	Random Forest	0.7726	0.0030	0.7684	0.7726	0.7768
Median	Random Forest	0.7748	0.0034	0.7690	0.7748	0.7803
KNN	Random Forest	0.7742	0.0035	0.7689	0.7739	0.7801
FAMD	Random Forest	0.7730	0.0031	0.7679	0.7730	0.7782
missForest	Random Forest	0.7721	0.0029	0.7682	0.7718	0.7771

Table 4.3: 각 결측 대에 대한 LightGBM의 AUC 요약

결측 보정	머신러닝	평균	표준편차	5%	50%	95%
Gold-standard	LightGBM	0.7384	0.0033	0.7327	0.7381	0.7440
Complete-case	LightGBM	0.7347	0.0035	0.7289	0.7349	0.7400
Mean	LightGBM	0.7366	0.0033	0.7307	0.7365	0.7418
Median	LightGBM	0.7368	0.0033	0.7302	0.7364	0.7419
KNN	LightGBM	0.7369	0.0033	0.7316	0.7368	0.7422
FAMD	LightGBM	0.7364	0.0034	0.7312	0.7363	0.7423
missForest	LightGBM	0.7331	0.0032	0.7283	0.7333	0.7384

Table 4.4: 각 결측 대응에 대한 ADABoost의 AUC 요약

결측 보정	머신러닝	평균	표준편차	5%	50%	95%
Gold-standard	ADABoost	0.7333	0.0032	0.7283	0.7329	0.7386
Complete-case	ADABoost	0.7288	0.0034	0.7232	0.7289	0.7337
Mean	ADABoost	0.7307	0.0033	0.7250	0.7304	0.7360
Median	ADABoost	0.7310	0.0035	0.7255	0.7303	0.7373
KNN	ADABoost	0.7321	0.0034	0.7264	0.7322	0.7383
FAMD	ADABoost	0.7172	0.0050	0.7101	0.7170	0.7263
missForest	ADABoost	0.7286	0.0034	0.7232	0.7288	0.7347

표를 통해 대략적인 imputation의 효과 여부를 확인할 수 있고, 2.5절에서 언급한 통계적 검정을 이어서 시행한다. 2.5절에서 언급한 바와 같이 A_i를 Complete-case의 AUC, B_i를 imputation을 진행한 후 훈련해서 얻은, 즉 Mean, Median, KNN, missMDA, missForest에 해당하는 AUC라고 하자. $D_i = B_i - A_i$와 D_i의 평균을 μ_d 라 하면, 검정하고자 하는 가설은

$$H_0 : \mu_d = 0$$

$$H_1 : \mu_d > 0$$

로 표현할 수 있으며, 이에 대한 p-value를 표에 포함했다.

Table 4.5: XGBoost의 complete-case와 imputation의 AUC 비교에 대한 검정 결과

결측 대치	머신러닝	Permutation test	Wilcoxon signed rank test	t-test
Mean	XGBoost	<0.001	<0.001	<0.001
Median	XGBoost	<0.001	<0.001	<0.001
KNN	XGBoost	<0.001	<0.001	<0.001
FAMD	XGBoost	<0.001	<0.001	<0.001
missForest	XGBoost	0.944	0.943	0.949

Table 4.6: Random Forest의 complete-case와 imputation의 AUC 비교에 대한 검정 결과

결측 대치	머신러닝	Permutation test	Wilcoxon signed rank test	t-test
Mean	Random Forest	<0.001	<0.001	<0.001
Median	Random Forest	<0.001	<0.001	<0.001
KNN	Random Forest	<0.001	<0.001	<0.001
FAMD	Random Forest	<0.001	<0.001	<0.001
missForest	Random Forest	<0.001	<0.001	<0.001

Table 4.7: LightGBM의 complete-case와 imputation의 AUC 비교에 대한 검정 결과

결측 대치	머신러닝	Permutation test	Wilcoxon signed rank test	t-test
Mean	LightGBM	<0.001	<0.001	<0.001
Median	LightGBM	<0.001	<0.001	<0.001
KNN	LightGBM	<0.001	<0.001	<0.001
FAMD	LightGBM	<0.001	<0.001	<0.001
missForest	LightGBM	0.99	0.99	0.99

Table 4.8: ADABoost의 complete-case와 imputation의 AUC 비교에 대한 검정 결과

결측 대치	머신러닝	Permutation test	Wilcoxon signed rank test	t-test
Mean	ADABoost	<0.001	<0.001	<0.001
Median	ADABoost	<0.001	<0.001	<0.001
KNN	ADABoost	<0.001	<0.001	<0.001
FAMD	ADABoost	0.99	0.99	0.99
missForest	ADABoost	0.714	0.6855	0.7227

표와 검정의 결과를 보면 Mean, median, KNN imputation의 경우 적합한 모든 머신러닝 기법에 대해 AUC 상승의 원인이 되었다. FAMD의 경우에는 세 가지 기법(XGBoost, Random Forest, LightGBM)에 대해 성능 향상을 일으켰지만, ADABoost의 경우에는 AUC 상승이 일어나지 않고 오히려 감소하는 경향을 보였다. missForest의 경우에는 Random Forest 적합 시에만 AUC가 증가했고, 그 외 머신러닝 기법에 대해서는 AUC 감소를 초래했다.

전반적으로 imputation으로 인한 머신러닝 적합에 효과가 있었음을 확인할 수 있다. 하지만, missForest의 사례처럼 imputation은 때때로 약간의 왜곡이 발생하게 하여 성능의 하락을 초래할 수도 있다. 이에 여러 기법을 비교해보고 적절한 imputation 기법을 활용하는 것 역시 중요하다고 판단할 수 있다.

Chapter 5

결론

결측은 데이터 수집 전반에 걸쳐 필연적으로 발생한다. 이에 통계 패키지의 원활한 사용을 위해 결측을 포함한 데이터를 모두 제거하여 완전한 데이터를 활용하는 방법이 활용되곤 하지만, 이는 정보량의 손실로 이어질 가능성이 크다. 이에 본 논문에서는 이 손실을 최소화하고자 하는 여러 통계적 imputation 기법을 다뤘고, oversampling 기법과 머신러닝 적합을 통해 사례 연구를 진행하였다.

사례 연구의 결과에서 증가한 AUC와 통계적 검정을 통해 결측 대치가 머신러닝 훈련의 성능 향상을 끌어낸 결과를 확인할 수 있었다. 이를 통해 listwise deletion보다는 적절한 imputation을 활용하는 것이 좋다고 판단할 수 있을 것이다. 물론, 모든 경우에 대해 imputation이 성능 향상을 이끌지 않고 왜곡을 일으킨 경우도 존재하였으며, 그 성능의 향상이 미미하다고 해석할 여지도 있다. 하지만, 대치를 통한 최대한의 정보량 보존은 하나의 적절한 대안이 될 것이다. 또한, 본 논문에서 활용한 결측 대치 기법뿐 아니라 그 외의 많은 기법을 통한 다양한 연구 역시 수반될 수 있을 것이다.

Reference

Allison, P. D. (2001). *Missing data*. Sage publications.

Audigier, V., F. Husson, and J. Josse (2016). A principal component method to impute missing values for mixed data. *Advances in Data Analysis and Classification 10*, 5–26.

Brandt, J. and E. Lanzén (2021). A comparative review of smote and adasyn in imbalanced data classification. Technical report, Uppsala University.

Breiman, L. (2001). Random forests. *Machine Learning 45*, 5–32.

Chawla, N. V., K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer (2002). Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research 16*, 321–357.

Chen, T. and C. Guestrin (2016). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 785–794.

Freund, Y. and R. E. Schapire (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences 55*, 119–139.

Guo, X., Y. Yin, C. Dong, G. Yang, and G. Zhou (2008). On the class imbalance problem. In *2008 Fourth International Conference on Natural Computation*, Volume 4, pp. 192–201.

He, H., Y. Bai, E. A. Garcia, and S. Li (2008). Adasyn: Adaptive synthetic sampling approach for imbalanced learning. In *2008 IEEE international joint conference on neural networks (IEEE world congress on computational intelligence)*, pp. 1322–1328. IEEE.

Huang, P. J. (2015). *Classification of Imbalanced Data Using Synthetic Over-Sampling Techniques*. University of California, Los Angeles.

Jaeger, M. (2006). On testing the missing at random assumption. In *European Conference on Machine Learning*, pp. 671–678. Springer.

Josse, J. and F. Husson (2016). missmda: A package for handling missing values in multivariate data analysis. *Journal of Statistical Software 70*, 1–31.

Ke, G., Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu (2017). Lightgbm: A highly efficient gradient boosting decision tree. *Advances in Neural Information Processing Systems 30*, 3146–3154.

Li, C. (2013). Little's test of missing completely at random. *The Stata Journal 13*, 795–809.

Lo, A., K. Siah, and C. Wong (2019). Machine learning with statistical imputation for predicting drug approval. *Harvard Data Science Review 1*.

Misztal, M. (2013). Some remarks on the data imputation using "missforest" method. *Acta Universitatis Lodziensis. Folia Oeconomica 285*.

Mostafa, T., M. Narayanan, B. Pongiglione, B. Dodgeon, A. Goodman, R. J. Silverwood, and G. B. Ploubidis (2021). Missing at random assumption made more plausible: evidence from the 1958 british birth cohort. *Journal of Clinical Epidemiology 136*, 44–54.

Myers, T. A. (2011). Goodbye, listwise deletion: Presenting hot deck imputation as an easy and effective tool for handling missing data. *Communication Methods and Measures 5*, 297–310.

Natekin, A. and A. Knoll (2013). Gradient boosting machines, a tutorial. *Frontiers in Neurorobotics 7*, 21.

Nnamoko, N. and I. Korkontzelos (2020). Efficient treatment of outliers and class imbalance for diabetes prediction. *Artificial Intelligence in Medicine 104*, 101815.

Pedersen, A. B., E. M. Mikkelsen, D. Cronin-Fenton, N. R. Kristensen, T. M. Pham, L. Pedersen, and I. Petersen (2017). Missing data and multiple imputation in clinical epidemiological research. *Clinical epidemiology 9*, 157–166.

Potthoff, R. F., G. E. Tudor, K. S. Pieper, and V. Hasselblad (2006). Can one assess whether missing data are missing at random in medical studies? *Statistical Methods in Medical Research 15*, 213–234.

Rubin, D. B. (1976). Inference and missing data. *Biometrika 63*, 581–592.

Scheffer, J. (2002). Dealing with missing data. *Research Letters in the Information and Mathematical Sciences 3*, 153–160.

Stekhoven, D. J. and P. Bühlmann (2012). Missforest—non-parametric missing value imputation for mixed-type data. *Bioinformatics 28*, 112–118.

van Ginkel, J. R., M. Linting, R. C. Rippe, and A. van der Voort (2020). Rebutting existing misconceptions about multiple imputation as a method for handling missing data. *Journal of Personality Assessment 102*, 297–308.

Zhu, X. (2014). Comparison of four methods for handing missing data in longitudinal data analysis through a simulation study. *Open Journal of Statistics 4*, 933–934.

Abstract

Sung hyun Lim
Department of Statistics
Seoul National University

Listwise deletion is the process of analyzing only complete data after removing all missing data. Although this is a method to deal with missing data that is frequently adopted due to its convenience, it is highly likely to cause information loss and biased results. Therefore, the purpose of this study is to investigate whether the utilization of the imputation methods has an advantage over listwise deletion.

In this paper, we show that statistical imputation is better than listwise deletion by comparing the use of complete data in the machine learning training process with the use of data that is handled missing by imputation. To properly discuss this, we will develop the content, including missing patterns, machine learning techniques, and methods for handling class imbalance, and check the effectiveness of the imputation in the classification problem of predicting delinquency using bank customer data.

Keywords : Missing pattern, Statistical Imputation, Classification using machine learning, Class imbalance
Student Number : 2020 − 20361